

Обеспечение безопасности веб-приложений

Общие задачи обеспечения безопасности

- Аутентификация
 - > Проверка того, что пользователь является именно тем, за кого себя выдает
- Авторизация (управление доступом)
 - > Процесс определения прав доступа пользователя к конкретному ресурсу
 - > Пользователь должен быть аутентифицирован
- Конфиденциальность
 - > Защита уязвимых данных при передаче по сети

Требования к безопасности в веб-слое

- Предотвращение доступа неавторизованных пользователей к защищенным веб-ресурсам
 - > При попытке неаутентифицированного пользователя получить доступ к защищенному веб-ресурсу веб-контейнер **автоматически** запрашивает аутентификацию
 - > После аутентификации пользователя веб-контейнер (и/или веб-компоненты) применяет **ограничения доступа**
- Предотвращение чтения и подмены уязвимых данных при передаче по сети
 - > Данные можно защитить с использованием протокола **SSL**

Безопасность в веб-слое — Аутентификация

1. Ввод пользователем данных для аутентификации
 - > Обычно выполняется в браузере
 - > Как правило для аутентификации нужны имя пользователя и пароль
 - > Это часто называется “**вход в систему**”
2. Передача аутентификационных данных на веб-сервер
 - > небезопасно (HTTP) или безопасно (HTTP over SSL)



Безопасность в веб-слое — Аутентификация

3. Проверка идентичности пользователя с использованием **домена безопасности**

- > Веб-контейнер проверяет совпадение аутентификационных данных, введенных пользователем, с хранящимися в домене безопасности
- > В домене безопасности хранится
 - > Имя пользователя, пароль, **группы** и т.д.
- > Организация домена безопасности и управление им зависит от конкретного продукта и операционной среды
 - > LDAP, RDBMS, плоский файл, Solaris PAM, Windows AD



Безопасность в веб-слое — Аутентификация

4. Отслеживание аутентифицированных пользователей в веб-контейнере

- > Внутри веб-контейнера поддерживается состояние сеанса работы пользователя с веб-приложением
- > В запросе ([HttpServletRequest](#)) содержатся данные о «контексте безопасности»
 - > Объект пользователя (Principal), роль, имя пользователя

Безопасность в веб-слое — Авторизация

- Разработчик и/или установщик веб-приложения устанавливают ограничения доступа к ресурсам
 - > Декларативное и/или программное управление доступом
 - > Ограничения устанавливаются для **ролей**, определенных в веб-приложении
 - > Пользователь может исполнять N ролей

Безопасность в веб-слое — Конфиденциальность данных

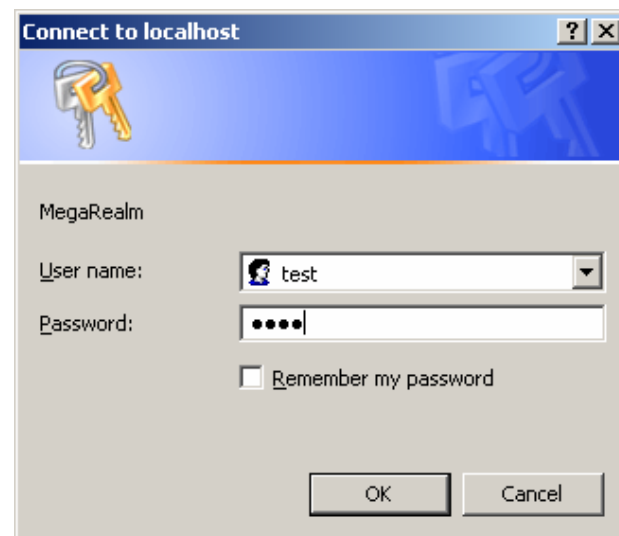
- Обеспечение конфиденциальности уязвимых данных, передаваемых по сети
 - > Между браузером и веб-сервером
 - > Пример: номер кредитной карты
 - > Используется протокол SSL

