GUIAS DE COMPONENTES DE ADMINISTRACIÓN ELECTRÓNICA:

PASE- Pasarela de Acceso a los Servicios Electrónicos

Referencia:	PAECARM-doct-Guia PASE(1).docx
Creación:	28 de junio de 2017
Consejería:	Consejería de Presidencia y Hacienda
DIC:	Dirección General de Informática Corporativa
Servicio:	SIAC

Historial de versiones

Versión	Fecha	Elaborada	Revisada
0.1	28.06.2017	Jorge Galián Pedreño	
0.2	04.08.2017	José Antonio Sánchez Pujante	
0.3	19.09.2017	José Antonio Sánchez Pujante	
0.4	10.11.2017	José Antonio Sánchez Pujante	
0.5	02.01.2018 12.01.2018	José Antonio Sánchez Pujante	
1	23.03.2020	Antonio Bravo Meseguer	
1.1	15.12.2020	Jesús Poveda Cobarro	
Cambios			
Nuevas características: • Permitir invocar pase con URL de vuelta con varios parámetros			

Contenido

1 INTRODUCCIÓN

1.1 ¿Qué es un servidor CAS?

El Servicio de Autenticación Central (CAS) es un protocolo de inicio de sesión único para aplicaciones web. Su objetivo es permitir que un usuario acceda a múltiples aplicaciones mientras proporciona sus credenciales (como identificador de usuario y contraseña) solo una vez. De este modo se impide que dichas aplicaciones necesiten implementar un mecanismo propio de login y mantener, en la mayoría de los casos, una base de datos local con los usuarios del sistema contra la que poder validar las credenciales de usuario. Esto asegura al usuario que una vez realizada su identificación contra la pantalla de login de CAS, no se le volverán a solicitar sus credenciales durante un tiempo determinado mientras utilice aplicaciones del dominio.

El nombre CAS también se refiere a un paquete de software que implementa este protocolo. Dicho protocolo, por lo general, comprende tres partes:

1. Un navegador web de cliente
2. La aplicación web que solicita autenticación
3. El servidor CAS.

Cuando el cliente visita una aplicación que requiere autenticación, la aplicación la redirige a CAS. CAS valida la autenticidad del cliente, generalmente al verificar un nombre de usuario y una contraseña en una base de datos, LDAP, Cl@ve, etc.

Si la autenticación tiene éxito, CAS devuelve el cliente a la aplicación, pasando un ticket de servicio. La aplicación valida el ticket contactando a CAS a través de una conexión segura y proporcionando su propio identificador de servicio y el ticket. A continuación, CAS proporciona a la aplicación información confiable sobre si un usuario en particular se ha autenticado con éxito, y mediante el protocolo SAML(Security Assertion Markup Language) se recuperan todos los atributos objeto de dicha autenticación (nombre, apellidos, DNI, etc).

1.2 Proceso de Autenticación y Validación del ticket

El proceso de autenticación y validación de un navegador que accede a varias aplicaciones de un dominio CAS.
Inicialmente contemplaremos el caso de un usuario que utiliza un navegador limpio (sin sesiones ni cookies establecidas), que intenta acceder por primera vez a una aplicación que está bajo el dominio de CAS (integrada con CAS). Hasta llegar a la situación donde la aplicación conoce la identidad del usuario que pretende acceder, la secuencia de pasos que se realiza es:

1. **Solicitud del navegador** de un recurso de la aplicación (http://servidor/aplicacion/recurso).
2. Intercepción de la solicitud por parte del control de autenticación. Comprueba que no existe sesión en servidor ni ticket de CAS.

3. **Redirección a la página de login de CAS** pasando como parámetro "service" la url interceptada, ejemplo: https://pase.carm.es/pase/login?service=http://servidor/aplicacion/recurso
4. **Comprobación de la cookie por parte de CAS**. Comprueba que no existe aún.
5. Solicitud de credenciales de usuario. Se realiza la acción de login correctamente.
6. Establecimiento de Cookie y **redirección a la URL original** pasando como parámetro un ticket http://servidor/aplicacion/recurso?ticket=ST-65-ZSnofRUrSKhBqQ1tPvY3-cas
7. **Intercepción de la solicitud** por parte del control de autenticación. Comprueba que no existe sesión pero si existe ticket.
8. Intercepción de la solicitud por parte del control de validación. Comprueba la existencia de ticket.
9. **Validación del ticket**, inicio de sesión y establecimiento del principal y atributos SAML.
10. Respuesta del recurso solicitado por parte de la aplicación.

