Safe storage of app secrets in development in ASP.NET Core

12/05/2019 • 10 minutes to read • +6

In this article

By Rick Anderson, Daniel Roth, and Scott Addie

View or download sample code (how to download)

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Version

This document explains techniques for storing and retrieving sensitive data during development of an ASP.NET Core app on a development machine. Never store passwords or other sensitive data in source code. Production secrets shouldn't be used for development or test. Secrets shouldn't be deployed with the app. Instead, secrets should be made available in the production environment through a controlled means like environment variables, Azure Key Vault, etc. You can store and protect Azure test and production secrets with the Azure Key Vault configuration provider.

Environment variables

Environment variables are used to avoid storage of app secrets in code or in local configuration files. Environment variables override configuration values for all previously specified configuration sources.

Consider an ASP.NET Core web app in which **Individual User Accounts** security is enabled. A default database connection string is included in the project's *appsettings.json* file with the key `DefaultConnection`. The default connection string is for LocalDB, which runs in user mode and doesn't require a password. During app deployment, the `DefaultConnection` key value can be overridden with an environment variable's value. The environment variable may store the complete connection string with sensitive credentials.

⚠ **Warning**

Environment variables are generally stored in plain, unencrypted text. If the machine or process is compromised, environment variables can be accessed by untrusted parties. Additional measures to prevent disclosure of user secrets may be required.

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

The `:` separator doesn't work with environment variable hierarchical keys on all platforms. `__`, the double underscore, is:

- Supported by all platforms. For example, the `:` separator is not supported by Bash, but `__` is.
- Automatically replaced by a `:`

Version

Secret Manager

The Secret Manager tool stores sensitive data during the development of an ASP.NET Core project. In this context, a piece of sensitive data is an app secret. App secrets are stored in a separate location from the project tree. The app secrets are associated with a specific project or shared across several projects. The app secrets aren't checked into source control.

⚠ **Warning**

The Secret Manager tool doesn't encrypt the stored secrets and shouldn't be treated as a trusted store. It's for development purposes only. The keys and values are stored in a JSON configuration file in the user profile directory.

How the Secret Manager tool works

The Secret Manager tool abstracts away the implementation details, such as where and how the values are stored. You can use the tool without knowing these implementation details. The

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Version

values are stored in a JSON configuration file in a system-protected user profile folder on the local machine:

Windows | Linux / macOS

File system path:

`%APPDATA%\Microsoft\UserSecrets\<user_secrets_id>\secrets.json`

In the preceding file paths, replace `<user_secrets_id>` with the `UserSecretsId` value specified in the *.csproj* file.

Don't write code that depends on the location or format of data saved with the Secret Manager tool. These implementation details may change. For example, the secret values aren't encrypted, but could be in the future.

Enable secret storage

The Secret Manager tool operates on project-specific configuration settings stored in your user profile.

The Secret Manager tool includes an `init` command in .NET Core SDK 3.0.100 or later. To use user secrets, run the following command in the project directory:

.NET Core CLI   Copy

```
dotnet user-secrets init
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API   PDFCROWD

The preceding command adds a `UserSecretsId` element within a `PropertyGroup` of the *.csproj* file. By default, the inner text of `UserSecretsId` is a GUID. The inner text is arbitrary, but is unique to the project.

Version

XML

```xml
<PropertyGroup>
  <TargetFramework>netcoreapp2.1</TargetFramework>
  <UserSecretsId>79a3edd0-2092-40a2-a04d-dcb46d5ca9ed</UserSecretsId>
</PropertyGroup>
```

Tip

In Visual Studio, right-click the project in Solution Explorer, and select **Manage User Secrets** from the context menu. This gesture adds a `UserSecretsId` element, populated with a GUID, to the *.csproj* file.

Set a secret

Define an app secret consisting of a key and its value. The secret is associated with the project's `UserSecretsId` value. For example, run the following command from the directory in which the *.csproj* file exists:

.NET Core CLI

```
dotnet user-secrets set "Movies:ServiceApiKey" "12345"
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

In the preceding example, the colon denotes that `Movies` is an object literal with a `ServiceApiKey` property.

