Cloud platforms Lead to Open and Universal access for people with Disabilities and for All

Security Gateway: Deployment step by step documentation

Project Acronym	**Cloud4all**
Grant Agreement Number	**FP7-289016**

Authors	**Barcelona Digital Technology Centre**
Status	**First Draft**
Dissemination Level	**Consortium**
Delivery Date	**01/11/2013**
Number of Pages	**9**

Funded by EU's FP7 Led by Technosite Grupo Fundosa

Version History

Table 1. Version history

Revision	Date	Author	Organization	Description
1	10/21/2012	Marcel Malet	BDigital	First draft

Statement of originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Table of contents

List of Figures

1 Components

The Security Gateway is composed by 3 internal elements.

- The **Proxy server**, which is the central point of the gateway.
- The **Authorization Server**, which acts as a decision point and where all security policies are stored.
- The **Monitoring system**, which records all the activity for log and audit purposes.

To test the Security Gateway we have chosen a SP3 application called EAsit4all. This application will send its requests to the security gateway.

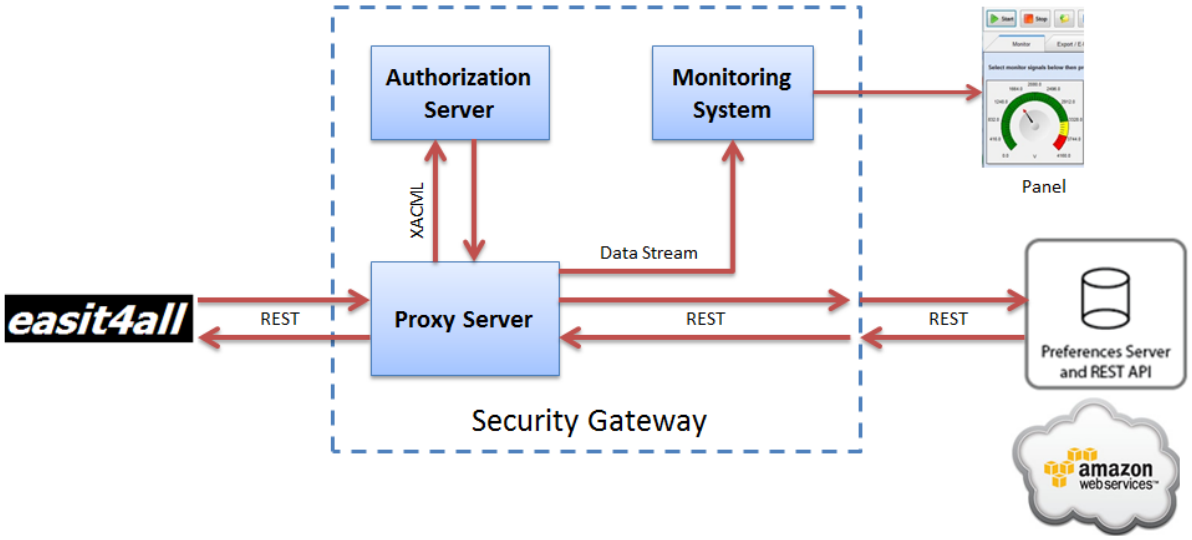

Figure 1. Security Gateway Architecture

1.1 EAsit4all

This application has been modified in order to send the user preferences request to the security gateway instead of directly to the preferences server. When the application makes a request asking for user preferences can receive the user preferences or an empty response, which means that its request has been denied buy the gateway for any reason.

1.2 Proxy Server

This server is responsible for receive the request from applications, in this prototype, from EAasit4all. Then, the Proxy server extracts parameters form the request, such as the user token, an application ID (EAsit4all), the source IP, and information about the device (specifically, the operating system).

Once collected this information, the Proxy server creates an XACML request with these information, and this request is sent to Identity server.

1.3 Authorization Server

The Authorization Server stores all XACML policies of all users. At the same time, it has an XACML engine, which means that this server is able to evaluate all policies from a query.

The Proxy server sends the XACML request to this component and the Authorization Server evaluates all policies (all users' policies). Then, send to Proxy server a response, that can be "permit", "deny" or "not applicable" (if there are not policies to evaluate for a concrete request). The system deals this last option as a "deny".

1.4 Monitor

As a central point, Proxy Server collects all requests and extracts their information. In the same way it sends requests to Authorization server, it sends information to Monitoring system.

1.5 Entitlement mediator

This element is a java binary in the Proxy server. It is responsible for creating the XACML request. You can find this binary in:

"wso2esb-4.7.0\repository\components\patches\patch0000"

2 Deployment

As mentioned before, there are 3 main components; they are servers that can be placed where you want. In the next GitHub repository can be found all components.

https://github.com/marcelbdigital/GatewayPrototype

There you can find 6 folders, included the documentation folder:

- 1. Proxy Server: In this folder you can find the Proxy server ready to run.
- 2. Authorization Server: In this folder you can find the Authorization Server also ready to run.
- 3. Monitoring System (BAM): It is the Monitoring system (Business Activity Monitor-BAM), also ready to run.
- 4. Easit4all: Cloud4all SP3 app, modified to call the Security Gateway.
- 6. SourceCode: where you can find the XMLs needed for each server and the java code of:

- Entitlement_mediator: This is a Java plugin for the Proxy server. Here can be found part of the source code needed for the Proxy server.

2.1 Requirements for the servers

- All servers need Java jdk1.7.0.
- Linux OS (any distribution).

2.2 Run the servers (Proxy, Authorization and Monitoring Servers)

For all proxies you don't need to install anything, you only have to execute it.

2.2.1 Proxy Server (ESB)

Go to folder "1.Proxy Server/wso2esb-4.7.0/bin" and execute "wso2server.sh". A management console can be found via Web browser in: (the user&pass is always "admin" and "admin").

https://(IP_machine):9443/carbon/

The ESB has to be connected to Monitoring system (BAM), for it you has to configure the right IP where is the BAM.

1. Go to https://localhost:9444/carbon/
2. Go to "Configure" and "BAM Server Profile", and fill in your IP (where you have, or will have, the Monitoring system). See the next image.
3. After this change, reload the Server.

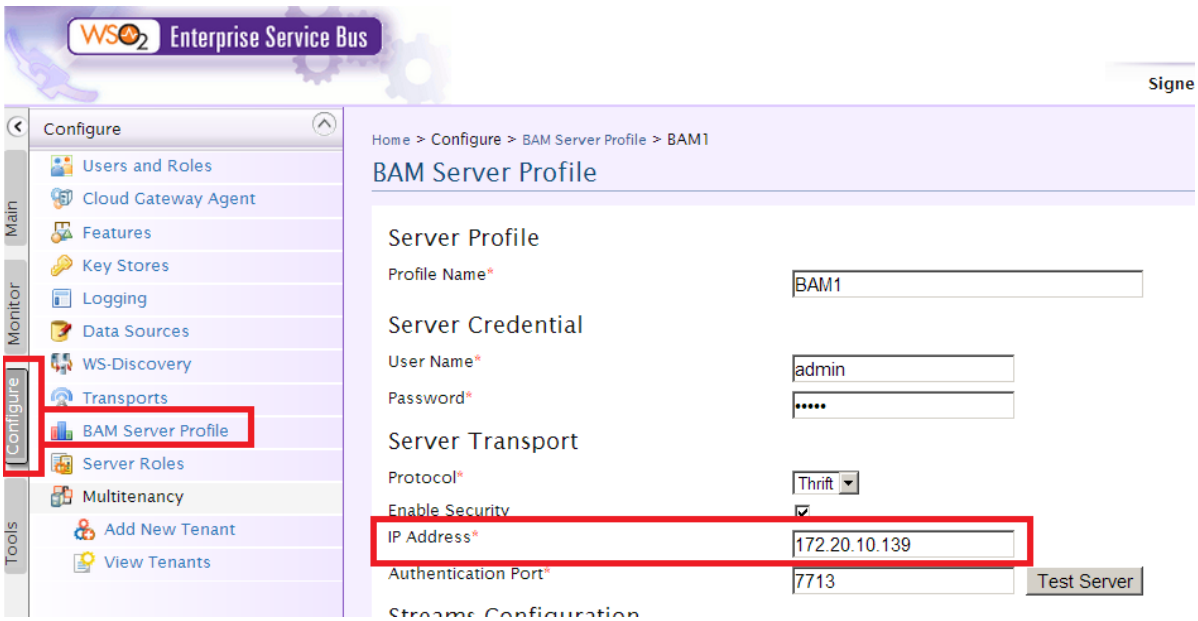

Figure 2. BAM server profile

2.2.1.1 The proxy

In the Proxy server we have created a proxy for authorizing request.

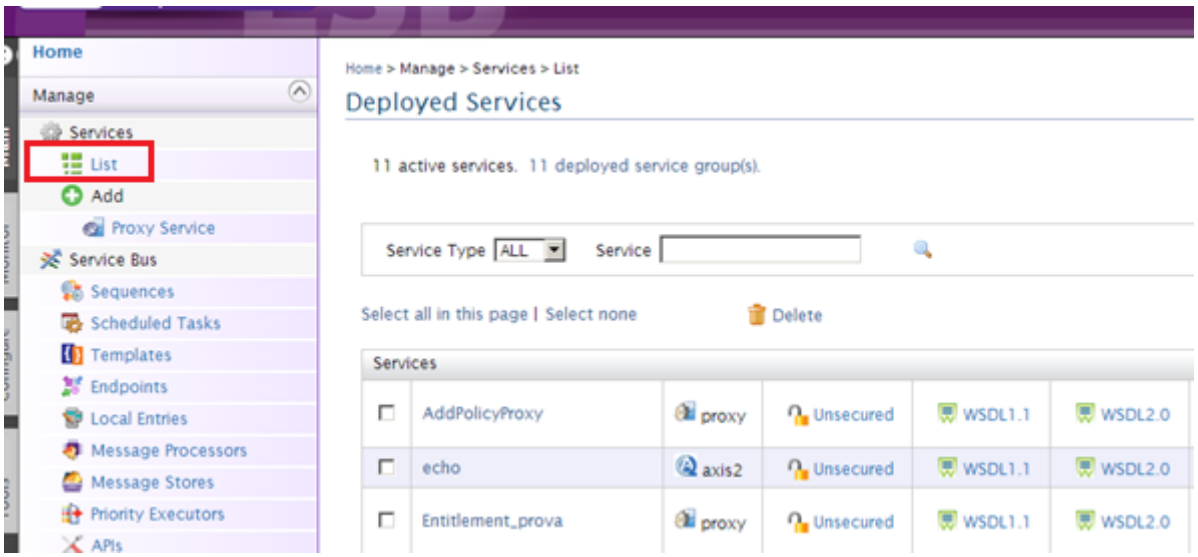

Figure 3. List of proxies

How it works:

1. First the proxy extracts the parameters from the requests, the user token, the application ID, and the origin IP.
2. Verifies that these values are not null, in this case the proxy responds directly with a deny.
3. The proxy creates a XACML request with all parameters.
4. The proxy sends this request to the Authorization Server.
5. The Authorization Server evaluates all needed policies and responds to a Proxy Server with a “permit” or “deny”.
6. If the Proxy Server receives a “permit”, it forwards the user preferences request to the server in the cloud and retrieve the preferences. If it receives a “deny”, sends a 202 http message to the original application (Easit4all).

2.2.2 Authorization Server

Go to folder “2. Authorization Server/wso2esb-4.5.0/bin” and execute “wso2server.sh”. (the user&pass is always “admin” and “admin”).

https://(IP_machine):9445/carbon/

In the dashboard you can see the security policies.

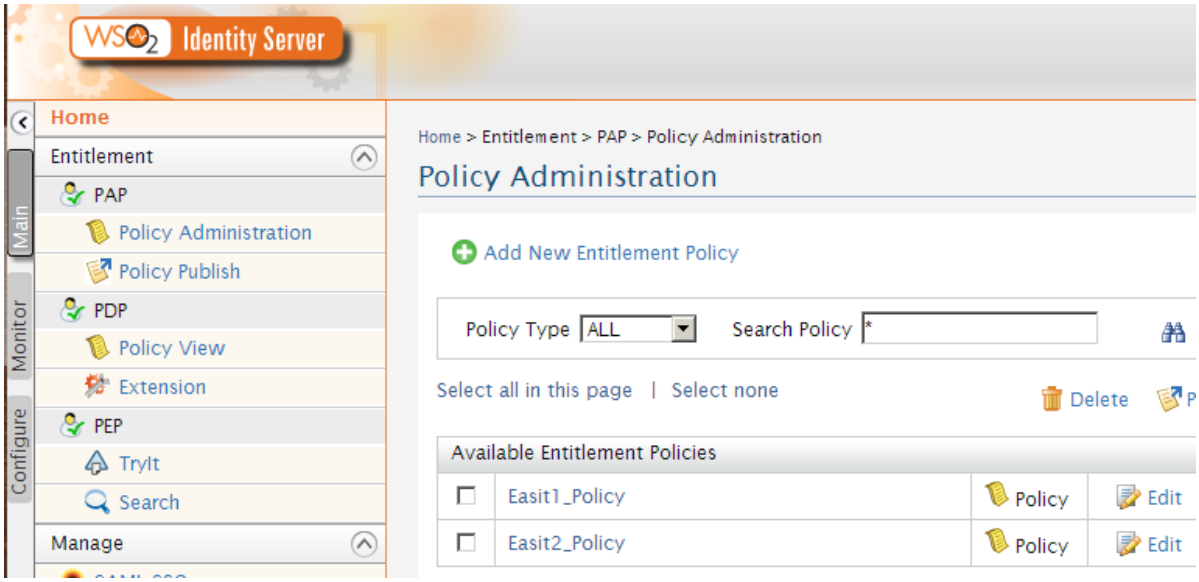

Figure 4. List of Policies, with policies for users "Easit1" and "Easit2".

2.2.3 Monitoring System

Go to folder "Monitoring System (BAM)/wso2esb-2.3.0/bin" and execute "wso2server.sh". (the user&pass is always "admin" and "admin").

https://(IP_machine):9446/carbon/

After the login (admn&admin), to see the dashboard, where you can see how many requests (permits & denies) has been received, you have to click on the "View Portal".

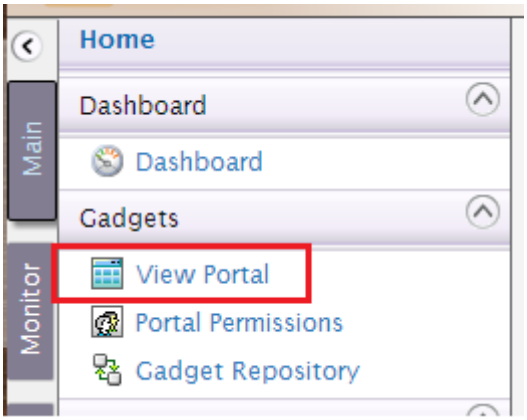

Figure 5. BAM menu

Or enter to:

https://(IP_machine):9446/carbon/dashboard/index.jsp?region=region1&item=portal_menu

The panel shows two tables:

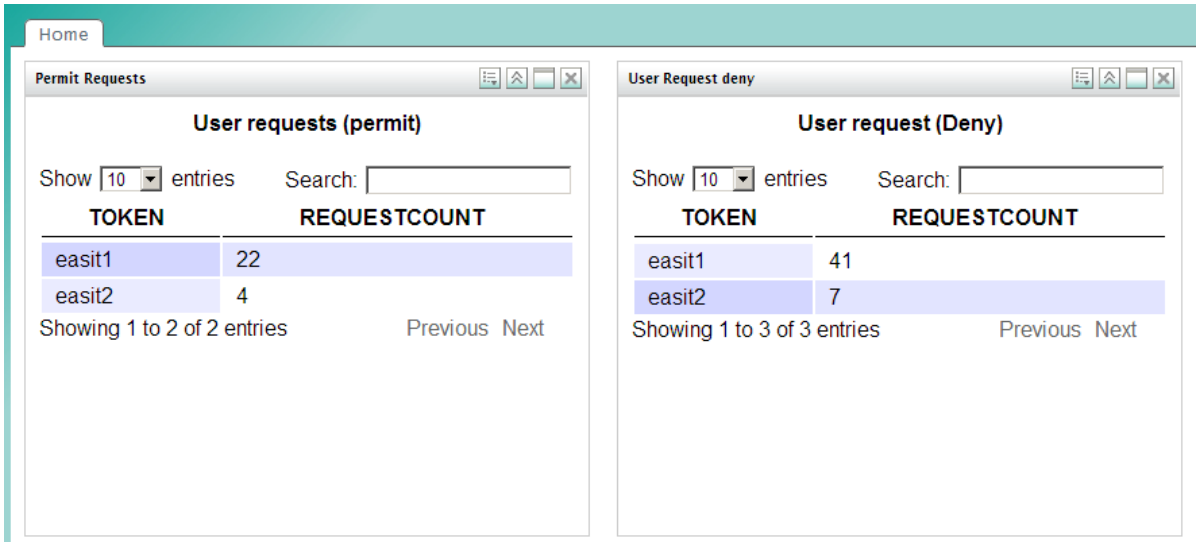

Figure 6. BAM dashboard

2.2.4 EAsit4all

It is an adapted version of this Cloud4all application. In order to use this app you have to:

1. You need to install a Mysql server.
2. Create a database called “asit”.
3. Create a user called “asit”, and password also “asit”.
4. Then import the dump file called: “Dump_easit_04032013.sql”.
5. Create users in account table, “easit1”, “easit2”, with any password.

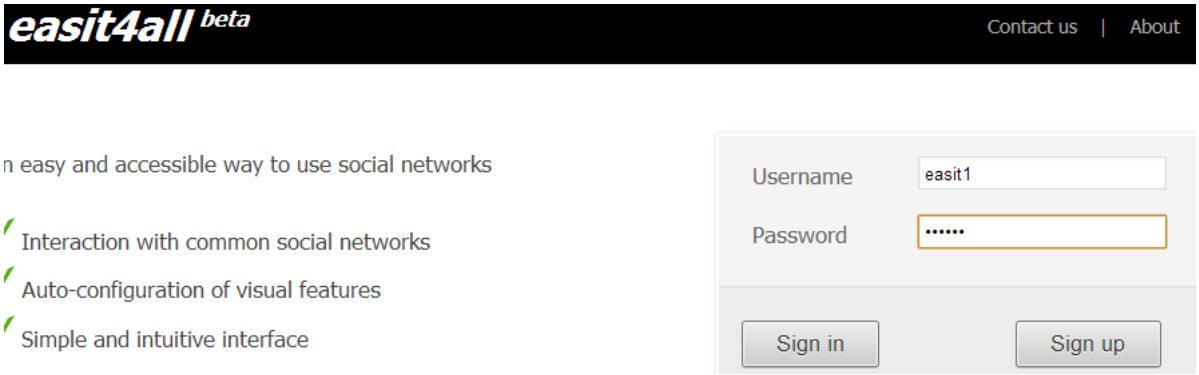

Figure 7. Easit4all application login

3 How to test it

3.1 Policies

Change security policies with your local IP. For this, go to Authorization Server console (https://(IP_machine):9445/carbon/). After login click on "Policy Administration", then click on "edit" of any policy.

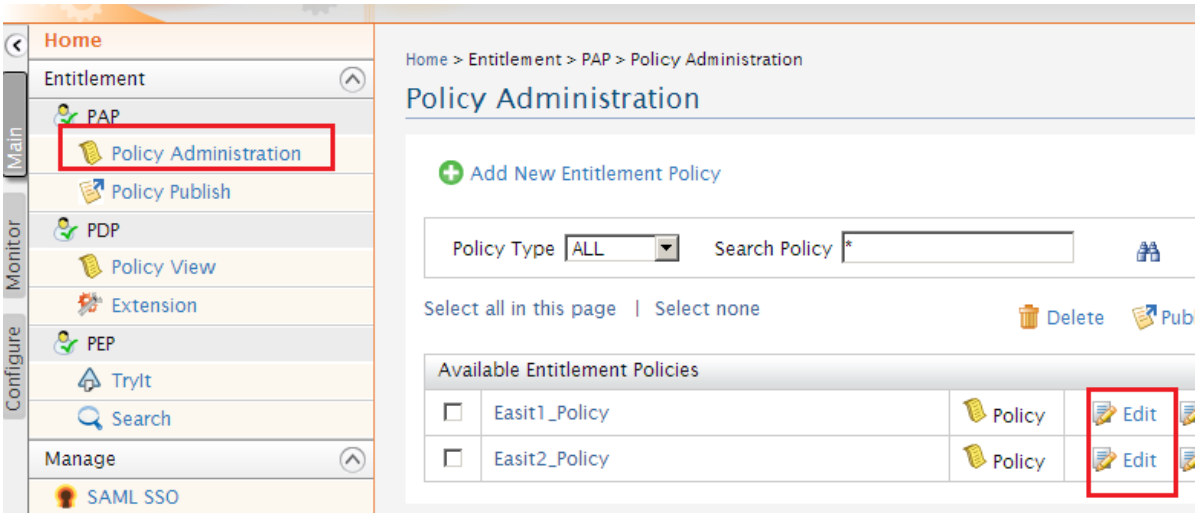

Figure 8. Edit security policies in Authorization Server

You can edit the policy. You can change the IP, and fill in yours. The following policy has one rule. The policy only will be evaluated if the user token is "easit1" (the target). The incoming request with token "easit1" will be permitted if the action is "read" (always is read in this prototype), if the origin IP is "172.20.100.127", and the application ID is "12345", otherwise will be denied.

```
<Policy xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"  PolicyId="Easit1_Policy" RuleCombiningAlgId="urn:oasis:name
   <Target>
      <AnyOf>
         <AllOf>
            <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-regexp-match">
               <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">easit1</AttributeValue>
               <AttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id" Category="urn:oasis:names
            </Match>
         </AllOf>
      </AnyOf>
   </Target>
   <Rule Effect="Permit" RuleId="Rule1">
      <Target></Target>
      <Condition>
         <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:and">
            <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-is-in">
               <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">read</AttributeValue>
               <AttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id" Category="urn:oasis:names:t
            </Apply>
            <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:any-of">
               <Function FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal"></Function>
               <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">172.20.100.127</AttributeValue>
               <AttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:environment:IP" Category="urn:oasis:names:tc:
            </Apply>
            <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:any-of">
               <Function FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal"></Function>
               <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">12345</AttributeValue>
               <AttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:environment:IDApp" Category="urn:oasis:names:
            </Apply>
         </Apply>
      </Condition>
   </Rule>
</Policy>
```

Figure 9. Policy editor

Once modified the policy you have to click on "Publish to my PDP", and then "publish" again.

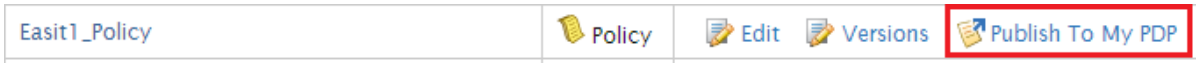

Figure 10. Publish to my PDP button.

3.2 Easit4all

First you have to change the URL of the Security Gateway in the "applicaton.properties" file, in "4. EAsit4all\easit4all\web\src\main\webapp\WEB-INF".

```
<!-- Preference server -->
preferenceServer.name=cloud4all
#preferenceServer.url=http://preferences.gpii.net/user/
#preferenceServer.common=http://registry.gpii.org/common/

preferenceServer.url= http://172.20.10.118:8280/services/Entitlement_prova/<userToken>/settings/<appToken>/net/<localIp>/device/<device>
preferenceServer.common=http://registry.gpii.org/common/
preferenceServer.token=12345
```

Then, recompile the application with maven "mvn clean install". Also you have to execute the script in easit4all folder "install.sh". Finally run the app "mvn tomcat:run".

Log in in the easit4all application with user "easit1" or "easit2". If you have set up a policy permitting you will get your policies, otherwise you will get the next message:

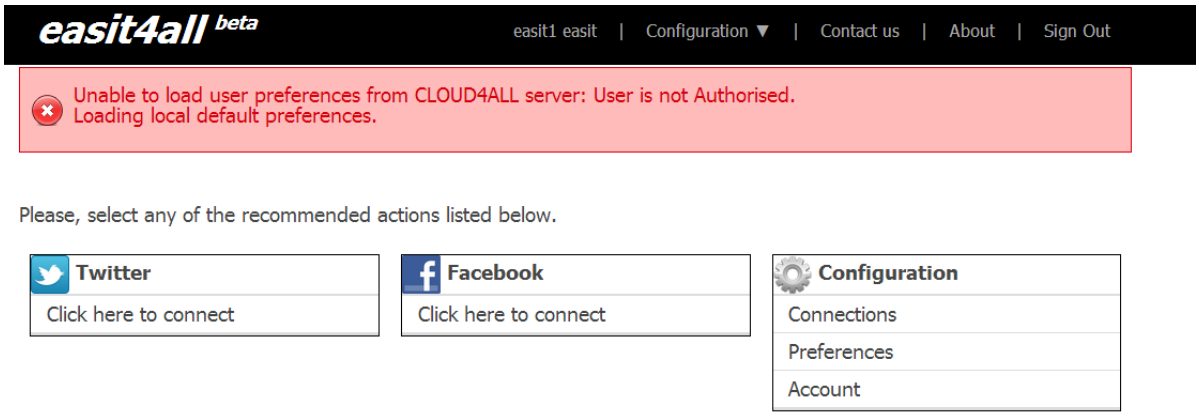

Figure 11. Easit4all not authorized message

4 What is the platform and what is the source code?

A source code is a set of java code files and XML files, and in internal configurations on each server. Specifically:

4.1 Source Code for Proxy Server

The proxy is a XML file. It can be found in "6.SoucreCode/Proxy Server". Also there is a Java plugin, the Entitlement_mediator in the same folder.

4.2 Source Code for Authorization Server

This is the place for the security policies (in database). Also in 6.SoruceCode/Authorization Server

4.3 Source Code for BAM

Here you have a script for analytics (in a file). It can be found in “6.SoucreCode/BAM”