APM .NET: Integração com Login Cidadão

Posted on **14/04/2020** by **Aldo Carlos**

Índice

A partir do código fonte da APM .NET (versão 3 e 4) foram criados alguns exemplos de como adaptar a autenticação das aplicações para funcionar com o Login Cidadão.

Como os códigos originais da APM se baseiam na autenticação com SOE, foram incluídos apenas um conjunto mínimo de modificações necessárias para demonstrar o funcionamento da parte de autenticação. Portanto, algumas funções presentes no código como verificar permissão e acesso ao SOE WS, por exemplo, não foram removidas e não irão funcionar e/ou fazer sentido em função do usuário autenticado não ser um usuário do SOE.

Segue abaixo alguns exemplos conforme o tipo de aplicação (API e MVC), cenários e verão do Framework .NET.

Exemplo 1 – Aplicação API – APM 4

Este exemplo acrescenta a autenticação com o *Access Token* gerado pelo Login Cidadão mantendo a autenticação com o SOE Auth existente. Sendo assim, a API possui 2 *"AuthenticationSchemes "* do .NET podendo aceitar aceita receber 2 tipos de Access Tokens diferentes.

Este cenário é útil para quando se deseja adicionar alguma funcionalidade nova na API que será "autenticada" pelo Login Cidadão. Para aceitar essa 2ª autenticação, deve ser adicionado na *Controller* desta funcionalidade o seguinte atributo:

```
1 [Authorize(AuthenticationSchemes = LoginCidadaoOptions.DefaultScheme)]
```

Este exemplo usa o padrão ***implicit flow*** do OAuth2.

Para acessar o código com o conjunto completo de ajustes clique aqui.

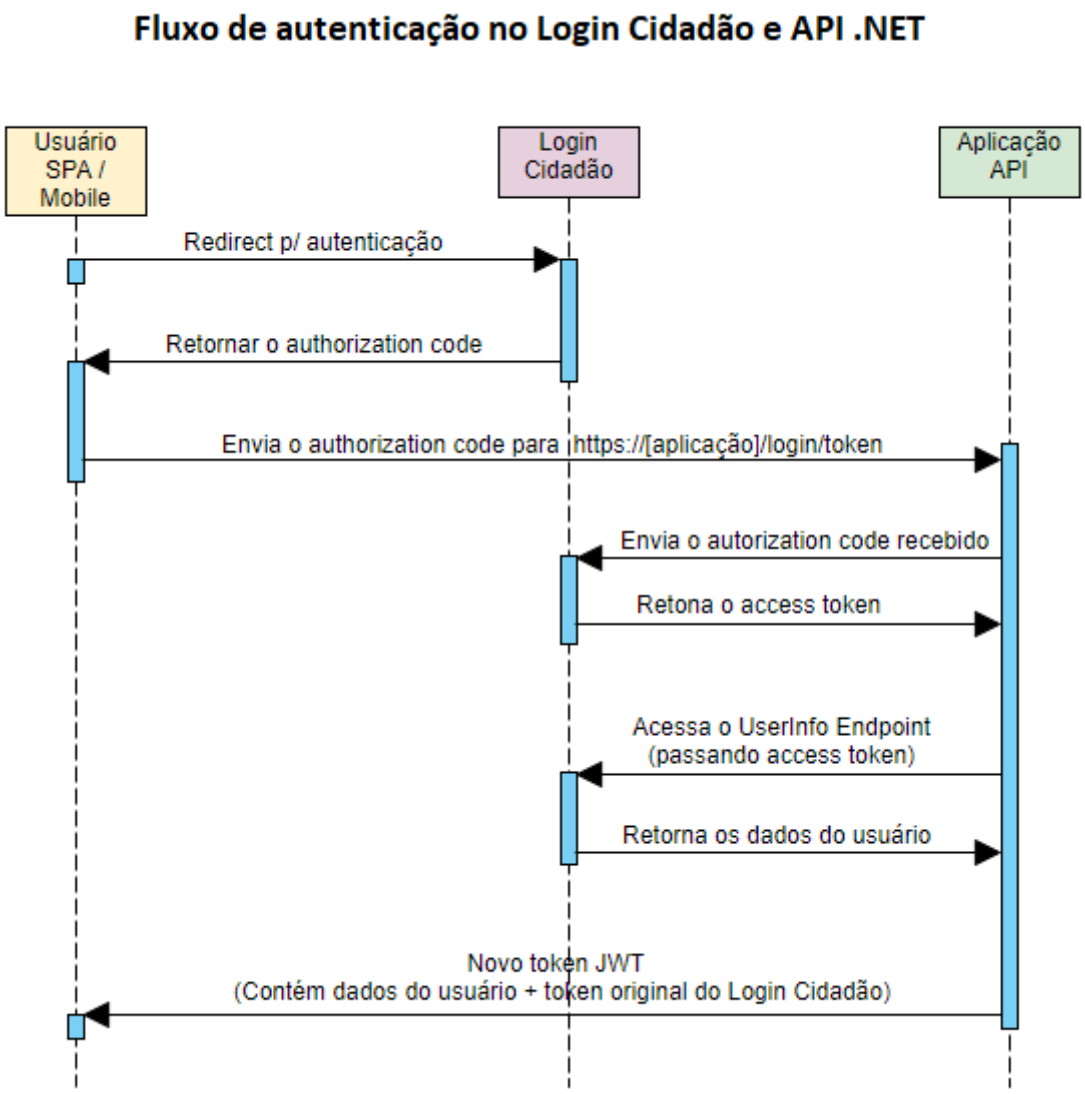
Exemplo 2 – Aplicação API – APM 4 (versão 2)

