客户端认证为OAuth2的认证模式之一，其针对的场景是客户端不存在或不关系用户，如客户端是定时服务器或者客户端是某个工具软件

**认证服务器配置**

1. 定义

    public class Config

    {

        // 身份资源

        public static IEnumerable<IdentityResource> GetIdentityResources()

        {

            return new List<IdentityResource>

            {

                new IdentityResources.OpenId(),

                new IdentityResources.Profile(),

            };

        }

        // API域

        public static IEnumerable<ApiScope> GetApis()

        {

            return new List<ApiScope>

            {

                new ApiScope("server1.all", "ice All API")

            };

        }

        // API资源

        public static IEnumerable<ApiResource> GetApiResources()

        {

            return new List<ApiResource>

            {

                new ApiResource("server1", "My API")

                {

                    Scopes = { "server1.all" }

                },

            };

        }

        // 客户端

        public static IEnumerable<Client> GetClients()

        {

            return new List<Client>

            {

                new Client

                {

                    ClientId = "client1",

                    // 没有交互性用户，使用 clientid/secret 实现认证。

                    AllowedGrantTypes = GrantTypes.ClientCredentials,

                    // 用于认证的密码

                    ClientSecrets =

                    {

                        new Secret("secret".Sha256())

                    },

                    // 当前客户端有权访问的api域（Scopes）

                    AllowedScopes = { "server1.all" }

                }

            };

        }

    }

1. 在.NET中配置

public void ConfigureServices(IServiceCollection services)

{

    services.AddIdentityServer()    // 使用IdentityServer

        .AddDeveloperSigningCredential()    // 生成临时密匙

// 添加Api资源

        .AddInMemoryApiScopes(Config.GetApis())

        .AddInMemoryApiResources(Config.GetApiResources())

        .AddInMemoryClients(Config.GetClients());   // 添加客户端

}

运行此项目，打开浏览器访问http://localhost:5000/.well-known/openid-configuration你将会看到所谓的发现文档。

**获取Access Token**

启动后我们访问如下API去获取客户的access token

POST /connect/token

CONTENT-TYPE application/x-www-form-urlencoded

body

client\_id=client1& client\_secret=secret& grant\_type=client\_credentials& scope=server1.all

根据该API获取的AccessToken为jwt token，我们访问资源服务器时将该token放置在请求头部即可

**资源服务器配置**

资源服务器要进行jwt认证，需要添加如下配置

services

    .AddAuthentication(JwtBearerDefaults.AuthenticationScheme)

    .AddJwtBearer(options =>

    {

        options.Authority = "http://localhost:5000";

        options.RequireHttpsMetadata = true;

        options.Audience = "server1";

    });

Audience对应的就是资源名