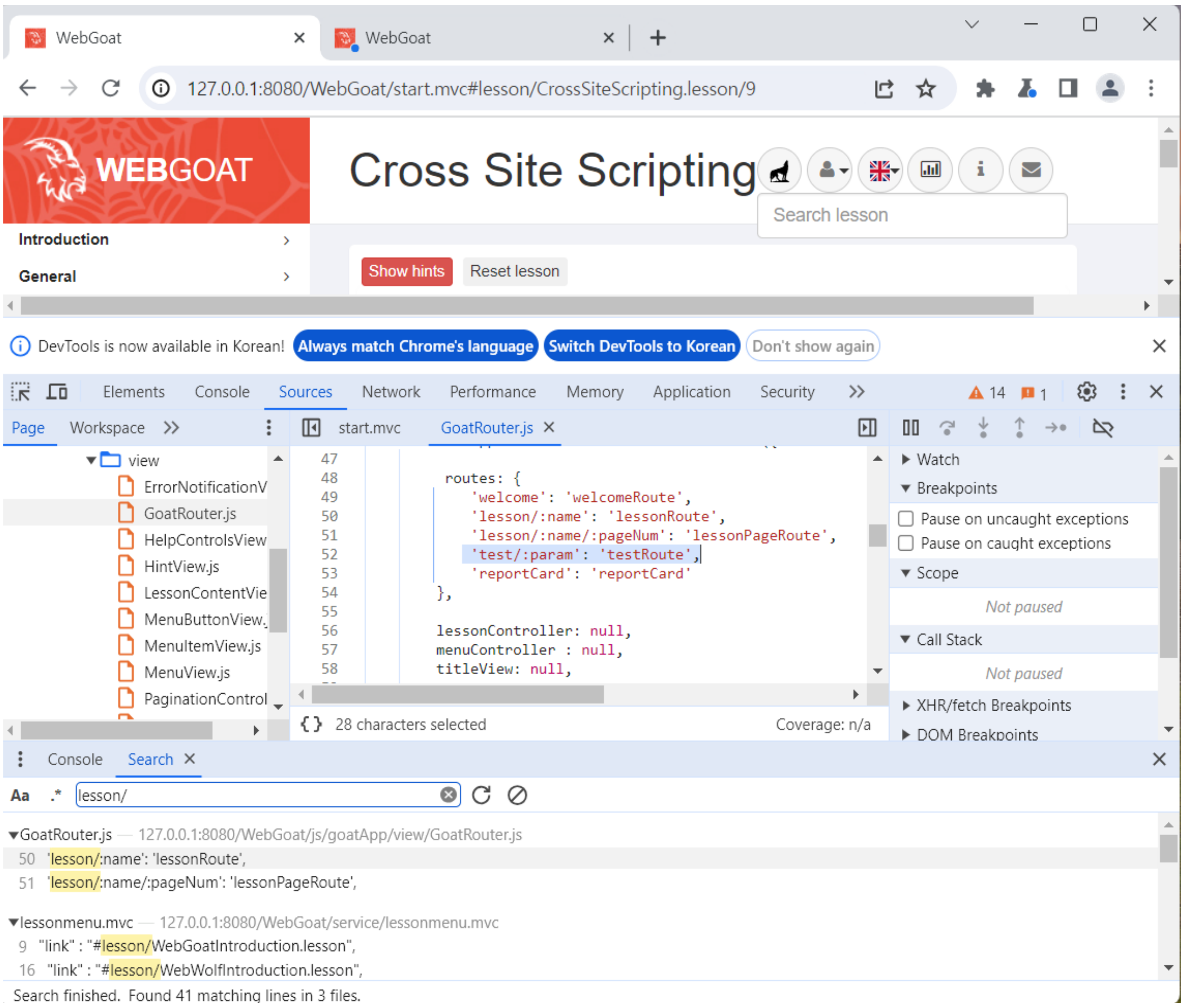
인터넷응용보안 9주차 과제

202121556 곽지현

DOM Based XSS (1)

개발자도구 검색창에서 lesson/ 검색 -> js 파일 열기

Base route가 start.mvc#lesson/ 이므로 테스트용 route는 start.mvc#test/ 임을 유추



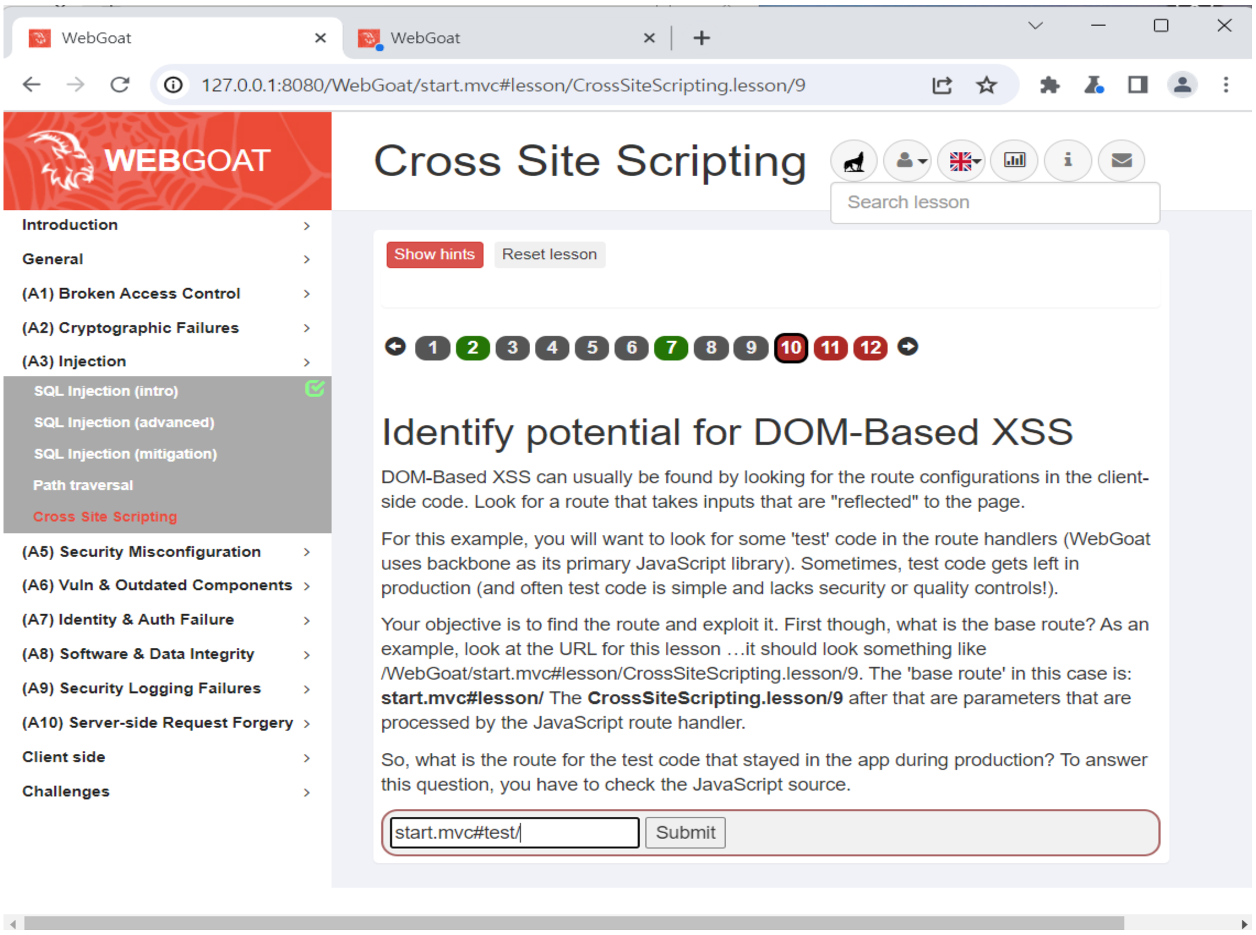
```
    routes: {
      'welcome': 'welcomeRoute',
      'lesson/:name': 'lessonRoute',
      'lesson/:name/:pageNum': 'lessonPageRoute',
      'test/:param': 'testRoute',
      'reportCard': 'reportCard'
   },

   lessonController: null,
   menuController : null,
   titleView: null,
```

```
▾GoatRouter.js — 127.0.0.1:8080/WebGoat/js/goatApp/view/GoatRouter.js
 50  'lesson/:name': 'lessonRoute',
 51  'lesson/:name/:pageNum': 'lessonPageRoute',

▾lessonmenu.mvc — 127.0.0.1:8080/WebGoat/service/lessonmenu.mvc
 9   "link" : "#lesson/WebGoatIntroduction.lesson",
 16  "link" : "#lesson/WebWolfIntroduction.lesson",
Search finished.  Found 41 matching lines in 3 files.
```