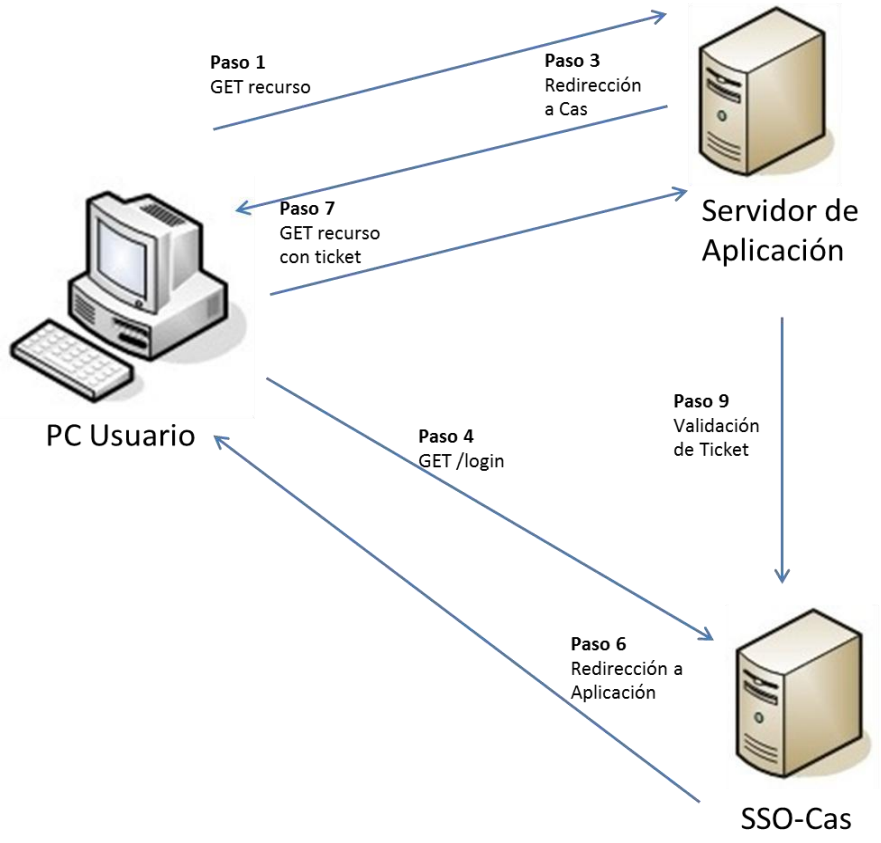


Figura 1. Proceso de autenticación y valicación ticket Cas

1.3 Pase

PASE (Pasarela de Acceso a los Servicios Electrónicos), accesible desde la URL https://pase.carm.es, es un servidor CAS creado para permitir tanto la autenticación de usuarios propios de la CARM como de otros externos, para poder identificarse contra los distintos servicios/aplicaciones disponibles.

Es decir, un usuario de la CARM, por medio de PASE, podrá autenticar contra aplicaciones internas para el desempeño de su propio trabajo, y además autenticar contra otros servicios de la CARM para realizar otras gestiones distintas.

Por lo tanto, se puede afirmar que cualquier ciudadano (bien sea de la propia administración o no), representante de persona física o jurídica, etc. podrán autenticar contra los distintos servicios que se habiliten en la CARM por medio de PASE.

PASE ha sido desarrollado por medio de la aplicación CAS (Enterprise Single Sign-On for All), que es una aplicación universal de código abierto para permitir la autenticación de servicios, por medio de un inicio de sesión único.

El inicio de sesión único aporta la posibilidad de utilizar aplicaciones variadas, realizando la autenticación sólo una vez, manteniendo la identidad del usuario para las todas las aplicaciones y evitando al usuario realizar la acción de autenticación varias veces.

Al contrario que en antiguas versiones de CAS, ahora se proporciona un comportamiento idóneo para realizar la desconexión de sesión de usuario (logout) basado en el concepto de Single Logout (SLO), que permite que la salida de una aplicación no afecte a otras que tenga activas.

Debe considerar que puede configurar su aplicación para que autentique contra PASE de igual forma a como lo realizaría con cualquier otro servidor CAS. En este manual pondremos sobre la mesa algún escenario de configuración de aplicación, pero que puede no coincidir con el que usted necesita, por lo tanto, puede encontrar multitud de documentación adicional en Internet, como por ejemplo las que le indicamos en el apartado de documentación de referencia.

2 2 INTEGRACIÓN DEL SERVICIO EN APLICACIONES

2.1 Métodos de acceso

1. **Credenciales de empleado público de la CARM**

En este caso se puede iniciar la sesión única indicando el "login" de usuario y la clave, de forma similar a la realizada en los fichajes de personal, para que le permita acceder al servicio, que por lo general será una aplicación corporativa de la CARM sobre la que tiene permiso de acceso.

2. Certificado Digital

Pulsando sobre el botón "Acceso con Certificado Digital" se puede acceder al servicio que autentica PASE por medio de un certificado digital válido, que será obtenido y verificado conectando con los servicios de administración electrónica de la CARM.

3. Cl@ve

Pulsando sobre el botón "Acceso con Cl@ve" se puede acceder al servicio que autentica PASE por medio del servicio que ofrece el Ministerio, que incorpora las siguientes opciones:

- 3.1 Certificado /DNIe.
- 3.2 Clave permanente.
- 3.3 Clave pin.

