

# CS304

## Software Engineering

**TAN, Shin Hwei**

陈馨慧

**Instructor: 胡春风**

Southern University of Science and Technology

Reference: <https://www.youtube.com/watch?v=T807GT0XCg4>

# Security Engineering

- Watch the video “Security risks and common mistakes in mobile application development” in lab folder (From: <https://www.youtube.com/watch?v=T807GT0XCg4>)
- Accept the invitation links at:
  - <https://classroom.github.com/a/RwgDwl38>

# Security Engineering



# Top 10 Java Vulnerabilities

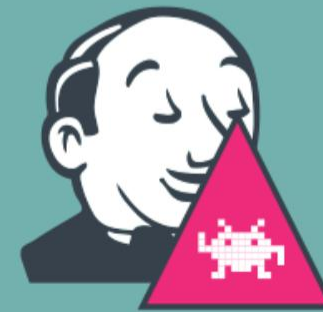
## 1 JUnit

JUnit files that come with other applications can harbor vulnerabilities. For example, versions of the Google Web Toolkit (GWT) before 2.5.1 RC contain multiple cross-site scripting (XSS) vulnerabilities.



## 2 Jenkins

Popularity as continuous integration tool usually means more vulnerabilities and exploits—and in Jenkins' case, multiple XSS, cross-site request forgery (CSRF), and denial-of-service (Dos) vulnerabilities exist.



# Top 10 Java Vulnerabilities

## 3 Hibernate

Hibernate versions 4.1.0 before 4.2.1, 4.3.x before 4.3.2, and 5.x before 5.1.2 of the open source tool contain a vulnerability that can allow attackers to bypass Java Security Manager (JSM).



## 4 Maven

A vulnerability in Apache Maven 3.0.4 allows for remote hackers to spoof servers in a man-in-the-middle attack.



# Top 10 Java Vulnerabilities

## 5 Tomcat

Tomcat has amassed a relatively impressive range of security gaps—from XSS to CSRF vulnerabilities—many of which have been exploited in the wild.



## 6 Java 7

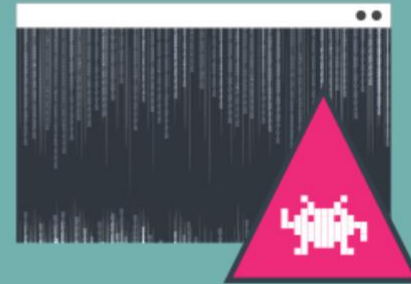
Any version of Java below 7 should be updated immediately—even version 7 needs significant remediation for its fleet of vulnerabilities.



## Top 10 Java Vulnerabilities

### 7 Spring Framework

As an open source project, Spring is not without its fair share of documented vulnerabilities.



### 8 JavaServer Faces

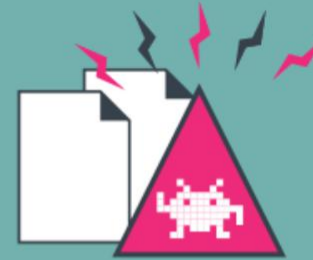
A vulnerability in Apache MyFaces Core 2.0.x before 2.0.12 and 2.1.x before 2.1.6 can give remote attackers the ability to read arbitrary files.



# Top 10 Java Vulnerabilities

## 9 Eclipse IDE

Certain versions of Eclipse IDE help files are vulnerable to XSS-related exploits.



## 10 Vaadin

An XSS vulnerability in the Vaadin framework can allow remote attackers to inject arbitrary scripts into the pages.





# What is cross-site scripting (XSS)?

- Injection attack.
- When an attacker uses a trusted web site to send malicious code to an unsuspecting user, generally in the form of a JavaScript or HTML browser-side script.

A benign user enters a comment as intended.

## Show me an example of XSS

Your site has this:

```
<b> <%= username %> </b></br>
<div> <%= profileText %> </div>
```

A benign user enters a comment as intended.

```
http://www.example.com/saveComment?
comment=Great+Site!
```

```
<h3> Thank you for your comment! </h3>
```

You wrote:

```
<p/>
Great Site!
<p/>
```

An attacker uses the field to send malicious code to your site.

```
http://www.example.com/saveComment?comment=
<script>alert('xss');</script>
```

```
<h3>Thank you for your comments!</h3>
```

You wrote:

```
<p/>
<script>alert('xss');</script>
<p/>
```

# What is cross-site scripting (XSS)?

## Show me an example of XSS

Your site has this:

```
<b> <%= username %> </b></br>
<div> <%= profileText %> </div>
```

A benign user enters a comment as intended.

