SG/SPSSI/CPII
DOSO/PNE Sécurité

RAPPORT

26/08/2013

Intégration Cerbère dans les applications

Utilisation du protocole CAS dans Cerbère

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE
DE L'ÉGALITÉ DES TERRITOIRES
ET DU LOGEMENT
www.territoires.gouv.fr

MINISTÈRE DE L'ÉCOLOGIE,
DU DÉVELOPPEMENT DURABLE
ET DE L'ÉNERGIE
www.developpement-durable.gouv.fr

Historique des versions du document

Version	Date	Commentaire
	26/08/2013	Spécifications SAML 1.1.
	21/03/2013	Compléments.
	19/03/2007	Version initiale.

Affaire suivie par

Erwan SALMON - SG/SPSSI/CPII – PNE Sécurité
Tél. : 05 56 70 65 70
Courriel : Erwan.Salmon@developpement-durable.gouv.fr

Rédacteur

Erwan SALMON – SG/SPSSI/CPII – PNE Sécurité

Relecteurs

SOMMAIRE

1 - Présentation générale

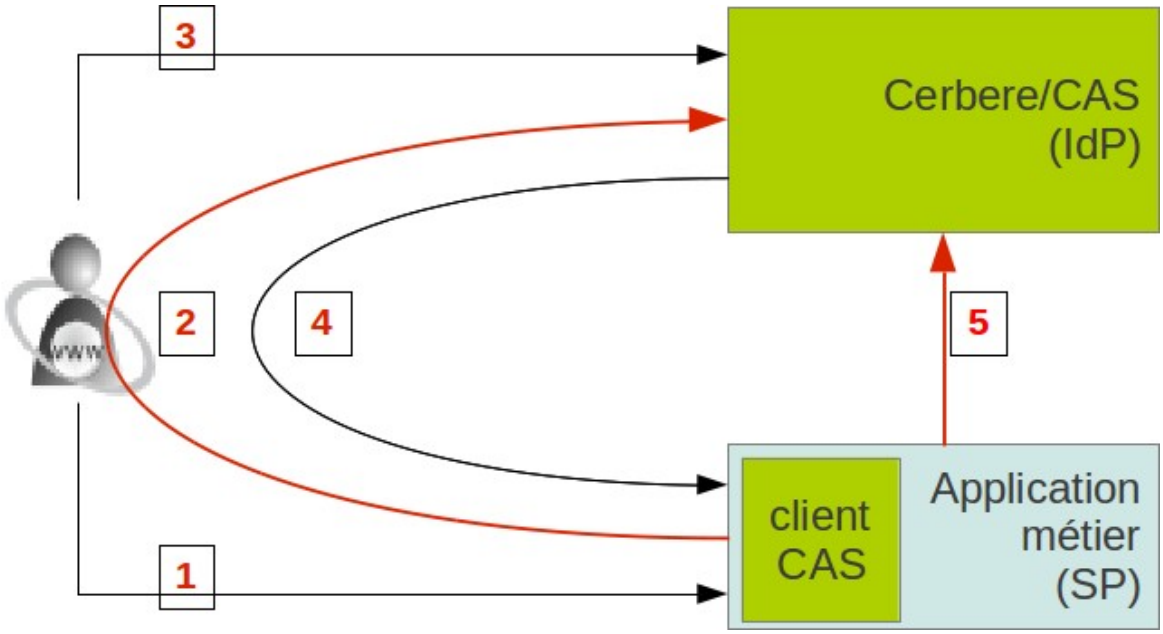
CAS est un système d'authentification unique libre initié par l'université de Yal et intégré dans de nombreux systèmes. Le projet CAS repose sur des protocoles normalisés "de fait" (CAS 1.0 et 2.0) et sur une norme (SAML 1.1).
Pour plus d'informations sur CAS, se référer au site du projet : http://www.jasig.org/cas

Cerbère est le système d'authentification unique des ministères du Logement (METL) et de l'Écologie (MEDDE). Cerbère s'appuie également sur les normes SAML et CAS.

Cette note explique comment intégrer Cerbère dans une application à l'aide du protocole CAS et des clients CAS standards.

Cerbère et CAS assurent l'authentification de leurs utilisateurs selon une même cinématique, basée sur un portail d'authentification centralisé :

1. Demande d'accès à une application métier (*SP* ou *Service Provider* selon la terminologie CAS).
2. **Demande d'authentification CAS** : redirection vers le service d'authentification Cerbère/CAS (*IdP* ou *Identity Provider* selon la terminologie CAS).
3. Authentification sur le portail Cerbère/CAS et génération d'un jeton d'authentification à usage unique.
4. Redirection vers l'application métier (*SP*) avec jeton d'authentification à usage unique.
5. **Validation CAS** : validation du jeton d'authentification par l'application métier (SP).



Le protocole CAS définit :

- La demande d'authentification (étape 2).
- Le format du jeton d'authentification (étape 3)
- La demande de validation (étape 5).

Le serveur Cerbère/CAS a en charge l'authentification de l'utilisateur et la transmission d'informations sur son identité et ses droits.

Le client CAS a en charge le contrôle des accès, le contrôle du jeton d'authentification Cerbère et la transmission à l'application des informations issues de Cerbère.

Le contenu de la réponse Cerbère/CAS dépend du protocole CAS choisi :

- CAS 2.0 : seul l'identifiant utilisateur (**netid**) est retourné.

- SAML 1.1 : les assertions SAML contiennent l'**identité détaillée** de l'utilisateur (nom, prénom, courriel, organisation, …) ainsi que ses **droits** sur l'application.

Cette réponse est détaillée dans la suite du document.

2 - Les applications Cerbère

L'authentification Cerbère repose sur une notion d'application. Cerbère propose deux types d'application :

- des applications génériques conçues pour la seule authentification, sans habilitation préalable des utilisateurs.
- Des applications dédiées conçues pour la gestion centralisée de droits dans Cerbère.

