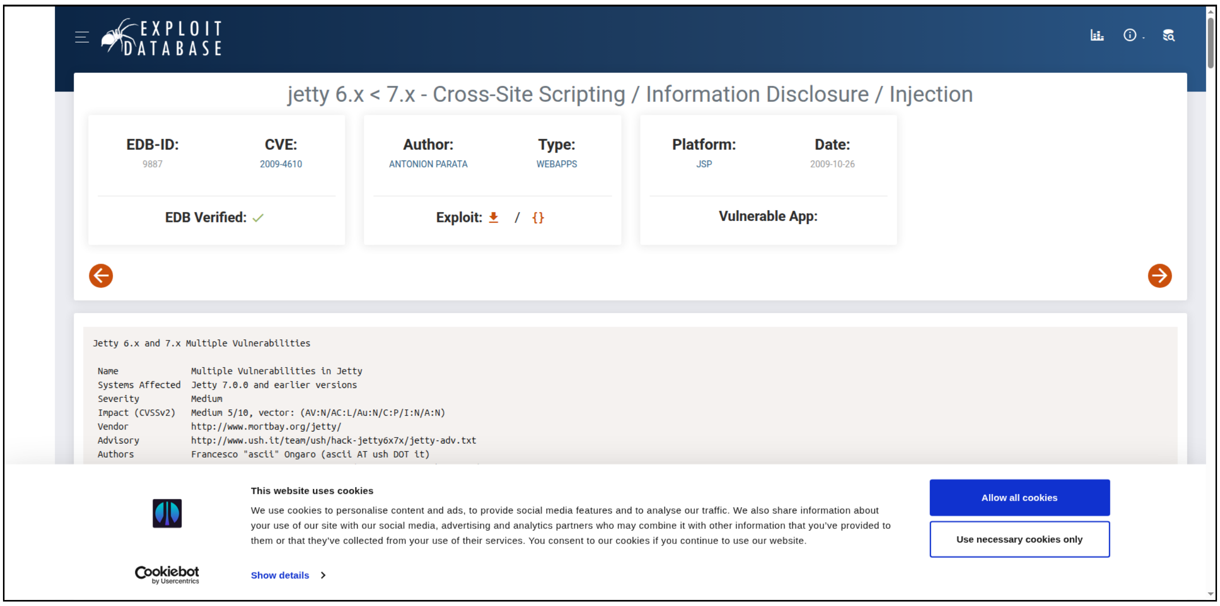
PRESS REVIEW ARCHIVE

Digital Media Monitoring & Documentation Service

Source URL:	https://www.exploit-db.com/exploits/9887
Archived Date:	August 15, 2025 at 14:47
Published:	October 26, 2009
Document Type:	Web Page Archive
Wayback Machine:	https://web.archive.org/web/*/https://www.exploit-db.com/exploits/9887

Page Screenshot



jetty 6.x < 7.x - Cross-Site Scripting / Information Disclosure / Injection

EDB-ID: 9887

CVE: 2009-4610

Author: ANTONION PARATA

Type: WEBAPPS

Platform: JSP

Date: 2009-10-26

EDB Verified: ✓

Exploit: ⤓ / {}

Vulnerable App:

```
Jetty 6.x and 7.x Multiple Vulnerabilities

 Name              Multiple Vulnerabilities in Jetty
 Systems Affected  Jetty 7.0.0 and earlier versions
 Severity          Medium
 Impact (CVSSv2)   Medium 5/10, vector: (AV:N/AC:L/Au:N/C:P/I:N/A:N)
 Vendor            http://www.mortbay.org/jetty/
 Advisory          http://www.ush.it/team/ush/hack-jetty6x7x/jetty-adv.txt
 Authors           Francesco "ascii" Ongaro (ascii AT ush DOT it)
```

jetty 6.x < 7.x - Cross-Site Scripting / Information Disclosure / Injection

EDB-ID:
9887

CVE:
2009-4610

EDB Verified: ✓

Author:
ANTONION PARATA

Type:
WEBAPPS

Exploit: ⬇ / {}

Platform:
JSP

Date:
2009-10-26

Vulnerable App:

```
Jetty 6.x and 7.x Multiple Vulnerabilities

 Name              Multiple Vulnerabilities in Jetty
 Systems Affected  Jetty 7.0.0 and earlier versions
 Severity          Medium
 Impact (CVSSv2)   Medium 5/10, vector: (AV:N/AC:L/Au:N/C:P/I:N/A:N)
 Vendor            http://www.mortbay.org/jetty/
 Advisory          http://www.ush.it/team/ush/hack-jetty6x7x/jetty-adv.txt
 Authors           Francesco "ascii" Ongaro (ascii AT ush DOT it)
                   Giovanni "evilaliv3" Pellerano (evilaliv3 AT ush DOT it)
                   Antonio "s4tan" Parata (s4tan AT ush DOT it)
 Date              20091024

I. BACKGROUND

Jetty is an open-source project providing a HTTP server, HTTP client and
javax.servlet container. These 100% java components are full-featured,
standards based, small foot print, embeddable, asynchronous and
enterprise scalable. Jetty is dual licensed under the Apache Licence
2.0 and/or the Eclipse Public License 1.0. Jetty is free for commercial
use and distribution under the terms of either of those licenses.

Jetty is used in a wide variety of projects and products: embedded in
phones, in tools like the the eclipse IDE, in frameworks like GWT, in
application servers like Apache Geronimo and in huge clusters like
Yahoo's Hadoop cluster.

The latest version at the time of writing can be obtained from:
http://dist.codehaus.org/jetty/jetty-7.0.0/jetty-hightide-7.0.0.v2009100
5.tar.gz

Running Jetty 7.0.x is very easy, from the documentation page at:
http://docs.codehaus.org/display/JETTY/Running+Jetty-7.0.x

- From an unpacked release directory of jetty-7,
  the server can be started with the command: java -jar start.jar

- This will start a HTTP server on port 8080 and
  deploy the test web application at: http://localhost:8080/test

II. DESCRIPTION

Multiple Vulnerabilities exist in Jetty software.

III. ANALYSIS

Summary:

 A) "Dump Servlet" information leak
    (Affected versions: Any)

 B) "FORM Authentication demo" information leak
    (Affected versions: Any)

 C) "JSP Dump" reflected XSS
    (Affected versions: Any)

 D) "Session Dump Servlet" stored XSS
    (Affected versions: Any)

 E) "Cookie Dump Servlet" escape sequence injection
    (Affected versions: Any)

 F) Http Content-Length header escape sequence injection
    (Affected versions: Any)

 G) "Cookie Dump Servlet" stored XSS
    (Affected versions: =<6.1.20)

 H) WebApp JSP Snoop page XSS
    (Affected versions: =<6.1.21)


A) "Dump Servlet" information leak
   (Affected versions: Any)

By requesting the demo "Dump Servlet" at an URL like "/test/dump/"
it's possible to obtain a number of details about the remote Jetty
instance.

Variables: getMethod, getContentLength, getContentType, getRequestURI,
getRequestURL, getContextPath, getServletPath, getPathInfo,
getPathTranslated, getQueryString, getProtocol, getScheme,
getServerName, getServerPort, getLocalName, getLocalAddr,
getLocalPort, getRemoteUser, getRemoteAddr, getRemoteHost,
getRemotePort, getRequestedSessionId, isSecure(), isUserInRole(admin),
getLocale, getLocales, getLocales

Plus a dump of all the HTTP request headers, the request parameters
and much more.

Five forms can be used to perform a series of functionality tests
including:

  - Form to generate GET content
  - Form to generate POST content
  - Form to generate UPLOAD content
  - Form to set Cookie
  - Form to get Resource

While this is a feature we think that demo utilities should be
disabled by default. Many live deployments of Jetty exhibit demo
pages that leak important information and expose several vulnerabilites.

B) "FORM Authentication demo" information leak
   (Affected versions: Any)

An example application often erroneously deployed is the "FORM
Authentication demo" (logon.html and logonError.html pages) that uses
the standard "j_security_check" component.

By requesting the "/test/logon.html" page it's possible to detect the
presence of a Jetty installation.

As noted before we think that demo utilities should be disabled by
default.

C) "JSP Dump" reflected XSS
   (Affected versions: Any)

It has been found that the demo "JSP Dump" feature is vulnerable to
reflected Cross Site Scripting attacks. This can be replicated by
issuing a GET request to the "/test/jsp/dump.jsp" page:
"/test/jsp/dump.jsp?%3Cscript%3Ealert(%22hello%20world%22)%3C/script%3E"

Any GET key and value that reach the remote is reflected unencoded.

The problem resides in the "jsp/dump.jsp" file from the
"webapps/test.war" archive.
```

"webapps/test.war" archive.

```
--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--

<html><head>
<%@ page import="java.util.Enumeration" %>
</head><body>
<h1>JSP Dump</h1>

<table border="1">
<tr><th>Request URI:</th><td><%= request.getRequestURI() %></td></tr>
<tr><th>ServletPath:</th><td><%= request.getServletPath() %></td></tr>
<tr><th>PathInfo:</th><td><%= request.getPathInfo() %></td></tr>

<%
   Enumeration e =request.getParameterNames();
   while(e.hasMoreElements())
   {
       String name = (String)e.nextElement();
%>
<tr>
  <th>getParameter("<%= name %>")</th>
  <td><%= request.getParameter(name) %></td></tr>
<% } %>

</table>
</body></html>

--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--
```

As shown no encoding is applied to user inputs.

D) "Session Dump Servlet" stored XSS
   (Affected versions: Any)

It has been found that the "Session Dump Servlet" feature is affected by a stored Cross Site Scripting vulnerability.

The servlet, mapped in "/session/", normally uses HTTP POST parameters but also accepts values by HTTP GET granting easier exploitation.

The issue can be verified by requesting an URL like the following:

```
"/test/session/?R=0&Name=%3Cscript%3Ealert(%27name%27)%3C/script%3E&Valu
e=%3Cscript%3Ealert(%27value%27)%3C/script%3E&Action=Set"
```

Any consecutive request to "/test/session/" without parameters will include the previously inserted payloads.

Session keys and values are reflected unencoded both on the first and successive requests.

E) "Cookie Dump Servlet" escape sequence injection
   (Affected versions: Any)

Making a POST request to the form at "/test/cookie/" with the "Age" parameter set to a string throws a "java.lang.NumberFormatException" exception.

The backtrace output is not sanitized from escape sequences, this vulnerability is similiar to CVE-2003-0020 [1] and CVE-2003-0083 [2].

While the backtrace is protected from Cross Site Scripting attacks it still reflects as-is many binary characters including ESC. These special characters are used in control sequences to instruct the terminal to perform special operations like executing commands [3, 4] or dumping the buffer to a file [5, 6].

This issue can be demonstrated with the following Proof of Concept using a non-dangerous escape sequence that will change the Xterm title.

In the first terminal:

```
$ echo -en "\x1b]2;safe?\x07\x0a"; java -jar start.jar etc/jetty.xml;
```

In the second terminal:

```
$ curl -kis "http://localhost:8080/cookie/" -d "Action=Set&Name=aa&Value
=bb&Age=%1b%5d%32%3b%6f%77%6e%65%64%07%0a"
```

Logs can be found in logs/[date].stderrout.log or are printed directly to the terminal. A user that views (cat, tail -f) these logs or runs Jetty in his tty/pty may get influenced in a negative way by a malicious control sequence.

The code involved in this vulnerability is in the Java class found at "test-jetty-webapp/src/main/java/com/acme/CookieDump.java".

```
--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--

    protected void handleForm(HttpServletRequest request,
                          HttpServletResponse response)
    {
        String action = request.getParameter("Action");
        String name =  request.getParameter("Name");
        String value =  request.getParameter("Value");
        String age =  request.getParameter("Age");

        if (name!=null && name.length()>0)
        {
            Cookie cookie = new Cookie(name,value);
            if (age!=null && age.length()>0)
                cookie.setMaxAge(Integer.parseInt(age));
            response.addCookie(cookie);
        }
    }

--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--
```

The problem also exists for other demo pages, see for example the "/test/jsp/expr.jsp" page (eg: "/test/jsp/expr.jsp?A=LetteralString").

This issue is generic to all the Java applications that use the standard error handling mechanism. Consider this an advisory also for Java JVM.

F) HTTP Content-Length header escape sequence injection
   (Affected versions: Any)

The same attack vector presented in point "E" can be exploited by requesting a page using an HTTP request "Content-Length" header set to a letteral string. This raises a type mismatch exception as previously shown. See the following PoC:

```
$ curl 127.0.0.1:8080 -H "Content-Length: abcde"
```

The difference between this possibility and the one exposed in "E" is that this works not only on Jetty default demo applications but on every application that Jetty will serve.

G) "Cookie Dump Servlet" stored XSS
   (Affected versions: =<6.1.20)

On Tue, 06 Oct 2009 the CORE-2009-0922 advisory killed part of our research. The advisory titled "Jetty Persistent XSS in Sample Cookies

Application" is about this specific point.

Out initial writing is presented anyway:

The "Cookie Dump Servlet" works in a similar way as the previous "Session Dump Servlet", accepting GET parameters. The difference is that two requests are needed (as it was a stored POST XSS) since the first will trigger the Set-Cookie and the second request will echo it. This issue can be replicated with the following request:

```
"/cookie?R=1&Name=<token1>&Value=<token2>&Age=60&Action=Set"
```

Input values are stored and presented unescaped.

H) WebApp JSP Snoop page XSS
   (Affected versions: =<6.1.21)

All the Jetty 6.1.X versions are affected by a reflected XSS in the JSP Snoop page. This does not work on the 7.X branch.

When called by it's deploy the "WebApp JSP Snoop page" (/jspsnoop) is vulnerable to XSS:

```
"/jspsnoop/ERROR/%3Cscript%3Ealert(123)%3C/script%3E"
"/jspsnoop/IOException/%3Cscript%3Ealert(123)%3C/script%3E"
"/jspsnoop/%3Cscript%3Ealert(123)%3C/script%3E"
```

"Path translated" and "Path info" are not encoded.

This in not exploitable when the page is implicitly called, for example to handle a 404 page as the "Path translated" is always "/ERROR/404". The same happens when requiring the page by its filesystem location (/snoop.jsp).

IV. DETECTION

Jetty 7.0.0 and possibly earlier versions are vulnerable.

Jetty can be identified using the following examples or by directly requesting files and locations marked as "information leak" in this advisory.

Some examples:

  - intitle:"Powered By Jetty"
  - intitle:"JSP snoop page" "WebApp JSP Snoop page"
  - inurl:"snoop.jsp
  - "Welcome to Jetty 7"

V. WORKAROUND

The vendor decided not to modify the release schedule in order to publish a version to address the presented issues. We have been sayd that the next release (available in 1 to 3 weeks) will resolve all the issues in the demo applications and the command sequence injection vulnerability.

The ESC insertion problems will be resolved by:

- Handling the particular exceptions you found (NumberFormateException).
- Updating the stderrlogger so that all user supplied output is stripped of non whitespace ISO control characters.
- Stripping ISO control characters from generated error pages.

