Fine penetration tests for fine websites

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Pentest-Report GreatFire FreeBrowser Android App & Library 03.-04.2022

Cure53, Dr.-Ing. M. Heiderich, Dipl.-Ing. A. Inführ

Index

Introduction

> *“We are an anonymous organization based in China. We launched our first project in 2011 in an effort to help bring transparency to online censorship in China. Now we focus on helping Chinese to freely access information. Apart from being widely discussed in most major mass media, GreatFire has also been the subject of a number of academic papers from various research institutions. FreeWeibo.com won the 2013 Deutsche Welle “Best Of Online Activism” award in the “Best Innovation” category. In 2016, GreatFire won a Digital Activism fellowship from Index on Censorship.”*
>
> From https://en.greatfire.org/

This report - entitled GF-04 - details the scope, results, and conclusory summaries of a concise penetration test and source code audit against the GreatFire FreeBrowser, with a particular focus on the Android library and web application. The work was requested by GreatFire.org in December 2021 and initiated by Cure53 in late March and early April 2022, namely in CW13. A total of four days were invested to reach the coverage expected for this project.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

The work comprised one sole work package (WP), which reads as follows:

- **WP1**: White-box tests and security audits against GreatFire FreeBrowser

Even though this test constitutes the fourth collaborative engagement between GreatFire and Cure53, the FreeBrowser itself has not been allocated as a scope item on any of the previous occasions. This audit, therefore, marks the inaugural assessment against the FreeBrowser by Cure53. Cure53 was supplied with the mobile application sources, APK, pertinent documentation, and any alternative means of access required to complete testing. For these purposes, the methodology chosen was white-box, and a team of two senior testers was assigned to the project's preparation, testing, audit execution, and finalization.

All preparatory actions were completed in March 2022, namely in CW12, to ensure that the testing phase could proceed without hindrance. Communications were facilitated via a dedicated Signal channel deployed to combine the workspaces of GreatFire and Cure53, thereby allowing an optimal collaborative working environment to flourish. All participatory personnel from both parties were invited to partake throughout the test preparations and discussions.

One can denote that communications proceeded smoothly on the whole. The scope was well-prepared and clear, no noteworthy roadblocks were encountered throughout testing, and cross-team queries were kept to a minimum as a result. GreatFire delivered excellent test preparation and assisted the Cure53 team in every respect to procure maximum coverage and depth levels for this exercise. Cure53 gave frequent status updates concerning the test and any related findings, whilst simultaneously offering prompt queries and receiving efficient, effective answers from the maintainers. Live reporting was not requested, which in hindsight proved a sufficient decision considering the relatively low severity levels of the findings detected.

Regarding the findings in particular, the Cure53 testing team achieved comprehensive coverage over the single scope item, identifying a total of five. All five findings were categorized as security vulnerabilities, with none deemed general weaknesses with lower exploitation potential.

Generally speaking, the overall volume of findings is considerably low, which could be perceived as a positive indication of the platform's perceived security strength. However, Cure53 must stipulate that a full browser audit is impossible to achieve within a four-day allocation, thereby one can assume that hitherto unforeseen vulnerabilities will continue to blight the FreeBrowser until further auditing is administered.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Two of the discovered issues were assigned a *Medium* severity, which highlights that FreeBrowser's threat surface remains relatively restricted. However, one of the five findings was issued a severity rating of *High*; given the threat model of this software, this should evidently be addressed at the earliest possible convenience.

All in all, the conclusion can be made that the GreatFire development team has already established strong security practices for the FreeBrowser implementation, with only a handful of minor adjustments required to reach a first-rate security posture.

The report will now shed more light on the scope and testing setup as well as provide a comprehensive breakdown of the available materials. Subsequently, the report will list all findings identified in chronological order, starting with the detected vulnerabilities and followed by the general weaknesses unearthed. Each finding will be accompanied by a technical description and Proof of Concepts (PoCs) where applicable, plus any relevant mitigatory or preventative advice to action.