The Secret Manager tool can be used from other directories too. Use the `--project` option to supply the file system path at which the *.csproj* file exists. For example:

Version

.NET Core CLI

Copy

```
dotnet user-secrets set "Movies:ServiceApiKey" "12345" --project "C:\apps\Web
```

JSON structure flattening in Visual Studio

Visual Studio's **Manage User Secrets** gesture opens a *secrets.json* file in the text editor. Replace the contents of *secrets.json* with the key-value pairs to be stored. For example:

JSON

Copy

```json
{
  "Movies": {
    "ConnectionString": "Server=(localdb)\\mssqllocaldb;Database=Movie-1;Trus
    "ServiceApiKey": "12345"
  }
}
```

The JSON structure is flattened after modifications via `dotnet user-secrets remove` or `dotnet user-secrets set`. For example, running `dotnet user-secrets remove`

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

`"Movies:ConnectionString"` collapses the `Movies` object literal. The modified file resembles the following:

Version

JSON

```json
{
  "Movies:ServiceApiKey": "12345"
}
```

Set multiple secrets

A batch of secrets can be set by piping JSON to the `set` command. In the following example, the *input.json* file's contents are piped to the `set` command.

Windows | Linux / macOS

Open a command shell, and execute the following command:

.NET Core CLI

```
type .\input.json | dotnet user-secrets set
```

Access a secret

The [ASP.NET Core Configuration API](#) provides access to Secret Manager secrets.

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Version

In ASP.NET Core 2.0 or later, the user secrets configuration source is automatically added in development mode when the project calls CreateDefaultBuilder to initialize a new instance of the host with preconfigured defaults. `CreateDefaultBuilder` calls AddUserSecrets when the EnvironmentName is Development:

C# Copy

```
public static IHostBuilder CreateHostBuilder(string[] args) =>
    Host.CreateDefaultBuilder(args)
        .ConfigureWebHostDefaults(webBuilder =>
        {
            webBuilder.UseStartup<Startup>();
        });
```

When `CreateDefaultBuilder` isn't called, add the user secrets configuration source explicitly by calling AddUserSecrets in the `Startup` constructor. Call `AddUserSecrets` only when the app runs in the Development environment, as shown in the following example:

C# Copy

```
public Startup(IWebHostEnvironment env)
{
    var builder = new ConfigurationBuilder()
        .SetBasePath(env.ContentRootPath)
        .AddJsonFile("appsettings.json",
                     optional: false,
                     reloadOnChange: true)
        .AddEnvironmentVariables();

    if (env.IsDevelopment())
    {
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
        builder.AddUserSecrets<Startup>();
    }

    Configuration = builder.Build();
}
```

Version

User secrets can be retrieved via the `Configuration` API:

C#

Copy

```
public class Startup
{
    private string _moviesApiKey = null;

    public Startup(IConfiguration configuration)
    {
        Configuration = configuration;
    }

    public IConfiguration Configuration { get; }

    public void ConfigureServices(IServiceCollection services)
    {
        _moviesApiKey = Configuration["Movies:ServiceApiKey"];
    }

    public void Configure(IApplicationBuilder app)
    {
        app.Run(async (context) =>
        {
            var result = string.IsNullOrEmpty(_moviesApiKey) ? "Null" : "Not
            await context.Response.WriteAsync($"Secret is {result}");
        });
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
    }
}
```

Version

Map secrets to a POCO

Mapping an entire object literal to a POCO (a simple .NET class with properties) is useful for aggregating related properties.

Assume the app's *secrets.json* file contains the following two secrets:

JSON

Copy

```json
{
  "Movies:ConnectionString": "Server=(localdb)\\mssqllocaldb;Database=Movie-1
  "Movies:ServiceApiKey": "12345"
}
```

To map the preceding secrets to a POCO, use the `Configuration` API's object graph binding feature. The following code binds to a custom `MovieSettings` POCO and accesses the `ServiceApiKey` property value:

C#

Copy

```csharp
var moviesConfig = Configuration.GetSection("Movies")
                                .Get<MovieSettings>();
_moviesApiKey = moviesConfig.ServiceApiKey;
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

The `Movies:ConnectionString` and `Movies:ServiceApiKey` secrets are mapped to the respective properties in `MovieSettings`:

C#

Copy

