Published in New Alchemy

New Alchemy Follow
Jul 18, 2018 · 27 min read · Listen

XMR Wallet Security Review

Xmr Wallet engaged New Alchemy to perform a security review of their `XMRWallet` web application shown in figure 1 below. The review was technical in nature and focused on identifying the susceptibility to security flaws in the application's behavior that may impact trustworthiness. The user interface and web traffic were inspected, along with a portion of the source code as dynamically delivered from the server at https://www.xmrwallet.com.

The review and test was performed in early June 2018. XMRWALLET.COM additionally provided access to the client-side application at https://github.com/XMRWallet/Website with a commit hash of `b03152135cf4577c320aae2ee51f9a6fae25437c` . The private server-side API functionality, obfuscated client code and cryptography was out of scope. This document describes

The `XMRWallet` application exhibits a high-quality user experience, a modern development approach, and a clear separation of client and server functionality. However, the security review has identified a number of potential vulnerabilities. Given the nature of the application, these were conservatively rated in severity from critical to minor. All are believed to be fixable. Several examples include:

- A cross-site scripting vulnerability stemming from the price feed
- Outdated component dependencies on both the client and the server
- Missing security-relevant headers as received from the server
- Inadvisable display of private fields and input auto-completion
- Potentially risky usage of JavaScript and HTML/DOM functionality

Re-test v2.0: XMRWallet's audit response indicates that all critical issues reported by New Alchemy have been addressed.

Re-test v2.0: New Alchemy has inspected the application and repository provided for re-testing and believes that critical issues #1, #4, #5 and #7 have been fixed, critical issues #2 and #3 have been partially fixed, and critical issues #6 has not yet been fixed. Further details are included in the relevant sections below, including re-test notes for each of the other moderate and minor issues.

Re-test v2.1: New Alchemy has discussed the prior re-test results with XMRWallet, advised on efficient and effective mitigation approaches, inspected a subsequent application and repository provided for re-testing, and concludes:

- All seven critical issues have been fixed and their risks mitigated.
- Five of six moderate issues have been fixed and their risks mitigated. The remaining issue has been re-marked as 'informational' as it highlights a more general concern rather than a specific security issue.
- Three of six minor issues have been fixed and their risks mitigated. Of the remaining three, one has been partially fixed and two have been re-marked as

does not present a significant risk.

Supporting Files

The review primarily involved examining the application behavior, interface and web traffic. This was partially supported by the code residing in the GitHub repository at https://github.com/XMRWallet/Website with a commit hash of `b03152135cf4577c320aae2ee51f9a6fae25437c`. The file `/src/js/monero.js` was out of scope as it was obfuscated and derived from open source Monero code.

```
src/js:
  app.js            jquery.i18n.js     jquery.js          select2.js

src/template:
  account.twig      index.twig         login.twig
receive.twig
  create.twig       landing.twig       maintenance.twig   send.twig
  dashboard         layout             policy_login.twig
support_login.twig
  dashboard.twig    login_public.twig  policy.twig
support.twig
  terms_login.twig  terms.twig         transactions.twig
```

Re-test v2.0: New Alchemy has inspected the same repository with a new commit hash of `233ddd412d5a5afc877ba9dbab1524104767c338` under the same conditions noted above. The actual application under test was served from the same URL as noted above.

Re-test v2.1: New Alchemy has inspected the same repository with a new commit hash of `abc0ab0a3d2c9cf587672ca03d80ade199e8e6ee` under the same conditions noted above. The actual application under test was served from the same URL as noted above.

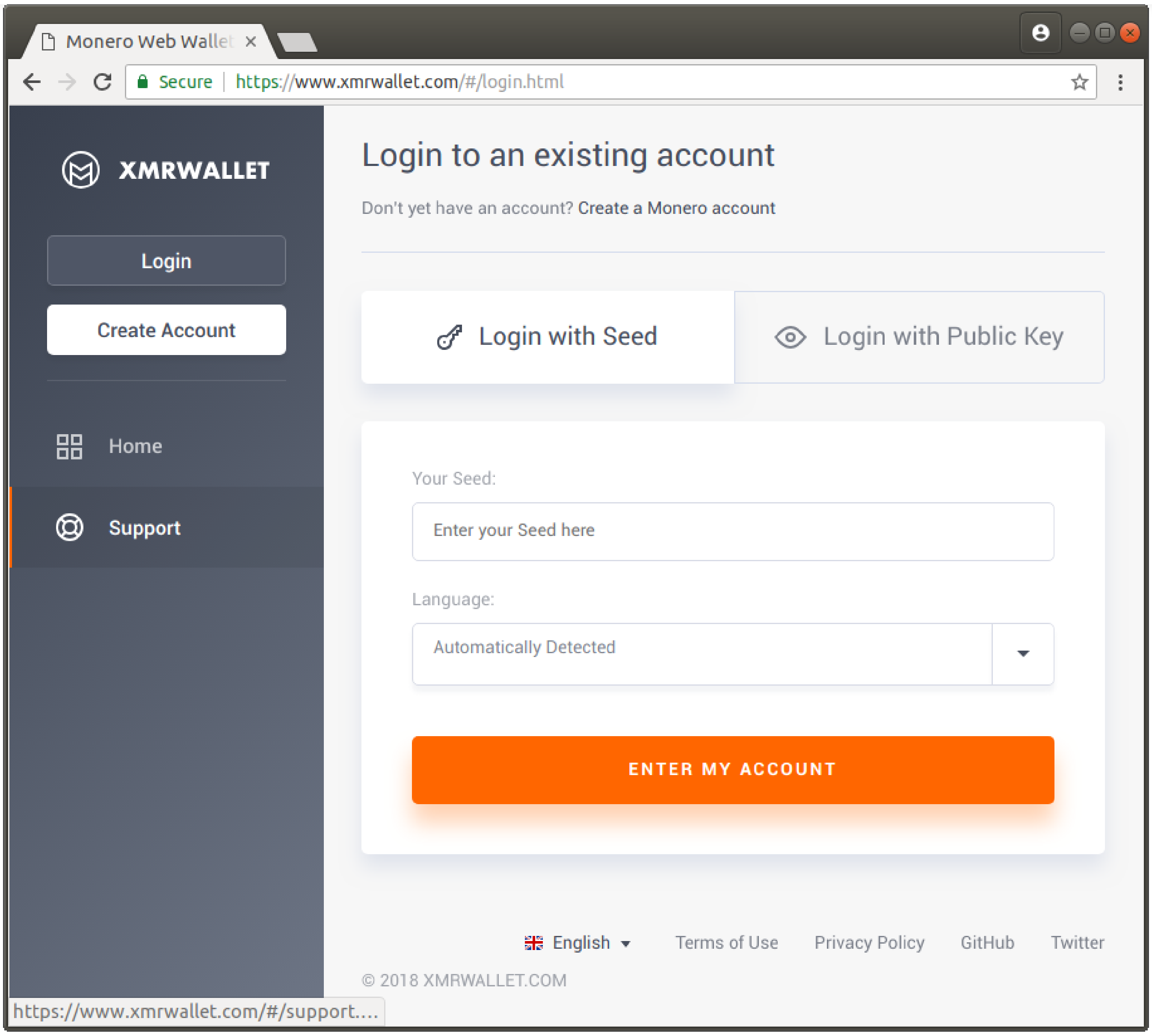


Figure 1: The https://www.xmrwallet.com Application

General Discussion

The `XMRWallet` application provides an excellent and intuitive user interface. Each aspect of the application was exercised, including value transfers to and from multiple counter-parties. The code organization and development process facilitated understanding how components fit together. A key strength of the application is minimal endpoints, minimal external data dependencies and minimal unrelated web traffic.

The client-side application has two clear portions. First, the 'twig' files noted above reside on the server and are dynamically delivered to the client as pages or portions of

The server-side application consists of numerous PHP API endpoints. This code and related functionality was out of scope. However, any relevant observations are noted in this report.

The primary client pages are listed below on the left. Server endpoints are listed on the right. Some server endpoints are extraneous (see Finding 17) and all were reachable via `https://`.

- www.xmrwallet.com
- www.xmrwallet.com/auth.php
- www.xmrwallet.com/account.html
- www.xmrwallet.com/css/index.php
- www.xmrwallet.com/dashboard.html
- www.xmrwallet.com/font/index.php
- www.xmrwallet.com/index.html
- www.xmrwallet.com/getbalance.php
- www.xmrwallet.com/js/app.js
- www.xmrwallet.com/getfees.php
- www.xmrwallet.com/login.html
- www.xmrwallet.com/getheightsync.php
- www.xmrwallet.com/login_public.html
- www.xmrwallet.com/getoutputs.php
- www.xmrwallet.com/receive.html
- www.xmrwallet.com/getrandomoutputs.php
- www.xmrwallet.com/robots.txt

- www.xmrwallet.com/getunspentoutputs.php
- www.xmrwallet.com/support_login.html
- www.xmrwallet.com/i/index.php
- www.xmrwallet.com/transactions.html
- www.xmrwallet.com/js/index.php
- www.xmrwallet.com/logout.php
- www.xmrwallet.com/submittransaction.php
- www.xmrwallet.com/updateoutputs.php

As the client-side application is dynamically delivered from the server and subsequently works in conjunction with out of scope server-side functionality, a full review would require complete source code of both sides. The dynamic nature of the application prevents stable hashes from being easily inspected by the user. Portions of obfuscated code represent out of scope black-boxes. Nonetheless, the client-side application's behavior, user experience and web traffic can be robustly reviewed.

The review uncovered a number of findings ranging in severity from critical to minor. From inspecting the application, it appears that fixes will not require major code rip-up or drive fundamental architectural change. The resulting fixes will provide a significant uplift in application trustworthiness.

Critical Issues

1. Fixed: Insecure Auto-completion of Login Fields

Impact

An attacker lacking any credentials can use the browser's auto-completion feature to login into a victims account. The attacker will be able to impersonate the victim and perform malicious transactions.

is enabled by default and must be disabled on sensitive fields. The `XMRWallet` application does not disable this on the seed login field.

The server returns the following snippet in response to a POST request to `login.html`.

```
...
<form id="login_account">
  <div class="form-group">
    <label class="label">
          <span data-i18n="label_private_login_key">Your Seed:
</span>
         </label>
    <input id="private_seed" type="text" class="form-control"
               placeholder="Enter your Seed here"/>
  </div>
...
```

An attacker is able to open up a fresh window and have the previous user's seed provided as shown in figure 2 below.