빈칸에 start.mvc#test/ 넣고 Submit 버튼 클릭


WebGoat
WebGoat
127.0.0.1:8080/WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/9
WEBGOAT
Search lesson
Introduction
General
(A1) Broken Access Control
(A2) Cryptographic Failures
(A3) Injection
SQL Injection (intro)
SQL Injection (advanced)
SQL Injection (mitigation)
Path traversal
Cross Site Scripting
(A5) Security Misconfiguration
(A6) Vuln & Outdated Components
(A7) Identity & Auth Failure
(A8) Software & Data Integrity
(A9) Security Logging Failures
(A10) Server-side Request Forgery
Client side
Challenges
Show hints
Reset lesson
1 2 3 4 5 6 7 8 9 10 11 12
Submit

Cross Site Scripting

Identify potential for DOM-Based XSS

DOM-Based XSS can usually be found by looking for the route configurations in the client-side code. Look for a route that takes inputs that are "reflected" to the page.

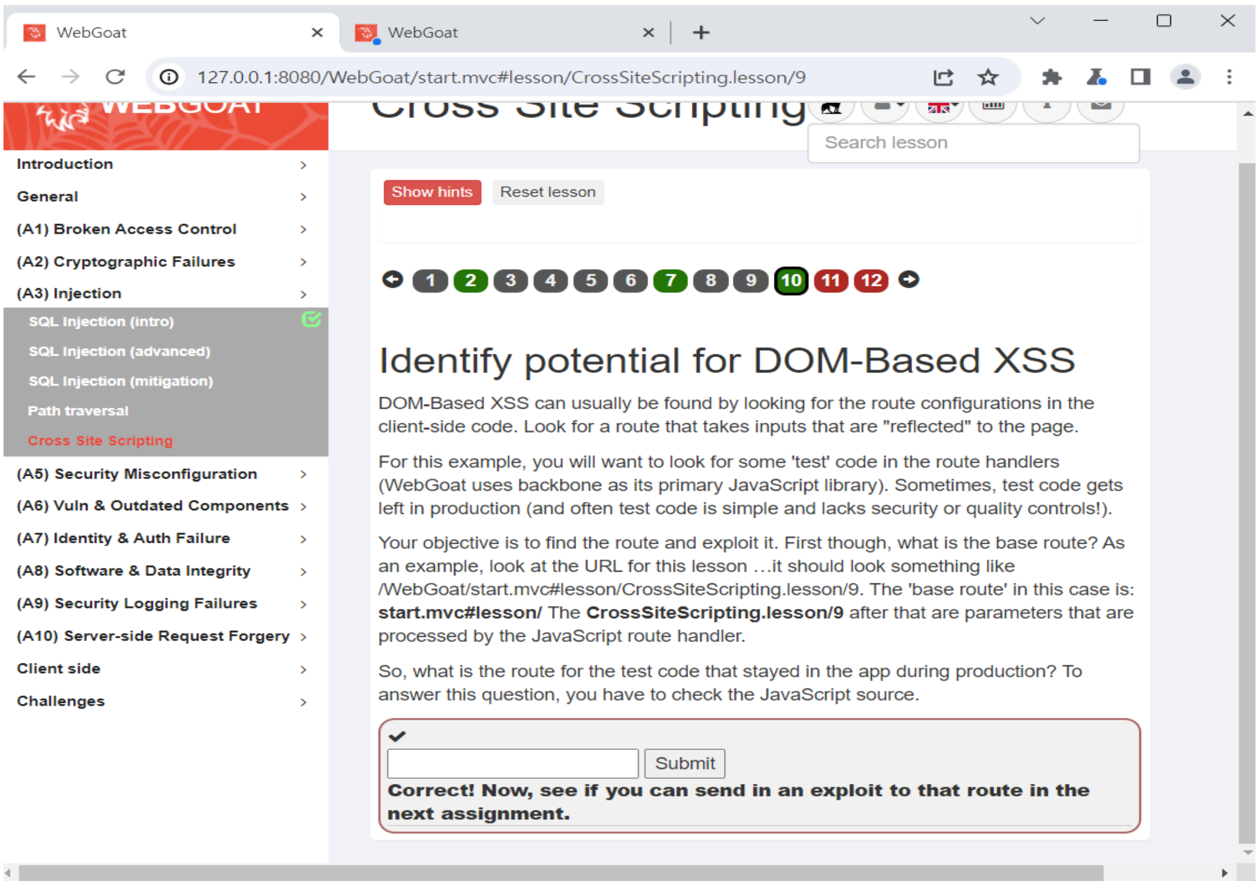
For this example, you will want to look for some 'test' code in the route handlers (WebGoat uses backbone as its primary JavaScript library). Sometimes, test code gets left in production (and often test code is simple and lacks security or quality controls!).

