Retour de l'URN sur mise en place de CAS 6.4.1

contexte

Eté 2021, l'Université de Rouen Normandie a procédé à une montée de version de son service d'authentification CAS en **6.4.0** (puis **6.4.1** en octobre 2021)

Cette page wiki vient en complément de notre page **Retour de l'URN sur mise en place de CAS 6.0.4** ; les retours que l'on avait fait sur la 6.0.4 sont en effet toujours d'actualité ; nous en profitons ici pour repréciser 2-3 éléments et donner nos fichiers de configurations.

Un retour a également été fait au travers d'une communication aux esup-days #32 en septembre 2021, le diaporama est disponible ici : https://esupportail.github.io/presentations/esup-days-32-esup-otp-cas/

cas-overlay

Notre CAS est construit grâce au maven overlay disponible ici : https://github.com/apereo/cas-overlay-template.git

Les fichiers modifiés/ajoutés sont les suivant :

```
build.gradle
gradle.properties
lib/jcifs-ext.jar
src/main/resources/static/images/cas-logo.png
src/main/resources/templates/fragments/footer.html
src/main/resources/templates/fragments/pmlinks.html
src/main/resources/templates/layout.html
```

Notre build.gradle comporte les modules supplémentaires que l'on a décidés d'ajouter.

Pour les modules ESUP, on a utilisé https://jitpack.io , on a ainsi ajouté dans repositories :

```
    maven {
        url "https://jitpack.io"
    }
```

Ainsi que les dépendances suivantes :

```
implementation "org.apereo.cas:cas-server-support-redis-ticket-registry:${casServerVersion}"
implementation "org.apereo.cas:cas-server-core-authentication:${casServerVersion}"
implementation "org.apereo.cas:cas-server-support-ldap:${casServerVersion}"
implementation "org.apereo.cas:cas-server-support-json-service-registry:${casServerVersion}"
implementation "org.apereo.cas:cas-server-support-spnego-webflow:${casServerVersion}"
implementation files("${projectDir}/lib/jcifs-ext.jar")
implementation "org.apereo.cas:cas-server-support-trusted-mfa:${casServerVersion}"
implementation "org.apereo.cas:cas-server-support-trusted-mfa-mongo:${casServerVersion}"
implementation "org.apereo.cas:cas-server-support-throttle:${casServerVersion}"
implementation "org.apereo.cas:cas-server-support-reports:${casServerVersion}"
implementation "com.github.vbonamy:cas-server-support-agimus-cookie:cas-6.4.x-SNAPSHOT"
implementation "com.github.vbonamy:cas-server-support-agimus-logs:cas-6.4.x-SNAPSHOT"
implementation "com.github.EsupPortail:esup-otp-cas:283db88044"
```

