Securing remote JSON Web Services

Liferay dev.life session @ 23. 09. 2014

😸 Tomáš Polešovský 😸
Software Security Engineer

Session Overview

1. Basics

2. Create Public + Authenticated JSON WS

3. Add Authorization

4. Consuming Services From 3rd Party Site
4.1. Using JSONP
4.2. Using CORS

5. API Access Control Extension Points

1. Basics

Basics

1. Obsolete JSON interface
/c/portal/json_service

~ **json**.service.* portal properties
… will be probably deprecated shortly

2. New /api/jsonws interface

~ **jsonws**.web.service.* portal properties

Security Basics

1. Portal is configured to be secure-by-default
=> secure unless you “enhance” it

Example: keep CSRF token check enabled

```
json.service.auth.token.enabled=true
auth.token.check.enabled=true
```

2. Services are responsible for data permission checks to prevent unauthorized access and/or modification of data

```
permissionChecker.hasPermission(...)
```

JSON WS Basics

JSON WS is built around Liferay Remote services

```
<?xml version="1.0"?>
<!DOCTYPE service-builder PUBLIC "-//Liferay//DTD Service Builder 6.2.0//EN" "ht

<service-builder package-path="com.liferay.devlife">
    <namespace>jsonws</namespace>
    <entity name="User" local-service="false" remote-service="true" />
</service-builder>
```

Liferay Services:

- are Spring services with a lot of Spring advices
- have naming conventions
- honor custom class hierarchy

=> it's not easy to make any Spring service work with JSON WS

Good news everyone: this will change with Liferay 7 and OSGI

p_auth and CSRF

- JSON WS services and any public API suffers from Cross-site Request Forgery issue

→ accessing /api/jsonws with authenticated browser can trigger API execution
→ <img src=”/api/jsonws/change-email/evil@attacker.com”> can change current user email address

=> when browser accesses /api/jsonws, there must be p_auth token to prevent CSRF

2. Create Public + Authenticated JSON WS

Public JSON Web Service

~ can be accessed without authentication

Is defined by

```
@AccessControlled(guestAccessEnabled = true)
```

LPS-44605 – public method

- cannot call other services that require authentication
  - has to perform all permission checking itself

🐱 *Time for coding* 🐱

Public and authenticated methods

https://gist.github.com/topolik/4adc3cd300b5f9321631

- Create empty portlet plugin

- Add service.xml

```
ant build-service
```

- Create methods and deploy

```
ant build-service deploy
```

- Test using http://localhost:8080/jsonws/api/

3. Add Authorization

Add Authorization

Only the service is responsible for data permission checking to prevent illegal access to it

Authorization is not only data permission checking

- We can check remote authentication method
- We can check current time
- We can check whatever we need to check

🐱 *Time for coding* 🐱

Add Authorization

https://gist.github.com/topolik/3b047c7751c3b727ab2d

- Filter users based on permissions
- Check access to method based on parameter

… update service.xml, create methods and deploy

```
ant build-service deploy
```

- Test using http://localhost:8080/jsonws/api/

3. Consuming Services From 3rd Party Site

Consuming From 3rd Party Site

Let's stick to JavaScript apps only, which use browser cookies or custom HTTP headers to authenticate

The main problem are cross-origin calls. Example: my app at http://alpha.cz wants to consume services of http://beta.cz

Cross-origin calls are prohibited:
→ request can be submitted and without CSRF token will succeed
→ **but response cannot be read by JavaScript**

~> JSONP to the rescue, still there are security concerns
~> Cross-Origin Resource Sharing (CORS) should help

 *Time for coding* 

Consume service using JSONP

https://gist.github.com/topolik/5033c64bc56247cf9439

- Add a service
- Whitelist service from CSRF check using hook
- Deploy
- Add alpha.cz into hosts file
- Go to alpha.cz:8080, add WCM
- Consume localhost:8080 service from alpha.cz:8080 using JSONP

- Note: Doesn't work in all browsers

🐱 *Time for coding* 🐱

Consume service using CORS

https://gist.github.com/topolik/ad7d776ed573a2d5208d

- Add a service
- Add a CORS filter using hook
- Deploy
- Add alpha.cz into hosts file
- Go to alpha.cz:8080, add WCM
- Consume localhost:8080 service from alpha.cz:8080 using CORS

5. API Access Control Extension Points

Access Control Extension Points

AuthVerifier API (Liferay 6.2)

AuthVerifier interface for pluggable authentication

```
public interface AuthVerifier {

public String getAuthType();

public AuthVerifierResult verify(
    AccessControlContext accessControlContext, Properties properties)
  throws AuthException;
}
```

🐱 *Time for coding* 🐱

Add new header-based authentication

https://gist.github.com/topolik/088ccd025d819fad502a

- Add an AuthVerifier using hook
- Deploy
- Test

```
curl -i 'http://localhost:8080/api/jsonws/jsonws-services-portlet.example4/get-current-user' -H 'X-Secret: X-Files'
```

Access Control Extension Points

AccessControl API (Liferay 7)

Using AccessControlAdvisor it's possible to register custom policy class

The class executes before accessing remote service.

Example: AccessControlAdvisorImpl, currently responsible mainly for guarding public access

(the API is private in 6.2 ~> possibility to extend using EXT plugin)

Questions?

Thank you!