Authentification unique avec CAS

Guillaume BOURREAU

AFUP Poitiers - Avril 2019

Sommaire

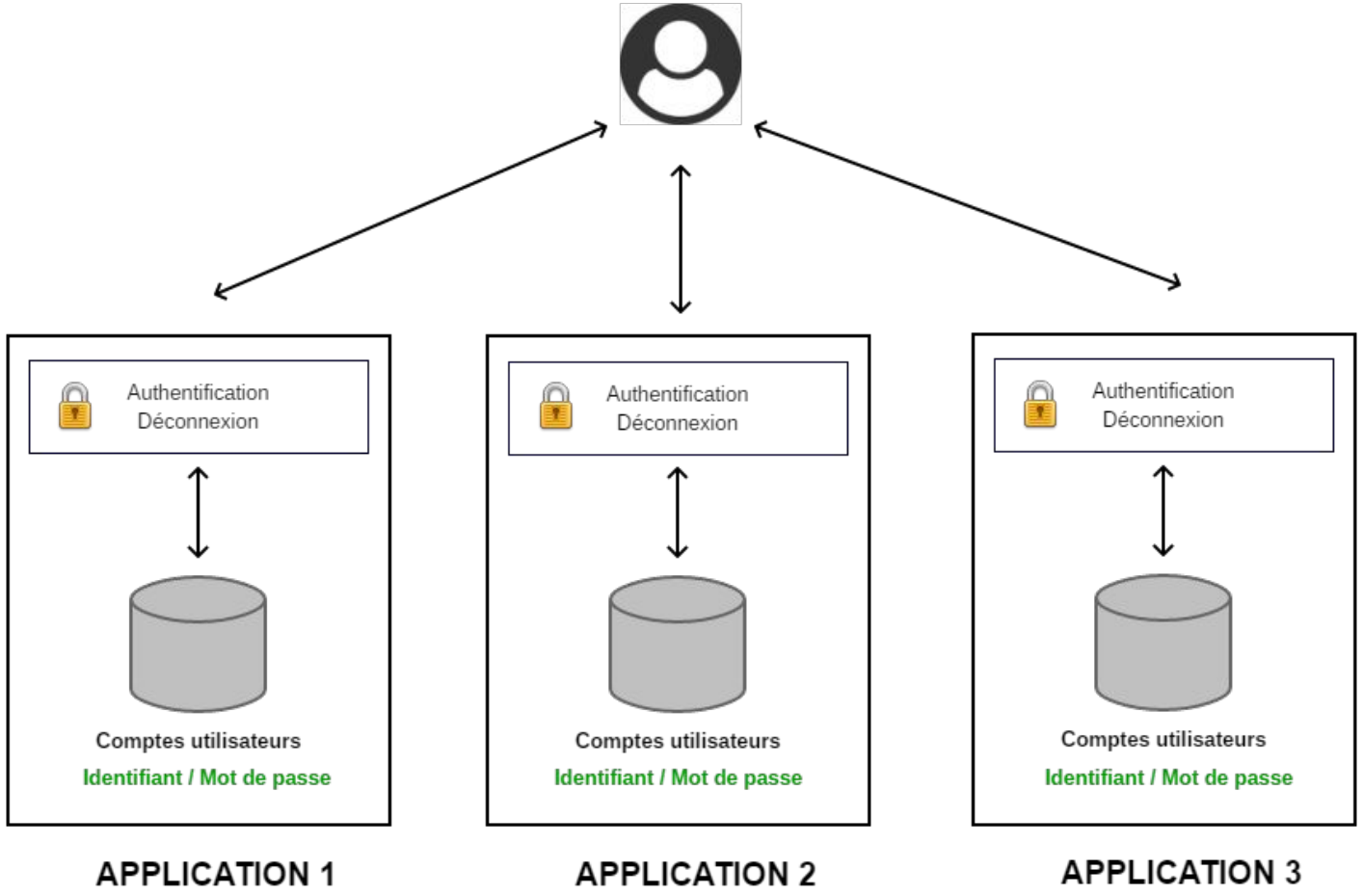
- L’authentification
- Présentation de CAS
- Exemples d’implémentation