2.1 - Les applications Cerbère génériques

Les applications Cerbère génériques permettent permettent d'authentifier un utilisateur sans avoir à lui attribuer de profil Cerbère.
Ces applications permettent d'obtenir l'identifiant utilisateur (CAS 2.0) ou des informations détaillées sur son compte (SAML 1.1). Elle ne transmettent pas de profils d'application.

Les applications Cerbère génériques doivent être utilisées lorsque les droits utilisateurs sont entièrement gérés par l'application métier. Cerbère/CAS n'assure alors que l'authentification utilisateur.

Cerbère propose deux applications d'authentification génériques, l'une réservée aux comptes certifiés, l'autre ouverte à tous les comptes.

Les comptes certifiés sont :

- Tous les comptes issus de l'annuaire Amande (comptes METL-MEDDE, DDI, Préfecture, DRAAF).
- Tous les comptes externes créés manuellement par un administrateur Cerbère.
- Les comptes de professionnels créés par certificat.
- Les comptes de particuliers et de professionnels créés sans certificat et certifiés par la suite par un administrateur Cerbère.

Ces comptes sont dit certifiés car les informations qu'ils portent (nom, prénom, organisation, ...) ont été validées par une tierce personne (administrateur Amédée, administrateur Cerbère, gestionnaire Fimad, organisme de délivrance de certificat, ..).

L'application CAS générique réservée aux comptes certifiés est appelée dans la suite du document "**CAS certifiée**".
L'application CAS générique ouverte à tous les comptes est appelée dans la suite du document "**CAS publique**".

L'authentification CAS publique est conçue pour les systèmes d'informations gérant en interne l'inscription de leurs utilisateurs ou pour lesquels les informations d'identité des utilisateurs sont sans importance (courriel hormis).

2.2 - Les applications Cerbère dédiées

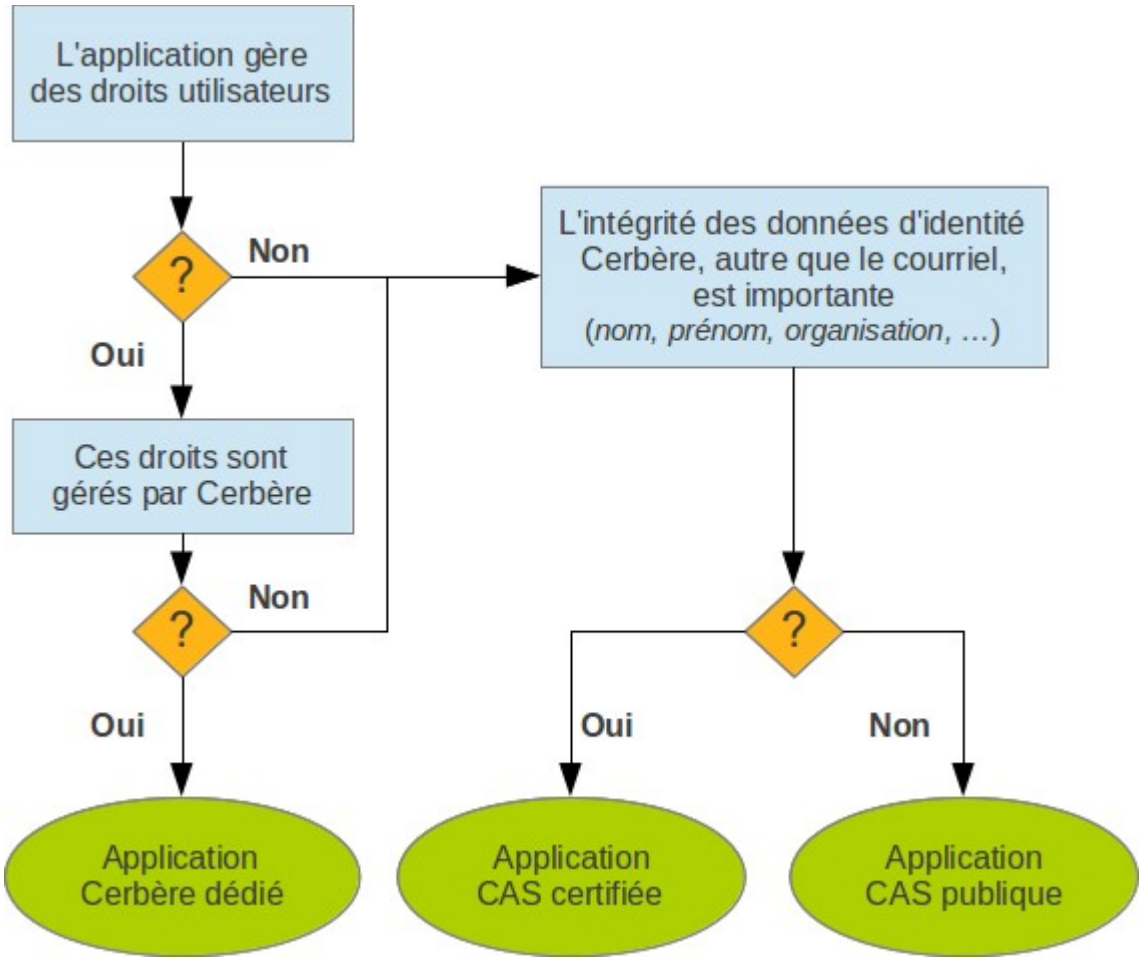
Définir une application dans Cerbère permet de lui associer un ensemble de profils et d'habiliter des comptes sur cette application.
L'authentification sur une application Cerbère dédiée permet d'obtenir l'identifiant utilisateur (CAS 2.0) ou des informations détaillées sur son compte (SAML 1.1) ainsi que les autorisations de l'utilisateur sur cette application (SAML 1.1)

Il est nécessaire de déclarer une application dans Cerbère dès lors que droits des utilisateurs sont gérés dans Cerbère.

