ADUCID Integration Manual
Spring Security

Version 3.0.4

Release date 1. February 2016

Antala Staška 2027/79, 140 00 Praha 4 | T +420 271 100 100 | F +420 271 100 101
Vídeňská 204/125, Přízřenice, 619 00 Brno | T +420 547 100 100 | F +420 547 100 101
Sady 5. května 59, 301 00 Plzeň | T +420 271 100 100 | F +420 271 100 101
Teslova 30, 821 02 Bratislava | T +421 (2) 4821 3111 | F +421 (2) 4821 3199

ISO 9001 ISO 14001 ISO/IEC 27001 ISO/IEC 20000

CZ > IČ 25313029 | Rejstříkový soud KS Brno, spis. zn. B2113
SK > IČ 35787546 | Obchodný register OS Bratislava, oddíl Sa, vložka 2431/B

Table of Contents

1. Purpose of this document

This document serves as an integration manual for incorporating ADUCID® technology into the Spring Security Framework. Version of Spring Security Framework greater or equal 3.0 is currently supported. Previous versions differ in security API.

2. Prerequisites

When reading this document, the following basic knowledge is assumed:

- Spring Framework (http://www.springsource.org/)
- Spring Security Framework (http://www.springsource.org/spring-security)
- aducid-architecture.pdf
- ADUCIDServerKit-administration-guide.pdf (if you are also the administrator of ADUCID Server Kit)
- ADUCIDServerKit-installation-guide.pdf (if you are also the administrator of ADUCID Server Kit)
- Knowledge of web technologies, programming and integration of web applications

3. Architecture and integration

In order to simplify the integration of ADUCID® technology with Spring Security Framework, the ADUCID® security extension was created.

From deployment perspective, this represents the following logical model:

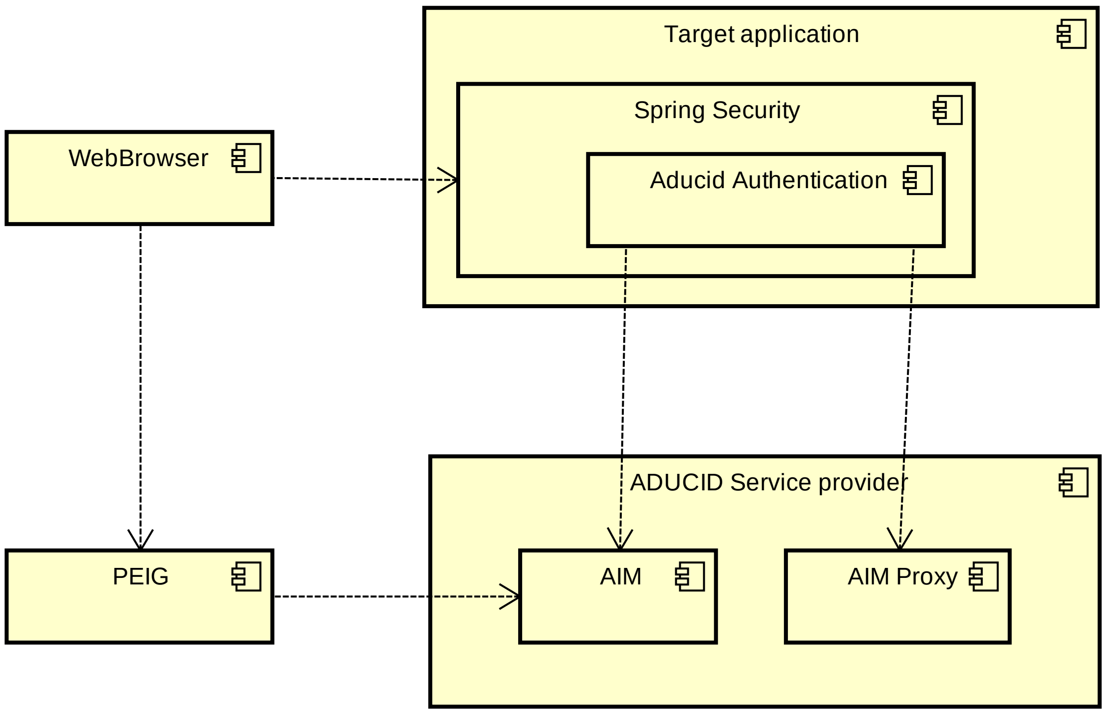


Figure 3-1 Logic deployment diagram

4. Basic components

ADUCID Spring Security adapter consists of several components that implement Spring Security Framework standard interface for authentication mechanisms. We highly recommend to read Spring

Security architecture documentation before, because understanding the way how the ADUCID® security is integrated into Spring Security Framework is necessary to correctly configure the adapter.

Before describing each component the main authentication flow is described.