Your objective is to find the route and exploit it. First though, what is the base route? As an example, look at the URL for this lesson ...it should look something like /WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/9. The 'base route' in this case is: **start.mvc#lesson/** The **CrossSiteScripting.lesson/9** after that are parameters that are processed by the JavaScript route handler.

So, what is the route for the test code that stayed in the app during production? To answer this question, you have to check the JavaScript source.

start.mvc#test/

DOM Based XSS (1) 성공!


WebGoat
WebGoat
127.0.0.1:8080/WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/9
Search lesson
Introduction
General
(A1) Broken Access Control
(A2) Cryptographic Failures
(A3) Injection
SQL Injection (intro)
SQL Injection (advanced)
SQL Injection (mitigation)
Path traversal
Cross Site Scripting
(A5) Security Misconfiguration
(A6) Vuln & Outdated Components
(A7) Identity & Auth Failure
(A8) Software & Data Integrity
(A9) Security Logging Failures
(A10) Server-side Request Forgery
Client side
Challenges
Show hints
Reset lesson
1 2 3 4 5 6 7 8 9 10 11 12
Submit

Identify potential for DOM-Based XSS

DOM-Based XSS can usually be found by looking for the route configurations in the client-side code. Look for a route that takes inputs that are "reflected" to the page.

For this example, you will want to look for some 'test' code in the route handlers (WebGoat uses backbone as its primary JavaScript library). Sometimes, test code gets left in production (and often test code is simple and lacks security or quality controls!).

Your objective is to find the route and exploit it. First though, what is the base route? As an example, look at the URL for this lesson ...it should look something like /WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/9. The 'base route' in this case is: **start.mvc#lesson/** The **CrossSiteScripting.lesson/9** after that are parameters that are processed by the JavaScript route handler.

So, what is the route for the test code that stayed in the app during production? To answer this question, you have to check the JavaScript source.

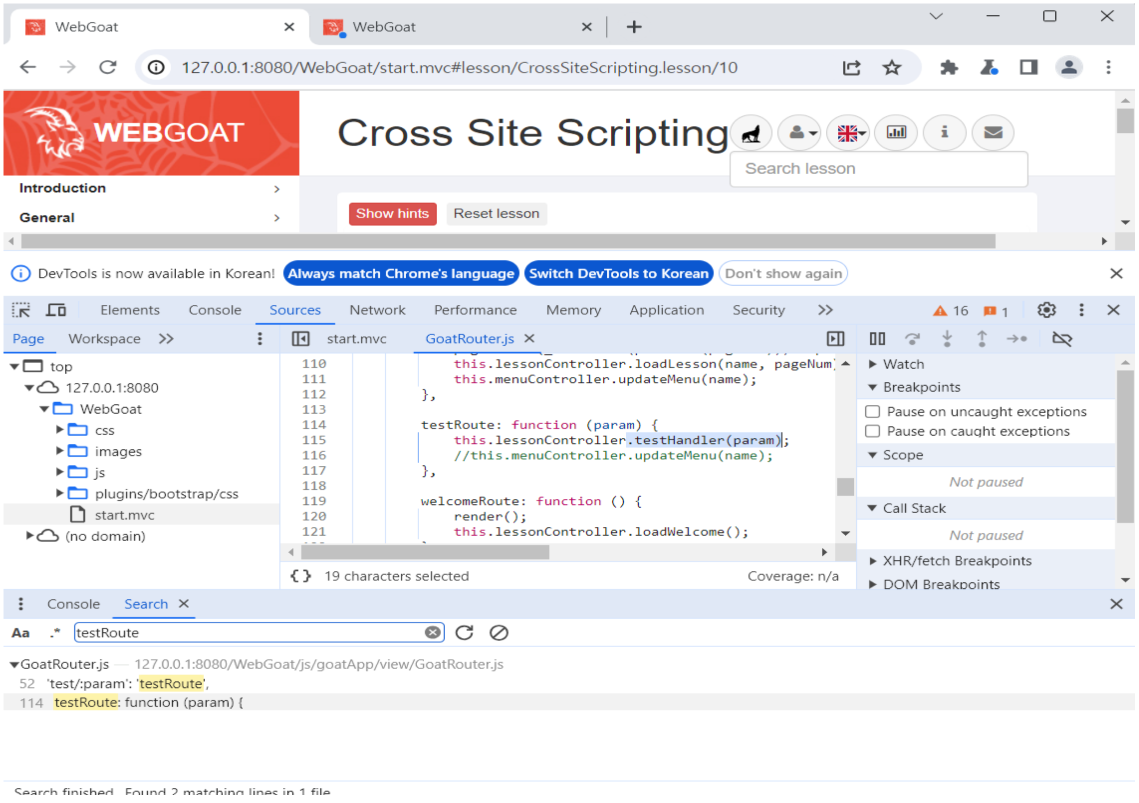
✓

Correct! Now, see if you can send in an exploit to that route in the next assignment.

DOM Based XSS (2)

개발자도구 검색창에서 testRoute 검색 -> js 파일 열기

testHandler 에서 파라미터를 처리하고 있는 것을 확인



```
        this.lessonController.loadLesson(name, pageNum
        this.menuController.updateMenu(name);
    },

    testRoute: function (param) {
        this.lessonController.testHandler(param);
        //this.menuController.updateMenu(name);
    },

    welcomeRoute: function () {
        render();
        this.lessonController.loadWelcome();
```