Une application dédiée est représentée dans Cerbère par un **identifiant d'application** (numérique).

2.3 - Synthèse des applications Cerbère

Le synoptique suivant permet de déterminer quelle application Cerbère/CAS répond aux besoins d'un système d'information.



3 - Spécifications de l'interface Cerbère/CAS

Le protocole CAS est documenté sur le site du projet CAS :
http://www.jasig.org/cas/protocol

3.1 - Les adresses du serveur Cerbère/CAS

Les actions CAS sont décrites sous forme d'URL normalisées. Ces URLS sont de la forme :

[CAS_URL_BASE]/*action*

Où :

- [CAS_URL_BASE] représente l'adresse de base de l'application d'authentification Cerbère.
- *action* , variable, dépend de l'action CAS demandée.

3.1.1 - Adresse de base Cerbère ([CAS_URL_BASE])

L'adresse de base Cerbère dépend de l'application d'authentification utilisée et du réseau depuis lequel se fait l'authentification (Morea, Internet, Ader).

• **Adresse de base Cerbère [CAS_URL_BASE] sur le réseau intranet Morea**

Pour l'application CAS publique :
https://authentification-cerbere.application.i2/cas/public
Pour l'application CAS certifiée:
https://authentification-cerbere.application.i2/cas
Pour une application dédiée d'identifiant *N* :
https://authentification-cerbere.application.i2/cas/*N*

• **Adresse de base Cerbère [CAS_URL_BASE] sur le réseau internet**

Pour l'application CAS publique :
https://authentification.application.developpement-durable.gouv.fr/cas/public
Pour l'application CAS certifiée:
https://authentification.application.developpement-durable.gouv.fr cas
Pour une application dédiée d'identifiant *N* :
https://authentification.application.developpement-durable.gouv.fr/cas/*N*

• **Adresse de base Cerbère [CAS_URL_BASE] sur le réseau Ader**

Pour l'application CAS publique :
https://authentification.application.developpement-durable.ader.gouv.fr/cas/public
Pour l'application CAS certifiée:
https://authentification.application.developpement-durable.ader.gouv.fr/cas
Pour une application dédiée d'identifiant *N* :
https://authentification.application.developpement-durable.ader.gouv.fr/cas/*N*

3.1.2 - Adresse d'authentification

L'adresse d'authentification Cerbère/CAS, est :
[CAS_URL_BASE]**/login**

Cette adresse **doit** contenir un argument nommé "**service**" indiquant l'URL de retour vers l'application métier.

Exemple : L'adresse d'authentification sur le site Giseh http://pne.metier.i2 est :
[CAS_URL_BASE]/login?service=http://pne.metier.i2

3.1.3 - Adresse de validation

La validation d'un jeton d'authentification Cerbère/CAS permet de valider l'authentification d'un utilisateur et de connaître son identité.

• **Adresse de validation CAS 2.0**

L'adresse de validation Cerbère/CAS en protocole CAS 2.0 est :
[CAS_URL_BASE]**/serviceValidate**

Remarque : l'adresse [CAS_URL_BASE]/proxyValidate peut aussi est utilisée.

• **Adresse de validation SAML 1.1**

L'adresse de validation Cerbère/CAS en protocole SAML 1.1 est :
[CAS_URL_BASE]**/samlValidate**

Remarque : Une demande de validation SAML 1.1 se fait par appel à un service web SOAP en mode **POST**.

3.1.4 - Adresse de déconnexion

L'adresse de déconnexion Cerbère/CAS, est :

[CAS_URL_BASE]**/logout**

Cette adresse peut contenir un argument nommé "**url**" indiquant l'URL de retour vers l'application métier.

Exemple : L'adresse de déconnexion sur le site Giseh http://pne.metier.i2 est :
[CAS_URL_BASE]/logout?url=http://pne.metier.i2

3.2 - Format des réponses Cerbère/CAS

La réponse Cerbère/CAS dépend du protocole de validation utilisé :

- CAS 2.0 : la réponse CAS ne contient que l'identifiant utilisateur.
- SAML 1.1 : la réponse CAS contient des informations détaillées sur l'utilisateur, ainsi que ses profils d'application s'il s'agit d'une application Cerbère dédiée.

3.2.1 - Réponse simple CAS 2.0, identifiant seul

En protocole CAS 2.0, seul l'identifiant utilisateur (*netid* dans la terminologie CAS) est retourné à l'application.

Par convention, l'identifiant utilisateur (**netid** CAS) retourné par Cerbère est le **courriel**.

Exemple de réponse CAS 2.0 sur une authentification réussie d'une personne dont le courriel est "sebastien.martin@developpement-durable.gouv.fr" :

```
<cas:serviceResponse xmlns:cas="http://www.yale.edu/tp/cas">
  <cas:authenticationSuccess>
    <cas:user>sebastien.martin@developpement-durable.gouv.fr</cas:user>
  </cas:authenticationSuccess>
</cas:serviceResponse>
```

3.2.2 - Réponse détaillée SAML 1.1

Une requête de validation CAS/SAML 1.1 permet d'obtenir des informations détaillées sur l'utilisateur courant.

La réponse Cerbère contient des informations sur le compte utilisateur. Elle contient également les profils Cerbère sur l'application métier si celle-ci fait l'objet d'une déclaration dans Cerbère.

