Fine penetration tests for fine websites

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

Analysis-Report Chinese Police App “Feng Cai” 03.2019

By Cure53 (various members of the Cure53 team contributed to this report)

Index

Introduction

This report documents the findings of a Cure53 assessment targeting the integrated Feng Cai mobile application. The project was requested and funded by the Open Technology Fund (OTF) with the main objective of finding out whether the Feng Cai mobile application violates human rights. This corresponds to previous analyzes commissioned by the OTF team in this realm, especially in checking how the Feng Cai app relates to the previously examined JingWang and IJOP projects.

In terms of specific tasks and timeline, Cure53 carried out a source code audit and a dedicated review of the Feng Cai mobile application in late March 2019. The project followed a so-called white-box methodology as much as possible, particularly in the sense that the test-targets entailed decompiled source code of the application, which had been made available to Cure53 as an APK file by the OTF team.

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

The assessment took a total of ten days, excluding a separate budget allocated to preparations, communications and write-up of this report. Note that the project was executed stealthily, i.e. the Cure53 team aimed at avoiding detection. This meant refraining from any form of behavior that could be perceived as “noisy” and, thus, alert the maintainers of the application and/or server owners. As noted above, the assessment tackled both the source code and the running application. Throughout this assignment, the Cure53 and the OTF team maintained frequent contact on a shared Slack channel. This allowed for fast messaging and a good flow during the test, additionally ascertaining that Cure53 does not pursue exploration avenues and directions useless from the OTF’s point of view.

In should be emphasized that the European Convention on Human Rights (ECHR) served as a baseline for this project in that Cure53 set out to determine - through the technical reviews and audits - whether (and which) communication capabilities and functionality of the Feng Cai mobile application can be seen as directly contradicting what the ECHR guarantees. Therefore, this was a search for any human rights violations that the operational scope of Feng Cai could enable and facilitate.

For the purpose of documenting the findings, Cure53 devised a classification system that could help discern the level of harm and certainty about each potential violation of human rights. This system is discussed in more detail below. At this stage, it can nevertheless be stated that Cure53 managed to prove two cases of clear human rights (HR) violations. All other items, grouped in an array of five discoveries, should also be considered, yet only in the context of having all vital information at hand.

In the following sections, the report first elaborates on the scope and then sheds light on the link between severities of the findings and the employed classification system. The questions posed by OTF, along with detailed answers and research results, are discussed next. Further, Cure53 reiterates some of the key points in the *Conclusions* and, last but not least, supplies various relevant material in the *Appendix*. It is hoped that this will facilitate further research or other actions to be taken by either OTF or any other - involved or interested – parties.

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

Scope of the Assessment

- **Feng Cai Chinese Police App**
  - The assessment focused on the mobile application for Android phones. The application - labelled as **Feng Cai** - is supposedly used in specific regions of China by Law Enforcement personnel to gather and manage data about specific groups of citizens and/or minorities.
  - In a nutshell, Cure53 got access to the APK of the app through OTF and was tasked with finding out whether the app is novel or has been analyzed before. If proven to be new, Cure53 was to find out what the app is capable of.
  - Of particular importance was the question of the activities and features of the app that could be used in a way that violated human rights. It was assumed that the app would be forcefully or secretly installed by local authorities on citizens' phones.
  - Cure53 received a briefing and all of the already existing documentation about the context of this assessment from OTF. Communications ensued in the aforementioned Slack channel.

Classification of Findings

During this assessment, Cure53 team has been using the following classification to specify the level of certainty regarding the documented findings. Given that this research had to happen on the basis of reverse-engineering and needed to be executed in a stealthy manner, it is necessary to classify the findings to make sure which level of reliability can be assumed.

- *Proven* - Source code and the analyzed activity clearly indicate a HR violation.
- *Evident* - Source code strongly suggests a HR violation.
- *Assumed* - Indications of a HR violation were found but a broader context remains unknown.
- *Unclear* - Initial suspicion was not confirmed. No HR violation can be assumed.

