在配置 Client 客户端的时候 Token 的类型有两种，IdentityServer4 默认使用 JWT 类型。

/// <summary>

/// Access token types.

/// </summary>

public enum AccessTokenType

{

/// <summary>

/// Self-contained Json Web Token

/// </summary>

Jwt = 0,

/// <summary>

/// Reference token

/// </summary>

Reference = 1

}

Jwt 类型

当使用 AccessTokenType 类型为 Jwt 时候，就会使用 Jwt 规范来生成 Token，签名的算法是采用 RSA (SHA256 签名) ，在服务端 IdentityServer4 使用私钥对 Token 进行签名，当客户端去资源端获取资源的时候，API 端(资源服务器)收到第一个请求后去服务端获得公钥然后验签（**调**用 /.well-known/openid-configuration/jwks 获取公**钥这个过程只发生在客户端第一次请求，所以当服务端更换证书，资源端也需要重启服务**）。

一般开发环境使用 [AddDeveloperSigningCredential](https://https//github.com/IdentityServer/IdentityServer4/blob/dev/src/IdentityServer4/Configuration/DependencyInjection/BuilderExtensions/Crypto.cs" \l "L110" \t "https://www.cnblogs.com/Irving/articles/_blank) 方法使用临时证书即可

以 .net core 为例，客户端添加如下认证：

包：IdentityServer4.AccessTokenValidation

services.AddAuthentication("Bearer")

.AddIdentityServerAuthentication("Bearer", options =>

{

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

options.RequireHttpsMetadata = false;

})

Reference类型

当使用 Reference token 的时候，服务端会对 Token 进行持久化，当客户端请求资源端（API）的时候，资源端需要每次都去服务端通信去验证 Token 的合法性[[/connect/introspect](https://identityserver4.readthedocs.io/en/release/endpoints/introspection.html" \t "https://www.cnblogs.com/Irving/articles/_blank)]，IdentityServer4.AccessTokenValidation 中间件中可以配置缓存一定的时候去验证,并且 Token 是支持撤销[[/connect/revocation](https://identityserver4.readthedocs.io/en/release/endpoints/revocation.html" \t "https://www.cnblogs.com/Irving/articles/_blank)]的。

使用Reference类型需要配置 ApiSecrets

以 .net core 为例，客户端添加如下认证：

包：IdentityServer4.AccessTokenValidation

services.AddAuthentication("Bearer")

.AddIdentityServerAuthentication("Bearer", options =>

{

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

options.RequireHttpsMetadata = false;

options.ApiName = "api1";

options.ApiSecret = "apisecret";

})

其http流程：

POST [http://localhost:5000/connect/introspect HTTP/1.1](http://localhost:5000/connect/introspect)

Content-Type: application/x-www-form-urlencoded

cache-control: no-cache

Postman-Token: 0c326c74-aaac-4ce3-854c-51559197093b

User-Agent: PostmanRuntime/7.4.0

Accept: \*/\*

Host: localhost:5000

accept-encoding: gzip, deflate

content-length: 138

Connection: keep-alive

token=43c561bed20656f93ffe28dc8f7a22955fe791c6cbbe3732d1da93aedcc85dd7&token\_type\_hint=access\_token&client\_id=api1&client\_secret=apisecret

HTTP/1.1 200 OK

Cache-Control: no-store, no-cache, max-age=0

Pragma: no-cache

Transfer-Encoding: chunked

Content-Type: application/json; charset=UTF-8

Server: Kestrel

X-Content-Type-Options: nosniff

X-XSS-Protection: 1; mode=block

X-Frame-Options: SAMEORIGIN

X-SourceFiles: =?UTF-8?B?RjpccHJvamVjdHNcTWVtYmVyQXBpXEF1dGhlbnRpY2F0aW9uQ2VudGVyXHNyY1xBdXRoZW50aWNhdGlvbkNlbnRlci5XZWJcY29ubmVjdFxpbnRyb3NwZWN0?=

X-Powered-By: ASP.NET

Date: Fri, 30 Nov 2018 02:59:44 GMT

102

{"iss":"<http://localhost:5000>","nbf":1543544793,"exp":1543548393,"aud":["<http://localhost:5000/resources>","api1","api2","api3"],"client\_id":"mvc","sub":"4127","auth\_time":1543544788,"idp":"local","amr":"pwd","name":"18888888887","active":true,"scope":"api1"}

0