cas.properties

Notre fichier de propriétés /etc/cas/config/cas.properties reprend les configurations de CAS et modules apereo associés :

```
######################
## Parametre serveur ##
######################
cas.server.name: https://cas.univ-rouen.fr
cas.server.prefix: https://cas.univ-rouen.fr

cas.audit.ignoreAuditFailures=false
cas.audit.appCode=CAS
cas.audit.numberOfDaysInHistory=30
cas.audit.includeValidationAssertion=false
cas.audit.alternateServerAddrHeaderName=
cas.audit.engine.alternate-client-addr-header-name=X-Forwarded-For
cas.audit.useServerHostAddress=false

# clearpass
cas.clearpass.cacheCredential=true
cas.clearpass.crypto.encryption.key=xxxxxxxxxxxxxxxxxxxxxxxxx
cas.clearpass.crypto.signing.key=xxxxxxxxxxxxxxxxxxxxxxxxx

# décommenter pour tester le modifs html sans redémarrer tomcat
#spring.thymeleaf.cache=false

### On active le cache pour le formulaire d'authentication (par défaut:true)
# When true, will inject the following headers into the response for non-static resources.
# Cache-Control: no-cache, no-store, max-age=0, must-revalidate
# Pragma: no-cache
# Expires: 0
cas.http-web-request.header.cache=true

### Modification du timeout pour l'affichage du session report dans le dashboard (par défaut: 5s)
cas.http-client.asyncTimeout=PT60S
cas.http-client.connectionTimeout=PT5S
cas.http-client.readTimeout=PT5S


### Valeur d'expiration des TGTs
# Set to a negative value to never expire tickets
cas.ticket.tgt.primary.max-time-to-live-in-seconds=1209600
cas.ticket.tgt.primary.time-to-kill-in-seconds=28800

### Valeur expiration tickets service/proxy : 10sec par défaut, on augmente car latence sur les sogo ...
cas.ticket.pt.timeToKillInSeconds=60
cas.ticket.st.timeToKillInSeconds=60

# rememberme
cas.ticket.tgt.rememberMe.enabled=true
cas.ticket.tgt.rememberMe.timeToKillInSeconds=1209600
# rememberme 2 semaines ok pour le cookie
tgc.remember.me.maxAge=1209600
cas.tgc.rememberMeMaxAge=1209600
# pinToSession = false pour que ça fonctionne même si chgt d'IP par exemple.
cas.tgc.pinToSession=false

# Redirection vers l'url du service au lieu de la page de déconnexion du cas après logout
cas.logout.follow-service-redirects=true


#########
## SLO ##
#########
cas.slo.asynchronous=true
cas.slo.disabled=false


######################
## Access dashboard ##
######################
```

```
# https://apereo.github.io/2018/11/06/cas6-admin-endpoints-security/
management.endpoints.web.exposure.include=*
management.endpoints.enabled-by-default=true
cas.monitor.endpoints.endpoint.defaults.access=IP_ADDRESS
cas.monitor.endpoints.endpoint.defaults.requiredIpAddresses=127.0.0.1,10.0.0.3


#########
## Log ##
#########
logging.config: file:/etc/cas/config/log4j2.xml


##############################
## parametre generique CAS ##
##############################
cas.authn.accept.users=
# Liste des atributs à renvoyer par défaut (si non défini dans le service)
cas.authn.attributeRepository.core.defaultAttributesToRelease=uid,mail,displayName,eduPersonPrincipalName,
eduPersonAffiliation,sn,givenname
# throttle (pour limiter le nombre de tentative d'authentification)
cas.authn.throttle.failure.range-seconds=30
cas.authn.throttle.failure.threshold=12
# pour throttle sur IP *et* username
cas.authn.throttle.core.username-parameter=username

############################
## Authentification LDAP ##
############################
cas.authn.ldap[0].name=openldap
cas.authn.ldap[0].order=0
cas.authn.ldap[0].type=DIRECT
cas.authn.ldap[0].ldapUrl=ldap://ldap.univ-rouen.fr ldap://ldap-clone.univ-rouen.fr
cas.authn.ldap[0].baseDn=ou=people,dc=univ-rouen,dc=fr
cas.authn.ldap[0].searchFilter=(&(uid={user})(objectclass=eduPerson))
cas.authn.ldap[0].dnFormat=uid=%s,ou=people,dc=univ-rouen,dc=fr
cas.authn.ldap[0].connectionStrategy=ROUND_ROBIN
cas.authn.ldap[0].useSsl=false
cas.authn.ldap[0].useStartTls=false
cas.authn.ldap[0].connectTimeout=20
cas.authn.ldap[0].subtreeSearch=false
#cas.authn.ldap[0].principalAttributeId=uid
# limite le nombre d'attributs demandés lors du bind
cas.authn.ldap[0].principalAttributeList=

# le cache utilise en clef un hash non injectif (collisions possibles)
cas.authn.attribute-repository.core.expiration-time = 0
# Attribute repository pour récupérer les attributs (utile si authentification autre que openldap)
cas.authn.attributeRepository.maximum-cache-size=0
# hack pour ne PAS récupérer TOUS les attributs LDAP (permet aussi de mapper les attributs, ex: xxx.ldap[0].
attributes.nom=givenName)
# pour spnego, le .* DOIT être évité
#cas.authn.attributeRepository.ldap[0].attributes.*=
cas.authn.attributeRepository.ldap[0].attributes.uid = uid