Questions & Answers

OTF and Cure53 had frequent discussions throughout the project. The research results have been shared and commented, usually in real-time. In addition to the reverse-engineering of the app's features as a means to find human rights violations, Cure53 was also tasked with answering specific questions sent in by the OTF team. To fully cover the scope of the assessment and all related exchanges, the questions asked by OTF are incorporated into this report. Cure53's research-based responses can be found in the ensuing sections.

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

FNG-01-001 Information gathering *(Proven)*

The questions posed by the OTF team were as follows:

Q1: *"What files/information is gathered by the app?"*

The following information is gathered:

- All calendar entries, phone contacts, country codes and dialed numbers;
- General information about the Phone (IMEI, IMSI, PhoneSN, WifiMac, BluetoothMac, and if the device is rooted);
- Information about the Android Model (*CPU_ABI*, *BOARD*, *HARDWARE*);
- All stored text messages (SMS);
- Information about the current base-station;
- Information about the used hardware (Mac Addresses);
- All information accessible for various installed apps + an MD5 hash of the app;
- Mac Addresses;
- Current phone number;
- Extracted information searched by the *GetVirAccount* application which searches the */sdcard* for specific data of specific China-related apps. This information contains phone numbers and email addresses.

Affected File:
com/fenghuo/utils/Global.java

The list was compiled based on the information contained in the files, as well as by inspecting the generated report.

GetVirAccount is one of the applications shipped by the police-app. During the installation process they are extracted and stored in the apps data storage. The *GetVirAccount* gets executed by the app. *GetVirAccount* parses the *id.conf* file, which is shipped by the app as well.

The *id.conf* file contains the following data:

```
com.tencent.mobileqq    tencent/MobileQQ/   DIR    (^[1-9][0-9]+)
com.tencent.mobileqq    Tencent/MobileQQ/   DIR    (^[1-9][0-9]+)
com.tencent.mobileqq    tencent/QWallet/    DIR    (^[1-9][0-9]+)
com.tencent.mobileqq    Tencent/QWallet/    DIR    (^[1-9][0-9]+)
com.renren.mobile.android
Android/data/com.renren.mobile.android/cache/talk_log/    FILE    talk_log_([0-
9]+)_.*
```

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

```
com.duowan.mobile    yymobile/logs/sdklog/    FILE_CONTENT    logs-yypush_.*txt
safeParseInt ([0-9]*)
com.immomo.momo    immomo/users/    DIR    (^[1-9][0-9]+)
cn.com.fetion    Fetion/Fetion/    DIR    (^[1-9][0-9]+)
com.alibaba.android.babylon
Android/data/com.alibaba.android.babylon/cache/dataCache/    FILE    (^[1-9][0-
9]+)
#"phone":"18551411***"
com.sdu.didi.psnger    Android/data/com.sdu.didi.psnger/files/omega
FILE_CONTENT    e.cache    "phone":"([0-9]*)"
#aaaa
com.sankuai.meituan    Android/data/com.sankuai.meituan/files/elephent/im/    DIR
(^[1-9][0-9]+)
com.sogou.map.android.maps    Android/data/com.sogou.map.android.maps/cache/
FILE_CONTENT    cache    "a":"([^"]*)"
#com.sina.weibo    loginname=red***@163.com&
com.sina.weibo    sina/weibo/weibolog/    FILE_CONTENT    sinalog.*txt
loginname=([^&]*)&
```

All the referenced file paths are looked up inside the */sdcard/* folder of the phone. The regular expressions defined show that the application is searching for phone numbers and login names of the defined apps.

Q2: *“Is there evidence that it actually downloads messages, etc?”*

The app uses its SMS permissions to dump all stored text messages and includes them in the report.