19 characters selected   Coverage: n/a

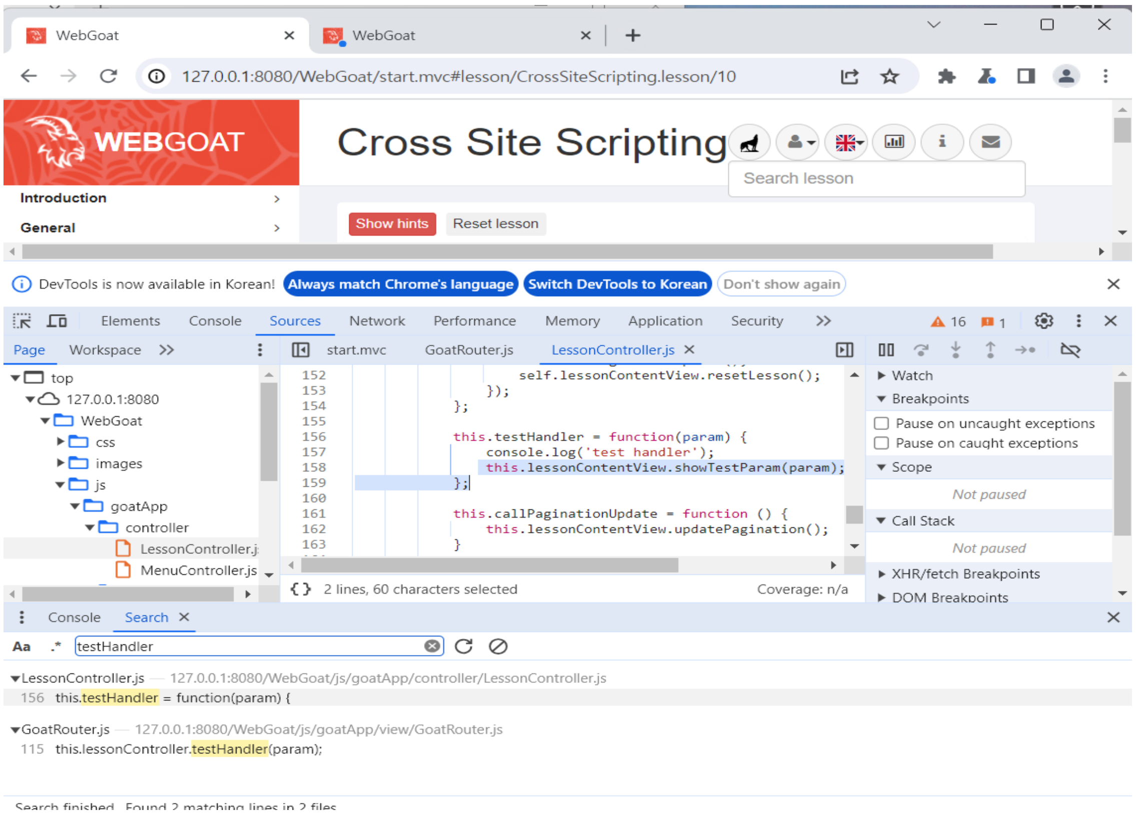
Search: testRoute

```
GoatRouter.js — 127.0.0.1:8080/WebGoat/js/goatApp/view/GoatRouter.js
 52  'test/:param': 'testRoute',
114  testRoute: function (param) {

Search finished. Found 2 matching lines in 1 file
```

다시 검색창에서 testHandler 검색 -> js 파일 열기

showTestParam() 에서 파라미터를 처리하고 있는 것을 확인



```
                self.lessonContentView.resetLesson();
            });
        };

        this.testHandler = function(param) {
            console.log('test handler');
            this.lessonContentView.showTestParam(param);
        };

        this.callPaginationUpdate = function () {
            this.lessonContentView.updatePagination();
        }
```

2 lines, 60 characters selected   Coverage: n/a

Search: testHandler

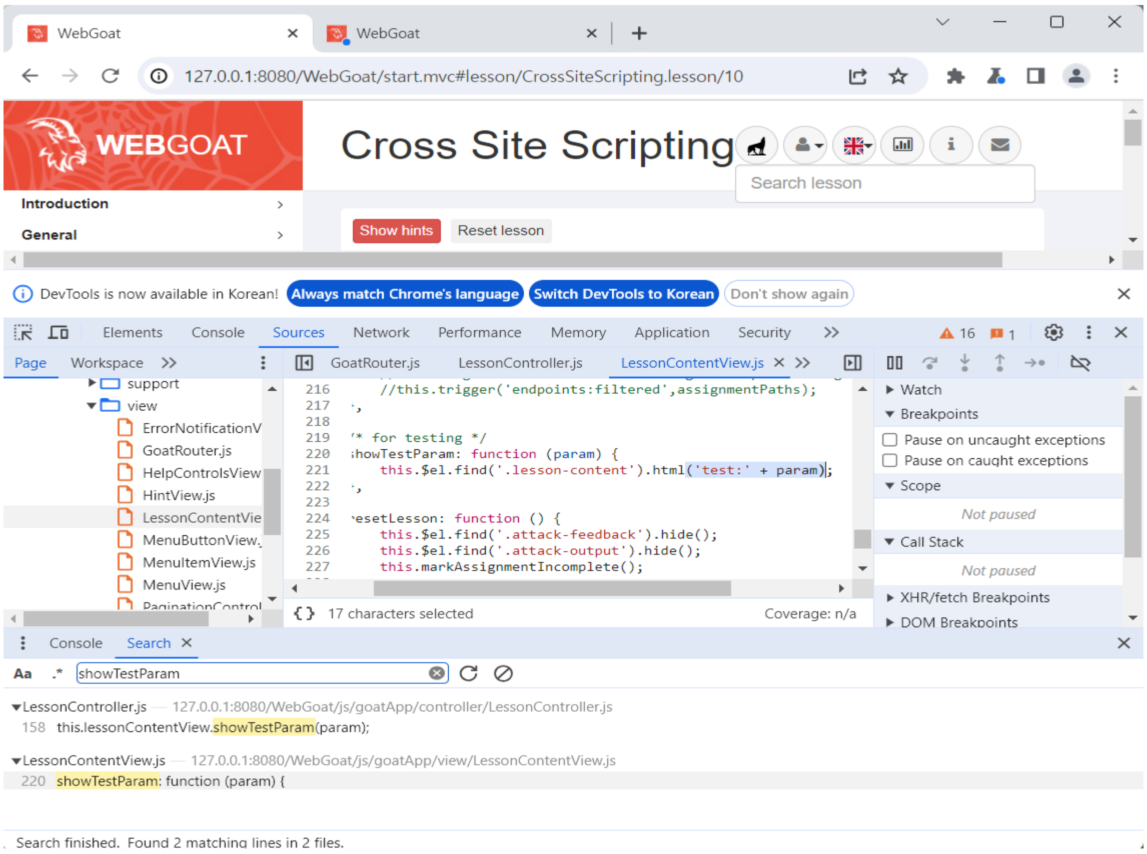
```
LessonController.js — 127.0.0.1:8080/WebGoat/js/goatApp/controller/LessonController.js
156  this.testHandler = function(param) {

GoatRouter.js — 127.0.0.1:8080/WebGoat/js/goatApp/view/GoatRouter.js
115  this.lessonController.testHandler(param);

Search finished. Found 2 matching lines in 2 files
```