L’authentification

Authentification standard

- Principe : tout en local à chaque application
  - Identifiants et mots de passe potentiellement différents pour chaque application
  - Authentification et déconnexion 100% locales à chaque application
  - L'utilisateur doit s'authentifier à chaque application

Authentification standard



Fédération d’identité

- Un premier pas vers le SSO
- **Principe** : utiliser un compte proposé par un **tiers de confiance**
- Pas d’authentification centralisée, uniquement une facilitation de création de compte
- Connecteur à implémenter en local de chaque application

- Exemples de tiers de confiance
  - France Connect
    - Impôts
    - La Poste
    - Ameli
    - Etc.
  - Google (en partie)
  - Facebook (en partie)
  - Etc.

SSO via une application source

- Second pas vers le SSO : SSO généraliste
- Grandes applications :
  - Google
  - Facebook
  - Etc.
- Connecteur(s) à implémenter en local de chaque application : **impact fort**

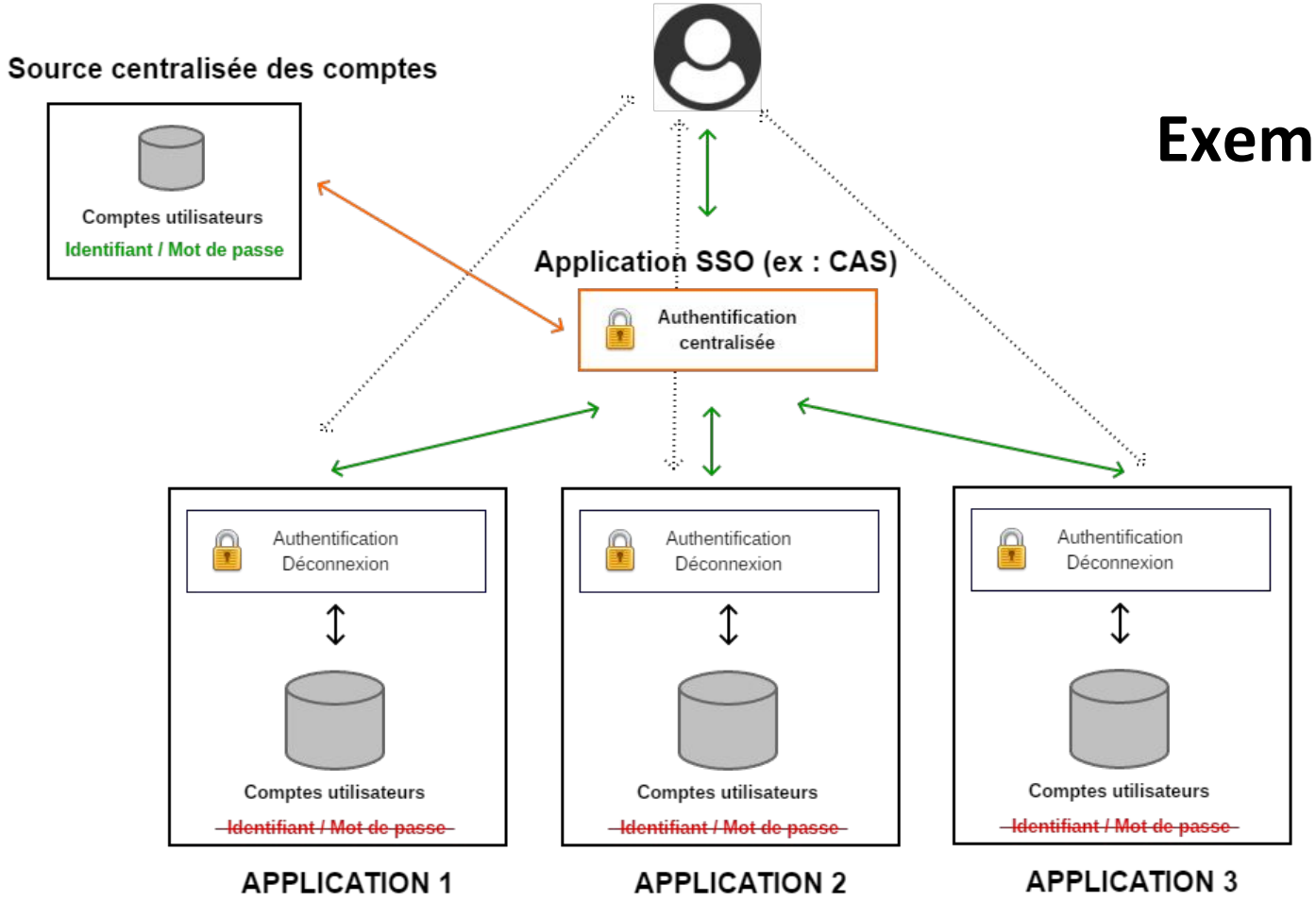
Authentification unique : SSO

- SSO : Single-Sign-On = Authentification unique
- Implémentation avancée du principe de fédération d'identité
  - **Centralisation des comptes utilisateurs** (identifiant et mot de passe)
  - **L'utilisateur s'authentifie une seule fois au départ**, et peut ensuite accéder aux applications compatibles
- Les comptes utilisateurs (identifiant/mot de passe + méthode d'authentification) peuvent être centralisés **n'importe où**
- Connecteur à implémenter en local de chaque application : **impact fort**

Authentification unique : SSO

- Levons le mystère du SSO
  - Rien de magique !
  - Cela **ne remplace pas complètement les systèmes d'authentification et déconnexion** locaux à chaque application
  - Les applications auront toujours besoin de **comptes utilisateurs locaux** pour chaque besoin spécifique
  - **Étape** supplémentaire à implémenter dans les processus existants

Authentification unique : SSO



Exemple standard

Présentation de CAS

Central **A**uthentication **S**ervice

Historique rapide

- Conçu et développé par Shawn Bayern à l'université de Yale
- En 2004, CAS devient un projet de JASIG (groupe universitaire qui promeut la création d'application Java dans l'enseignement)
- En 2008, JASIG est responsable du développement de CAS
- Puis **Apereo Foundation** (regroupement de JASIG et Sakai Foundation) depuis 2012

