Implementing CAS Authentication for Grouper

There are different ways of enabling CAS authentication to protect Grouper resources. These have been tested with Grouper 2.4.0 primarily as a proof of concept, although there is some reports of success from similar configurations based on these.

Here is most recent CAS Grouper doc

Method 1: Tomcat Container Authentication

See also: https://github.com/apereo/java-cas-client

The context definition in server.xml for Tomcat looks like this:

```
<Context docBase="/ucd/opt/grouper-ui/dist/grouper" path="/grouper"
    reloadable="false" mapperContextRootRedirectEnabled="true" mapperDirectoryRedirectEnabled="true">


  <Realm className="org.jasig.cas.client.tomcat.v85.PropertiesCasRealm"
     propertiesFilePath="/etc/tomcat/grouper-users.properties"
   />

        <!--
       If you do not need to map users to roles via a grouper-users.properties file use this.
       <Realm className="org.jasig.cas.client.tomcat.v85.AssertionCasRealm" />
        -->

  <Valve className="org.jasig.cas.client.tomcat.v85.Cas20CasAuthenticator"
     encoding="UTF-8"
     casServerLoginUrl="https://CAS_SERVER/cas/login"
     casServerUrlPrefix="https://CAS_SERVER/cas/"
     serverName="GROUPER_SERVER"
   />


  <!-- Single sign-out support -->
  <Valve className="org.jasig.cas.client.tomcat.v85.SingleSignOutValve"
    artifactParameterName="SAMLart"
  />
</Context>
```

The following jar files will need to go into the Tomcat lib directory (with current versions as of May 2019):

- org.jasig.cas.client : cas-client-core (v3.5.1) [Download]
- org.jasig.cas.client : cas-client-integration-tomcat-common (v3.5.1) [Download]
- org.jasig.cas.client : cas-client-integration-tomcat-v85 (v3.5.1) [Download]

- org.slf4j : slf4j-api (v1.7.26) [Download]

In Grouper's WEB-INF/web.xml, comment out the login-config and security-role sections. The security-constraint sections should remain so that authentication is triggered. The role-name can be changed to "*" (or "**" if that doesn't work) to allow all validated users to log in.

For other versions of Tomcat, change v85 to v8, v7, or v6 as appropriate.

Method 2: Client Configuration Using web.xml

This method makes changes solely within the Grouper web application, without affecting the Tomcat configuration.

1. Download the cas-client-core jar file (current version is cas-client-core-3.5.1.jar as of May 2019)

2. Copy the web applications top-level index.jsp to a new subdirectory cas/

3. Edit cas/index.jsp to reference parent directory instead of the current one

```
<%@ include file="../WEB-INF/grouperUi2/assetsJsp/commonTaglib.jsp"%>
String location="../grouperUi/app/UiV2Main.index?operation=UiV2Main.indexMain";
```

4. Add to WEB-INF/classes/Owasp.CsrfGuard.overlay.properties

```
org.owasp.csrfguard.unprotected.CASLogin=%servletContext%/cas/*
```

5. Add to WEB-INF/web.xml, changing parameters as needed. Based on https://apereo.atlassian.net/wiki/spaces/CASC/pages/103252594/Configuring+the+Jasig+CAS+Client+for+Java+in+the+web.xml

```
<filter>
  <filter-name>CAS Authentication Filter</filter-name>
  <filter-class>org.jasig.cas.client.authentication.AuthenticationFilter</filter-class>
  <init-param>
    <param-name>casServerUrlPrefix</param-name>
    <param-value>http://localhost:8080/cas</param-value>
  </init-param>
  <init-param>
    <param-name>serverName</param-name>
    <param-value>http://localhost:8080</param-value>
  </init-param>
</filter>
<filter-mapping>
    <filter-name>CAS Authentication Filter</filter-name>
    <url-pattern>/cas/*</url-pattern>
</filter-mapping>
<filter>
  <filter-name>CAS Validation Filter</filter-name>
  <filter-class>org.jasig.cas.client.validation.Cas20ProxyReceivingTicketValidationFilter</filter-class>
  <init-param>
    <param-name>casServerUrlPrefix</param-name>
    <param-value>http://localhost:8080/cas</param-value>
  </init-param>
  <init-param>
    <param-name>serverName</param-name>
    <param-value>http://localhost:8080</param-value>
  </init-param>
</filter>
<filter-mapping>
    <filter-name>CAS Validation Filter</filter-name>
    <url-pattern>/cas/*</url-pattern>
</filter-mapping>
<filter>
  <filter-name>CAS HttpServletRequest Wrapper Filter</filter-name>
  <filter-class>org.jasig.cas.client.util.HttpServletRequestWrapperFilter</filter-class>
</filter>
<filter-mapping>
    <filter-name>CAS HttpServletRequest Wrapper Filter</filter-name>
    <url-pattern>/cas/*</url-pattern>
</filter-mapping>
```

