Fine penetration tests for fine websites

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

Analysis-Report “Study the Great Nation” 08.-09.2019

Cure53, Dr.-Ing. M. Heiderich, various Cure53 members and an external Contractor

Index

Introduction

> *“Study the Great Nation is a multifunctional smartphone application created around the Communist Party of China and the life of Xi Jinping. It was developed by the Parties Propaganda Department and tech-giant Alibaba. It allows to video chat with friends, send messages that get deleted after having being read, create a personal calendar, get informed through the state media or watch TV series about the History of the Communist Party of China.”*
>
> From https://en.wikipedia.org/wiki/Study_the_Great_Nation

This report documents the results of an analysis targeting the so-called “Study the Great Nation” mobile application. Cure53 was tasked with reviewing the premise of this app by the Open Technology Fund and completed the assessment in late August 2019.

To clarify the interest in this topic and scope, it should be noted that the “Study the Great Nation” mobile application has gained a massive following. Allegedly, it has been downloaded more than 100 million times. According to various sources, the app is getting heavily promoted by various powerful stakeholders, such as Chinese state media, universities, schools and similar parties.

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

In this context, the Open Technology Fund decided to sponsor an investigation into the application's capacities and operations. In particular, Cure53 was supposed to analyze what the "Study the Great Nation" application was actually doing "under the hood". One of the main objectives was to verify if any activities were performed by the app without the public's awareness, i.e. as regards advertising and/or documentation. An explicit focus was therefore placed on unadvertised features which could be seen as aiding the maintainers of the app in data collection. From there, a careful leap would be made in terms of looking at ways of collecting data in a manner that violates human rights. To guarantee transparency, it should be emphasized that the European Convention on Human Rights (ECHR) served as a baseline for Cure53 during this project.

In terms of resources and timeline, Cure53 performed the analysis in Calendar Weeks 34 and 35, i.e. late August 2019. A number of team members and an external contractor were involved in this assessment and spent a total of fifteen days on this project. The budget was dedicated to the core research and analyses, as well as to communications and documentation of findings.

It needs to be mentioned that Cure53 has accumulated considerable expertise in conducting similar projects over the years. Consequently, as with similar reports, for example the 2018 BXAQ mobile app analysis[1] or the 2019 IJOP mobile application analysis[2], the team decided to deploy a dedicated and optimized classification system for the findings and aspects. This should be seen as a means to discern both the level of harm and the existing certainty about each potential violation of human rights. This system is discussed in greater detail below.

At this stage, it can be stated that Cure53 managed to prove one case of a clear human rights violation (HRV) by the "Study the Great Nation" app. All other items, grouped in an array of six discoveries, should also be considered, yet only in the context of having all vital information at hand. Three of the six findings were rated as "*Evident"* HRV and the remaining two remain unclear as regards human rights implications.

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. After addressing all findings, Cure53 reiterates some of the key points about the studied application in the context of human rights in the *Conclusions* section. More broadly, it is hoped that this Cure53 report will provide a stepping stone for further research capable of shedding more light on what this application does, especially in terms of processing information it aggregates on the server-side.

[1] https://cure53.de/analysis-report_bxaq.pdf
[2] https://cure53.de/analysis-report_ijop.pdf

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

Scope

- **Analysis of the "Study the Great Nation" Mobile App**
  - **Background info on the analyzed mobile app**
    - https://www.nytimes.com/2019/04/07/world/asia/china-xi-jinping-study-the-great-nation-app.html
    - https://www.nytimes.com/2019/02/14/technology/china-communist-app.html?module=inline
    - https://gizmodo.com/chinas-little-red-book-like-app-is-blowing-up-the-chart-1833886502
  - **Sources for obtaining the app during this analysis**
    - https://www.xuexi.cn/
    - https://apps.apple.com/cn/app/%E5%AD%A6%E4%B9%A0%E5%BC%BA%E5%9B%BD/id1426355645
    - https://h5.xuexi.cn/page/download.html
  - **Detailed Scope:**
    - Cure53 was to check if the app does anything it does not publicly advertise, as well as specifically look at what possibly points in the direction of possible human rights (HR) violations.
    - Special focus was placed on collection of PII or phone details - IMSI, IMEI, PII, microphone data, file-system data, etc.
    - Attention was given to checking if any of the servers this app talks to is related to the servers of the other apps, with the question being about connections.
    - Special focus was placed on the external companies that help build and maintain this app - i.e. who runs the servers, who licenses libraries to them, and so on.
  - **Spotted prior research**
    - https://github.com/fuck-xuexiqiangguo/Fuck-XueXiQiangGuo

