ComponentSpace
SAML for ASP.NET Core
Web Farm Guide

www.componentspace.com

Contents

Introduction

This guide describes the steps required to support SAML SSO in a web farm (i.e. multi-server) deployment.

SSO Session Store

Both identity providers and service providers use the SSO session store.

The default SSO session store stores data in an IDistributedCache.

The implementation of IDistributedCache may be specified through dependency injection.

The default implementation of IDistributedCache caches to memory.

This is suitable for 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 should be specified.

ID Cache

A service provider stores SAML assertion IDs in an ID cache to detect potential replay attacks. An identity provider does not use the ID cache.

The default ID cache stores IDs in an IDistributedCache.

The implementation of IDistributedCache may be specified through dependency injection.

The default implementation of IDistributedCache caches to memory.

For web farm deployments, an IDistributedCache implementation such as the RedisCache or SqlServerCache should be specified.

If acting as an identity provider, the ID cache may be ignored.

Artifact Cache

Artifacts are stored in an artifact cache when SAML messages are transmitted using the HTTP Artifact binding. Otherwise, the artifact cache is not used.

The default artifact cache stores artifacts in an IDistributedCache.

The implementation of IDistributedCache may be specified through dependency injection.

The default implementation of IDistributedCache caches to memory.

For web farm deployments, an IDistributedCache implementation such as the RedisCache or SqlServerCache should be specified.

If the HTTP artifact binding is not used, the artifact cache may be ignored.

Redis Cache

The following code configures a Redis cache that's used for the SSO session store, ID cache and Artifact cache.

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

// Add SAML SSO services.
builder.Services.AddSaml(builder.Configuration.GetSection("SAML"));
```

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 SSO session store, ID cache and Artifact cache.

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

// Add SAML SSO services.
builder.Services.AddSaml(builder.Configuration.GetSection("SAML"));
```

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.

For example, the following create a database on localdb:

```
sqlcmd -S "(localdb)\MSSQLLocalDB" -E -Q "CREATE DATABASE DistCache"
```

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 DistCache
```

For information on distributed caches, refer to:

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