cas.authn.attributeRepository.ldap[0].attributes.mail = mail
cas.authn.attributeRepository.ldap[0].attributes.displayName = displayName
cas.authn.attributeRepository.ldap[0].attributes.eduPersonPrincipalName = eduPersonPrincipalName
cas.authn.attributeRepository.ldap[0].attributes.eduPersonAffiliation = eduPersonAffiliation
cas.authn.attributeRepository.ldap[0].attributes.radiusFilterId = radiusFilterId
cas.authn.attributeRepository.ldap[0].attributes.memberOf = memberOf
cas.authn.attributeRepository.ldap[0].attributes.dn = dn
cas.authn.attributeRepository.ldap[0].attributes.givenname = givenname
cas.authn.attributeRepository.ldap[0].attributes.sn = sn
cas.authn.attributeRepository.ldap[0].attributes.mobile = mobile
cas.authn.attributeRepository.ldap[0].ldapUrl=ldap://ldap.univ-rouen.fr
cas.authn.attributeRepository.ldap[0].connectionStrategy=ROUND_ROBIN
cas.authn.attributeRepository.ldap[0].order=0
cas.authn.attributeRepository.ldap[0].useSsl=false
cas.authn.attributeRepository.ldap[0].useStartTls=false
cas.authn.attributeRepository.ldap[0].connectTimeout=20
cas.authn.attributeRepository.ldap[0].baseDn=ou=people,dc=univ-rouen,dc=fr
```

```
cas.authn.attributeRepository.ldap[0].searchFilter=(&(uid={user})(objectclass=eduPerson))
cas.authn.attributeRepository.ldap[0].subtreeSearch=false
cas.authn.attributeRepository.ldap[0].bindDn=cn=cas,dc=univ-rouen,dc=fr
cas.authn.attributeRepository.ldap[0].bindCredential=xxxxxxxxxxxxxxxxxxxxxxxx
cas.authn.attributeRepository.ldap[0].poolPassivator=NONE
cas.authn.attributeRepository.ldap[0].minPoolSize=2
cas.authn.attributeRepository.ldap[0].maxPoolSize=10
cas.authn.attributeRepository.ldap[0].validateOnCheckout=true
cas.authn.attributeRepository.ldap[0].validatePeriodically=true
cas.authn.attributeRepository.ldap[0].validatePeriod=60
cas.authn.attributeRepository.ldap[0].validateTimeout=30
cas.authn.attributeRepository.ldap[0].failFast=true
cas.authn.attributeRepository.ldap[0].idleTime=300
cas.authn.attributeRepository.ldap[0].prunePeriod=300
cas.authn.attributeRepository.ldap[0].blockWaitTime=300
cas.authn.attributeRepository.ldap[0].providerClass=org.ldaptive.provider.unboundid.UnboundIDProvider

## ETAT de sante LDAPs
cas.authn.ldap[0].validator.type=SEARCH
cas.authn.ldap[0].validator.baseDn=ou=people,dc=univ-rouen,dc=fr
cas.authn.ldap[0].validator.searchFilter=(objectClass=organizationalUnit)
cas.authn.ldap[0].validator.scope=OBJECT
cas.authn.ldap[0].validator.attributeName=objectClass
cas.authn.ldap[0].validator.attributeValues=top
cas.authn.ldap[0].validator.dn=ou=people,dc=univ-rouen,dc=fr

## Passivator (utilisé si utilisation de DIRECT ou AUTHENTICATED pour l'authn Ldap)
# permet de passifier les connexions dans le pool (https://apereo.github.io/cas/6.4.x/integration/Attribute-
Release-Consent-Storage-LDAP.html#configuration)
cas.authn.ldap[0].poolPassivator=NONE
cas.authn.ldap[0].minPoolSize=1
cas.authn.ldap[0].maxPoolSize=10
cas.authn.ldap[0].validateOnCheckout=false
cas.authn.ldap[0].validatePeriodically=true
cas.authn.ldap[0].validatePeriod=60
cas.authn.ldap[0].validateTimeout=30
cas.authn.ldap[0].failFast=true
cas.authn.ldap[0].idleTime=300
cas.authn.ldap[0].prunePeriod=300
cas.authn.ldap[0].blockWaitTime=300


######################
## SPNEGO           ##
######################

cas.authn.spnego.mixedModeAuthentication=true
#cas.authn.spnego.supportedBrowsers=MSIE,Trident,Firefox,AppleWebKit
cas.authn.spnego.supportedBrowsers=Firefox
cas.authn.spnego.send401OnAuthenticationFailure=true
cas.authn.spnego.ntlmAllowed=false
cas.authn.spnego.principalWithDomainName=false
cas.authn.spnego.name=spnego
cas.authn.spnego.ntlm=false
cas.authn.spnego.order=0

cas.authn.spnego.system.kerberosConf=file:/etc/krb5.conf
cas.authn.spnego.system.loginConf=file:/etc/cas/config/login.conf
cas.authn.spnego.system.kerberosRealm=UR.UNIV-ROUEN.FR
cas.authn.spnego.system.kerberosDebug=false
cas.authn.spnego.system.useSubjectCredsOnly=false
cas.authn.spnego.system.kerberosKdc=10.0.0.12

cas.authn.spnego.properties[0].jcifsUsername=cas-spnego
cas.authn.spnego.properties[0].jcifsDomainController=notread.univ-rouen.fr
cas.authn.spnego.properties[0].jcifsDomain=UR.UNIV-ROUEN.FR
cas.authn.spnego.properties[0].jcifsServicePassword=xxxxxxxxxxxxxxxxxxxxxxxx
cas.authn.spnego.properties[0].jcifsPassword=xxxxxxxxxxxxxxxxxxxxxxxx
cas.authn.spnego.properties[0].jcifsServicePrincipal=HTTP/nommachinecas.univ-rouen.fr@UR.UNIV-ROUEN.FR
cas.authn.spnego.properties[0].cachePolicy=600
cas.authn.spnego.properties[0].timeout=300000
```

```
cas.authn.spnego.properties[0].jcifsNetbiosWins=

### SPNEGO Client Selection Strategy
cas.authn.spnego.hostNameClientActionStrategy=hostnameSpnegoClientAction

