CAS, la théorie

R. Ferrere, S. Layrisse

ANF Les systèmes d'authentification dans la communauté ESR : étude, mise en oeuvre et interfaçage dans un laboratoire de Mathématique
Angers, 22 - 26 septembre 2014

Plan

Introduction

Principe de fonctionnement

Techniques de CASification d'une application (mod_cas, phpCAS...)

CASification des applications web

CASification des applications non web

Couplage LDAP

Conclusion

Quelques références...

Progression

Introduction

Principe de fonctionnement

Techniques de CASification d'une application (mod_cas, phpCAS...)

CASification des applications web

CASification des applications non web

Couplage LDAP

Conclusion

Quelques références...

Quésaco le CAS ?

- CAS (Central Authentication Service) - Université de Yale (Etats-Unis), projet ESUP-Portail, Fondation Apero
- Développé en Java (servlets), service et protocole
- Version 4.0 depuis mai 2014, Open Source
- Système d'authentification SSO libre gérant des identités numériques
- Destiné à de nombreux services et applications web
- 1 compte unique utilisateur —> N applications/services

Pourquoi SSO avec CAS ?

- Généralisation des ENT
- Authentifier les utilisateurs (LDAP,Kerberos)
- Accéder à des applications tierces et en multi-tiers(proxy)
- Sécurité : fin des mots passe en clair, réauthentification
- Certifier des identités (transmission de cookies privés - identifiants de session)
- Protéger des pages Web : librairies clientes (phpCAS) et module Apache (mod_cas)
- Authentifier par service Unix avec PAM (pam_cas)

Progression

Basé sur le navigateur internet

Navigateur = Utilisateur

- Navigateur récent et à jour
- Redirections HTTP
- Interprétation javascript
- Stockage de cookies
- Moteur de chiffrement —> HTTPS

Basé sur un client CAS

- Agent d'authentification (application web avec libraire CAS, serveur web avec mod_cas)
- Agent de protection

Nombreux clients CAS

- phpCAS
- .Net Cas client
- CAS client pour java
- Module auth_cas
- Framework Spring java (JavaBeans et MVC)
- Module shiro-cas

Basé sur des tickets

- Requêtes HTTP (navigateur-client CAS)
- Redirection HTTPS + passage d'informations par cookies (serveur CAS)
- Modes de fonctionnement : standard et proxy