Para más información visita http://clave.gob.es/clave_Home/clave.html

A parte del método de identificación mediante el frontal web, es posible realizar la autenticación mediante un método alternativo, usando servicios REST proporcionados por PASE. Para más información puede acudir al siguiente enlace: https://apereo.github.io/cas/5.1.x/protocol/REST-Protocol.html

2.2 Nivel de seguridad (QAA)

El Esquema Nacional de Seguridad (ENS) determina tres niveles de seguridad en la autenticación: bajo, medio y alto, encuadrándose los métodos indicados arriba en estos de la siguiente forma:

- Bajo, que se corresponde con un QAA (Quality Authentication Assurance) 2.
  - 1, 3.2 sin segunda validación, Medio y Alto
- Medio, que se corresponde con un QAA 3
  - 2, 3.2 con segunda validación, 3.3, 3.1 cuando el certificado es software y Alto
- Alto, que se corresponde con un QAA 4
  - 3.1 cuando se usa certificado hardware

La forma que tiene PASE de determinar qué nivel es el que usa (y por tanto métodos) para presentar al usuario la autenticación es:

- A partir de la url de vuelta que le pasa la aplicación (en la forma "?service=https://aplicacion.carm.es"), ve si está registrada (1) para:
  - Pedir un nivel concreto, en cuyo caso lo usa
  - Pasar como parámetro el nivel a usar, en cuyo caso comprueba si va en la petición, y si está lo usa. La forma de pasárselo es: "?qaa=X", con X entre 2 y 4
- Si no se da ninguno de los casos anteriores, usará el nivel por defecto de la instalación. Actualmente en pruebas y producción intranet hay un qaa 2 y en internet un qaa 3.

(1) El registro se hace para una url concreta (con o sin parámetros), y se interpreta a modo de dominio, teniendo en cuenta si hay excepciones, buscando la url que mas se adapte a la indicada. Ejemplos de forma de uso son los siguientes, teniendo en cuenta, en cada caso, que están registradas las urls 1 y/o 2:

ejemplo	url1 registrada	qaa1		url2 registrada	qaa2		url pasada como parámetro a pase por la aplicación	qaa que pedirá pase
1	https://jad.carm.es	3					https://jad.carm.es	3
2	https://jad.carm.es	3					https://jad.carm.com	no encuentra nada, el que se tenga por defecto
3	https://jad.carm.es	3					https://jad.carm.es/subap	3
4	https://jad.carm.es	3	y	https://jad.carm.es/comint	2		https://jad.carm.es/comint	2
5	https://jad.carm.es	3	y	https://jad.carm.es/comint	2		https://jad.carm.es/comint/subap	2
6	https://jad.carm.es	3	y	https://jad.carm.es/comint	2		https://jad.carm.es/subap	3
7	https://jad.carm.es	3	y	https://jad.carm.es/comint	2		https://jad.carm.es/comint?par=2	2
8	https://jad.carm.es	3	y	https://jad.carm.es/comint?par=3	2		https://jad.carm.es/comint?par=2	3
9	https://jad.carm.es	3	y	https://jad.carm.es/comint?par=3	2		https://jad.carm.es/comint?par=3	2
10	https://jad.carm.es	3	y	https://jad.carm.es/comint?par=3	2		https://jad.carm.es/comint?par=3&par2=4	2
11	https://jad.carm.es	3	y	https://jad.carm.es/comint?par=3	2		https://jad.carm.es/comint?par2=4&par=3	2
12	https://jad.carm.es	3	y	https://jad.carm.es/comint?par=3	2		https://jad.carm.es/subap/comint?par=3	3
13	https://jad.carm.es	3	y	https://jad.carm.es/comint?par=3	2		https://jad.carm.es/	3
14	https://jad.carm.es	3	y	https://jad.carm.es/comint?par=3	2		https://jad.carm.es/comint/subap?par=3	3
15	https://jad.carm.es	3	y	https://jad.carm.es/comint	2		https://jad.carm.es/comint2	3
16	https://jad.carm.es	3	y	https://jad.carm.es/comint	2		https://jad.carm.es/comint/subap?par=4	2
17	https://jad.carm.es	3	y	https://jad.carm.es/comint	2		https://jad2.carm.es/comint/subap?par=4	no encuentra nada, el que se tenga por defecto
18	https://jad.carm.es	3	y	https://jad.carm.es/comint	2		http://jad.carm.es	no encuentra nada, el que se tenga por defecto

2.3 Uso de varios parámetros en url de vuelta

Si se desea volver, tras la invocación a PASE, a una página web usando dos o más parámetros, PASE no podrá identificar qué parámetros pertenecen a esa dirección y cuales debería interpretar él mismo.
Por ejemplo, supongamos la siguiente url:

https://pase.carm.es/pase/login?service=https://jad-inter.carm.es/jAD/PRUBOR/BorradoresPre.xhtml?URLParam.tipdoc=D5&proc=2100&qaa=2

Para pase no es posible saber que el parámetro “proc” pertenece al service mientras el qaa debería interpretarlo. Para ello se debe codificar la url (o al menos el símbolo &) para escapar esos caracteres especiales y que no se interpreten erróneamente.