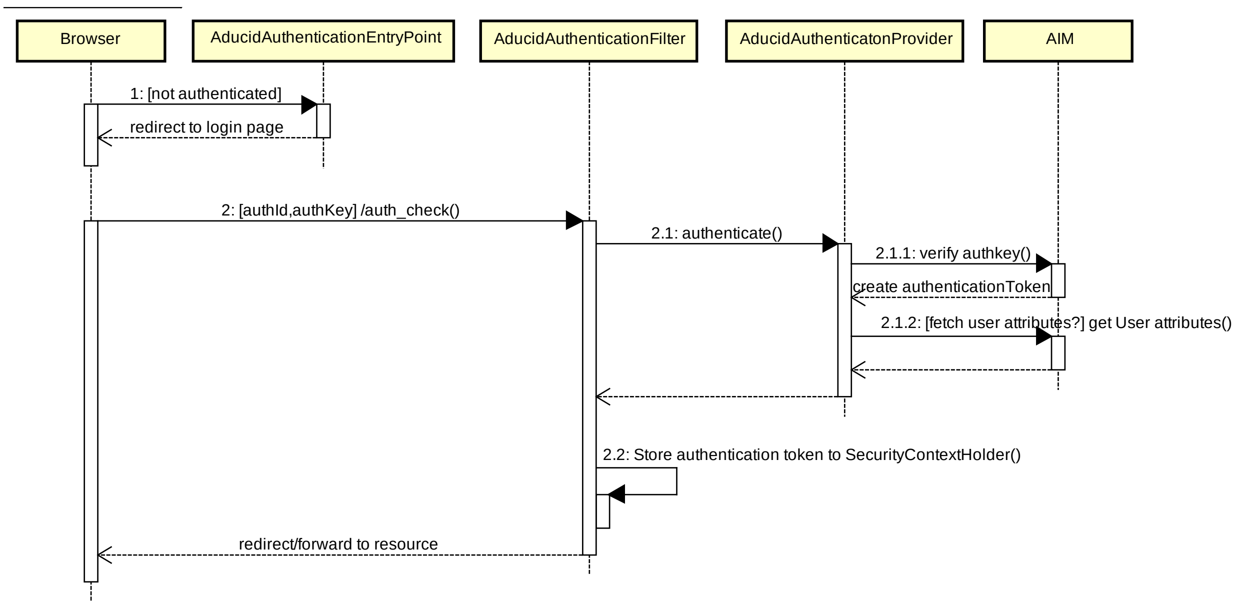

Browser
AducidAuthenticationEntryPoint
AducidAuthenticationFilter
AducidAuthenticatonProvider
AIM
1: [not authenticated]
redirect to login page
2: [authId,authKey] /auth_check()
2.1: authenticate()
2.1.1: verify authkey()
create authenticationToken
2.1.2: [fetch user attributes?] get User attributes()
2.2: Store authentication token to SecurityContextHolder()
redirect/forward to resource

Figure 4-1 Basic components

1/ User accesses a secure page within the application.
2/ Spring Security Framework detects unauthenticated request and delegates the processing to AducidAuthenticationEntryPoint component to start ADUCID authentication process. Current implementation uses AIM-Proxy for authentication.
3/ Communication is returned back to Browser, where authentication process is started.
4/ Once the authentication process has finished, the result is returned along with ADUCID credentials back to the Spring Security Framework (to the /auth_check endpoint), especially to the AducidAuthenticationFilter component.
5/ AducidAuthenticationFilter verifies the credentials through AducidAuthenticationProvider component against the AIM's authentication server.
6/ If the authentication succeeds the AducidAuthenticationProvider reads User attributes.
7/ Both AducidSession and User attributes are stored to the SecurityContextHolder object. This is typically persisted to HTTP session for subsequent requests.
8/ Redirect to the original page is performed.

4.1.1. AducidAuthenticationEntryPoint

This component ensures start of ADUCID authentication. The component implements AuthenticationEntryPoint interface and performs redirect to the page, where the logic of starting ADUCID authentication resides. Typically AIM-Proxy component is used.

It performs following operations:

- Creates ADUCID authentication session in AIM server using ADUCID Java SDK
- Triggers redirect to page defined by defaultLoginUrl parameter

AducidAuthenticationEntryPoint exposes following parameters:

Property	**Description**
aimUrl	URL to AIM R4 server
defaultLoginUrl	URL to page where ADUCID authentication will start. It directs in most cases to AIM-Proxy process servlet.

The snippet of XML configuration is attached below:

4.1.1.1. Example of XML configuration

```
<beans:bean id="aducidEntryPoint"
     class="com.aducid.spring.security.AducidAuthenticationEntryPoint">
     <beans:property name="aimUrl" value="${aimUrl}/services/R4" />
     <beans:property name="defaultLoginUrl" value="${aimProxyUrl}/process" /></beans:bean>
```

4.1.2. AducidAuthenticationFilter

This component takes ADUCID credentials and performs verification. This component filters request/responses for request directed to „/auth_check“ URL and extracts pair of (authId,authKey) parameters from the request. It is standard implementation of AbstractAuthenticationProcessingFilter from Spring Security Framework and is inserted into standard Spring Security filter chain.

