VIP ANR-09-COSI-03
Virtual Imaging Platform

D2.4.1: VIP integration in Maat-G gateway

Tristan Glatard, Baptiste Grenier, Jérôme Revillard and Rafael Ferreira da Silva

Abstract

This deliverable reports on the integration of the VIP portal in Maat's gateway. The authentication system used in Maat's gateway, namely the Central Authentication Service (CAS) was enabled in VIP. As a result, users of Maat gateways can transparently login to VIP. This enables further exploitation of the VIP platform through Maat's gateway.

Contents

1 Introduction

This deliverable reports on the integration of the VIP portal (VIP client, see *D2.3.4: VIP client to semantic and execution services*) in Maat's gateway. The motivation for this work is to provide some sustainability to VIP by technically enabling Maat to reuse VIP in other solutions.

In practice, some effort was devoted towards the harmonization of authentication systems. As a result, users with an account on the Maat gateway can transparently connect to VIP. The portal was also integrated in Maat's gateway as an `iframe`.

2 Authentication in VIP portal

Authentication in VIP follows a classical 3-tier model:

- a database stores user credentials;
- the server application can check the database and set session variables in the browser;
- the browser holds session variables.

In essence, the `authenticate()` method on Figure 1 is implemented by the server to check user authentication. Is is called by every server-side operation.

3 Authentication in Maat's gateway

Maat's gateway uses the Central Authentication Service (CAS[1]) to authenticate its users. It is a single sign-on protocol where multiple web applications can refer to a single server to authenticate users without having access to their passwords.

When a web application needs to authenticate a client, it redirects it to a CAS server that checks the client's username and password, and returns a *ticket* to the application. The application can then validate the ticket by contacting the CAS server. More information about the client (e.g., email and other parameters) can also be returned in the ticket.

[1] http://www.jasig.org/cas

```
boolean authenticate(){
   if (browser.getVIPSession() == null)
        //No VIP session. Redirecting to login page.
        return  login();
    else{
       //A session exists. Is it valid?
       if(database.isValid(browser.getVIPSession())){
         //Valid session exists. User is authenticated.
         return true;
       } else
           //Redirect to login page
           return login();
}

boolean login(){
     //Ask login/password.
     (username, password) = browser.askUserCredentials();
      if(database.isValid(username,password)){
           //Create VIP session.
           database.storeSession(UID,user);
           browser.setVIPSession(UID);
           return true;
       } else {
          //Redirect to login page again
          return login();
       }
}
```

Figure 1: Pseudo-code of the method implemented by the VIP server to authenticate users. `database` and `browser` are eponymous software entities.

4 Integration

The integration of the two authentication systems in VIP was done along the following principles:

- The authentication mechanism must be transparent to the users, i.e., it should be automatically inferred from the user context;
- No extra account creation should be necessary for CAS users.

The following solutions were adopted:

- CAS authentication is specified in the URL used to access VIP, for instance `http://vip.creatis.insa-lyon.fr/?login=CAS`. This URL is used to refer to VIP from within Maat's gateway.
- Accounts are created and validated automatically for CAS users.

The pseudo-code of the resulting authentication algorithm is detailed on Fig. 2.

The implementation is done in Java using version `3.2.1` of package `org.jasig.cas.client`. The key lines used for CAS ticket validation are shown below:

```
private User getCASUser(String ticket, URL serviceURL) {
if (ticket != null) {
  Saml11TicketValidator sv =
    new Saml11TicketValidator(Server.getInstance().getCasURL());
  Assertion a;
  try {
      a = sv.validate(ticket, serviceURL.toString());
  } catch (TicketValidationException ex) {
      return null;
  }
  String email=
    (a.getPrincipal().getAttributes().get(``USER_EMAIL_ADDRESS'')).toString();
    ... //get or create user
  return user;
}
```

Finally, VIP was integrated in Maat's gateway as an `iframe`, using the CAS authentication URL mentioned above. As a result, Maat users can transparently connect to VIP, as shown on Fig. 3.