Механизмы аутентификации

- HTTP Basic
 - > С использованием или без SSL
- Form-based (с использованием формы)
 - > С использованием или без SSL
- Client-certificate
 - > Использование SSL обязательно
- HTTP Digest
 - > Нет необходимости в использовании SSL

Аутентификация HTTP Basic

- Веб-сервер запрашивает аутентификацию, возвращая сообщение с кодом статуса **401**
- Браузер выводит окно для ввода имени пользователя и пароля
- Имя пользователя и пароль передаются в заголовке HTTP-запроса **без шифрования**
- Для шифрования пароля необходимо SSL-соединение



Аутентификация с использованием формы

- Веб-приложение обеспечивает ввод имени пользователя и пароля (а также другой нужной информации) в **форме** на **особой странице входа** (login page)
- Имя пользователя и пароль передаются в теле HTTP-запроса **без шифрования**
- Для шифрования пароля необходимо SSL-соединение

Select your Language

- | | |
|----------------------------------|----------------------------------|
| ● Czech (Czech Republic) | ● Português (Brasil) |
| ● Deutsch (Deutschland) | ● Română (România) |
| ● English (British) | ● Русский |
| ● English (U.S.) | ● Slovenčina (Slovensko) |
| ● Español (Argentina) | ● Suomi (Suomi) |
| ● Español (España) | ● Svensk (Sverige) |
| ● Français | ● Ελληνικά (Ελλάδα) |
| ● Italiano (Italia) | ● Български (Република България) |
| ● Magyar (Magyar Köztársaság) | ● □□□ (□□) |
| ● Nederlands (Nederlands) | ● □□□ |
| ● Norsk (Norge) | ● □□□□ЪChinaЪ |
| ● Polska (Rzeczpospolita Polska) | ● ЪЪ (ЪЪ) |

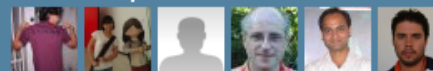
Войти в Open Source University Meetup

Адрес	<input type="text"/>
электронной	
почты	
Пароль	<input type="password"/>
<input type="button" value="Вход"/>	или зарегистрируйтесь

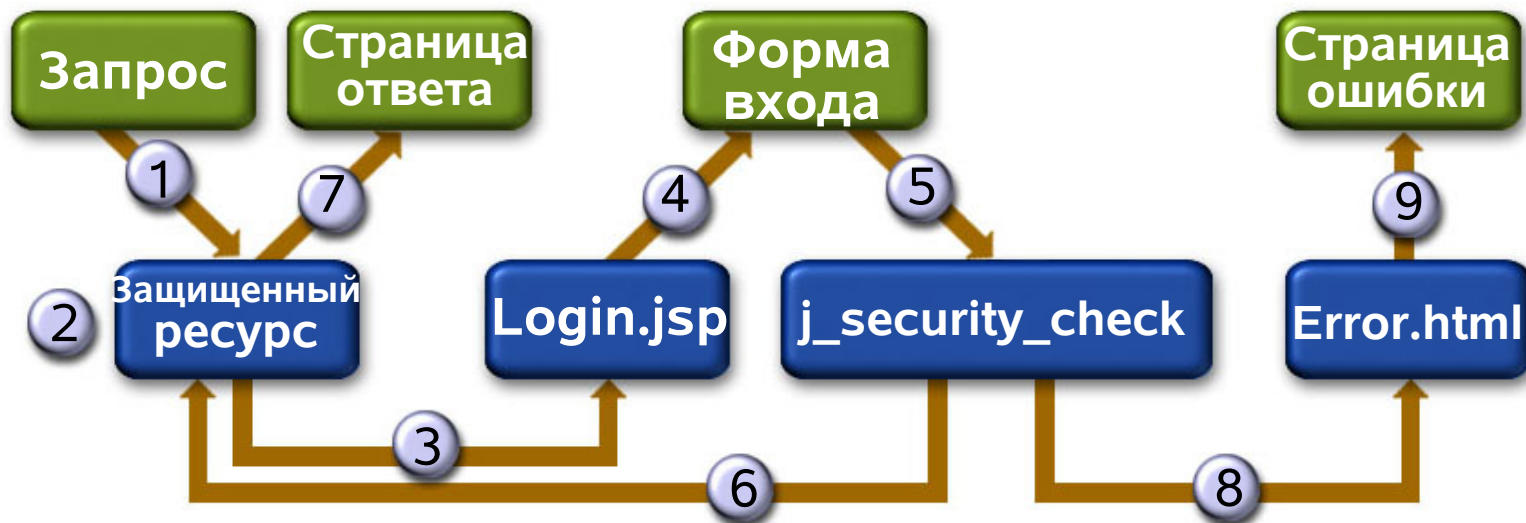
[Забыли пароль?](#)