Verification of the ADUCID credentials is delegated to AducidAuthenticationProvider component.

During processing following operations are performed:

- Checks if the user is already authenticated (looks for AducidAuthenticationToken in SecurityContextHolder.
- If the requested URL is not /auth_check, it forwards processing to the next filter in the chain, otherwise creates AducidAuthenticationToken with authId and authKey let it to verify in AducidAuthenticationProvider
- In case of successful verification (AducidAuthenticationProvider returns verified token) stores it in the SecurityContextHolder.
- In case of unsuccessful verification, Spring Security standard error handling is invoked.

AducidAuthenticationFilter exposes following parameters:

Property	**Description**
authenticationManager	Standard Spring security manager tied to AducidAuthenticationProvider described later.
authenticationFailureHandler	Implementation of Spring security AuthenticationFailureHandler class to handle authentication errors. When authentication error occurs during authentication, AuthenticationException or its subclass is thrown and processed by this handler.

4.1.2.1. Example of XML configuration

```
<beans:bean id="aducidFilter"
```

```
     class="com.aducid.spring.security.AducidAuthenticationFilter">
     <beans:property name="authenticationManager" ref="aducidAuthenticationManager" />
     <beans:property name="authenticationFailureHandler">
            <beans:bean id="aducidAuthenticationFailureHandler"
     class="org.springframework.security.web.authentication.SimpleUrlAuthenticationFailure
Handler">
                   <beans:property name="defaultFailureUrl" value="/error.html" />
                   <beans:property name="useForward" value="true" />
            </beans:bean>
     </beans:property>
</beans:bean>

<authentication-manager alias="aducidAuthenticationManager">
     <authentication-provider ref="aducidAuthenticationProvider" />
</authentication-manager>
```

4.1.3. AducidAuthenticationProvider

This component performs ADUCID credentials verification. It is implementation of AuthenticationProvider interface from Spring security. This component takes authentication credentials and creates new AducidAuthenticationToken, which holds ADUCID related information.

It performs following operations:

- Verify ADUCID authentication credential (authId, authKey) against AIM server and stores identity data into authData atribute of AducidAuthenticationToken.
- Based on personalObjectId reads user attributes stored on AIM and creates Spring security principal with these attributes. Creation of principal delegates to AducidUserDetailService class.

AducidAuthenticationProvider exposes following parameters:

Property	**Description**
personalObjectId	Name of user attribute set (assigned to application by AIM administrator)
aimUrl	URL to AIM R4 server
allowNullPrincipal	T/F flag, that enforces non null Principal during authentication process. If Principal can't be created (for example no user attributes exists for the identity), AuthenticationException is thrown.
userDetailService	The service, that produces Spring security Principal objects. The Principal holds both userAttributes and roles (GrantedAuthorities)

4.1.3.1. Example of XML configuration

```
<beans:bean id="aducidAuthenticationProvider"
     class="com.aducid.spring.security.AducidAuthenticationProvider">
     <beans:property name="userDetailService" ref="aducidUserDetailService" />
     <beans:property name="allowNullPrincipal" value="false"/>
     <beans:property name="aimUrl" value="${aimUrl}/services/R4" />
     <beans:property name="personalObjectId" value="TEST" />
</beans:bean>

<beans:bean id="aducidUserDetailService"
```

```
     class="com.aducid.common.services.DefaultAducidUserDetailService">
     <beans:property name="authoritiesUserAttribute" value="testRole" />
</beans:bean>
```

4.1.4. AducidAuthenticationToken

This object is extension of AbstractAuthenticationToken from Spring security Framework. This object is stored into SecurityContextHolder after successful ADUCID authentication,

It contains following attributes.

Property	**Description**
credentials	Instance of AducidCredentials, that encapsulates (authId, authKey) used for verification.
principal	The object implementing standard UserDetails interface and holding list of user attributes fetched from AIM server and list of roles (Granted authorities) used in Spring security Framework for authorization
authData	Instance of AIMGetPSLAttributesResponse, that holds information about cybernetic identity.

5. The Hello World application

For better understanding how the ADUCID Spring security adapter works and how to configure it, a sample application is delivered.