La URL codificada quedaría de la siguiente manera:

https://pase.carm.es/pase/login?service=https%3A%2F%2Fjad-inter.carm.es%2FjAD%2FPRUBOR%2FBorradoresPre.xhtml%3FURLParam.tipdoc%3DD5%26proc%3D2100

2.4 Instancias de PASE

La aplicación PASE está desplegada para poder autenticar y realizar pruebas sobre distintos entornos. Por lo tanto, se dispone de varias instancias distintas que se detallan a continuación:

** **Entorno de desarrollo:**
https://pase-des.carm.es

Por lo general, esta aplicación está continuamente con cambios y nuevas mejoras, avanzando a las últimas versiones de CAS y corrigiendo errores, por lo que no es aconsejable autenticar contra la misma.

** **Entorno de pruebas:**
https://pase-pru.carm.es

Será una versión medianamente estable sobre la cual se prueban los servicios y aplicaciones de entornos de desarrollo y pruebas.

** **Entorno de producción**
https://pase.carm.es

Es una versión estable de PASE accesible para entornos de producción. Aunque bajo el mismo nombre de dominio, cuando se accede desde dentro de la CARM se accede a dicha instancia y cuando se accede desde fuera de la CARM (Internet) se accede a otra instancia distinta configurada para pedir por defecto un QAA 3, aunque con registro previo, se puede usar un 2.

Si desde dentro de la CARM deseamos autenticar contra el PASE de Internet deberemos referenciarlo bajo el nombre:
https://pase-inter.carm.es

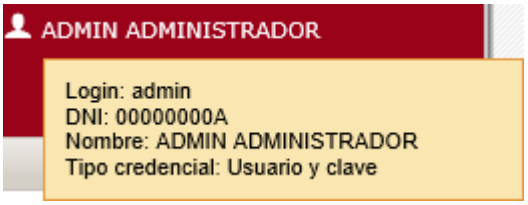
2.5 Atributos de identificación de PASE

Por el hecho de autenticar con PASE, la aplicación dispondrá de varios atributos del usuario conectado. Dichos atributos son proporcionados por medio del protocolo SAML y serán los siguientes:

Nombre del atributo	Descripción
username	Identificador de usuario conectado, que puede corresponderse con el valor de los atributos “logincarm” o “dni”, cuando se trate de personal interna a la CARM o no, respectivamente.
dni	DNI (Documento Nacional de Identidad)
nombre	Nombre
apellido1	Primer apellido (*)
apellido2	Segundo apellido (*)
apellidos	Primer y segundo apellido
tipousuario	Tipo de usuario, que si es una persona que encuentra en la base de datos de seguridad de javato tendrá el valor de “Funcionario”.
correo	Correo electrónico
cif	CIF de la empresa (*)
razonsocial	Nombre de la empresa (*)
logincarm	Login de la CARM
metodoidentificacion	Método de identificación: "Usuario y clave" o "Certificado Digital"
tipocertificado	Indica información extra de la clase de certificado con el que se ha autenticado el usuario, lo que sólo disponible si hemos autenticado con Certificado Digital
citizenQAALevel	Nivel de calidad de la autenticación del ciudadano (Citizen Quality Authentication Assurance Level)

	Indica el nivel de seguridad (QAA) con el que se ha autenticado, según el método usado. Los posibles valores son: **- Nivel 2:** Usuario y contraseña CARM Cl@ve permanente sin segunda validación **- Nivel 3:** Certificado software (*) Cl@ve permanente con segunda validación Cl@ve pin **- Nivel 4:** Certificado hardware (*) Notar que los certificados autenticados con eA serán de nivel 3 ya que no se puede determinar si es software o hardware.

Le mostramos a continuación varios ejemplos de información recuperada al autenticar de diversas formas:



2.6 JAVATO

2.6.1 JAD – DESARROLLO DE APLICACIONES DINÁMICAS

Si utilizamos jAD y jDAD para el desarrollo de aplicaciones, no es necesaria ninguna integración con PASE ya que es proporcionada por defecto, pudiendo acceder a los atributos de autenticación por medio de variables que son proporcionadas.

