

# WebGoat使用指南

郭苏越、游伟  
(中国人民大学)





## Webgoat介绍

WebGoat是OWASP组织研制出的用于进行web漏洞实验的Java靶场程序，用来说明web应用中存在的安全漏洞。WebGoat运行在带有java虚拟机的平台之上，当前提供的训练课程有30多个，其中包括：跨站点脚本攻击（XSS）、访问控制、线程安全、操作隐藏字段、操纵参数、弱会话cookie、SQL盲注、数字型SQL注入、字符串型SQL注入、web服务、Open Authentication失效、危险的HTML注释等等。

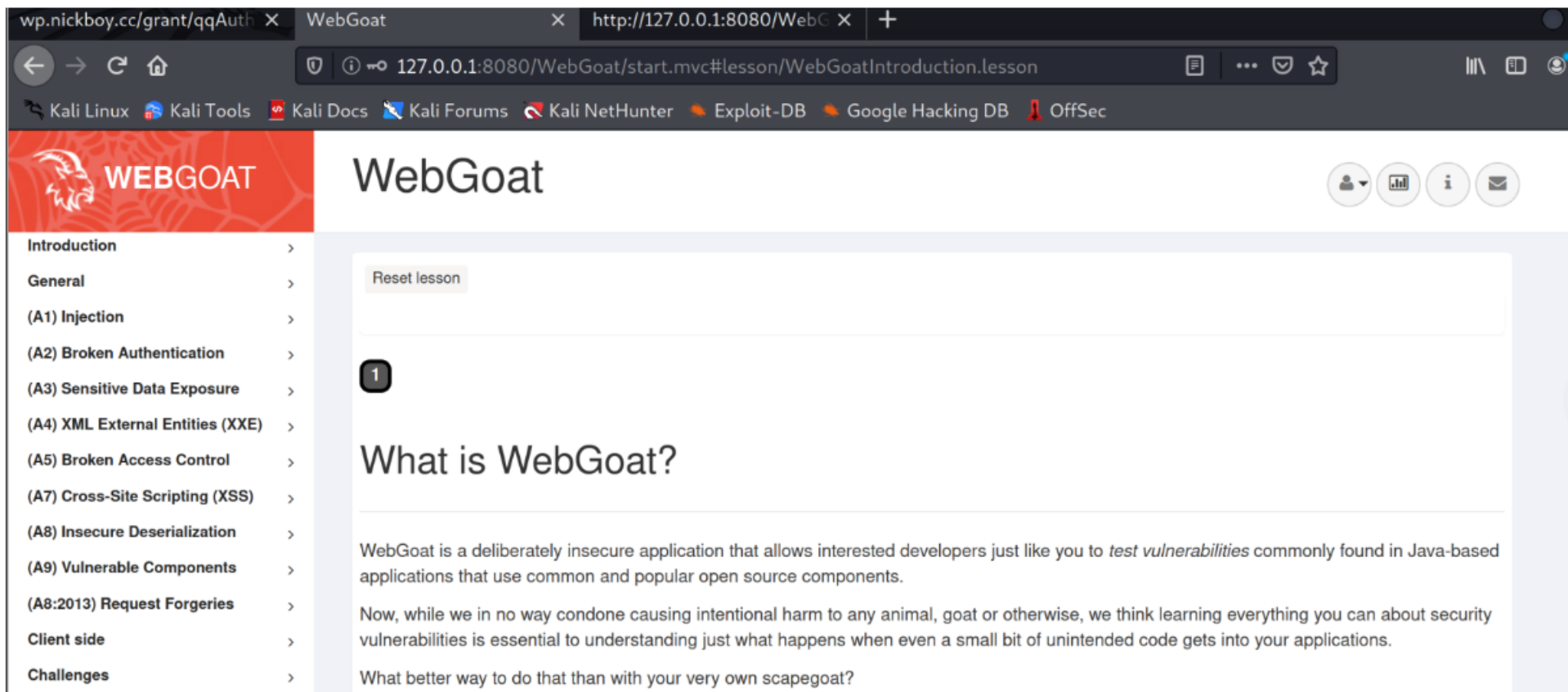
GitHub地址为<https://github.com/WebGoat/WebGoat>