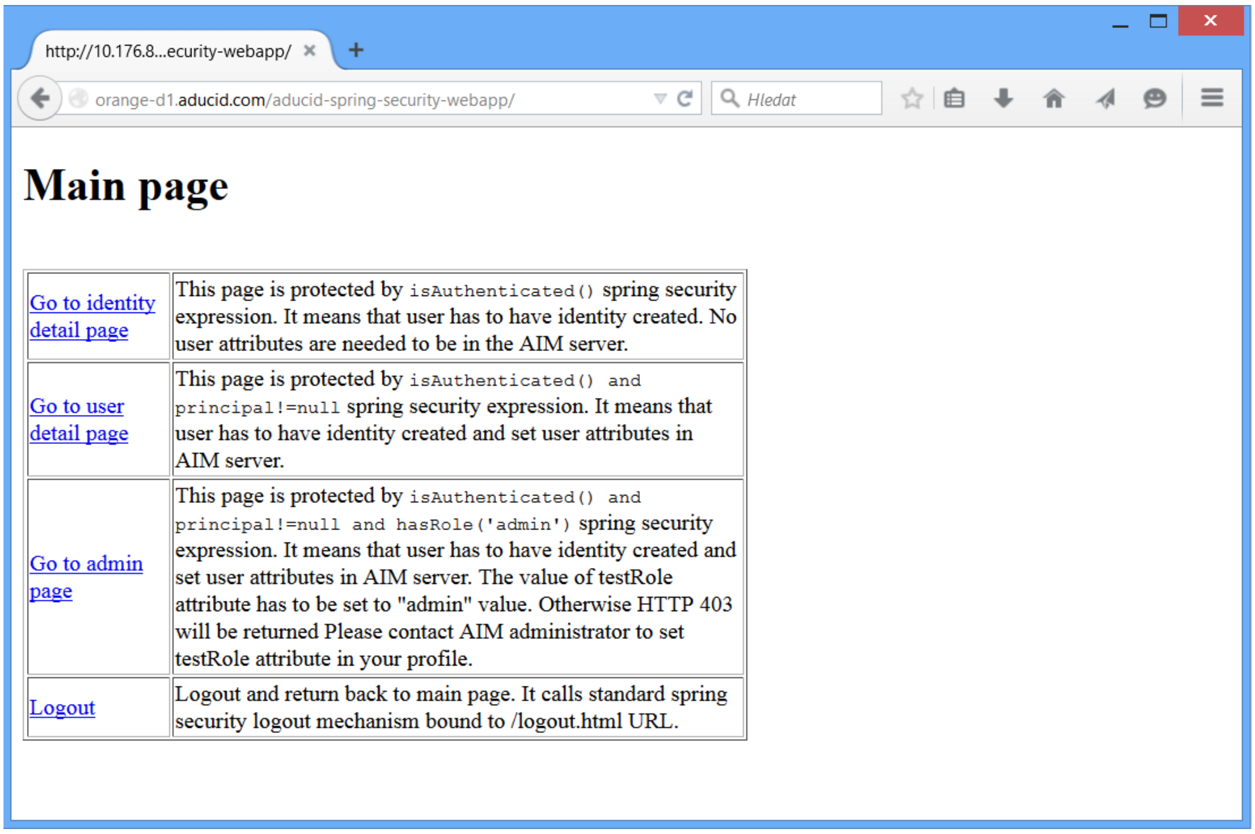
The application can be used in any JEE Web containers, supporting Servlet >2.4 specification. The application is available at http://your_server/aducid-spring-security-webapp/ (we recommend remove version number from archive name before deployment).

It has following structure:

```
Root
->   META-INF
     ->     spring
            ->     applicationContext.xml      (holding spring security configuration)
            ->     webmvc-config.xml           (holding spring mvc configuration)
->   WEB-INF
     ->     pages
            ->     index.jsp                   (main page)
            ->     identity.jsp                (cybernetic identity attributes)
            ->     user.jsp                    (user attribute set and roles)
            ->     admin.jsp                   (admin page secured by role admin)
            ->     error.jsp                   (error page)
     ->     lib                                (additional libraries)
     ->     classes                            (classes and property files)
     ->     web.xml                            (web application descriptor)
```

5.1. Main page

Main page contains links to other pages with examples. The Logout link clears current authentication session and allows log in again.


http://10.176.8...ecurity-webapp/
orange-d1.aducid.com/aducid-spring-security-webapp/
Hledat