In summation, the report will finalize with a conclusion in which the Cure53 team will elaborate on the impressions gained toward the general security posture of the GreatFire FreeBrowser, giving high-level hardening advice where applicable.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Scope

- **Penetration tests and code audits against GreatFire Android library and web application**
  - **WP1**: White-Box Tests & Security Audits against GreatFire FreeBrowser
    - Production APKs:
      - https://bitbucket.org/greatfire/wiki/raw/master/FreeBrowser.apk
    - Binaries can also be downloaded on:
      - https://freebrowser.org/
      - https://github.com/greatfire/wiki
  - **Threat model and security expectations**
    - **What does the FreeBrowser try to achieve**
      - Provide users primarily in mainland China with a comprehensive circumvention solution which includes both the circumvention technology and suggested content outside the firewall.
    - **What privacy and security guarantees does FreeBrowser give to users**
      - Access to an uncensored internet and, other than that, no additional security and privacy guarantees that an off-the-shelf Chromium browser would offer
    - **Who is the intended audience for FreeBrowser**
      - Chinese Internet users with little or no experience of accessing the internet outside the GFW.
    - **Who would want to attack FreeBrowser and its users**
      - Most obvious attacker model would include the CCP - in order to either bring the service down, or to monitor what users do while using.
  - **All relevant sources were shared with Cure53**

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Identified Vulnerabilities

The following sections list all vulnerabilities and implementation issues identified throughout the testing period. Please note that findings are listed in chronological order rather than by their degree of severity and impact. The aforementioned severity rank is simply given in brackets following the title heading for each vulnerability. Furthermore, each vulnerability is given a unique identifier (e.g., *GF-04-001*) to facilitate any future follow-up correspondence.

GF-04-001 WP1: Proxy bypass via overly permissive whitelisting *(Medium)*

The GreatFire FreeBrowser defines a list of domains that are not proxied; in this way, the browser directly communicates with the domain. Testing confirmed that this bypass contains overly broad bypass rules such as *.sohu**. This can lead to a domain bypassing the proxy unintentionally whilst the user remains oblivious, for the simple reason that a certain keyword is present in the domain name.

Example domain name:
*test**.sohu**doman.dnsdigger.h4x.tv*

Affected file:
xpatch-fbapp-5.1.2/browser/android/xpatch_session.cc

Affected Code:

```
g_bypass_rules.ParseFromString(".cn, .baidu*, .zhihu*, .paypal*, .163.*,
*.doubleclick.net, *.taboola.com, *.googlesyndication.com, adservice.google.com,
.qq*, .gtimg.com, .jd*, .weixin*, .sohu*, .douban*,
*analytics.com, .sina*, .ali*, .taobao*, .weibo.com, .bdstatic.com, .github.io,
github.com, .githubassets.com, .github.com, startpage.local, whoer.net"
```

To mitigate this issue, it is recommended to define the bypass list more strictly to ensure arbitrary domains cannot be sent without being initially proxified.

GF-04-002 WP1: Debug code included in release build *(Info)*

Whilst assessing FreeBrowser's local-file process implementation, the discovery was made that files are read from the local *tmp* folder despite being utilized for additional debugging purposes. Notably however, since a malicious APK does not possess the default permissions to write to the */data/local/tmp/* directory, this weakness does not introduce a significant security issue in isolation.

Affected file:
xpatch-fbapp-5.1.2/xpatchng.cc

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Affected code:

```
if(access("/data/local/tmp/gf.mynode", F_OK) == 0) {
    enable_mynode_ = true;
  }
```

Affected file:
xpatch-fbapp-5.1.2/wrapped_calcer.cc

Affected code:

```
handle = dlopen("libcatboostmodel.so", RTLD_LAZY);
    if(handle == NULL) {
       handle = dlopen("/data/local/tmp/libcatboostmodel.so", RTLD_LAZY);
    }
```

To mitigate this issue, one should consider removing these debugging features for release builds. This method is considered a strong practice toward deterring the introduction of accidental and erroneous security issues.