Une réponse SAML 1.1 **simplifiée** typique d'une authentification réussie :

```
<Response>
  <Status>
    <StatusCode Value="Success"/>
  </Status>
  <Assertion>
    <AttributeStatement>
      <Subject>
        <NameIdentifier>ID_Cerbere</NameIdentifier>
      <Subject>
      <Attribute AttributeName="NOM_ATTRIBUT_1">
        <AttributeValue>VALEUR_ATTRIBUT_2</AttributeValue>
      </Attribute>
      <Attribute AttributeName="NOM_ATTRIBUT_2">
        <AttributeValue>VALEUR_ATTRIBUT_2</AttributeValue>
      </Attribute>
      <!-- ... Autres attributs -->
      <Attribute AttributeName="NOM_ATTRIBUT_N">
```

```
        <AttributeValue>VALEUR_ATTRIBUT_N</AttributeValue>
      </Attribute>
    </AttributeStatement>
  </Assertion>
</Response>
```

Les informations utilisateurs et les profils sur l'application, détaillés ci-après, sont contenus dans les attributs de l'assertion d'authentification SAML. Ces attributs sont analysés et mis à disposition par le client CAS.

Tout attribut non documenté ci-dessous et présent dans la requête SAML est susceptible d'évoluer, il ne doit pas être utilisé dans les applications.
Ces attributs supplémentaires, non documentés, existent à des seules fins techniques ou de compatibilité ascendante.

3.2.2.a - Attributs d'identité

Les attributs d'identité renvoyés par Cerbère sont les suivants.

Attributs d'identité	
Nom attribut	**Description**
UTILISATEUR.ID	Identifiant interne Cerbère (numérique).
UTILISATEUR.NOM	Nom.
UTILISATEUR.PRENOM	Prénom.
UTILISATEUR.CIVILITE	Civilité. Les valeurs possibles sont : • 'M' pour un homme • 'F' pour une femme • Vide si non renseigné dans Cerbère.
UTILISATEUR.MEL	Courriel.
UTILISATEUR.MATRICULE	Matricule de ressources humaine ("code RH"), vide si non présent dans Cerbère.
UTILISATEUR.TEL_FIXE	Numéro de téléphone fixe, vide si non présent dans Cerbère.
UTILISATEUR.TEL_MOBILE	Numéro de téléphone mobile, vide si non présent dans Cerbère.
UTILISATEUR.FAX	Numéro de fax, vide si non présent dans Cerbère.
UTILISATEUR.ADR_RUE	Composante "rue" de l'adresse postale, vide si non présente dans Cerbère.
UTILISATEUR.ADR_VILLE	Composante "ville" de l'adresse postale, vide si non présente dans Cerbère.
UTILISATEUR.ADR_CODEPOSTAL	Composante "code postal" de l'adresse postale, vide si non présente dans Cerbère.
UTILISATEUR.ADR_PAYS_CODE	Code pays 2 lettres (ISO 3166-1 A2) du pays de la personne, vide si non présent dans Cerbère .
UTILISATEUR.ADR_PAYS_NOM	Nom du pays de la personne, vide si non présent dans Cerbère.
UTILISATEUR.CERTIFICAT	Si la personne s'est authentifié par certificat, cet attribut contient le certificat X509 employé. Il est vide sinon.
UTILISATEUR.EST_VERIFIE	Vaut 1 s'il s'agit d'un compte certifié, 0 sinon.
UTILISATEUR.UNITE	Unité complète de la personne ("Service/Unite/Sous-Unite/...") .

	Exemple : SG/SPSSI/CPII/DOSO/ET

3.2.2.b - Attributs d'application

Les attributs d'application renvoyés par Cerbère sont les suivants.

Attributs d'application	
Nom attribut	**Description**
APPLICATION.NOM	Nom de l'application d'authentification utilisée.
APPLICATION.NIVEAU_AUTHENTIFICATION	Niveau d'authentification minimum requis par l'application : mot de passe (=0) , certificat 1* (=1), 2* (=2), 3* (=3).
APPLICATION.EST_SSO	Vaut 1 si l'application accepte les authentifications uniques entre applications, 0 sinon.

3.2.2.c - Attributs d'entreprise

S'il s'agit d'un compte de professionnel externe, les assertions SAML contiennent des attributs d'identité de l'entreprise. Sinon tous ces attributs sont vides.

Attributs d'entreprise (*compte de professionnel seul*)	
Nom attribut	**Description**
ENTREPRISE.SIREN	Numéro SIREN de l'entreprise
ENTREPRISE.RAISON_SOCIALE	Raison sociale de l'entreprise.
ENTREPRISE.ADR_RUE	Composante "rue" de l'adresse postale, vide si non présente dans Cerbère.
ENTREPRISE.ADR_VILLE	Composante "ville" de l'adresse postale, vide si non présente dans Cerbère.
ENTREPRISE.ADR_CODEPOSTAL	Composante "code postal" de l'adresse postale, vide si non présente dans Cerbère.
ENTREPRISE.ADR_PAYS_CODE	Code pays 2 lettres (ISO 3166-1 A2) du pays de l'entreprise, vide si non présent dans Cerbère .
ENTREPRISE.ADR_PAYS_NOM	Nom du pays de l'entreprise, vide si non présent dans Cerbère.

3.2.2.d - Gestion des profils

Si l'authentification s'est faite sur une application Cerbère dédiée, alors les attributs contiennent également les profils du compte.

Ces profils sont définis dans l'attribut multi-valué **AUTORISATION.PROFILS**. Chaque valeur correspond à un profil distinct.

Chaque profil est défini par un triplet {nom du profil, portée, restriction}. Ce triplet est encodé sous forme d'une chaîne de caractère, préfixée de "PROFIL=", dont les valeurs sont séparées par un point-virgule.
Les profils doivent être analysé selon la syntaxe suivante :

PROFIL=*NomProfil*;*Portée*;*Restriction*

Les parties soulignées sont variables, les autres fixes.
Une restriction absente est représentée par la valeur particulière "none".

Exemple : Une personne possède deux profils :

- Un profil ADMINISTRATEUR sur la portée DREAL Aquitaine, avec une restriction de valeur R01.
- Un profil CONSULTATION sur la portée DREAL Aquitaine, sans restriction.