Main page

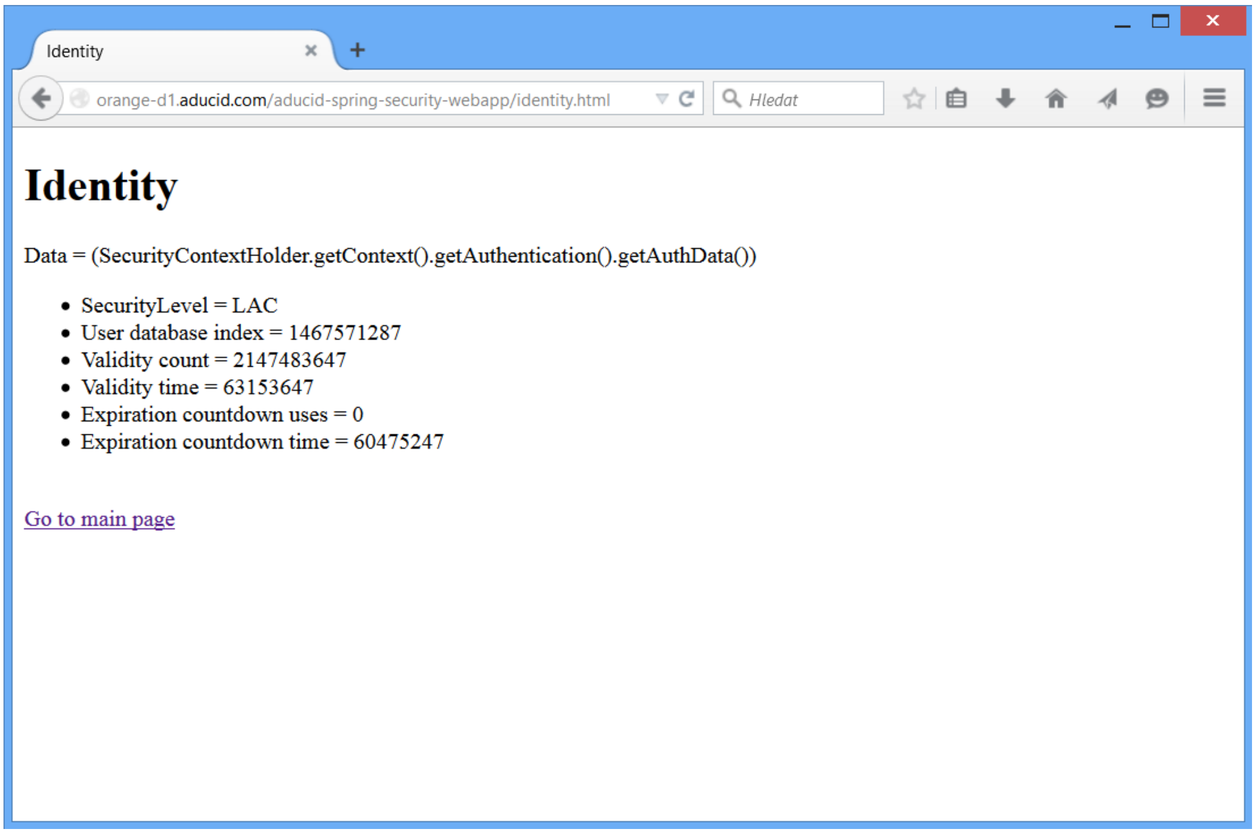
Go to identity detail page	This page is protected by `isAuthenticated()` spring security expression. It means that user has to have identity created. No user attributes are needed to be in the AIM server.
Go to user detail page	This page is protected by `isAuthenticated() and principal!=null` spring security expression. It means that user has to have identity created and set user attributes in AIM server.
Go to admin page	This page is protected by `isAuthenticated() and principal!=null and hasRole('admin')` spring security expression. It means that user has to have identity created and set user attributes in AIM server. The value of testRole attribute has to be set to "admin" value. Otherwise HTTP 403 will be returned Please contact AIM administrator to set testRole attribute in your profile.
Logout	Logout and return back to main page. It calls standard spring security logout mechanism bound to /logout.html URL.

Figure 5-1 Main page

5.2. Identity page

Identity page shows an example how to get cybernetic identity attributes, when you have logged in with ADUCID. It contains jstl + scriptlets to get these information from authenticated AducidAuthenticationToken object stored in SecurityContextHolder.

To access this page, at least cybernetic identity has to be created (it is not necessary to create user attributes-user profile).


Identity
orange-d1.aducid.com/aducid-spring-security-webapp/identity.html
Hledat

Identity

Data = (SecurityContextHolder.getContext().getAuthentication().getAuthData())

- SecurityLevel = LAC
- User database index = 1467571287
- Validity count = 2147483647
- Validity time = 63153647
- Expiration countdown uses = 0
- Expiration countdown time = 60475247