In the meantime the vendor provides the following workaround recommendations:

- The test webappplications must not be deployed on production sites. An administrator can remove the "webapps/test.war" file and/or the "webapps/test directory", plus the contexts/test.xml file. To be clear again: DON'T DEPLOY THE TEST WEBAPP ON PRODUCTION.

- Do not run Jetty as the root user. Instead run as a limited user and redirect port 80 to the port that jetty is using. See http://docs.codehaus.org/display/JETTY/port80.

- Do not run production jetty instances from a console. Instead start in the background using an /etc/init.rc/jetty.sh script or similar.

- Redirect stderr to a file. This can be done with the jetty-logging.xml file as follows:
  ```
  java -jar start.jar etc/jetty.xml etc/jetty-logging.xml
  ```

- Process log files with cat -v if you wish to display them on a console without using an editor.

VI. VENDOR RESPONSE

Vendor will not release a new version to address these issues but is working on them in the SNAPSHOT versions.

http://svn.codehaus.org/jetty/jetty/branches/jetty-6.1/VERSION.txt

We had no precise response about remediation status and plans but we were told that it was okay to release this advisory. It elapsed about a week from the last email sent to the vendor, since we got no reply we assume that it's okay to release.

VII. CVE INFORMATION

No CVE at this time.

VIII. DISCLOSURE TIMELINE

20090616 Bug discovered
20091006 CORE-2009-0922 reveals an XSS issue (point G)
20091006 Jetty branch 7 kills the "jspsnoop" issue (point H)
20091011 Internal version of this advisory finalized
20091013 First vendor contact
20091014 Vendor Response, "We are working on XSS on demo apps"
20091015 Asking for release timeline
20091015 Vendor Response, Suggests a remediation, "Okay to release"
20091015 Remediation doesn't fix the Escape Sequence issue
20091015 Vendor Response, ACK, "Wait until next week"
20091016 Vendor Response, "We are not going to rush out a release to fix the escape injection problem"
20091024 Advisory release

IX. REFERENCES

[1] Apache does not filter terminal escape sequences from error logs
    http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-0020
[2] Apache does not filter terminal escape sequences from access logs
    http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-0083
[3] Debian GNU/Linux XTERM (DECRQSS/comments) Weakness Vulnerability
    http://www.milw0rm.com/exploits/7681
[4] Terminal Emulator Security Issues
    http://marc.info/?l=bugtraq&m=104612710031920&w=2
[5] Eterm Screen Dump Escape Sequence Local File Corruption Vulnerability

```
    https://www.securityfocus.com/bid/6936/discuss
[6] RXVT Screen Dump Escape Sequence Local File Corruption Vulnerability
    https://www.securityfocus.com/bid/6938/discuss

X. CREDIT

Francesco "ascii" Ongaro, Giovanni "evilaliv3" Pellerano and
Antonio "s4tan" Parata are credited with the discovery of this
vulnerability.

Francesco "ascii" Ongaro
web site: http://www.ush.it/
mail: ascii AT ush DOT it

Giovanni "evilaliv3" Pellerano
web site: http://www.ush.it/, http://www.evilaliv3.org/
mail: evilaliv3 AT ush DOT it

Antonio "s4tan" Parata
web site: http://www.ush.it/
mail: s4tan AT ush DOT it

X. LEGAL NOTICES

Copyright (c) 2009 Francesco "ascii" Ongaro

Permission is granted for the redistribution of this alert
electronically. It may not be edited in any way without mine express
written consent. If you wish to reprint the whole or any
part of this alert in any other medium other than electronically,
please email me for permission.

Disclaimer: The information in the advisory is believed to be accurate
at the time of publishing based on currently available information. Use
of the information constitutes acceptance for use in an AS IS condition.
There are no warranties with regard to this information. Neither the
author nor the publisher accepts any liability for any direct, indirect,
or consequential loss or damage arising from use of, or reliance on,
this information.
```

Tags:

Advisory/Source: Link