COMUNE DI BOLOGNA

AUTENTICAZIONE RETE CIVICA IPERBOLE - SPECIFICHE DI INTEGRAZIONE

vers. 1.0

ENGINEERING

ENGINEERING INGEGNERIA INFORMATICA SPA
VIA SAN MARTINO DELLA BATTAGLIA, 56 – ROMA

Indice

1. PREMESSA

In questo documento è descritta l'integrazione tra il sistema di autenticazione della Rete Civica Iperbole e una qualsiasi applicazione web esterna.

La Rete Civica Iperbole utilizza un sistema di autenticazione open source denominato CAS (Central Authentication Service http://jasig.github.io/cas/4.0.x/index.html), che offre i servizi di autenticazione e Single Sign On (SSO) a tutti i sistemi con esso integrati.

Tale sistema è stato esteso per consentire l'autenticazione attraverso diversi provider di identità, quali Facebook, LinkedIn, Twitter e Google+ e con il sistema di autenticazione federata dell'Emilia Romagna FedERa.

Il primo accesso dell'utente alla Rete Civica Iperbole prevede una fase di registrazione del profilo. I sistemi integrati potranno accedere a tali informazioni utilizzando dei servizi REST appositamente implementati.

2. SPECIFICHE DI INTEGRAZIONE

Per integrare una applicazione web con il sistema di autenticazione della Rete Civica Iperbole (CAS), che d'ora in poi chiameremo CAS, sono necessarie le seguenti fasi:

- integrazione CAS client
- integrazione con i servizi di profilazione utente

La prima fase riguarda l'autenticazione e il SSO, la seconda si occupa di definire come saranno reperite le informazioni di profilo dell'utente.

2.1. INTEGRAZIONE CAS CLIENT

Per consentire ad una applicazione esterna di integrarsi con il CAS, tale l'applicazione dovrà implementare un client che si occuperà di:

- intercettare le richieste alle pagine web che necessitano di autenticazione
- fare una redirect alla pagina di login della Rete Civica Iperbole
- recuperare l'utenza in seguito all'autenticazione

Il client utilizzato dipenderà dal tipo di piattaforma che si desidera integrare. E' possibile trovare la documentazione necessaria, in base al tipo di piattaforma che si intende utilizzare, a questo indirizzo: http://jasig.github.io/cas/4.0.x/integration/CAS-Clients.html. Di seguito si utilizzerà come esempio l'integrazione con una applicazione web java.

2.1.1. INTEGRAZIONE CAS CLIENT JAVA

Per poter integrare una applicazione web java è necessario aggiungere, alla propria applicazione, la libreria client disponibile a questo indirizzo https://github.com/Jasig/java-cas-client.

Dopo aver integrato tale libreria sarà necessario modificare il web.xml dell'applicazione, in modo da intercettare le richieste e inoltrarle al CAS. Per fare cioè bisogna inserire dei web filters e mappare i percorsi che si desidera proteggere con l'autenticazione.

Vediamo di seguito un esempio di web filters necessari per una corretta integrazione con il CAS:

```
<filter>
  <filter-name>CAS Authentication Filter</filter-name>
  <filter-class>org.jasig.cas.client.authentication.Saml11AuthenticationFilter</filter-class>
  <init-param>
    <param-name>casServerLoginUrl</param-name>
    <param-value> casURLLogin</param-value>
  </init-param>
  <init-param>
    <param-name>serverName</param-name>
```

```
    <param-value>http://www.acme-client.com</param-value>
  </init-param>
</filter>
<filter>
  <filter-name>CAS Validation Filter</filter-name>
  <filter-class>org.jasig.cas.client.validation.Saml11TicketValidationFilter</filter-class>
  <init-param>
    <param-name>casServerUrlPrefix</param-name>
    <param-value>casURL</param-value>
  </init-param>
  <init-param>
    <param-name>serverName</param-name>
    <param-value>http://www.acme-client.com</param-value>
  </init-param>
</filter>
<filter>
  <filter-name>CAS HttpServletRequest Wrapper Filter</filter-name>
  <filter-class>org.jasig.cas.client.util.HttpServletRequestWrapperFilter</filter-class>
</filter>
```

Descriviamo di seguito i parametri di interesse:

- *casServerLoginUrl*: url alla pagina di login della Rete Civica Iperbole
- *casServerUrlPrefix*: url alla pagina del CAS
- *serverName*: url dell'applicazione che si sta integrando