Classification of Findings

During this assessment, the 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 address a given level of reliability that can be assumed for each discovery.

- *Proven* - Source code and the analyzed activity clearly indicate a HR violation.
- *Evident* - Source code strongly suggests an HRV.
- *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.

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

SGN-01-001 Information gathering *(Assumed)*

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

The app collects the following information:

- General information about the phone (IMEI, device model, brand, device ID, AppKey, info on whether the device is rooted);
- Connection information (Wifi-SSID, carrier, VPN-check)
- User-information (UIDs, cookies, session-IDs, Event-, Page- and Track-IDs, calls, call statistics, contacts);
- Location;
- Running processes and services.

Affected Files:
com/alibaba/android/rimet/statistics/service/api/lwp/models/DataPlatParams.java
com/alibaba/android/rimet/statistics/service/api/lwp/models/DataPlatStaticParams.java
com/alibaba/android/rimet/statistics/service/api/lwp/models/DataPlatLogsParams.java
cn/trust/mobile/key/api/d/a.java

The list was compiled on the basis of information contained in the files, as well as by Inspecting the logs generated on the filesystem.

defpackage/dqk.java (ThreadDumpUtil):
found to collect data on all running processes/services; stores logs to external storage (i.e. SD card).

defpackage/ebc.java (SeerParamsService):
found to perform basic phone data enumeration.

defpackage/jru.java (ReportUtil):
found to exfiltrate location data.

defpackage/bpx.java:
This code reveals that the app checks if the system has configured a *tun0* or a *ppp0* interface. These kinds of interfaces are typically used for establishing a connection to VPN services. Certain files listed below perform extensive device information collection without obfuscation.

Affected Files:
cn/trust/mobile/key/config/DeviceInfoEntity.java

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

cn/trust/mobile/key/config/a.java
defpackage/ebc.java
defpackage/kew.java
defpackage/inp.java

The *com/alibaba/android/teleconf/service/ListenPhoneService.java* service collects phone calls statistics.

After the initial start-up of the app, the first meta-data is logged into the following path:
/data/data/cn.xuexi.android/files/data/logs/lwp_perf

The app was found to perform extensive logging and storing of a significant amount of debug information in the SD card. This is done in part via *Doraemon.registerArtifactFetcher*, in *com/alibaba/android/rimet/RimetDDContext.java*.

Although the amount of the data gathered by the app is extensive, this practice is not uncommon for commercial applications. That said, when it is performed at the government-level and given that the majority of citizens run this application, it essentially gives the government the capacity to determine - among other information - the location of every citizen at any single point in time,

SGN-01-002 File transmission and protection *(Evident)*

Q2: *“Where and how are the files transmitted?”*

Transmission of user and mobile device Information

The logged information about the phone, connection and user (see SGN-01-001) is transported to the following location via the *LWP* protocol over TLS:

lws://lws.xuexi.cn:443?sni=defaultXueXi

Once a successful connection is established, tunnel domains and cookies are received to process log data through it. The information on how the connection is established and where the data is sent was gathered from the log file below.

Log File:
/data/data/cn.xuexi.android/files/data/logs/lwp_sdk/2019-XX-XX-XX.log

In addition to this, very detailed app log reports are sent to various entities, as the examples below demonstrate.