2.6.2 JSEMILLA

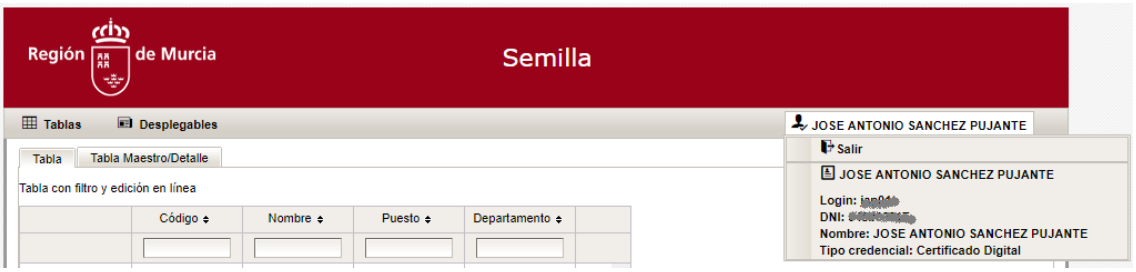
Podemos integrarnos con PASE utilizando como referencia la aplicación Semilla de javato, que utilizando la librería javato-framework 4.0.2 o posterior es capaz de autenticar y recuperar los atributos del usuario conectado.

Para configurar la conectividad con PASE tan sólo es necesario indicar la propiedad "casServerUrlPrefix" del siguiente fichero:
localconfig.properties

```
casServerUrlPrefix=https://pase.carm.es/pase
```

Esta aplicación se puede descargar desde la siguiente URL de Subversión:
https://vcs.carm.es/svn/javato/trunk/jSemilla

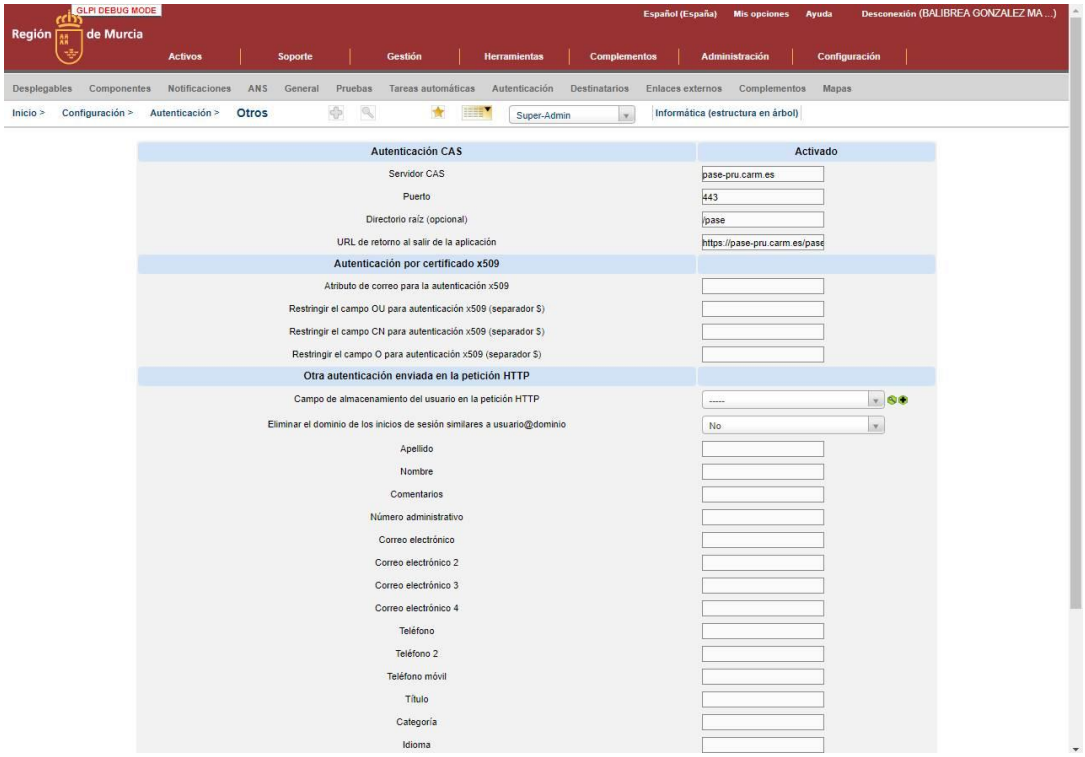
Además, se puede probar la aplicación accediendo a una instancia viva de desarrollo que corre bajo la siguiente URL:
https://jad-des.carm.es/jSemilla



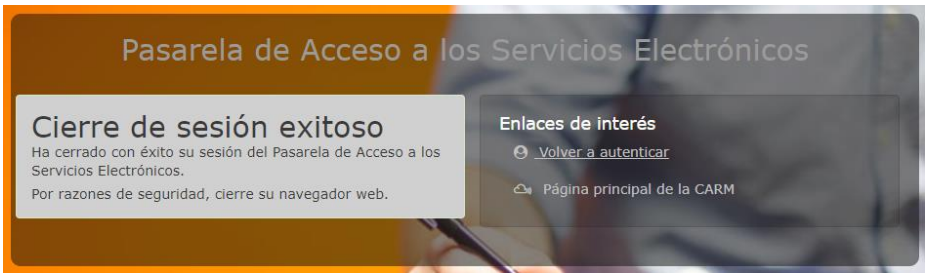
2.7 Aplicación Web Estándar

Nos podemos encontrar con una aplicación web que ya lleve incorporada de forma nativa la configuración contra distintos sistemas de autenticación, como es el caso de **GLPI**, que proporciona una interfaz para poder indicar el método de identificación y los parámetros necesarios para hacerlo efectivo.