A benign user enters a comment as intended.

```
http://www.example.com/saveComment?
comment=Great+Site!
<h3> Thank you for your comments! </h3>
You wrote:
<p/>
Great Site!
<p/>
```

**Benign (good) users**

An attacker uses the field to send malicious code to your site.

```
http://www.example.com/saveComment?comment=
<script>alert('xss');</script>
<h3>Thank you for your comments!</h3>
You wrote:
<p/>
<script>alert('xss');</script>
<p/>
```

**Attackers**

# XSS Prevention

Getting it right: This is the easiest case. Just HTML escape the dynamic content:

```
<div>
  Hello ${cov:htmlEscape(name)}!
</div>
```

```
11 +
12 + } 🚫
```

```
▼ 18 ■■■■■ ...marconivr/jacopo/microblog/security/services/implementations/SimpleSanitationService.java 📄 ...
...  ... @@ -0,0 +1,18 @@
1 + package edu.marconivr.jacopo.microblog.security.services.implementations;
2 +
3 + import org.springframework.stereotype.Service;
4 + import org.springframework.web.util.HtmlUtils;
5 +
6 + import edu.marconivr.jacopo.microblog.security.services.ISanitationService;
7 +
8 + @Service
9 + public class SimpleSanitationService implements ISanitationService
10 + {
11 +
12 +     @Override
13 +     public String escapeHTML(String original)
14 +     {
15 +         return HtmlUtils.htmlEscape(original);
16 +     }
17 +
18 + } 🚫
```

# Have you ever encountered a security problem in apps?

- For example, 步道乐跑

9:27 2.63K/s 联通 4G 87%

< 身份认证 修改信息

我是  学生  教职工

学校 南方科技大学

院系 致仁书院

姓名 王小二

学号 66666666

性别 男

入学年份 2011级

辅助证明材料  易碎品

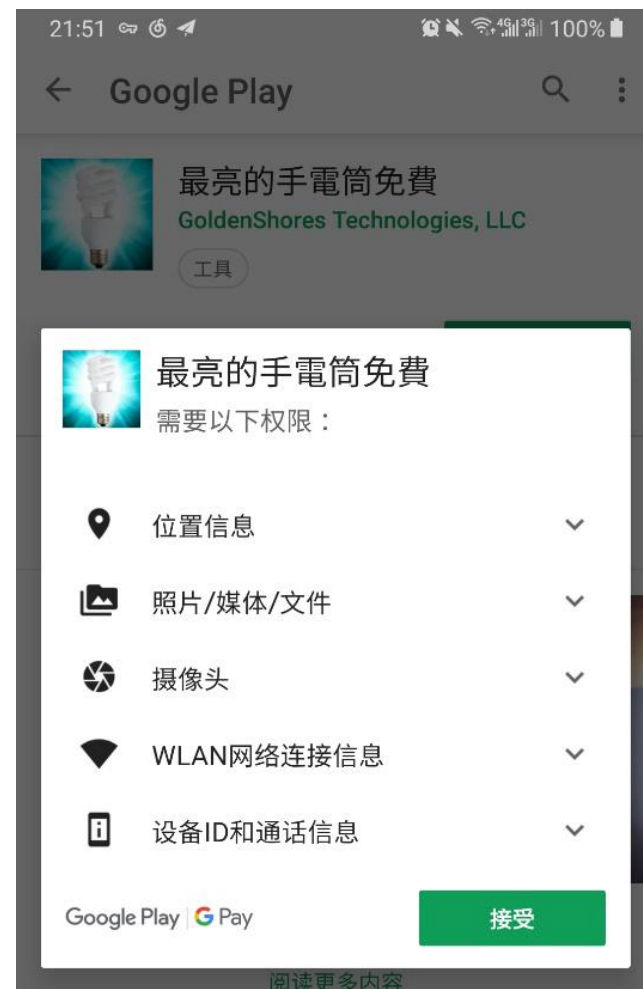
请上传清晰的一卡通或学生证照片 [查看示例](#)

9:27 0.00K/s 联通 4G 87%

< 步道乐跑 DECODE AS

TEXT

```
{
  "data": {
    "id": "446194",
    "uid": "1642108",
    "role": "1",
    "student_num": "66666666",
    "name": "王小二",
    "sex": "男",
    "avatar": "http://data.lptiyu.com/Public/Upload/pic/student_photo/2018-09-04/313/0e14c9d9-501d-490b-988e-449267125c71.png",
    "auth_password": "",
    "school_id": "264",
    "academy_id": "5196",
    "major_id": "0",
    "class_id": "0",
    "status": "1",
    "grade_id": "2011",
    "school": "南方科技大学",
    "academy": "致仁书院",
    "major": "",
    "class": "",
    "status_info": "待认证",
    "grade": "2011级",
    "modify_num_left": "999",
    "auth_type": "1",
    "auth_tips": [
      "辅助证明材料",
      "请上传清晰的一卡通或学生证照片"
    ],
    "auth_photo": "http://api.lptiyu.com/run/Public/Upload/pic/example_photo/whqgd.jpg",
    "school_list": [
      {
        "id": "2",
        "name": "武汉大学",
        "auth_type": "2",
        "auth_tips": [
          "请输入学校提供的认证密码"
        ],
        "auth_photo": "",
        "auth_password": "123456"
      }
    ]
  }
}
```