L'attribut AUTORISATION.PROFILS contiendra deux valeurs :
- PROFIL=ADMINISTRATEUR;DREAL Aquitaine;R01
- PROFIL=CONSULTATION;DREAL Aquitaine;none

L'attribut SAML associé :
```
<Attribute AttributeName="AUTORISATION.PROFILS">
<:AttributeValue>PROFIL=ADMINISTRATEUR;DREAL
Aquitaine;R01<AttributeValue>
<AttributeValue>PROFIL=CONSULTATION;DREAL Aquitaine;none<AttributeValue>
</Attribute>
```

4 - Mise en œuvre de Cerbère/CAS

4.1 - Applications Java

Le client CAS Java est fourni sur le site du projet CAS, en version 3.2.1 à la date d'écriture de ce document.

Le dossier "outils/" contient des exemples d'intégration Cerbère/CAS en Java.

4.1.1 - Configuration du serveur d'application Java

Le serveur d'application Java doit valider le certificat du serveur d'authentification Cerbère. Il est nécessaire de lui fournir le certificat de l'autorité racine IGC/A dans son "truststore". Un "truststore" java contenant ce certificat est fourni en annexe (trustore-igca-1.0-20021213.ks), son mot de passe est "cerbere".

Pour que le serveur d'application prenne en compte ce "truststore", il faut le lui indiquer au démarrage par les options Java suivantes (à placer dans le script de démarrage du serveur d'application) :
- javax.net.ssl.trustStore : chemin complet du fichier trustore-igca-1.0-20021213.ks;
- javax.net.ssl.trustStorePassword : mot de passe du truststore ("cerbere").

4.1.2 - Installation du client java CAS

L'installation minimale du client java CAS est triviale et décrite dans la documentation sur le site du projet CAS.

Les étapes sont les suivantes (voir la documentation du site CAS pour plus de détails) :
- Ajouter les librairies cas-client-core , commons-codec, commons-logging, opensaml-1.1 et xmlsec à celle de l'application.
- Déclarer les filtres CAS dans le descripteur de déploiement web.xml de l'application.

Ci-après un extrait du fichier web.xml en protocoles CAS 2.0 et SAML 1.1 dans le cas où ce filtre protège toute l'application ("/*). L'application métier est accessible sur l'interface locale, port 8080.

Attention : le caractère "↵" indique un saut de ligne typographique à ne pas prendre en compte.

4.1.2.a - Configuration CAS 2.0

```
<!-- Filtre d'authentification CAS 2.0 -->
<filter>
  <filter-name>FiltreCAS_AuthCAS20</filter-name>
```

```
  <filter-class>
    org.jasig.cas.client.authentication.AuthenticationFilter
  </filter-class>
  <init-param>
    <param-name>casServerLoginUrl</param-name>
    <param-value>
      [CAS_URL_BASE]/login
    </param-value>
  </init-param>
  <init-param>
    <param-name>serverName</param-name>
    <param-value>http://localhost:8080</param-value>
  </init-param>
</filter>

<!-- Validation de ticket CAS en version 2.0 -->
<filter>
  <filter-name>FiltreCAS_TicketCAS20</filter-name>
  <filter-class>org.jasig.cas.client.validation
.Cas20ProxyReceivingTicketValidationFilter
  </filter-class>
  <init-param>
    <param-name>casServerUrlPrefix</param-name>
    <param-value>[CAS_URL_BASE]</param-value>
  </init-param>
  <init-param>
    <param-name>serverName</param-name>
    <param-value>http://localhost:8080</param-value>
  </init-param>
</filter>

<!-- Récupérer l'identifiant utilisation dans la méthode getRemoteUser()
de la servlet -->
<filter>
  <filter-name>FiltreCAS_Wrapper</filter-name>
  <filter-class>
    org.jasig.cas.client.util.HttpServletRequestWrapperFilter
  </filter-class>
</filter>

<!-- Portée du filtre dans l'application. -->
<filter-mapping>
  <filter-name>FiltreCAS_AuthCAS20</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
<filter-mapping>
  <filter-name>FiltreCAS_TicketCAS20/filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
<filter-mapping>
  <filter-name>FiltreCAS_Wrapper</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

4.1.2.b - Configuration SAML 1.1

```
<!-- Filtre d'authentification -->
<filter>
  <filter-name>FiltreCAS_AuthSAML11</filter-name>
  <filter-class>
    org.jasig.cas.client.authentication.Saml11AuthenticationFilter
  </filter-class>
```

```
  <init-param>
    <param-name>casServerLoginUrl</param-name>
    <param-value>
      [CAS_URL_BASE]/login
    </param-value>
  </init-param>
  <init-param>
    <param-name>serverName</param-name>
    <param-value>http://localhost:8080</param-value>
  </init-param>
</filter>

<!-- Validation de ticket CAS en version 2.0 -->
<filter>
  <filter-name>FiltreCAS_TicketSAML11</filter-name>
  <filter-class>org.jasig.cas.client.validation
.Saml11TicketValidationFilter
  </filter-class>
  <init-param>
    <param-name>casServerUrlPrefix</param-name>
    <param-value>[CAS_URL_BASE]</param-value>
  </init-param>
  <init-param>
    <param-name>serverName</param-name>
    <param-value>http://localhost:8080</param-value>
  </init-param>
</filter>

<!-- Portée du filtre dans l'application. -->
<filter-mapping>
  <filter-name>FiltreCAS_AuthSAML11</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
<filter-mapping>
  <filter-name>FiltreCAS_TicketCAS20</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

4.1.3 - Utilisation du client java CAS

4.1.3.a - Protocole CAS 2.0

Utiliser la méthode `HttpServletRequest#getRemoteUser()` de la requête pour obtenir le courriel (*netid*) de l'utilisateur.

4.1.3.b - Protocole SAML 1.1.