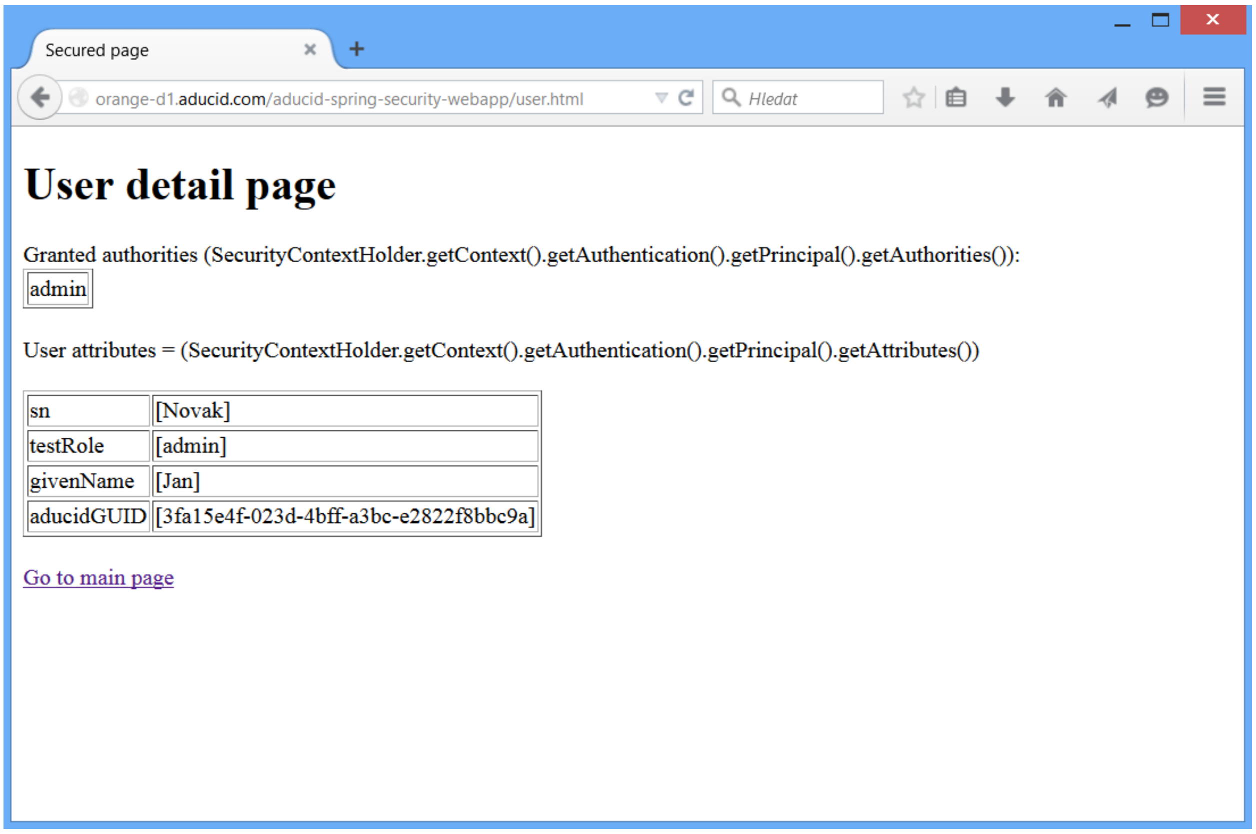
Go to main page

Figure 5-2 Identity page

5.3. User detail page

User detail page shows an example how to access user attributes tied to cybernetic identity. It contains jstl + scriptlets to get these information from authenticated AducidAuthenticationToken object stored in SecurityContextHolder.

To access this page, at least the cybernetic identity and user attributes has to be created. When these conditions are not fulfilled, http 403 is thrown back. It is not necessary to have role assigned to the user.



Secured page
orange-d1.aducid.com/aducid-spring-security-webapp/user.html
Hledat

User detail page

Granted authorities (SecurityContextHolder.getContext().getAuthentication().getPrincipal().getAuthorities()):

admin

User attributes = (SecurityContextHolder.getContext().getAuthentication().getPrincipal().getAttributes())

sn	[Novak]
testRole	[admin]
givenName	[Jan]
aducidGUID	[3fa15e4f-023d-4bff-a3bc-e2822f8bbc9a]