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

Example 1: Alipay

URL:
https://mobilegw.alipay.com/mgw.htm

Related Source Code:
RPCUploaderV2, package: *com.alipay.deviceid.module.x*

Example 2: Tencent

URL:
http://wspeed.qq.com/w.cgi

Related Source Code:
openSDK_LOG.ReportManager: package defpackage

Q3: *"Does the app intentionally weaken cryptographic procedures to ensure third-party decryption?"*

Insecure cryptographic algorithms in classes for biometric data and emails

The *Xuexi Qiangguo* app suffers from multiple occurrences of insecure encryption algorithm called *DES* in Java classes handling sensitive user-information.

The *DES* (*Data Encryption Standard*)[3] algorithm is insecure due to its small key size of 56 bit which can be brute-forced in less than a week[4]. The use of this insecure algorithm in features such as *mail* or *biometrics,* as indicated by the file path and source code, could facilitate third-party's ability to decrypt sensitive information. A third-party with sufficient computation resources and a capacity to monitor network traffic of the app - such as a government agency, would be able to decrypt emails and biometric data on a mass-scale.

This could provide the opportunity to efficiently collect, map and analyze personal information, biometric data and private messages in a centralized database. The mainland Chinese media reported over 100 million registered users of the app which means that, provided that this number is true, a great many individuals could be at risk of having their sensitive information decrypted and stored.

[3] https://csrc.nist.gov/csrc/media/publications/fips/46/3/archive/1999-10-25/documents/fips46-3.pdf
[4] https://www.copacobana.org/

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

The *DES* encryption algorithm is employed in the *ak* class used by the *FaceLivenessActivity* class.

Affected Files:
com/alibaba/security/biometrics/build/ak.java
com/alibaba/security/biometrics/face/auth/FaceLivenessActivity.java

Affected Code:

```
/*The following method is responsible for encrypting images using the insecure
DES algorithm.*/
private static byte[] a(String str, byte[] bArr) {
[...]
            String a2 = isr.a(str);
            Key a3 = isr.a(Base64.decode(a2, 0));
            Cipher instance = Cipher.getInstance("DES/CBC/PKCS5Padding");
            byte[] bArr2 = new byte[8];
            while (i2 < 8 && i2 < a2.getBytes().length) {
                bArr2[i2] = a2.getBytes()[i2];
                i2++;
            }
            instance.init(1, a3, new IvParameterSpec(bArr2));
            return instance.doFinal(bArr);
[...]

/*It is assumed that the following operation prepares to derive the encryption
key from the current system time in milliseconds.*/
j2.setK(iqq.a(System.currentTimeMillis() + (Math.random() * 10000.0d)));
[...]
byte[] a2 = ((itf) akVar.j.get(i3)).a(200);

/*This operation encrypts the image.*/
byte[] a4 = a(j2.getK(), a2);
ImageResult imageResult = new ImageResult();
String str = akVar.d.getFilesDir().toString() + "/" + iqq.a("q_" + i3 + "_1") +
".jpeg";
[...]
imageResult.setP(str);

/*The following operation creates a new file based on the encrypted image
contained in variable a4.*/
     if (ist.a(new File(imageResult.getP()), a4)) {
                actionResult.addImageResult(imageResult);
            } else {
                itg.a("Save action image fail:" + imageResult);
            }
```

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

Note that the *DES* encryption algorithm is employed in the *AlimeiEncrypt* class, which is used by the *MailDetailCaptureActivity* and *MailInterfaceImpl* classes.

Affected Files:
com/alibaba/alimei/mail/AlimeiEncrypt.java
com/alibaba/alimei/mail/activity/MailDetailCaptureActivity.java
com/alibaba/alimei/cmail/impl/MailInterfaceImpl.java

Affected Code:

```
public final String a(String data, String key) {
        dex2jar6.b(dex2jar6.a() ? 1 : 0);
        try {
            Key k = a(aep.b(key));
            Cipher cipher = Cipher.getInstance("DES/CBC/PKCS5Padding");
            cipher.init(1, k, new IvParameterSpec("alimei12".getBytes()));
            return aep.a(cipher.doFinal(data.getBytes()));
        } catch (Exception e) {
            return null;
        }
    }
```

Due to the limitations of reverse-engineering in the *Xuexi Qiangguo* app, it is unclear how the mentioned objects with weakly encrypted user-data are further processed in the app. Further, no data could be acquired about a server this information is sent to. If the transmission is conducted with *Transport Layer Security,* it could be ensured that the decryption capabilities are limited to the receiving server (possibly *Alibaba*) and are not exposed in local area networks.

Q4: *“Is data dumped in the SD Card from where it could be retrieved later without even entering the PIN to unlock the device?”*

Insecure file storage

The app stores multiple files insecurely in the SD card, from which other apps can read them. For example, the following subdirectories in the SD card’s logs provides a high-level overview about the type of logs saved. Clear-text seemed as the preferred storage method encountered during this review.

Path:
/mnt/sdcard/Android/data/cn.xuexi.android/files/logs/trace/0

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

Directories:

```
alpha            crypto           H5        mini_log_performance  Resource
uicdd
AntiJackWatcher  dingtalkbase     hardware  mini_log_sdk          RuntimeConfig
uicsdk
auth             doraemon         hpm       monitor               safeTunnel
user_ct
base             FastConfigEngine ipc       no_lg_switch
SplashActivityContext  user_lg
CMail            feature          JsApi     oa                    tele_conf
WebConfig
config_switch    general          lightapp  permission            UCCore
WebException
```

SGN-01-003 Code execution and backdoors *(Evident)*

Q5: *"Can the shell commands that the app runs lead to RCE in any way?"*

The app contains code resembling a backdoor which is able to run arbitrary commands with *superuser* privileges. This is described below.

Q6: *"Has the app some kind of backdoor?"*

It was found that the application has some code for running arbitrary commands with *superuser* privileges via "*su*". However, no evidence of usage could be identified during this test and further investigation is required to unequivocally determine whether this code is used to perform malicious activities such as running arbitrary commands on the phones of citizens.

Affected File:
cn/trust/mobile/key/api/d/b.java

Affected Code (decompiled):

```
public class b {
    public C0160b f14354a = new C0160b("sh");
    public C0160b b = new C0160b("su");
….
       public Process a(String str) {
            dex2jar5.b(dex2jar5.a() ? 1 : 0);
            try {

/*The following operation assumes superuser privileges for
       commands to be executed.*/
                Process exec = Runtime.getRuntime().exec(this.b);
                DataOutputStream dataOutputStream =
                    new DataOutputStream(exec.getOutputStream());
```

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

```
/*The following operation appears to insert a string that
       should be executed on the filesystem.*/
                dataOutputStream.writeBytes("exec " + str + "\n");
                dataOutputStream.flush();
                return exec;
            } catch (Exception e) {
                return null;
            }
        }
```

Q7: *“Does the app require root access?”*

The app did not prompt the team for *root* privileges during dynamic testing. However, the backdoor-like-looking snippet above executes the *su* command to elevate privileges prior to running the actual command. Furthermore, the following code tries to determine whether *root* functionality is available on the phone.

File:
com/alibaba/lightapp/runtime/plugin/device/Base.java

Affected Code (decompiled):

```
public static boolean isRoot() {
       try {
              if (new File("/system/bin/su").exists() || new File("/system/xbin/
su").exists()) {
                     return true;
            }
```

File:
com/alipay/mobile/nebulacore/Nebula.java

Affected Code (decompiled):

```
public static boolean isRooted() {
[...]
        try {
            value =
Class.forName("android.os.SystemProperties").getMethod("get", new Class[]
{String.class}).in
voke(null, new Object[]{"ro.secure"});
        }
…
if (value != null && "0".equals(value)) {
            ret = true;
        }
…
```

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

```
        if (!ret) {
            if (new File("/system/bin/su").exists()) {
                ret = true;
            } else if (new File("/system/xbin/su").exists()) {
                ret = true;
            }
        }
```

