Open-source Single Sign-On with CAS (Central Authentication Service)

Pascal Aubry, Vincent Mathieu & Julien Marchal

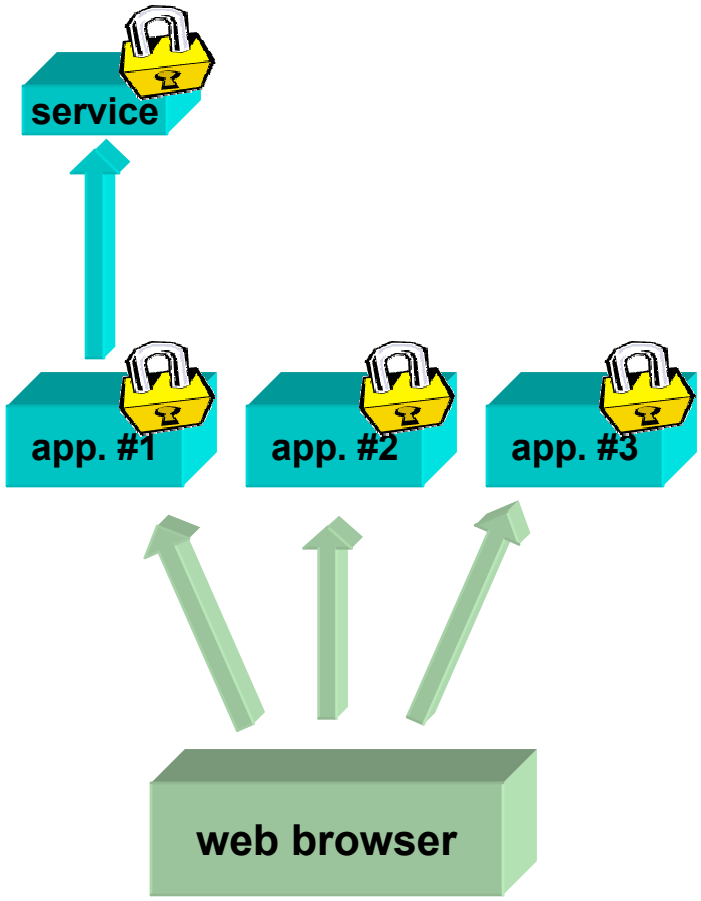
Open-source Single Sign-On with CAS

- **Single Sign-On**
  - Why SSO?
  - The main principles of web SSO
  - The choice of CAS
- **CAS (Central Authentication Service)**
  - How does it work?
  - How to CAS-ify applications
    - Web applications
    - Non-web applications
- **Limits**
- **The effort of the ESUP-Portail consortium around CAS**

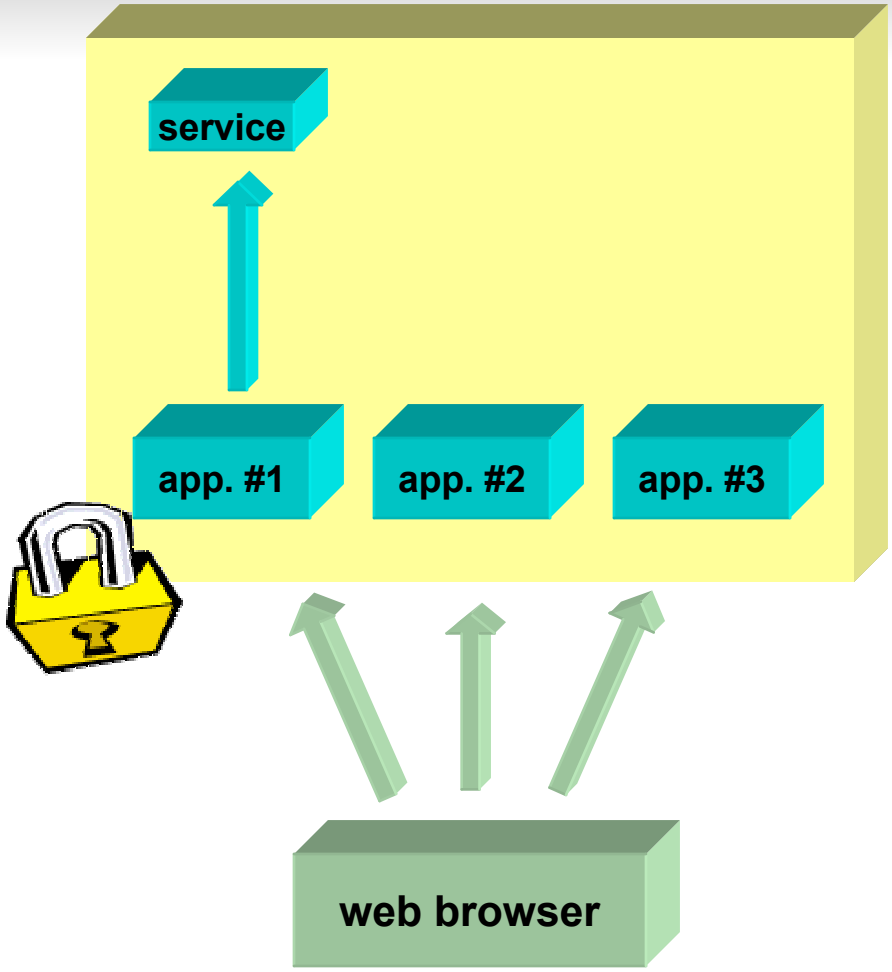
Why Single Sign-On?

- **Unique accounts but several authentications**
  - Each time users access an application
- **Security (password stealing)**
  - Protect password transmission
  - Do not transmit passwords to applications
    - Simplify applications
    - Delegate developments without delegating authentication
- **Abstract authentication**
  - LDAP, NIS, database, NT, Active Directory, X509 certificates, …

SSO: the user's point of view



without SSO



with SSO

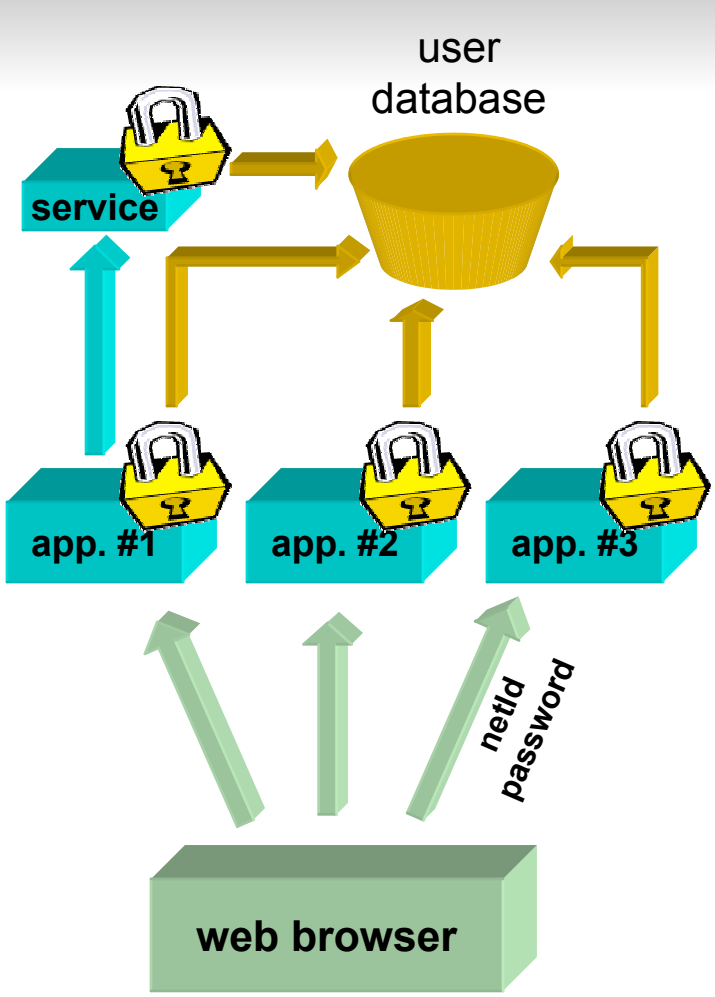
SSO: principles on the web

- **Authentication is centralized**
  - One (redundant) authentication server
- **Transparent HTTP redirections**
  - From applications to the authentication server (when not authenticated)
  - From the authentication server to applications (when authenticated)
- **Tokens propagate identities**
  - Cookies, CGI parameters