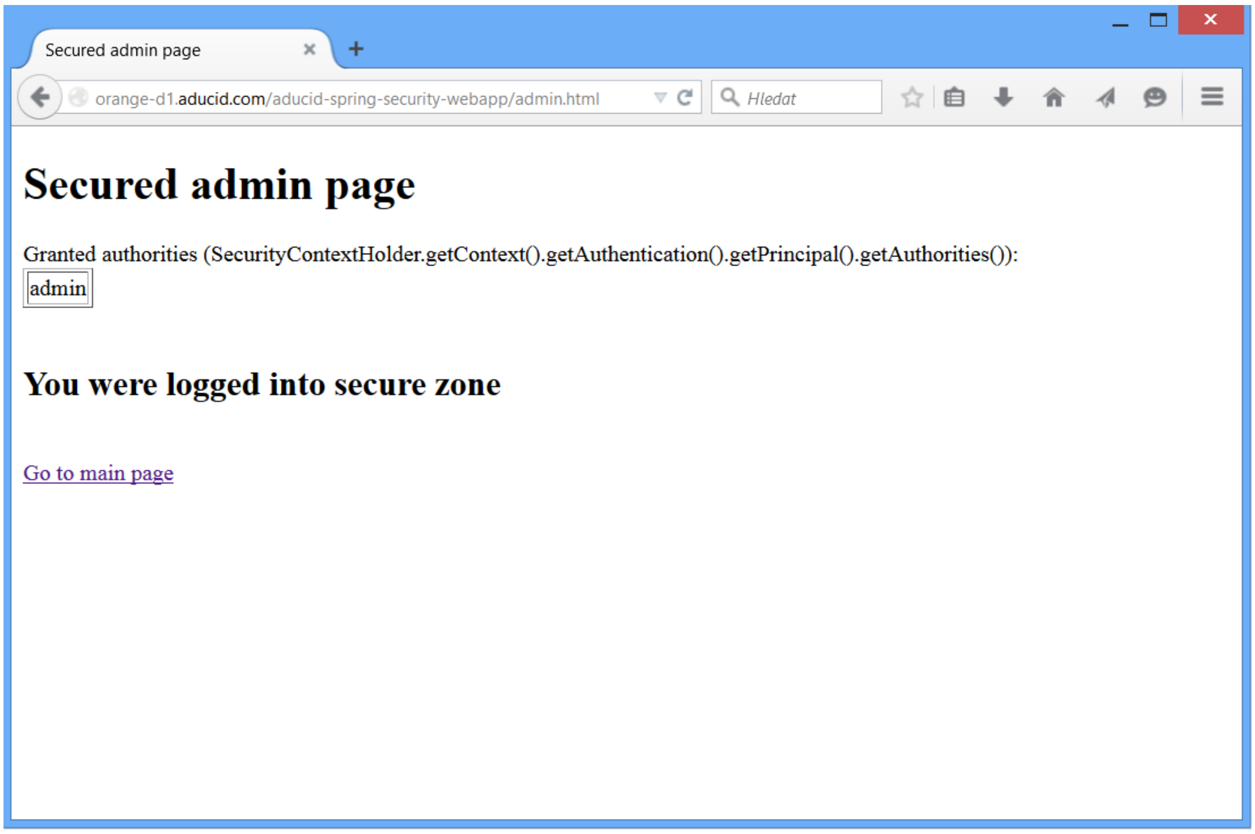
Go to main page

Figure 5-3 User detail page

5.4. Secured admin page

Secured admin page is the example of how the role based authorization works. The security constraint is configured in applicationContext.xml using spring security el language, more detail hasRole(‚admin‘) construct is used.

To access this page, the cybernetic identity and user attributes have to be created and attribute testRole has to be filled by UIM application manager. Here value “admin” is expected. When these conditions are not fulfilled, http 403 is thrown back.


Secured admin page
orange-d1.aducid.com/aducid-spring-security-webapp/admin.html
Hledat

Secured admin page

Granted authorities (SecurityContextHolder.getContext().getAuthentication().getPrincipal().getAuthorities()):