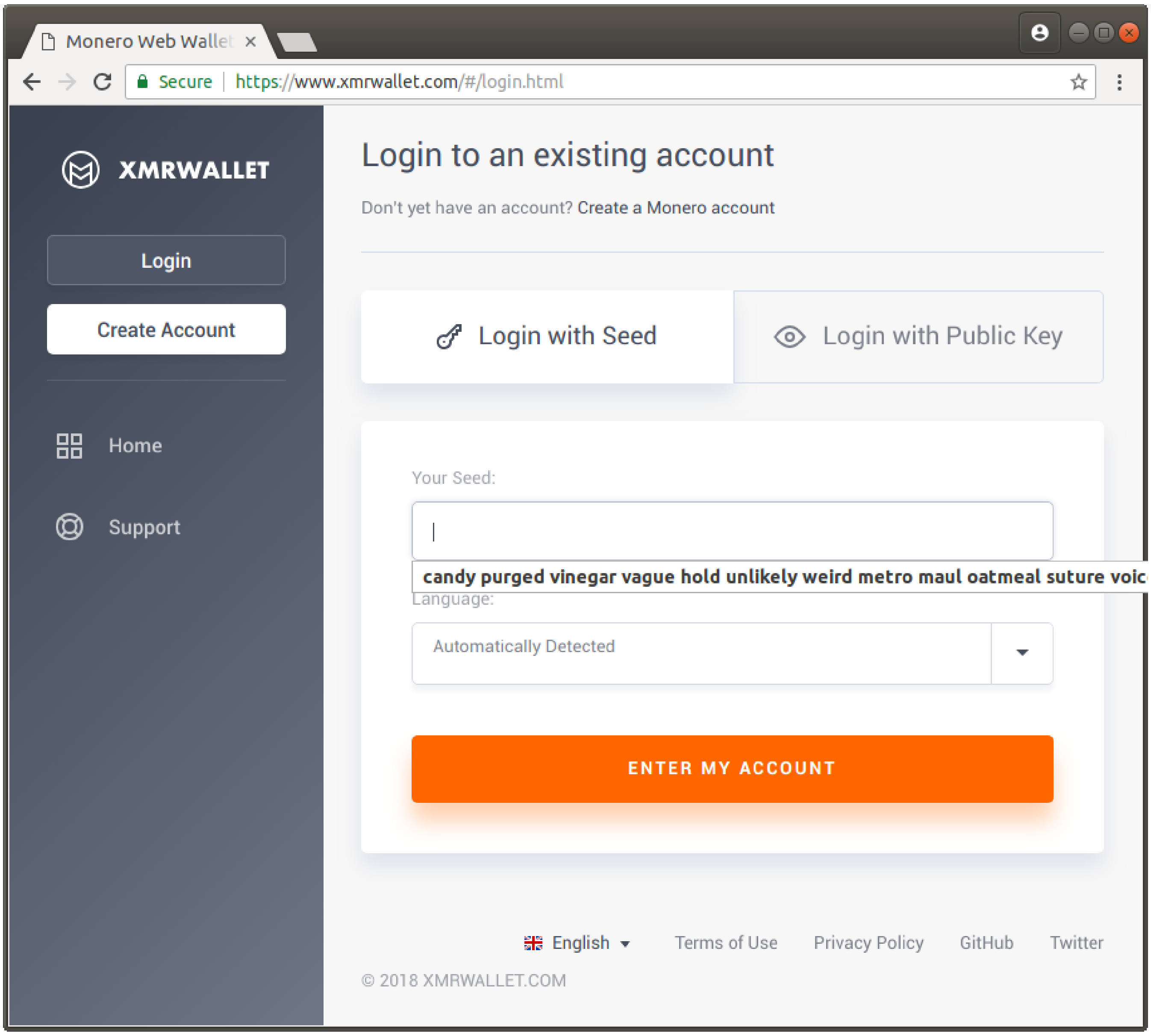


Figure 2: Auto-completion of Seed Field

Remediation

Include the `autocomplete="off"` (or an invalid value) attribute in the form element. Mozilla has a good article describing the oddities around Disabling Autocompletion.

Re-test v2.0: New Alchemy has inspected the application and has determined that this issue has been fixed and the associated risk mitigated. This was confirmed by inspecting the HTML fragment returned by `login.html` for the presence of an `autocomplete="off"` attribute, along with manually testing the function.

2. Fixed: Cross-Site Scripting Vulnerability

The `XMRWallet` application is vulnerable to cross-site scripting (XSS) via the response from the https://min-api.cryptocompare.com/data/price?fsym=XMR&tsyms=USD,BTC price feed. An attacker can steal a victim's session tokens, log their keystrokes, steal private data, or perform malicious transactions in the context of a victim's session.

Description and Discussion

Cross-site scripting is a vulnerability class related to web application input and output validation. In XSS, the application accepts malicious input and later displays it without properly encoding HTML meta-characters. This allows an attacker to inject JavaScript code into views of the susceptible page. A user may fall victim to the attack just by using the application. XMRWallet's requests are easily distinguished by the presence of the `Referer: https://www.xmrwallet.com/` header as discussed in finding 6.

The exploitation of this vulnerability is outlined below:

```
The application makes a request to the price quote API endpoint at
      https://min-api.cryptocompare.com/data/price?
fsym=XMR&tsyms=USD,BTC

   The remote server API is expected to return a response of the
form
      {"USD":166.08,"BTC":0.02155}

   HTML encoding an example payload of "XSS Injection Via
CryptoCompare", including
   the result inside of an malformed image tag onerror alert, and
adjusting the API
   return value gives:
      {"USD":999,"BTC": "<img src=x
onerror=alert(%22%58%53%53%20%49%6e%6a%65%63

%74%69%6f%6e%20%56%69%61%20%43%72%79%70%74%6f%43%6f%6d%70%61%72%65%2
2)>"}

   When the above payload is presented to the application, it
demonstrates the
   resulting XSS exploitation shown in figure 3 below.
```

effectively break every user wallet at once. Additional background information on XSS can be found on OWASP's XSS Page.

Remediation

Application input may originate from maliciously modified “normal user clients”, man-in-the-middle attackers, and/or from unreliable data providers. Thus all external application input should be considered untrusted and very carefully processed before any usage. This generally includes distinct stages of canonicalization, normalization, sanitization and validation in that order.

Canonicalization reduces input that may have several possible character representations into a standard character set (e.g. à á â ã a). Normalization reduces the input into its simplest possible form (e.g. ../../etc/passwd /etc/passwd). Sanitization eliminates unwanted characters and sequences from the input by means of recursively removing, replacing, encoding, or escaping the characters and sequences (e.g. <scr-<script>-ipt> scr-script-ipt). Validation ensures the input conform to the required schema and is consistent for use (e.g. alphanumeric only, 1+1=3). The collection of these stages are often loosely referred to as “validation” and their implementation can vary depending upon situational applicability.

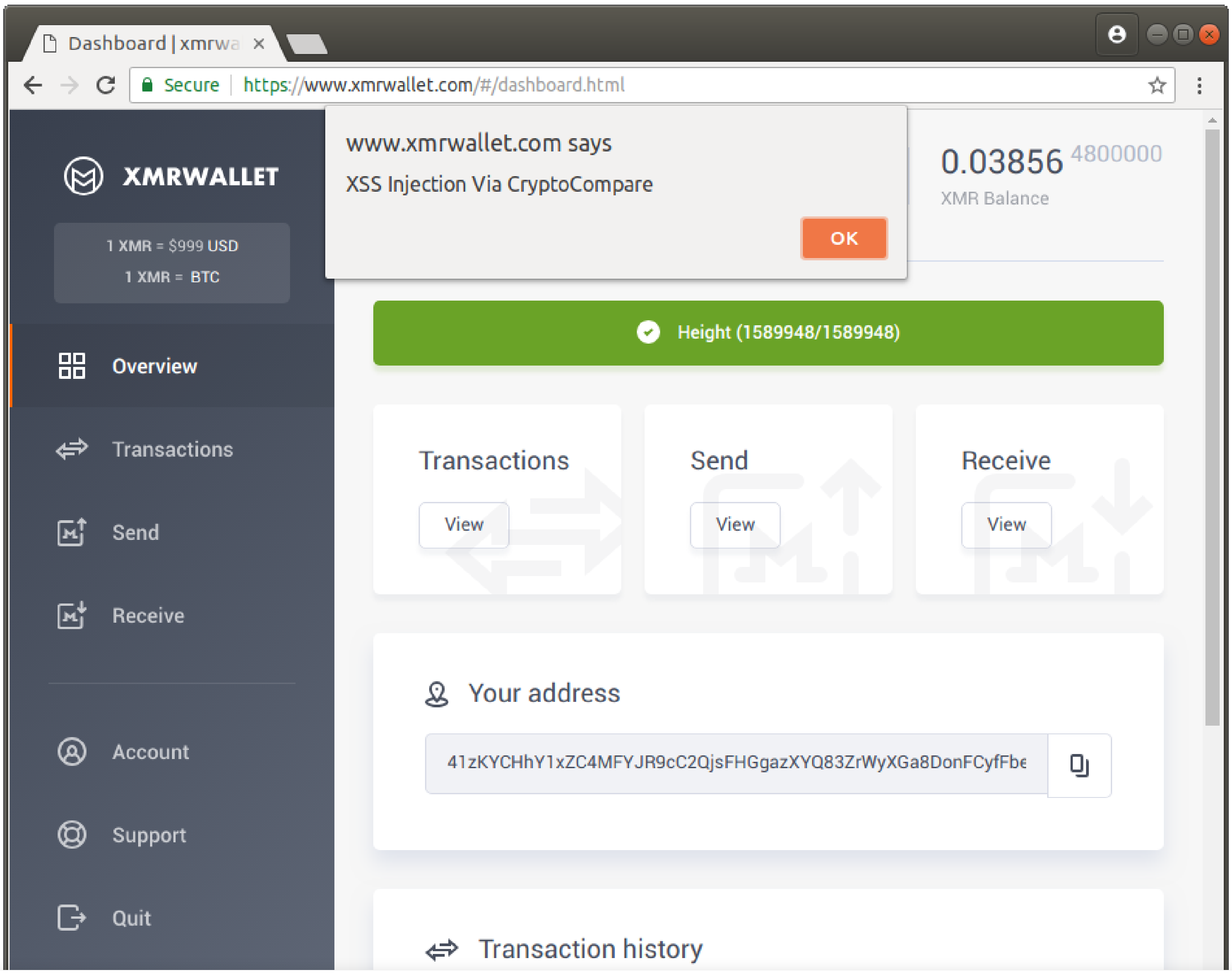
In this specific instance, parsing expected values via a tightly constrained `regex` , validating it against expected ranges, and escaping the related output will be sufficient. Further work may be necessary on the server side as indicated by finding 4 below.

Re-test v2.0: New Alchemy has inspected the application and has determined that this issue has been partially fixed. The price data is now directly sourced from the application server via a new `getprice.php` endpoint rather than the third party service as originally noted. As such, it is no longer *directly* dependent upon the integrity of the third party pricing response, which is a large improvement. Note that New Alchemy does not have visibility into the new server-side endpoint sourcing or handling of the pricing data.

Re-test v2.0: However, the application continues to neglect validation of external input via the `getprice.php` response as originally noted. New Alchemy confirmed that validating input against expected ranges and escaping the related output remains strongly recommended.

Re-test v2.0: To be clear, New Alchemy considers the risk to be drastically reduced but not sufficiently eliminated to be considered fully fixed. An undiscovered scenario similar to that demonstrated in issue #4 below (OS Command Injection) on the server could result in the broadcast of malicious pricing data causing all user wallets to be completely compromised. Validating the pricing data as recommended would prevent this.

Re-test v2.1: New Alchemy has inspected the application and identified the new code which locally multiplies incoming price quotes by 1.0. As a result, (potentially malicious) strings no longer have a route into the DOM. While a minimalistic approach, it is sufficient to consider this issue fixed and the risk mitigated.



www.xmrwallet.com says

XSS Injection Via CryptoCompare

OK

1 XMR = $999 USD
1 XMR = BTC

0.03856 4800000
XMR Balance

Height (1589948/1589948)

41zKYCHhY1xZC4MFYJR9cC2QjsFHGgazXYQ83ZrWyXGa8DonFCyfFbe

3. Fixed: Outdated Client-Side Application Dependencies

Impact

Outdated dependencies may expose the application to publicly known vulnerabilities, providing easily discoverable methods of attack. This approach was used in the well known 'epic' Equifax breach.

Description and Discussion

The application codebase contains several outdated client-side dependencies, some with publicly known vulnerabilities. The dependencies listed below are outdated and expose the `XMRWallet` application to known vulnerabilities of medium-to-low severity.

- atob 1.1.3 is approximately 2 years old; the latest version of atob is v2.1.1
- atob 2.0.3 is approximately 2 years old; the latest version of atob is v2.1.1
- hoek 2.16.3 is approximately 3 years old; the latest version of hoek is v5.0.3
- jquery 1.7.1 is more than 5 years old; the latest version of jQuery is v3.3.1
- lodash 1.0.2 is approximately 5 years old; the latest version of lodash is v4.17.10
- mixin-deep 1.3.0 is only marginally outdated; the latest version of mixin-deep is 1.3.1
- mustache 2.0.0 is approximately 3 years old; the latest version of mustache is v2.3.0
- sshpk 1.13.1 is approximately 1 year old; the latest version of sshpk is v1.14.2

JavaScript is well known for the complexity of transitive dependencies — multiple versions of the same library can be loaded into memory simultaneously and reached via different execution paths. In addition, out of date dependencies can often be located at the end of a very long transitive chain. In some cases, build dependencies are mixed with test and deployment dependencies. Retire.js is a useful tool to check the freshness of dependencies.