### SPNEGO Client Selection Hostname

cas.authn.spnego.alternativeRemoteHostAttribute=alternateRemoteHeader
# IPs internes sans VPN ? TODO : à fixer ... :-/
#cas.authn.spnego.ipsToCheckPattern=10\.[1-9][0-9]\d{0,3}\.[0-9]\d{0,3}\.[0-9]\d{0,3}
cas.authn.spnego.ipsToCheckPattern=.+
cas.authn.spnego.dnsTimeout=2000
# hostname du boitier vpn à exclure de spnego
#cas.authn.spnego.hostNamePatternString=^(?!.*vpn[ct]-aa\.univ-rouen\.fr).*$
cas.authn.spnego.hostNamePatternString=^(?!.*(vpn|vqn|vrn|vsn)-aa\.univ-rouen\.fr).*$

######################
## Service REGISTRY ##
######################
cas.serviceRegistry.schedule.repeatInterval=10000
cas.serviceRegistry.schedule.startDelay=1000


#########################################################
## Parametrage JSON pour la persistence des services ##
#########################################################
cas.serviceRegistry.json.location=file:/etc/cas/services


################################################
## Paramétrage WebFlow (Client-side Sessions) ##
################################################
cas.webflow.crypto.enabled=true
cas.webflow.crypto.signing.key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
cas.webflow.crypto.signing.keySize=512
cas.webflow.crypto.encryption.keySize=16
cas.webflow.crypto.encryption.key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
cas.webflow.crypto.alg=AES


######################
## Paramétrage TGC ##
######################
# Chiffre les TGC ... à noter que lorsque chiffré, le TGC contient l'ip et le user-agent du demandeur.
# Si pour x raisons il y a un changement d'user-agent (rare) ou d'adresse IP (plus fréquent avec l'itinérance)
pendant la session,
# CAS le détecte lors de la présentation du cookie et celui-ci est invalidé. Une réauthentification est alors
nécessaire.
# true pour activer
cas.tgc.crypto.enabled=true
cas.tgc.crypto.encryption.key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
cas.tgc.crypto.signing.key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx


#########################################################
## Parametrage REDIS pour la persistence des tickets ##
#########################################################
cas.ticket.registry.redis.host=localhost
cas.ticket.registry.redis.database=0
cas.ticket.registry.redis.port=6379
cas.ticket.registry.redis.password=
cas.ticket.registry.redis.timeout=2000
cas.ticket.registry.redis.useSsl=false

cas.ticket.registry.redis.crypto.enabled=false
cas.ticket.registry.redis.crypto.signing.key=
cas.ticket.registry.redis.crypto.signing.keySize=512
cas.ticket.registry.redis.crypto.encryption.key=
cas.ticket.registry.redis.crypto.encryption.keySize=16
cas.ticket.registry.redis.crypto.alg=AES
```

```
##############################
## Locale                   ##
##############################
cas.locale.defaultValue=fr

##############################
## ESUP-SMSU                ##
##############################
cas.smsProvider.rest.url=http://esup-smsu-api.univ-rouen.fr/apereo-cas
#cas.smsProvider.rest.url=http://esup-smsu-api-test.univ-rouen.fr/apereo-cas
cas.smsProvider.rest.basicAuthUsername=sms-cas-account
cas.smsProvider.rest.basicAuthPassword=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx


##############################
## MFA                      ##
##############################
# cas.authn.mfa.globalProviderId=mfa-esupotp
cas.authn.mfa.groovy-script.location=file:/etc/cas/config/mfaGroovyTrigger.groovy

# Add translations, you will need to check what are the default from CAS "Message Bundles"
properties

cas.messageBundle.baseNames=classpath:custom_messages,classpath:messages,classpath:esupotp_message


##############################
## MFA TRUSTED DEVICES      ##
##############################

cas.authn.mfa.trusted.mongo.clientUri=mongodb://localhost/cas-mongo-database
cas.authn.mfa.trusted.authenticationContextAttribute=isFromTrustedMultifactorAuthentication
cas.authn.mfa.trusted.deviceRegistrationEnabled=true
cas.authn.mfa.trusted.expiration=7
cas.authn.mfa.trusted.timeUnit=DAYS

cas.authn.mfa.trusted.crypto.enabled=true
cas.authn.mfa.trusted.crypto.encryption.key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
cas.authn.mfa.trusted.crypto.signing.key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
cas.authn.mfa.trusted.deviceFingerprint.cookie.crypto.encryption.key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
cas.authn.mfa.trusted.deviceFingerprint.cookie.crypto.signing.key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

agimus.properties

Pour la partie esup-agimus le fichier /etc/cas/config/agimus.properties contient les propriétés suivantes :