다시 검색창에서 showTestParam 검색 -> js 파일 열기

showTestParam() 함수는 파라미터를 lesson-content 클래스를 통해 내장 HTML 코드로 Test 단어와 함께 추가한다는 것을 확인



127.0.0.1:8080/WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/10

Cross Site Scripting

```
    //this.trigger('endpoints:filtered',assignmentPaths);
},

/* for testing */
showTestParam: function (param) {
    this.$el.find('.lesson-content').html('test:' + param);
},

resetLesson: function () {
    this.$el.find('.attack-feedback').hide();
    this.$el.find('.attack-output').hide();
    this.markAssignmentIncomplete();
```

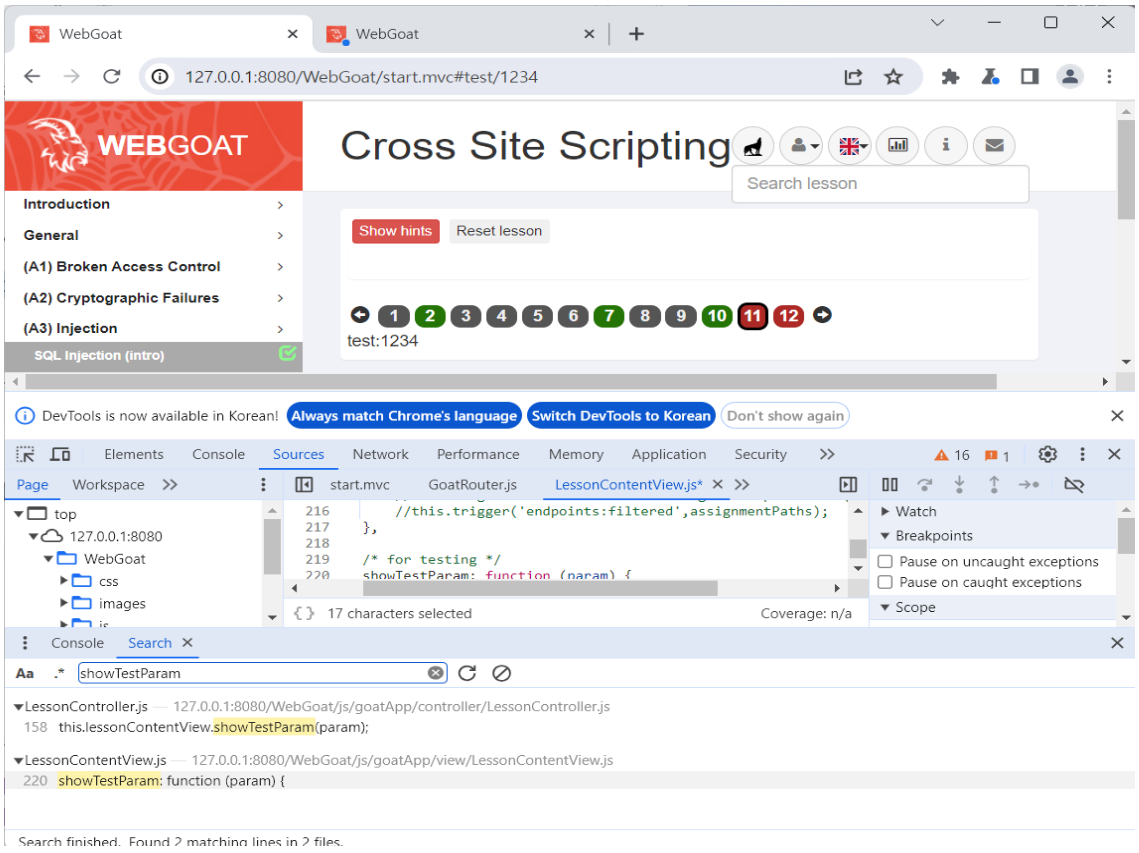
```
showTestParam

▾LessonController.js — 127.0.0.1:8080/WebGoat/js/goatApp/controller/LessonController.js
158  this.lessonContentView.showTestParam(param);

▾LessonContentView.js — 127.0.0.1:8080/WebGoat/js/goatApp/view/LessonContentView.js
220  showTestParam: function (param) {

Search finished.  Found 2 matching lines in 2 files.
```

주소창에 start.mvc#test/1234 넣고 수행 -> test: 1234가 화면에 표시



127.0.0.1:8080/WebGoat/start.mvc#test/1234

Cross Site Scripting

test:1234

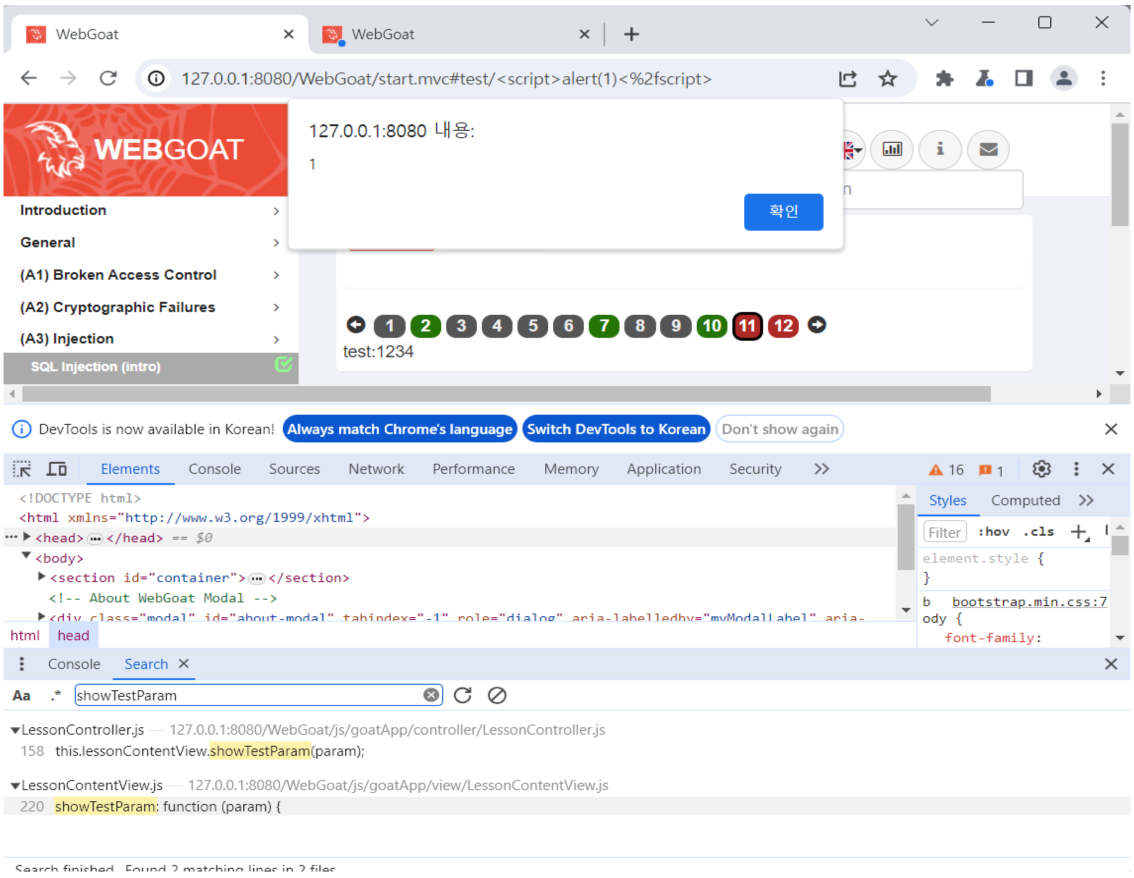
```
showTestParam

▾LessonController.js — 127.0.0.1:8080/WebGoat/js/goatApp/controller/LessonController.js
158  this.lessonContentView.showTestParam(param);

▾LessonContentView.js — 127.0.0.1:8080/WebGoat/js/goatApp/view/LessonContentView.js
220  showTestParam: function (param) {

Search finished.  Found 2 matching lines in 2 files.
```