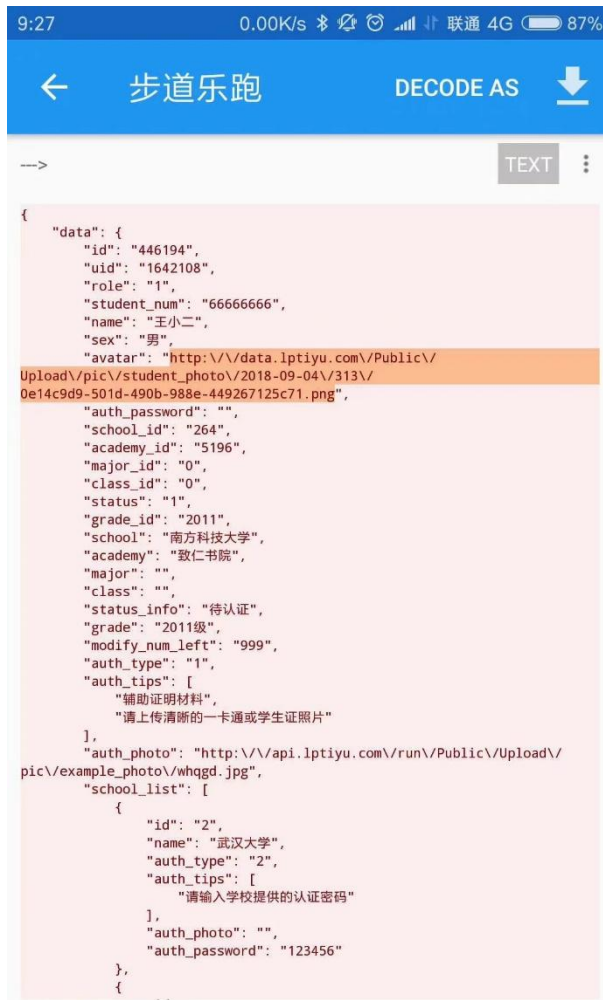
## Question: How many security risk types are there?

- Weak Server Side Controls
- Insufficient Transport Layer Protection
- Poor authentication and Authorization
- Improper Session Handling
- Sensitive Information Disclosure

## Weak Servier Side Controls

- What will you do when you find out a company handle your username and password in plain text?
- What will happen if their database was attacked?
- Question: How many of you store data in plain text in your App?
- You should encrypt the sensitive information in the server database, including username, password, location, tokens, etc.

# Insufficient Transport Layer Protection



- If a hacker attacks your network, will your app becomes insecure?
- This picture again.
- Question: How many of you transfer data like 步道乐跑?
- Do not be like them!

# Poor Authentication and Authorization

- Misusing already built in authentication libraries.
- Develop security logic to be managed and checked in the API and not in the APP itself. Because the client can be DECOMPILED!



# Improper Session Handling

- Session Must TIMEOUT in case of inactively!
- 15 minutes for high security apps
- 30 minutes for medium security apps
- 1 hour for low priority apps

## Sensitive Information Disclosure

- APIs can be crawled by search engine bots and hackers
- Expose only required information to your API!
- Carefully design your API, if someone maliciously attacks your public API, it should be robust enough to defence itself!

# Answer the questions in Github Classroom README.md

---

Link: <https://classroom.github.com/a/RwgDwl38>

What kind of security problems you have in your app?

What you can do to induce the risk and improve security?

# Reminder

- **Project Final Presentation Uploaded:**
  - due on 22 May 2021, 11.59pm
- **All lab exercises due on 24 May 2021, 11.59pm**
  - coverage lab: <https://classroom.github.com/a/S8hiiL1J>
  - junit lab(Pair programming): <https://classroom.github.com/g/trrjQPYS>
  - metrics lab: <https://classroom.github.com/a/qMOJKfSm>
  - reverse engineering lab: <https://classroom.github.com/a/4Dfhej1A>
  - ui-ci: <https://classroom.github.com/a/Yuzgn5gJ>
  - security (this week): <https://classroom.github.com/a/RwgDwl38>