```
#########################
## Paramétrages AGIMUS ##
#########################

agimus.cookieName=AGIMUS
agimus.cookieMaxAge=259200
agimus.cookiePath=/
agimus.cookieValueMaxLength=10
agimus.cookieDomain=univ-rouen.fr
agimus.cookieValuePrefix=TRACEAGIMUS
agimus.traceFileSeparator=:
```

esupotp.properties

Les propriétés du module CAS esupotp-cas sont données dans le fichier /etc/cas/config/esupotp.properties :

```
esupotp.rank=0
esupotp.urlApi=https://esup-otp-api.univ-rouen.fr
esupotp.usersSecret=xxxxxxxxxxxxxx
esupotp.apiPassword=xxxxxxxxxxxxxx
esupotp.byPassIfNoEsupOtpMethodIsActive=false
esupotp.trustedDeviceEnabled=true
esupotp.isDeviceRegistrationRequired=false
```

MFA via trigger groovy

Le MFA est activé ou non suivant l'exécution d'un script groovy /etc/cas/config/mfaGroovyTrigger.groovy ainsi :

```
import java.util.*

class SampleGroovyEventResolver {
    def String run(service, registeredService, authentication, httpRequest, logger, ... other_args) {

        def mobile = authentication.principal.attributes.mobile
        def ip = httpRequest.getRemoteAddr()
        def memberOf = authentication.principal.attributes.memberOf

        /*
        logger.info("ip : [{}]", httpRequest.getRemoteAddr())
        logger.info("mobile : [{}]", mobile)
        logger.info("registeredService.id : [{}]", registeredService.id)
        */

        if ((int)registeredService.id in [22] && 'cn=from.grouper.admin,ou=groups,dc=univ-rouen,dc=fr' in
memberOf) {
            logger.warn("mfa required for grouper !", authentication.principal.id)
            return "mfa-esupotp"
        }

        if(!('cn=from.cas.otp,ou=groups,dc=univ-rouen,dc=fr' in memberOf)) {
            return null;
        }

        if ((int)registeredService.id in [12,13,14,18,21,22] && !ip.startsWith("10.0.1.")) {
            logger.warn("mfa for [{}] !", authentication.principal.id)
            return "mfa-esupotp"
        }

        if ((int)registeredService.id in [11, 18] && !ip.startsWith("10.0.1.") && 'cn=for.multipass.admin,
ou=groups,dc=univ-rouen,dc=fr' in memberOf) {
            logger.warn("mfa for [{}] !", authentication.principal.id)
            return "mfa-esupotp"
        }
        return null
    }
}
```

services

distinction d'un service provider shibboleth

Le service grouper qui est une application shibbolethisée est détecté via un service spécifique porté par /etc/cas/services/grouper_univ_rouen_fr-22.json :

