

## Re: Application Security

**Source:** <http://www.derkeiler.com/Newsgroups/microsoft.public.dotnet.security/2003-08/0296.html>

---

**From:** Alek Davis (*alek\_DOT\_davis\_AT\_intel\_DOT\_com*)

**Date:** 08/26/03

Date: Mon, 25 Aug 2003 16:48:01 -0700

Nathan,

It would've helped if you defined what you meant by ".NET (C#) clients." Are these Windows Forms applications, ASP.NET applications, Windows services, or something else? Do they run on end users' desktops or on the servers? Who are the users of these applications? Are they end users or administrators responsible for installing and configuring these client applications? Does your application run in a Windows domain (Active Directory) environment or a workgroup? You see, depending on how you answer this question, the solution may be easy, not so easy, difficult, or very difficult.

I assume that your C# clients are not distributed to end users. If they are and they do connect to the database server directly, you should probably rethink your application architecture. It is normally not recommended to allow front-end applications (such as Windows Forms or Web Forms) to connect to database servers directly. The recommended approach is: the front end (GUI) calls the middle tier implementing the business logic and the middle tier makes the database calls. Calls between the GUI and the business layer are normally done using remoting, Web services, or in cases of ASP.NET applications class libraries (since in the latter scenario the GUI generating logic already runs on a remote – i.e. Web – server). In this architecture, you are not concerned about end users being able to get your connection string info as much as if you would otherwise, although you still need to implement at least some protection against internal hackers (or risk of compromised system).

Under certain conditions, the best option would be to use Integrated Windows authentication, so the C# clients would not have to specify SQL credentials when connecting to the database server (and therefore would not have to store them). Unfortunately, in quite a few cases, this is not feasible or efficient approach, so if you need to use SQL authentication, you have a challenge. There are certain options available, but in the end, you will basically have to pick between bad solutions and very bad solutions.

You can review Microsoft recommendations related to this issue at <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnnetsec/html/SecNetch12.asp> (Data Access Security), but all of their suggestions have limitations.

microsoft.public.dotnet.security: Re: Application Security

"Writing Secure Code" by Michael Howard and David LeBlanc also has a chapter dedicated to this topic (it is not available online, though, but I would recommend this book to every developer). In my humble opinion, CipherSafe.NET (<http://www.obviex.com/ciphersafe/>) comes closest to the ideal solution, but depending on the type of your application, it may not work for you.

--

Alek

"Nathan Bullock" <nathan\_kent\_bullock@yahoo.ca> wrote in message news:52f8effc.0308251443.a435262@posting.google.com...

> Hi everyone,

>

> I am trying to figure out how to create a secure application. Here is the situation:

>

> SQL Server database

> .NET (C#) clients

>

> There is a global SQL Server userid and password.

> In the database we store an application specific user id and SHA-1 encoded password.

>

> Currently this SQL Server password is just stored in the registry, it is encrypted in some fashion. This allows the application to connect to SQL then the application checks whether this user is actually allowed to be using the application and what access they have within the application.

>

> Anyways here is the problem: if a user has access to the bytecode of the application then they could always decompile it, determine how we are encrypting the SQL Server password, and login to SQL Server giving them full access to the information in the system. Due to this security problem it doesn't really matter how strongly encrypted there application password is. Who knows what the proper way to handle security in this sort of application.

>

> Hopefully this is explained okay.

>

> Nathan