Authentication

Summary

To access the contents (information or functions) opened to users, it goes through authentication to input an ID and a password.

That is, authentication means the process that determines whether specific user is effective user.
This guide explains the authentication method used in development framework and basic environment for authentication.

Description

The certification of eGov framework uses the DB based JDBC certification rather than XML. The basic certification mechanism is only used once initially.
To use authentication provided by Spring Security, only one AuthenticationManager implement is included and setting of AuthenticationProvider is required.
AuthenticationManager is obligated to deliver the request to AuthenticationProvider chain.
AuthenticationProvider uses UserDetails and UserDetailsService interface by default.
Returned UserDetails of loadUserByUsername method in UserDetailsService method provides the basic authentication information such as username, password, authority(GrantedAuthority[]) and whether to use or not (enabled).
In this regard, e-government framework expands the UserDetails interface and uses the user information from user table based in JDBC in the form of VO.

Minimum configuration

Next is the user certification through minimum configuration.

Configuration

```
<http auto-config='true'>
          <intercept-url pattern="/**" access="ROLE_USER"/>
</http>
```

- <intercept-url> can be defined several and url pattern url pattern corresponding in the order from the above is applied. Define the access url using regular formula. "access" property indicates the authority of user that can access.
- If not designating <form-login>, use the login form provided by Spring Security.

To process the authentication request, authentication manager uses user information defined in xml(or DB, LDAP, etc.).

```
<authentication-provider>
          <user-service>
                    <user name="jimi" password="jimispassword" authorities="ROLE_USER,
ROLE_ADMIN" />
                    <user name="bob" password="bobspassword" authorities="ROLE_USER" />
          </user-service>
</authentication-provider>
```

- user-service: service obtained from user information(user ID, user password, authority, etc.)

JDBC Certification

JdbcDaoImpl provides two properties to modify SQL clause. When more customizing is required, JdbcDaoImpl can be used as separate implementation class.

DAO Certification

Server Security includes the implementation of AuthenticationProvider interface, DaoAuthenticationProvider class which processes DB based certification.
Use the UserDetailsService to obtain the user name, password and authority information from DB.

Sample

```
<bean id="daoAuthenticationProvider"
class="org.acegisecurity.providers.dao.DaoAuthenticationProvider">
        <property name="userDetailsService"><ref bean="inMemoryDaoImpl"/></property>
        <property name="saltSource"><ref bean="saltSource"/></property>
        <property name="passwordEncoder"><ref bean="passwordEncoder"/></property>
</bean>
```

PasswordEncoder and SaltSource are optional. PasswordEncoder provides the encoding and decoding of password contained in UserDetails object returned from UserDetailsService established.
SaltSource enables to spread 'Salt' to the password. This salt reinforces the security of password in the authentication storage.
PasswordEncoder implements provided by Spring Security includes MD5, SHA, cleartext encoding. In Spring Security, 2 SaltSource implements are provided; SystemWideSaltSource executing encoding to the same salt for all password as well as ReflectionSaltSource examining the property of returned UserDetails and obtaining salt. See JavaDoc for details on selective functions.

HTTP Form Certification Mechanism

HTTP form certification uses AuthenticationProcessingFilterto process the log-in form.   This is most widely used format.
Login form includes j_username and j_password input file simply, and posts it to URL monitored by filter. Default value is j_spring_security_check.

Other certification

BASIC certification
Digest certification
Anonymous certification
Remember-Me certification
X509 certification
LDAP certification
CAS certification
Container adapter certification

Sample Configuration

WEB Security service

```
<http access-denied-page="/system/accessDenied.do" path-type="regex" lowercase-
comparisons="false">
        <form-login login-processing-url="/j_spring_security_check"
                        authentication-failure-url="/cvpl/EgovCvplLogin.do?login_error=1"
                        default-target-url="/index.jsp?flag=L"
                        login-page="/cvpl/EgovCvplLogin.do" />
        <anonymous/>
        <logout logout-success-url="/cvpl/EgovCvplLogin.do"/>
</http>
```

- Enable access of users allowed with <http> setting.
- If designating <form-login>, can use the login form directly developed by the developer.

```
<authentication-provider user-service-ref="jdbcUserService">
        <password-encoder hash="md5" base64="true"/>
</authentication-provider>
```

- Function to get user information (user ID, user password, authority, etc.) is taken charge of by jdbcUserService.
- password-encoder: use md5 for encoding algorithm of saved password.

JDBC User Service

The user service that attains the user information can be certified using the user information stored in DB.

```
<jdbc-user-service id="jdbcUserService" data-source-ref="dataSource"
                    users-by-username-query="SELECT USER_ID,PASSWORD,ENABLED,BIRTH_DAY
FROM USERS WHERE USER_ID = ?"
                    authorities-by-username-query="SELECT USER_ID,AUTHORITY FROM AUTHORITIES
WHERE USER_ID = ?"/>
```

Sample Source

```
<form action="<s:url value='/j_spring_security_check'/>" method="POST">
          <table>
                    <tr><td>User:</td><td><input type='text' name='j_username'></td></tr>
                    <tr><td>Password:</td><td><input type='password'
name='j_password'></td></tr>
                    <tr><td colspan='2' align="center"><input name="submit" type="submit" value="
login">
                                        &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
                                             <input name="reset" type="reset"
value="cancel"></td></tr>
          </table>
</form>
```

Session management

The session function of Security Service is implemented using the extended JdbcUserDetailsManager interface of the Spring Security and the session information is managed adding information of other users other than username, password, enabled fields written in the basic table.

