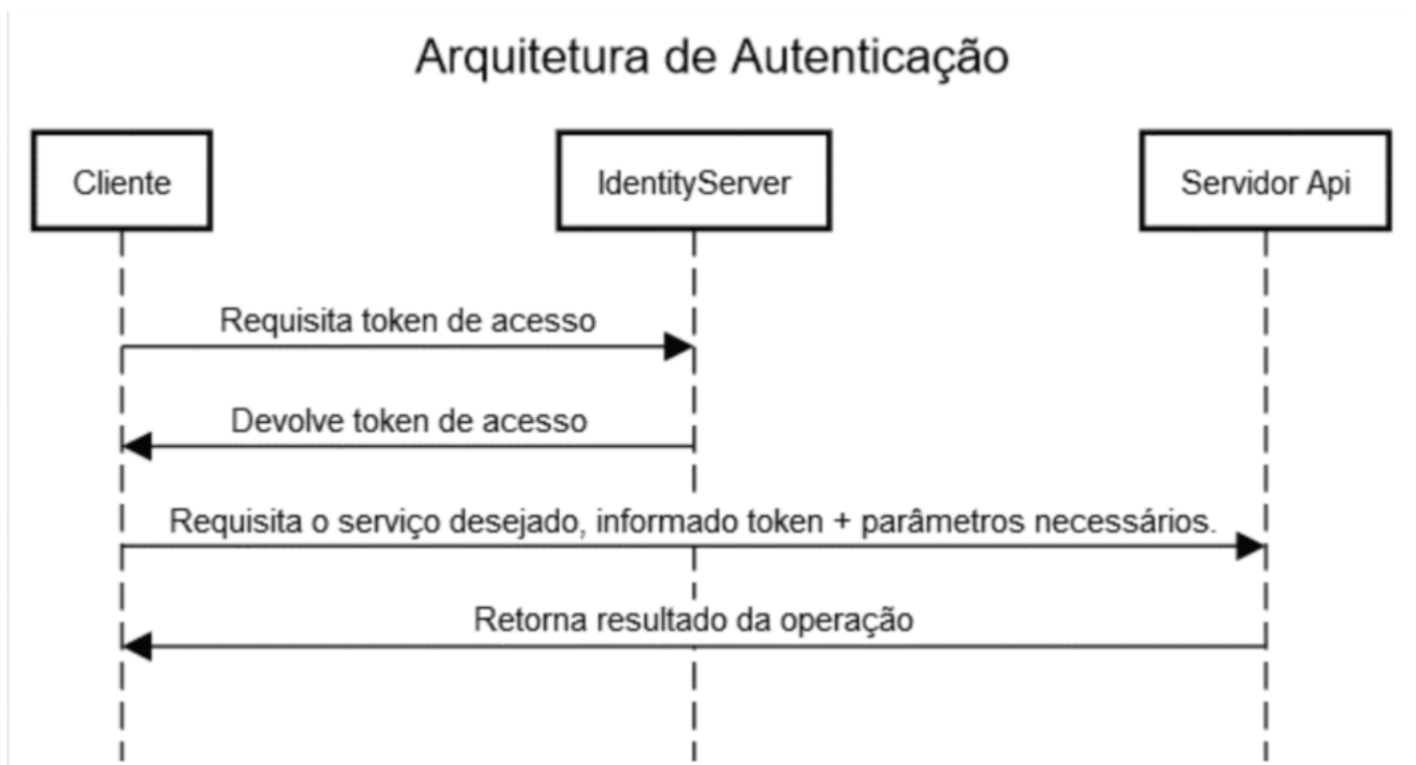


# Manual de autenticação de sistema - Interno

## Introdução

Este documento tem o objetivo de descrever os passos necessários para possibilitar a autenticação de uma aplicação externa ao “Cebi.Login”, sendo este a estratégia oficial da Cebi para autenticação única, apoiada na aplicação open source chamada IdentityServer.

**Importante:** este manual é apenas para uso interno e **NÃO** deve ser distribuído a clientes. Existe um **manual a parte**, mais enxuto e que pode ser distribuído a clientes para uso externo.



## Grant Types

Existem alguns tipos de concessão de acesso que são suportados pelo IdentityServer, entre eles: Implicit, Resource Owner Password e Client Credentials. As duas primeiras são úteis para obter acesso à uma aplicação em nome de um usuário, porém, obviamente neste caso seria necessário informar o login e senha no momento de solicitar um token de acesso.