Q3: *“Are apps, which allow End-To-End encryption, logged?”*

The current code does not indicate that it checks any of the installed apps and instead all installed applications are hashed and included in the ZIP file uploaded by Feng Cai, see FNG-01-002. The *Wifiscan* application does check all files stored on the SD card via hash comparison but it is not known which exact files it is looking for with only the hashes being known.

Q4: *“Does it check if the “phone” was in specific countries?”*

Every call and every message are extracted and the acquired data could be used to arrive at conclusions about visits to foreign countries.

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

FNG-01-002 File transmission and protection *(Proven)*

The questions posed by the OTF team have been answered by Cure53 below.

Q5: *"Where and how are the files transmitted?"*

All extracted information is bundled as a *ZIP* file, without applying any protection like a password. The *ZIP* file is then sent via an HTTP *POST* request to http://192.168.43.1:8080/. This shows that not only no transport security (e.g. *https://*) is in place, but also that an internal IP address is used.

Q6: *"If it is a wifi server, as suggested by the package name (i.e. com.fiber-home.wifiserver), can unauthenticated attackers dump the data too or only the police?"*

Although the name indicates the presence of a Wi-Fi server, the code does not implement a feature which would allow to open a hotspot. Instead, the code indicates that the phone gets connected to a specific WLAN connection, for which authentication details are removed as soon as the app is uninstalled.

File:
app/src/main/java/com/fenghuo/qzj/WelcomeActivity.java

Affected Code:
```
uninstall.setOnClickListener(new View.OnClickListener()
{
public void onClick(View paramAnonymousView)
{
[...]
paramAnonymousView = (WifiManager)getSystemService("wifi");
int i = WelcomeActivity.this.getConnectionWifiSsid(paramAnonymousView);
paramAnonymousView.removeNetwork(i);
paramAnonymousView.saveConfiguration();
if (Build.VERSION.SDK_INT >= 23) {
paramAnonymousView.disableNetwork(i);
}
```

However, the app uses the Android *AllowAllHostnameVerifier* hostname-verifier which could lead to Man-in-the-Middle issues.

File:
/com/fenghuo/http/TrustAllSSLSocketFactory.java

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

Affected Code:
setHostnameVerifier(new AllowAllHostnameVerifier());

The official Android documentation[1] has the following to say about the *AllowAllHostnameVerifier*:

The ALLOW_ALL HostnameVerifier essentially turns hostname verification off. This implementation is a no-op, and never throws the SSLException.

Furthermore, the app has an empty implementation of *checkServerTrusted* which could also cause Man-in-the-Middle problems.

File:
/com/fenghuo/http/HttpsManager.java

Affected Code:
```
@Override
        public void checkServerTrusted(X509Certificate[] arrx509Certificate,
String string2) throws CertificateException {
        }

        @Override
        public X509Certificate[] getAcceptedIssuers() {
            return null;
        }
```

File:
/com/fenghuo/http/TrustAllSSLSocketFactory.java

Affected Code:
```
    public void checkServerTrusted(X509Certificate[]
paramArrayOfX509Certificate, String paramString)
      throws CertificateException
    {}

    public X509Certificate[] getAcceptedIssuers()
    {
      return null;
    }
```

Q7: *"Is data dumped in the SD Card from where it could be retrieved later without even entering the PIN to unlock the device?"*

[1] https://developer.android.com/reference/org/apache/http/conn/ssl/AllowAllHostnameVerifier

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

The application stores all scan-related data in its own data directory. The only file stored on the SD card is the *cjlog.txt* file. The file is stored in an encrypted form and contains information about when the last scan took place.

FNG-01-003 Pre-processing/MD5 hashing of files and applications *(Assumed)*

The OTF questions and corresponding Cure53's responses concerning this realm are included below.

Q8: *"Is the data gathered already pre-processed by the app?"*

The application utilizes the *Wifiscan* application to scan the SD card. It compares the hash of each file with the encrypted hashes stored in *bk_samples.bin*. Furthermore, the date of the last scan remains encrypted on the phone.

Q9: *"Are files MD5 hashed as done by other apps? If yes - which files?"*

All files on the SD card are MD5-hashed and matched with the hashes stored in *bk_samples.bin*. Furthermore, the app takes an *md5* for all installed APKs on the phone. The MD5 hash is then saved into *app_list*, a file inside the ZIP file that is sent in the format shown next.

***app_list* content:**

```
Example Wallpapers    com.example.android.livecubes    installed    6.0-233
14689    1456040695    /system/app/CubeLiveWallpapers/CubeLiveWallpapers.apk
23    null    e89b158e4bcf988ebd09eb83f5378e87    null
```

MD5 hash:

```
e89b158e4bcf988ebd09eb83f5378e87
```

Q10: *"What exactly does the app search for? What does it identify as a 'hit'? "*

The app appears to be using the *Wifiscan* application to search for suspicious files. A hit seems to be a file on the SD card matching an MD5 hash in the *bk_samples.bin* file.

Example file-size and hash in *bk_samples.bin:*

```
135510055
E624931E72EB7D0736B8E43BE9BBA4B6
```

The app is completely wiped after an "*uninstall",* with the exception of the following file remaining on the SD card.

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

File:
/sdcard/android/cjlog.txt

The file is stored encrypted and contains the time of when the last scan took place. Additionally, it contains a number but it is not used by the application. It can be decrypted/deobfuscated with the *gen_wifi_cj_flag* application. When the file is decrypted, it is stored as *cjlog_plain.txt* on the phone's SD card.

Decryption File:
app/src/main/java/com/fenghuo/qzj/WelcomeActivity.java

Affected Code:
```
ShellCommands.doSuCmds("sh", Global.absolutefilesPath_ + "/gen_wifi_cj_flag_pie
" + Global.mSdCardPath_ + "/Android/cjlog.txt " + Global.mSdCardPath_ +
"/cjlog_plain.txt" + " 2>" + Global.absolutefilesPath_ + "/log_file 1>" +
Global.absolutefilesPath_ + "/error_file");

// command line
/data/data/com.fiberhome.wifiserver/files/gen_wifi_cj_flag_pie
/sdcard/Android/cjlog.txt /sdcard/cjlog_plain.txt
2>/data/data/com.fiberhome.wifiserver/files/log_file 1>
/data/data/com.fiberhome.wifiserver/files/error_file
```

Generation of file:
```
/data/data/com.fiberhome.wifiserver/files/wifiscan_pie sm /sdcard
2>/data/data/com.fiberhome.wifiserver/files/error_file
1>/data/data/com.fiberhome.wifiserver/qzj/000000000000000/scandir_temp
```

FNG-01-004 Code Execution and Backdoors *(Unclear)*

In terms of the code execution and backdoors, another set of questions is supplied and discussed next.

Q11: *"Can the shell commands that the app runs lead to RCE in any way?"*

Yes, this is possible by modifying the default external storage path. The modified path has to include a folder's name, which actually contains shell commands, for example *touch /tmp/test*. The app is using the shell to execute the applications it ships (*wifiscan*, *GetVirAccount*, etc.). It does so by executing the *sh* command and then by writing the commands and file paths to the standard input of the shell and then it sends a "return" key to execute those. The file paths are retrieved by the app by looking up the default external storage path (which is used by the *wifiscan* application as an example)

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

In case a user would want to exploit this, they could modify their Android system to have the external storage path contain a folder name, which then contains shell commands like *id*. Given how the app handles the file path, this would lead to the folder name being treated as a shell command itself and therefore execute code.

Since the code is executed by the Feng Cai app, the shell command is executed with the logged in user's privileges. However note that this is a theoretical scenario. A normal user would probably not modify his phone in any way that would enable this.

Q12: *"Has the app some kind of backdoor?"*

No backdoors were discovered in the app.

Q13: *"Does the app requires root access?"*

No. Most of the data is accessed by requesting the needed permissions in the app's *Manifest* file. The functionalities, which scan files related to other apps, only do this on files stored on the external storage (e.g. the SD card). As such, having *root* access is not required.