The known CVE vulnerabilities from the dependencies listed above include:

- CVE-2015–9251 jQuery before 3.0.0 is vulnerable to Cross-site Scripting attacks
- CVE-2018–3745 atob 2.0.3 and earlier allocates uninitialized Buffers

This finding is categorized as critical due to the number of known CVEs and breadth of packages. The cross-site scripting exposure directly relates to finding 1. In this instance, updating dependencies is not expected to introduce functional breakage.

Remediation

Extract the definitive list of deployed server-side and client-side dependencies, review against recommended production versions, and update the correct versions into the build process. Incorporate a periodic review of dependency versions into the development schedule that gates delivery of code into production. Update the build and test dependencies in a similar fashion.

Re-test v2.0: New Alchemy has inspected the application and has determined that this issue has been partially fixed. Due to the rapidly changing and deep transitive nature of nodeJs dependencies it is unrealistic to expect perfection. However, there remain several outdated packages that would greatly benefit from updating and thereby reduce CVE exposure. These include (see `package.json`):

```
Re-test v2.0:
   "gulp-sass": "^3.2.1"          --->     "gulp-sass": "^4.0.1"
   "gulp-sass-image": "^1.0.2"    --->     "gulp-sass-image":
"^1.1.0"
   "browserify": "^14.5.0"             --->     "browserify":
"^16.2.2"
   "dialog-polyfill": "^0.4.9"    --->     "dialog-polyfill":
"^0.4.10"
```

Re-test v2.0: Please regularly re-run `retire` (see https://www.npmjs.com/package/retire) to uncover the use of vulnerable libraries and ensure the application remains up to date.

Re-test v2.1: New Alchemy has inspected the respository and considers this issue fixed and the risk mitigated.

An attacker can inject OS commands into the server resulting in complete compromise of the server, application and associated data. While the functionality behind the server API is out of scope, it is important to test the robustness of client-server communication protocols.

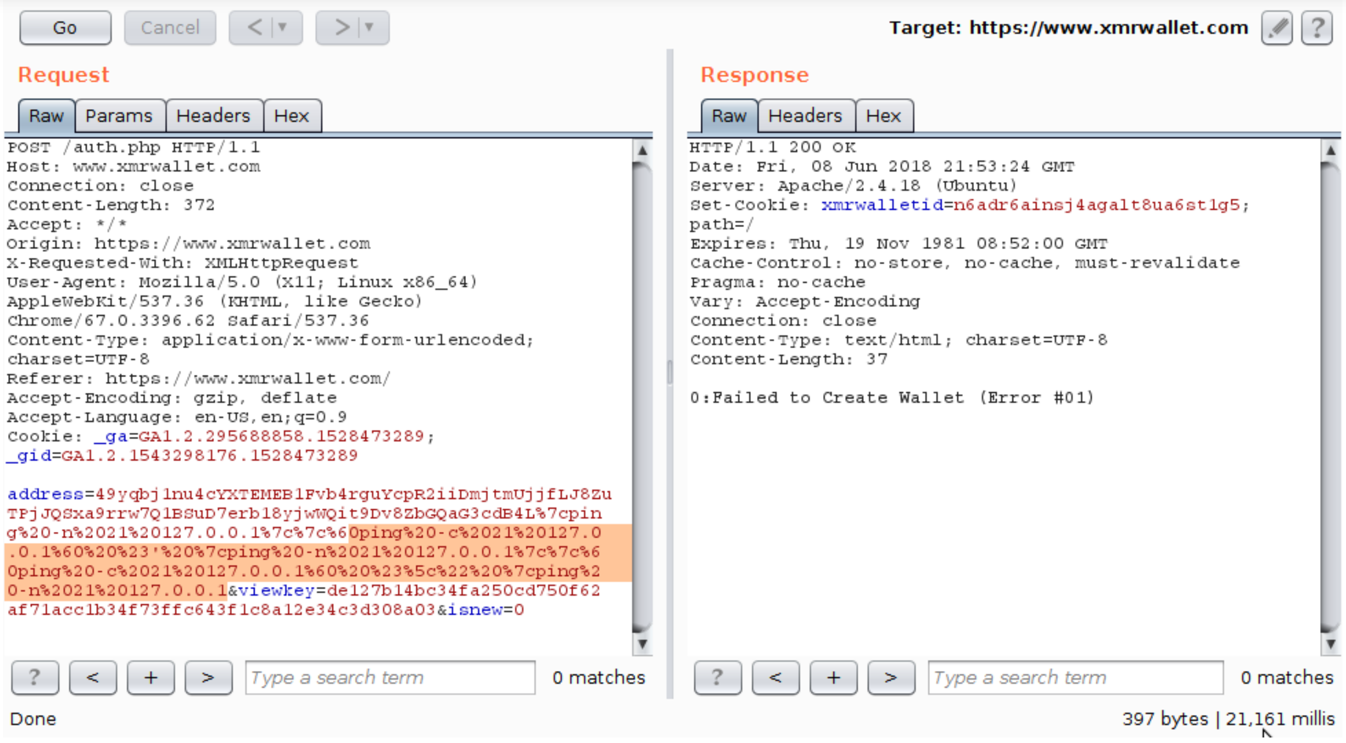
Description and Discussion

The address parameter of the POST `/auth.php` API endpoint appears to be vulnerable to OS command injection attacks. It is possible to use various shell metacharacters to inject arbitrary OS commands. The command output does not appear to be returned in the application's responses, however it is possible to inject time delay commands to verify the existence of the vulnerability. The payload

> |ping -n 21 127.0.0.1||`ping -c 21 127.0.0.1` #' |ping -n 21 127.0.0.1||`ping -c 21 127.0.0.1` #" |ping -n 21 127.0.0.1

involving 21 second delays was submitted in the address parameter. The application took 21161 milliseconds to respond to the request, compared with 234 milliseconds for the original request. This is a clear indication that the injected command caused a time delay.

Figure 4 below shows the specific request made to the server API endpoint on the left. The corresponding server response text (which is not significant) is shown on the right. In the lower right corner, the tool significantly indicates 21161mS of round-trip delay which is far outside of normal response time.



```
POST /auth.php HTTP/1.1
Host: www.xmrwallet.com
Connection: close
Content-Length: 372
Accept: */*
Origin: https://www.xmrwallet.com
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (X11; Linux x86_64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/67.0.3396.62 Safari/537.36
Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
Referer: https://www.xmrwallet.com/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: _ga=GA1.2.295688858.1528473289;
_gid=GA1.2.1543298176.1528473289

address=49yqbj1nu4cYXTEMEB1Fvb4rguYcpR2iiDmjtmUjjfLJ8Zu
TPjJQSxa9rrw7Q1BSuD7erb18yjwWQit9Dv8ZbGQaG3cdB4L%7cpin
g%20-n%2021%20127.0.0.1%7c%7c%60ping%20-c%2021%20127.0
.0.1%60%20%23'%20%7cping%20-n%2021%20127.0.0.1%7c%7c%6
0ping%20-c%2021%20127.0.0.1%60%20%23%5c%22%20%7cping%2
0-n%2021%20127.0.0.1&viewkey=de127b14bc34fa250cd750f62
af71acc1b34f73ffc643f1c8a12e34c3d308a03&isnew=0
```

```
HTTP/1.1 200 OK
Date: Fri, 08 Jun 2018 21:53:24 GMT
Server: Apache/2.4.18 (Ubuntu)
Set-Cookie: xmrwalletid=n6adr6ainsj4agalt8ua6st1g5;
path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Connection: close
Content-Type: text/html; charset=UTF-8
Content-Length: 37

0:Failed to Create Wallet (Error #01)
```

Figure 4: OS Command Injection

Remediation

If possible, applications should avoid incorporating user-controllable data into operating system commands. In almost every situation, there are safer alternative methods of performing server-level tasks, which cannot be manipulated to perform additional commands than the one intended.

If it is considered unavoidable to incorporate user-supplied data into operating system commands, the following two layers of defense should be used to prevent attacks:

- The user data should be strictly validated. Ideally, a whitelist of specific accepted values should be used. Otherwise, only short alphanumeric strings should be accepted. Input containing any other data, including any conceivable shell metacharacter or whitespace, should be rejected.
- The application should use command APIs that launch a specific process via its name and command-line parameters, rather than passing a command string to a shell interpreter that supports command chaining and redirection. For example, the Java API Runtime.exec and the ASP.NET API Process.Start do not support shell metacharacters. This defense can mitigate the impact of an attack even in the event

Re-test v2.0: New Alchemy has inspected the application and was unable to recreate this issue as described above. Thus, it is considered fixed and the associated risk mitigated.

5. Fixed: Insufficient Server-Side Session Expiration

Impact

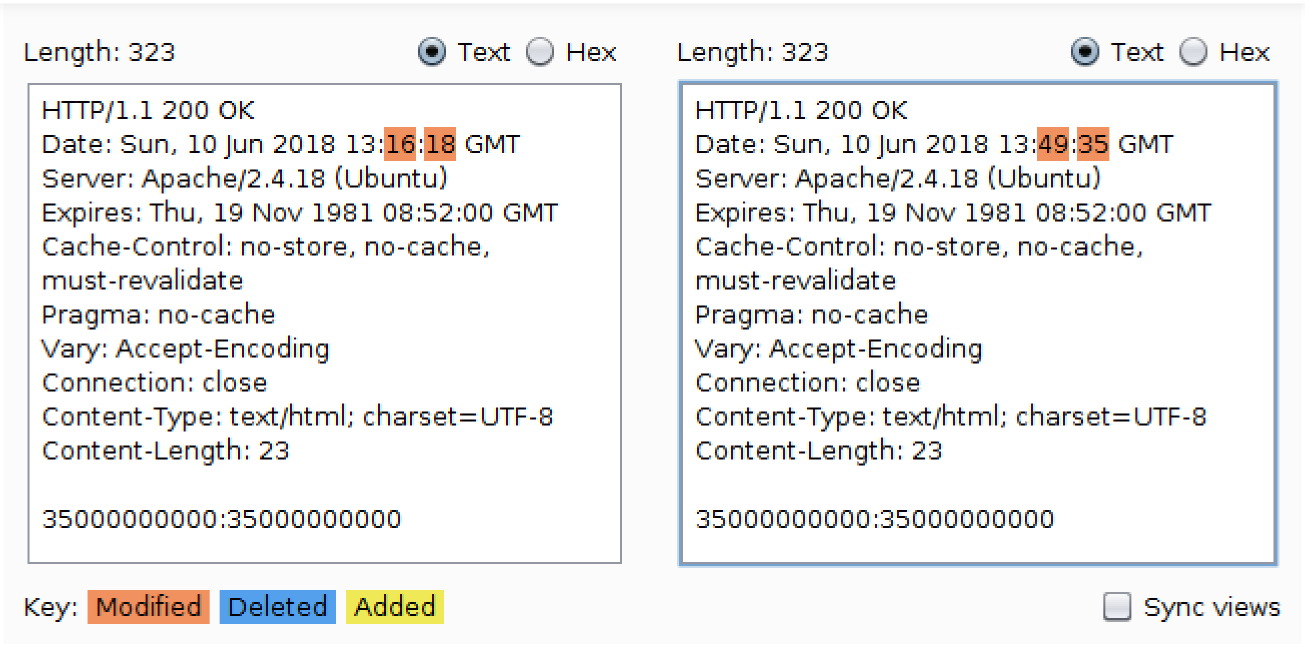
While the browser auto-expires sessions after 30 minutes, the server does not. Server expiration logic should be consistent with the client. An attacker may be able to utilize the incorrectly non-expired session credentials to operate the application and perform transactions as an impostor.

Description and Discussion

The browser appears to auto-expire sessions after 30 minutes, which is good. However, the primary responsibility for web application session management must reside on the server. The server must also auto-expire sessions and not purely rely on co-operative clients.

A browser session was initiated and the application exercised while using an intercepting proxy to record web traffic. The browser was then left idle until it auto-expired the session. The proxy was then used to successfully replay traffic with the original session credentials.

Figure 5 below shows the initial response from POST `/getbalance.php` on the left and a replayed version after browser session expiry on the right. The responses are identical with the exception of 33 minutes of elapsed idle time to test browser and server expiration logic consistency.

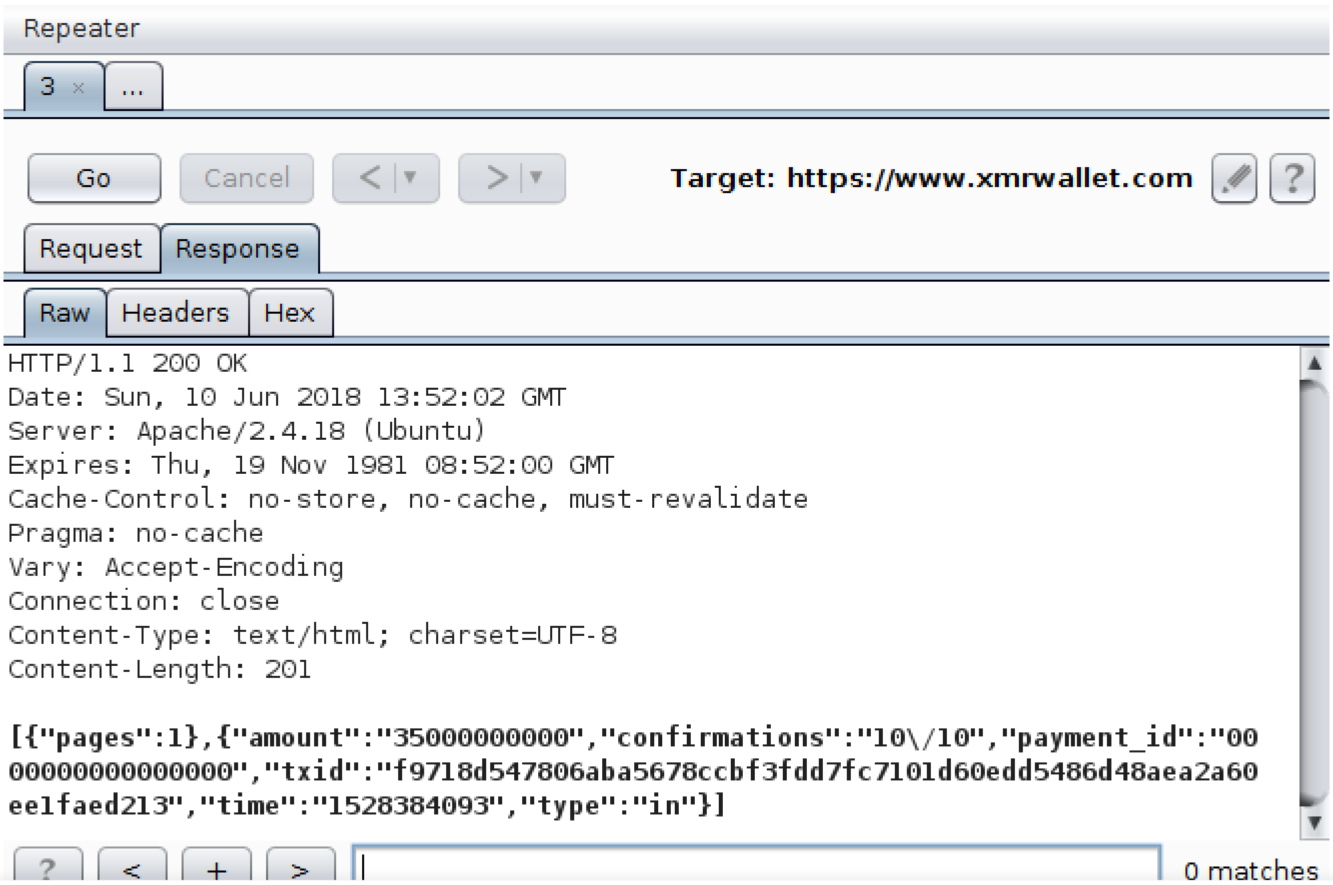


```
HTTP/1.1 200 OK
Date: Sun, 10 Jun 2018 13:16:18 GMT
Server: Apache/2.4.18 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache,
must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Connection: close
Content-Type: text/html; charset=UTF-8
Content-Length: 23

35000000000:35000000000
```

```
HTTP/1.1 200 OK
Date: Sun, 10 Jun 2018 13:49:35 GMT
Server: Apache/2.4.18 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache,
must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Connection: close
Content-Type: text/html; charset=UTF-8
Content-Length: 23

35000000000:35000000000
```

Figure 5: Non-Expired Session Balance

Figure 6 below shows a POST `/gettransactions.php` also replayed after session expiration. This leaks private transaction history details. No attempt was made to initiate malicious transactions due to project time constraints.



```
HTTP/1.1 200 OK
Date: Sun, 10 Jun 2018 13:52:02 GMT
Server: Apache/2.4.18 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Connection: close
Content-Type: text/html; charset=UTF-8
Content-Length: 201

[{"pages":1},{"amount":"35000000000","confirmations":"10\/10","payment_id":"00
00000000000000","txid":"f9718d547806aba5678ccbf3fdd7fc7101d60edd5486d48aea2a60
ee1faed213","time":"1528384093","type":"in"}]
```

Remediation

Ensure sessions are properly expired on the server in a time-frame corresponding to that of the browser.

Re-test v2.0: New Alchemy has inspected the application and determined this issue has been fixed and thus the associated risk mitigated. This was confirmed by initiating a valid session, waiting for the browser to timeout and then replaying prior requests to `gettransactions.php` and `getbalance.php` which did not return valid results.

6. Fixed: Missing Security Headers

Impact

The browser must receive sufficient and correctly configured headers in order to fully deploy all of its security mechanisms. Without these, the user will have increased exposure to cross-site scripting, malicious framing, caching of sensitive data and privacy leakage. This is a critical finding due to the number of missing headers and usefulness of leaking information to the XSS vulnerability in finding 2.

Description and Discussion

Multiple security-related headers are missing from server responses. They are listed below alongside a brief description and a link to full supporting documentation.

- Strict-Transport-Security This header supports confidentiality by helping to prevent HTTPS to HTTP protocol downgrade attacks.
- X-Frame-Options This header supports authenticity by preventing content from being presented inside of frames on other sites.
- X-XSS-Protection This header instructs the browser to inspect and analyze server responses to prevent reflected XSS.
- X-Content-Type-Options This header instructs the browser to 'make no guesses' on interpreting malformed content.
- Content-Security-Policy This header controls and constrains the resources allowed to load for a given page to prevent XSS.

- Cache-control: no-store (via login.html) This header instructs the browser not to cache sensitive content.
- Pragma: no-cache (via login.html) This header also instructs the browser not to cache sensitive content.

The server presents none of the above headers to the browser as can be seen in figure 7 below.

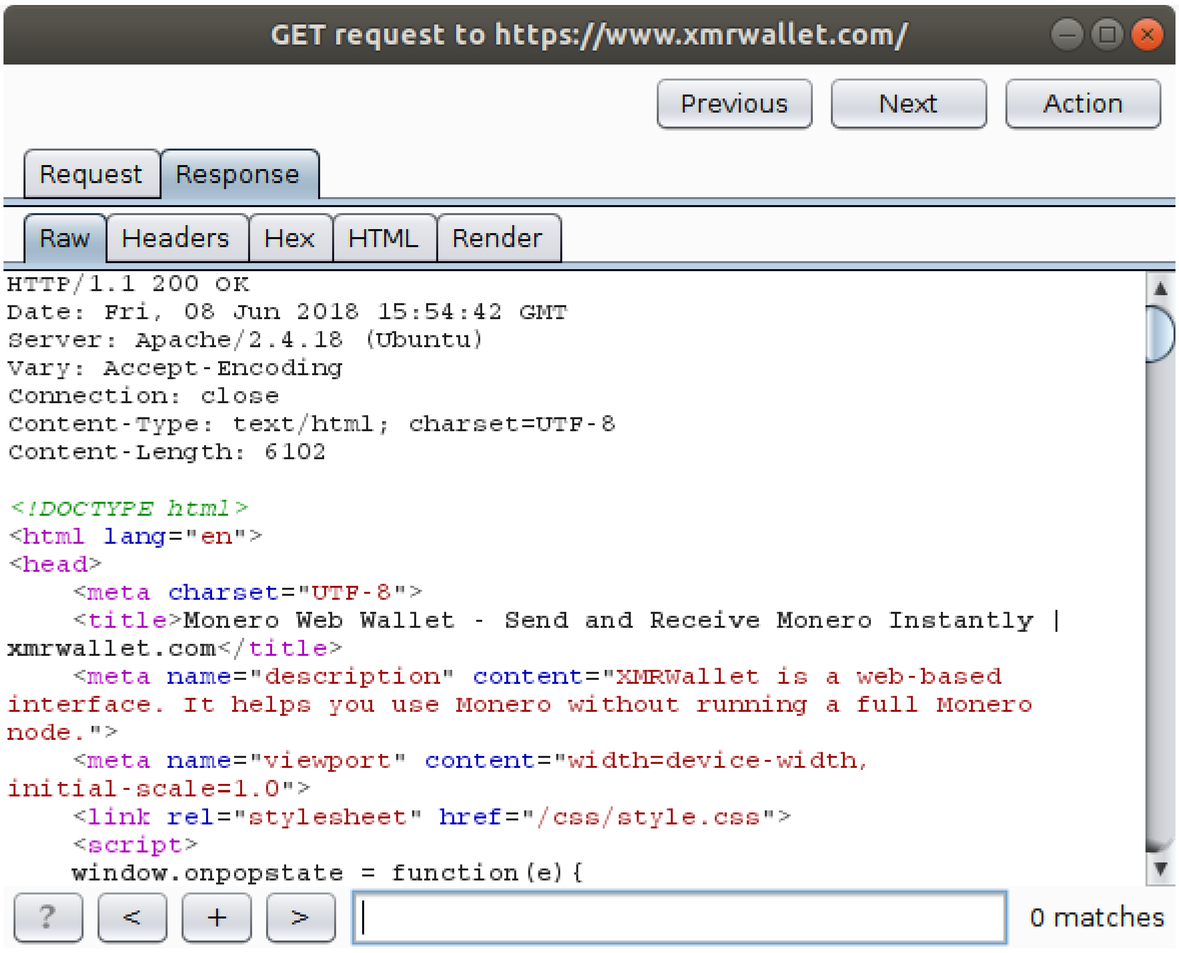
Note that the absence of the `ReferrerPolicy` header increases the potential for attack in the second finding (XSS) above. OWASP provides a useful reference at their Secure Headers Project.

Remediation

The server side application must be configured to provide these necessary headers. The links provided above provide solid context and specific guidance.

Re-test v2.0: New Alchemy has inspected the application and confirmed that the headers listed above are now present on the response to GET '/'. However these headers are missing from the vast majority of other endpoints, including: (POST) `/login.html`, `/send.html`, `/auth.php`, `/getbalance.php`, `/getprice.php` and `/gettransactions.php` among others.

Re-test v2.1: New Alchemy has inspected the application and confirmed that appropriate headers are now present on each category of response (as static assets, GET .html and POST .php endpoints have slightly different requirements). This issue is considered fixed and the risk mitigated.



```
HTTP/1.1 200 OK
Date: Fri, 08 Jun 2018 15:54:42 GMT
Server: Apache/2.4.18 (Ubuntu)
Vary: Accept-Encoding
Connection: close
Content-Type: text/html; charset=UTF-8
Content-Length: 6102

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Monero Web Wallet - Send and Receive Monero Instantly |
xmrwallet.com</title>
    <meta name="description" content="XMRWallet is a web-based
interface. It helps you use Monero without running a full Monero
node.">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <link rel="stylesheet" href="/css/style.css">
    <script>
    window.onpopstate = function(e){
```

Figure 7: Server response from GET request to https//:www.xmrwallet.com

7. Fixed: Lack of Randomness in Ring Signature Outputs

Impact

Reduced randomness in the selection of mix-in outputs may reduce the privacy and increase the traceability of Monero transactions.

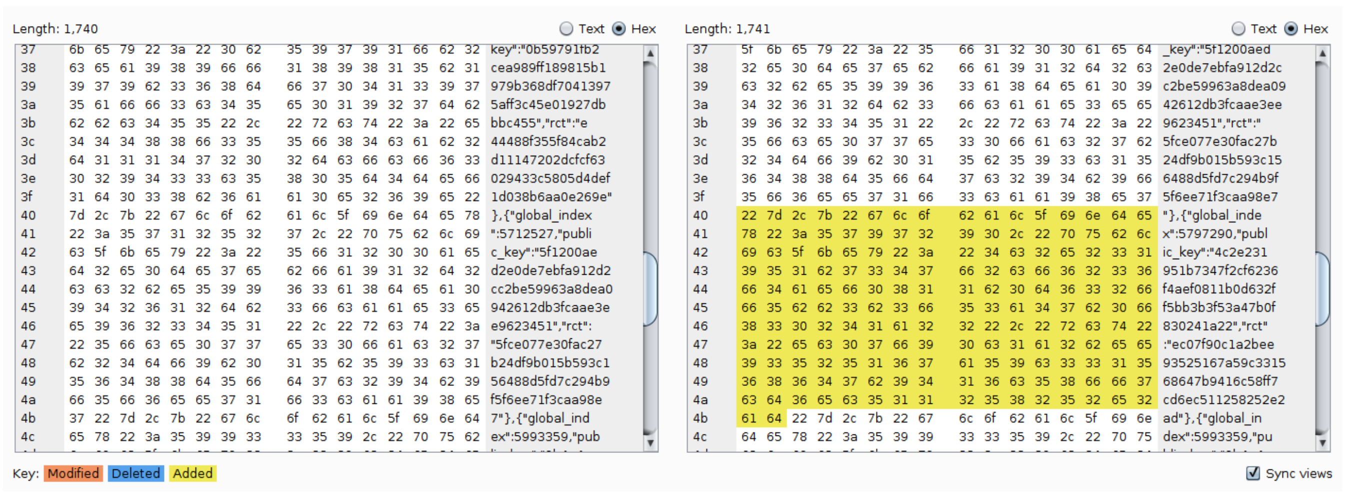
Description and Discussion

One of the primary advantages of the Monero cryptocurrency is improved privacy and reduced traceability. These advantages partially rely upon several transaction characteristics including the number of randomly chosen mix-in outputs. Inspecting the `XMRWallet` application web traffic indicates that eight mix-ins are utilized, which is a relatively strong number. However, their randomness is a concern.

Monero supports untraceability using a cryptographic primitive known as ring signatures. This allows a sender to anonymously sign a transaction on behalf of a set of

Repeated `POST` requests were made to `/getrandomoutputs.php` and the response contents inspected. Of the eight mix-in outputs returned, responses only differed by the insertion of one mix-in and the deletion of another.

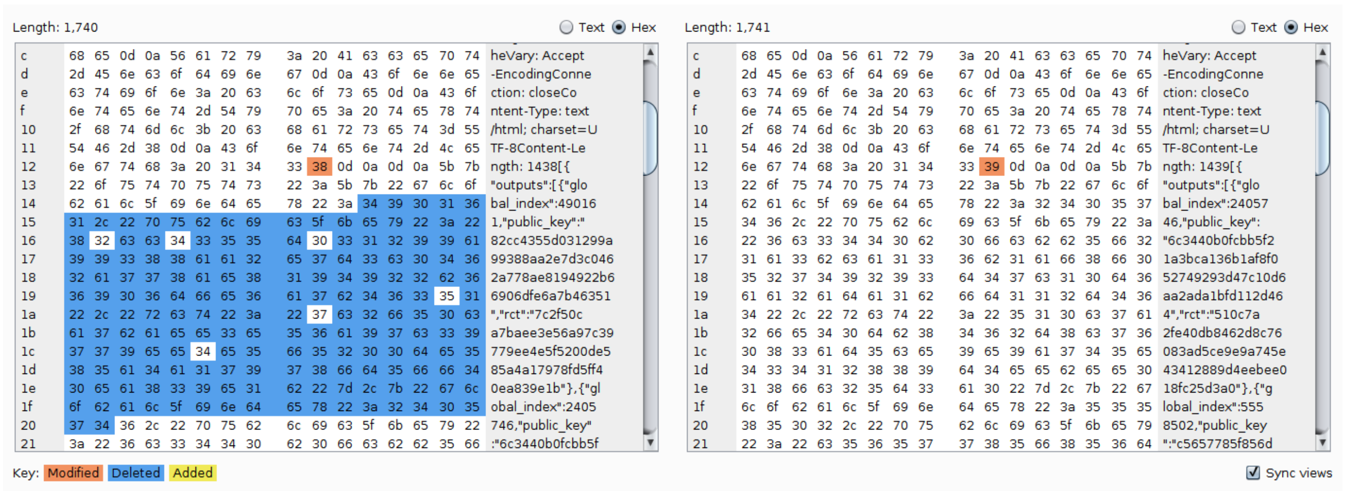
Figure 8 below shows two separate responses side by side. The yellow highlighted mix-in `public_key` has been inserted. Figure 9 below also shows the same two responses side by side. The blue highlighted mix-in `public_key` has been deleted. There are no other differences beyond timestamps and a 1-byte difference in content length.



```
Length: 1,740                                                          ○ Text ● Hex
37   6b 65 79 22 3a 22 30 62   35 39 37 39 31 66 62 32   key":"0b59791fb2
38   63 65 61 39 38 39 66 66   31 38 39 38 31 35 62 31   cea989ff189815b1
39   39 37 39 62 33 36 38 64   66 37 30 34 31 33 39 37   979b368df7041397
3a   35 61 66 66 33 63 34 35   65 30 31 39 32 37 64 62   5aff3c45e01927db
3b   62 62 63 34 35 35 22 2c   22 72 63 74 22 3a 22 65   bbc455","rct":"e
3c   34 34 34 38 38 66 33 35   35 66 38 34 63 61 62 32   44488f355f84cab2
3d   64 31 31 31 34 37 32 30   32 64 63 66 63 66 36 33   d11147202dcfcf63
3e   30 32 39 34 33 33 63 35   38 30 35 64 34 64 65 66   029433c5805d4def
3f   31 64 30 33 38 62 36 61   61 30 65 32 36 39 65 22   1d038b6aa0e269e"
40   7d 2c 7b 22 67 6c 6f 62   61 6c 5f 69 6e 64 65 78   },{"global_index
41   22 3a 35 37 31 32 35 32   37 2c 22 70 75 62 6c 69   ":5712527,"publi
42   63 5f 6b 65 79 22 3a 22   35 66 31 32 30 30 61 65   c_key":"5f1200ae
43   64 32 65 30 64 65 37 65   62 66 61 39 31 32 64 32   d2e0de7ebfa912d2
44   63 63 32 62 65 35 39 39   36 33 61 38 64 65 61 30   cc2be59963a8dea0
45   39 34 32 36 31 32 64 62   33 66 63 61 61 65 33 65   942612db3fcaae3e
46   65 39 36 32 33 34 35 31   22 2c 22 72 63 74 22 3a   e9623451","rct":
47   22 35 66 63 65 30 37 37   65 33 30 66 61 63 32 37   "5fce077e30fac27
48   62 32 34 64 66 39 62 30   31 35 62 35 39 33 63 31   b24df9b015b593c1
49   35 36 34 38 38 64 35 66   64 37 63 32 39 34 62 39   56488d5fd7c294b9
4a   66 35 66 36 65 65 37 31   66 33 63 61 61 39 38 65   f5f6ee71f3caa98e
4b   37 22 7d 2c 7b 22 67 6c   6f 62 61 6c 5f 69 6e 64   7"},{"global_ind
4c   65 78 22 3a 35 39 39 33   33 35 39 2c 22 70 75 62   ex":5993359,"pub
Key: Modified Deleted Added
```

```
Length: 1,741                                                          ○ Text ● Hex
37   5f 6b 65 79 22 3a 22 35   66 31 32 30 30 61 65 64   _key":"5f1200aed
38   32 65 30 64 65 37 65 62   66 61 39 31 32 64 32 63   2e0de7ebfa912d2c
39   63 32 62 65 35 39 39 36   33 61 38 64 65 61 30 39   c2be59963a8dea09
3a   34 32 36 31 32 64 62 33   66 63 61 61 65 33 65 65   42612db3fcaae3ee
3b   39 36 32 33 34 35 31 22   2c 22 72 63 74 22 3a 22   9623451","rct":"
3c   35 66 63 65 30 37 37 65   33 30 66 61 63 32 37 62   5fce077e30fac27b
3d   32 34 64 66 39 62 30 31   35 62 35 39 33 63 31 35   24df9b015b593c15
3e   36 34 38 38 64 35 66 64   37 63 32 39 34 62 39 66   6488d5fd7c294b9f
3f   35 66 36 65 65 37 31 66   33 63 61 61 39 38 65 37   5f6ee71f3caa98e7
40   22 7d 2c 7b 22 67 6c 6f   62 61 6c 5f 69 6e 64 65   "},{"global_inde
41   78 22 3a 35 37 39 37 32   39 30 2c 22 70 75 62 6c   x":5797290,"publ
42   69 63 5f 6b 65 79 22 3a   22 34 63 32 65 32 33 31   ic_key":"4c2e231
43   39 35 31 62 37 33 34 37   66 32 63 66 36 32 33 36   951b7347f2cf6236
44   66 34 61 65 66 30 38 31   31 62 30 64 36 33 32 66   f4aef0811b0d632f
45   66 35 62 62 33 62 33 66   35 33 61 34 37 62 30 66   f5bb3b3f53a47b0f
46   38 33 30 32 34 31 61 32   32 22 2c 22 72 63 74 22   830241a22","rct"
47   3a 22 65 63 30 37 66 39   30 63 31 61 32 62 65 65   :"ec07f90c1a2bee
48   39 33 35 32 35 31 36 37   61 35 39 63 33 33 31 35   93525167a59c3315
49   36 38 36 34 37 62 39 34   31 36 63 35 38 66 66 37   68647b9416c58ff7
4a   63 64 36 65 63 35 31 31   32 35 38 32 35 32 65 32   cd6ec511258252e2
4b   61 64 22 7d 2c 7b 22 67   6c 6f 62 61 6c 5f 69 6e   ad"},{"global_in
4c   64 65 78 22 3a 35 39 39   33 33 35 39 2c 22 70 75   dex":5993359,"pu
                                                                       ☑ Sync views
```

Figure 8: Two Random Output Responses; A Single public_key Addition



```
Length: 1,740                                                          ○ Text ● Hex
c    68 65 0d 0a 56 61 72 79   3a 20 41 63 63 65 70 74   heVary: Accept
d    2d 45 6e 63 6f 64 69 6e   67 0d 0a 43 6f 6e 6e 65   -EncodingConne
e    63 74 69 6f 6e 3a 20 63   6c 6f 73 65 0d 0a 43 6f   ction: closeCo
f    6e 74 65 6e 74 2d 54 79   70 65 3a 20 74 65 78 74   ntent-Type: text
10   2f 68 74 6d 6c 3b 20 63   68 61 72 73 65 74 3d 55   /html; charset=U
11   54 46 2d 38 0d 0a 43 6f   6e 74 65 6e 74 2d 4c 65   TF-8Content-Le
12   6e 67 74 68 3a 20 31 34   33 38 0d 0a 0d 0a 5b 7b   ngth: 1438[{
13   22 6f 75 74 70 75 74 73   22 3a 5b 7b 22 67 6c 6f   "outputs":[{"glo
14   62 61 6c 5f 69 6e 64 65   78 22 3a 34 39 30 31 36   bal_index":49016
15   31 2c 22 70 75 62 6c 69   63 5f 6b 65 79 22 3a 22   1,"public_key":"
16   38 32 63 63 34 33 35 35   64 30 33 31 32 39 39 61   82cc4355d031299a
17   39 39 33 38 38 61 61 32   65 37 64 33 63 30 34 36   99388aa2e7d3c046
18   32 61 37 37 38 61 65 38   31 39 34 39 32 32 62 36   2a778ae8194922b6
19   36 39 30 36 64 66 65 36   61 37 62 34 36 33 35 31   6906dfe6a7b46351
1a   22 2c 22 72 63 74 22 3a   22 37 63 32 66 35 30 63   ","rct":"7c2f50c
1b   61 37 62 61 65 65 33 65   35 36 61 39 37 63 33 39   a7baee3e56a97c39
1c   37 37 39 65 65 34 65 35   66 35 32 30 30 64 65 35   779ee4e5f5200de5
1d   38 35 61 34 61 31 37 39   37 38 66 64 35 66 66 34   85a4a17978fd5ff4
1e   30 65 61 38 33 39 65 31   62 22 7d 2c 7b 22 67 6c   0ea839e1b"},{"gl
1f   6f 62 61 6c 5f 69 6e 64   65 78 22 3a 32 34 30 35   obal_index":2405
20   37 34 36 2c 22 70 75 62   6c 69 63 5f 6b 65 79 22   746,"public_key"
21   3a 22 36 63 33 34 34 30   62 30 66 63 62 62 35 66   :"6c3440b0fcbb5f
Key: Modified Deleted Added
```

```
Length: 1,741                                                          ○ Text ● Hex
c    68 65 0d 0a 56 61 72 79   3a 20 41 63 63 65 70 74   heVary: Accept
d    2d 45 6e 63 6f 64 69 6e   67 0d 0a 43 6f 6e 6e 65   -EncodingConne
e    63 74 69 6f 6e 3a 20 63   6c 6f 73 65 0d 0a 43 6f   ction: closeCo
f    6e 74 65 6e 74 2d 54 79   70 65 3a 20 74 65 78 74   ntent-Type: text
10   2f 68 74 6d 6c 3b 20 63   68 61 72 73 65 74 3d 55   /html; charset=U
11   54 46 2d 38 0d 0a 43 6f   6e 74 65 6e 74 2d 4c 65   TF-8Content-Le
12   6e 67 74 68 3a 20 31 34   33 39 0d 0a 0d 0a 5b 7b   ngth: 1439[{
13   22 6f 75 74 70 75 74 73   22 3a 5b 7b 22 67 6c 6f   "outputs":[{"glo
14   62 61 6c 5f 69 6e 64 65   78 22 3a 32 34 30 35 37   bal_index":24057
15   34 36 2c 22 70 75 62 6c   69 63 5f 6b 65 79 22 3a   46,"public_key":
16   22 36 63 33 34 34 30 62   30 66 63 62 62 35 66 32   "6c3440b0fcbb5f2
17   31 61 33 62 63 61 31 33   36 62 31 61 66 38 66 30   1a3bca136b1af8f0
18   35 32 37 34 39 32 39 33   64 34 37 63 31 30 64 36   52749293d47c10d6
19   61 61 32 61 64 61 31 62   66 64 31 31 32 64 34 36   aa2ada1bfd112d46
1a   34 22 2c 22 72 63 74 22   3a 22 35 31 30 63 37 61   4","rct":"510c7a
1b   32 66 65 34 30 64 62 38   34 36 32 64 38 63 37 36   2fe40db8462d8c76
1c   30 38 33 61 64 35 63 65   39 65 39 61 37 34 35 65   083ad5ce9e9a745e
1d   34 33 34 31 32 38 38 39   64 34 65 65 62 65 65 30   43412889d4eebee0
1e   31 38 66 63 32 35 64 33   61 30 22 7d 2c 7b 22 67   18fc25d3a0"},{"g
1f   6c 6f 62 61 6c 5f 69 6e   64 65 78 22 3a 35 35 35   lobal_index":555
20   38 35 30 32 2c 22 70 75   62 6c 69 63 5f 6b 65 79   8502,"public_key
21   22 3a 22 63 35 36 35 37   37 38 35 66 38 35 36 64   ":"c5657785f856d
                                                                       ☑ Sync views
```

Figure 9: Two Random Output Responses; A Single public_key Deletion

Remediation

Server side functionality should be adjusted to increase the randomness of output mixcapturing a legitimate transaction to `/getrandomoutputs.php` , replaying it twice and confirming significant differences between server responses.

Moderate Issues

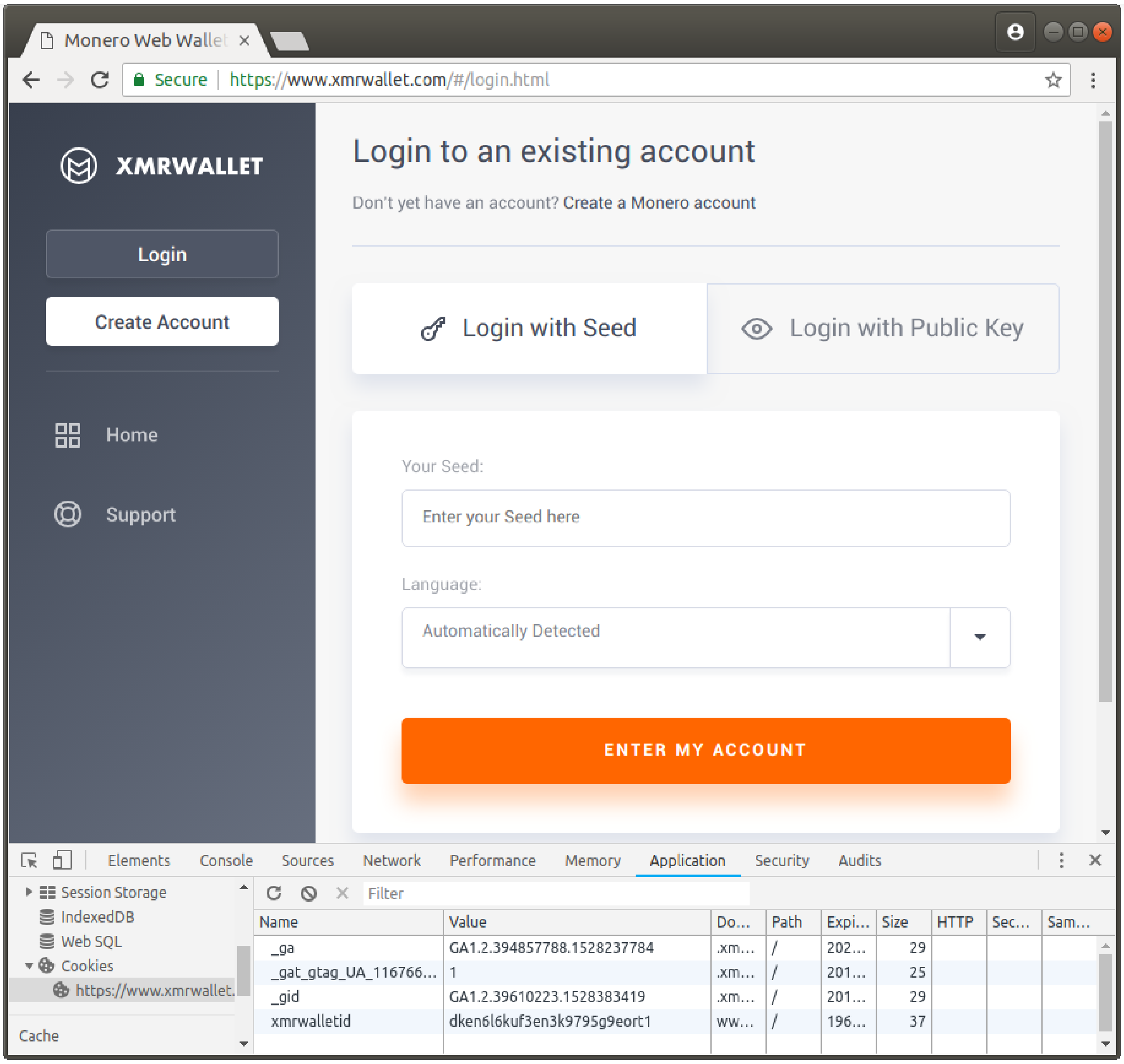
8. Fixed: Cookies not Destroyed Upon Logout

Impact

An attacker can access sensitive cookie/session information after the user has logged out. This information can be very useful for subsequent attacks.

Description and Discussion

When the user clicks 'Quit' to logout, all cookies should be destroyed. As shown below, in practice the `xmrwalletid` , `_ga` , `_gat_gtag_UA_116766241_1` and `_gid` remain after log out.



Name	Value	Do…	Path	Expi…	Size	HTTP	Sec…	Sam…
_ga	GA1.2.394857788.1528237784	.xm…	/	202…	29			
_gat_gtag_UA_116766…	1	.xm…	/	201…	25			
_gid	GA1.2.39610223.1528383419	.xm…	/	201…	29			
xmrwalletid	dken6l6kuf3en3k9795g9eort1	ww…	/	196…	37			

Figure 10: Cookies Remain After Logout

Remediation

The server should respond to the `/logout.php` POST request with a `Set Cookie: xmrwalletid=; path=/; expires=Thu, 01 Jan 1970 00:00:00 GMT` header.

Re-test v2.0: XMRWallet has correctly indicated that the `_ga`, `_gat_gtag_UA_116766241_1` and `_gid` cookies are extraneous to the application functionality and it is unecessary to destroy them at logout. However, the `xmrwalletid` cookie is relevant and should be destroyed at logout. The original report

Re-test v2.1: XMRWallet has indicated that the `xmrwalletid` cookie is part of a desired 'remember-me' function and that the actual session id is destroyed in the browser at logout. New Alchemy has confirmed that the session id is present on the `POST /logout.php` request and then absent on the immediately subsequent `POST login.html` request. This issue is considered fixed and the risk mitigated.

9. Fixed: Non-Obfuscated Display of Private Fields

Impact

An attacker may 'shoulder surf' a victim to obtain private information for subsequent malicious use.