6. Start Tomcat, check catalina and localhost logs if any startup errors

7. Go to URI /grouper/cas/index.jsp to trigger the start of a CAS session.

Previous way to integrate CAS with Grouper (< 2.4.0)

The yale-cas-auth java jar file is included with the installation of the Grouper UI. There are a few steps we needed to implement it:

Configuration Steps to enable CAS Authentication

1. Add the cas authentication piece to the build.xml file in the Grouper UI home/build folder: /deploy/AppServers/grouper-ui folder:

/deploy/AppServers/grouper-ui/build.xml

```
<ant antfile="build.xml" target="webapp" dir="${contrib.dir}/yale-cas-auth" inheritrefs="true" />
```

It should go just below the following section in the build.xml file:

```
<!-- Call any site specific build script. This may be used to introduce site specific Struts action,
                            local Subject implementations etc -->
                <antcall target="-additional-build">
                        <param name="target" value="webapp"/>
                                                <reference refid="ui.class.path.for.run"/>
                </antcall>
```


Implementer note: There are several *-additional-build* sections. Ensure you find the one that has a *target* with a value of "webapp".

2. Modify the following 3 lines in the build.properties file that is in the yale-cas-auth folder -- enter proper URLs for your organization:

/deploy/AppServers/grouper-ui/contrib/yale-cas-auth/build.properties

```
#Grouper CAS Integration for CalPoly
sso.login.url=https://mydev.YourCampus.edu/cas/login
sso.validate.url=https://mydev.YourCampus.edu:443/cas/serviceValidate
grouper.server.name=s-grouper.its.YourCampus.edu
```

3. Modify the struts-config.xml file to skip the login prompt by changing the callLogin path to home.do instead of login.do:

/deploy/AppServers/grouper-ui/webapp/WEB-INF/struts-config.xml

```
<forward name="callLogin" path="/home.do" redirect="true"/>
```

4. Ensure the REMOTE_USER value that is returned from CAS is configured as one of the subject identifiers in the sources.xml:

/deploy/AppServers/grouper/conf/sources.xml

```
<init-param>
  <!-- col which identifies the row, perhaps not subjectId -->
  <param-name>subjectIdentifierCol0</param-name>
  <param-value>SUBJECT_NAME</param-value>
</init-param>
```


This is based on using the GrouperJdbcSourceAdapter2 source adapter type

5. Ensure the grouper URL has been added to the CAS Services Registry.

Deployment Steps

1. From the /deploy/AppServers/grouper-ui directory, create a new war file:

```
ant war
```

2. Remove the grouper directory in the $TOMCAT_HOME/webapps folder:

```
cd /deploy/AppServers/tomcat/webapps
rm -rf grouper
```

3. Copy the new war file to the webapps directory (overwrite existing grouper.war file):

```
cp /deploy/AppServers/grouper-ui/dist/grouper.war .
```

4. Stop and restart Tomcat.

Troubleshoot

See debug information in logs in log4j.properties

```
log4j.logger.edu.internet2.middleware.grouper.ui.GrouperUiFilter = DEBUG
```