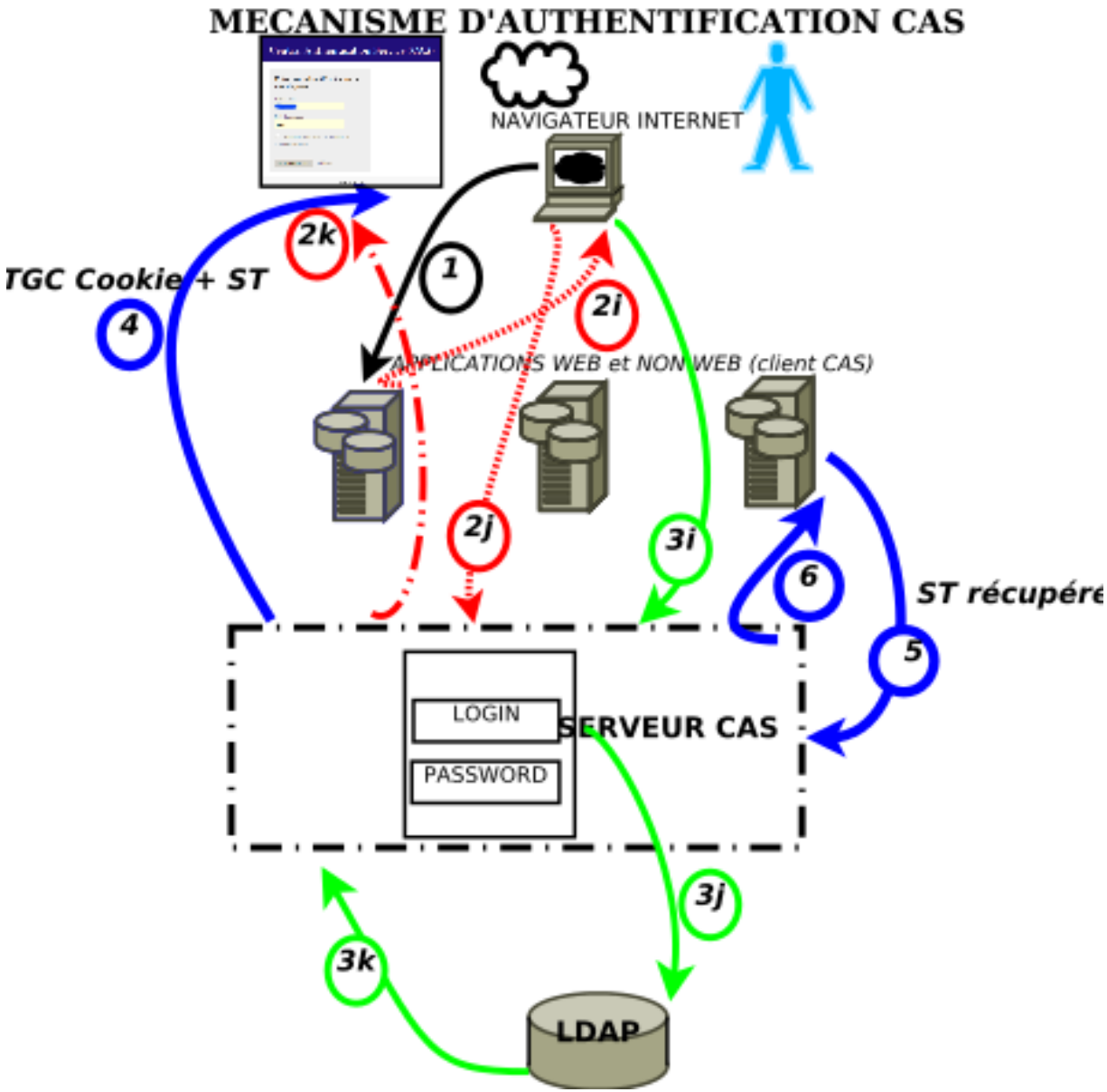
Authentification

- base : TGC (Ticket Granting Ticket) envoyé
- proxy : PGT (Proxy Granting Ticket)

Accès aux ressources/applications

- base : ST (Service Ticket)
- proxy : PT (Proxy Ticket)