---



# Webgoat使用

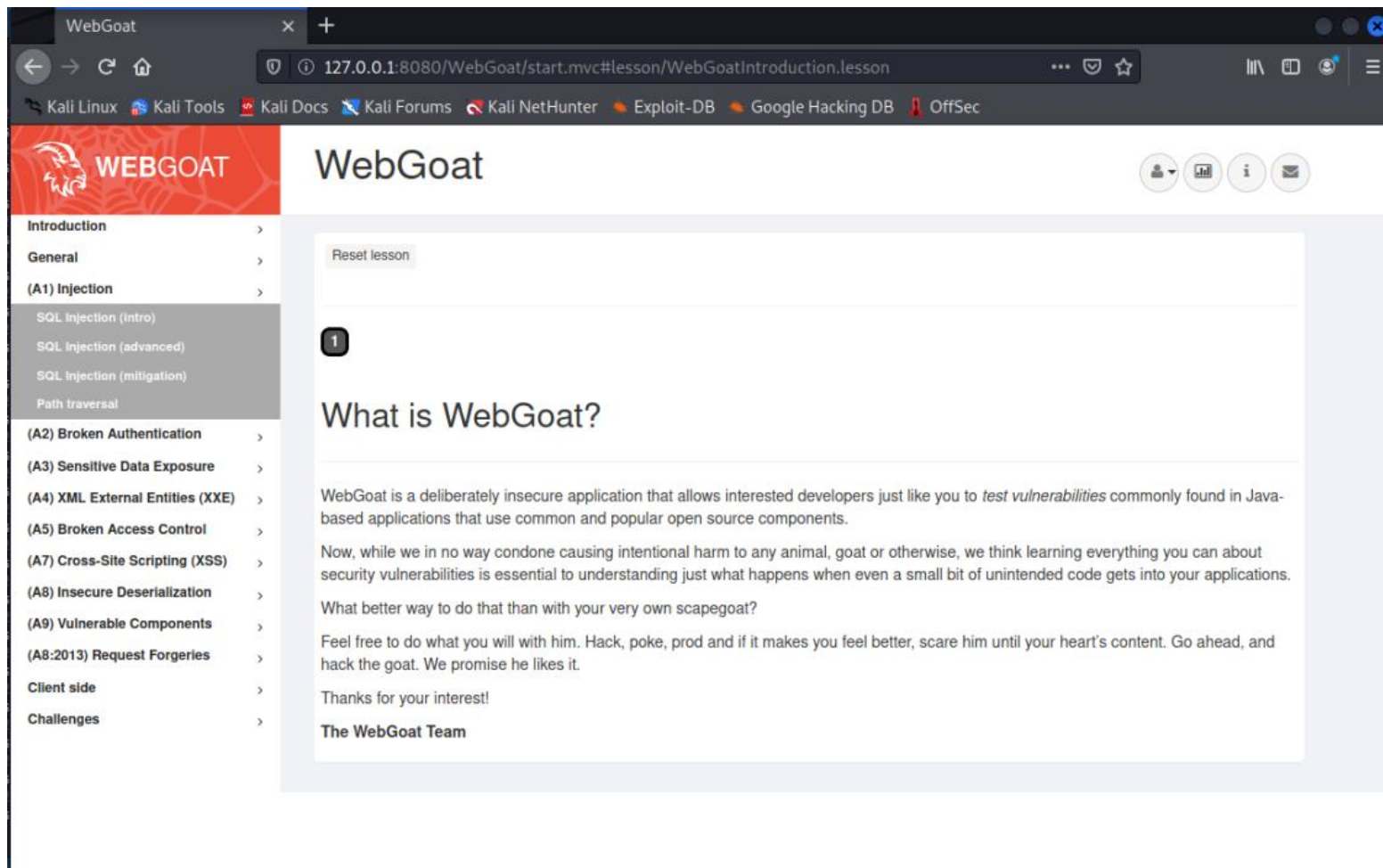
下图为webgoat初始界面





# Webgoat使用

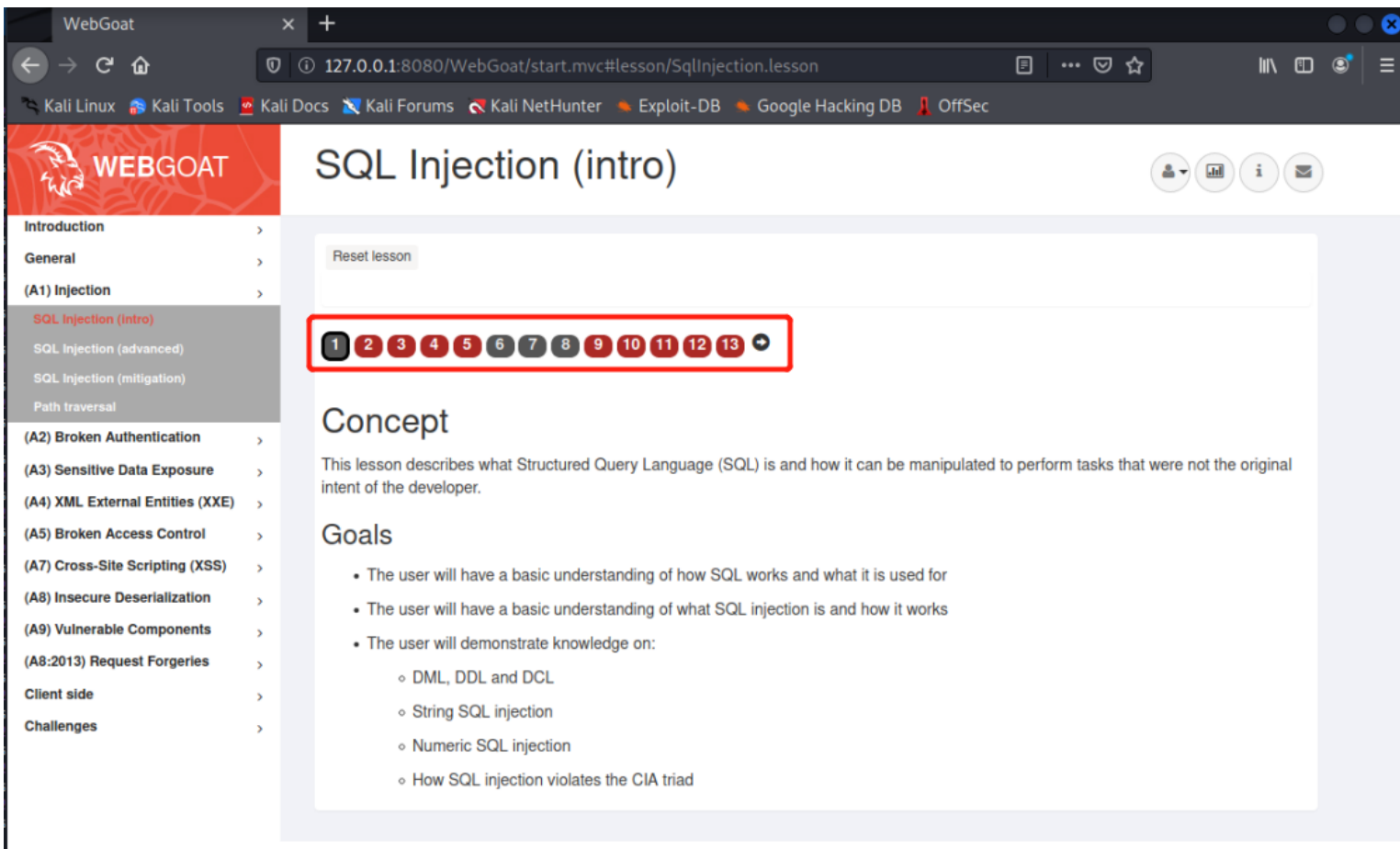
下图为webg在界面左边是目录，webgoat提供了不同类型WEB漏洞的学习，点击相应的章节会显示章节的小节





# Webgoat使用

点击小节标题进入到该小节中，每个小节有好几个关卡，如下图红框所示，灰色的部分是教学关卡，只需要阅读即可，红色部分是实操关卡，完成练习之后按钮会从红色变为绿色





## Webgoat使用

在每个实践关卡中，都会有任务说明（如下图红框），在箭头指示处填入答案

- Data Control Language (DCL)

Each of these command types can be used by attackers to compromise the confidentiality, integrity, and/or availability of a system. Proceed with the lesson to learn more about the SQL command types and how they relate to protections goals.