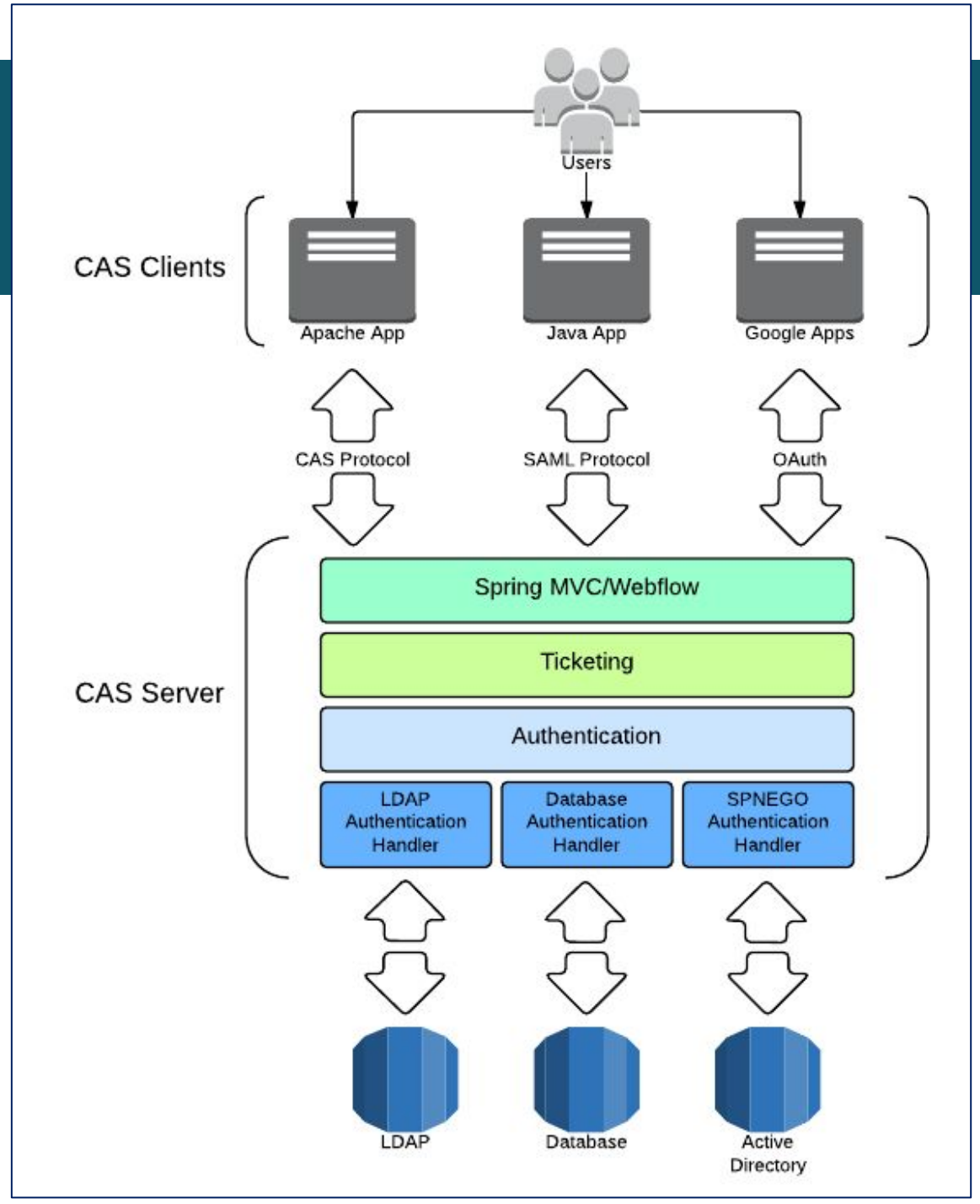
- Version en cours : 6.0.0 de décembre 2018

Technologies

- Application libre 100% open source
- Webapp Java : archive war à déployer dans un Tomcat
- Gestion avec Maven (librairies, création du war)