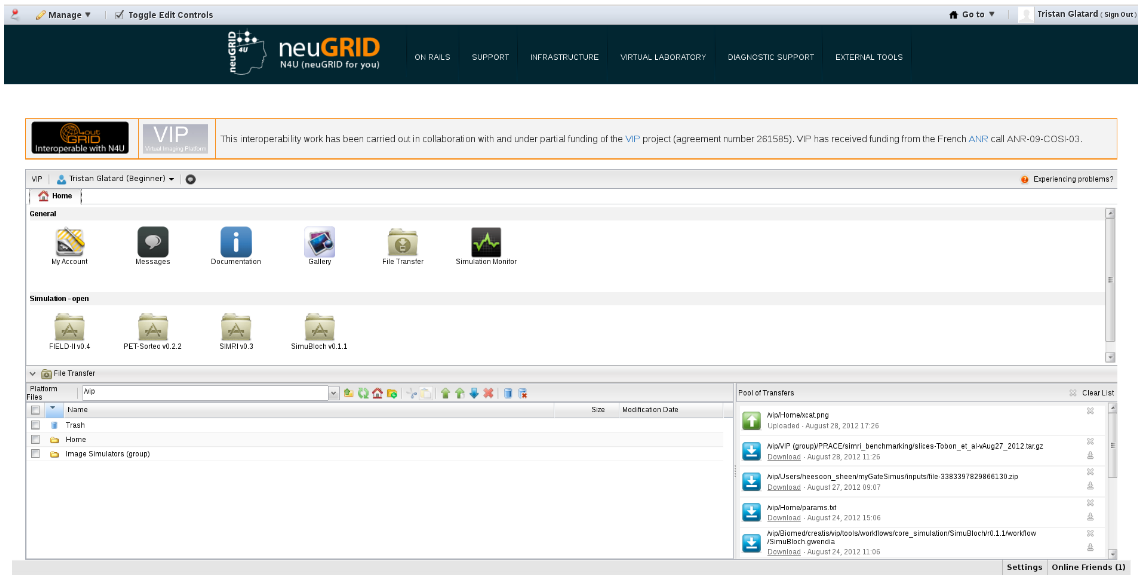
5 Conclusion

The VIP portal can now be accessed transparently by users of Maat's gateways, without creating a new account. Technically, VIP could therefore be exploited by Maat in other projects.

```
boolean authenticate(){
   if  (browser.getURL().contains(``CASN4U'')
                        || browser.getURL().getCASTicket() != null){
   //CAS authentication method is requested
      if(!browser.getURL().getCASTicket().valid()){
         //CAS ticket is not valid. Redirecting to CAS login page
         return browser.showCASLogin();
      } else{
          //CAS ticket is valid
          username = casServer.getUserName(browser.getURL().getCASTicket());
          //Automatically create new user if user doesn't exist
          if(!database.exist(userName))
             database.createUser(userName);
          //Set VIP session
          database.storeSession(UID,userName);
          browser.setVIPSession(UID);
          return true;
      }
   }
   //Regular VIP authentication
   if (browser.getVIPSession() == null)
        //No VIP session. Redirecting to login page.
        return  login();
    else{
       //A session exists. Is it valid?
       if(database.isValid(browser.getVIPSession())){
         //Valid session exists. User is authenticated.
         return true;
       } else
           //Redirect to login page
           return login();
}

boolean login(){
     //Ask login/password.
     (username, password) = browser.askUserCredentials();
      if(database.isValid(username,password)){
           //Create VIP session.
           database.storeSession(UID,user);
           browser.setVIPSession(UID);
           return true;
       } else {
          //Redirect to login page again
          return login();
       }
}
```

Figure 2: Pseudo-code of the authentication method supporting CAS in VIP.



This interoperability work has been carried out in collaboration with and under partial funding of the VIP project (agreement number 261585). VIP has received funding from the French ANR call ANR-09-COSI-03.

Name	Size	Modification Date
Trash		
Home		
Image Simulators (group)		

Pool of Transfers — Clear List

/vip/Home/xcat.png
Uploaded - August 28, 2012 17:26

/vip/VIP (group)/PRACE/simri_benchmarking/slices-Tobon_et_al-vAug27_2012.tar.gz
Download - August 28, 2012 11:26

/vip/Users/heesoon_sheen/myGateSimus/inputs/file-3383397829866130.zip
Download - August 27, 2012 09:07

/vip/Home/params.txt
Download - August 24, 2012 15:06

/vip/Biomed/creatis/vip/tools/workflows/core_simulation/SimuBloch/r0.1.1/workflow
/SimuBloch.gwendia
Download - August 24, 2012 11:06

Figure 3: VIP portal, as integrated in the Maat-G gateway (example on the Neugrid gateway). VIP accounts are automatically created from Maat accounts.