Considerando um cenário onde não há uma tela de autenticação, como por exemplo numa chamada via aplicação console ou até mesmo via PL/SQL, o tipo de concessão mais recomendado é o Client Credentials pois permite solicitar um token de acesso em nome de um cliente (aplicação) e não de um usuário. Portanto, esse será o tipo de concessão utilizado neste tipo de situação.

# Criação do Cliente no IdentityServer

A primeira coisa a ser feita é a criação do cliente que será utilizado na autorização. Quando o token for solicitado, o IdentityServer tentará localizar um cliente com o nome informado no parâmetro 'client\_id'.

Para criar um novo cliente, utilize o programa Cebi.Util.GerenciamentoIntegracoesApi (localizado no BitBucket) ou fale com a equipe de arquitetura.

O programa vai solicitar alguns dados, o nome do usuário do cliente (recomendamos o padrão camelCase) e o nome amigável do cliente. Ele retornará o usuário e senha gerados, assim como o script de criação do mesmo, para execução na base de testes e na base do cliente.

**Importante: O programa indicado ou a equipe de arquitetura NÃO mantém um controle dos usuários, senhas e SQLs criados pelo programa. Cada equipe é responsável pelo seu próprio controle**

## Criação manual do cliente no IdentityServer

Abaixo um exemplo de um registro criado na tabela ID\_CLIENTS:

Campo	Valor
ID	(AUTO)

Campo	Valor
ENABLED	1
CLIENT_ID	oracle.plsql (exemplo)
CLIENT_NAME	Oracle PL/SQL (exemplo)
CLIENT_URI	(NULL)
REQUIRE_CONSENT	0
ALLOW_REMEMBER_CONSENT	1
ALLOW_ACCESS_TOKENS_VIA_BROWSE	1
FLOW	3
ALLOW_CLIENT_CREDENTIALS_ONLY	0
LOGOUT_SESSION_REQUIRED	0
REQUIRE_SIGN_OUT_PROMPT	0
ALLOW_ACCESS_TO_ALL_SCOPES	1
IDENTITY_TOKEN_LIFETIME	3600
ACCESS_TOKEN_LIFETIME	3600
AUTHORIZATION_CODE_LIFETIME	3600
ABSOLUTE_REFRESH_TOKEN_LIFETIME	0
SLIDING_REFRESH_TOKEN_LIFETIME	3600
REFRESH_TOKEN_USAGE	0
UPDATE_ACCESS_TOKEN_ON_REFRESH	0
REFRESH_TOKEN_EXPIRATION	1
ACCESS_TOKEN_TYPE	0
ENABLE_LOCAL_LOGIN	1
INCLUDE_JWT_ID	0
ALWAYS_SEND_CLIENT_CLAIMS	1
PREFIX_CLIENT_CLAIMS	0
ALLOW_ACCESS_TO_ALL_GRANT_TYPE	0

# Criação manual das senhas do cliente

Deve-se criar uma ou mais senhas para o cliente cadastrado no passo anterior. Segue abaixo exemplo de registro a ser criado na tabela:

Campo	Valor
ID	(AUTO)
VALUE	K7gNU3sdo...
TYPE	SharedSecret
Client_Id	(ID do registro criado anteriormente)

A senha deve ser gerada usando criptografia SHA256 + BASE64. No exemplo acima, a senha 'secret' foi transformada para 'K7gNU3sdo+OL0wNhqoVW3g6s1xYv72ol/pe/Unols='. Para gerar senhas diferentes, você pode usar o site: <https://approsto.com/sha-generator>.

A senha deve terminar em "=".

## Criação manual dos escopos

Neste passo, deve-se criar o escopo relacionado ao recurso que desejado. Em tese, este item já deverá existir pois do contrário, este recurso não estaria disponível para os usuários que acessam o recurso via aplicação web. De qualquer maneira, estará aqui para que faça sentido a sua utilização na requisição do token que virá a seguir. Segue abaixo exemplo de registro da tabela ID\_SCOPES:

Campo	Valor
ENABLED	1
NAME	Cebi.Api
DISPLAY_NAME	Api's CEBI Informática
DESCRIPTION	Permite o acesso às API's da CEBI Informática
REQUIRED	0
EMPHASIZE	0
TYPE	1
INCLUDE_ALL_CLAIMS_FOR_USER	1
CLAIMS_RULE	(NULL)
SHOW_IN_DISCOVERY_DOCUMENT	1
ALLOW_UNRESTRICTED_INTROSPECTI	0

# Requisição do token

Após a criação dos registros necessários na base de dados do IdentityServer, podemos então fazer a requisição do token. Segue abaixo um modelo de chamada HTTP.