- Cf lien utile p32 pour toutes les autres technos utilisées

Architecture



Les services

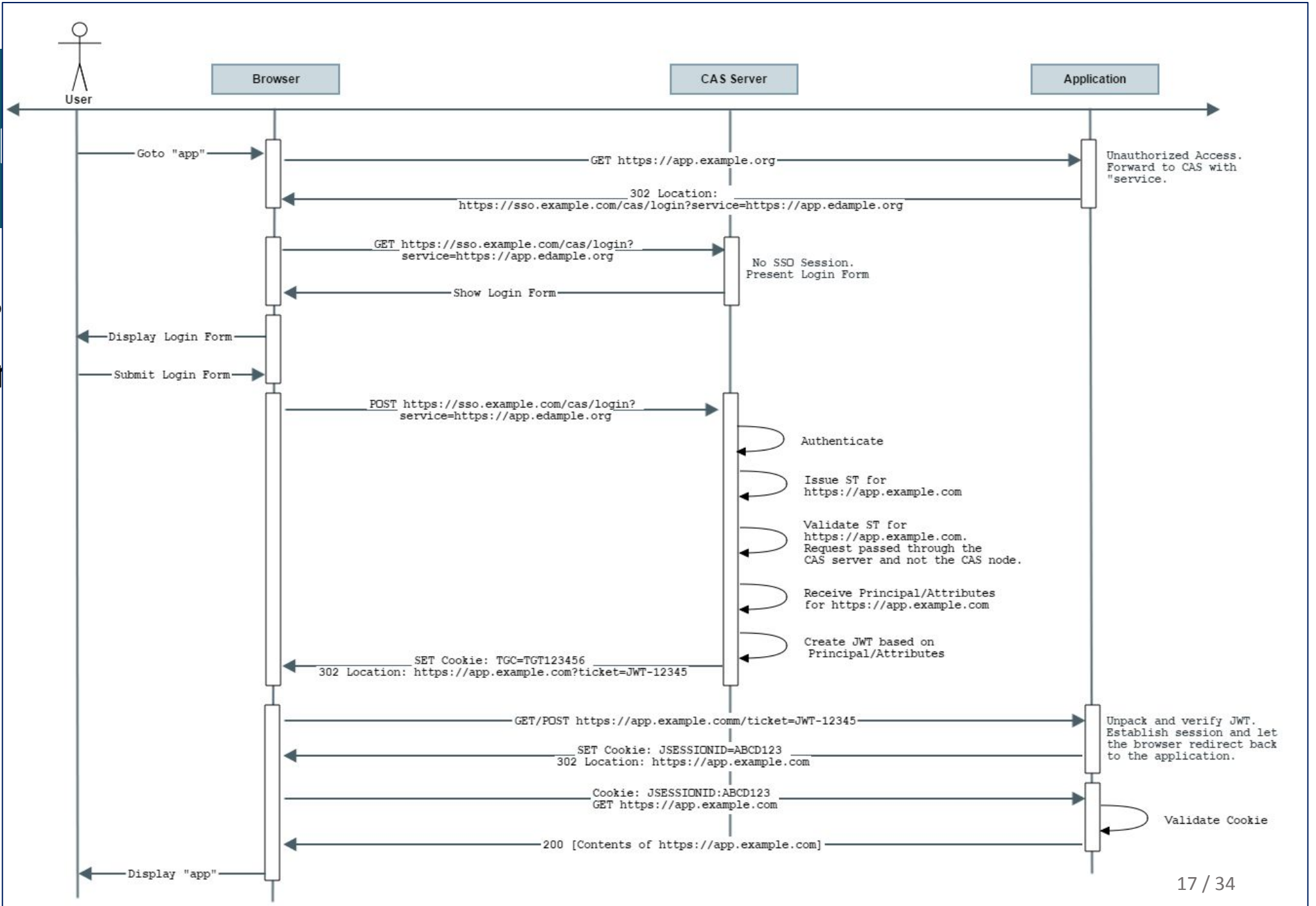
- Un **service** ou **client CAS** est une application, identifiée notamment par son url, qui se connecte à CAS
- Configuration du comportement de CAS
  - Services autorisés
  - Gestion des attributs
  - Proxy
  - Thèmes (pages de connexion et d'erreur)
- Configuration potentiellement spécifique par service

Sources et méthodes d’authentification

- CAS est « branché » sur une (ou plusieurs…) source(s)
- Source = comptes utilisateurs + méthode d’authentification
- Exemples :
  - LDAP / AD
  - Base de données quelconque
  - Authentification Windows
  - Et beaucoup d’autres…

Sché

- **Proces**
- Versio



Attributs

- **Objectif final** : récupérer les informations de l'utilisateur authentifié
- **Pourquoi** ? Pour l'authentifier dans l'application, car l'authentification dans CAS ne suffit évidemment pas
- Attribut par défaut
  - Identifiant (Principal ID)
- Attributs secondaires : spécifiques
  - Email
  - Prénom, nom
  - Autres informations selon le besoin

Thèmes des pages de CAS

- Cas propose un thème par défaut de ses pages
  - Formulaire d'authentification
  - Gestion des erreurs
  - Page de déconnexion (rarement visible)
  - Etc.
- Il est possible de créer son propre thème (HTML/CSS/JS) : **indispensable**

Processus de déconnexion

- On ne se déconnecte pas depuis CAS directement, mais depuis une des applications
- Plusieurs approches possibles
  - Déconnexion de l'application uniquement
  - Déconnexion de l'application + du CAS
  - Déconnexion du CAS seulement
  - Déconnexion du CAS avec déconnexion de toutes les applications (SLO)
- Chaque application doit vérifier régulièrement, ou tout le temps, si l'authentification de CAS est toujours valide

Exemple

- Mini-démo pendant la présentation

Librairies : ne partons pas de zéro !

