Tabla de Contenidos

Descripción General

SPN ofrece dos soluciones de Single Sign-On (SSO) para integrar los sistemas de autenticación corporativa de los clientes:

- **Autoservicio Web SSO**: Integración SAML 2.0 para aplicaciones web
- **SPN Mobile App SSO**: Integración OAuth 2.0/MSAL para aplicaciones móviles

Autoservicio Web SSO

Tabla de Contenidos

Autoservicio Descripción General

La solución SSO (Single Sign-On) para Autoservicio SPN permite a los clientes integrar su sistema de autenticación corporativo con la plataforma Autoservicio, proporcionando una experiencia de usuario unificada y segura.

Beneficios Principales

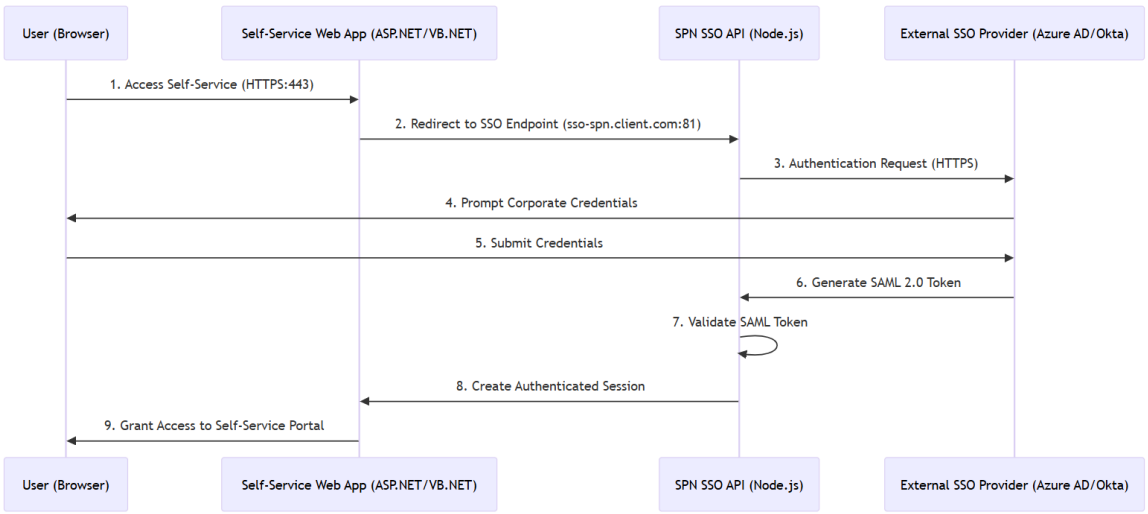
- **Autenticación unificada**: Los usuarios utilizan sus credenciales corporativas existentes
- **Seguridad mejorada**: Centralización de políticas de autenticación y autorización
- **Experiencia del usuario**: Eliminación de múltiples logins y contraseñas
- **Cumplimiento**: Adherencia a políticas corporativas de seguridad

Autoservicio Arquitectura de la Solución

```
┌─────────────────────────────────────────────────────┐
│              Cliente (Servidor Windows)             │
├─────────────────────────────────────────────────────┤
│ 1. Autoservicio Web App (ASP.NET/VB.NET)            │
│    - Puerto: 80/443                                 │
│    - Interfaz de usuario principal                  │
│                    ↕                                │
│ 2. SPN SSO API (Node.js)                            │
│    - Puerto: 81/443                                 │
│    - Procesamiento SAML                             │
│    - Dominio: sso-spn.cliente.com                   │
│                    ↕                                │
│ 3. Proveedor SSO Cliente (Azure AD / Okta)          │
│    - Configurado y administrado por el cliente      │
│    - Autenticación de usuarios corporativos         │
└─────────────────────────────────────────────────────┘
```

Flujo de Autenticación

1. **Usuario accede** a Autoservicio
2. **Autoservicio redirige** al SPN SSO API
3. **API redirige** al proveedor SSO del cliente
4. **Usuario se autentica** con credenciales corporativas
5. **Proveedor SSO** envía token SAML a la API
6. **API procesa** el token y valida la información
7. **Usuario es redirigido** a Autoservicio con sesión activa



Proveedores SSO Soportados

Microsoft Azure AD

- **Protocolo**: SAML 2.0
- **Formato de usuario**: Email corporativo
- **Configuración**: Enterprise Application
- **Ideal para**: Organizaciones con ecosistema Microsoft 365

Okta

- **Protocolo**: SAML 2.0
- **Formato de usuario**: ID de empresa o email
- **Configuración**: SAML 2.0 Web App
- **Ideal para**: Organizaciones con identidades híbridas o multi-cloud

Autoservicio Responsabilidades del Cliente

El Cliente es Responsable de:

Infraestructura y Hosting

- Proporcionar servidor Windows para hospedar Autoservicio y SPN SSO API
- Mantener y administrar la infraestructura de servidor
- Configurar y mantener conectividad de red
- Gestionar backups y recuperación ante desastres

Proveedor SSO

- **Administrar completamente su entorno SSO** (Azure AD, Okta, etc.)
- **Configurar la aplicación SAML** en su proveedor SSO
- **Asignar usuarios** a la aplicación SSO
- **Mantener usuarios activos** y gestionar accesos
- **Proporcionar credenciales de prueba** para validación

Certificados SSL

- **Adquirir certificado SSL válido** para el dominio de la API
- **Instalar y mantener** el certificado en IIS
- **Renovar certificados** antes de su expiración

Vinculación de Usuarios

- **Identificar método de vinculación** (email o ID de empresa)
- **Proporcionar mapping** de usuarios SSO con usuarios de Autoservicio
- **Mantener sincronización** de usuarios entre sistemas

Autoservicio Requerimientos Técnicos

Servidor y Software

- **Windows Server 2016 o superior**
- **Internet Information Services (IIS) 8.5+**
- **Node.js 16 LTS o superior**
- **iisnode module** para IIS
- **.NET Framework 4.7.2+** (para Autoservicio)

Red y Conectividad

- **Acceso a internet** para comunicación con proveedores SSO
- **Puerto 80/443** disponible para Autoservicio
- **Puerto adicional** (ej: 81/443) para SPN SSO API
- **Firewall configurado** para permitir tráfico HTTPS
- **DNS configurado** para subdominio de la API

Certificado SSL

- **Certificado SSL válido** para el dominio de la API SSO
- **Formato .pfx** con clave privada incluida
- **Cadena de certificados** completa si es requerida
- **Válido por al menos 12 meses**

Proveedor SSO

- **Tenancy activo** en Azure AD u Okta
- **Permisos administrativos** para configurar aplicaciones SAML
- **Usuarios configurados** y activos en el sistema
- **Licenciamiento apropiado** según número de usuarios

Autoservicio Información Requerida por SPN

Para Configuración del API

Del Proveedor SSO

- **Metadata XML** del proveedor SSO
- **Certificado X.509** (si no está incluido en metadata)
- **Entity ID** del proveedor
- **Single Sign-On URL**
- **Single Logout URL** (si está disponible)

Del Entorno Cliente

- **Dominio definitivo** para la API SSO (ej: sso-spn.cliente.com)
- **Método de vinculación** preferido (email o ID de empresa)
- **Información de usuarios de prueba** para validación

Acceso para Instalación

- **Acceso remoto** al servidor de producción
- **Credenciales de administrador** local del servidor
- **Ventana de mantenimiento** acordada para instalación

Autoservicio Proceso de Implementación

Fase 1: Planificación y Diseño

Duración: 1-2 días
Responsable: SPN + Cliente

- Definición de arquitectura específica del cliente
- Selección de dominio para API SSO
- Identificación de método de vinculación de usuarios
- Planificación de ventana de implementación

Fase 2: Preparación del Cliente

Duración: 3-5 días
Responsable: Cliente

- Adquisición y configuración de certificado SSL
- Preparación del servidor Windows
- Configuración de aplicación SAML en proveedor SSO
- Generación de metadata y certificados del proveedor

Fase 3: Desarrollo y Configuración

Duración: 2-3 días
Responsable: SPN

- Configuración del SPN SSO API con información del cliente
- Preparación de archivos de configuración
- Testing en ambiente de desarrollo
- Preparación de paquete de instalación

Fase 4: Instalación y Despliegue

Duración: 1 día
Responsable: SPN + Cliente

- Instalación de prerequisitos en servidor
- Despliegue del SPN SSO API
- Configuración de IIS y certificados SSL
- Integración con Autoservicio existente

Fase 5: Testing y Validación

Duración: 1-2 días
Responsable: SPN + Cliente

- Pruebas de conectividad y autenticación
- Validación de vinculación de usuarios
- Testing de flujo completo de login
- Resolución de issues identificados

Fase 6: Go-Live y Soporte

Duración: 1 día + soporte continuo
Responsable: SPN + Cliente

- Activación en producción
- Monitoreo inicial del sistema
- Entrenamiento a administradores del cliente
- Documentación de configuración final

Autoservicio Configuraciones del Cliente

En Azure AD

El cliente debe configurar una **Enterprise Application** con:

- **Identifier (Entity ID)**: Dominio de la API SSO
- **Reply URL**: Endpoint ACS de la API SSO
- **Sign on URL**: Endpoint de login de la API SSO
- **Attribute mappings**: Email, nombre, y ID de usuario
- **User assignment**: Usuarios que tendrán acceso

En Okta

El cliente debe configurar una **SAML 2.0 Web App** con:

- **Single sign on URL**: Endpoint ACS de la API SSO
- **Audience URI**: Dominio de la API SSO
- **Name ID format**: EmailAddress o Unspecified
- **Attribute statements**: Mapping de atributos de usuario
- **Application assignments**: Usuarios o grupos con acceso

Configuración de Usuarios

Vinculación por Email (Azure AD típico)

- Usuarios en Autoservicio deben tener email que coincida con Azure AD
- Verificar que emails sean únicos y válidos
- Sincronizar cambios de email entre sistemas

Vinculación por ID de Empresa (Okta típico)

- Usuarios en Autoservicio deben tener campo de ID de empresa
- Mapear IDs corporativos con usuarios de Autoservicio
- Mantener consistencia en formato de IDs

Autoservicio Validación y Testing

Pruebas de Conectividad

- Verificar acceso a URLs del proveedor SSO
- Confirmar resolución DNS del dominio de la API
- Validar certificado SSL instalado correctamente

Pruebas de Autenticación

- Login exitoso con usuario de prueba
- Verificar información recibida en token SAML
- Confirmar redirección correcta a Autoservicio

Pruebas de Vinculación

- Validar que usuarios SSO se vinculen correctamente
- Probar escenarios de usuario no encontrado
- Verificar permisos y roles en Autoservicio

Pruebas de Sesión

- Confirmar duración apropiada de sesiones
- Probar logout y cierre de sesión
- Validar comportamiento con múltiples pestañas/ventanas

Autoservicio Soporte y Mantenimiento

Responsabilidades de SPN

- **Soporte técnico** del SPN SSO API
- **Actualizaciones** de seguridad y funcionalidad
- **Monitoreo** de logs y errores del API
- **Documentación** y guías de troubleshooting

Responsabilidades del Cliente

- **Mantenimiento** del proveedor SSO
- **Gestión** de usuarios y accesos
- **Renovación** de certificados SSL
- **Mantenimiento** de infraestructura de servidor
- **Backup** y recuperación de configuraciones

Escalación de Soporte

- **Nivel 1**: Issues del SPN SSO API → Soporte SPN
- **Nivel 2**: Issues del proveedor SSO → Administrador del cliente
- **Nivel 3**: Issues de infraestructura → Equipo de IT del cliente

Monitoreo y Alertas

- Logs de autenticación en SPN SSO API
- Monitoreo de certificados próximos a expirar
- Alertas de conectividad con proveedor SSO
- Métricas de usuarios y sesiones activas

Argumentos Técnicos y Arquitectónicos

Argumentos sobre Certificado SSL

Requerimiento Técnico No Negociable

- Los navegadores modernos bloquean "Mixed Content" (HTTPS → HTTP)
- Azure AD y Okta rechazan automáticamente endpoints HTTP para SAML
- El protocolo SAML 2.0 requiere cifrado en tránsito por estándar
- Sin HTTPS, literalmente no hay comunicación posible con proveedores SSO

Flujo Real de Usuario

- Usuario desde su PC → Autoservicio (HTTPS) → Usuario redirigido a API SSO
- La llamada viene del navegador del usuario, NO del servidor
- Es comunicación externa que requiere DNS público y certificado válido

Argumentos contra Servidor Separado

Riesgos de Seguridad Aumentados

- **Superficie de ataque ampliada**: Dos servidores = doble exposición
- **Complejidad de firewall**: Más puertos y reglas a gestionar
- **Sincronización de parches**: Dos sistemas operativos que mantener
- **Logs distribuidos**: Auditoría más compleja y propensa a errores

Problemas de Comunicación

- Latencia adicional entre Autoservicio y API SSO
- Dependencia de red interna que puede fallar
- Complejidad en troubleshooting de issues de conectividad
- Más puntos de falla en la cadena de autenticación

Compliance Falso

- Separar componentes relacionados NO mejora compliance
- Compliance real requiere cifrado, auditoría y controles de acceso
- Tener dos servidores puede violar principios de "defense in depth"
- Es arquitectura menos segura, no más segura

Argumentos de Arquitectura Técnica

Por qué API Separada del .NET Legacy

- **.NET Framework antiguo**: No soporta librerías SAML modernas
- **Node.js para SAML**: Librerías actualizadas y mantenidas activamente
- **Separación de responsabilidades**: UI (.NET) vs Autenticación (Node.js)
- **Mantenibilidad**: Más fácil actualizar componente específico

Por qué Mismo Servidor es Óptimo

- **Cohesión funcional**: Ambos componentes son parte de la misma solución
- **Red interna**: Comunicación server-to-server ultra rápida
- **Gestión unificada**: Un solo servidor que patchear y monitorear
- **Costo reducido**: Una sola licencia de Windows Server

Respuestas a Objeciones Específicas

"¿Por qué necesitamos dominio público?"

- Azure AD/Okta necesitan enviar respuestas SAML de vuelta
- El usuario es quien navega entre sistemas, no el servidor
- DNS interno no es accesible para proveedores SSO externos
- Es requerimiento del protocolo SAML, no preferencia de SPN

"Compliance requiere separación"

- **Falso**: Compliance requiere controles, no separación física
- Separación sin propósito técnico introduce más riesgos
- Mejor compliance = menos componentes que asegurar
- Auditorías prefieren arquitecturas simples y bien documentadas

"¿Qué pasa si API falla?"

- En servidor separado: Autoservicio queda totalmente inaccesible
- En mismo servidor: Problema localizado, más fácil diagnóstico
- Logs centralizados permiten correlación rápida de eventos
- Recovery más rápido con componentes colocalizados

Propuesta Técnica Recomendada

Arquitectura Segura en Mismo Servidor

- IIS con dos sites: Puerto 80/443 (Autoservicio) + Puerto 81/443 (API)
- Firewall interno: Solo permitir 80/443 público, 81/443 restringido
- Certificado SSL con SAN para ambos dominios si es necesario
- Logs unificados en Event Viewer y archivos centralizados

Configuración de Red Segura

- API en puerto no estándar (81) reduce exposición
- Mismo servidor = tráfico interno no sale a red
- DNS split-brain: Interno apunta a localhost, externo al servidor
- WAF opcional para protección adicional de ambos endpoints

Mobile App SSO

Tabla de Contenidos

Mobile Descripción General

La solución de Single Sign-On (SSO) para la aplicación móvil SPN Mobile permite a los clientes autenticar a sus usuarios mediante Azure Active Directory, brindando una experiencia de acceso unificada y segura desde dispositivos móviles.

Para esta integración se utiliza la librería MSAL (Microsoft Authentication Library), orientada a facilitar la autenticación desde apps Flutter hacia Azure AD.

Beneficios Principales

- Unificación de autenticación: Uso de credenciales corporativas de Azure AD.
- Seguridad: Tokens válidos emitidos por Azure con soporte de refresh automático.
- Experiencia optimizada: Autenticación fluida con PIN o biometría.
- Compatibilidad multiplataforma: Compatible con Android e iOS.

Mobile Arquitectura de la Solución

```
┌──────────────────────────────────────────┐
│ SPN Mobile App (Flutter)                 │
├──────────────────────────────────────────┤
│ - Muestra pantalla de ID empresa         │
│ - Detecta si empresa usa SSO             │
│ - Si SSO activo, inicia sesión con MSAL  │
│ - Guarda accessToken / refreshToken      │
│ - Autenticación biométrica con token local │
│ - Logout usando MSAL logout              │
└──────────────────────────────────────────┘

↓ ↑
API REST SPN ↔ Base de datos de clientes
↓
Azure Active Directory (cliente)
```

Flujo General

1. El usuario ingresa el ID de empresa.
2. La app consulta la API para saber si la empresa usa SSO.
3. Si usa SSO:
   - Se inicia el login con MSAL.
   - El usuario ingresa sus credenciales corporativas.
   - Azure AD responde con tokens.
4. Si no usa SSO: se muestra el login tradicional con cédula/contraseña.
5. En siguientes aperturas, la app intenta autenticarse con PIN o biometría, validando el token almacenado.

Proveedor SSO Utilizado

Microsoft Azure Active Directory

- Protocolo: OAuth 2.0 / OpenID Connect
- Librería usada: MSAL para Flutter
- Formato de usuario: Email corporativo
- Método de integración: Registro de la app en Azure como Public client/native app

Mobile Responsabilidades del Cliente

Registro en Azure AD

- Registrar la app móvil “SPN Mobile” en Azure.
- Configurar redirecciones:
  - msauth:///auth
- Establecer Client ID y Tenant ID
- Asignar usuarios permitidos a la app registrada.

Provisión de información

- Proveer los siguientes datos para cada empresa en la base de datos:
  - SSO_Client_ID
  - SSO_Tenant_ID
  - SSO_Enable (booleano)

Configuraciones por plataforma

- Para Android:
  - Registrar SHA-1 de firma de la app.
- Para iOS:
  - Bundle ID y esquemas personalizados (custom URI scheme)

Mobile Requerimientos Técnicos

Flutter App

- Versión mínima de Flutter: 2.17.0
- Dependencia clave: azure_oauth: ^2.3.1
- Otras dependencias:
  - shared_preferences
  - local_auth (para biometría)
  - flutter_bloc (gestión de estado)

Mobile Información Requerida por SPN

Por cada empresa que desea integrar SSO:

- Client ID de la app en Azure
- Tenant ID del directorio
- Estado del SSO (activo/inactivo)
- SHA-1 de firma (Android)
- Bundle ID (iOS)
- URI scheme autorizado
- Usuario de prueba

Mobile Proceso de Implementación

Fase 1: Diseño y Preparación

- Se define el flujo de login dentro de la app Flutter.
- Se identifican empresas que usarán SSO.

Fase 2: Registro en Azure AD

- Cada cliente debe registrar la app con la configuración adecuada.
- Se proporcionan los valores técnicos necesarios (SHA, Bundle ID, etc.)

Fase 3: Implementación en la App

- Se crea la pantalla de empresa (ID empresa).
- Se implementa login condicional (SSO vs Tradicional).
- Se almacena el token y se valida localmente en cada sesión.
- Se usa el refresh token de manera silenciosa.

Fase 4: Testing y Validación

- Se prueban ambos flujos (SSO y tradicional).
- Se valida el token y la persistencia.
- Se prueba biometría y fallback a MSAL login.

Mobile Configuraciones del Cliente

En Azure AD

- Registrar App como Public Client.
- Establecer:
  - Redirect URI: msauth:///auth
  - Client ID: generado por Azure
  - Tenant ID: identificador del AD corporativo
- Asignar permisos (delegados):
  - openid, profile, email, offline_access
- Configurar usuarios permitidos

Mobile Validación y Testing

Pruebas de Autenticación

- Verificar login exitoso con MSAL
- Validar tokens almacenados
- Simular expiración y renovar con refresh token

Pruebas de fallback

- Simular error en token → forzar nuevo login
- Validar mensaje general en error de credenciales

Pruebas biométricas

- Activar login con biometría (si disponible)
- Validar que no requiera login cada vez

Mobile Soporte y Mantenimiento

SPN

- Mantiene lógica de integración MSAL
- Soporte para errores de autenticación
- Actualización de librerías MSAL en futuras versiones

Cliente

- Mantiene la app registrada en Azure
- Actualiza certificados o permisos cuando sea necesario
- Proporciona usuarios de prueba ante cambios

Escalación

- Nivel 1: Errores dentro de SPN App → Soporte SPN
- Nivel 2: Problemas de Azure (registro, permisos) → Equipo IT del cliente
- Nivel 3: Issues de autenticación global → Microsoft Support

Soporte y Mantenimiento

Consideraciones de Seguridad

Mejores Prácticas

- **Certificados SSL** siempre válidos y actualizados
- **Tokens SAML** con tiempo de vida limitado
- **Logs de auditoría** habilitados en todos los componentes
- **Acceso limitado** solo a usuarios autorizados

Consideraciones Finales

Para Mobile App

- Se recomienda que cada cliente tenga una app registrada separada en Azure.
- El uso de refresh token reduce la necesidad de login constante.
- La app no almacena contraseñas, solo tokens válidos emitidos por Azure.
- El proceso de logout ejecuta el cierre completo de sesión en Azure.

Documento versión 1.0 - Septiembre 2025