[Проблемы со входом?](#)

About Open Source University Meetup



Выполнение аутентификации с использованием формы



1. Клиент делает запрос
2. Клиент аутентифицирован?
3. Неаутентифицированный клиент перенаправляется
4. Клиенту возвращается форма входа
5. Клиент отправляет форму входа

6. **Аутентификация** успешна, перенаправление на ресурс

7. Проверка **авторизации**, возврат результата
8. Неудачный вход, перенаправление на страницу ошибки
9. Клиенту возвращается страница ошибки

Сравнение механизмов HTTP Basic и Form-based

HTTP Basic

- Ввод имени и пароля в диалоге браузера
- Можно ввести только имя и пароль
- Внешний вид диалога отличается от сайта
- Для передачи имени и пароля используется HTTP-заголовок
- Нельзя создать нового пользователя

Form-based

- Страница входа в веб-приложение
- Можно ввести любые нужные данные
- Единообразный внешний вид
- Для передачи имени и пароля используется форма
- Можно создать нового пользователя



Аутентификация Client Certificate

- Для аутентификации клиент отправляет веб-серверу **клиентский сертификат** (Client certificate)
 - > Если сервер также аутентифицирует себя перед клиентом, то это взаимная аутентификация
 - > Сертификаты в формате стандарта X.509 — содержат больше, чем имя пользователя / пароль
- Клиентский сертификат нужно создать заранее для каждого клиента
 - > Причина низкой популярности
 - > 1 компьютер — N пользователей,
1 пользователь — N рабочих мест
- Используется протокол HTTP over SSL
- Позволяет выполнять взаимную аутентификацию взаимодействующих программ

Аутентификация HTTP Digest

- Браузер выводит окно для ввода имени пользователя и пароля
- В заголовке HTTP-запроса передается имя пользователя (без шифрования) и «дайджест» пароля
 - > Алгоритм хеширования MD5
 - > По «дайджесту» нельзя получить исходный пароль
 - > При изменении хотя бы одного бита в исходном пароле изменяется и значение «дайджеста»
 - > Пароль не передается в открытом (или Base64-кодированном) виде даже по не-SSL соединению
- Сервер сравнивает полученный «дайджест» со своим, вычисленным по исходному паролю
- Проблемы с **интероперабельностью**
 - > Неправильная реализация в IE5, IE6, IIS
(http://en.wikipedia.org/wiki/Digest_access_authentication)

Управление доступом (авторизация)

Декларативное

- Правила доступа объявляются в установочном дескрипторе
- Контейнер выполняет управление доступом
- «Все или ничего» - доступ либо предоставляется, либо нет

Программное

- Управление доступом кодируется в программе
- Программный код выполняет управление доступом
- Возможно «тонкое» управление доступом
 - > Управление доступом на основе бизнес-логики
 - > Частичное сокрытие данных объекта

Декларативное и программное управление доступом

- Часто комбинируются
 - > Декларативное ограничение обращений к компоненту
 - > Функционирование компонента зависит от роли пользователя, которая проверяется программно
- Пример: вывод списка сотрудников
 - > Разрешен только сотрудникам организации (декларативное ограничение)
 - > Зарплата сотрудников отображается только для пользователей с ролью manager (программное ограничение)