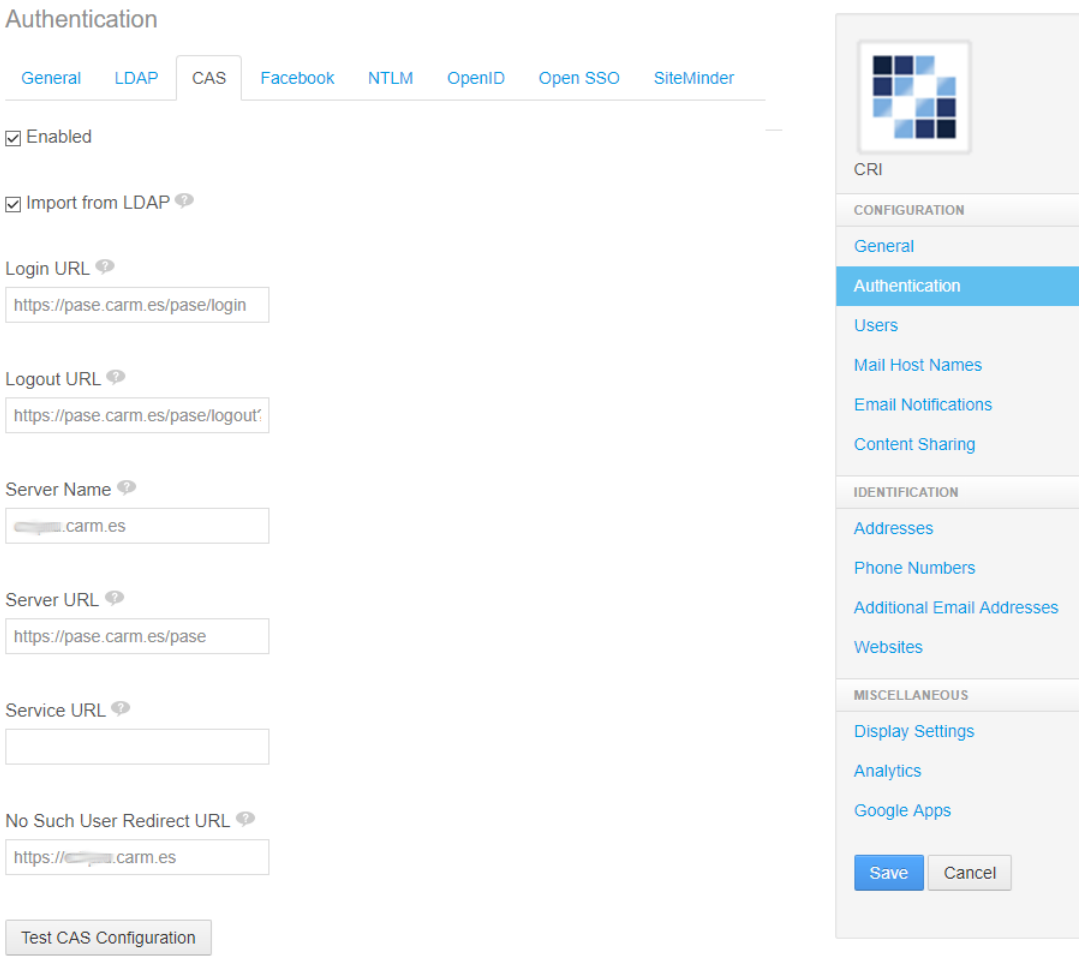
Servidor CAS: pase.carm.es
Puerto: 443
Dirección Raíz: /pase
URL de salida: https://pase.carm.es/pase/logout?service=http://glpi.carm.es



Notar que en la URL de salida se ha indicado "?service=http://glpi.carm.es", que es necesario indicar ya que al hacer logout de la sesión, hay un enlace en PASE para volver a autenticar de forma sencilla sin necesidad de volver a introducir de nuevo la URL del servicio.



Otro ejemplo de aplicación web, como **Liferay**, se configuraría de forma similar a la indicada previamente, pero tan sólo indicando la URL de login y logout, como se muestra en la siguiente pantalla:



2.8 INTEGRACIÓN CON SPRING SECURITY

En este apartado se describe la integración de PASE en la autenticación de una aplicación que hace uso de Spring Security.

Como se ha comentado anteriormente, PASE está basado en JA-SIG Central Authentication Service (CAS), y Spring Security soporta CAS como uno de los mecanismos de autenticación disponibles. Por consiguiente, la tarea de integración de PASE con Spring Security se facilita bastante.

Esta documentación se ha preparado para la versión 4.2.2.RELEASE de Spring Security. Está basada en la documentación "Spring Security Reference" apartado "32. CAS Authentication". Para versiones distintas es posible que sea necesario llevar a cabo ajustes en lo descrito a continuación.