```csharp
public class MovieSettings
{
    public string ConnectionString { get; set; }

    public string ServiceApiKey { get; set; }
}
```

Version

String replacement with secrets

Storing passwords in plain text is insecure. For example, a database connection string stored in *appsettings.json* may include a password for the specified user:

JSON

Copy

```json
{
  "ConnectionStrings": {
    "Movies": "Server=(localdb)\\mssqllocaldb;Database=Movie-1;User Id=johndo
  }
}
```

A more secure approach is to store the password as a secret. For example:

.NET Core CLI

Copy

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
dotnet user-secrets set "DbPassword" "pass123"
```

Remove the `Password` key-value pair from the connection string in *appsettings.json*. For example:

Version

JSON Copy

```json
{
  "ConnectionStrings": {
    "Movies": "Server=(localdb)\\mssqllocaldb;Database=Movie-1;User Id=johndo
  }
}
```

The secret's value can be set on a SqlConnectionStringBuilder object's Password property to complete the connection string:

C# Copy

```csharp
public class Startup
{
    private string _connection = null;

    public Startup(IConfiguration configuration)
    {
        Configuration = configuration;
    }

    public IConfiguration Configuration { get; }

    public void ConfigureServices(IServiceCollection services)
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Version

```
    {
        var builder = new SqlConnectionStringBuilder(
            Configuration.GetConnectionString("Movies"));
        builder.Password = Configuration["DbPassword"];
        _connection = builder.ConnectionString;
    }

    public void Configure(IApplicationBuilder app)
    {
        app.Run(async (context) =>
        {
            await context.Response.WriteAsync($"DB Connection: {_connection}"
        });
    }
}
```

List the secrets

Assume the app's *secrets.json* file contains the following two secrets:

JSON Copy

```json
{
  "Movies:ConnectionString": "Server=(localdb)\\mssqllocaldb;Database=Movie-1
  "Movies:ServiceApiKey": "12345"
}
```

Run the following command from the directory in which the *.csproj* file exists:

.NET Core CLI Copy

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
dotnet user-secrets list
```

The following output appears:

console

```console
Movies:ConnectionString = Server=(localdb)\mssqllocaldb;Database=Movie-1;Trus
Movies:ServiceApiKey = 12345
```

Version

In the preceding example, a colon in the key names denotes the object hierarchy within *secrets.json*.

Remove a single secret

Assume the app's *secrets.json* file contains the following two secrets:

JSON

```json
{
  "Movies:ConnectionString": "Server=(localdb)\\mssqllocaldb;Database=Movie-1
  "Movies:ServiceApiKey": "12345"
}
```

Run the following command from the directory in which the *.csproj* file exists:

.NET Core CLI

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
dotnet user-secrets remove "Movies:ConnectionString"
```

The app's *secrets.json* file was modified to remove the key-value pair associated with the `MoviesConnectionString` key:

Version

JSON

```json
{
  "Movies": {
    "ServiceApiKey": "12345"
  }
}
```

Running `dotnet user-secrets list` displays the following message:

console

```console
Movies:ServiceApiKey = 12345
```

Remove all secrets

Assume the app's *secrets.json* file contains the following two secrets:

JSON

```json
{
  "Movies:ConnectionString": "Server=(localdb)\\mssqllocaldb;Database=Movie-1
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
  "Movies:ServiceApiKey": "12345"
}
```

Version

Run the following command from the directory in which the *.csproj* file exists:

.NET Core CLI

```
dotnet user-secrets clear
```

All user secrets for the app have been deleted from the *secrets.json* file:

JSON

```
{}
```

Running `dotnet user-secrets list` displays the following message:

console

```
No secrets configured for this application.
```

Additional resources

- See this issue for information on accessing Secret Manager from IIS.
- Configuration in ASP.NET Core
- Azure Key Vault Configuration Provider in ASP.NET Core

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Is this page helpful?

Yes No

Feedback

Version

Send feedback about

This product This page

You may also leave feedback directly on **GitHub**.

Loading feedback...

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD