

REPORT



제 목 : 13주차 실습 보고서

과 목 명 : 시큐어코딩

담당교수 : 우사무엘 교수님

이 름 : 조 정 민

학 번 : 32164420



단국대학교
Dankook University

파일 업로드 취약점 공격 및 방어 실습

[파일 업로드 취약점 공격 실습]

우리의 실습환경에서는 업로드한 파일에 직접 접근 가능한 취약점과 업로드 되는 파일의 타입을 확인하지 않는 취약점을 내포하고 있다고 가정한다. 이럴 경우, 파일 업로드 취약점을 이용한 공격에 매우 취약하다.

업로드된 파일에 직접 접근이 가능한지 확인 후 웹shell을 업로드해 실행 가능한지 체크한다.

업로드 파일 직접 접근 가능 여부 확인

안전한 소프트웨어를 만들기 위한 노력 (주)오픈이지

새 글 쓰기

제목

파일 업로드 테스트

내용

파일 업로드 테스트

파일

파일 선택

1.jpg

* 임의의 파일명이 변경될 수 있습니다.

재작성

확인

[테스트]님 환영합니다.
로그아웃 정보수정

Copyright (C) (주)오픈이지(http://openeg.co.kr), 2018

안전한 소프트웨어를 만들기 위한 노력 (주)오픈이지

파일 업로드 테스트

작성자	조회수	추천수	작성일
테스트	0	0	2021-05-30

내용

첨부파일: 1.jpg

파일 업로드 테스트

댓글

댓글 쓰기

확인

삭제

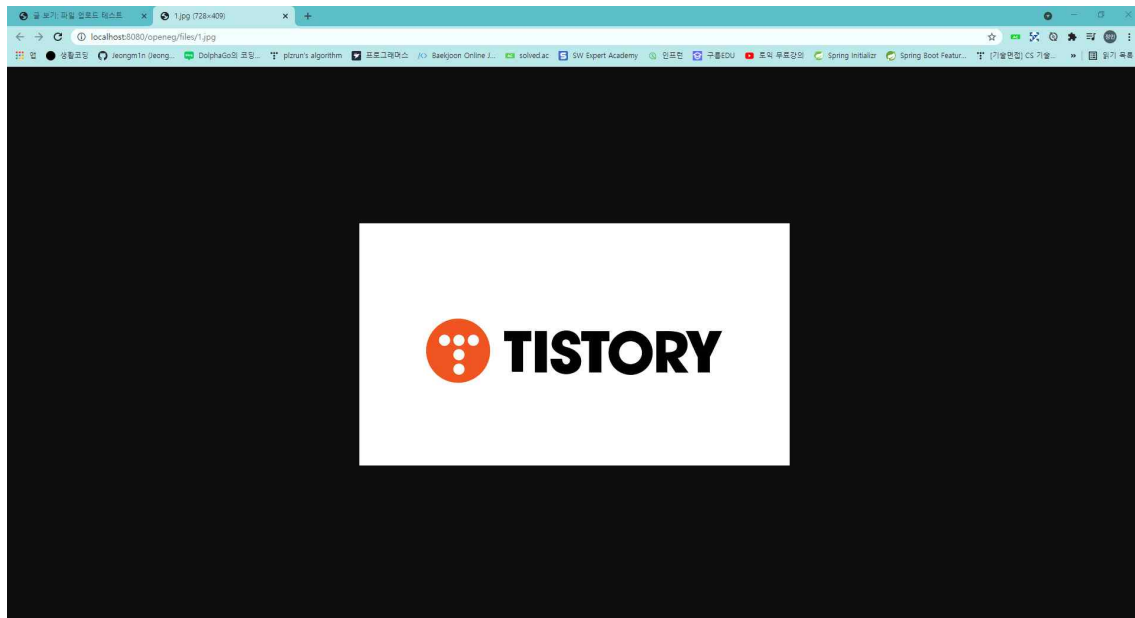
수정

목록

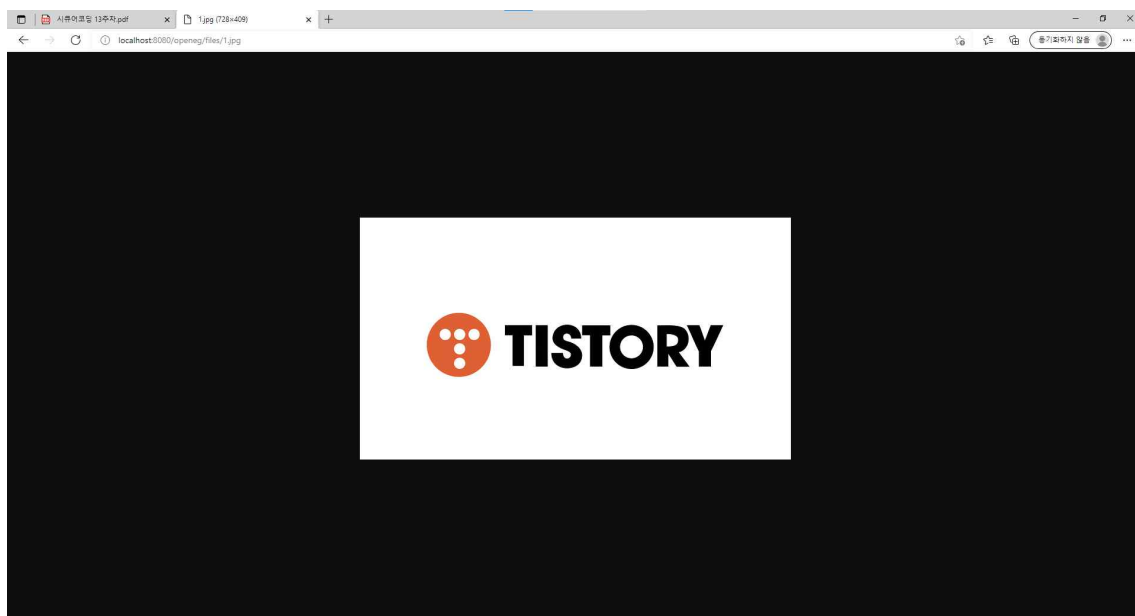
[테스트]님 환영합니다.
로그아웃 정보수정

Copyright (C) (주)오픈이지(http://openeg.co.kr), 2018

<openeg 게시글에서 첨부파일 내용>



<openeg 게시글 첨부파일 url을 이용하여 다른 웹브라우저에서 접근한 내용>



우리가 실습에 사용하고 있는 웹서버에서는 저장된 파일에 직접 접근할 수 있는 취약점을 가지고 있다.

웹셀을 업로드하여 실행 가능 여부 확인

안전한 소프트웨어를 만들기 위한 노력 (주)오픈이지

새 글 쓰기

제목	파일 업로드 테스트2
내용	<div style="border: 1px solid black; height: 200px; width: 100%;"></div>
파일	<div style="display: flex; align-items: center;"> <div style="border: 1px solid black; padding: 2px 5px; margin-right: 5px;">파일 선택</div> pwnshell.jsp </div> <p style="font-size: small; color: gray;">* 임의로 파일명이 변경될 수 있습니다.</p>

재작성

확인

[테스트]님 환영합니다.
로그아웃 정보수정

안전한 소프트웨어를 만들기 위한 노력 (주)오픈이지

[테스트]님 환영합니다.
로그아웃 정보수정

작성자	조회수	추천수	작성일
테스트	0	0	2021-05-30

내용

첨부파일: pwnshell.jsp

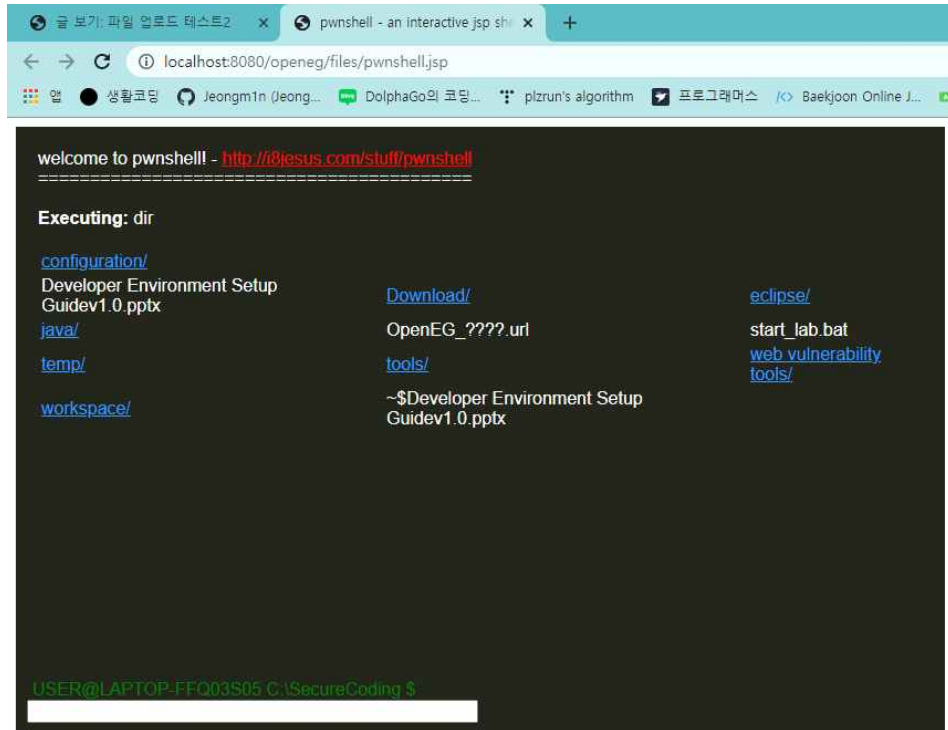
파일 업로드 테스트2

댓글

댓글 쓰기

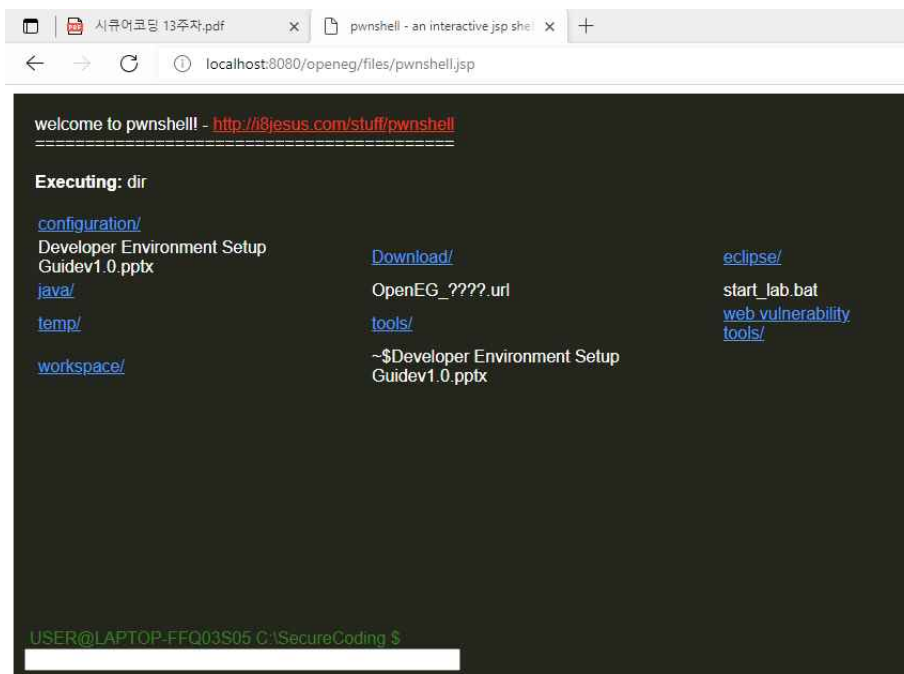
해당 웹shell이 악의적인 코드일 경우, 이 웹shell이 동작하는 서버에서 공격자는 다양한 행위 수행이 가능하다. 웹shell 파일을 클릭한 후, 웹서버에 있는 디렉터리나 파일을 확인하는 작업을 수행해보자.

<openeg 게시글에서 첨부파일 내용>



웹쉘이 게시판에 게시되어 디렉터리와 파일 구조 확인이 가능하다.

<openeg 게시글 첨부파일 url을 이용하여 다른 웹브라우저에서 접근한 내용>



이처럼 웹쉘에 직접 접근 가능하고, 명령 또한 웹서버에서 수행됨을 알 수 있다. 이와 같이 웹서버에 게시글을 업로드할 때 악의적인 파일인지 확인 절차가 없다면 웹쉘과 같은 악성 코드가 게시판에 업로드되어 공격자의 공격에 적극적으로 활용될 수 있다.

[파일 업로드 취약점 방어 실습]

업로드 되는 파일의 타입을 제한하고 외부에서 직접 접근이 불가능한 경로에 파일을 저장하여 앞서 수행한 공격을 차단할 수 있다.

- <http://openeg.co.kr> -> BLOG -> BoardController(파일업로드취약점제거)

- 해당 파일의 내용을 기존 BoardController에 적용

<직접 접근 불가능한 위치로 구현하는 코드>

<파일 사이즈 제한 코드>

<파일의 확장자 확인하여 업로드 차단>

<저장 위치 확인 및 파일을 얻어올 수 있도록 코드 작성>

<get_image.do 명령 수행 파일 작성>

```
@RequestMapping(value="/write.do", method=RequestMethod.POST)
public String boardWriteProc(@ModelAttribute("BoardModel") BoardModel boardModel, MultipartHttpServletRequest request, HttpSession session){
    //파일저장 위치
    String uploadPath = session.getServletContext().getRealPath("/")
        + "WEB-INF/files/";
    System.out.println("uploadPath: "+uploadPath);
    MultipartFile file = request.getFile("file");
    //업로드 되는 파일 사이즈 제한
    if (file != null && !"".equals(file.getOriginalFilename())
        && file.getSize() < 1024000 && file.getContentType().contains("image")){
        //업로드 파일명
        String fileName = file.getOriginalFilename();
        if (fileName.toLowerCase().endsWith(".jpg")){
            //저장할 파일명을 랜덤하게 생성하여 사용한다.
            String savedFileName = UUID.randomUUID().toString();
            File uploadFile = new File(uploadPath+savedFileName);
            try {
                file.transferTo(uploadFile); //실제 파일이 저장되는 라인
            } catch (Exception e) {
                System.out.println("upload error");
            }
            boardModel.setFileName(fileName);
            boardModel.setSavedFileName(savedFileName);
        }
    }

    String content = boardModel.getContent().replaceAll("\r\n", "<br />");
    boardModel.setContent(content);

    service.writeArticle(boardModel);

    return "redirect:list.do";
}
```

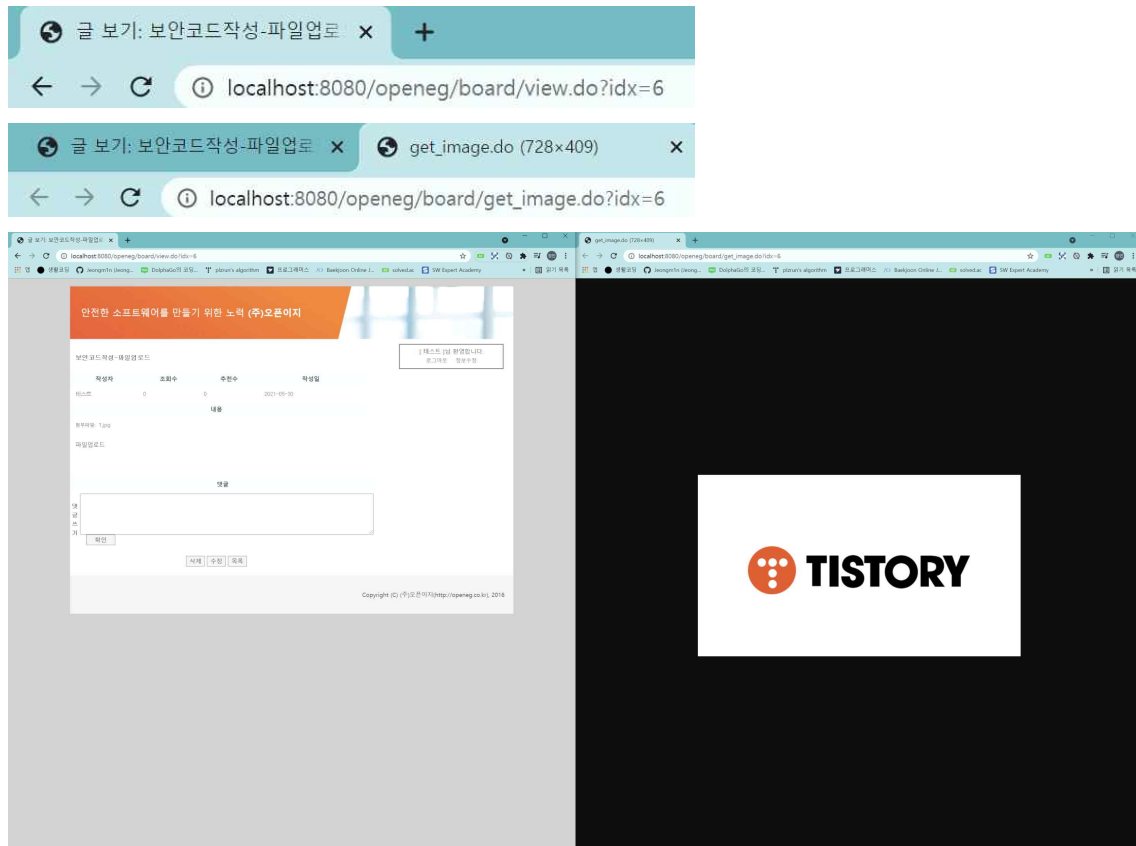
- openeg/WebContent/WEB-INF/board/view.jsp 수정

<get_image.do 명령 수행 코드 입력>

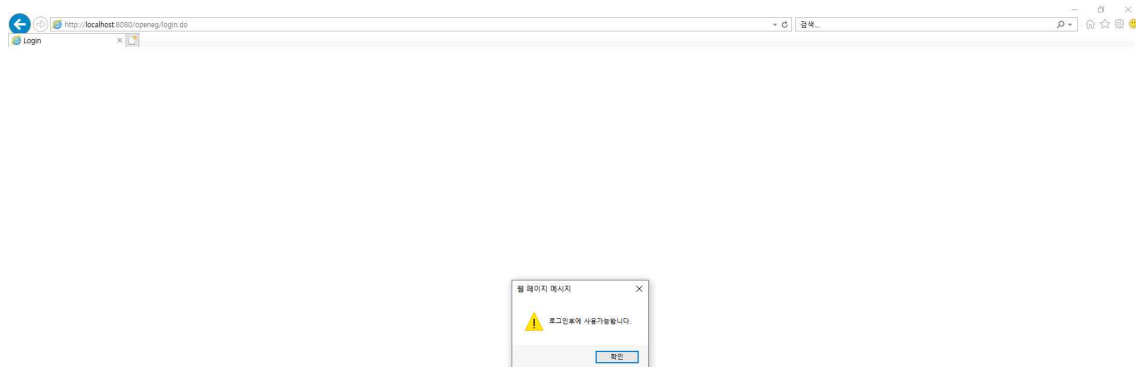
```
<tr>
    <td colspan="4" align="left">
        <span class="date">첨부파일:&nbsp;  
        <a href="get_image.do?idx=${board.idx}"
            target="_blank">${board.fileName}</a></span></td>
</tr>
```

방어코드 완성 후 직접 접근 가능한지 여부를 확인한다. 변경된 파일들의 수정사항을 저장하고 서버를 재가동한다.

<openeg 게시글에서 첨부파일 내용>



<openeg 게시글 첨부파일 url을 이용하여 다른 웹브라우저에서 접근한 내용>



이처럼 직접 접근이 불가능한 방어를 완벽하게 구현했다.

안전한 소프트웨어를 만들기 위한 노력 (주)오픈이지

보안코드작성-파일업로드2

작성자	조회수	추천수	작성일
테스트	0	0	2021-05-30

내용

웹셀 파일 업로드

댓글

댓글
쓰기

[테스트]님 환영합니다.
로그아웃 정보수정

Copyright (C) (주)오픈이지(<http://openeg.co.kr>), 2016

사전에 필터링을 하거나 직접 접근할 수 없는 위치에 저장하는 코드를 작성하여 파일 업로드 취약점을 방어할 수 있다.

파라미터 조작 공격 및 방어 실습

우리 실습 환경에서는 사용자별 접근 권한 체크가 적절하게 적용되어 있지 않아 파라미터 조작을 통해 인가되지 않은 작업 수행이 가능하다.

[파라미터 조작 공격 실습]

실습에 앞서 프록시 서버를 이용하고 파로스를 실행한다. 파로스를 통해 관리자가 서버에 전송하는 파라미터를 확인하고 해당 파라미터를 조작할 수 있다.

수동 프록시 설정

이더넷 또는 Wi-Fi 연결에 프록시 서버를 사용합니다. 이 설정은 VPN 연결에 적용되지 않습니다.

프록시 서버 사용



컴

주소

127.0.0.1

포트

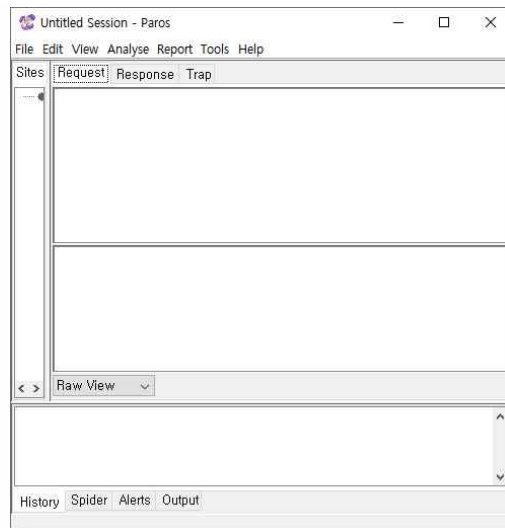
8081

다음 항목으로 시작하는 주소를 제외하고 프록시 서버를 사용합니다. 여러 항목은 세미콜론(;)으로 구분합니다.

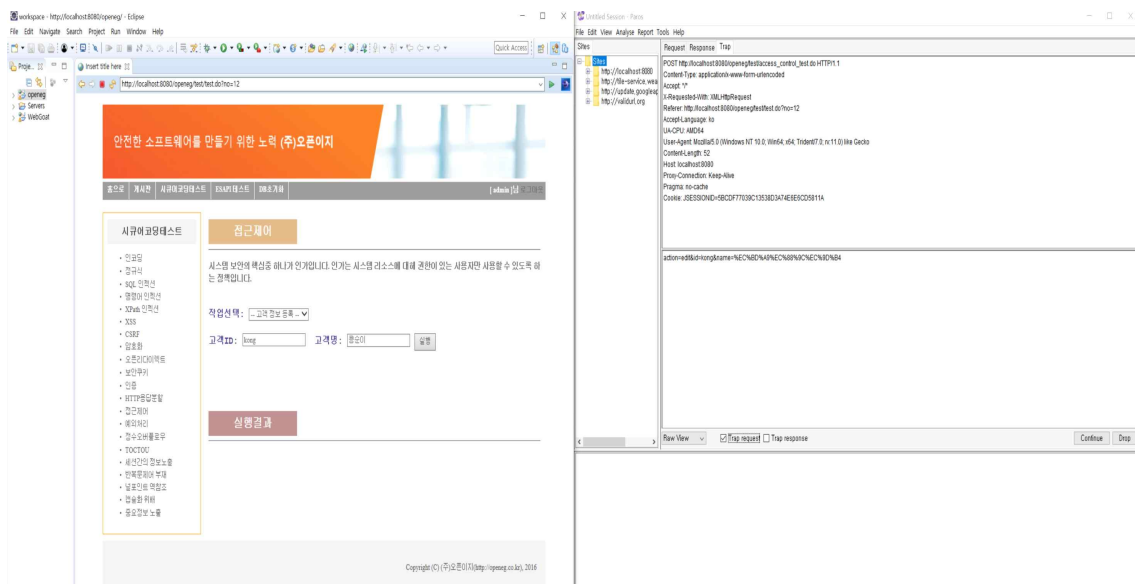
127.0.0.1:16105;127.0.0.1:16106;127.0.0.1:
21300;<-loopback>

☐ 로컬(인트라넷) 주소에 프록시 서버 사용 안 함

저장



- admin 계정
- 고객 정보 등록(action=edit&id=kong&name=%EC%BD%A9%EC%88%9C%EC%9D%B4)



- kong(콩순이) 계정 등록 완료

- kong(콩순이) 계정 로그인

안전한 소프트웨어를 만들기 위한 노력 (주)오픈이지

홈으로	게시판	시큐어코딩테스트	ESAPI 테스트	DB초기화	[kong] 님 로그아웃
-----	-----	----------	-----------	-------	-----------------

소프트웨어 보안은 보안소프트웨어가 아닙니다.

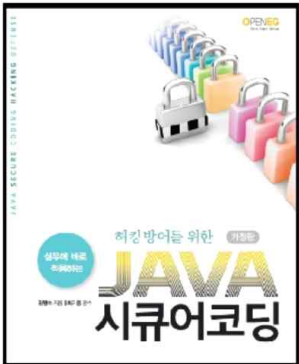
소프트웨어 보안을 유지한다는 것은 암호화화 같은 다양한 보안기능의 적용을 위함 하는 것보다 소프트웨어 라이프 사이클 전반에 걸쳐 여러가지 안전한 소프트웨어 개발의 모범사례를 적용하는 것을 의미합니다.

보안문제는 특정 보안기능보다 안전한 시스템을 구성하는 표준의 문제로 인해 발생할 수 있습니다.

그래서 소프트웨어 보안은 전체 개발단계의 라이프 사이클 접근 방식의 일부가 되어야 하는 중요한 이유입니다.

[콩순이] 님 환영합니다.

로그아웃 정보수정

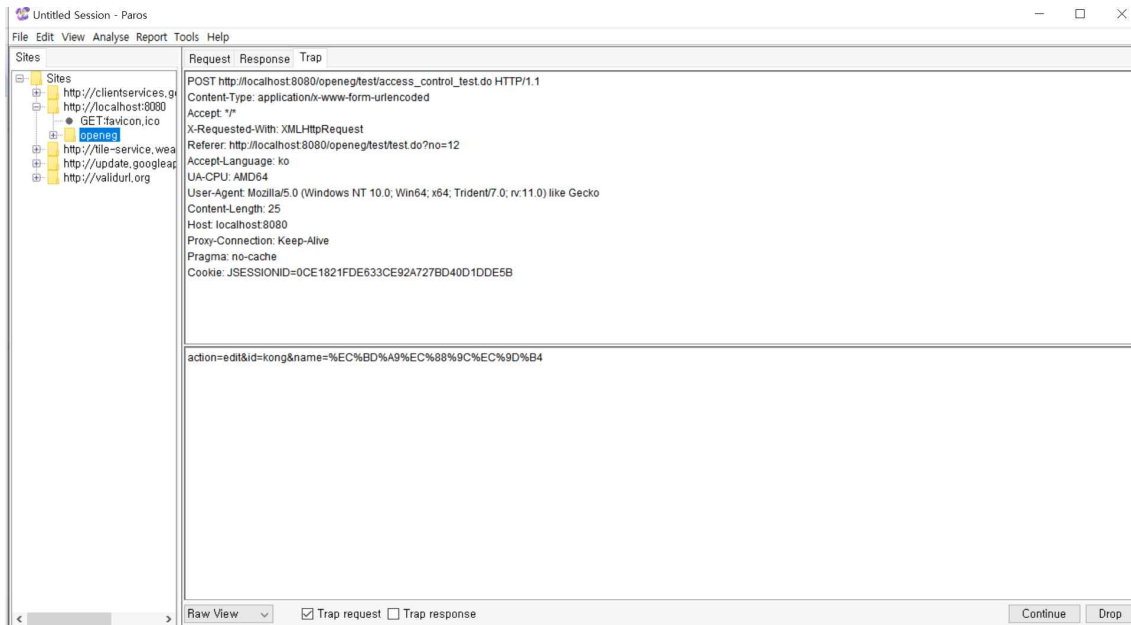


- admin 계정
- kong(공순이) 정보 삭제

- test 계정
- test 계정에서는 정보조회 밖에 하지 못함
- trap request를 활성화하고 정보조회 실행

<파라미터를 관리자가 생성한 파라미터로 변경>

- trap request 해제 후 continue



- 행결과(정보조회에서 kong(공순이) 계정 등록 완료)

안전한 소프트웨어를 만들기 위한 노력 (주)오픈이지

홈으로
게시판
시큐어코딩테스트
ESAPI 테스트
DB초기화
[test]님 로그아웃

시큐어코딩테스트

- 인코딩
- 정규식
- SQL 인젝션
- 명령어 인젝션
- XPath 인젝션
- XSS
- CSRF
- 암호화
- 오픈리다이렉트
- 보안쿠키
- 인증
- HTTP응답분할
- 접근제어
- 예외처리
- 정수오버플로우
- TOCTOU
- 세션간의 정보노출
- 반복문제어 부재
- 널포인트 역참조
- 캡슐화 위배
- 중요정보 노출

접근제어

시스템 보안의 핵심중 하나가 인가입니다. 인가는 시스템 리소스에 대해 권한이 있는 사용자만 사용할 수 있도록 하는 정책입니다.

작업선택 : -- 정보 조회 --

고객ID : test 연락처 : 실행

실행결과

kong 사용자 등록을 완료하였습니다.

사용자ID: kong
고객명: 공순이
전화번호:
가입일자: 2021-05-30

Copyright (C) (주)오픈이지(http://openeg.co.kr), 2016

- kong(콩순이) 계정 로그인 화면

안전한 소프트웨어를 만들기 위한 노력 (주)오픈이지

홈으로

게시판

시큐어코딩테스트

ESAPI 테스트

DB초기화

[kong]님 로그인

소프트웨어 보안은 보안소프트웨어가 아닙니다.

소프트웨어 보안을 유지한다는 것은 암호화화 같은 다양한 보안기능의 적용을 위함 하는 것보다 소프트웨어 라이프 사이클 전반에 걸쳐 여러가지 안전한 소프트웨어 개발의 모범사례를 적용하는 것을 의미합니다.

보안문제는 특정 보안기능보다 안전한 시스템을 구성하는 표준의 문제로 인해 발생할 수 있습니다.

그래서 소프트웨어 보안은 전체 개발단계의 라이프 사이클 접근 방식의 일부가 되어야 하는 중요한 이유입니다.

[콩순이]님 환영합니다.
로그아웃 정보수정

Copyright (C) (주)오픈이지(<http://openeg.co.kr>), 2016

이처럼, 관리자의 권한을 남용할 수 있는 매우 심각한 파라미터 조작 취약점이 있음을 확인하는 실습을 진행했다.

[파라미터 조작 방어 실습]

파라미터 조작 취약점을 이용한 공격을 차단한다. 인가정책을 사용하여 공격을 차단할 수 있다.

<TestController 코드 수정>

- edit 파라미터를 전달받았다면 사용자의 계정을 확인하는 방법으로 인가 수행
- 전송한 계정이 admin이 아니라면, 비정상적 계정 생성 방어

```
// 새로운 고객 정보 등록
}else if ( "edit".equals(action) && "admin".equals(session.getAttribute("userId"))) {
```

- 기존 kong(콩순이) 계정 삭제

안전한 소프트웨어를 만들기 위한 노력 (주)오픈이지

홈으로 | 게시판 | 시큐어코딩테스트 | ESAPI 테스트 | DB초기화 | [admin]님 로그인

시큐어코딩테스트

- 인코딩
- 정규식
- SQL 인젝션
- 명령어 인젝션
- XPath 인젝션
- XSS
- CSRF
- 암호화
- 오픈리다이렉트
- 보안쿠키
- 인증
- HTTP응답분할
- 접근제어
- 예외처리
- 점수오버플로우
- TOCTOU
- 세션간의 정보노출
- 반복문제어 부재
- 널포인트 역참조
- 랑술화 위배
- 중요정보 노출

접근제어

시스템 보안의 핵심중 하나가 인가입니다. 인가는 시스템 리소스에 대해 권한이 있는 사용자만 사용할 수 있도록 하는 정책입니다.