L'objet `org.jasig.cas.client.authentication.AttributePrincipal` donne accès à l'ensemble des attributs de la réponse SAML.
Ces attributs sont mis à disposition dans un dictionnaire Java (`java.util.Map`) dont les clés sont définies dans le format de réponse Cerbère/CAS.

Ci-dessous un exemple **simplifié** de code pour lire les attributs nom, prénoms, courriel et unité.

Attention : le caractère "↵ " indique un saut de ligne typographique à ne pas prendre en compte.

```
AttributePrincipal principal = (AttributePrincipal)↵
request.getUserPrincipal();
```

```
Map attributs = principal.getAttributes();
String nom = (String) attributs.get("UTILISATEUR.NOM");
String prenom = (String) attributs.get("UTILISATEUR.PRENOM");
String courriel = (String) attributs.get("UTILISATEUR.MEL");
String unite = (String) attributs.get("UTILISATEUR.UNITE");
```

Les profils Cerbère sont présents dans l'attribut "AUTORISATION.PROFILS".
L'exemple de code suivant illustre leurs lecture.

```
List<String> profils = new ArrayList<String>();
Object profils = attributs.get("AUTORISATION.PROFILS");
```

Attention : Cet attribut est peut être multi-valué. De façon générale, la valeur d'un attribut fournie par le client Java CAS est de type "`String`" si mono-valué et "`List`" si multi-valué". La liste des profils pouvant contenir une ou plusieurs valeurs, il faut vérifier son format avant de la lire ("`String`" ou "`List`").

Les profils doivent être analysé selon la syntaxe suivante :

`PROFIL=`*`NomProfil`*`;`*`Portée`*`;`*`Restriction`

Les parties soulignées sont variables, les autres fixes.

4.1.3.c - Librairie Java d'analyse des assertions SAML Cerbère/CAS.

L'API Java client Cerbère SAML 1.1 simplifie l'accès aux attributs SAML.
Cette API modélise les assertions CAS/SAML Cerbère sous forme d'objets :

- `Personne` : propriétés d'identité de la personne authentifiée.
- `Utilisateur` : propriétés d'authentification et profils Cerbère de la personne.
- `Habilitation` : profils Cerbère
- `Application` : propriétés de l'application Cerbère courante.
- `Entreprise` : propriétés de l'entreprise de la personne, s'il s'agit d'un compte d'entreprise.

L'objet Cerbère est le point d'entrée de cette API. Il s'instancie par la méthode Cerbere.creationSAML11(attributs_SAML11).

Exemple :

```
 // Instanciation de l'API Cerbère.
 AttributePrincipal ppal (AttributePrincipal)
request.getUserPrincipal();
 Map attributs = ppal.getAttributes();
 Cerbere cerbere = Cerbere.creationSAML11(attributs);
 Utilisateur utilisateur = cerbere.getUtilisateur();

 //Propriété de l'utilisateur.
 String nom = utilisateur.getNom();

 // Liste des profils.
 Habilitation habilitation = cerbere.getHabilitation();
```

Cette API, sa documentation et un exemple complet d'utilisation sont fournis dans le dossier "outils/".

4.2 - Applications PHP

Le client CAS PHP est fourni sur le site du projet CAS, en version 1.3.2 à la date d'écriture de ce document.

Le dossier "outils/" contient des exemples d'intégration Cerbère/CAS en PHP.

4.2.1 - Configuration du serveur d'application PHP

Le serveur PHP (module Apache PHP par exemple) doit avoir été compilé avec le support des librairies curl, openssl et zlib (voir documentation du projet phpCAS).

Il est nécessaire de disposer du certificat racine de l'IGC/A fourni (AC-igca-1.0-20021213.pem).

4.2.2 - Installation du client phpCAS

L'installation du client phpCAS est triviale et décrit dans la documentation sur le site du projet CAS.

Il s'agit essentiellement de décompresser l'archive phpCAS sur le serveur PHP et de la mettre à disposition dans le chemin de recherche PHP.

4.2.3 - Utilisation du client phpCAS

4.2.3.a - Protocole CAS 2.0

Ci-dessous un exemple **simplifié** d'utilisation du client phpCAS.

L'identifiant utilisateur (*netid*) est obtenu par le méthode `phpCAS::getUser()`.

```
<?php
// Chargement de la librairie CAS.
include_once(CAS.php);

// Les paramètres du serveur CAS
$cas_serveur = "authentification-cerbere.application.i2";
$cas_port = 443;
$cas_serveur = "/cas";
$cas_chemin_igc = "/chemin/vers/AC-igca-1.0-20021213" ;

// Initialisation phpCAS en protocole CAS 2.0
phpCAS::client(CAS_VERSION_2_0, $cas_serveur, $cas_port,
    $cas_serveur, true);

// Le certificat de l'autorité IGC/A.
phpCAS::setCasServerCACert($cas_chemin_igca);

// L'authentification.
phpCAS::forceAuthentication();

// Lecture identifiant utilisateur (courriel)
$netid = phpCAS::getUser();
?>
```

4.2.3.b - Protocole SAML 1.1

Le protocole SAML 1.1 est indiqué à l'initialisation du client phpCAS :

```
...
// Initialisation phpCAS en protocole SAML 1.1
phpCAS::client(SAML_VERSION_1_1, $cas_serveur, $cas_port,
  $cas_serveur, true);
...
```

La méthode `phpCAS::getAttributes()` donne accès aux attributs SAML 1..1.
Ci-dessous un exemple **simplifié** de code pour lire les attributs nom, prénoms, courriel et unité.

```
$attrSAML = phpCAS::getAttributes();
$nom = $attrSAML["UTILISATEUR.NOM"];
$prenom = $attrSAML["UTILISATEUR.PRENOM"];
$courriel = $attrSAML["UTILISATEUR.MEL"];
$unite = $attrSAML["UTILISATEUR.UNITE"];
```

Les profils Cerbère sont présents dans l'attribut "AUTORISATION.PROFILS".
Cet attribut est une chaîne de caractère si le profil est unique, un tableau sinon.