Mécanisme de base - CAS



Principe - authentification réussie

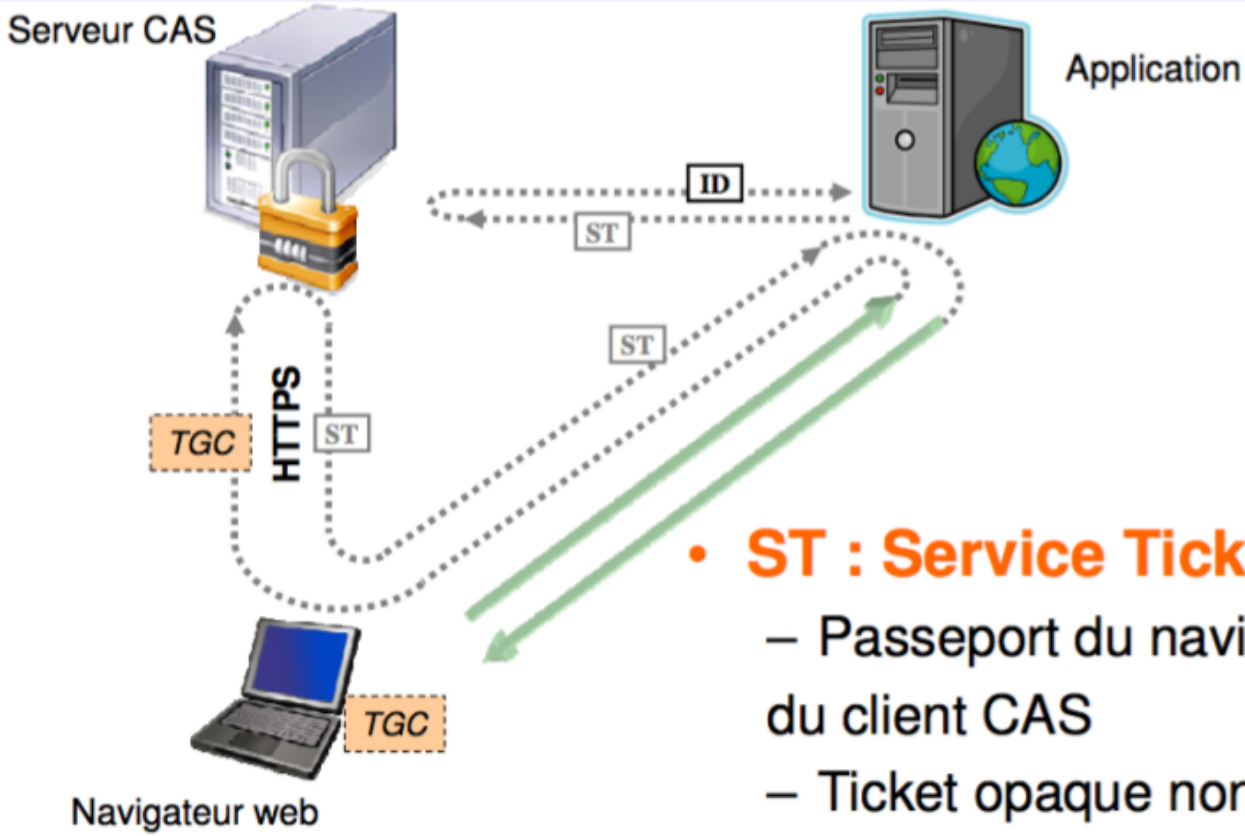
- **1)** Requête utilisateur non authentifiée vers application
- **2i,j,k)** Renvoi au navigateur d'une redirection vers CAS avec URL de service
- **3i,j,k)** Saisie du login/mot de passe, validation LDAP
- **4)** Dépôt d'une cookie de session sécurisé (TGC) avec un ticket de service (ST), association de l'identifiant utilisateur, contrôle de validité du service demandé, redirection de l'utilisateur vers l'URL du service en lui attribuant le ST
- **5)** Récupération du ST par l'application pour CAS
- **6)** Contrôle CAS de l'URL <—> ST, envoi de l'identifiant utilisateur

Si échec validation du ST, redirection vers CAS en 2....

Cinématique CAS

(présentée par Philippe Depouilly au RNBM - octobre 2010)

Issue de la présentation faite par J. Marchal aux Josy 2010 :



- **ST : Service Ticket**
  - Passeport du navigateur auprès du client CAS
  - Ticket opaque non rejouable
  - Limité dans le temps

Principe en multi-tiers proxy

- Via mandataire (proxy) CAS : portail ENT ou de messagerie
- Redirection différente du mécanisme de base

Déroulement

- Demande d'un ticket proxy ne passe plus par le navigateur
- Validation du ST + Récupération PGT
- Validation d'un PT
- Et ainsi de suite...si d'autres serveurs mandataires CAS

Avantages - efficacité

- Authentification, transmission et certification de l'identité des utilisateurs (tiers de confiance)
- Activation d'un SSO natif et de bout en bout
- Plus de circulation de mot de passe mais plutôt un ticket
- Plus de réauthentification (une seule fois)
- Délégation de gestion de l'authentification pour les différents services web
- Annuaires LDAP –> référentiel commun
- Support API Memcache

Dépendance - fichier pom.xml

cas-server-integration-memcache

Progression

CASification / sarCASmes !

- CASifier : rendre compatible avec CAS pour ajouter de l'authentification
- Plusieurs techniques ou modules pour adapter et intégrer
- CASification des applications web par des librairies ou bibliothèques (client java, client CAS)
- CASification des applications non web par des modules type PAM (pam_cas) codé en C

Exemples d'applications casifiables

- SquirreMail, Webmail HORDE (Cyrus et IMAP)
- Client Webmail Roundcube
- GLPI, GRR
- WIKI (MediaWiki...)
- PhpGroupware
- CMS comme SPIP, DRUPAL
- Tomcat/Apache
- RADIUS (portail captif web FreeRADIUS)

Casifier avec module Apache - mod_auth_cas

- Authentification apache via CAS (au lieu de htpassword)
- Protection de pages Web (.htaccess)

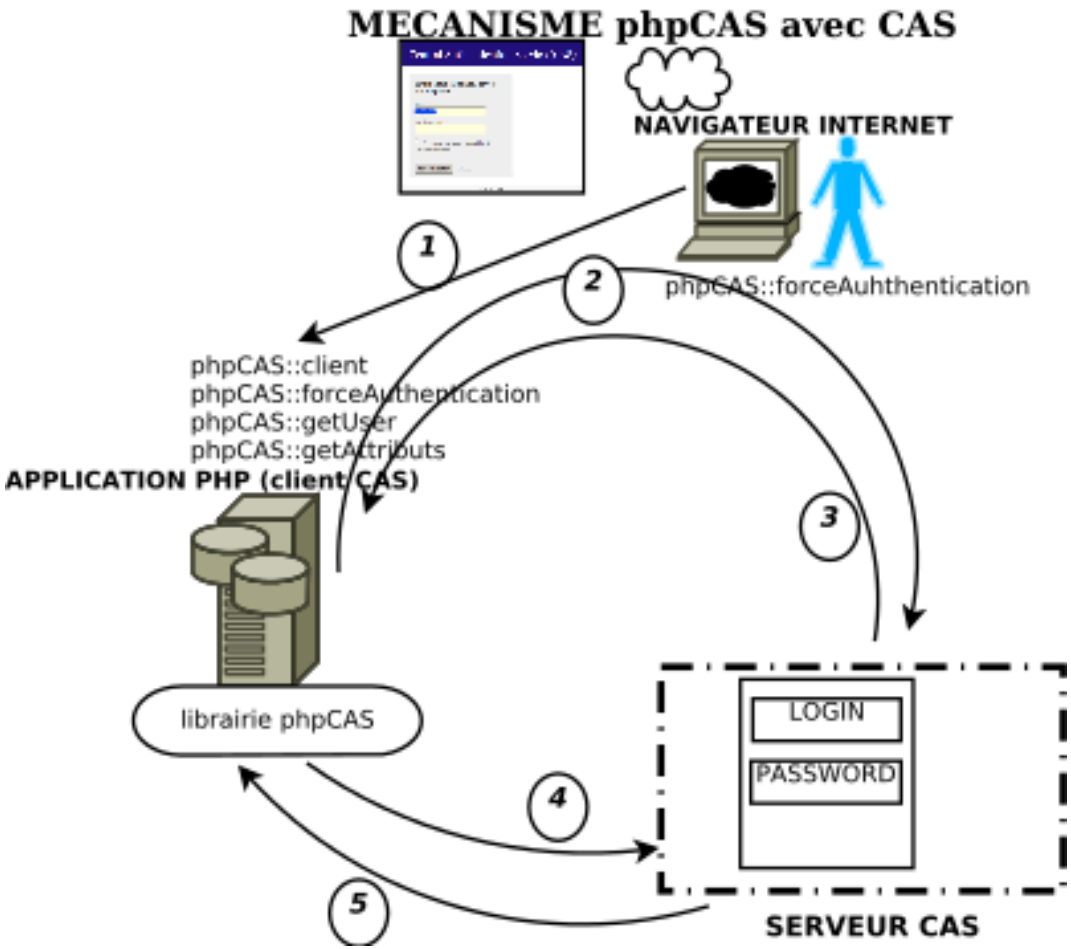
directives apache

AuthType CAS
Require valide-user (Require all granted)

Casifier avec librairie phpCAS

- Repris et développé par ESUP-Portail, maintenu par P.AUBRY/J.MARCHAL
- Applications PHP, Version actuelle CAS-1.3.3
- https ://wiki.jasig.org/display/CASC/phpCAS
- pré-requis : serveur web/apache,IIS..., PHP supportant cURL, OpenSSL, DOM(xml), Zlib
- Avec ou sans PEAR pour la portabilité du composant phpCAS

Fonctionnement schématique de la librairie phpCAS



phpCAS et flux

1. Requête utilisateur exige une authentification
2. Redirection vers la page de connexion CAS (phpCAS : :forceAuthentication)
3. Authentification, redirection du navigateur avec ticket de service (ST) dans l'URL
4. Obtention du ST pour validation CAS
5. Ticket valide, réponse par un message de validation avec ID et attributs utilisateur

phpCAS : méthodes et fonctions

- phpCAS : :client(CAS_VERSION....), initialisation phpCAS
- phpCAS : :forceAuthentication , authentification obligatoire
- phpCAS : :getUser(), utilisation connecté
- phpCAS : :setDebug(), cause d'erreur
- phpCAS : :getServiceURL(), URL d'accès au service
- phpCAS : :logout....(), URL de déconnexion

Casifier avec librairie client JAVA

- Ajouter filtres fichier web.xml
- Ajouter listener SSO
- Indiquer URL fichier seraph-config.xml
- Configurer CAS Logout fichier xwork.xml

Casifier avec module PAM et Apache

- Module pam_cas avec le client CAS (codé en C)
- Authentification par un service de base Unix
- Réception Proxy Ticket par le service Unix, validé par pam_cas auprès de CAS
- Fonctionnement multi-tiers : accès mandataire client CAS fournisseur de Ticket

Casifier serveur Cyrus-IMAP

- Librairie cyrus-sasl autonome
- Possibilité d'utiliser un démon Unix : salsauthd
- Pour les différents mécanismes d'authentification (PAM, LDAP, Kerberos...)
- Cache pour les mots de passe utilisateurs : Proxy Ticket est stocké et rejoué (serveur CAS moins sollicité)

Mise en oeuvre

- Activation du module pam_cas pour le service IMAP
- Compilation de cyrus-sasl avec saslauthd
- Méthode saslauthd dans /etc/imapd.conf
- Cache : saslauthd avec l'option -c (gain de performance)

Progression

Configuration authentification LDAP

Mode d'accès fonction structure LDAP

2 modes d'accès - GenericHandler

FASTBIND (accès direct) –> connexion réussie

- CAS –> LDAP avec DN utilisateur
- Classe : FastBindLdapAuthenticationHandler
- Filtre %u et jeton

BIND (simple recherche anonyme)

- Classe : BindLdapAuthenticationHandler
- Plusieurs attributs de filtre
- Autorisation de comptes multiples

Redondance

Liste URLs LDAP vue comme réplicas

LDAP - serveur CAS

- Installer et compiler le bundle LDAP
- Configurer le bundle
- Ajouts - deployerConfigContext.xml (webapps/cas/WEB-INF/deployerConfigContext.xml)
  1. Attributs
  2. Authentification
  3. ContextSource (contexte LDAP)
- Modification de ldap-auth.xml (bean attribRepository)

Configuration AuthenticationHandler

- LDAP via SSL (ldaps)
- Propriétés et dépendance –> fichier pom.xml
- Propriétés du bean (filter, searchBase, ContextSource,...)
  1. filter (mail utilisateur)
  2. searchBase (ou=people ou cn=Users)
  3. searchContextSource
  4. ContextSource (url en ldaps, userDN, password)
  5. pooledContextSource
- Pooling ContextSource (performance réseau et HA)

Progression

Introduction

Principe de fonctionnement

Techniques de CASification d'une application (mod_cas, phpCAS...)

CASification des applications web

CASification des applications non web

Couplage LDAP

Conclusion

Quelques références...

En résumé CAS

- Un grand intérêt
  1. Solution libre de SSO relativement simple - LDAP et Kerberos
  2. Indispensable pour protéger les applications Web et non Web
  3. Beaucoup de possiblités côté clients CAS
  4. Sécurité et conformité (tickets)
- Des limitations...
  1. Disponibilité du serveur CAS exigée - élément central du réseau
  2. Propre à un réseau local d'établissement (ENT Université)
  3. Connexions inter-établissement –> Shibboleth
  4. Pas de propagation d'attributs et d'authentification entre établissements
  5. Tolérance aux pannes !

Progression

liens internet et articles

- Documentation CAS - Wiki installation CAS
- Protocole CAS - Clients CAS
- SSO avec CAS - JRES 2003 Pascal Aubry
- Fondation Apero
- Authentification LDAP avec CAS - GenericHandler
- Memcache
- Tuto Authentication SSO-CAS

liens internet et articles

- Librairie phpCAS et exemples
- LDAP Authentification avec CAS et GenericHandler item Casification du serveur Cyrus-IMAP
- "Mise en oeuvre de phpCAS", GNU/Linux Magazine numéro 114
- Gestionnaire d'Authentication RADIUS - Handler
- "Ecriture d'un module RAIDIUS : validation de tickets CAS", GNU/Linux Magazine numéro 123
- Plugin CAS Roundcube

- Merci pour votre attention.
- Avez-vous des questions ?