Configuration

Configuration

To use the session, the configuration of JDBC certification should be changed to eGov framework server security.
Before change (Not use)

```
<jdbc-user-service id="jdbcUserService" data-source-ref="dataSource"
          users-by-username-query="SELECT USER_ID,PASSWORD,ENABLED FROM USERS WHERE
USER_ID = ?"
          authorities-by-username-query="SELECT USER_ID,AUTHORITY FROM AUTHORITIES WHERE
USER_ID = ?"/>
```

After change (use)

```
<b:bean id="jdbcUserService"
                    class="egovframework.rte.fdl.security.userdetails.jdbc.EgovJdbcUserDetailsManager"
>
          <b:property name="usersByUsernameQuery" value="SELECT
USER_ID,PASSWORD,ENABLED,USER_NAME,BIRTH_DAY,SSN FROM USERS WHERE USER_ID = ?"/>
          <b:property name="authoritiesByUsernameQuery" value="SELECT USER_ID,AUTHORITY
FROM AUTHORITIES WHERE USER_ID = ?"/>
          <b:property name="roleHierarchy" ref="roleHierarchy"/>
```

```
            <b:property name="dataSource" ref="dataSource"/>
            <b:property name="mapClass"
value="egovframework.rte.fdl.security.userdetails.EgovUserDetailsMapping"/>
</b:bean>
```

- class: egovframework.rte.fdl.security.userdetails.jdbc.EgovJdbcUserDetailsManager
- usersByUsernameQuery: inquires user information from user table for user authentication.
- authoritiesByUsernameQuery: inquires user authentication information from user authentication table for user authentication.
- roleHierarchy: sets the hierarchy roles for hierarchical management of role.
- mapClass: sets the mapping class between session query and session VO for session use.
- dataSource: sets dataSource for JDBC.

VO Class

```
public class EgovUserDetailsVO {
            private String userId;
            private String passWord;
            private String userName;
            private String ssn;
            private String lsYn;
            private String birthDay;
            private Integer age;
            private String cellPhone;
            private String addr;
            private String email;

            public String getUserId() {
                        return userId;
            }
            public void setUserId(String userId) {
                        this.userId = userId;
            }
            public String getPassWord() {
                        return passWord;
            }
            public void setPassWord(String passWord) {
                        this.passWord = passWord;
            }
            public String getUserName() {
                        return userName;
            }
            public void setUserName(String userName) {
                        this.userName = userName;
            }
            public String getSsn() {
                        return ssn;
            }
            public void setSsn(String ssn) {
                        this.ssn = ssn;
            }

            .
            .
            .
```

Mapping Class

```
public class EgovUserDetailsMapping extends EgovUsersByUsernameMapping {
            public EgovUserDetailsMapping(DataSource ds, String usersByUsernameQuery) {
                        super(ds, usersByUsernameQuery);
            }
```

```
		@Override
		protected Object mapRow(ResultSet rs, int rownum) throws SQLException {
				String userid = rs.getString("user_id");
				String password = rs.getString("password");
				boolean enabled = rs.getBoolean("enabled");

				String username = rs.getString("user_name");
				String birthDay = rs.getString("birth_day");
				String ssn = rs.getString("ssn");

				EgovUserDetailsVO userVO = new EgovUserDetailsVO();
				userVO.setUserId(userid);
				userVO.setPassWord(password);
				userVO.setUserName(username);
				userVO.setBirthDay(birthDay);
				userVO.setSsn(ssn);

				return new EgovUserDetails(userid, password, enabled, userVO);
		}
}
```

- Inheriting EgovUsersByUsernameMapping class
- mapRow method redefinition
- Data mapping to VO created in ResultSet class
- Returning by containing VO object in EgovUserDetails (userid, password, enabled and userVO)

Using Sessions

Brining Sessions

```
import egovframework.rte.fdl.security.userdetails.util.EgovUserDetailsHelper;
  .
  .
  .

EgovUserDetailsVO user =
		(EgovUserDetailsVO)EgovUserDetailsHelper.getAuthenticatedUser();

assertEquals("jimi",			user.getUserId());
assertEquals("jimi test",	user.getUserName());
assertEquals("19800604",	user.getBirthDay());
assertEquals("1234567890123",	user.getSsn());
```

No certification

```
Boolean isAuthenticated = EgovUserDetailsHelper.isAuthenticated();
assertFalse(isAuthenticated.booleanValue());
```

or

```
assertNull(EgovUserDetailsHelper.getAuthenticatedUser());
```

Concurrents session

```
<http access-denied-page="/system/accessDenied.do" path-type="regex" lowercase-
comparisons="false">
		<form-login login-processing-url="/j_spring_security_check"
						authentication-failure-url="/cvpl/egovCvplLogin.do?login_error=1"
						default-target-url="/index.jsp"
						login-page="/cvpl/egovCvplLogin.do" />
```

```
            <anonymous/>
            <logout logout-success-url="/index.jsp"/>
            <concurrent-session-control max-sessions="1" exception-if-maximum-exceeded="false"
expired-url="/EgovCvplLogout.jsp" />
</http>
```

concurrent-session-control

- max-sessions: maximum number of allowed session
- exception-if-maximum-exceeded: true ⇒ if maximum number of session is exceeded, Exception occurs , false ⇒ if maximum number of session is exceeded, forceful logout.

N. Reference