- Librairies officielles (ou non) disponibles dans toutes les technos : .NET, Java, PHP, Apache
- PHP
  - **phpCAS** (https://github.com/apereo/phpCAS)
  - Symfony **CasAuthBundle** : (https://github.com/PRayno/CasAuthBundle)
- Apache
  - **mod_auth_cas** (https://github.com/apereo/mod_auth_cas)
  - Autres modules en PERL

Étapes d'implémentation

- Installation et configuration de CAS
- Conception complète
  - Identification de la source des comptes (ou des sources)
  - Gestion des droits (centralisés ou non) : attributs
  - Durée des sessions
  - Etc.
- Pour chaque application (service)
  - Revue du système d'authentification : simple configuration, surcharge via un client CAS, modification du kernel (hum…)
  - Revue du système de déconnexion

Autres solutions de SSO

- Fournisseurs de SSO Cloud
  - AuthAnvil
  - oneLogin
  - Okta
  - Etc.
- SSO en entreprise
  - Evidian
  - Synetis
  - Enovacom
  - Etc.

Exemples d’implémentation

Basés sur CAS 3.5.2

Configuration des services autorisés

```
<bean id="serviceRegistryDao" class="org.jasig.cas.services.InMemoryServiceRegistryDaoImpl">
    <property name="registeredServices">
            <list>
                <bean class="org.jasig.cas.services.RegexRegisteredService">
                    <property name="id" value="0" />
                    <property name="name" value="HTTP and IMAP" />
                    <property name="description" value="Allows HTTP(S) and IMAP(S) protocols" />
                    <property name="serviceId" value="^(https?|imaps?)://.*" />
                    <property name="evaluationOrder" value="10000001" />
                </bean>
            </list>
    </property>
</bean>
```

Source LDAP

```
<property name="authenticationHandlers">
    <list>
        <bean class="org.jasig.cas.authentication.handler.support.HttpBasedServiceCredentialsAuthenticationHandler"
            p:httpClient-ref="httpClient" />

        <bean class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
            <property name="filter" value="%u@sdis86.net" />
            <property name="contextSource" ref="contextSource" />
            <property name="ignorePartialResultException" value="yes" />
        </bean>
    </list>
</property>
```

```
<bean id="contextSource" class="org.springframework.ldap.core.support.LdapContextSource">
      <property name="anonymousReadOnly" value="false" />
      <property name="pooled" value="false" />
      <property name="url" value="ldap://192.168.XXX.XXX/" />
      <property name="userDn" value="CN=XXXX,OU=SDIS86,DC=sdis86,DC=net"/>
      <property name="password" value="XXXXXX"/>
      <property name="baseEnvironmentProperties">
            <map>
                  <entry key="com.sun.jndi.ldap.connect.timeout" value="3000" />
                  <entry key="com.sun.jndi.ldap.read.timeout" value="3000" />
                  <entry key="java.naming.security.authentication" value="simple" />
            </map>
      </property>
  </bean>
```

Source spécifique : base de données

```
<property name="authenticationHandlers">
    <list>
        <bean
            class="org.jasig.cas.authentication.handler.support.HttpBasedServiceCredentialsAuthenticationHandler"
            p:httpClient-ref="httpClient" />

        <bean
            class="com.ayaline.cas.adaptors.ez.EzAuthenticationHandler">
            <property name="sql">
                <value>
                    SELECT u.login, u.password_hash
                    FROM ezuser u, ezuser_setting us
                    WHERE u.contentobject_id = us.user_id
                    AND us.is_enabled = 1
                    AND ((lower(u.login) = lower(?)) OR (lower(u.email) = lower(?)))
                </value>
            </property>
            <property name="dataSource" ref="ezDataSource" />
            <property name="passwordEncoder" ref="ezPasswordEncoder"></property>
        </bean>
    </list>
</property>
```