Description and Discussion

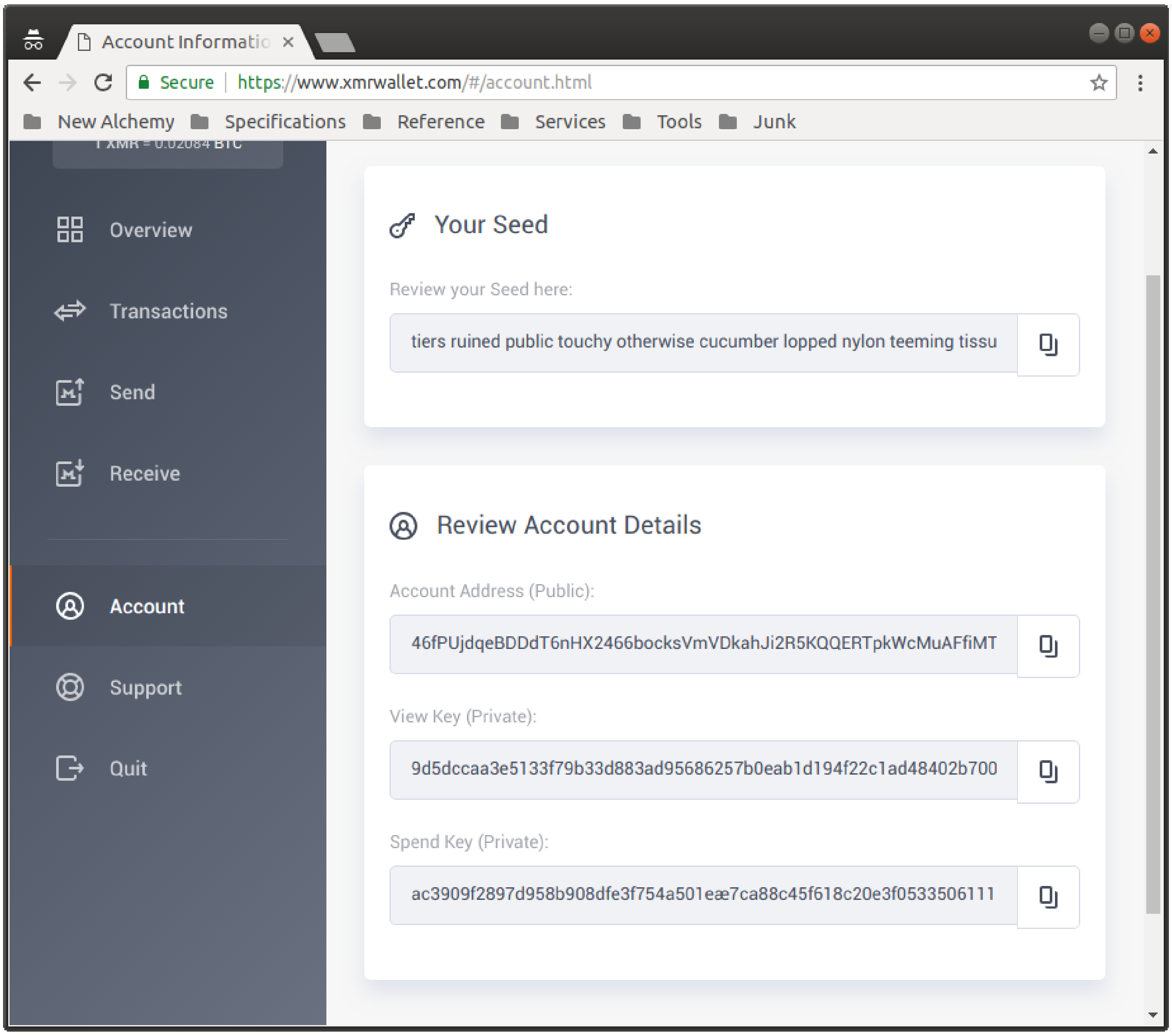
The `/account.html` page shows the user's full account details in a single location. Unfortunately, the private view key and private spend key are clearly shown by default and not initially obfuscated by default. This is shown in figure 11 below.

Remediation

Use the `password` attribute on private form fields. Mozilla has a nice input and password reference that provides additional background and insight.

Re-test v2.0: New Alchemy has inspected the application and was able to recreate this issue as described above. Thus, it is not yet fixed.

Re-test v2.1: New Alchemy has inspected the application and determined this issue has been fixed and thus the associated risk mitigated.



Your Seed

Review your Seed here:

tiers ruined public touchy otherwise cucumber lopped nylon teeming tissu

Review Account Details

Account Address (Public):

46fPUjdqeBDDdT6nHX2466bocksVmVDkahJi2R5KQQERTpkWcMuAFfiMT

View Key (Private):

9d5dccaa3e5133f79b33d883ad95686257b0eab1d194f22c1ad48402b700

Spend Key (Private):

ac3909f2897d958b908dfe3f754a501eæ7ca88c45f618c20e3f0533506111

Figure 11: Non-Obfuscated Private Fields

10. Fixed: Missing Cookie Attributes

Impact

Cookies often support session handling and can thus represent very sensitive user credentials. Improper attributes can allow the browser to send cookies over unencrypted channels, provide cookie access to malicious JavaScript running in the browser, and/or send them to an incorrect website. An attacker with user credentials is able to operate the application as an impostor.

Description and Discussion

allow an attacker to perform a Session Hijacking attack. This would generally require the ability to observe and/or modify the traffic via a man-in-the-middle attack.

The HttpOnly attribute should be set as true to inform browsers that the session cookie should not be accessible via client-side JavaScript. If the attribute is set to false or missing, client-side JavaScript can read or set the cookie. If an application includes malicious third-party JavaScript or has a cross-site scripting vulnerability, the application could then disclose users' session cookies to undesired parties or perform a Session Fixation attack. If an attacker is able to gain access to a user's session cookie, the attacker could gain persistent access to that account.

The newly emerging Same Site attribute allows the server to advise the browser that cookies should only be sent if the request originates from the same web-site the cookie came from. Requests triggered from a URL different than the one that appears in the URL bar will not include any of the cookies tagged with this new attribute. This is primarily aimed at helping defend against cross-site request forgery (CSRF) attacks and is not yet broadly supported. The session cookie should utilize the Strict setting.

Remediation

The `xmrwalletid` cookie sent from the server does not include the `secure` flag, the `HttpOnly` attribute, nor the `SameSite` attribute. These should be added. In some cases the application may rely upon JavaScript having access to cookies - in these cases the `HttpOnly` attribute can be omitted.

Re-test v2.0: New Alchemy has inspected the application and was able to recreate this issue as described above. Thus, it is not yet fixed.

Re-test v2.1: New Alchemy has inspected the application for the presence of `secure` and `httpOnly` attributes, and determined this issue has been fixed and the associated risk mitigated. While the Same Site attribute is still not present, its absence is not considered critical due to its very limited browser support.

11. Informational: Potentially Unsafe JavaScript/HTML usage

Description

There are three aspects to this finding.

Extensive use of JavaScript `==` rather than `===` . There are approximately 455 instances of the JavaScript `==` abstract equality comparator in the code. While visually and conceptually similar, the `==` abstract comparator tests for loose equality which involves an automatic type coercion process. The `===` strict equality comparator is preferred as it uses simple strict equality without type coercion. This forces the developer to clearly anticipate types a priori and surfaces bugs early.

Twig Autoescaping. All data inserted into the DOM should be HTML escaped to prevent attacks including XSS. `Twig` provides excellent support for the autoescaping process that relieves the developer from the minutia. No evidence of its use was observed.

Use of innerHTML. Grepping the code reveals multiple instances of `innerHTML` use. This is dangerous as HTML elements are evaluated, for example the text `blah blah <img src=x onerror=alert(1)> blah blah` can insert XSS JavaScript via the `onerror` attribute. See this Mozilla Article. The critical XSS finding 2 was injected via this exact route.

Remediation

1. Review all usages of '==' and replace them with '===' where possible.
2. Utilize Twig autoescaping functionality to the maximum extent possible.
3. Replace instances of `innerHTML` with the safer `node.textContent`

Re-test v2.0: New Alchemy has inspected the application and was able to recreate this issue as described above. Specifically a `grep ==` run against the twig files returns significant usage. The same operation for `innerHTML` also returns several instances including lines relevant to critical issue #2 (XSS). Thus, it is not yet fixed.

Re-test v2.1: After further discussion with XMRWallet, New Alchemy agrees that this issue does not identify a current and specific security vulnerability. As it does highlight practices counter to industry norms, which in fact led to the existence of critical issue

Impact

Session ID tokens that are too short (or not sufficiently random) may enable an attacker to make educated guesses, utilize brute force techniques or employ a combination of the two. If session IDs are discovered, they may be used to impersonate a legitimate user.

Description and Discussion

Over 2000 samples of the 13-character session ID value returned in the POST `/auth.php` response body were extracted then analyzed. The effective entropy was calculated to be approximately 50 bits at a significance level of 1%. Note that a 13-digit hexadecimal number corresponds to 52-bit number so any more entropy is unobtainable with this length. Best practice requires 128 bits of entropy which will require lengthening this field.

FYI, A similar analysis was performed on the `xmrwalletid` cookie and entropy approaches 128 bits - excellent.

It is unclear how the two session IDs are utilized over the life-cycle of the application. This finding is written out of an abundance of caution.

Remediation

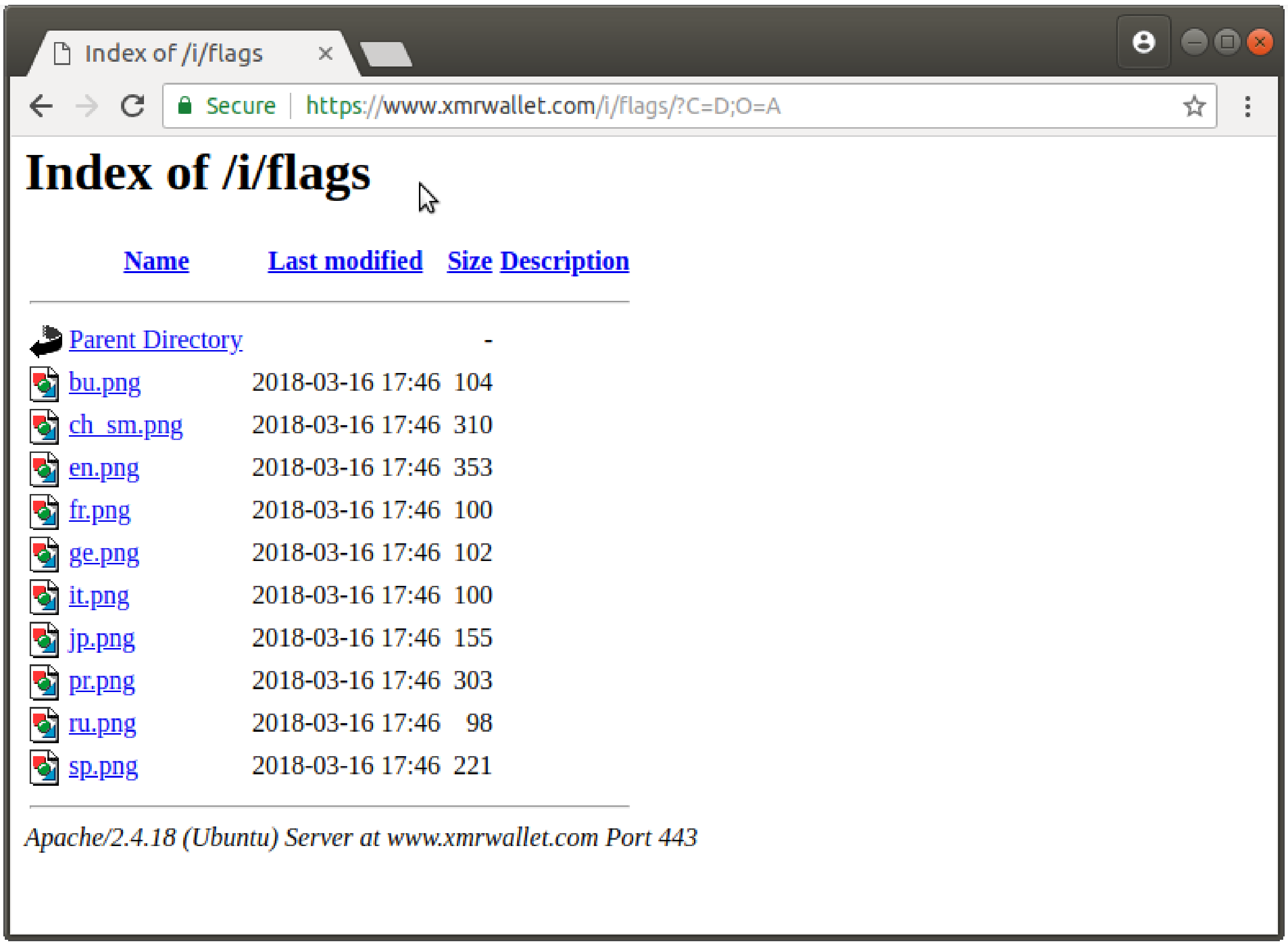
Increase the length of all session ID tokens while maintaining maximum entropy. Ensure all tokens are generated via a cryptographically secure random number generator.