주소창에 start.mvc#test/<script>alert(1)<%2fscript> 넣고 수행 -> Alert 창이 뜨는 것을 확인


127.0.0.1:8080/WebGoat/start.mvc#test/<script>alert(1)<%2fscript>

127.0.0.1:8080 내용:
1
확인

test:1234

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
▶<head>…</head> == $0
▼<body>
  ▶<section id="container">…</section>
  <!-- About WebGoat Modal -->
```

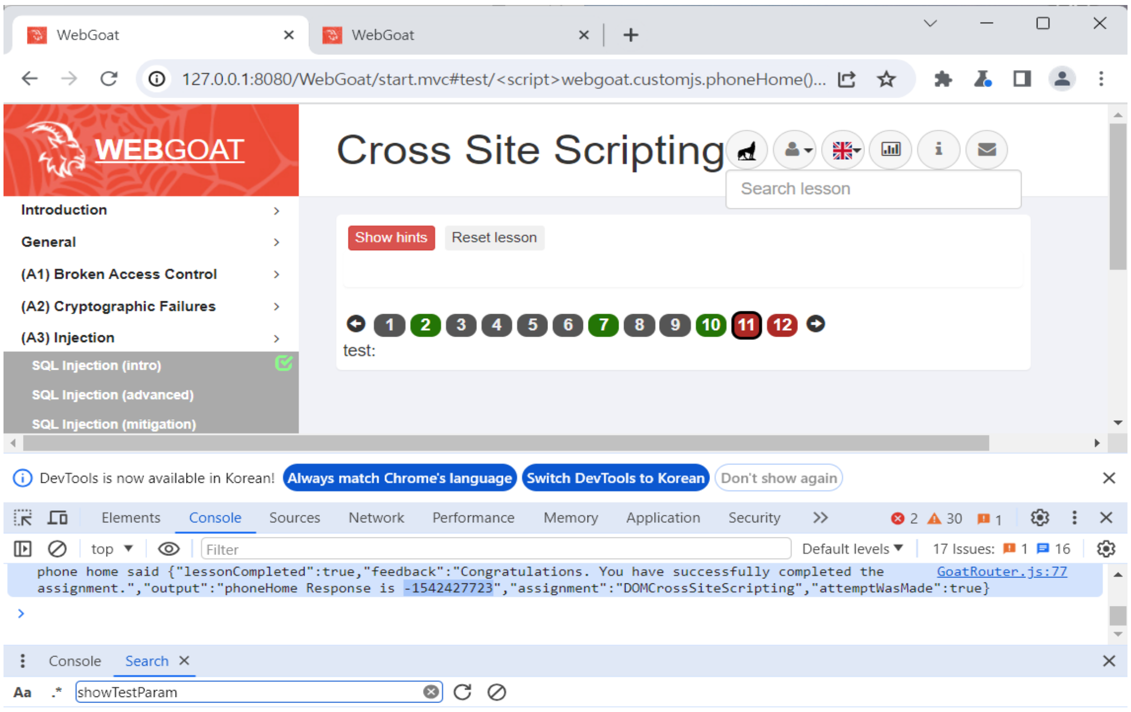
Search: showTestParam

```
▼LessonController.js — 127.0.0.1:8080/WebGoat/js/goatApp/controller/LessonController.js
  158  this.lessonContentView.showTestParam(param);

▼LessonContentView.js — 127.0.0.1:8080/WebGoat/js/goatApp/view/LessonContentView.js
  220  showTestParam: function (param) {