L'exemple de code suivant crée une liste contenant les profils.

```
...
if(is_string($attrSAML["AUTORISATION.PROFILS"])) {
    $profils = array($attrSAML["AUTORISATION.PROFILS"]);
} else {
    $profils = $attrSAML["AUTORISATION.PROFILS"];
}
...
```

Les profils doivent être analysé selon la syntaxe suivante :
PROFIL=*NomProfil*;*Portée*;*Restriction*
Les parties *italiques* et soulignées sont variables, les autres fixes.

4.2.3.c - Librairie PHP d'analyse des assertions SAML Cerbère/CAS

L'API client Cerbère SAML 1.1 simplifie l'accès aux attributs SAML.

Cette API modélise les assertions CAS/SAML Cerbère sous forme d'objets :

- `Personne` : propriétés d'identité de la personne authentifiée.
- `Utilisateur` : propriétés d'authentification et profils Cerbère de la personne.
- `Profil` : détail d'un profil Cerbère.
- `Application` : propriétés de l'application Cerbère courante.
- `Entreprise` : propriétés de l'entreprise de la personne, s'il s'agit d'un compte d'entreprise.

L'objet Cerbère est le point d'entrée de cette API. Il s'instancie par la méthode `Cerbere::creationSAML11($attributs)`.

Exemple :

```
// Instanciation de l'API Cerbère.
$cerbere = Cerbere::Cerbere::creationSAML11($attributs);
$utilisateur = $cerbere->getUtilisateur();

//Propriété de l'utilisateur.
$nom = $utilisateur->getNom();

// Liste des profils.
$profils = $cerbere->getProfils();
```

Cette API, sa documentation et un exemple complet d'utilisation sont fournis dans le dossier "outils/".

4.3 - Module Apache

Il est possible de sécuriser une application, quelque soit sa technologie, par un module Apache 2. Les requêtes sont interceptées par Apache et l'utilisateur renvoyé vers Cerbère pour l'authentification.

Le module Apache mod_auth_cas est fourni sur le site projet CAS, en version 1.0.9.1 à la date de rédaction de ce document.

L'identité de l'utilisateur (netid) est fourni dans la variable d'environnement REMOTE_USER.

4.3.1 - Installation du module Apache

L'installation de ce module Apache est décrite dans la documentation sur le site projet CAS. Ce module peut être compilé ou être disponible dans la distribution Linux utilisée.

Quelle version de mod_auth_cas utiliser?

Les versions **1.8** et **1.9** fonctionnent pour une authentification en protocole **CAS 2.0**.

La dernière version à la date de rédaction de ce document (v1.0.9.1) contient des anomalies qui ne permettent pas de les utiliser en protocole SAML 1.1 sur Cerbère:

- Certaines de ces anomalies sont déjà enregistrées sur le projet mod_auth_cas. Le PNE Sécurité en a reproduit quelques ([MAS-66] notamment).
- D'autres anomalies constatées par le PNE Sécurité, ne sont pas enregistrées mais sont corrigée dans la dernière version de code (espace de nommage samlp codé en dur par exemple).

Fin août 2013, seule la dernière version issue des sources du projet permet d'utiliser les assertions SAML 1.1. Il est donc nécessaire de compiler manuellement ce module depuis la dernière version des sources pour une utilisation en SAML 1.1.

Il est nécessaire de disposer du certificat racine de l'IGC/A fourni (AC-igca-1.0-20021213.pem).

4.3.2 - Configuration du module Apache

La configuration minimale est la suivante :

```
# chargement du module
LoadModule auth_cas_module /chemin/vers/mod_auth_cas.so

# Verifier le certificat du serveur CAS.
CASValidateServer on

# URL d'authentification
CASLoginURL [CAS_URL_BASE]/login

# URL de validation CAS 2.0
CASValidateURL [CAS_URL_BASE]/serviceValidate
```

4.3.2.a - Cas particulier de la version 1.8

Cette version notamment fournie avec la distribution Debian Squeeze (la version fournie avec Debian Wheezy n'est pas concernée).

La version 1.8 ne valide pas correctement les chaînes de certification (anomalie [MAS-51] constatée par le PNE Sécurité sur un certificat signé de l'IGC du ministère). Il est alors nécessaire de désactiver la validation de certificat :

```
...
CASValidateServer Off
```

```
...
```

4.3.2.b - Cas particulier du protocole SAML 1.1

Quelques paramètres diffèrent.

```
...
# Utiliser le protocole SAML 1.1
CASValidateSAML On

# URL de validation SAML 1.1
CASValidateURL [CAS_URL_BASE]/samlValidate
...

#Délimiteur de profils multiples, choisir une valeur non utilisée.
CASAttributeDelimiter "|"
```

4.3.2.c - Intégration du certificat racine IGC/A

L'autorité de certification racine du serveur Cerbère/CAS (IGC/A) doit être connue du serveur Apache. C'est nativement le cas de certains systèmes (Debian par exemple).
Si cette autorité est inconnue du serveur, elle doit être précisée dans la configuration du module :

```
...
# Emplacement de l'autorité de certification IGC/A
CASCertificatePath /chemin/vers/certificat_IGCA.pem
...
```

4.3.3 - Protection des ressources Apache

Ci-dessous un exemple qui sécurise l'accès aux URLs de type /secret/*. Tout utilisateur authentifié est autorisé.

```
<Location /secret >
  # Authentifier par CAS
  AuthType CAS
  AuthName "Accès restreint"

  # Autoriser toute personne authentifiée.
  Require valid-user

  # Définir explicitement la portée de l'authentification
  CASScope /secret
</Location>
```

4.3.3.a - Cas particulier du protocole SAML 1.1

Le module mod_auth_cas peut transmettre les attributs SAML dans les en-têtes HTTP de la requête.
Pour cela il est nécessaire de renseigner au préalable le paramètre CASAuthNHeader :

```
...
# En-tête HTTP contenant l'identifiant CAS.
CASAuthNHeader CAS_ID
...
```

4.3.4 - Utilisation des les applications.