Este exemplo configura uma aplicação API para autenticar com o Login Cidadão.

Diferente do Exemplo 1, este exemplo cria um *Access Token* do tipo JWT a partir do *Access Token* original retornado pelo Login Cidadão (que não é JWT).

Esta implementação é útil para cenários onde será previsto um volume grande acessos, já que a validação deste tipo de token é mais performática e a aplicação tente a consumir menos memória (não necessita manter cache de tokens como Exemplo 1).

Uma desvantagem desta implementação em relação ao Exemplo 1 é que processo de autenticação é um pouco mais complexo, estendendo o padrão OAuth2 (usa o ***authorization code flow***). Segue abaixo um diagrama que resume como ficou o fluxo de autenticação:



Para acessar o código com o conjunto completo de ajustes clique aqui.

Exemplo 3 – Aplicação MVC – APM 4

Neste exemplo foi acrescentada a autenticação com o Login Cidadão em conjunto com a autenticação do SOE Auth existente. São 2 autenticações do tipo OpenId Connect.

Na tela de logon existente foi acrescentado um botão para o novo logon:

Para diferenciar o tipo de autenticação na autorização das páginas (ou *Controllers*) foi utilizado o recurso de *Policies* do .Net Core. No trecho de código abaixo, do arquivo *Startup.cs,* temos a seguinte configuração:

```
141     // Configura 2 'policies' para autorizar de acordo com o tipo de autenticação (Soe ou Login Cidadão)
142     services.AddAuthorization(options =>
143     {
144         options.AddPolicy("Soe", policy =>
145             policy.AddAuthenticationSchemes(OpenIdConnectDefaults.AuthenticationScheme).RequireAuthenticatedUser());
146
147         options.AddPolicy("LoginCidadao", policy =>
148             policy.AddAuthenticationSchemes("LoginCidadao").RequireAuthenticatedUser());
149     });
```

Para aplicar as restrições das *Policies* nas *Controllers* basta informar a propriedade no atributo *Authorize*, por exemplo:

```
[Authorize(Policy = "LoginCidadao")]
1 reference
public class ExemploController : Controller
{
```

No arquivo *Startup.cs*, também é possível informar a propriedade de *Policy* nas regras (ou *Conventions*) que definem as autorizações para as páginas Razor:

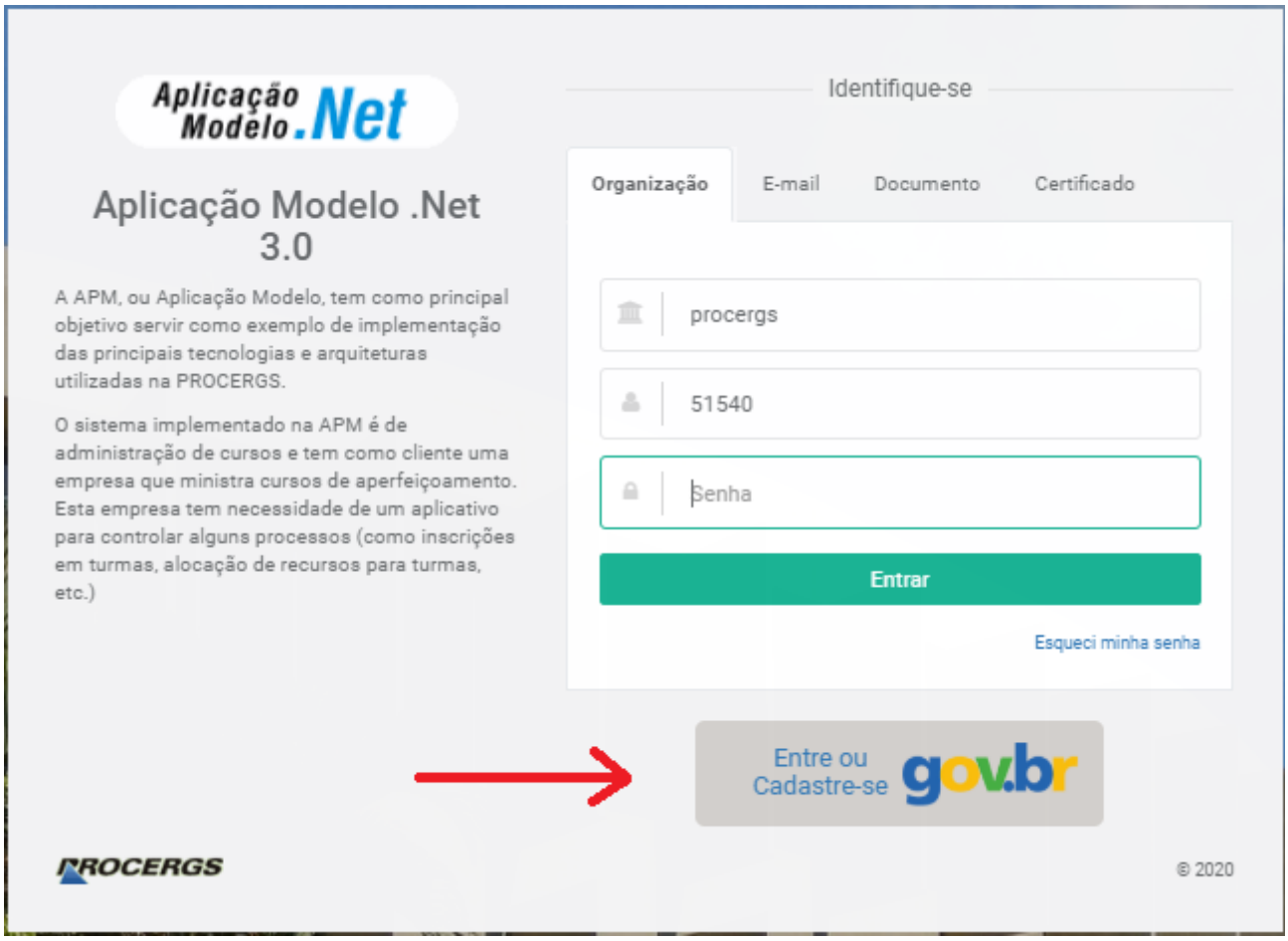
```
130     .AddRazorPagesOptions(options =>
131     {
132         options.Conventions
133             .AuthorizeFolder("/")
134             .AuthorizePage("/Exemplos/SoeWsExemplo", "Soe") // Permite somente a policy 'Soe'
135             .AuthorizePage("/Exemplos/TesteLoginCidadao", "LoginCidadao") // Permite somente a policy 'LoginCidadao'
136             .AllowAnonymousToPage("/Error")
137             .AllowAnonymousToPage("/Index");
138     });
```

Uma vantagem do uso das ***Policies*** do .Net Core, é quando o usuário não está autenticado e tenta acessar uma página, ele será automaticamente redirecionado para o provedor de autenticação correto para aquela página (Soe Auth ou Login Cidadão).

Para acessar o código com o conjunto completo de ajustes clicar aqui.

