This post was published to Code Blog at 12:10:00 PM 9/26/2016

Encrypt Secrets in your code

Category Security ; Azure ; VSTS

## Intro, Why Encrypt?

You don't want to check in your code to source control VSTS or deploy an app to azure with your secrets/passwords/connection strings in plain text.

The web.config file contains the configuration options for a web service, some of which could contain sensitive information that you don’t want to be kept in plain text. For example, it is very likely that you’ll connect to a database in your web service, and the connection string you use will be kept in the web.config file. The connection string includes the username and password used in the database server, so if you plan to keep your web service in a 3rd party server or if someone that you don’t want to learn the database credentials needs to work on the source code, you could add an extra layer of security by encrypting this data.

## Tutorial

In this tutorial we are going to encrypt a web.config file in an ASP.NET web application. We will check the encrypted code into VSTS then configure our Azure Web App to use our custom certificate to decrypt the config file that contains our secrets.

This is the code we are encrypting:

web.config

<configuration>

  <appSettings>

    <add key="secret" value="password"/>

  </appSettings>

</configuration>

Basically you create a secrets config file out of source control, then manually add the secrets in the azure portal when you deploy to Azure. You will need to manually share the secrets file to your team locally. See section below: Encrypted One Note - For sharing secret safely amongst your team

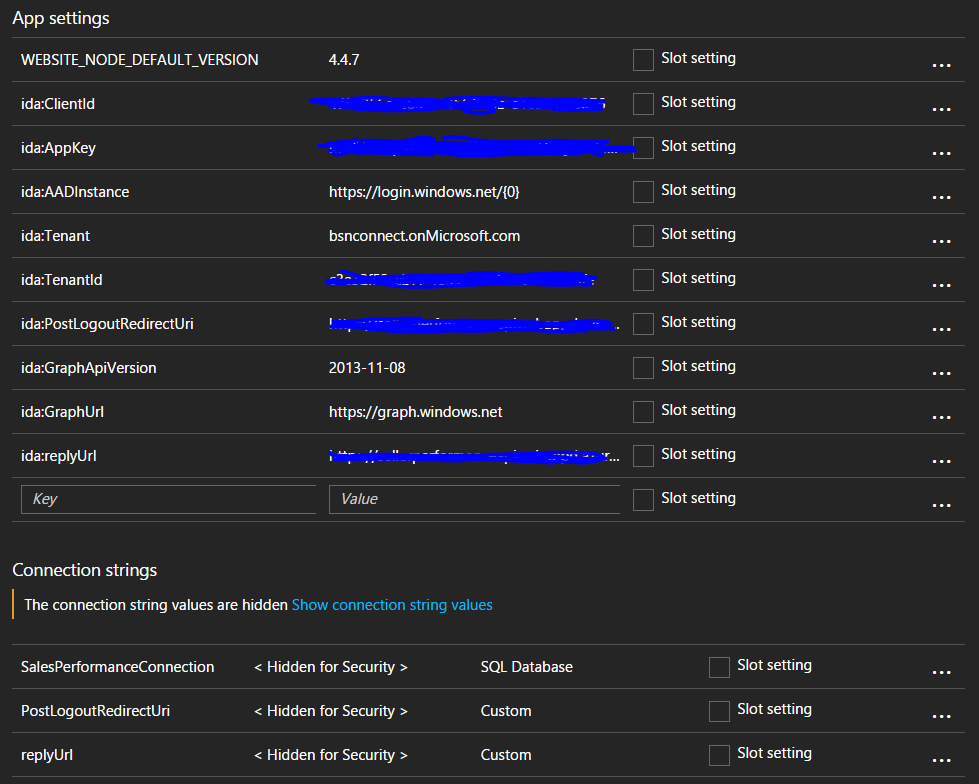
Follow this article on the best practice to secure your secrets, passwords, and other sensative data:

<https://www.asp.net/identity/overview/features-api/best-practices-for-deploying-passwords-and-other-sensitive-data-to-aspnet-and-azure>

<connectionStrings configSource="ConnectionStrings.config">

</connectionStrings>

After following the instructions in the above article, you may find an issue running your Azure site (non-local environment), because it is trying to find the local app settings/ connection string files. If that is the case, in Web.Debug.config add: xdt:Transform="Replace" to your connectionStrings and appSettings tags and ensure you have the same key name and values in your portal application settings. Pictures are shown below.



