



ComponentSpace OpenID Connect for ASP.NET Core Web Farm Guide

Contents

Introduction.....	1
In-Memory Cache	1
Redis Cache	1
SQL Server Cache	2

Introduction

This guide describes the steps required to support an OpenID provider in a web farm deployment.

In-Memory Cache

The OpenID session state and tokens are cached through the `IDistributedCache` interface.

The implementation of `IDistributedCache` may be specified through dependency injection.

The default implementation of `IDistributedCache` caches to memory.

This is suitable for single server deployments and web farm deployments where a load balancer and sticky sessions are used.

For web farm deployments where sticky sessions are not used, an `IDistributedCache` implementation such as the `RedisCache` or `SqlServerCache` must be used.

Redis Cache

The following code configures a Redis cache that's used for the OpenID cache.

```
// Add the Redis cache.
builder.Services.AddStackExchangeRedisCache(options =>
{
    options.Configuration = builder.Configuration.GetConnectionString("RedisConnection");
});

// Add OpenID provider services.
builder.Services.AddOpenIDProvider(builder.Configuration.GetSection("OpenIDProvider"));
```

The following configuration specifies a Redis cache on localhost using the default port 6379.

```
"ConnectionStrings": {
  "RedisConnection": "localhost"
},
```

For information on Redis, refer to:

<https://redis.io/>

For information on distributed caches, refer to:

<https://docs.microsoft.com/en-us/aspnet/core/performance/caching/distributed>

For information on Redis configuration strings, refer to:

<https://stackexchange.github.io/StackExchange.Redis/Configuration.html>

For testing purposes, a Windows port of Redis is available at:

<https://github.com/MicrosoftArchive/redis>

SQL Server Cache

The following code configures a SQL server cache that's used for the OpenID cache.

```
// Add the SQL server cache.
builder.Services.AddDistributedSqlServerCache(options =>
{
    options.ConnectionString = builder.Configuration.GetConnectionString("DistCache");
    options.SchemaName = "dbo";
    options.TableName = "Cache";
});

// Add OpenID provider services.
builder.Services.AddOpenIDProvider(builder.Configuration.GetSection("OpenIDProvider"));
```

The following configuration specifies the connection string for the cache database on (localdb)\MSSQLLOCALDB. In a production environment, SQL Server or another RDBMS should be used.

```
"ConnectionStrings": {
  "DistCache":
  "Server=(localdb)\MSSQLLOCALDB;Database=DistCache;Trusted_Connection=True;MultipleActiveResultSets=true"
},
```

The database may be created using various tools including the SQL Server Management Studio and Visual Studio's SQL Server Object Explorer.

Once created, run the sql-cache tool to initialize the database.

```
dotnet sql-cache create "Data Source=(localdb)\MSSQLLocalDB;Initial
Catalog=DistCache;Integrated Security=True;" dbo Cache
```

For information on distributed caches, refer to:

<https://docs.microsoft.com/en-us/aspnet/core/performance/caching/distributed>