```
{
    "@class" : "org.apereo.cas.services.RegexRegisteredService",
    "serviceId": "^https://idp\\.univ-rouen\\.fr/idp/Authn/External\\?conversation=[a-z0-9]
*&entityId=https://grouper\\.univ-rouen\\.fr",
    "name": "Université de Rouen Normandie",
    "informationUrl": null,
    "privacyUrl": null,
    "id": 22,
    "ssoEnabled" : "true",
    "description": "Gestion des groupes de l'Université de Rouen Normandie",
    "evaluationOrder":0,
    "usernameAttributeProvider":
    {
      "@class": "org.apereo.cas.services.PrincipalAttributeRegisteredServiceUsernameProvider",
      "canonicalizationMode": "NONE",
      "encryptUsername": "false",
      "usernameAttribute": "uid"
    }
  "attributeReleasePolicy" : {
    "@class" : "org.apereo.cas.services.ReturnAllowedAttributeReleasePolicy",
    "allowedAttributes" : [ "java.util.ArrayList", [ "uid", "mail", "displayName", "eduPersonPrincipalName",
"eduPersonAffiliation", "sn", "givenname", "radiusFilterId", "memberOf" ] ]
  }
    "proxyPolicy" : {
    "@class" : "org.apereo.cas.services.RegexMatchingRegisteredServiceProxyPolicy",
    "pattern" : "^https?://.*"
  }
}
```

Notez que pour que cette identification puisse fonctionner ainsi nous avons configuré shib-cas-authn au niveau de l'**IdP** avec

```
 shibcas.entityIdLocation=embed
```

La désacivtation des sessions au niveau de l'**IdP** est également nécessaire si on souhaite qu'effectivement chaque authentification d'un SP auprès de l'IdP soit réévaluée côté CAS (pour activation ou non du MFA dans notre cas) :

```
  idp.session.enabled = false
```

service proxy-cas avec clearpass

Pour l'ENT avec la partie esup-filemanager qui utilise les mécanismes de proxy-cas / clearpass, la fonctionnalité du passage de mot de passe en tant qu'attribut (chiffré) dit "clearpass" est configurée ainsi dans le fichier /etc/cas/services/esup_filemanager_univ_rouen_fr-2.json

```
{
        "@class" : "org.apereo.cas.services.RegexRegisteredService",
    "serviceId": "^https?://([A-Za-z0-9_-]+\\.)*normandie-univ\\.fr(/.*)?",
    "name": "Service opéré par la COMUE",
    "informationUrl": null,
    "privacyUrl": null,
    "id": 3,
    "description": "Vous avez demandé à vous connecter à un service proposé par la COMUE Normandie Université.",
        "evaluationOrder":3,
    "usernameAttributeProvider":
    {
      "@class": "org.apereo.cas.services.PrincipalAttributeRegisteredServiceUsernameProvider",
      "canonicalizationMode": "NONE",
      "encryptUsername": "false",
      "usernameAttribute": "uid"
    }
    "proxyPolicy" : {
    "@class" : "org.apereo.cas.services.RegexMatchingRegisteredServiceProxyPolicy",
    "pattern" : "^https?://.*"
    },
    "attributeReleasePolicy" : {
        "@class" : "org.apereo.cas.services.ReturnAllowedAttributeReleasePolicy",
        "authorizedToReleaseCredentialPassword" : true,
    },
    "publicKey" : {
        "@class" : "org.apereo.cas.services.RegisteredServicePublicKeyImpl",
        "location" : "/etc/cas/univ-rouen-esup-filemanager-public.key",
        "algorithm" : "RSA"
    }
}
```

Au niveau d'esup-filemanager, la configuration d'une authentification via ce mécanisme se fait alors ainsi :