- Endereço da API: Informado pela Cebi
- Método: Post
- Cabeçalhos (headers):
  - Content-Type: application/x-www-form-urlencoded
- Corpo da mensagem (body):
  - A mensagem deve ser no formato x-www-form-urlencoded;
  - Campos da requisição:
    - grant\_type: client\_credentials
    - client\_id: Informado pela Cebi
    - client\_secret: Informado pela Cebi
    - scope: Cebi.Api

Ou, num formato mais técnico:

```
POST /login/connect/token HTTP/1.1
Host: win13.cebisist:20003
User-Agent: PostmanRuntime/7.16.3
Accept: */*
Cache-Control: no-cache
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
Content-Length: 90
Connection: keep-alive
cache-control: no-cache
grant_type=client_credentials&client_id=oracle.plsql&client_secret=secret&scope=Cebi.Api
```

The screenshot shows a REST client interface with the following details:

- URL:** http://win13.cebisist:50034/login/connect/token
- Method:** POST
- Body Type:** x-www-form-urlencoded
- Body Data:**

Key	Value
grant_type	client_credentials
client_id	cebi-teste-liberacao
client_secret	uJwQZIKaJpOKVhl
scope	Cebi.Api

# Resultado



# Requisição de serviço

Uma vez que o token de autenticação tenha sido recebido, basta fazer a requisição ao serviço desejado, informando o token no cabeçalho de autorização (Authorization: Bearer ). Veja exemplo de requisição:

```
GET /usuarios/api/appinfo HTTP/1.1
Host: win13.cebisist:20003
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUz...
User-Agent: PostmanRuntime/7.16.3
Accept: */*
Cache-Control: no-cache
Accept-Encoding: gzip, deflate
Connection: keep-alive
cache-control: no-cache
```

**Observação:** Não será colocado um exemplo de resposta pois ela varia de acordo com o serviço solicitado.

# Observações finais

# Ausência de dados do usuário

Vale lembrar que neste modo de autenticação, estamos fazendo um acesso em nome de um cliente (aplicação) e não de um usuário. Portanto, dados como “login” ou “usuário\_id” não estão disponíveis no token. Portanto, o serviço (API) deve considerar essa situação e atuar de acordo com a requisição.

# Autorização do cliente

A versão 1.2.6 ou inferior, possuía um filtro chamado CebiPerfilAuthorize que verificava se o usuário possuía privilégio para acessar o recurso desejado (rota). No entanto, para o cenário em que a requisição é feita por uma aplicação externa, usando um token que não possui dados de usuário, o filtro retornava o erro “Não foi possível identificar o usuário”. Por esta razão, foi feita uma modificação no filtro para verificar se o cliente que possui o token tem acesso à aplicação requisitada. Para que isso funcione corretamente, é necessário atualizar o pacote Cebi.Util.WebApi para a versão 1.2.7 e tornar explícito a autorização de acesso para o cliente desejado. Segue abaixo exemplo de como isso pode ser feito:

```
using Cebi.Permissoes.Dal;
using Cebi.Permissoes.Domain.Services;

.
.
namespace Cebi.Usuarios.Api.Services
{
    public class ConfiguradorFuncionalidadesUsuariosService : IConfiguradorFuncionalidades
    {
        public void CarregarFuncionalidades()
        {
            // <Código necessário para carregar as funcionalidades>
        }

        public void MapearPermissoes()
        {
            var unit = new PermissoesUnitOfWork();
            var mapearPermissoesService = new MapearPermissoesService(unit);
            var permissoes = mapearPermissoesService.Executar();

            CebiPerfilAuthorizeAttribute.FuncionalidadesAutorizadas = permissoes;

            // Aqui vai o código para autorizar os clientes desejados.
            CebiPerfilAuthorizeAttribute.ClientesAutorizados = new List<string>
            {
                "oracle.plsql"
            }
        }
    }
}
```

```
};  
}  
  
public List<string> ObterPermissoes()  
{  
    // <Código necessário para listar as permissões>  
}  
}  
}
```

---

Revisão #4

Criado 29 abril 2024 19:58:39 por Guilherme Bortolloti

Atualizado 11 dezembro 2024 18:21:03 por Guilherme Bortolloti