Декларативная авторизация + аутентификация Form-based

- 1) Определить роли в веб-приложении
- 2) Указать, доступ к каким веб-ресурсам должен быть ограничен
- 3) Указать, для каких веб-ресурсов должна гарантироваться целостность передаваемых данных и конфиденциальность (протокол SSL)
- 4) Указать использование для веб-приложения аутентификации с использованием формы
- 5) Создать страницы «Вход в систему», «Ошибка входа в систему»
- 6) Настроить домен безопасности (пользователей, группы)
- 7) Установить соответствие пользователей/групп и ролей



Шаг 1: Определить роли

- В установочном дескрипторе веб-приложения (файл **web.xml**)

```
<web-app>
```

```
...
```

```
  <security-role>
```

```
    <description>Administrator</description>
```

```
    <role-name>admin</role-name>
```

```
  </security-role>
```

```
  <security-role>
```

```
    <description>Executive Staff</description>
```

```
    <role-name>executive</role-name>
```

```
  </security-role>
```

```
...
```

```
</web-app>
```



Шаг 2: Определить ограничения доступа

<web-app>

```
...
<security-constraint>
  <web-resource-collection>
    <web-resource-name>WRCollection</web-resource-name>
    <url-pattern>/loadpricelist</url-pattern>
    <http-method>GET</http-method>
  </web-resource-collection>
  <auth-constraint>
    <role-name>admin</role-name>
    <role-name>executive</role-name>
  </auth-constraint>
  <user-data-constraint>
    <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>
<login-config>
  <auth-method>FORM</auth-method> <realm-name></realm-name>
</login-config>
...
```

</web-app>



Шаг 3: Определить требования к способу передачи запросов

<web-app>

...

<security-constraint>

<web-resource-collection>

<web-resource-name>WRCollection</web-resource-name>

<url-pattern>/loadpricelist</url-pattern>

<http-method>GET</http-method>

</web-resource-collection>

<auth-constraint>

<role-name>admin</role-name>

</auth-constraint>

<user-data-constraint>

<transport-guarantee>CONFIDENTIAL</transport-guarantee>

</user-data-constraint>

</security-constraint>

<login-config>

<auth-method>FORM</auth-method> <realm-name></realm-name>

</login-config>

...

</web-app>

Шаг 4: Указать механизм аутентификации

- В установочном дескрипторе веб-приложения (файл **web.xml**)

```
<web-app>
```

```
...
```

```
<security-constraint>...</security-constraint>
```

```
<login-config>
```

```
  <auth-method>FORM</auth-method>
```

```
  <realm-name>realm name</realm-name>
```

```
</login-config>
```

```
...
```

```
</web-app>
```

- Другие возможные значения <auth-method>:
BASIC, CLIENT-CERT, DIGEST

Шаг 5: Создать страницы «Вход в систему», «Ошибка входа»

- Допускаются и статические, и динамические страницы (и HTML, и JSP)
- Страница «Вход в систему» должна содержать:

```
<FORM ACTION="j_security_check" METHOD="POST">  
...  
<INPUT TYPE="TEXT" NAME="j_username">  
...  
<INPUT TYPE="PASSWORD" NAME="j_password">  
...  
</FORM>
```

- Страница «Ошибка входа в систему» может быть любой

Шаг 6: Настроить домен безопасности

- Каждый сервер приложений предоставляет **собственные** средства для конфигурирования доменов безопасности
 - > В спецификации сервлетов не определен интерфейс между веб-сервером и реализацией домена безопасности
- Реализация домена безопасности выбирается, исходя из **требований** операционной среды
 - > Плоские файлы, база данных, LDAP-каталог
 - > Хранение паролей с шифрованием или без

Шаг 6: Настроить домен безопасности

- Домен безопасности, используемый в Tomcat по умолчанию
 - > Файл <install-dir>/config/tomcat-users.xml
 - > Пароли не шифруются — простота настройки и поддержки

```
<?xml version='1.0'?>
```

```
<tomcat-users>
```

```
  <role rolename="manager"/>
```

```
  <role rolename="employee"/>
```

```
  <role rolename="admin"/>
```

```
  <user username="sang" password="sangPassword"  
    roles="manager,employee"/>
```

```
</tomcat-users>
```