If you are still struggling with SQL and need more information or practice, you can visit <http://www.sqlcourse.com/> for free and interactive online training.

### It is your turn!

Look at the example table. Try to retrieve the department of the employee Bob Franco. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.

SQL query

SQL query

Submit



## Webgoat使用

倘若答案正确，会有相应的提示

### It is your turn!

Look at the example table. Try to retrieve the department of the employee Bob Franco. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.



SQL query

Submit

**You have succeeded!**

```
select * from employees where first_name='Bob';
```

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE
--------	------------	-----------	------------	--------	----------	-------

96134	Bob	Franco	Marketing	83700	LO9S2V	null
-------	-----	--------	-----------	-------	--------	------



# Webgoat演示

insecure login

\*Loopback: lo

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http.request.method==POST

No.	Time	Source	Destination	Protocol	Length	Info
428	4.030875920	127.0.0.1	127.0.0.1	HTTP	560	POST /WebGoat/start.mvc HTTP/1.1 (text/plain)

\r\n  
[Full request URI: http://127.0.0.1:8080/WebGoat/start.mvc]  
[HTTP request 2/10]  
[Prev request in frame: 3]  
[Response in frame: 433]  
[Next request in frame: 435]  
File Data: 50 bytes  
Line-based text data: text/plain (1 lines)  
{"username":"CaptainJack","password":"BlackPearl"}

01a0 2e 30 2e 30 2e 31 3a 38 30 38 30 2f 57 65 62 47 .0.0.1:8 080/WebG  
01b0 6f 61 74 2f 73 74 61 72 74 2e 6d 76 63 0d 0a 43 oat/star t.mvc ·C  
01c0 6f 6f 6b 69 65 3a 20 4a 53 45 53 53 49 4f 4e 49 ookie: J SESSIONI  
01d0 44 3d 58 73 5a 77 41 5f 50 75 7a 38 6b 69 4a 76 D-XsZwA\_ Puz8kiJv  
01e0 45 65 36 30 42 4d 47 6f 77 71 36 53 2d 62 64 64 Ee60BMGo wq6S-bdd  
01f0 65 6e 59 58 49 7a 6f 57 79 51 0d 0a 0d 0a 7b 22 enYXIzoW yQ...{"  
0200 75 73 65 72 6e 61 6d 65 22 3a 22 43 61 70 74 61 username ":"Capta  
0210 69 6e 4a 61 63 6b 22 2c 22 70 61 73 73 77 6f 72 inJack", "passwor  
0220 64 22 3a 22 42 6c 61 63 6b 50 65 61 72 6c 22 7d d":"Blac kPearl"}  
0230





# Webgoat演示

missing function level access control

The screenshot shows the Webgoat web application in a browser window. The address bar displays the URL `127.0.0.1:8080/WebGoat/start.mvc#lesson/MissingFunctionAC.lesson/1`. The left sidebar contains a list of lessons, with 'Missing Function Level Access Control' highlighted. The main content area is titled 'Relying on Obscurity' and 'Finding Hidden Items'. It explains that one could rely on HTML, CSS, or javascript to hide links that users don't normally access. It mentions a case where a network router tried to protect (hide) admin functionality with javascript in the UI, with a link to <https://www.wired.com/2009/10/routers-still-vulnerable>.

The 'Your Mission' section states: 'Find two invisible menu items in the menu below that are, or would be, of interest to an attacker/malicious user and submit the labels for those menu items (there are no links right now in the menus).' Below this is a menu structure with the following items:

- Account
  - My Profile
- Messages
  - Security
  - Logout

Below the menu, there are two input fields labeled 'Hidden Item 1' and 'Hidden Item 2', and a 'Submit' button.

The bottom of the screenshot shows the browser's developer tools. The 'Inspector' tab is active, showing the HTML structure of the page. The following HTML elements are highlighted with red boxes:

- `<a href="/users">Users</a>`
- `<a href="/config">Config</a>`

The 'Styles' panel on the right shows the CSS rules for the selected elements, including 'Flexbox' and 'Grid'.



# Webgoat演示

missing function level access control

Burp Suite Professional v2021.4.3 - Temporary Project - licensed to google

Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Custom

1 x 2 x ...

