Best practices for private config data and connection strings in configuration in ASP.NET and Azure

January 6, '16 Comments [48] Posted in ASP.NET[1]
[2]

[3]

A reader emailed asking how to avoid accidentally checking in passwords and other sensitive data into GitHub or source control in general. I think it's fair to say that we've all done this once or twice - it's a rite of passage for developers old and new.

The simplest way to avoid checking in passwords and/or connection strings into source control is to (no joke) **keep passwords and connection strings out of your source.**

Sounds condescending or funny, but it's not, it's true. You can't check in what doesn't exist on disk.

That said, sometimes you just need to mark a file as "ignored," meaning it's not under source control. For some systems that involves externalizing configuration values that may be in shared config files with a bunch of non-sensitive config data.

ASP.NET 4.6 secrets and connection strings

Just to be clear, how "secret" something is is up to you. If it's truly cryptographically secret or something like a private key, you should be looking at data protection systems or a Key Vault like Azure Key Vault[4]. Here we are talking about medium business impact web apps with API keys for 3rd party web APIs and connection strings that can live in memory for short periods. Be smart.

ASP.NET 4.6 has web.config XML files like this with name/value pairs.

```
<appSettings>

<add key="name" value="someValue" />

<add key="name" value="someSECRETValue" />

</appSettings>
```

We don't want secrets in there! Instead, move them out like this:

```
<appSettings file="Web.SECRETS.config">

<add key="name" value="someValue" />

</appSettings>
```

Then you just put another appSettings section in that web.secrets.config file and it gets merged at runtime.

> *NOTE: It's worth pointing out that the AppSettings technique also works for Console apps with an app.config.*

Finally, be sure to add Web.secrets.config (or, even better, make it *.secrets and use a unique extension to identify your sensitive config.

This externalizing of config also works with the <connectionStrings> section, except you use the configSource attribute like this:

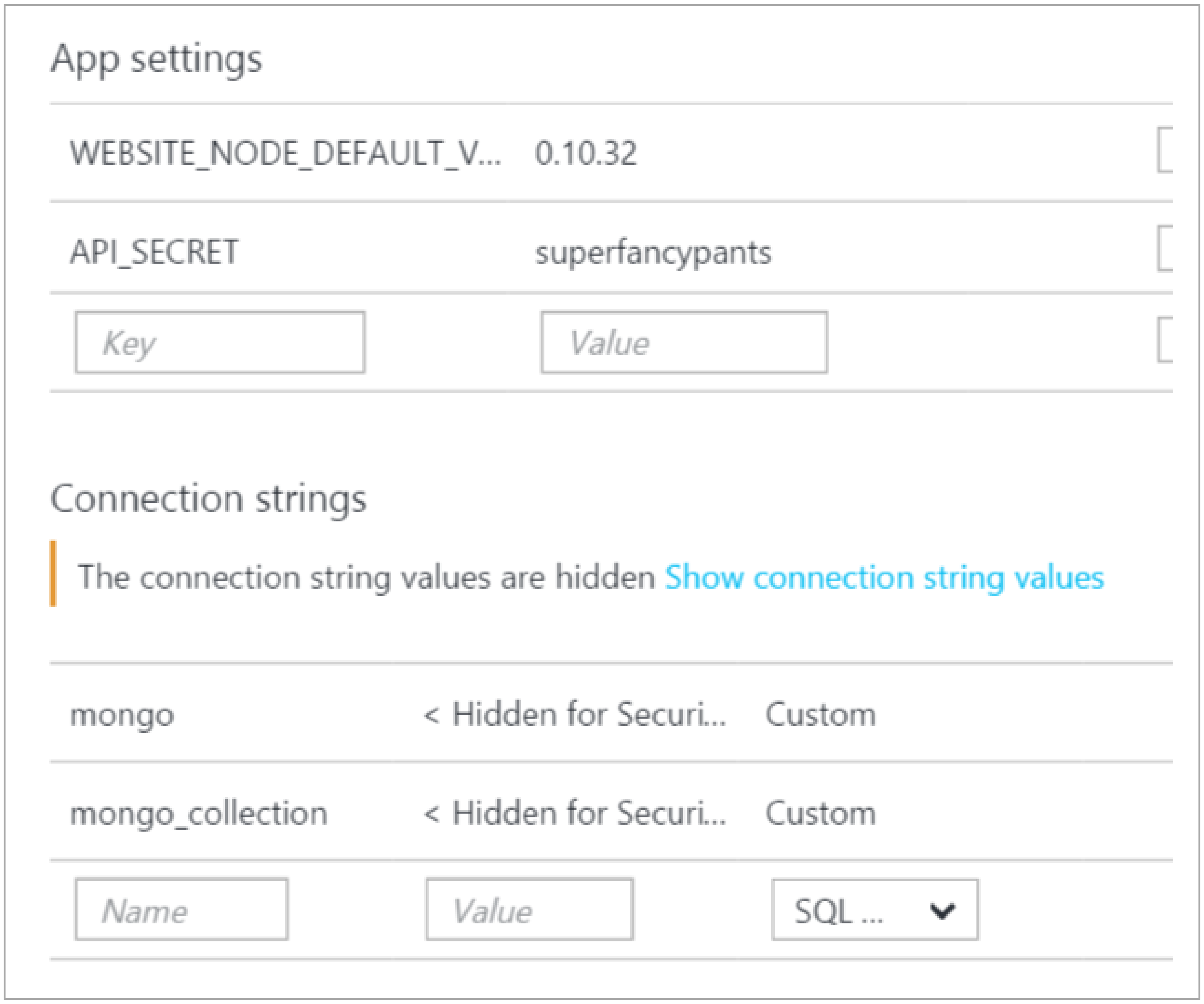
```
<connectionStrings configSource="secretConnectionStrings.config">

</connectionStrings>
```