Di seguito mostriamo un esempio riguardante il mapping dei percorsi da proteggere con i web filters precedentemente dichiarati:

```
<filter-mapping>
  <filter-name>CAS Authentication Filter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
<filter-mapping>
  <filter-name>CAS Validation Filter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
<filter-mapping>
  <filter-name>CAS HttpServletRequest Wrapper Filter</filter-name>
```

```
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

Per recuperare le informazioni dell'utente e gestirle a seconda delle proprie necessità è possibile estendere la classe *org.jasig.cas.client.validation.Saml11TicketValidationFilter*.

Per la descrizione di dettaglio si rimanda alla documentazione del CAS https://wiki.jasig.org/display/CASC/Configuring+the+Jasig+CAS+Client+for+Java+in+the+web.xml

2.2. PROFILAZIONE

La Rete Civica Iperbole offre la possibilità di accedere attraverso i seguenti provider di identità:

- Facebook
- Twitter
- LinkedIn
- Google+
- FedERa

A seconda dell'identity provider scelto, in fase di primo accesso, ad ogni profilo creato verrà associato un livello di affidabilità. Questo per consentire all'utente di effettuare operazioni diverse in base al tipo di profilo con il quale si è autenticato. I livelli di affidabilità o di autenticazione si distinguono in due tipologie:

- Autenticazione debole: associata agli identity provider Facebook, Twitter, LinkedIn e Google+
- Autenticazione forte: associata all'identity provider FedERa

2.2.1. SERVIZI DI PROFILAZIONE

I servizi di profilazione mettono a disposizione le seguenti operazioni:

Metodo	URI	Descrizione
GET	/persona/view/username/{idAccount}	Recupera il profilo dell'utente in sola visualizzazione
GET	/persona/exists/username/{idAccount}	Verifica se esiste il profilo di un determinato utente
GET	/persona/delete/username/{idAccount}	Esegue la cancellazione del profilo dell'utente
GET	/persona/edit/username/{idAccount}	Recupera il profilo dell'utente in modifica
POST	/persona/create	Esegue la creazione del profilo

		dell'utente
POST	/persona/update	Esegue l'aggiornamento del profilo dell'utente

2.2.1.1. VISUALIZZA PROFILO

Questo metodo offre la possibilità di recuperare i dati del profilo di una persona passando l'*idAccount* come parametro di input.

Di seguito mostriamo il formato della risposta che si ottiene invocando il metodo *view*.

GET: /persona/view/username/{idAccount}


```
{
  "status":{
    "ok":true,
    "errorMsg":"string"
  },
  "persona":{
    "idAccount":"string",
    "nome":"string",
    "cognome":"string",
    "cf":"string",
    "email":"string",
    "emailPec":"string",
    "telefono":"string",
    "cellulare":"string",
    "tipoAccount":"string",
    "livelloAutenticazione":"string",
    "nascitaData":"string",
    "nascitaIdComune":"string",
    "nascitaComune":"string",
    "residenzaVia":"string",
    "residenzaNumeroCivico":"string",
    "residenzaCap":"string",
    "residenzaIdComune":"string",
    "residenzaComune":"string",
    "domicilioVia":"string",
    "domicilioNumeroCivico":"string",
    "domicilioCap":"string",
    "domicilioIdComune":"string",
    "domicilioComune":"string",
```

```
    "professione":"string",
    "fotoBase64":"string",
    "logoEBolognaBase64":"string",
    "elencoInteressi":[ ],
    "profiloCompleto":"string",
    "primoAccesso":"string",
    "fotoMimeType":"string",
    "emailNewsletter":"string",
    "logoEBolognaNickName":"string",
    "logoEBolognaColor":"string",
    "logoEBolognaMimeType":"string",
    "logoEBolognaMimetype":"string",
    "logoEBolognaMixed":"string",
    "logoEBolognaMixedMimetype":"string",
    "statoIscrizioneNewsletter":"string"
  }
}
```


2.2.1.2. VISUALIZZA PROFILO IN MODIFICA

Questo metodo fornisce la possibilità di recuperare oltre ai dati del profilo della persona anche le informazioni sullo stato degli attributi. In particolare per ogni attributo restituisce due informazioni:

- ro: se l'attributo è in sola lettura (read only)
- required: se l'attributo è obbligatorio.

Di seguito mostriamo il formato della risposta che si ottiene invocando il metodo *edit* passando l'*idAccount* come parametro di input.

GET: /persona/edit/username/{idAccount}


```
{
  "status":{
    "ok":true,
    "errorMsg":"string"
  },
  "persona":{
    "idAccountCampo":{
      "valore":"string",
      "ro":true,
      "required":false
    },
    "nomeCampo":{
      "valore":"string",
      "ro":true,
```