FNG-01-005 Similarities & differences btw. Feng Cai, JingWang & IJOP *(Assumed)*

The next set of questions from the OTF team can be found with Cure53's responses next.

Q14: *"What is the relationship to this new app (let's call it Feng Cai) and jingwang and/or IJOP, if any? Is there any indication as to whether the app stays active on the user's device indefinitely or whether its functionality is time-bound in any way?"*

The app appears to be used in the following scenario:

- A police officer takes the phone of a citizen to his/her car and installs the Feng Cai app.
- The unlocked phone is then connected to a police WiFi hotspot from where *http://192.168.43.1:8080/* should be reachable. This is additionally supported by the fact that, as soon as the app starts, it prompts the user to change the WiFi settings.
- The connection is established over clear-text HTTP without any protections.
- The police officer scans the device and uploads the captured data to the local file server.

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

- The police then uninstall the app, which will clean-up all artifacts as temporary files written to the SD car, marking them for a later deletion. Uninstallation will delete the *ZIP* file which is kept in the protected, internal app storage.
- Only one encrypted trace, which contains the date of the scan, remains on the device. This possibly serves comparison purposes in future scans.
- The previously used WiFi network is disconnected and the respective credentials are removed on *uninstall*.

The theory of this scenario is supported by the evidence laid out in the following paragraphs. The following figure shows the icons of Feng Cai, JingWang and IJOP in the mentioned order.

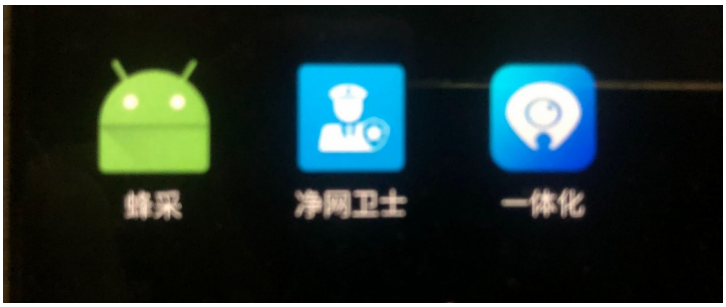

Fig.: The Feng Cai app relies on the default Android icon while JingWang and IJOP have official icons.

Feng Cai uses the default icon for Android apps, which means there is no attempt at being covert or discreet about it. This is because the app will only exist for the duration of a citizen's phone being in the hands of a police officer. The following comparison demonstrates the very different UI designs between Feng Cai and JingWang.

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

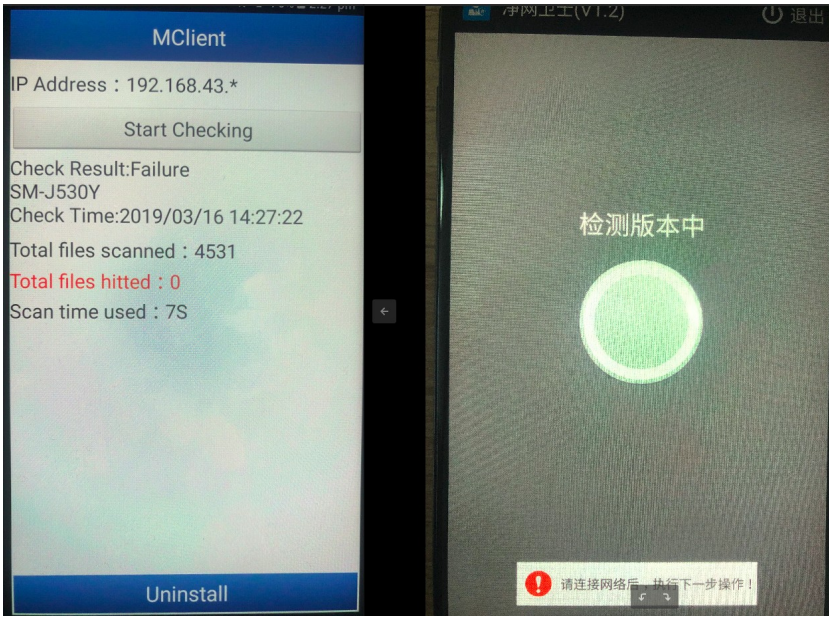

Fig.: The Feng Cai App serves only one activity with three features: Scan, Upload, Uninstall. JingWang makes a more official impression as it contains official icons and tries to establish a connection to a public server.