File:
com/aliyun/security/yunceng/android/sdk/umid/c.java

Affected Code:

```
    private boolean b() {
        dex2jar9.b(dex2jar9.a() ? 1 : 0);
        String[] arr$ = {"/system/app/Superuser.apk", "/sbin/su",
"/system/bin/su", "/system/xbin/su", "/data/local/xbin/su",
"/data/local/bin/su", "/system/sd/xbin/su", "/system/bin/failsafe/su",
"/data/local/su", "/su/bin/su"};
        for (int i$ = 0; i$ < 10; i$++) {
            if (new File(arr$[i$]).exists()) {
                return true;
            }
        }
        return false;
    }
```

Without context, it seems difficult to justify why an educational app requires code that looks like a backdoor. Especially if such backdoor could potentially run arbitrary commands on citizen phones with *superuser* privileges.

SGN-01-004 Obfuscation for hiding functionality *(Proven)*

Q8: *“Does the app use obfuscation techniques to hide code and and if yes for which files and directories?”*

The APK reviewed in this test employed strong anti-reversing techniques. This was quickly noticed early in the analysis because it prevented most decompilers from decoding many files.

Decompilation error examples:

```
WARN:           Method method_1639
(Ljava/lang/String;Ljava/lang/String;)Ljava/lang/String; couldn't be decompiled.
java.lang.RuntimeException: parsing failure!
[...]
INFO:  Decompiling class class_16
```

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

```
java.lang.IllegalStateException: Invalid BootstrapMethods attribute entry: 2
additional arguments required for method
java/lang/invoke/StringConcatFactory.makeConcatWithConstants, but only 1
specified.
[...]
WARN:        Method method_2146
(Laic;Lcom/alibaba/android/dingtalkbase/DingtalkBaseActivity;)Z couldn't be
written.
java.lang.NullPointerException
[...]
Processing anet.channel.status.NetworkStatusHelper
WARN:           Heavily obfuscated exception ranges found!
[...]
WARN:  Nested class com/alibaba/android/rimet/biz/SplashActivity$a missing!
[...]
Method method_1717 (Lcom/alibaba/alimei/emailcommon/Account;)Labq; couldn't be
decompiled.
[...]
INFO:  Decompiling class ali/mmpc/util/MmpcUtils
WARN:           Heavily obfuscated exception ranges found!
WARN:           Heavily obfuscated exception ranges found!
[...]
Method fromFile (Ljava/io/File;)Lcom/taobao/android/runtime/OatFile; couldn't be
decompiled.
java.lang.NullPointerException
```

This *com.taobao* component also creates files in a hidden directory on the SD card.

Affected Files:
*/mnt/sdcard/**.com.taobao.dp**/dd7893586a493dc3*
*/mnt/sdcard/**.com.taobao.dp**/hid.dat*

The *packer* in use appears to be *Ali,* as revealed by the following artifact in the decompiled sources.

File:
aliprotect.dat

Contents:

```
{"enhance_file_timestamp":"flag","enhancetype":1}
```

This matches the file signatures of the *Ali packer*, as documented in the paper titled *Understanding Android Obfuscation Techniques: A Large-Scale Investigation in the*

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

Wild[5] On the decompiled source code itself, several instances of obfuscation being in use were identified and are discussed next.

Example 1: Invocation of base64-encoded method

File:
com/alipay/deviceid/module/x/e.java

Affected Code:
```
return (File) Environment.class.getMethod(new
String(g.a("Z2V0RXh0ZXJuYWxTdG9yYWdlRGlyZWN0b3J5")), new Class[0]).invoke(null,
new Object[0]);
```

Encoded String	Decoded String
Z2V0RXh0ZXJuYWxTdG9yYWdlRGlyZWN0b3J5	getExternalStorageDirectory

Example 2: Search for hexadecimal-encoded app prefixes on the phone

Cure53 uncovered that the app tries to find specific running applications for which searchable prefixes are encoded in hexadecimal. This happens when system information is being collected and can be seen in the file below.

File:
com/alipay/deviceid/module/x/c.java

Affected Code (decompiled):
```
String str = new
String(i.a("636f6d2e74656e63656e742e6d6f62696c657171692c636f6d2e74656e63656e742e
71716c6976652c636f6d2e6e642e616e64726f69642e70616e6461686f6d652e68642c636f6d2e74
6f[...]
```

Decoded String:
```
com.tencent.mobileqqi,com.tencent.qqlive,com.nd.android.pandahome.hd,com.tongche
nglove.main,com.gdhbgh.activity,com.dragon.android.pandaspace,com.handsgo.jiakao
.android.kemu2,com.soufun.app,com.google.android.apps.maps,com.job.android,com.t
encent.qqpim,com.baidu.homework,com.storm.smart,ly.pp.justpiano,com.android.medi
acenter,com.android.cheyoohdrive,com.ymall.presentshop,com.eg.android.AlipayGpho
ne,com.miguo.ui,com.pplive.androidphone,com.android.lexun,com.redirectin.rockpla
yer.android.unified,com.sz.order,cn.mucang.kaka.android,com.qijia.o2o,com.chinai
deal.bkclient.tabmain,[...]
```

[5] https://arxiv.org/pdf/1801.01633.pdf

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

The list includes 960 applications, among which can be found:

Games:

com.lonelycatgames.Xplore
com.hupu.games
org.sbtools.gamehack
com.google.android.play.games
com.mobile17173.game
com.gamestar.pianoperfect
com.hjwordgames
com.anzogame.lol
com.lilysgame.calendar
com.superevilmegacorp.game

Navigator Apps:

com.baidu.navi
com.autonavi.xmgd.navigator.toc
com.mapbar.android.trybuynavi
cld.navi.mainframe
com.autonavi.xmgd.navigator
cn.com.tiros.android.navidog
com.uu.uunavi
com.autonavi.cmccmap
cn.com.tiros.android.navidog4x
com.staralliance.navigator

Travel / Trip Apps:

com.taobao.trip
com.tripadvisor.tripadvisor
ctrip.android.view
cn.safetrip.edog
com.breadtrip
com.tripadvisor.tripadvisor.daodao
com.umetrip.android.msky.app

Chat Apps:

com.kakao.talk
com.google.android.talk
com.talkweb.nciyuan
com.lenovo.videotalk.phone
com.google.android.marvin.talkback
com.hujiang.cctalk

Card Apps:

com.caimi.creditcard
com.imohoo.favorablecard
com.jiongji.andriod.card

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

```
cn.andson.cardmanager
```

Payment Apps:

```
com.eg.android.AlipayGphone
com.alipay.android.app
com.alipay.android.client.pad
com.suixingpay
com.chinatelecom.bestpayclient
com.tenpay.android
com.unionpay.uppay
com.iboxpay.iboxpay
com.weibopay.mobile
com.umpay.upay.wallet
com.wangyin.payment
cn.unicompay.wallet
com.unionpay
com.citicbank.cyberpay.ui
com.unicom.wopay
com.netease.epay
```

It appears as if the application is somehow attempting to find which popular apps are installed on the phone.

SGN-01-005 Collaboration with external companies *(Evident)*

Q9: *"Which external companies help build and maintain this app?"*

It appears that *Alibaba,* the official maintainer of the *Xuexi Qiangguo* app, is the driving force behind the questionable coding practices analyzed for this report.

The vast amount of device- and meta-data collected by the app originates from packages in the namespace of *Alibaba.* Moreover, the data is sent to servers presumably controlled by *Alibaba (xuexi.cn, alipay.com)*. However, some information was also sent to servers that are likely to be controlled by Tencent (*qq.com*).