```
    "required":true
  },
  "cognomeCampo":{
    "valore":"string",
    "ro":true,
    "required":true
  },
  "cfCampo":{
    "valore":"string",
    "ro":true,
    "required":false
  },
  "emailCampo":{
    "valore":"string",
    "ro":false,
    "required":true
  },
  "emailPecCampo":{
    "valore":"string",
    "ro":false,
    "required":false
  },
  "emailNewsletterCampo":{
    "valore":"string",
    "ro":false,
    "required":false
  },
  "telefonoCampo":{
    "valore":null,
    "ro":false,
    "required":false
  },
  "cellulareCampo":{
    "valore":"string",
    "ro":false,
    "required":false
  },
  "tipoAccountCampo":{
    "valore":"string",
    "ro":true,
    "required":false
  },
  "tipoAccountIdCampo":{
    "valore":"string",
    "ro":true,
    "required":false
  },
  "livelloAutenticazioneCampo":{
    "valore":"string",
    "ro":true,
    "required":false
  },
  "nascitaDataCampo":{
    "valore":"string",
    "ro":true,
    "required":false
```

```
},
"nascitaComuneCampo":{
   "valore":"string",
   "ro":true,
   "required":false
},
"nascitaIdComuneCampo":{
   "valore":"string",
   "ro":true,
   "required":false
},
"nascitaProvinciaCampo":{
   "valore":"string",
   "ro":null,
   "required":false
},
"residenzaViaCampo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"residenzaNumeroCivicoCampo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"residenzaCapCampo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"residenzaIdComuneCampo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"residenzaComuneCampo":{
   "valore":"string",
   "ro":true,
   "required":false
},
"residenzaProvinciaCampo":{
   "valore":"string",
   "ro":null,
   "required":false
},
"domicilioViaCampo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"domicilioNumeroCivicoCampo":{
   "valore":"string",
   "ro":false,
   "required":false
},
```

```
"domicilioCapCampo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"domicilioIdComuneCampo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"domicilioComuneCampo":{
   "valore":"string",
   "ro":true,
   "required":false
},
"domicilioProvinciaCampo":{
   "valore":"string",
   "ro":null,
   "required":false
},
"professioneCampo":{
   "valore":"string",
   "ro":true,
   "required":false
},
"professioneIdCampo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"fotoBase64Campo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"fotoMimeTypeCampo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"logoEBolognaBase64Campo":{
   "valore":"string",
   "ro":false,
   "required":false
},
"elencoInteressiCampo":[
],
"profiloCompleto":true,
"statoIscrizioneNewsletterCampo":{
   "valore":"string",
   "ro":true,
   "required":false
},
"statoIscrizioneNewsletterIdCampo":{
   "valore":"string",
   "ro":false,
```

```
        "required":false
      },
      "logoEBolognaColorCampo":{
        "valore":"string",
        "ro":false,
        "required":false
      },
      "logoEBolognaNickNameCampo":{
        "valore":"string",
        "ro":false,
        "required":false
      },
      "logoEBolognaMimeTypeCampo":{
        "valore":"string",
        "ro":false,
        "required":false
      },
      "logoEBolognaMixedCampo":{
        "valore":"string",
        "ro":false,
        "required":false
      },
      "logoEBolognaMixedMimetypeCampo":{
        "valore":"string",
        "ro":false,
        "required":false
      }
    }
}
```


2.2.1.3. CANCELLAZIONE PROFILO

Tale metodo si occupa di cancellare il profilo dell'utente fisicamente dalla base dati della Rete Civica Iperbole.

Di seguito mostriamo il formato della risposta che si ottiene invocando il metodo *delete* passando l'*idAccount* come parametro di input.

GET: /persona/delete/username/{idAccount}


```
{
  "status":{
    "ok":true,
    "errorMsg":"string"
  }
```


2.2.1.4. CREAZIONE PROFILO

Per creare il profilo di un utente utilizzando i servizi di profilazione è necessario eseguire una POST a tale metodo passando un oggetto json con lo stesso formato ottenuto invocando il metodo *view*.

Di seguito mostriamo il formato della risposta che si ottiene invocando il metodo *create.*

POST: /persona/create/
{ *"status":{* *"ok":true,* *"errorMsg":"string"* *}*

2.2.1.5. MODIFICA PROFILO

Per aggiornare il profilo di un utente utilizzando i servizi di profilazione è necessario eseguire una POST a tale metodo passando un oggetto json con lo stesso formato ottenuto invocando il metodo *edit*.

Di seguito mostriamo il formato della risposta che si ottiene invocando il metodo *update.*

POST: /persona/update/
{ *"status":{* *"ok":true,* *"errorMsg":"string"* *}*

2.2.2. ACCESSO AI SERVIZI DI PROFILAZIONE

I servizi non sono esposti su internet sarà pertanto fornito un accesso in VPN e con protocollo http.