Local server contacted by Feng Cai:
`192.168.43.1:8080`

Public server contacted by JingWang:
`47.93.5.238:8081`

IJOP is the least similar app of the ones in scope, merely by being a reporting tool for the police and not a scanning/spy tool. Cure53 has reason to believe that IJOP is used by a police officer to communicate with their HQ. It is not set for installation on the phones of suspicious citizens. More information specific to IJOP will be provided in a forthcoming publication.

The design of JingWang implies that it is meant to remain or to have a legitimate use-case on a citizen's phone. Although both, Feng Cai and JingWang, engage in hashing files, the former reports to a local server while the latter establishes a connection to a public IP address. Furthermore, Feng Cai requires more permissions than JingWang, which can be observed by comparing the Manifest files of the apps.

This leads Cure53 to the conclusion that Feng Cai is more intrusive than JingWang, possibly because it does not require a citizen's consent to be installed; the installation is enforced by a police officer.

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

Conclusions

This technical analysis and review of the Feng Cai mobile application has demonstrated that the concerns expressed by OTF are valid. Carried out by Cure53 in close collaboration with the OTF team, this March 2019 project sheds light on five items from the perspective of potential violations of human rights.

In a nutshell, judging by the research outputs and results of an in-depth analysis, the Cure53 team finds it evident and undeniable that the application is capable of collecting and managing vast amounts of very specific data. It is certain that the gathered material can become a basis for further action concerning a specific group (or groups) of citizens. According to the European Convention on Human Rights, which stands among other examples of agendas and corresponds to related court rulings, the above practice can be considered a human rights violation.

The main aspects that should be highlighted among the various findings with differently-evaluated severities pertain to the plethora of information-gathering executed by the app (see FNG-01-001). Similarly concerning is the transmission of a lot of data to a local police file server (see FNG-01-002). The MD5 hashing of applications and files on the SD card can be interpreted as an attempt at identifying forbidden content (see FNG-01-003). However, it is unclear what qualifies as forbidden content because no '*hit*' hash could be reproduced during this project. The *Appendix* contains a link to the *ZIP* file with the captured data uploaded by the Feng Cai app. This material should help with getting a first-hand impression of the app's operations

In a broader sense, the application's functionality leads Cure53 to believe that violations of human rights indeed take place. Especially the items noted with a *"Proven"* status serve as solid evidence of this fact. At the same time, it should be noted that Cure53 operated as a purely technically-driven team and an unbiased investigating entity. Therefore, it is not a party in any way involved in making final judgments as to whether human rights violations take place from legal, social or political standpoints. The Cure53 team works from a premise of collecting technical evidence, which is based on reverse-engineering operations.

Cure53 would like to thank OTF team for their excellent project coordination, support and assistance, both before and during this assignment.

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

Appendix

Appendix 1: *ZIP* uploaded by the Feng Cai app; contains all captured information

https://drive.google.com/open?id=1kYmH_DX6ASezM79BOBMA-fC2L5B8ZfYN

Appendix 2: *cjlog.txt* is left on the device after a scan

https://drive.google.com/open?id=1mEGfwT8EQvm-1clu2QCUNGh9iB3axUZz

Appendix 3: *cjlog_plain.txt* contains the unencrypted data of *cjlog.txt:*

https://drive.google.com/open?id=1AIBPqZVACSRtXBTKrpwDrvj_3xlVp_6p