admin

You were logged into secure zone

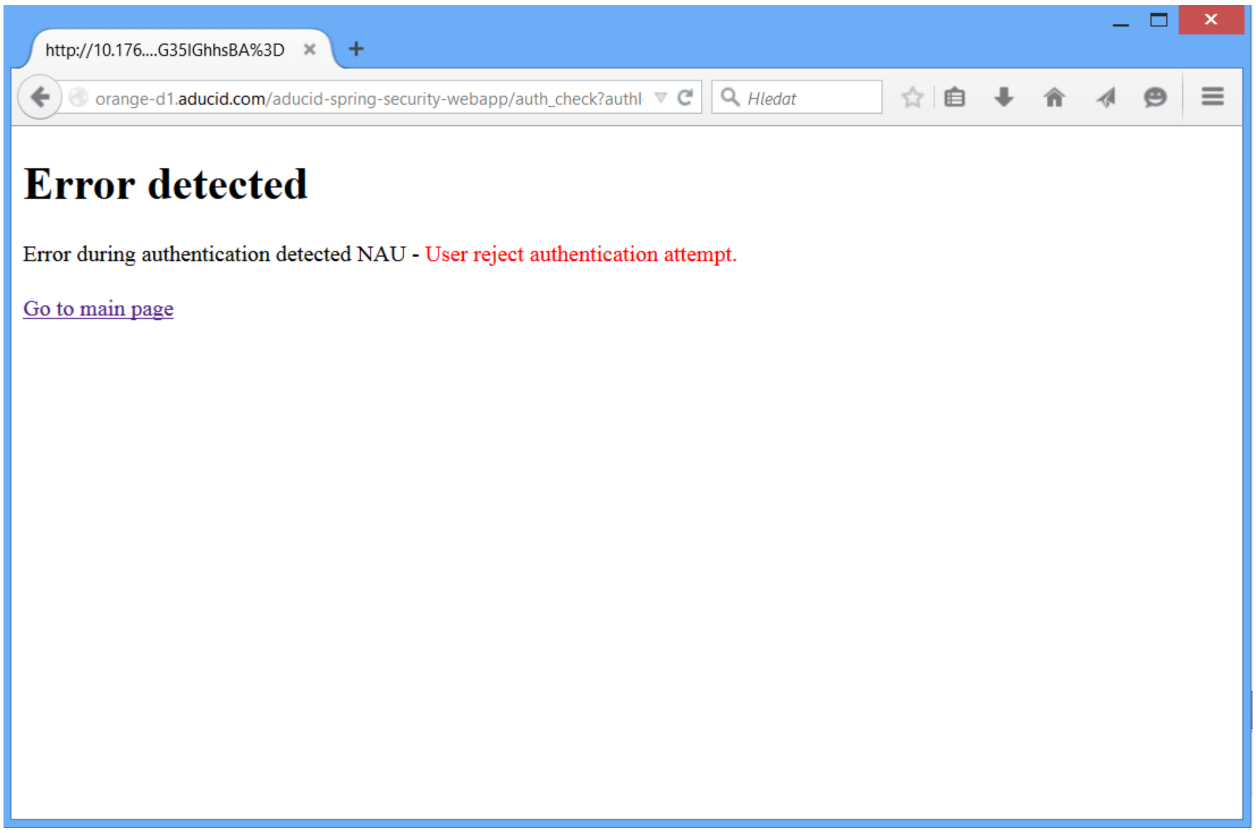
Go to main page

Figure 5-4 Secured admin page

5.5. Error page

Error page is displayed whenever any exception is thrown during authentication process. Two ADUCID specific exceptions can be thrown away.

- AducidAutheticationException – general exception that means something unspecified happened
- AducidStatusException - this exception holds status of the authentication process in attributes AIMStatus and AuthStatus. Based on these two statuses, more specific actions can be taken in the application.


http://10.176....G35IGhhsBA%3D

orange-d1.aducid.com/aducid-spring-security-webapp/auth_check?authI

Error detected

Error during authentication detected NAU - User reject authentication attempt.

Go to main page

Figure 5-5 Error page

6. Appendix A

This appendix contains complete sample xml configuration of ADUCID spring security adapter.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/security"
     xmlns:beans="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns:util="http://www.springframework.org/schema/util"
     xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
                        http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security.xsd
                        http://www.springframework.org/schema/util
http://www.springframework.org/schema/util/spring-util.xsd">

     <!-- HTTP security configurations -->
     <http entry-point-ref="aducidEntryPoint" auto-config="true"
            use-expressions="true">
            <custom-filter before="CAS_FILTER" ref="aducidFilter" />
            <logout logout-url="/logout.html" logout-success-url="/" />
            <intercept-url pattern="/identity.html" access="isAuthenticated()" />
            <intercept-url pattern="/user.html" access="isAuthenticated() and
principal!=null" />
            <intercept-url pattern="/admin.html" access="isAuthenticated() and
hasRole('admin')" />
     </http>

     <util:properties id="aducidProperties">
            <beans:prop key="aimUrl">http://localhost:8080/AIM</beans:prop>
            <beans:prop key="aimProxyUrl">http://localhost:8080/AIM-proxy</beans:prop>
```

```
      </util:properties>

      <beans:bean
      class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
             <beans:property name="properties" ref="aducidProperties"/>
      </beans:bean>

      <authentication-manager alias="aducidAuthenticationManager">
             <authentication-provider ref="aducidAuthenticationProvider" />
      </authentication-manager>

      <beans:bean id="aducidEntryPoint"
             class="com.aducid.spring.security.AducidAuthenticationEntryPoint">
             <beans:property name="aimUrl" value="${aimUrl}/services/R4" />
             <beans:property name="defaultLoginUrl" value="${aimProxyUrl}/process" />
      </beans:bean>

      <beans:bean id="aducidFilter"
             class="com.aducid.spring.security.AducidAuthenticationFilter">
             <beans:property name="authenticationManager" ref="aducidAuthenticationManager"
/>
             <beans:property name="authenticationFailureHandler">
                    <beans:bean id="aducidAuthenticationFailureHandler"
      class="org.springframework.security.web.authentication.SimpleUrlAuthenticationFailure
Handler">
                           <beans:property name="defaultFailureUrl" value="/error.html" />
                           <beans:property name="useForward" value="true" />
                    </beans:bean>
             </beans:property>
      </beans:bean>

      <beans:bean id="aducidAuthenticationProvider"
             class="com.aducid.spring.security.AducidAuthenticationProvider">
             <beans:property name="userDetailService" ref="aducidUserDetailService" />
             <beans:property name="allowNullPrincipal" value="true"/>
             <beans:property name="aimUrl" value="${aimUrl}/services/R4" />
             <beans:property name="personalObjectId" value="TEST" />
      </beans:bean>

      <beans:bean id="aducidUserDetailService"
             class="com.aducid.common.services.DefaultAducidUserDetailService">
             <beans:property name="authoritiesUserAttribute" value="testRole" />
      </beans:bean>

</beans:beans>
```