```
  <bean name="univ_rouen_cas_clearpass_auth" class="org.esupportail.portlet.filemanager.services.auth.cas.
ClearPassUserCasAuthenticatorService"
    scope="session">
        <property name="domain" value="ur"/>
      <property name="userCasAuthenticatorServiceRoot" ref="casUserAuthenticationServiceRoot"/>
      <property name="pkcs8Key" value="/opt/tomcat-esup/webapps/esup-filemanager/WEB-INF/classes/univ-rouen-
esup-filemanager-private.p8"/>
  </bean>
```

Rappel : la mise en oeuvre côté CAS (dont la génération des clefs) est documentée dans la documentation CAS - clearpass.

ldap policy

A noter l'usage de ldap policy à l'Université de Rouen Normandie : on ajoute l'attribut ldap PwdAccountLockedTime pour verrouiller les comptes détectés comme corrompus de manière automatique (via l'usage de fail2ban notamment).

Apereo CAS avec cas-server-support-ldap supporte en effet les codes d'erreur portés par ldap policy sur les bind ldap.

Pb de lisibilité des entrées username/password du formulaire de login dans Rocket.Chat

Cf la copie d'écran ci-dessous, le formulaire d'authentification de CAS 6.4 pose un problème de lisibilité au travers du client lourd Rocket.Chat.

La cause est indéterminée, le problème est sans doute lié à electron utilisé par le client lourd Rocket.Chat.

Pour contourner la chose, et en s'inspirant de https://github.com/material-components/material-components-web/issues/4447 nous avons ajouté le code javascript suivant dans notre footer.html :

```
window.setTimeout(() => {
  document
    .querySelectorAll('.mdc-text-field__input')
    .forEach(el => {
      const textField = el.parentNode;
      const label = textField.querySelector('.mdc-floating-label');
      const spanOutline = textField.querySelector('.mdc-notched-outline');
      if (label) {
        label.classList.add('mdc-floating-label--float-above');
      }
      if (spanOutline) {
        spanOutline.classList.add('mdc-notched-outline--notched');
      }
      if (textField.MDCTextField) {
        textField.MDCTextField.foundation_.notchOutline(true);
      }
    });
}, 300);
```

Ainsi au bout de 300ms les labels **Identifiant** / **Mot de passe** se positionnent au dessus par défaut et ne restent donc pas/plus en placeholder.

Le comportement global est un peu moins ergonomique mais ne pose plus de problème dans Rocket.Chat ainsi que dans d'autres contextes d'usage (utilisation d'extension navigateur comme Bitwarden par exemple).

Suppression des sessions CAS d'un utilisateur (en cas de phishing ou autre)

CAS 6.4 propose via un actuator endpoint (API REST poussée par spring boot) la possibilité de "détruire" une session d'un utilisateur via un appel REST.

Cet appel direct est beaucoup plus performant que ce qu'on pouvait faire avec la 6.0.

De plus, CAS déclenche également maintenant la procédure de Single LogOut (SLO) au travers de cet appel (attention toutefois à prendre une version > 6.4.5 pour que le TGT soit bien supprimé / bug introduit en cours de 6.4).

On implémente maintenant la destruction des sessions CAS d'un utilisateur depuis notre application de gestion de comptes (codée en java/spring) ainsi :

```
public class CasService {

        protected Logger log = Logger.getLogger(CasService.class);

        RestTemplate restTemplate;

        String casSsoSessionsUrl;

        public void setRestTemplate(RestTemplate restTemplate) {
                this.restTemplate = restTemplate;
        }

        public void setCasUrl(String casUrl) {
                casSsoSessionsUrl = casUrl + "/actuator/ssoSessions?type={type}&username={username}";
        }

        public synchronized String destroySsoSessions(String login) {
                log.info(String.format("Call Cas Destroy tickets for user %s", login));
                Map<String, String> urlVariables = new HashMap<String, String>();
                urlVariables.put("type", "ALL");
                urlVariables.put("username", login);
                ResponseEntity resp = restTemplate.exchange(casSsoSessionsUrl, HttpMethod.DELETE, null, String.
class, urlVariables);
                String msg = String.format("Cas Destroyed tickets of user %s - resp : %s", login, resp.
getBody());
                log.info(msg);
                return msg;
        }
}
```

Note : pour que la récupération du JSON de retour puisse se faire en simple String comme proposé ici, on a un RestTemplate défini avec en messageConverter org.springframework.http.converter.StringHttpMessageConverter