Los pasos a seguir para realizar la integración son:

1. Añadir al fichero `pom.xml` de la aplicación la dependencia `spring-security-cas`.
2. Configurar CAS en el fichero de configuración de Spring Security de la aplicación.
3. (Opcional) Crear subclase para obtener los datos del usuario y su autorización.

1. Añadir al fichero `pom.xml` de la aplicación la dependencia `spring-security-cas`

```
<dependency>
        <groupId>org.springframework.security</groupId>
        <artifactId>spring-security-cas</artifactId>
        <version>${org.springframework-security-version}</version>
</dependency>
```

2. Configurar CAS en el fichero de configuración de Spring Security de la aplicación

Añadir al fichero de configuración de Spring Security de la aplicación (por ejemplo `security-config.xml`), en el caso de disponer de él, o al fichero de configuración general de Spring de la aplicación, la configuración de los beans para CAS:

```
<beans:bean id="serviceProperties"
            class="org.springframework.security.cas.ServiceProperties">
    <beans:property name="service" value="https://servidor/aplicacion/login/cas" />
    <beans:property name="sendRenew" value="false"/>
</beans:bean>

<security:http use-expressions="true" entry-point-ref="casEntryPoint">
    <!-- Aquí la configuración de seguridad propia de la aplicación -->

    <!-- Configuración para CAS -->
    <security:custom-filter position="CAS_FILTER" ref="casFilter" />
    <security:custom-filter ref="requestSingleLogoutFilter" before="LOGOUT_FILTER" />
    <security:custom-filter ref="singleLogoutFilter" before="CAS_FILTER"/>

    <security:logout logout-success-url="/"/>
</security:http>

<beans:bean id="casFilter"
            class="org.springframework.security.cas.web.CasAuthenticationFilter">
    <beans:property name="authenticationManager" ref="authenticationManager"/>
</beans:bean>

<beans:bean id="casEntryPoint"
```

```
            class="org.springframework.security.cas.web.CasAuthenticationEntryPoint">
    <beans:property name="loginUrl" value="https://pase-pru.carm.es/pase"/>
    <beans:property name="serviceProperties" ref="serviceProperties"/>
</beans:bean>

<security:authentication-manager alias="authenticationManager">
    <security:authentication-provider ref="casAuthenticationProvider" />
</security:authentication-manager>

<beans:bean id="casAuthenticationProvider"
class="org.springframework.security.cas.authentication.CasAuthenticationProvider">
    <beans:property name="authenticationUserDetailsService"
                    ref="aplicacionCasAssertionUserDetailsService" />
        <beans:property name="serviceProperties" ref="serviceProperties" />
        <beans:property name="ticketValidator">
        <beans:bean class="org.jasig.cas.client.validation.Cas30ServiceTicketValidator">
            <beans:constructor-arg index="0"
                                   value="https://pase-pru.carm.es/pase" />
        </beans:bean>
    </beans:property>
    <beans:property name="key" value="casAuthenticationProviderKey"/>
</beans:bean>

<!-- This filter handles a Single Logout Request from the CAS Server -->
<beans:bean id="singleLogoutFilter"
            class="org.jasig.cas.client.session.SingleSignOutFilter" />

<!-- This filter redirects to the CAS Server to signal Single Logout should be
performed -->
<beans:bean id="requestSingleLogoutFilter"
            class="org.springframework.security.web.authentication.logout.LogoutFilter">
    <beans:constructor-arg value="https://pase-
pru.carm.es/pase/logout?service=https://servidor/aplicacion/login/cas"/>
    <beans:constructor-arg>
        <beans:bean
class="org.springframework.security.web.authentication.logout.SecurityContextLogoutHandler" />
    </beans:constructor-arg>
    <beans:property name="filterProcessesUrl" value="/logout/cas"/>
</beans:bean>
```

La configuración indicada, además del proceso de login, incluye también el Single Logout de CAS.

3. (Opcional) Crear subclase para obtener los datos del usuario y su autorización

Con gran probabilidad será necesario obtener los datos (atributos) que devuelve PASE del usuario una vez autenticado correctamente. Dichos datos se podrán emplear a su vez para obtener datos propios del usuario en la aplicación o su autorización (perfil de usuario).

Para ello se utiliza el `UserDetailsService` de Spring Security. Se creará una subclase de `org.springframework.security.cas.userdetails.AbstractCasAssertionUserDetailsService` que sobrescribirá el método `loadUserDetails`. Dicho método recibe los datos del usuario de PASE y devuelve un objeto usuario propio de la aplicación que implementará el interface `UserDetails`.