} Фактически это группы,
а не роли



Шаг 6: Настроить домен безопасности

Источник	Tomcat 6	SJSAS 9
1. Файл с данными о пользователях	MemoryRealm (по умолчанию), список пользователей хранится в XML-файле, по умолчанию – <code>conf/tomcat-users.xml</code>	file.FileRealm , определены домены file (по умолчанию) и admin-realm , список пользователей хранится в текстовом файле
2. Реляционная база данных	JDBCRealm (подключение через <code>DriverManager</code>), DataSourceRealm (подключение через источник данных)	jdbc.JDBCRealm (подключение через источник данных)
3. LDAP-каталог	JNDIRealm	ldap.LDAPRealm
4. Учетные записи пользователей операционной системы	—	solaris.SolarisRealm , определен домен solaris
5. Хранилище сертификатов	—	certificate.CertificateRealm , определен домен certificate
6. Модуль JAAS (Java Authentication & Authorization Service)	JAASRealm	—

Шаг 6: Настроить домен безопасности

[Home](#) [Version](#)
User: admin | **Domain:** domain1 | **Server:** localhost
Sun Java™ System Application Server Admin Console

- Service Assemblies
- Components
- Shared Libraries
- Custom MBeans
- Resources
- Configuration
 - Web Container
 - EJB Container
 - Java Message Service
 - Security
 - Realms**
 - JACC Providers
 - Audit Modules
 - Message Security
 - Transaction Service
 - HTTP Service

Configuration > Security > Realms

Realms

Manage security realms.

Realms (3)

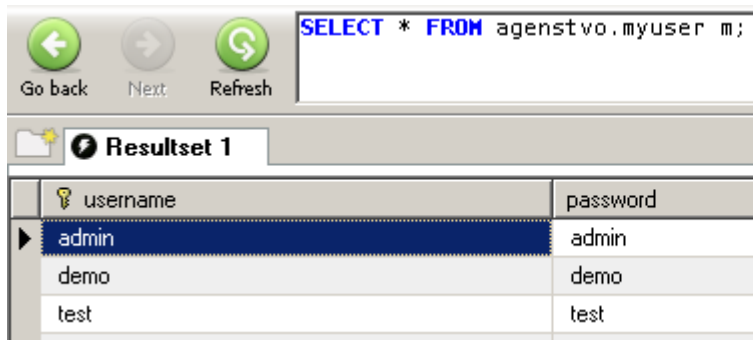
[New...](#) [Delete](#)

	Name	Classname
<input type="checkbox"/>	certificate	com.sun.enterprise.security.auth.realm.certificate.CertificateRealm
<input type="checkbox"/>	file	com.sun.enterprise.security.auth.realm.file.FileRealm
<input type="checkbox"/>	admin-realm	com.sun.enterprise.security.auth.realm.file.FileRealm

Шаг 6: Настроить домен безопасности

- Домен безопасности, основанный на базе данных
 - > БД содержит данные о пользователях и группах