작업선택 : -- 고객 정보 삭제 --

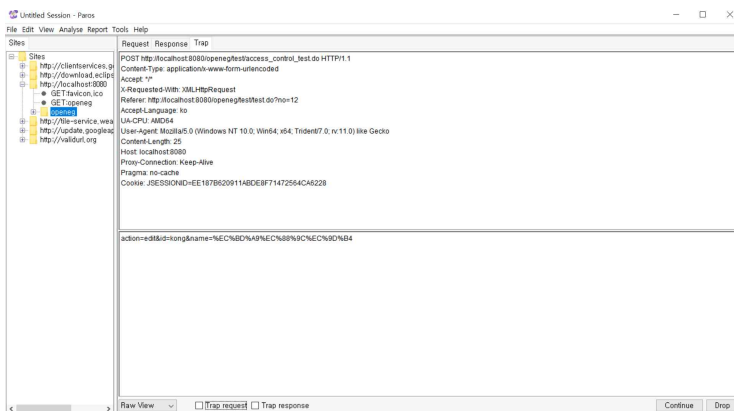
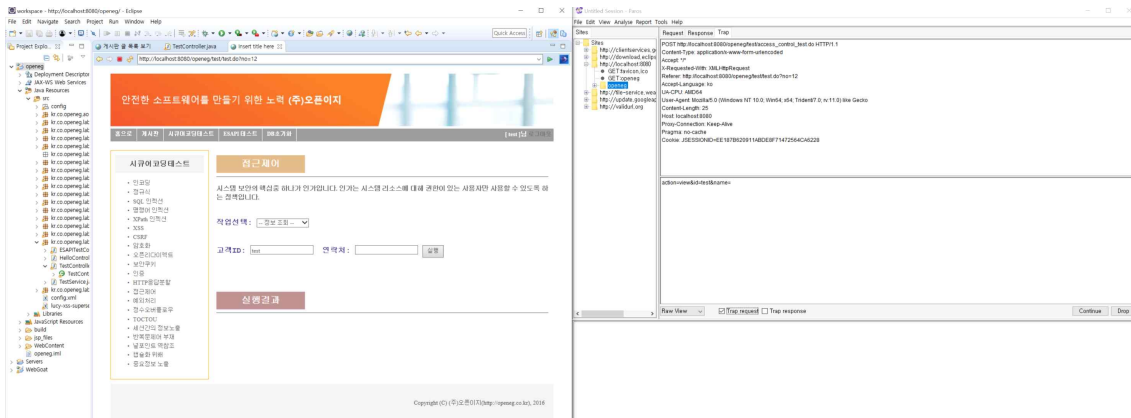
고객ID : kong 고객명 : 콩순이 실행

실행결과

kong님의 정보를 삭제하였습니다.

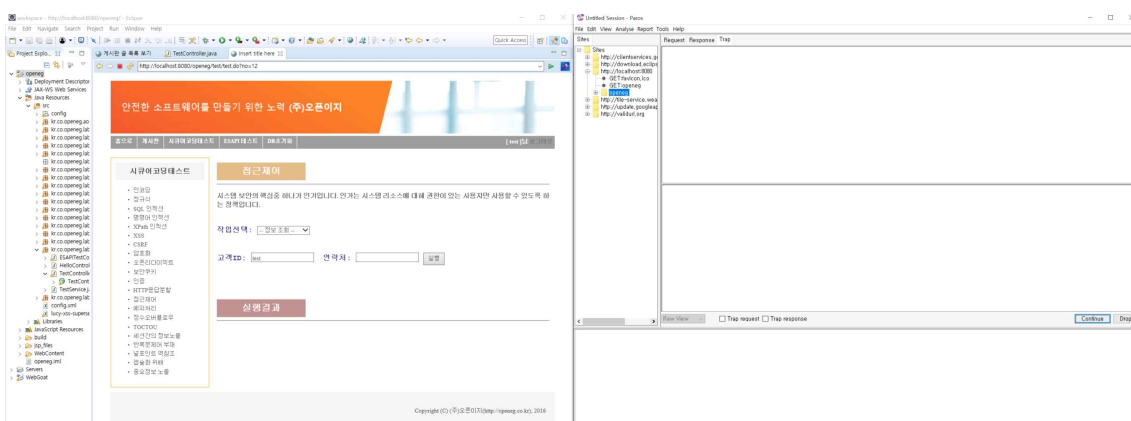
Copyright (C) (주)오픈이지(http://openeg.co.kr), 2016

- test 계정
- 위 공격방법과 같이 파라미터를 관리자가 생성한 파라미터로 변경
- trap request 해제 후, continue



<실행 결과>

변경 사항 없음(계정 생성 불가 확인)



이것은 일반 계정이 admin의 계정을 이용하여 특정한 파라미터를 변조하는 행위를 사전에 차단한 것이다. 일반 계정이 발생시켰던 파라미터 정보와 갖고 있는 권한을 비교하여 비정상적 행위를 차단하는 코드가 정상적으로 동작함을 확인할 수 있다.