Connection Strings/App Secrets in Azure

When you're deploying a web app to Azure (as often these apps are deployed from source/GitHub, etc) you should NEVER put your connection strings or appSettings in web.config or hard code them.

Instead, always use the Application Settings configuration section of Web Apps in Azure.



App settings

WEBSITE_NODE_DEFAULT_V...	0.10.32
API_SECRET	superfancypants
Key	Value

Connection strings

The connection string values are hidden Show connection string values

mongo	< Hidden for Securi...	Custom
mongo_collection	< Hidden for Securi...	Custom
Name	Value	SQL ...

These collection strings and name value pairs will automatically be made available *transparently* to your website so you don't need to change any ASP.NET code. Considered them to have more narrow scope than what's in web.config, and the system will merge the set automatically.

Additionally they are made available as Environment Variables, so you can Environment.GetEnvironmentVariable("APPSETTING_yourkey") as well. This works in any web framework, not just ASP.NET, so in PHP you just getenv('APPSETTING_yourkey") as you like.

The full list[5] of database connection string types and the prepended string used for environment variables is below:

- If you select "Sql Databases", the prepended string is "SQLAZURECONNSTR_"
- If you select "SQL Server" the prepended string is "SQLCONNSTR_"
- If you select "MySQL" the prepended string is "MYSQLCONNSTR_"
- If you select "Custom" the prepended string is "CUSTOMCONNSTR_"

ASP.NET 5

ASP.NET 5 has the concept of User Secrets or User-Level Secrets where the key/value pair *does* exist in a file BUT that file isn't in your project folder, it's stored in your OS user profile folder. That way there's no chance it'll get checked into source control. There's a secret manager[6] (it's all beta so expect it to change) where you can set name/value pairs.

ASP.NET also has very flexible scoping rules in code. You can have an appSettings, then an environment-specific (dev, test, staging, prod) appSettings, then User Secrets, and *then* environment variables. All of this is done via code configuration and is, as I mentioned, deeply flexible. If you don't like it, you can change it.

```
var builder = new ConfigurationBuilder()

.AddJsonFile("appsettings.json")

.AddJsonFile($"appsettings.{env.EnvironmentName}.json", optional: true);

if (env.IsDevelopment())

{

// For more details on using the user secret store see http://go.microsoft.com/fwlink/?
LinkID=532709
builder.AddUserSecrets();

}

builder.AddEnvironmentVariables();

Configuration = builder.Build();
```

So, in conclusion:

- Don't put private stuff in code.
  - Seems obvious, but[7]...

- Avoid putting private stuff in *common* config files
  - Externalize them AND ignore the externalized file so they don't get checked in

- Consider using Environment Variables or User-level config options.
  - Keep sensitive config out of your project folder at development time

I'm sure I missed something. What are YOUR tips, Dear Reader?

Resources

- Best practices for deploying passwords and other sensitive data to ASP.NET and Azure App Service[8]

Image Copyright Shea Parikh - used under license from http://getcolorstock.com[9]

Sponsor: Big thanks to Infragistics for sponsoring the blog this week! Responsive web design on any browser, any platform and any device with Infragistics jQuery/HTML5 Controls[10]. Get super-charged performance with the world’s fastest HTML5 Grid - Download for free now[11]!

About Scott

Scott Hanselman is a former professor, former Chief Architect in finance, now speaker, consultant, father, diabetic, and Microsoft employee. He is a failed stand-up comic, a cornrower, and a book author.

About[12] Newsletter[13]

Links

1. https://www.hanselman.com/blog/CategoryView.aspx?category=ASP.NET
2. http://getcolorstock.com/
3. http://getcolorstock.com/
4. https://azure.microsoft.com/en-us/services/key-vault/?WT.mc_id=-blog-scottha
5. https://azure.microsoft.com/en-us/blog/windows-azure-web-sites-how-application-strings-and-connection-strings-work/?WT.mc_id=-blog-scottha
6. https://docs.asp.net/en/latest/security/app-secrets.html
7. http://www.internetnews.com/blog/skerner/github-search-exposes-passwords.html
8. http://www.asp.net/identity/overview/features-api/best-practices-for-deploying-passwords-and-other-sensitive-data-to-aspnet-and-azure
9. http://getcolorstock.com/
10. http://bit.ly/1JWnScH
11. http://bit.ly/1JWnScH
12. http://hanselman.com/about
13. http://www.hanselman.com/newsletter