A continuación se muestra un ejemplo de esta implementación (el código está simplificado para facilitar su comprensión):

```
public class AplicacionCasAssertionUserDetailsService extends
                                            AbstractCasAssertionUserDetailsService {
    public static final String PASE_ATRIBUTO_DNI = "dni";

    @Override
    protected UserDetails loadUserDetails(Assertion assertion) {
        AttributePrincipal attributePrincipal = assertion.getPrincipal();

        LOGGER.info("Autenticación en PASE realizada correctamente: "
             + "attributePrincipal.name=" + attributePrincipal.getName()
             + ", attributePrincipal.attributes=" + attributePrincipal.getAttributes());
```

```
        String nif = obtenerAtributoCas(attributePrincipal.getAttributes(),
                                                                    PASE_ATRIBUTO_DNI);
        Usuario usuario = usuarioService.getUsuarioById(nif);

        // Añadir el perfil del usuario a la lista de perfiles
        List<GrantedAuthority> lstPerfiles = new ArrayList<GrantedAuthority>();
        lstPerfiles.add(new SimpleGrantedAuthority("ROLE_" + usuario.getPerfil()));

        // Guardar los atributos del contexto en el usuario
        AplicacionUserDetails aplicacionUserDetails = new AplicacionUserDetails();
        // ... omitido para simplificar
        return aplicacionUserDetails;
    }

    private String obtenerAtributoCas(Map<String, Object> atributosCas,
                                                                    String atributo) {
        return (String) atributosCas.get(atributo);
    }
}

public class AplicacionUserDetails implements UserDetails {
     // ... omitido para simplificar
}
```

2.9 INTEGRACIÓN DE APLICACIÓN JAVA ANTIGUA

En este apartado se indicará cómo configurar una aplicación antigua de java con el sistema de autenticación PASE:

Añadir la siguiente dependencia al **pom.xml**:

```
<dependency>
    <groupId>org.jasig.cas.client</groupId>
    <artifactId>cas-client-core</artifactId>
    <version>3.4.1</version>
</dependency>
```

Modificar **web.xml** añadiendo los siguientes filtros:

```
<filter>
  <filter-name>CAS Authentication Filter</filter-name>
  <filter-
class>org.jasig.cas.client.authentication.AuthenticationFilter</filt
er-class>
  <init-param>
    <param-name>casServerLoginUrl</param-name>
    <param-value>https://pase.carm.es/pase</param-value>
  </init-param>

  <init-param>
    <param-name>serverName</param-name>
    <param-value>URL aplicación (Debe terminar en /)</param-value>
  </init-param>
</filter>

<filter-mapping>
  <filter-name>CAS Authentication Filter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>


<filter>
  <filter-name>CAS Validation Filter</filter-name>
  <filter-
class>org.jasig.cas.client.validation.Cas30ProxyReceivingTicketValid
ationFilter</filter-class>
  <init-param>
    <param-name>casServerUrlPrefix</param-name>
    <param-value>https://pase.carm.es/pase</param-value>
  </init-param>

  <init-param>
    <param-name>serverName</param-name>
    <param-value> URL aplicación (Debe terminar en /)</param-value>
  </init-param>
</filter>

<filter-mapping>
  <filter-name>CAS Validation Filter</filter-name>
  <url-pattern>/*</url-pattern>
```

```
</filter-mapping>


<filter>
  <filter-name>CAS HttpServletRequest Wrapper Filter</filter-name>
  <filter-
class>org.jasig.cas.client.util.HttpServletRequestWrapperFilter</fil
ter-class>
</filter>

<filter-mapping>
  <filter-name>CAS HttpServletRequest Wrapper Filter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

Finalmente modificar el proyecto, quitando la pantalla de login antigua y haciendo en el controlador inicial que haga la autorización use request.getRemoteUser() para recuperar el nombre de usuario autenticado en pase.

3 Documentación de referencia

3.1 Literatura básica

https://www.adictosaltrabajo.com/tutoriales/introduccion-cas
https://es.slideshare.net/ltoledo2014/presentacin-sso-con-cas

3.2 Documentación oficial

* CAS Enterprise Single Sign-On:
https://apereo.github.io/cas/5.1.x/index.html

* Autenticar utilizando servicios REST:
https://apereo.github.io/cas/5.1.x/protocol/REST-Protocol.html

* Desarrollo de aplicaciones Java integradas con CAS:
https://wiki.jasig.org/display/casc/cas+client+for+java+3.1
https://github.com/apereo/java-cas-client

* Desarrollo de aplicaciones PHP integradas con CAS:
https://wiki.jasig.org/display/casc/phpcas

* Usar Spring-security 3 en CAS:
https://docs.spring.io/spring-security/site/docs/3.1.4.RELEASE/reference/cas.html

3.3 Ejemplos variados

* Ejemplo de aplicación java integrada con CAS:
https://www.adictosaltrabajo.com/tutoriales/implementando-ssocas/

* Ejemplo de aplicación php
https://github.com/Jasig/phpCAS/blob/master/docs/examples/example_simple.php

* Ejemplo de uso de los servicios REST desde Java:
https://www.adictosaltrabajo.com/tutoriales/cas-restlet/