Attributs : autres que le principal

```xml
<property name="resultAttributeMapping">
  <map>
    <!-- Mapping beetween LDAP entry attributes (key) and Principal's (value) -->
    <entry value="lastname" key="sn" />
    <entry value="firstname" key="givenName" />
    <entry value="sdis86_uid" key="pager" />
    <entry value="mail" key="userPrincipalName" />
  </map>
</property>
```

```xml
<bean id="serviceRegistryDao" class="org.jasig.cas.services.InMemoryServiceRegistryDaoImpl">
    <property name="registeredServices">
        <list>
            <bean class="org.jasig.cas.services.RegexRegisteredService">
                <property name="id" value="0" />
                <property name="name" value="HTTP and IMAP" />
                <property name="description" value="Allows HTTP(S) and IMAP(S) protocols" />
                <property name="serviceId" value="^(https?|imaps?)://.*" />
                <property name="evaluationOrder" value="10000001" />
                <property name="allowedAttributes">
                    <list>
                        <value>firstname</value>
                        <value>lastname</value>
                        <value>sdis86_uid</value>
                        <value>mail</value>
                    </list>
                </property>
            </bean>
        </list>
    </property>
</bean>
```

SPNEGO

- Authentification basée sur la session Windows et le contrôleur de domaine (authentification Kerberos)
- Navigateur à configurer pour permettre un échange d'informations concernant la session Windows entre le navigateur et CAS
- Librairie à ajouter dans CAS avec Maven

```
<!-- Configuration SPNEGO -->
<bean name="jcifsConfig" class="org.jasig.cas.support.spnego.authentication.handler.support.JCIFSConfig">
    <property name="jcifsServicePrincipal" value="${kerberos.user}" />
    <property name="kerberosConf" value="${kerberos.conf}" />
    <property name="kerberosDebug" value="false" />
    <property name="kerberosRealm" value="${kerberos.domain}" />
    <property name="kerberosKdc" value="${kerberos.kdc}" />
    <property name="loginConf" value="${path.loginConf}" />
</bean>
```

SPNEGO

```
<property name="authenticationHandlers">
    <list>
        <!-- Test 1 : SPNEGO -->
        <bean class="org.jasig.cas.support.spnego.authentication.handler.support.JCIFSSpnegoAuthenticationHandler">
            <property name="authentication">
                <bean class="jcifs.spnego.Authentication" />
            </property>
            <property name="principalWithDomainName" value="false" />
            <property name="NTLMallowed" value="false"/>
        </bean>

        <!-- Test 2 : Base de données -->
        <bean
            class="com.ayaline.cas.adaptors.ez.EzAuthenticationHandler">
            <property name="sql">
                <value>
                    SELECT u.login, u.password_hash
                    FROM ezuser u, ezuser_setting us
                    WHERE u.contentobject_id = us.user_id
                    AND us.is_enabled = 1
                    AND ((lower(u.login) = lower(?)) OR (lower(u.email) = lower(?)))
                </value>
            </property>
            <property name="dataSource" ref="ezDataSource" />
            <property name="passwordEncoder" ref="ezPasswordEncoder"></property>
        </bean>
    </list>
</property>
```

Bibliographie

Bibliographie

- Wikipédia : https://en.wikipedia.org/wiki/Central_Authentication_Service
- Sources CAS : https://github.com/apereo/cas
- Documentation CAS : https://apereo.github.io/cas
- Installation de CAS (6.0.x) : https://apereo.github.io/cas/6.0.x/installation/Configuring-Authentication-Components.html
- Source p24 : https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1203483-sso-single-sign-on-definition-traduction-et-acteurs

Bon courage !

Guillaume BOURREAU

AFUP Poitiers - Avril 2019