Send Cancel < >

**Request**

Pretty Raw In Actions

```
1 GET /WebGoat/users HTTP/1.1
2 Host: 127.0.0.1:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0)
  Gecko/20100101 Firefox/78.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Content-Type: application/json
9 Cookie: JSESSIONID=FD2182SPD9umcPC8WYrJuJboiW9nLv0kPUVrc
10 Upgrade-Insecure-Requests: 1
11
12
```

**Response**

Pretty Raw Render In Actions

```
1 HTTP/1.1 200 OK
2 Connection: close
3 X-XSS-Protection: 1; mode=block
4 X-Content-Type-Options: nosniff
5 X-Frame-Options: DENY
6 Content-Type: application/json
7 Date: Wed, 23 Feb 2022 08:30:15 GMT
8
9 {
10   {
11     "username": "ruciser",
12     "admin": false,
13     "userHash": "lb4JrqJpYinF61AdhkQv/vX++t9ebBipc0aaumEShRM="
14   }
15 }
```

INSF

Que

Body

Requ

Requ

Resp



## Webgoat演示

missing function level access control

推测是因为后端指定了application/json类型

```
@GetMapping(path = {"access-control/users"}, consumes = "application/json")
@ResponseBody
public ResponseEntity<List<DisplayUser>> userService() {
    return ResponseEntity.ok(userRepository.findAllUsers().stream().map(user -> new DisplayUser(user,
PASSWORD_SALT_SIMPLE)).collect(Collectors.toList()));
}
```



# Webgoat演示

authentication bypass

Burp Suite Professional v2021.4.3 - Temporary Project - licensed to google

Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options CustomScanner

1 x ...

Send Cancel < >

**Request**

Pretty Raw \n Actions

```
1 POST /WebGoat/auth-bypass/verify-account HTTP/1.1
2 Host: 127.0.0.1:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0)
  Gecko/20100101 Firefox/78.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 88
10 Origin: http://127.0.0.1:8080
11 Connection: close
12 Referer: http://127.0.0.1:8080/WebGoat/start.mvc
13 Cookie: JSESSIONID=EDe2J82SPDPumbcPC8WYrJuJboiW9nLvV0kpUVrc
14
15 secQuestion2=12&secQuestion3=123&jsEnabled=1&verifyMethod=
  SEC_QUESTIONS&user.Id=12300745
```

**Response**

Pretty Raw Render \n Actions

```
1 HTTP/1.1 200 OK
2 Connection: close
3 X-XSS-Protection: 1; mode=block
4 X-Content-Type-Options: nosniff
5 X-Frame-Options: DENY
6 Content-Type: application/json
7 Date: Wed, 23 Feb 2022 08:18:38 GMT
8
9 {
10   "lessonCompleted":true,
11   "feedback":"Congrats, you have successfully verified the accou
12   "output":null,
13   "assignment":"VerifyAccount",
14   "attemptWasMade":true
15 }
```

**INSPECT**

Query Par

Body Para

Request C

Request H

Response



# Webgoat演示

## authentication bypass

```
private HashMap<String, String> parseSecQuestions(HttpServletRequest req) {
    Map<String, String> userAnswers = new HashMap<>();
    List<String> paramNames = Collections.list(req.getParameterNames());
    for (String paramName : paramNames) {
        //String paramName = req.getParameterNames().nextElement();
        if (paramName.contains("secQuestion")) {
            userAnswers.put(paramName, req.getParameter(paramName));
        }
    }
    return (HashMap) userAnswers;
}

public boolean verifyAccount(Integer userId, HashMap<String,String> submittedQuestions ) {
    //short circuit if no questions are submitted
    if (submittedQuestions.entrySet().size() != secQuestionStore.get(verifyUserId).size()) {
        return false;
    }
    if (submittedQuestions.containsKey("secQuestion0") &&
        !submittedQuestions.get("secQuestion0").equals(secQuestionStore.get(verifyUserId).get("secQuestion0"))) {
        return false;
    }
    if (submittedQuestions.containsKey("secQuestion1") &&
        !submittedQuestions.get("seQuestion1").equals(secQuestionStore.get(verifyUserId).get("secQuestion1"))) {
        return false;
    }
    // else
    return true;
}
```

只要包含了secQuestion，就加入

是否有secQuestion0和secQuestion1，由于我们改成了secQuestion2和secQuestion3，就会直接跳过