The approach above is a Microsoft Best Practice. Here is an expert from an artcile on this topic:

**Configuration settings, and connection strings**

It's common practice to store connection strings, authentication credentials, and other sensitive information in configuration files. Unfortunately, these files may be exposed on your website, or checked into a public repository, exposing this information. A simple search on [GitHub](https://github.com/), for example, can uncover countless configuration files with exposed secrets in the public repositories.

The best practice is to keep this information out of your app's configuration files. App Service lets you store configuration information as part of the runtime environment as **app settings** and **connection strings**. The values are exposed to your application at runtime through environment variables for most programming languages. For .NET applications, these values are injected into your .NET configuration at runtime. Apart from these situations, these configuration settings will remain encrypted unless you view or configure them using the [Azure Portal](https://portal.azure.com/) or utilities such as PowerShell or the Azure CLI.

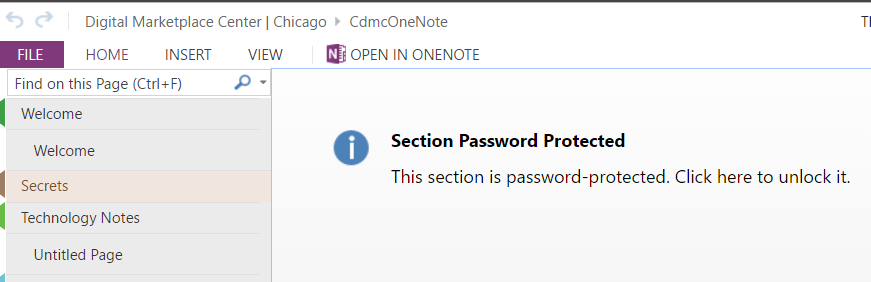
Storing configuration information in App Service makes it possible for the app's administrator to lock down sensitive information for the production apps. Developers can use a separate set of configuration settings for app development and the settings can be automatically superseded by the settings configured in App Service. Not even the developers need to know the secrets configured for the production app. For more information on configuring app settings and connection strings in App Service, see [Configuring web apps](https://docs.microsoft.com/en-us/azure/app-service-web/web-sites-configure).

 - <https://docs.microsoft.com/en-us/azure/app-service-web/web-sites-security>

## Encrypted One Note - For sharing secrets safely amongst your team

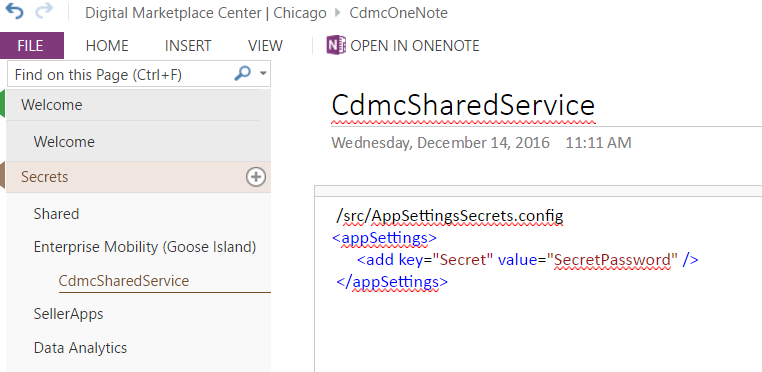
I recommend saving app secrets you need to share with your team in our OneNote on sharepoint here: [go/cdmconenote](http://go.dow.com/cdmconenote). It's also recommended to store secrets in SEAT ( it for some reason remains the "MET"...and i use that term loosly... although the tool has many security flaws).  The OneNote method is nice because you can copy your entire app.config or connection string files. It's also very secure because the secrets notebook is password protected and encrypted on a server. (obtain the password from your lead developer or the cloud security guys)

Here is a view of the locked notebook:



I recommend structuring the Secrets workbook like this:  TeamName > ProjectName > Secrets and Configs for that app

Here is a view of the notebook after it is unlocked:



## References

<https://azure.microsoft.com/en-us/blog/using-certificates-in-azure-websites-applications/>

<https://eren.ws/2014/02/04/encrypting-the-web-config-file-of-an-azure-cloud-service/>