The fact that insecure cryptographic algorithms like *DES* are used in a package provided by *Alibaba* suggests that *Alibaba* is actively participating in weakening the security of the *Xuexi Qiangguo* app. The code does not give away information that could indicate if these issues have been facilitated by Alibaba itself or mandated by a third-party as an intentional design weakness. However, the fact that secure cryptographic algorithms such as *AES* and *3DES* are employed in other parts of the app, raises the question as to why these parts of the app have not been treated the same but rather weakened.

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

The app uses the *packer* Ali, presumably developed by *Alibaba*, to obfuscate code functionality regarding potential code execution or weak encryption practices.

SGN-01-006 Similarities & differences to other spy apps *(Unclear)*

Q10: *"What is the relationship between the Xuexi Qiangguo* app*, BXAQ and JingWang if any?"*

The app is very different in nature when compared to BXAQ or JingWang.

- BXAQ is an app that police officers can install on a confiscated phone to collect various forms of data and uninstall before returning it to the affected individual.[6]
- JingWang is an app that the police required citizens to install in Xinjiang to collect information on the device. It was used to block certain sites from being visited and forbade installation of some applications.[7]
- What these apps have in common is that users are installing them/have them installed involuntarily. Further, both are specifically targeting the Xinjiang region.

In contrast, the *Xuexi Qiangguo* app is a publicly advertised app for the general population in the entire mainland China. It appears that the *Xuexi Qiangguo* app is trying to work within the boundaries of plausible deniability, as it could at least partially claim that the collection of device information (SGN-01-001) is merely done for statistical purposes and to improve user-experience. However, the unusual coding practices and obfuscated parts of the app mentioned in SGN-01-002, SGN-01-003 and SGN-01-004 strongly suggest that the app serves another purpose. Said other aim is not publicly advertised.

In conclusion, although the apps share shady coding practices that indicate HR violations, they use different means to achieve their objectives. This analysis did not find that the *Xuexi Qiangguo* app uses the same code or the same servers as the BXAQ or JingWang apps.

[6] https://cure53.de/analysis-report_bxaq.pdf
[7] https://archive.opentech.fund/sites/default/files/attachments/otf_jingwang_report_final.pdf

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

Conclusions

This technical analysis and review of the “Study the Great Nation” /*Xuexi Qiangguo* mobile application has demonstrated that the concerns expressed by Human Rights Watch are valid. Carried out by Cure53 in close collaboration with the OTF team, this August 2019 the project sheds light on six 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 examined 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 actions concerning a specific group (or groups) of citizens. Although some of the collection of meta-data and device information could be legitimized as being aggregated for statistical reasons or software improvement, it is questionable if this is necessary for an app that claims to be educational in nature.

Adding to the above, the arguably intentional use of weak encryption in the code which appears to be related to biometric data and email encryption cannot be set aside as any sort of requirement of education-driven goals. 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 violation of human rights.

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 SGN-01-001). Even more concerning is the use of the insecure *DES* encryption algorithm for biometric data and emails (see SGN-01-002) and the apparent preparations that could enable arbitrary code execution on citizens’ devices observed in SGN-01-003. However, the broader context of the evaluated coding practices remains unknown due to extensive obfuscation measures in the affected classes (see SGN-01-004). The analysis strongly indicates that the official maintainer, namely *Alibaba*, appears to be the architect of the questionable artifacts on the scope. Conversely, as there were no intersections noted with BXAQ and JingWang, it appears that those apps have different origin.

The app avails of significant, privacy-sensitive permissions and functionality, such as location, face recognition, microphone and camera access, call log and contact processing. Sharing many of these features is required by the app (i.e. for its chat integration with *DingTalk*). In essence, while the app has all the capabilities it would need for more invasive mass data collection, the scale and potential to exploit this

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

through hidden functionality in the obfuscated code should be the subject of further investigation.

In a broader sense, the application's functionality leads Cure53 to believe that violations of human rights are indeed taking place. Especially the items noted with *“Evident”* or *“Proven”* markers 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 judgements 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 Dan Blah, Sarah & Adam from the OTF team for their excellent project coordination, support and assistance, both before and during this assignment.