GF-04-003 WP1: Website JavaScript can detect FreeBrowser *(Low)*

The FreeBrowser offers an almost identical browsing experience as a benign Chromium browser. Nevertheless, two behaviors were unearthed following testing that permit a website to detect the FreeBrowser. A malicious site could utilize these discrepancies to display alternative information to FreeBrowser users, or to instigate user-IP tracking. The first technique specifically leverages FreeBrowser's incorrect handling of URL credentials, which leads to an HTTP 500 Server Error via the deployed proxy endpoint.

URL credential PoC:

```
x = new XMLHttpRequest();
x.open("GET",`${location.protocol}//a:b@${location.hostname}/`,false);
x.send();
    if (x.status == 500)
    {
        alert("FreeBrowser behavior detected");
    }
    else
    {
        alert("Default Browser behavior detected");
    }
```

Another detection mechanism was built around the *Fb-Xbackend* header, which is set by the proxy in an HTTP response. JavaScript can check for the presence of this header and reveal the usage of the FreeBrowser.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Fb-Xbackend response header PoC:

```
x = new XMLHttpRequest();
x.open("GET","./header_test.php",false);
x.send();
if (x.getResponseHeader("Fb-Xbackend") == "abcd")
{
    alert("Default Browser behavior detected");
}
else
{
   alert("FreeBrowser behavior detected");
}
```

HTTP request:

```
GET /v4/proxy/2207bd466e7ea926d0a9dbbd4c24561c HTTP/2
Host: www.virusimpact.net
[...]

HTTP/2 200 OK
Cache-Control: max-age=0
Content-Type: text/html; charset=UTF-8
Date: Wed, 30 Mar 2022 16:24:46 GMT
Fb-Xbackend: abcd
Fb-Xbackend: true
Server: nginx/1.10.3
X-Cdn: Verizon
X-Powered-By: PHP/7.3.33

text
```

To mitigate this issue, one can recommend addressing the documented discrepancies and adapting them to mimic the behavior of a benign Chromium browser instance. This would ensure that a malicious web page cannot simply discover any FreeBrowser usage scenarios.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

GF-04-004 WP1: Local startpage reveals FreeBrowser users via referrer *(Medium)*

When a user opens the FreeBrowser browser, a startpage hosted on *startpage.local* is displayed. Here, testing confirmed that the startpage presented on non-English or Persian Android systems triggers a request to *pv.sohu.com* to display the user's non-proxied IP. As this request contains the *startpage.local* domain in its *Referer* header, the *pv.sohu.com* domain owner would be able to detect any FreeBrowser user's IP by observing the access log.

Pertinently, the same behavior is present for *startpage.freebrowser.org* in addition, which is utilized as a fallback in the eventuality the app is not able to load *startpage.local* correctly.

Affected file:
xpatch-fbapp-5.1.2/xpatchng.cc

Affected code:
```
void XPatchNG::cloneStartPage() {
  MSGTYPE msgtype = MSG_GIT_OK;

  string startpage_dir(filesdir_);
  startpage_dir.append("/startpage");

  string giturl("https://github.com/sprattjack/congenial-octo-telegram.git");
  if(lang_.compare("fa") == 0) {
    giturl.assign("https://github.com/sprattjack/potential-octo-parakeet.git");
  } else if(lang_.compare("en") == 0) {
    giturl.assign("https://github.com/sprattjack/friendly-succotash.git");
  }
```

Affected file:
/congenial-octo-telegram/blob/master/index.cn.html

Affected code:
```
[...]
var t=document.createElement("script");t.src="https://pv.sohu.com/cityjson?
ie=utf-8",t.onerror=()=>{this.ips.in.push("?")
[...]
```

Sent HTTP request:
```
GET /cityjson?ie=utf-8 HTTP/2
Host: pv.sohu.com
Referer: http://startpage.local/
[...]
```

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

```
HTTP/2 200 OK
[...]
```

To mitigate this issue, the recommendation can be made to simply remove this feature. This would ensure that any usage of FreeBrowser is not immediately leaked to a third party via these means. An alternative approach could constitute removing or spoofing the *Referer* header to ensure it cannot be linked to the FreeBrowser application.