Search finished. Found 2 matching lines in 2 files.
```

주소창에 start.mvc#test/<script>webgoat.customjs.phoneHome()<%2fscript> 넣고 수행 -> 콘솔창에서 생성된 난수를 확인 (난수 : -1542427723)


127.0.0.1:8080/WebGoat/start.mvc#test/<script>webgoat.customjs.phoneHome()...

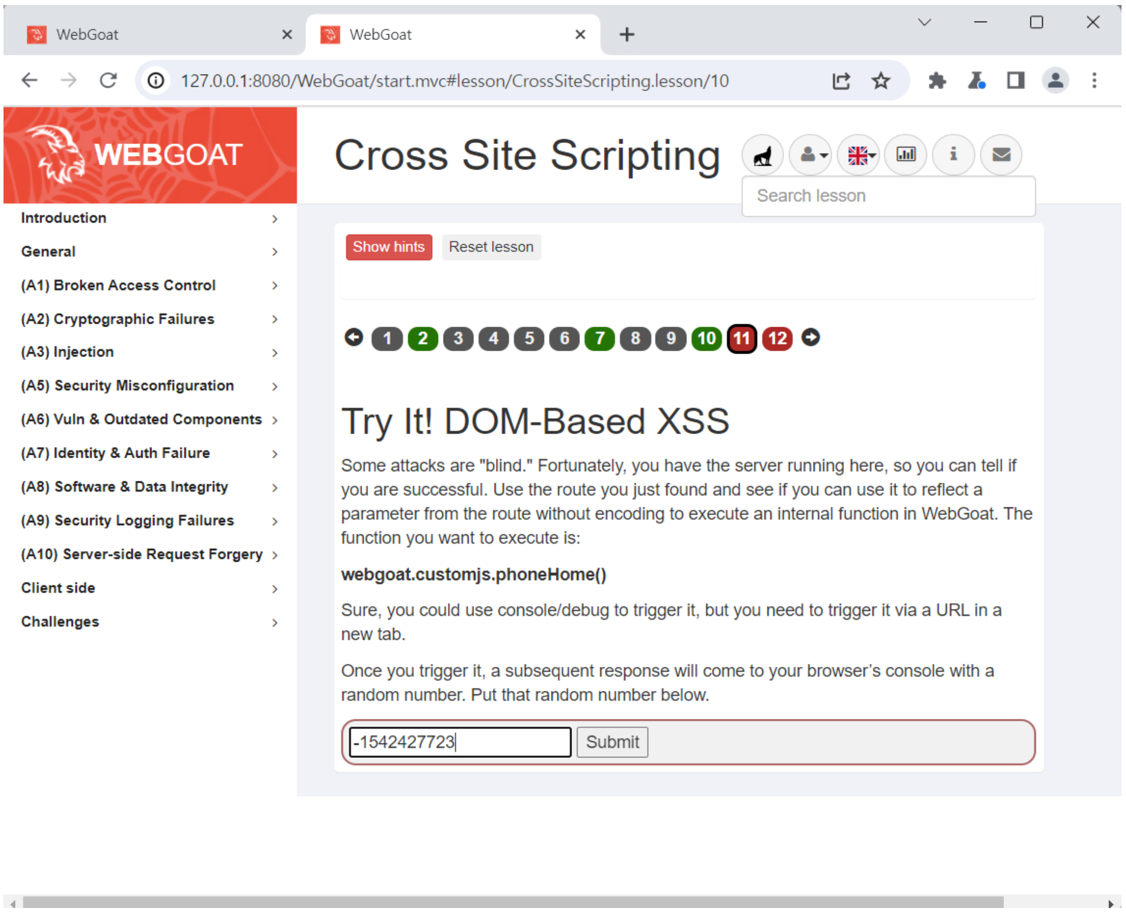
Cross Site Scripting

test:

```
phone home said {"lessonCompleted":true,"feedback":"Congratulations. You have successfully completed the      GoatRouter.js:77
assignment.","output":"phoneHome Response is -1542427723","assignment":"DOMCrossSiteScripting","attemptWasMade":true}
```

Search: showTestParam

뒤로 돌아가 빈칸에 생성된 난수 (-1542427723) 넣고 Submit 버튼 클릭


WebGoat
WebGoat
127.0.0.1:8080/WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/10
WEBGOAT
Introduction
General
(A1) Broken Access Control
(A2) Cryptographic Failures
(A3) Injection
(A5) Security Misconfiguration
(A6) Vuln & Outdated Components
(A7) Identity & Auth Failure
(A8) Software & Data Integrity
(A9) Security Logging Failures
(A10) Server-side Request Forgery
Client side
Challenges
Search lesson
Show hints
Reset lesson
1 2 3 4 5 6 7 8 9 10 11 12
Submit

Cross Site Scripting

Try It! DOM-Based XSS

Some attacks are "blind." Fortunately, you have the server running here, so you can tell if you are successful. Use the route you just found and see if you can use it to reflect a parameter from the route without encoding to execute an internal function in WebGoat. The function you want to execute is:

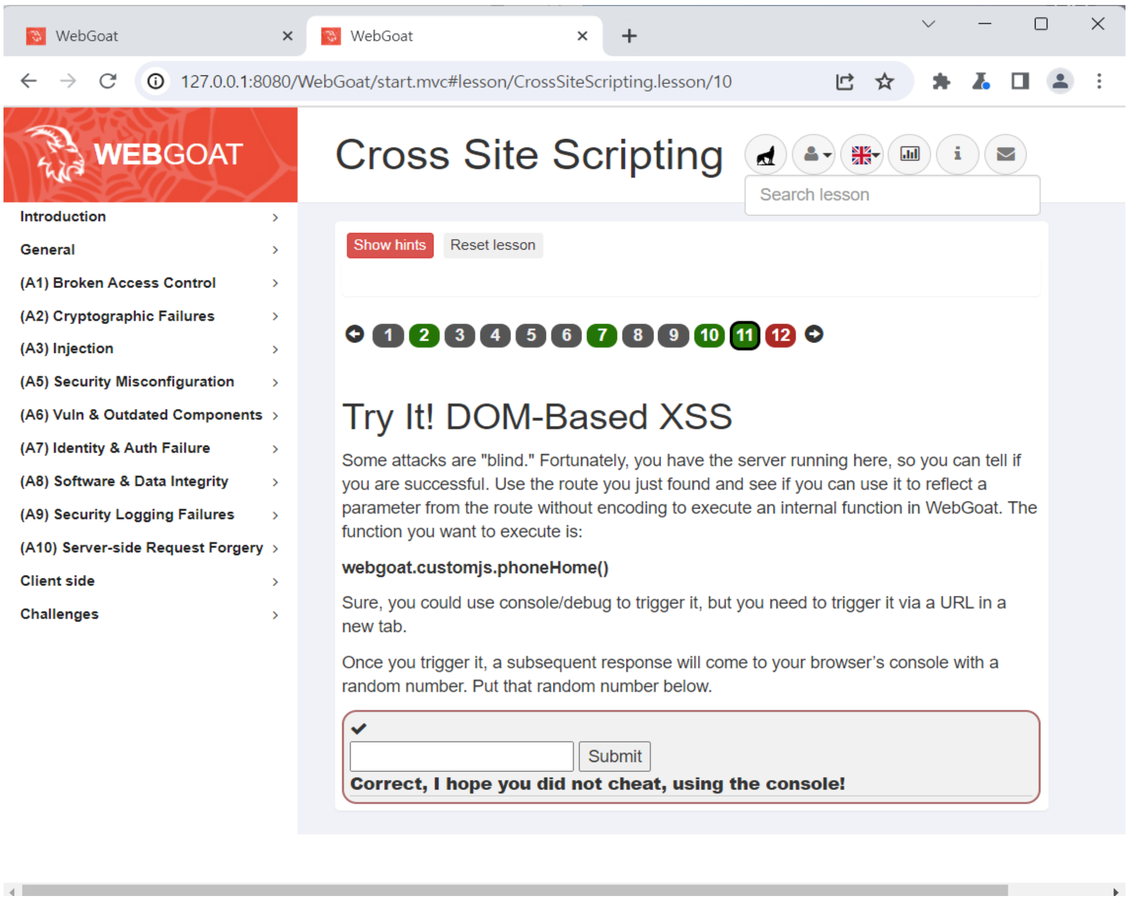
webgoat.customjs.phoneHome()

Sure, you could use console/debug to trigger it, but you need to trigger it via a URL in a new tab.

Once you trigger it, a subsequent response will come to your browser's console with a random number. Put that random number below.

-1542427723

DOM Based XSS (2) 성공!


WebGoat
WebGoat
127.0.0.1:8080/WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/10
WEBGOAT
Introduction
General
(A1) Broken Access Control
(A2) Cryptographic Failures
(A3) Injection
(A5) Security Misconfiguration
(A6) Vuln & Outdated Components
(A7) Identity & Auth Failure
(A8) Software & Data Integrity
(A9) Security Logging Failures
(A10) Server-side Request Forgery
Client side
Challenges
Search lesson
Show hints
Reset lesson
1 2 3 4 5 6 7 8 9 10 11 12
Submit

Cross Site Scripting

Try It! DOM-Based XSS

Some attacks are "blind." Fortunately, you have the server running here, so you can tell if you are successful. Use the route you just found and see if you can use it to reflect a parameter from the route without encoding to execute an internal function in WebGoat. The function you want to execute is:

webgoat.customjs.phoneHome()

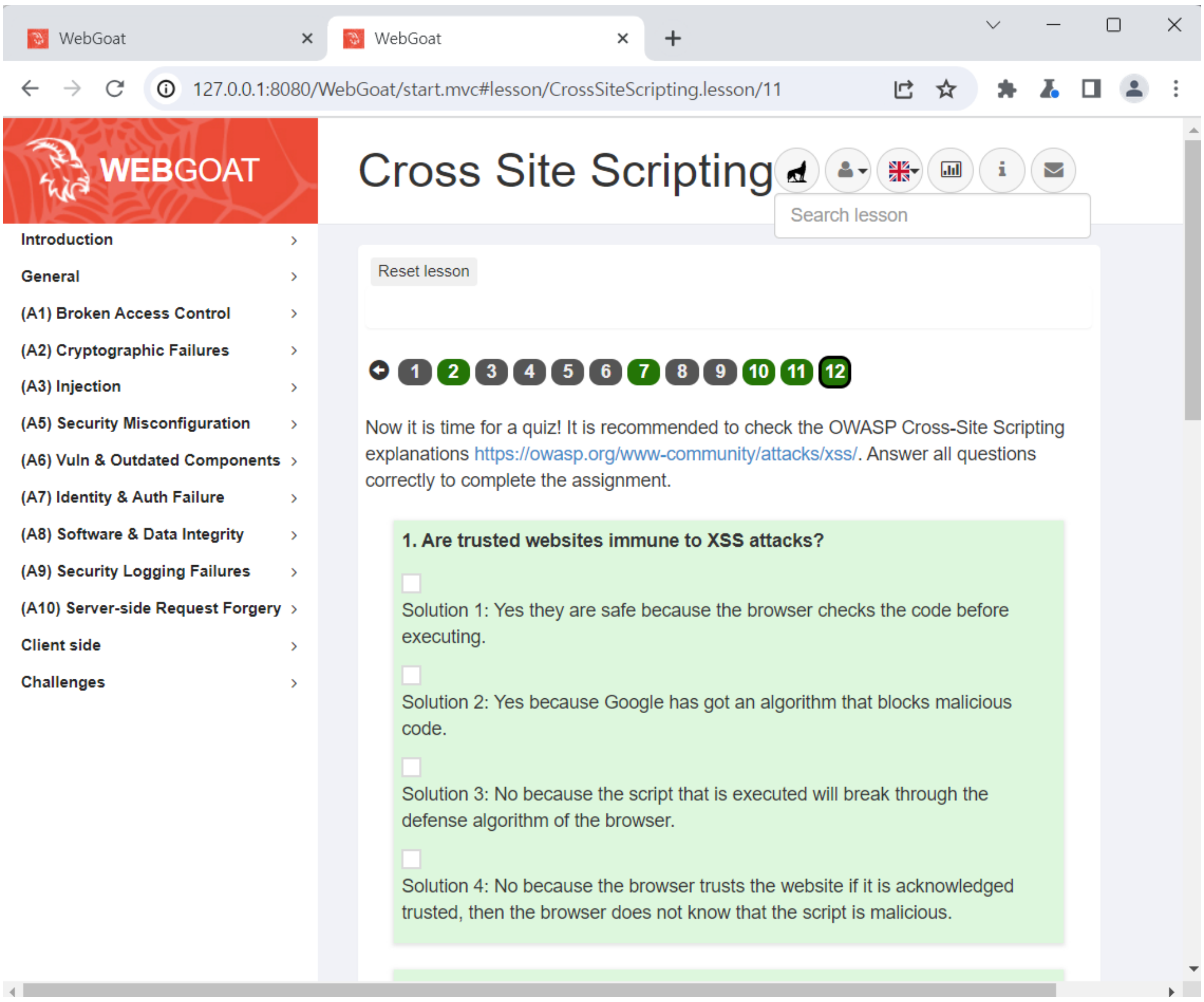
Sure, you could use console/debug to trigger it, but you need to trigger it via a URL in a new tab.

Once you trigger it, a subsequent response will come to your browser's console with a random number. Put that random number below.

✓

Correct, I hope you did not cheat, using the console!

모든 문제 성공!


WebGoat
WebGoat
127.0.0.1:8080/WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/11

WEBGOAT
Introduction
General
(A1) Broken Access Control
(A2) Cryptographic Failures
(A3) Injection
(A5) Security Misconfiguration
(A6) Vuln & Outdated Components
(A7) Identity & Auth Failure
(A8) Software & Data Integrity
(A9) Security Logging Failures
(A10) Server-side Request Forgery
Client side
Challenges
Search lesson
Reset lesson
1 2 3 4 5 6 7 8 9 10 11 12

Cross Site Scripting

Now it is time for a quiz! It is recommended to check the OWASP Cross-Site Scripting explanations https://owasp.org/www-community/attacks/xss/. Answer all questions correctly to complete the assignment.

1. Are trusted websites immune to XSS attacks?

☐ Solution 1: Yes they are safe because the browser checks the code before executing.

☐ Solution 2: Yes because Google has got an algorithm that blocks malicious code.

☐ Solution 3: No because the script that is executed will break through the defense algorithm of the browser.

☐ Solution 4: No because the browser trusts the website if it is acknowledged trusted, then the browser does not know that the script is malicious.