Exemplo 4 – Aplicação MVC – APM 3

Este exemplo acrescenta a possibilidade de autenticação com o Login Cidadão em uma aplicação MVC (no Arq.Net 3). Na tela de logon existente foi acrescentado um botão para o novo logon:



Para autorizar as páginas (ou Controllers) da aplicação de acordo com o tipo de logon, foi utilizado a propriedade “Roles” das Clains de autenticação (com os valores “LoginCidadao” e “Soe”):

```
namespace Procergs.Apm.Mvc.Controllers
{
    [Authorize(Roles = "LoginCidadao")]
    0 references
    public class ExemploController : Controller
    {
```

```
[Authorize(Roles = "Soe")]
0 references
public ActionResult About()...
```

Obs.: Usando o atributo [Authorize] sem a propriedade “Roles” informada, permite o acesso das 2 autenticações.

Para a montagem dos Menus de acordo com a autenticação, também foi usado o recurso das Roles:

```
@if (HttpContext.Current.User.IsInRole("Soe"))
{
    ...Itens do menu para usuários do SOE
}
else if(HttpContext.Current.User.IsInRole("LoginCidadao"))
{
    ...Itens do menu para usuários do Login Cidadão
}
```

Para a codificação da aplicação foi necessário instalar os seguintes pacotes Nuget (nestas versões):

- IdentityModel (versão 4.2.0)
- Microsoft.IdentityModel.Protocols.OpenIdConnect (versão 6.5.0)
- Microsoft.Owin.Security.OpenIdConnect (versão 4.1.0)

O “ponto de partida” para acompanhar os ajustes feitos pode ser ver o código acrescentado no arquivo *Startup.Auth.cs* (ver este link aqui).

O conjunto com todos ajustes adicionados pode ser obtido a partir deste commit.

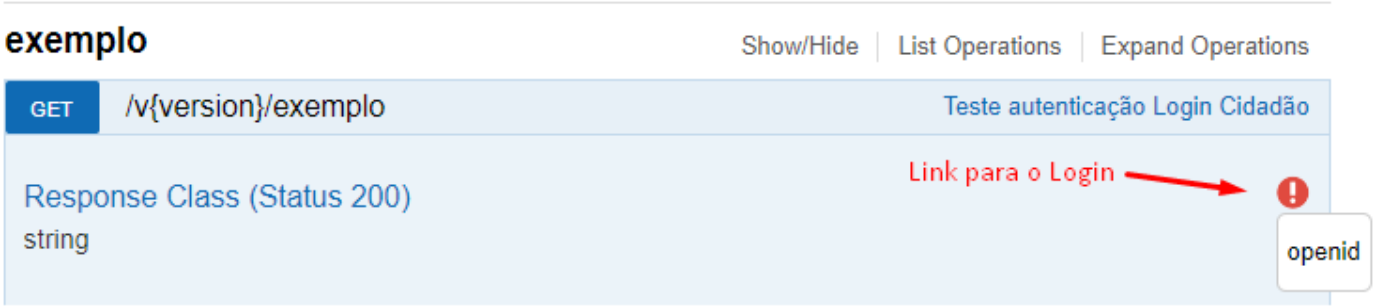
Exemplo 5 – Aplicação API – APM 3

Este exemplo acrescenta a autenticação com o *Access Token* gerado pelo Login Cidadão mantendo a autenticação com o SOE Web existente. Esse segundo autenticador é registrado no arquivo *Startup.Auth.cs*, conforme o trecho abaixo:

```
35      // Acrescenta a autenticação pelo Login Cidadão
36      var metadataAddress = ConfigurationManager.AppSettings["Oidc.MetadataAddress"];
37      app.UseOAuthBearerAuthentication(new OAuthBearerAuthenticationOptions()
38      {
39          AccessTokenFormat = new LoginCidadaoAuthenticationTicket(metadataAddress)
40      });
41
```

Ele configura o propriedade *AccessTokenFormat* para uma classe personalizada que irá fazer a validação correta do *access token* enviado pelo Login Cidadão.

O **Swagger** também foi atualizado para permitir as 2 autenticações. O método de exemplo abaixo permite fazer a autenticação do Login Cidadão:



O atributo *Authorize* da *controller* foi configurado com a propriedade *Roles* para diferenciar da autenticação do SOE:

```
namespace Procergs.Apm.Api.Controllers
{
    [Authorize(Roles = "LoginCidadao")]
    [ApiVersion("1")]
    [ControllerName("exemplo")]
    1 reference
    public class ExemploController : ApiController
    {
```

O conjunto com todos ajustes adicionados no código pode ser obtido a partir deste commit.