GF-04-005 WP1: Startpage hijack via GitHub DNS IP manipulation *(High)*

As relayed in ticket GF-04-004, the FreeBrowser application displays a locally-hosted startpage. This is implemented by cloning a specific repository from *GitHub.com* during startup. To ensure the security of the user, the certificate is validated by the FreeBrowser application. Whilst verifying this logic, the observation was made that the utilized *libgit2* library sends an HTTP request to the specified repositories */refs* endpoint prior to the defined certificate-callback trigger.

This behavior facilitates loading an arbitrary GitHub repository via an HTTP redirect and therefore displaying any startup page, as long as the DNS record of *GitHub.com* can be controlled by a third party. This startpage could leak the user's real IP address, display misinformation, and more.

The issue was verified by setting a local IP for *GitHub.com* that hosted an HTTP server with a self-signed certificate. After redirecting the request to another GitHub repository, all subsequent requests were forwarded to *GitHub.com*, therefore passing the deployed certificate verification.

HTTP request sent by libgit2:

```
GET /sprattjack/friendly-succotash.git/info/refs?service=git-upload-pack
HTTP/1.1
[...]
```

HTTP response:

```
HTTP/1.1 301 Moved Permanently
Location: https://github.com/cure53alexander/test.git/info/refs?service=git-
upload-pack
```

Affected file:
xpatch-fbapp-5.1.2/xpatchng.cc

Affected code:

```
bool XPatchNG::do_git_clone(const char* url, const char* path, string& errstr) {
[...]
```

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

```
clone_opts.fetch_opts.callbacks.certificate_check = ssl_cert_cb;
clone_opts.fetch_opts.callbacks.payload = NULL;
int error = git_clone(&cloned_repo, url, path, &clone_opts);
```

To mitigate this issue, it is recommended to inform the library owner of this detrimental behavior to ensure the library does not process any information before the utilized certificate is processed and accepted by the client.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Conclusions

The impressions gained during this report - which details and extrapolates on all findings identified during the CW13 testing against the GreatFire FreeBrowser, with a particular focus on the Android library and web application, by the Cure53 team - will now be discussed at length. To summarize, the confirmation can be made that the components under scrutiny have garnered a positive impression.

Since FreeBrowser's design comprises small modifications of the open source Chromium browser's network stack, no additional attack surface is exposed to third-party apps via additional activities and other instances. Ultimately, only minor debug functionality was confirmed as present in the release build; however, as documented in ticket GF-04-002, a third-party app would not be able to abuse trigger these features.

Another primary focus was placed upon the FreeBrowser initialization, including the startpage retrieval and obtainment of valid proxy information. The general design of the proxy domain discovery and validation seemed soundly composed and correctly implemented following testing. However, despite validating the certificate of the configured GitHub repository, an issue in the utilized library was detected (see GF-04-005). This underlines the essential necessity of verifying the behavior of deployed certificate-verification logic to uncover potential flaws in libraries as early as possible before they are able to proliferate and cause exponential damage.

Following the completion of the initialization process assessment, the FreeBrowser application was checked for IP leakage or other discrepancies to a benign Chromium browser, which could otherwise reveal the presence of the FreeBrowser. The deployed proxy-bypass list was deemed overly permissive, which can lead to the unintentional unproxying of domains (see GF-04-001). Another oversight in a feature within the deployed start page leaks the FreeBrowser's real IP to a third-party application and exposes the usage of FreeBrowser in addition (see GF-04-004).

Furthermore, this deep-dive audit of the FreeBrowser demonstrated mostly identical behaviors to its Chromium browser counterpart. Therefore, no additional issues were found that reveal user IPs, though two techniques were documented that allow a website to discover any FreeBrowser usage instances (see GF-04-003).

All in all, the security impression gained of the FreeBrowser application is evidently positive. The design and implementation have been established with strong security practices in mind, despite the few detected findings.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Following the mitigation of all issues raised in this report, Cure53 would be happy to confirm that the FreeBrowser application exhibits first-rate security protection for its users.

Cure53 would like to thank Charlie and Martin from the GreatFire.org team for their excellent project coordination, support and assistance, both before and during this assignment.