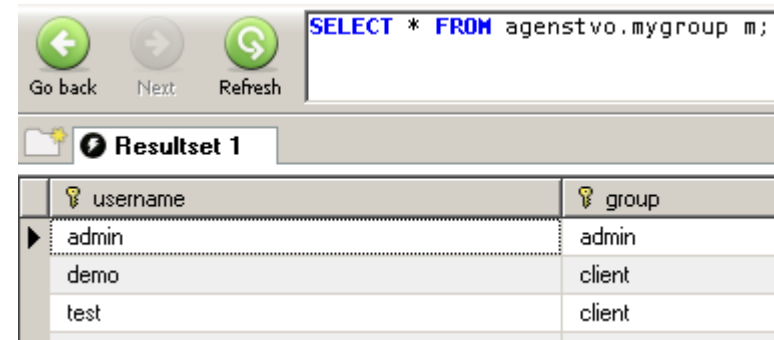
```
CREATE TABLE myuser (  
  `username` varchar(10) NOT NULL,  
  `password` varchar(20) NOT NULL,  
  `fio` varchar(100) NOT NULL,  
  `birthdate` date NOT NULL,  
  PRIMARY KEY (`username`)  
);  
CREATE TABLE mygroup (  
  `username` varchar(10) NOT NULL,  
  `group` varchar(20) NOT NULL,  
  PRIMARY KEY (`username`, `group`),  
  CONSTRAINT `FK_myuser_group_1` FOREIGN KEY (`username`)  
    REFERENCES myuser (`username`)  
);
```



SELECT * FROM agens tvo.myuser m;

Resultset 1

username	password
admin	admin
demo	demo
test	test



SELECT * FROM agens tvo.mygroup m;

Resultset 1

username	group
admin	admin
demo	client
test	client

Шаг 6: Настроить домен безопасности

> Создание домена безопасности в SJSAS Admin Console

Configuration > Security > Realms

New Realm

Create a new security realm.

Name: *

Class Name: ☒

☐

Class name for the realm you want to create

Properties specific to this Class

JAAS context: *

JNDI: *

User Table: *

User Name Column: *

Password Column: *

Group Table: *

Group Name Column: *

Assign Group:

Database User:

Database Password:

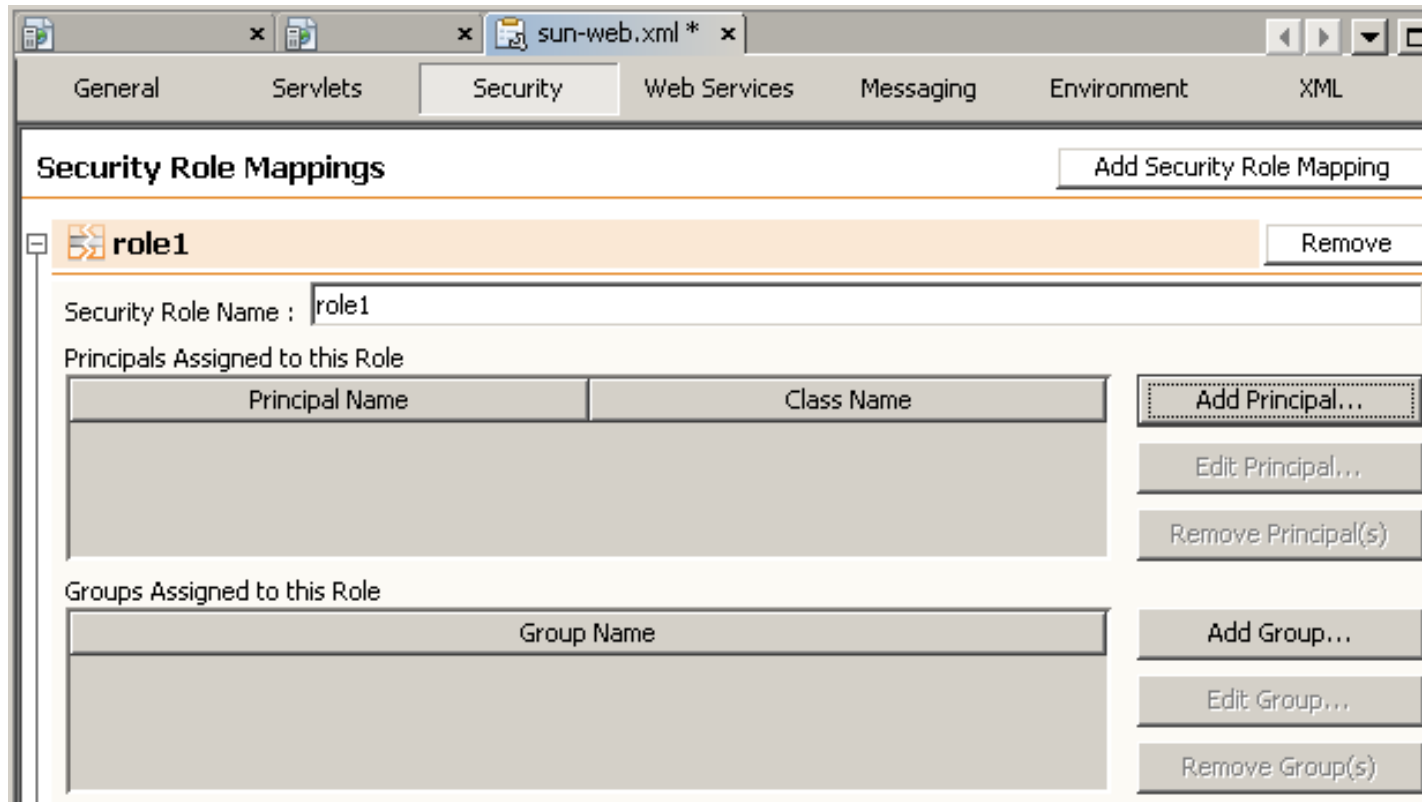
Digest Algorithm:

Encoding:

Charset:


Шаг 7: Установить соответствие пользователей/групп и ролей

- В установочном дескрипторе, специфичном для сервере приложения
 - > [sun-web.xml](#) для Glassfish и SJSAS



General Servlets **Security** Web Services Messaging Environment XML

Security Role Mappings [Add Security Role Mapping](#)

 **role1** [Remove](#)

Security Role Name :

Principals Assigned to this Role

Principal Name	Class Name
----------------	------------

[Add Principal...](#)
[Edit Principal...](#)
[Remove Principal\(s\)](#)

Groups Assigned to this Role

Group Name

[Add Group...](#)
[Edit Group...](#)
[Remove Group\(s\)](#)

Программная авторизация

- 1) Разработчик сервлета вставляет в код сервлета **программную логику авторизации**, в которой используются **абстрактные роли**
- 2) Установщик определяет соответствие между **абстрактными** и **действительными ролями** (для конкретной операционной среды) в **web.xml**
- 3) Установщик настраивает домен безопасности (пользователей, группы)
- 4) Установщик определяет соответствие пользователей/групп и ролей

Шаг 1: Программная логика авторизации

```
public interface javax.servlet.http.HttpServletRequest{
```

```
...
```

```
// Получение информации о пользователе
```

```
public java.security.Principal getUserPrincipal();
```

```
public String getRemoteUser();
```

```
// Исполняет ли пользователь определенную роль?
```

```
public boolean isUserInRole(String role);
```

```
...
```

```
}
```




Шаг 1: Пример — сотрудники видят только свою зарплату

```
public double getSalary(String employeeId) {  
    java.security.Principal userPrincipal =  
        request.getUserPrincipal();  
    String callerId = userPrincipal.getName();  
  
    // Роль "manager" видит зарплату любого сотрудника  
    // Роль "employee" видит только свою зарплату  
    // (используются абстрактные роли)  
    if ( (request.isUserInRole("manager")) ||  
        ((request.isUserInRole("employee")) &&  
         (callerId == employeeId)) ) {  
        // вернуть зарплату сотрудника  
        getSalaryInformationSomehow(employeeId);  
    } else {  
        throw new SecurityException("доступ запрещен");  
    }  
}
```



Шаг 2: Соответствие абстрактных и действительных ролей

```
<web-app>
```

```
...
```

```
<servlet>
```

```
  <servlet-name>...</servlet-name>
```

```
  <servlet-class>...</servlet-class>
```

```
  <security-role-ref>
```

```
    <role-name>manager</role-name>
```

<!-- абстрактная роль -->

```
    <role-link>managerOfAcme</role-link>
```

<!-- действительная роль -->

```
  <security-role-ref>
```

```
</servlet>
```

```
...
```

```
</web-app>
```