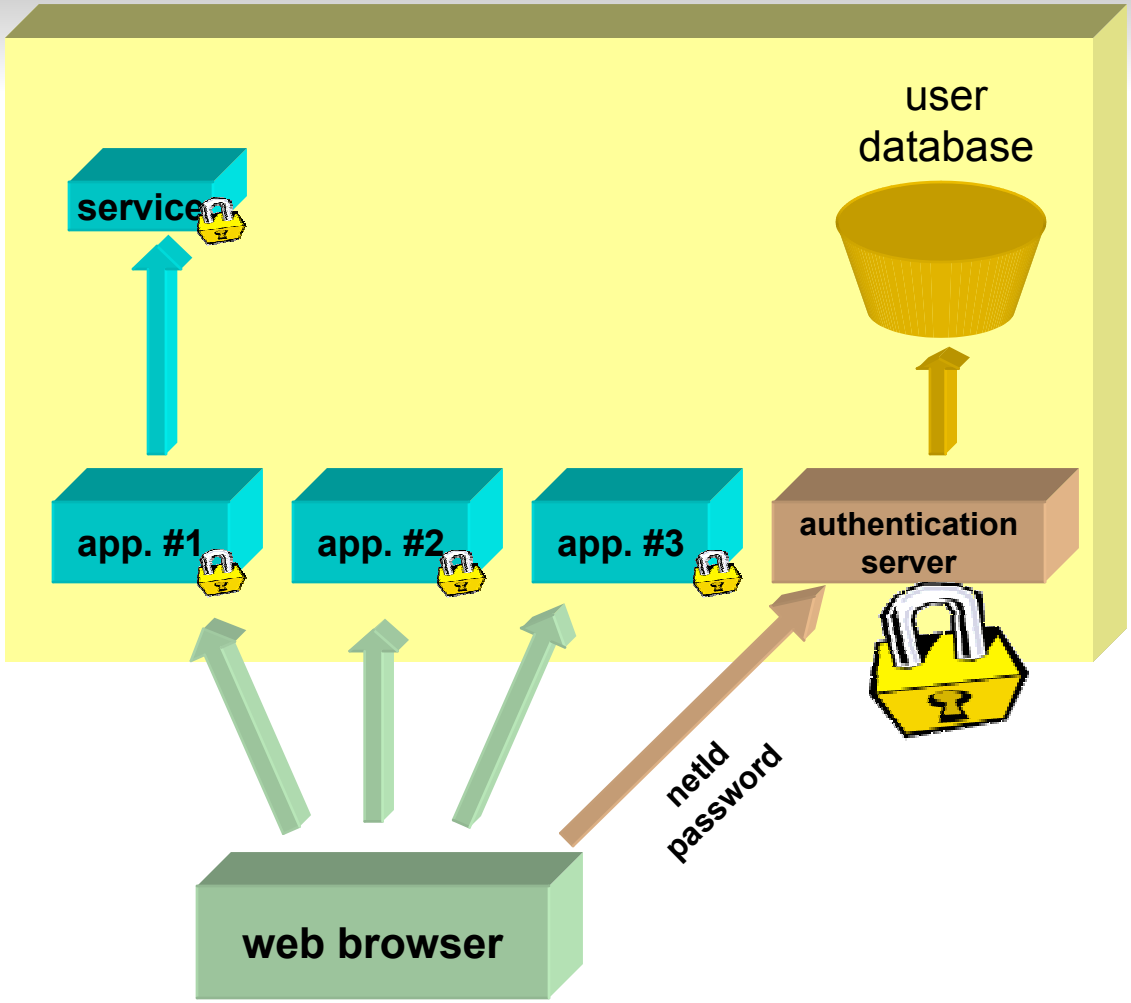
CAS: why did we choose it?

- **Security**
  - Password is never transmitted to applications
  - Opaque tickets are used
- **N-tier installations**
  - Without transmitting any password!
- **Portability (client libraries)**
  - Java, Perl, JSP, ASP, PHP, PL/SQL, Apache and PAM modules
- **Permanence**
  - Developed by Yale University
  - World-wide used (mainly Universities)
  - Adopted by all the French educational community
- **J2EE platform**
  - Very light code (about 1000 lines)
- **Open source**
- **Integrated into uPortal**

CAS: why did we choose it?

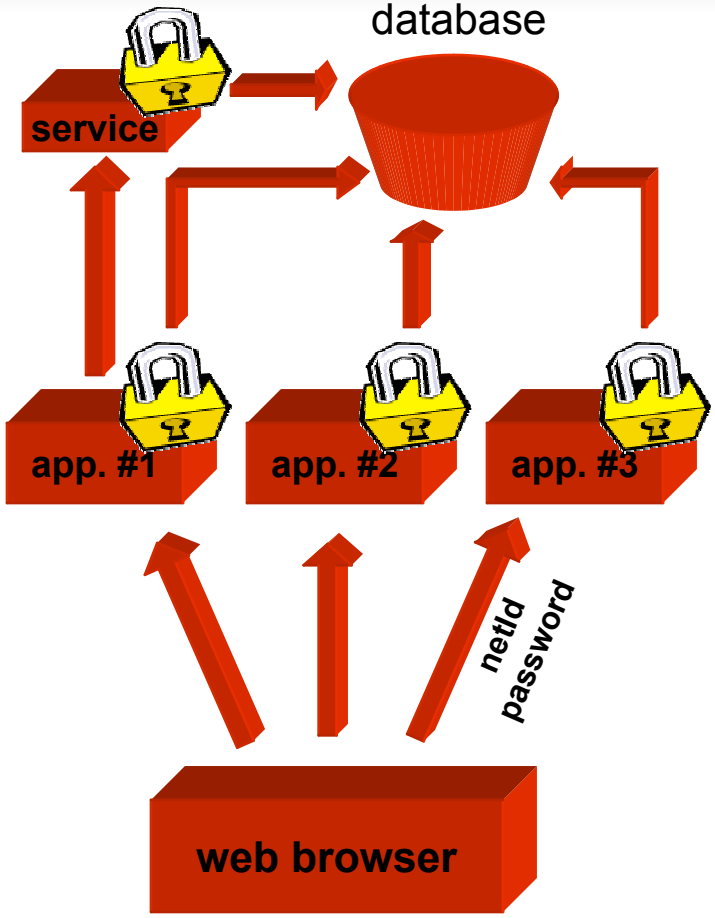


without SSO

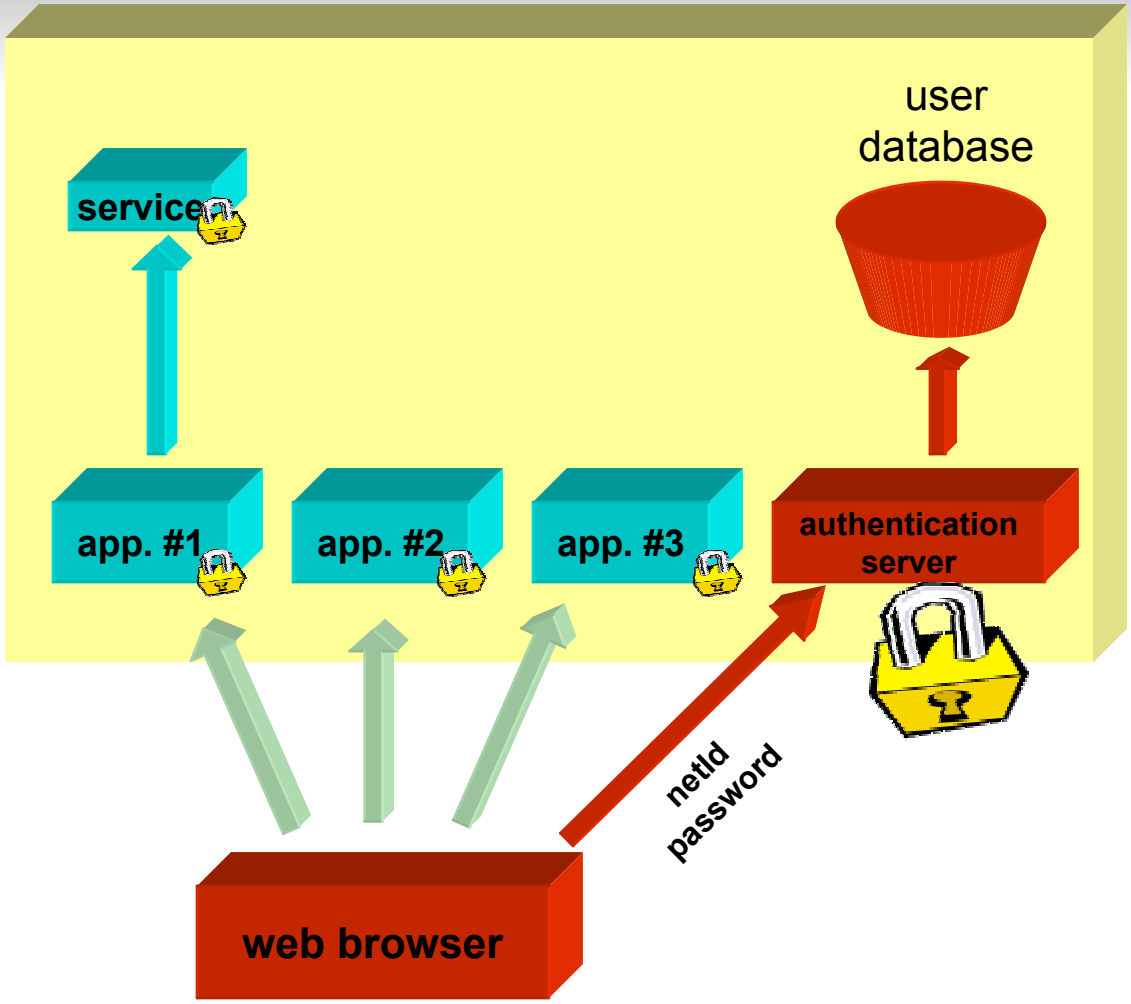


with CAS

CAS: why did we choose it?

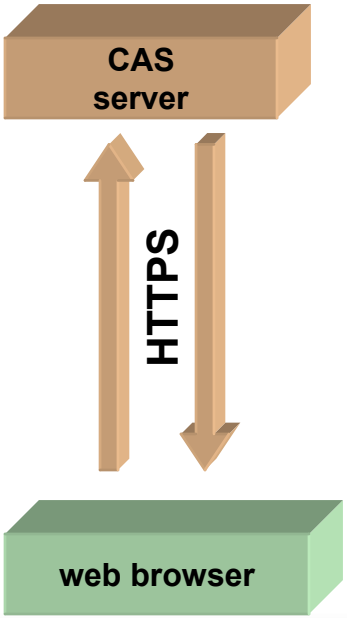


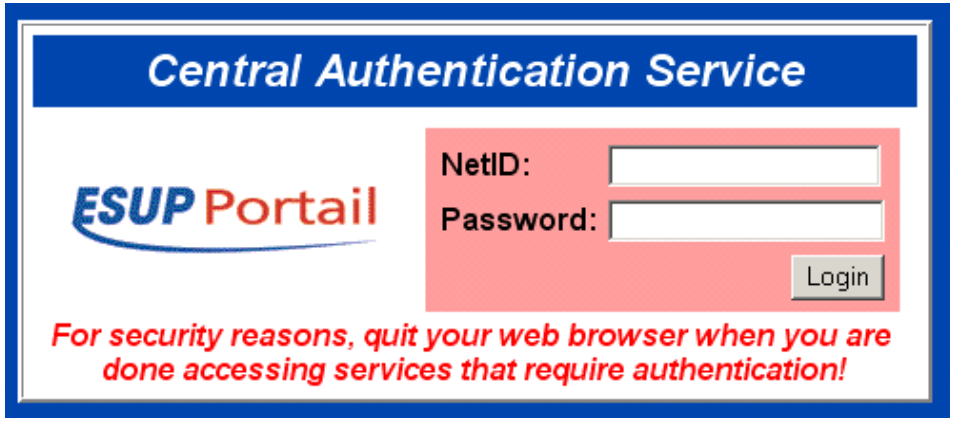
without SSO



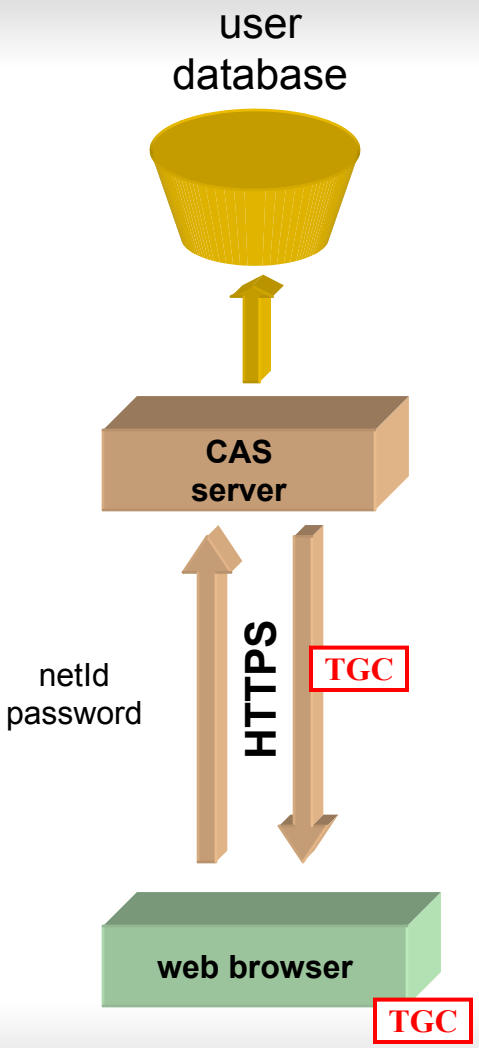
with CAS

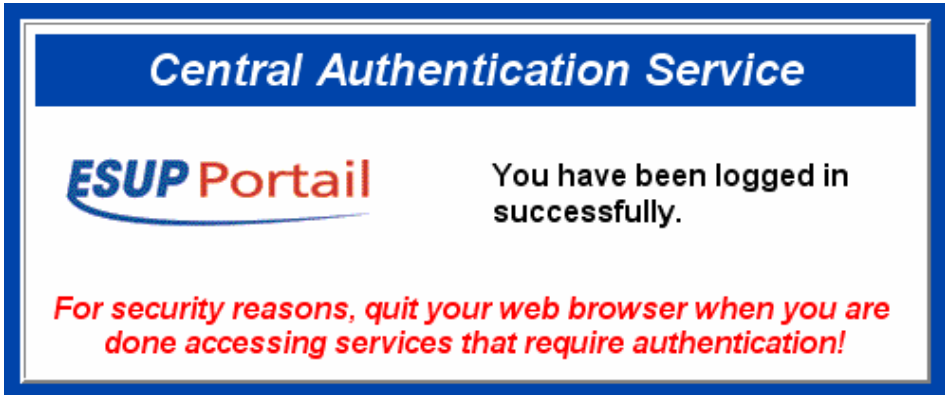
User authentication





User authentication

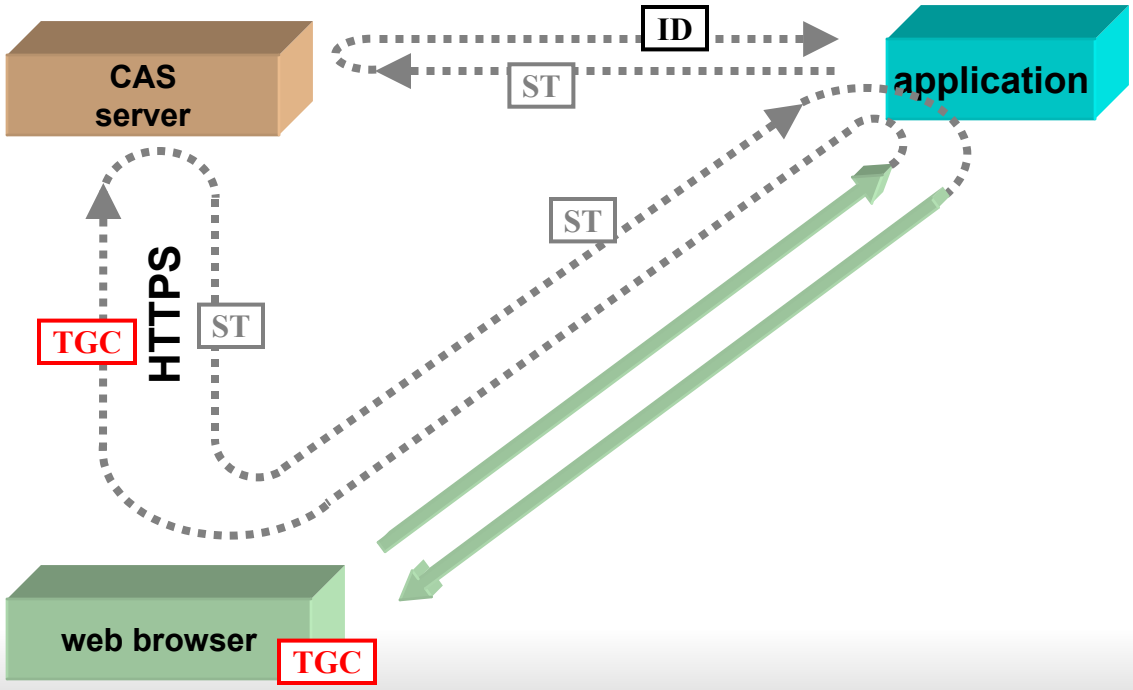




- **TGC: Ticket Granting Cookie**
  - User's passport to the CAS server
  - Private and protected cookie (the only one used by CAS, optional)
  - Opaque re-playable ticket

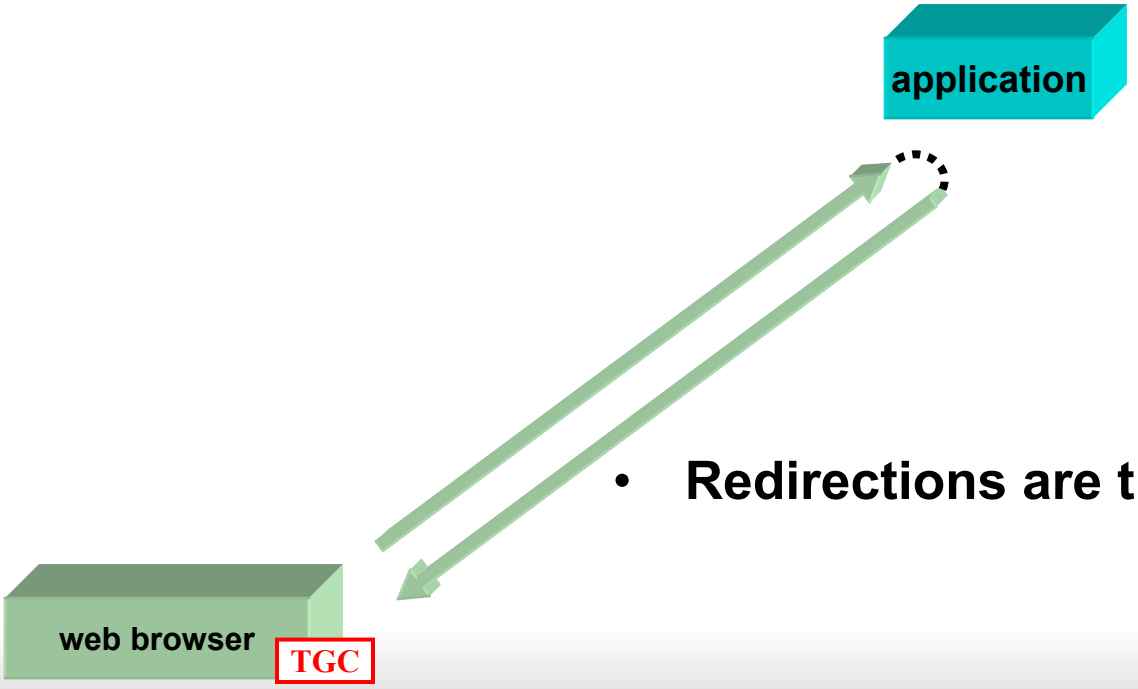
Accessing an application after authentication

- **ST: Service Ticket**
  - Browser's passport to the CAS client (application)
  - Opaque and non re-playable ticket
  - Very limited validity (a few seconds)



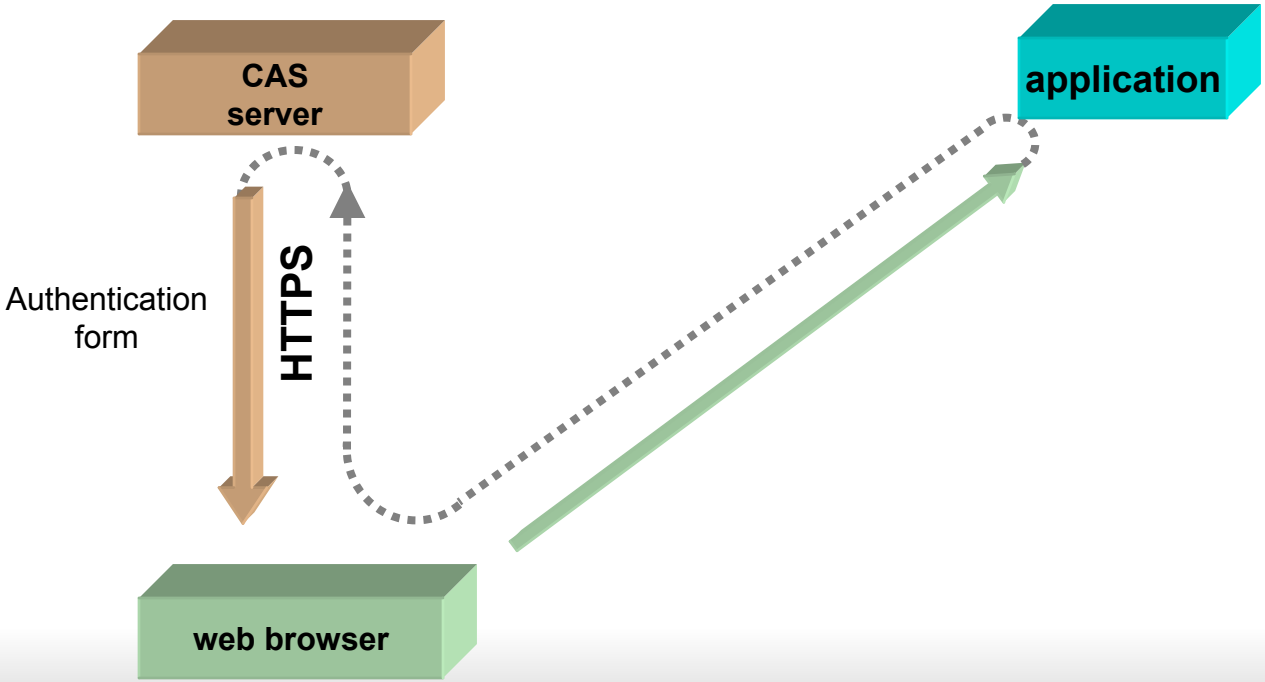
Accessing an application after authentication

- **ST: Service Ticket**
  - Browser's passport to the CAS client (application)
  - Opaque and non re-playable ticket
  - Very limited validity (a few seconds)

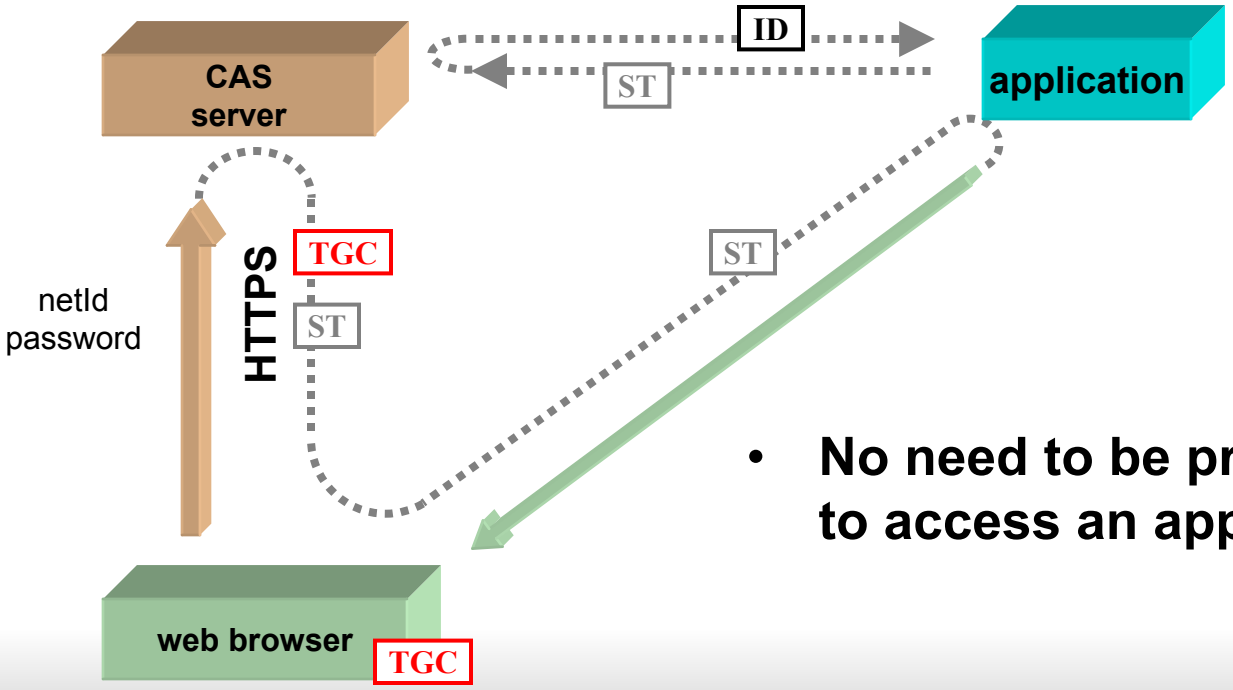


- **Redirections are transparent to users**

Accessing an application without authentication



Accessing an application without authentication



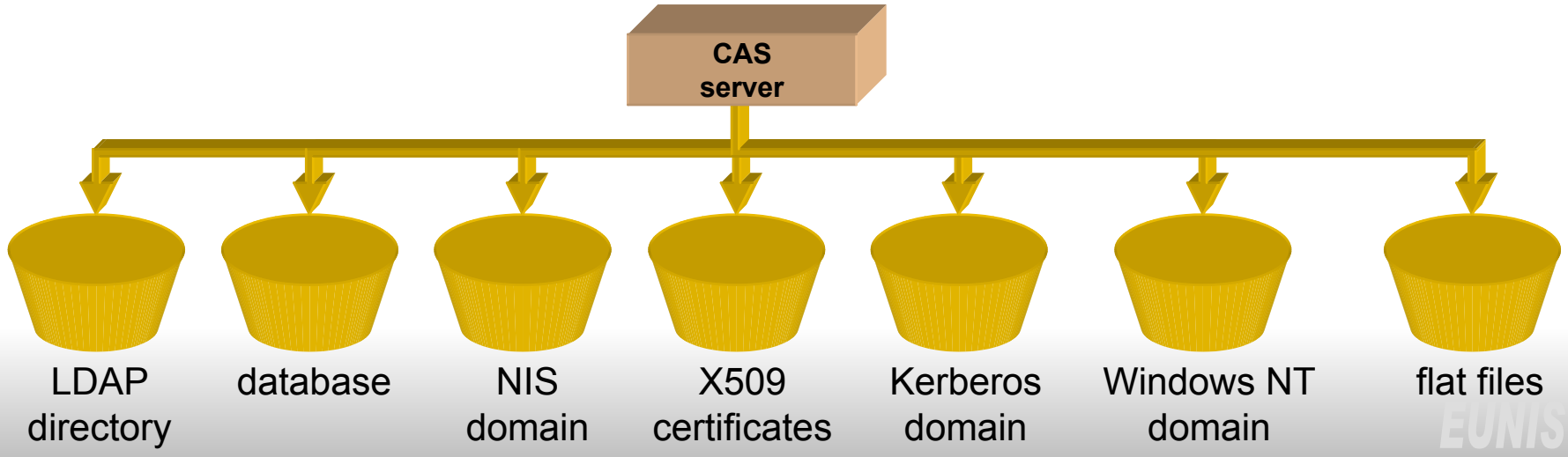
- **No need to be previously authenticated to access an application**

Remarks

- **Once a TGC acquired, authentication is transparent for the access to any CAS-ified application of the workspace**
- **Once authenticated by an application, a session should be used between the browser and the application**

Authenticating users with CAS

- **CAS authentication left to administrators**
- **ESUP-Portail CAS Generic Handler**
  - Mixed authentication
  - XML configuration



Using the ESUP-Portail CAS GH

```
<authentication debug="on">
  <handler>
    <classname>
      org.esupportail.cas.server.handlers.ldap.FastBindLdapHandler
    </classname>
    <config>
      <filter>uid=%u,ou=people,dc=esup-portail,dc=org</filter>
      <server>
        <url>ldap://ldap.esup-portail.org</url>
      </server>
    </config>
  </handler>
  <handler>
    <classname>
      org.esupportail.cas.server.handlers.nis.NisHandler
    </classname>
    <config>
      <domain>ESUP-PORTAIL</domain>
      <encryption>pammd5</encryption>
      <server>
        <host>nismaster.esup-portail.org</host>
        <host>nisslave.esup-portail.org</host>
      </server>
    </config>
  </handler>
</authentication>
```

CAS-ifying a web application

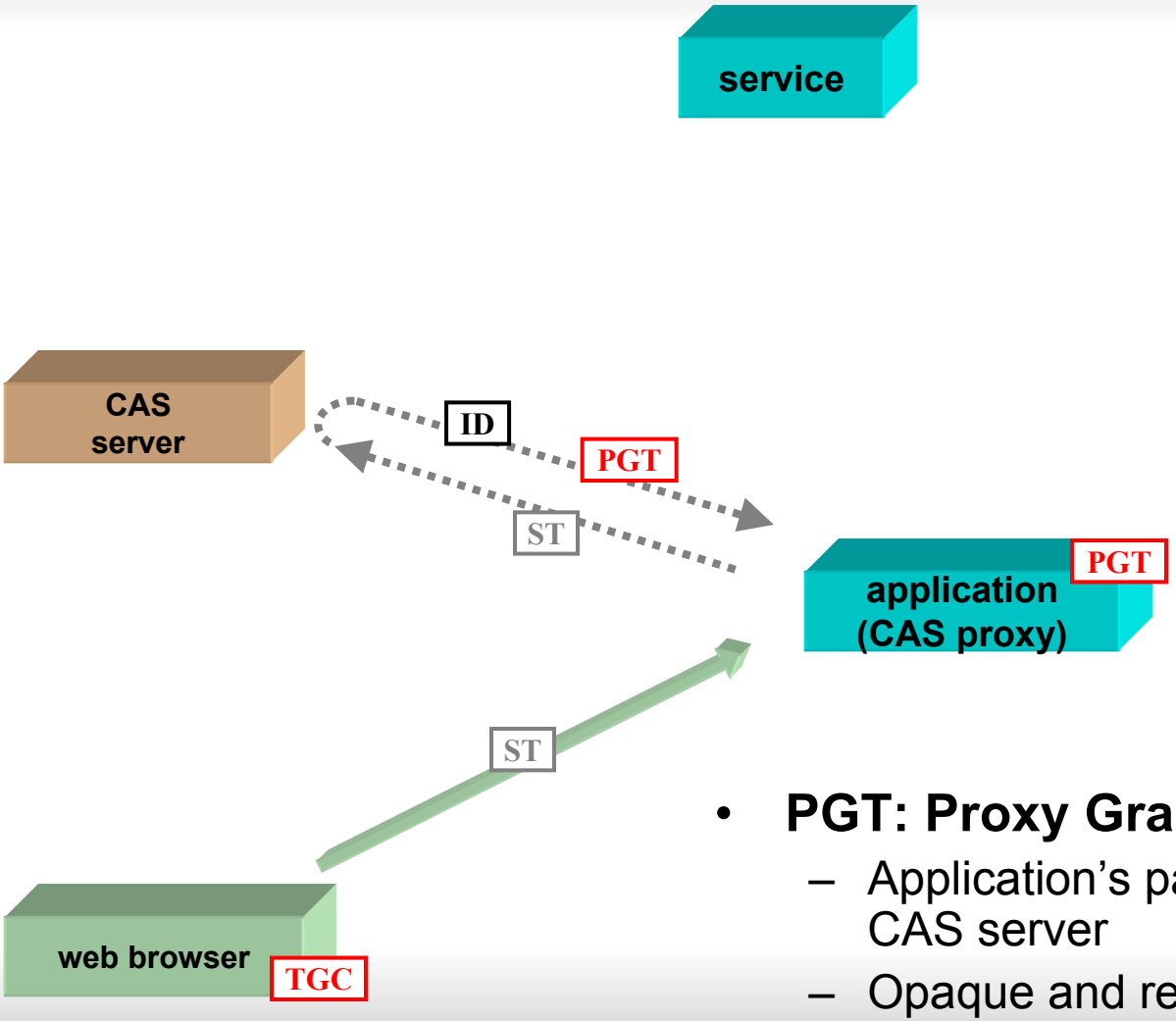
- **Use provided libraries**
- **Add a few lines of code**
- **Note: you can also protect static resources**
  - With mod_cas, an Apache module

CAS-ifying a web application

- **An example using phpCAS (ESUP-Portail)**

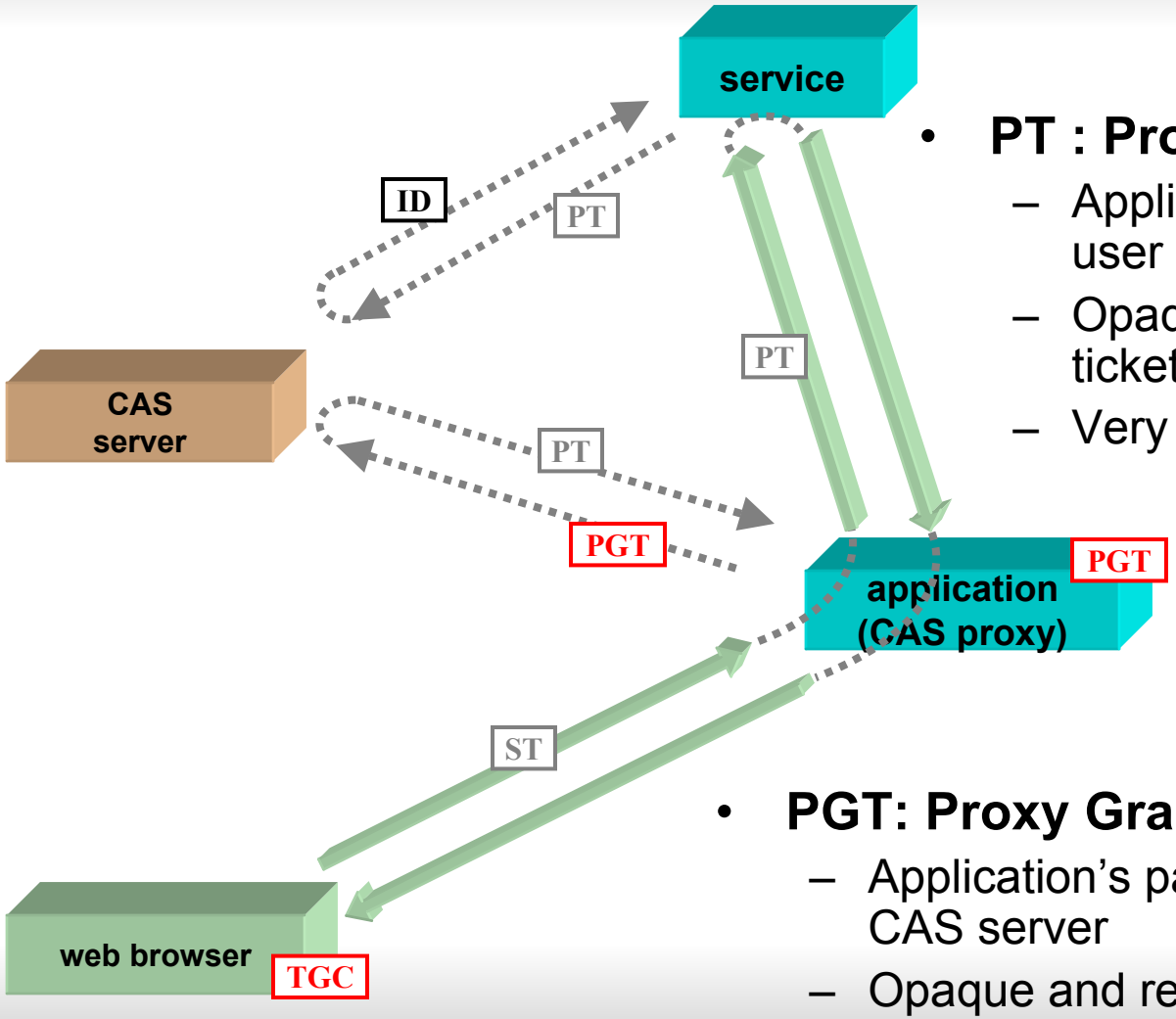
```php
<?php
  // include phpCAS library
  include_once('CAS/CAS.php');

  // declare our script as a CAS client
  phpCAS::client(CAS_VERSION_2_0,'auth.univ.fr',443,'');

  // redirect to the CAS server if needed
  phpCAS::authenticateIfNeeded();

  // at this point, the user is authenticated
?>
<h1>Successfull Authentication!</h1>
<p>User's login: <?php echo phpCAS::getUser(); ?>.</p>
```

N-tier installations



- **PGT: Proxy Granting Ticket**
  - Application's passport for a user to the CAS server
  - Opaque and re-playable ticket

N-tier installations



- **PT : Proxy Ticket**
  - Application's passport for a user to a tier service
  - Opaque and non re-playable ticket
  - Very limited validity

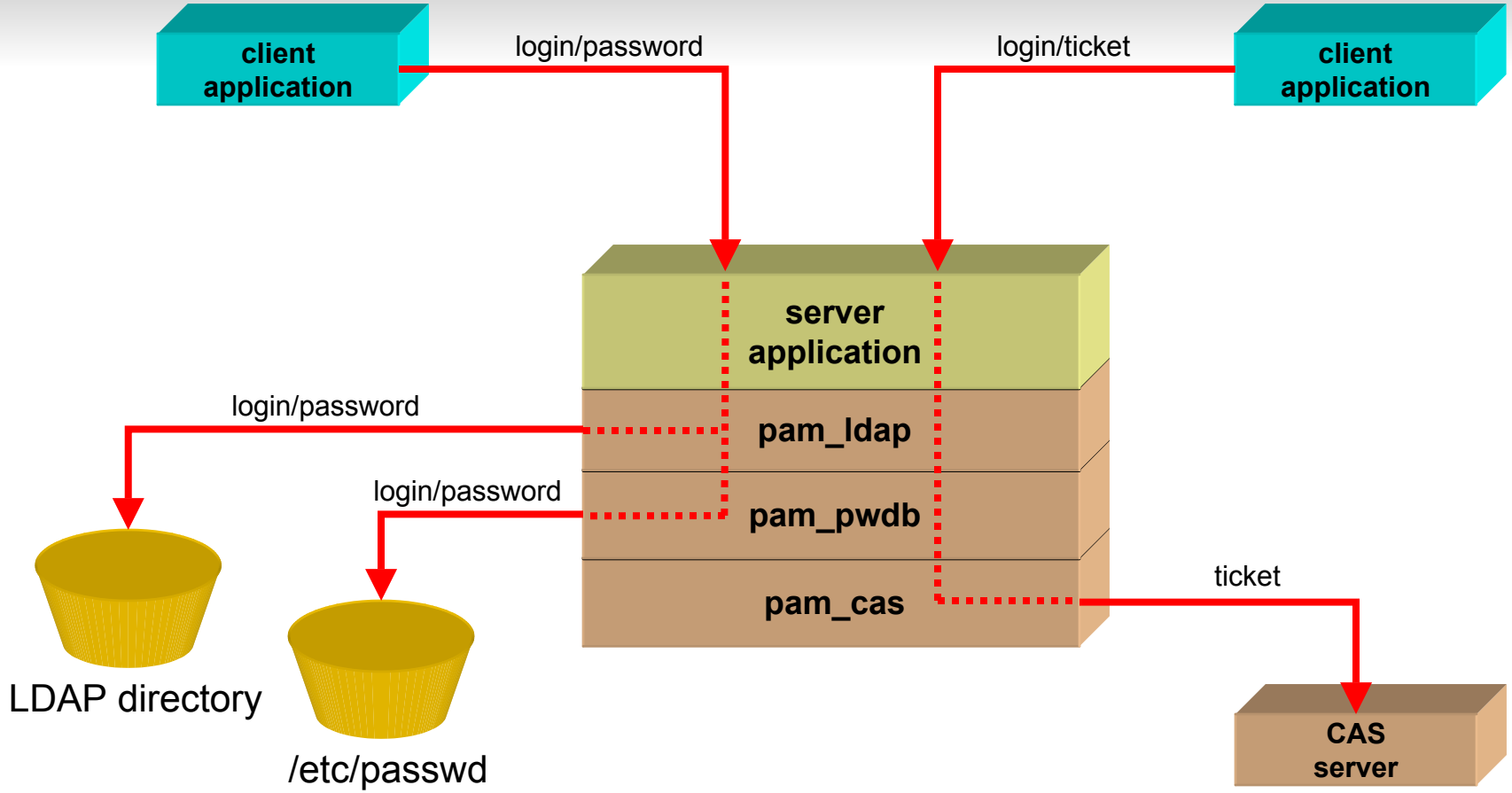
- **PGT: Proxy Granting Ticket**
  - Application's passport for a user to the CAS server
  - Opaque and re-playable ticket

CAS-ifying a non web application

- **One of the strongest points of CAS**
- **Use the pam_cas PAM module**
- **Example of PAM configuration:**

```
auth sufficient /lib/security/pam_ldap.so
auth sufficient /lib/security/pam_pwdb.so shadow nullok
auth required   /lib/security/pam_cas.so
```

The pam_cas PAM module

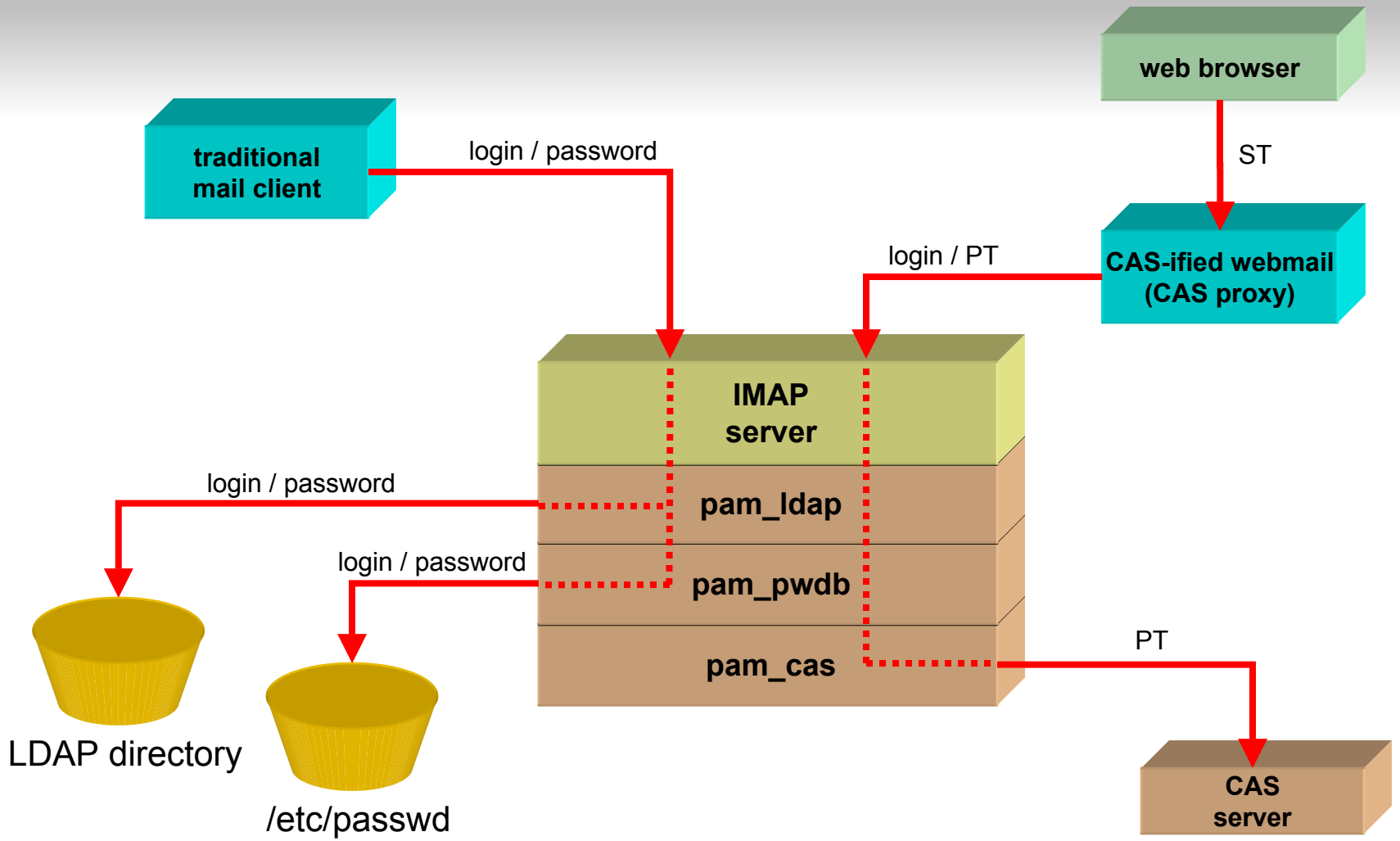


- **Pam_cas authenticates users with a CAS ticket**

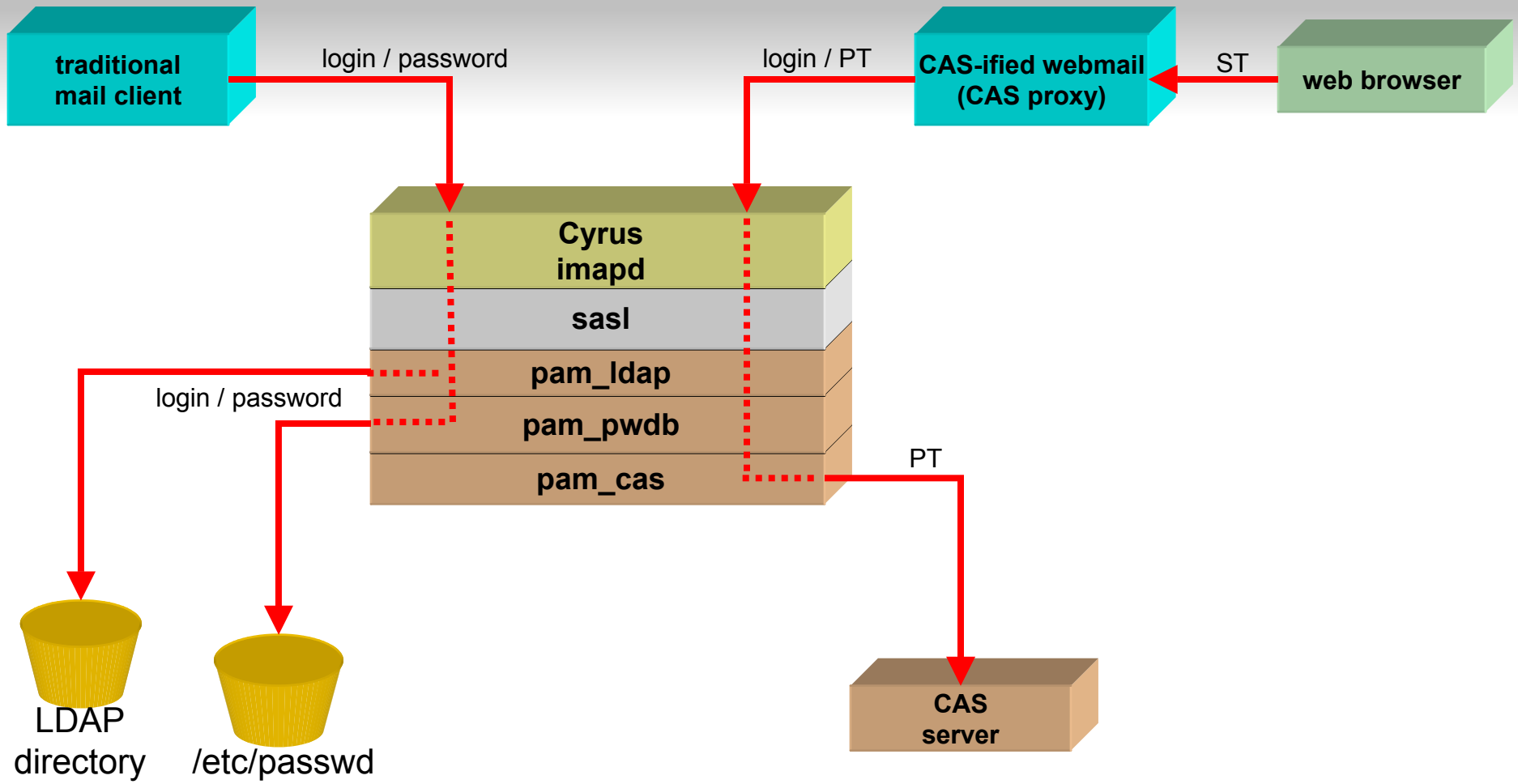
CAS-ifying an IMAP server

- **Objectives**
  - Access an IMAP server from a web application that does not know the password of the user connected
  - Let traditional mail clients authenticate “normally” (with a password)
  - Do not modify the IMAP server
- **The solution: pam_cas :-)**

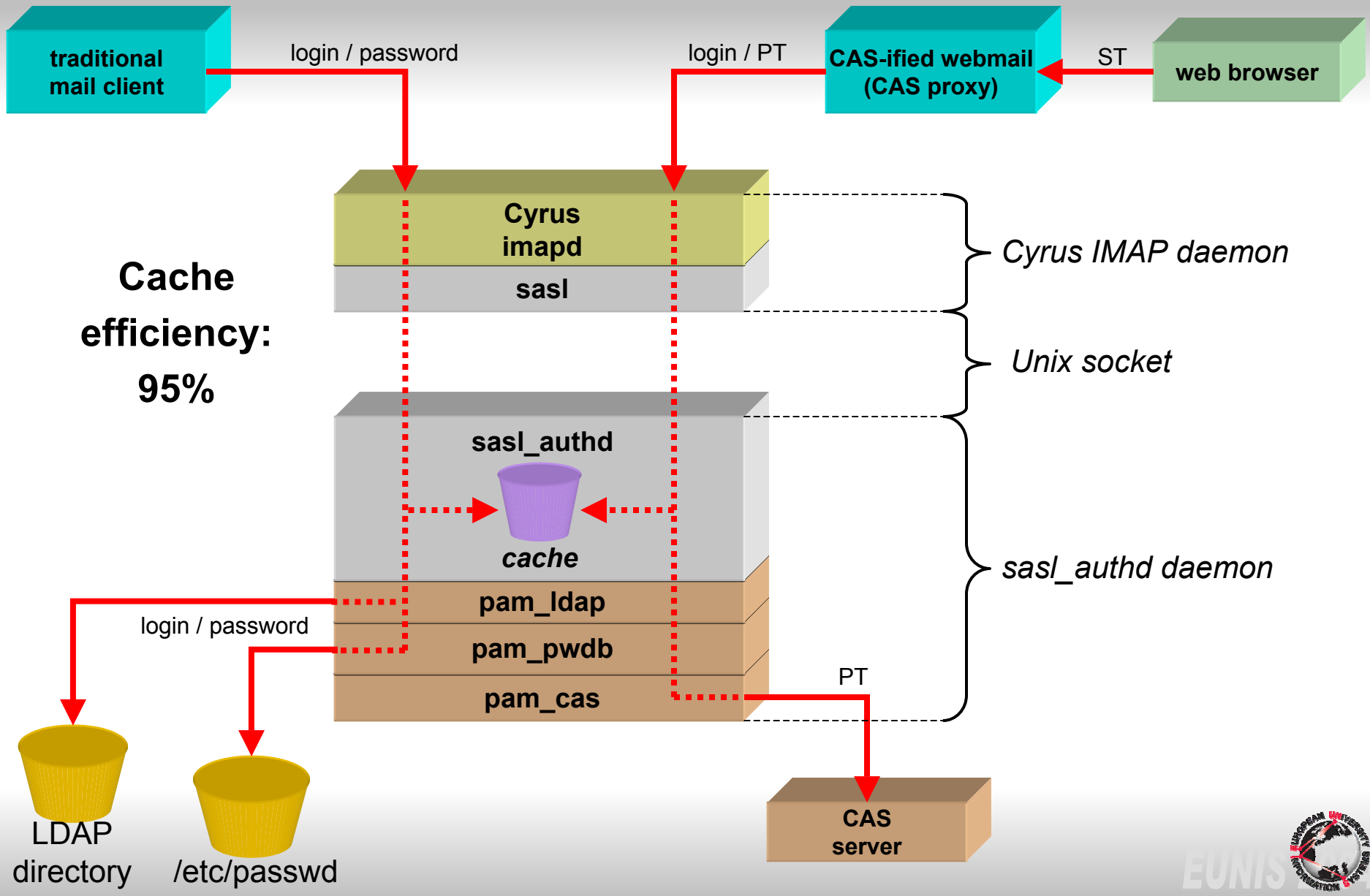
CAS-ifying an IMAP server



CAS-ifying Cyrus IMAPd



CAS-ifying Cyrus IMAPd



Limits (and perspectives)

- **CAS deals with authentication, not authorization**
  - Mixing CAS and Shibboleth?
- **No redundancy**
  - No native load-balancing (but low load)
  - No fault-tolerance (but very good reliability)
- **No Single Sign-Off**
- **A very poor documentation**

The effort of the ESUP-Portail consortium

- **Writing documentation**
- **Adding libraries (phpCAS, esup-mod_cas, esup-pam_cas)**
- **Adding features to the CAS server**
  - Authentication handlers (LDAP, NIS, files, databases, NT domains, …)
  - Mixed authentication
  - Authentication debug mode
  - Rendering customization (appearance, internationalization)
  - CAS quick start (Jakarta Tomcat + Yale CAS server + CAS GH)
- **Supporting the French CAS community**
  - Through forums and mailing lists

Enjoy CAS!

ESUP Portail