Re-test v2.0: Subsequent feedback indicates this finding was overly cautious as suggested above. New Alchemy considers this issue fixed and the associated risk mitigated.

13. Fixed: Server Discloses Directory Listing

Impact

Attackers can inspect the contents of the application filesystem, potentially finding secrets or useful information for a subsequent attack.



Index of /i/flags

```
     Name            Last modified      Size Description
----------------------------------------------------------
 Parent Directory                          -
 bu.png             2018-03-16 17:46  104
 ch_sm.png          2018-03-16 17:46  310
 en.png             2018-03-16 17:46  353
 fr.png             2018-03-16 17:46  100
 ge.png             2018-03-16 17:46  102
 it.png             2018-03-16 17:46  100
 jp.png             2018-03-16 17:46  155
 pr.png             2018-03-16 17:46  303
 ru.png             2018-03-16 17:46   98
 sp.png             2018-03-16 17:46  221
----------------------------------------------------------
Apache/2.4.18 (Ubuntu) Server at www.xmrwallet.com Port 443
```

Figure 12: Directory Contents

Remediation

Correct the Apache webserver configuration to eliminate all listing of directory contents.

Re-test v2.0: New Alchemy has inspected the application and has determined that this issue has been fixed and the associated risk mitigated.

Minor Issues

14. Partially-Fixed: Server Discloses Sensitive Information

Impact

An attacker may be able to use information revealed by the system to plan further attacks or as part of a social engineering attack.

credentials. Default content and pages also aid in the use of Google searches (or dorks) to identify vulnerable systems for specific exploits. The server unnecessarily discloses internal technical information that may be of subsequent use to an attacker.

The application notes that it is built upon Apache v2.4.18 in both the response header and body shown below.

```
HTTP/1.1 301 Moved Permanently
   Date: Thu, 07 Jun 2018 13:02:09 GMT
   Server: Apache/2.4.18 (Ubuntu)
   Location: https://www.xmrwallet.com/
   Content-Length: 317
   Connection: close
   Content-Type: text/html; charset=iso-8859-1

   <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
   <html><head>
   <title>301 Moved Permanently</title>
   </head><body>
   <h1>Moved Permanently</h1>
   <p>The document has moved <a
href="https://www.xmrwallet.com/">here</a>.</p>
   <hr>
   <address>Apache/2.4.18 (Ubuntu) Server at www.xmrwallet.com Port
80</address>
   </body></html>
```

Further, note that application API endpoints end with the .php suffix, which indicates PHP server technologies.

Remediation

Remove the Apache header. Remove the .php suffix from the API endpoint.

Re-test v2.0: New Alchemy has inspected the application and observed that while server header no longer identifies the specific version of Apache, it still indicates Apache. The `php` suffixes remain on the API endpoints. Thus, this issue is not yet fully fixed.

Re-test v2.1: No change from prior status. New Alchemy believes the removal of the

15. Fixed: Outdated server dependencies

Impact

Outdated dependencies may expose the application to publicly known vulnerabilities.

Description and Discussion

Apache version 2.4.18 (as noted in finding 12) was released in December 2015 and is exposed to a large number (15) of vulnerabilities. If the server were in scope, these would be investigated further and likely result in an elevation of finding criticality.

Remediation

Extract the definitive list of deployed server-side and client-side dependencies, review against recommended production versions, and incorporate the correct versions into the build process. Incorporate a periodic review of updates into the development schedule that gates delivery of code into production.

Re-test v2.0: New Alchemy has inspected the application and observed that the server header no longer identifies the specific version of Apache. Thus, this issue is fixed and the associated risk mitigated.

16. Informational: Lack of Attack Detection

Impact

The inability to detect application attacks delays service owners from realizing and responding to malicious activity. This increases exposure to breach size, user impact, remediation cost and reputational risk.

Description and Discussion

Attackers generally begin with endpoint scanning, probing and fuzzing, which can be a source of a large amount of unsuccessful requests. Lack of monitoring and timely response allows attackers to avoid detection while perfecting their techniques. The average time to breach detection in 2016 was 191 days. This allows attackers to refine their approach and ultimately break in.

While generating 2000 new sessions for entropy analysis, it was noted that the

Remediation

Include functionality to log unsuccessful endpoint calls, calculate running statistics, trigger alert thresholds and send this machine readable information to a centralized log management system. Also, support polling of this same information as part of the health check endpoint.

Re-test v2.0: New Alchemy has inspected the application and was able to recreate this issue as described above. Thus, it is not yet fixed. Note that this type of functionality is often considered on a medium-term roadmap and XMRWallet would not stand out from competitors while this functionality is absent.

Re-test v2.1: After further discussion with XMRWallet, New Alchemy agrees that this issue does not identify a current and specific security vulnerability. As it does highlight desirable future functionality, it has been retained under an 'Informational' heading.

17. Fixed: Server Responds to Extraneous Endpoints

Impact

This may be indicative of misconfiguration or server-side application bugs.

Description and Discussion

The server responds with code 200 to GET requests made to the following endpoints. This is unusual. If the server responds to other more sensitive URLs in an unexpected fashion, the application may be exposed to malicious attacks.

- www.xmrwallet.com/css/index.php
- www.xmrwallet.com/font/index.php
- www.xmrwallet.com/i/index.php
- www.xmrwallet.com/js/index.php

Remediation

Revisit the server-side application routing logic to ensure only the minimal correct endpoints are served. 404 the remainder.

Re-test v2.1: In further discussion, XMRWallet has elaborated the intentional reason behind the server responses to the items listed. New Alchemy considers this issue fixed and the risk mitigated.

18. Informational: Potentially Extraneous/Misconfigured Cookie

Impact

This may be indicative of misconfiguration or server-side application bugs.

Description and Discussion

The application is generating `_ga`, `_gat_gtag_UA_116766241_1` and `_gid` cookies as shown at the bottom of figure 10 of finding 8. These cookies appear be the correct format for Google Tags and the application does indeed pull from `https://www.googletagmanager.com/gtag/js?id=UA-116766241-1`. However the cookies are returned to the application rather than being sent to Google. This is very unusual.

Remediation

Review the usage of Google Tag Manager and ensure correct configuration. Note that the inclusion of 3rd party JavaScript may impact the perception of trustworthiness. The Google Tag Manager 'Custom HTML' method allows the addition of anything to your client — JavaScript, HTML and CSS included. It is possible an attacker may choose this route to insert malicious JavaScript.

Re-test v2.0: New Alchemy has inspected the application and was able to recreate this issue as described above. Thus, it is not yet fixed. *Note that this is essentially an informative issue.*

Re-test v2.1: XMRWallet has indicated this issue to be an accepted risk. It has been marked as informational per the v2.0 statement above.

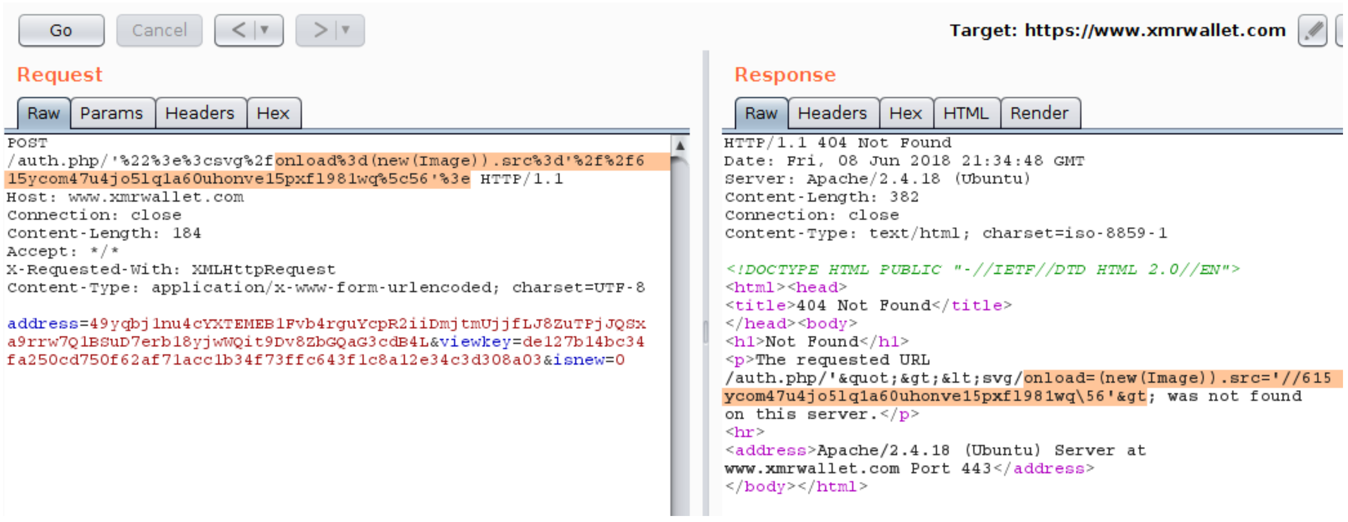
19. Fixed: URL Reflection

Impact

Reflection of input arises when data is copied from a request and echoed into the application's immediate response verbatim.

Input being returned in application responses is not a vulnerability in its own right. However, it is a prerequisite for many client-side vulnerabilities, including cross-site scripting, open redirection, content spoofing, and response header injection. Additionally, some server-side vulnerabilities such as SQL injection are often easier to identify and exploit when input is returned in responses. In applications where input retrieval is rare and the environment is resistant to automated testing (for example, due to a web application firewall), it might be worth subjecting instances of it to focused manual testing.

As shown in figure 10 below, the 404 page appears to be reflecting the URL request verbatim.



```
POST
/auth.php/'%22%3e%3csvg%2fonload%3d(new(Image)).src%3d'%2f%2f6
15ycom47u4jo51q1a60uhonve15pxf1981wq%5c56'%3e HTTP/1.1
Host: www.xmrwallet.com
Connection: close
Content-Length: 184
Accept: */*
X-Requested-With: XMLHttpRequest
Content-Type: application/x-www-form-urlencoded; charset=UTF-8

address=49yqbj1nu4cYXTEMEB1Fvb4rguYcpR2iiDmjtmUjjfLJ8ZuTPjJQSx
a9rrw7Q1BSuD7erb18yjwWQit9Dv8ZbGQaG3cdB4L&viewkey=de127b14bc34
fa250cd750f62af71acc1b34f73ffc643f1c8a12e34c3d308a03&isnew=0
```

```
HTTP/1.1 404 Not Found
Date: Fri, 08 Jun 2018 21:34:48 GMT
Server: Apache/2.4.18 (Ubuntu)
Content-Length: 382
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL
/auth.php/'&quot;&gt;&lt;svg/onload=(new(Image)).src='//615
ycom47u4jo51q1a60uhonve15pxf1981wq\56'&gt; was not found
on this server.</p>
<hr>
<address>Apache/2.4.18 (Ubuntu) Server at
www.xmrwallet.com Port 443</address>
</body></html>
```

Figure 13: Request Reflected in Response

Remediation

Scrub all external input. Use the Twig autoescaping functionality.

Re-test v2.0: New Alchemy has inspected the application and was able to recreate this issue as described above. Thus, it is not yet fixed.

Re-test v2.1: XMRWallet has added a custom 404 page to remediate this issue. New Alchemy considers this issue fixed and the risk mitigated.

Limitation and Disclaimer

The review makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about fitness of the application to purpose, or their bug-free status. The review documentation is for discussion purposes only.

The server-side API code, obfuscated client code and cryptography was out of scope.

Open in app Get started

New Alchemy · Jun 21, 2018

AUX Platform Security Audit

Ethereum 10 min read

New Alchemy · Jun 16, 2018

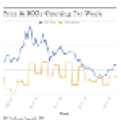
Ovcode Smart Contract Audit

Ethereum 6 min read

Alexander Behrens · Jun 6, 2018

ICO Market Update: How Price Action Affects Market Sentiment

Blockchain 7 min read



New Alchemy · May 22, 2018

The Rouge Project Smart Contract Audit

Ethereum 9 min read

Read more from New Alchemy