4.3.4.a - Protocole CAS 2.0

Le *netid* (courriel) de l'utilisateur est récupérable dans la variable REMOTE_USER.

4.3.4.b - Protocole SAML 1.1

Les attributs SAML sont transmis dans les en-têtes HTTP de la requête, ci-dessus un extrait de ces en-têtes :

```
...
CAS_ID: 123456
CAS_UTILISATEUR.ID: 123456
CAS_UTILISATEUR.MEL: Martin.Durant@developpement-durable.gouv.fr
CAS_UTILISATEUR.NOM: DURANT
CAS_UTILISATEUR.PRENOM: Martin
CAS_UTILISATEUR.ADR_VILLE: La DEFENSE
...
CAS_ENTITE.UNITE: SG/SPSSI/PSI/PSI4
...
CAS_AUTORISATION.PROFILS: PROFIL=ADMIN;SG;R01|PROFIL=CONSULT;fr;none
...
```

Ces attributs sont mis à disposition de l'application métier pour information ou filtrage.

4.4 - Installation d'un serveur Cerbère/CAS bouchon

Le serveur disponible en libre téléchargement sur le site du projet CAS peut être utilisé en mode bouchon pour simuler une réponse Cerbère/CAS.

4.4.1 - Installation du serveur CAS

Télécharger le serveur CAS sur le site du projet du même nom :
http://www.jasig.org/cas/download

L'application web se trouve dans "modules/cas-server-webapp-*VERSION*.war". La déployer sur un serveur Apache Tomcat.
L'adresse d'authentification est par défaut est :
http://*nomServeur:8080*/ cas-server-webapp-*VERSION*/login

L'URL de base Cerbère/CAS est :
[CAS_URL_BASE] = http://*nomServeur:8080*/ cas-server-webapp-*VERSION*

4.4.2 - Authentification CAS 2.0

Par défaut, le serveur CAS fonctionne en mode bouchon. Pour s'authentifier avec succès il suffit d'indiquer un identifiant de connexion quelconque avec un mot de passe égal à cet identifiant (exemple : martin/martin).

Pour simuler le fonctionnement Cerbère/CAS 2.0, il faut s'authentifier avec un identifiant de connexion de type courriel.

Exemple :
- Identifiant : sebastien.martin@developpement-durable.gouv.fr
- Passe : sebastien.martin@developpement-durable.gouv.fr

Le *netid* CAS retourné sera le courriel indiqué comme identifiant de connexion.

4.4.3 - Authentification SAML 1.1 – Utilisation d'attributs

Pour simuler une réponse Cerbère/CAS de type SAML 1.1, il est nécessaire de compléter l'installation du serveur CAS.

La librairie Cerbère bouchon, version 4.1.0 et supérieures doit être utilisée.

4.4.3.a - Création d'une base de compte de test – Fichier Cerbère bouchon

Utiliser le fichier cerbere-filtre-bouchon.xml du filtre Java Cerbère bouchon pour créer une base de comptes de test. Voir pour cela la documentation de Cerbère bouchon.

4.4.3.b - Intégration des fichiers Cèrbere bouchon dans le serveur CAS

Les fichiers du filtre Cerbère bouchon doivent être déployés sur le serveur CAS :

- Copier le fichier cerbere-filtre-bouchon.xml dans le dossier WEB-INF/ du serveur CAS.
- Copier la librairie cerbere-bouchon-*VERSION*.jar dans le dossier WEB-INF/lib/ du serveur CAS.

Il ne reste plus qu'à modifier la configuration du serveur CAS afin qu'il prenne en compte ces nouveaux éléments. Cette configuration se fait le fichier WEB-INF/**deployerConfigContext.xml** du serveur CAS. Deux éléments doivent être modifiés.

► Utiliser le fichier cerbere-filtre-bouchon.xml comme référentiel utilisateurs.

Modifier le composant (*bean*) "attributeRepository" pour remplacer son implémentation par défaut par celle de Cerbère bouchon.

```
<bean id="attributeRepository"
    class="i2.application.cerbere.bouchon.cas.GestionnaireAttributs">
  <property name="appName" value="NOM-APPLICATION" />
  <property name="fichier"
      value="/chemin/vers/cerbere-filtre-bouchon.xml" />
</bean>
```

Remplacer *NOM-APPLICATION* par le nom d'application défini dans le fichier cerbere-filtre-bouchon.xml (<application applicationId="*NOM-APPLICATION*">).

► Autoriser la diffusion des attributs SAML

Modifier le composant (*bean*) "RegexRegisteredService", lui-même compris dans le composant "serviceRegistryDao", en lui ajoutant une propriété "ignoreAttributes" de valeur "true".

```
<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="..." />
        <property name="serviceId" value="^(https?|imaps?)://.*" />
        <property name="evaluationOrder" value="10000001" />
        <property name="ignoreAttributes" value ="true" />
      </bean>
    </list>
  </property>
</bean>
```

Redémarrer le serveur Apache Tomcat.

4.4.3.c - Authentification sur le fichier Cerbère bouchon

La logique du serveur CAS en mode bouchon est toujours la même : l'identifiant de connexion et le mot de passe doivent être identiques.

S'authentifier avec le courriel de l'un des comptes du fichier cerbere-filtre-bouchon.xml, **ne pas utiliser les identifiants et mots de passe de ce fichier**.

Le serveur CAS renverra les informations présentes dans le fichier Cerbère bouchon sous forme d'assertions SAML 1.1.

■