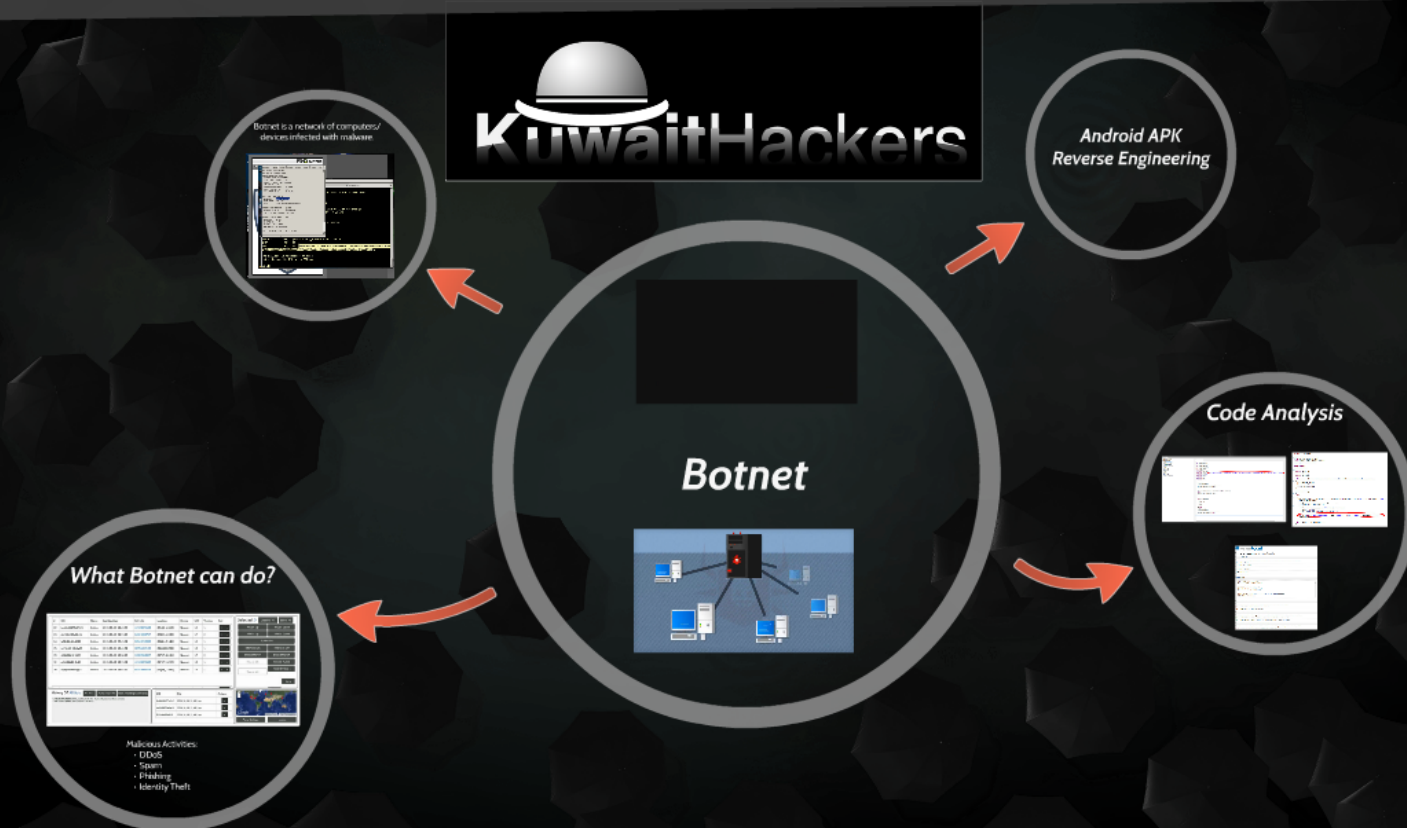
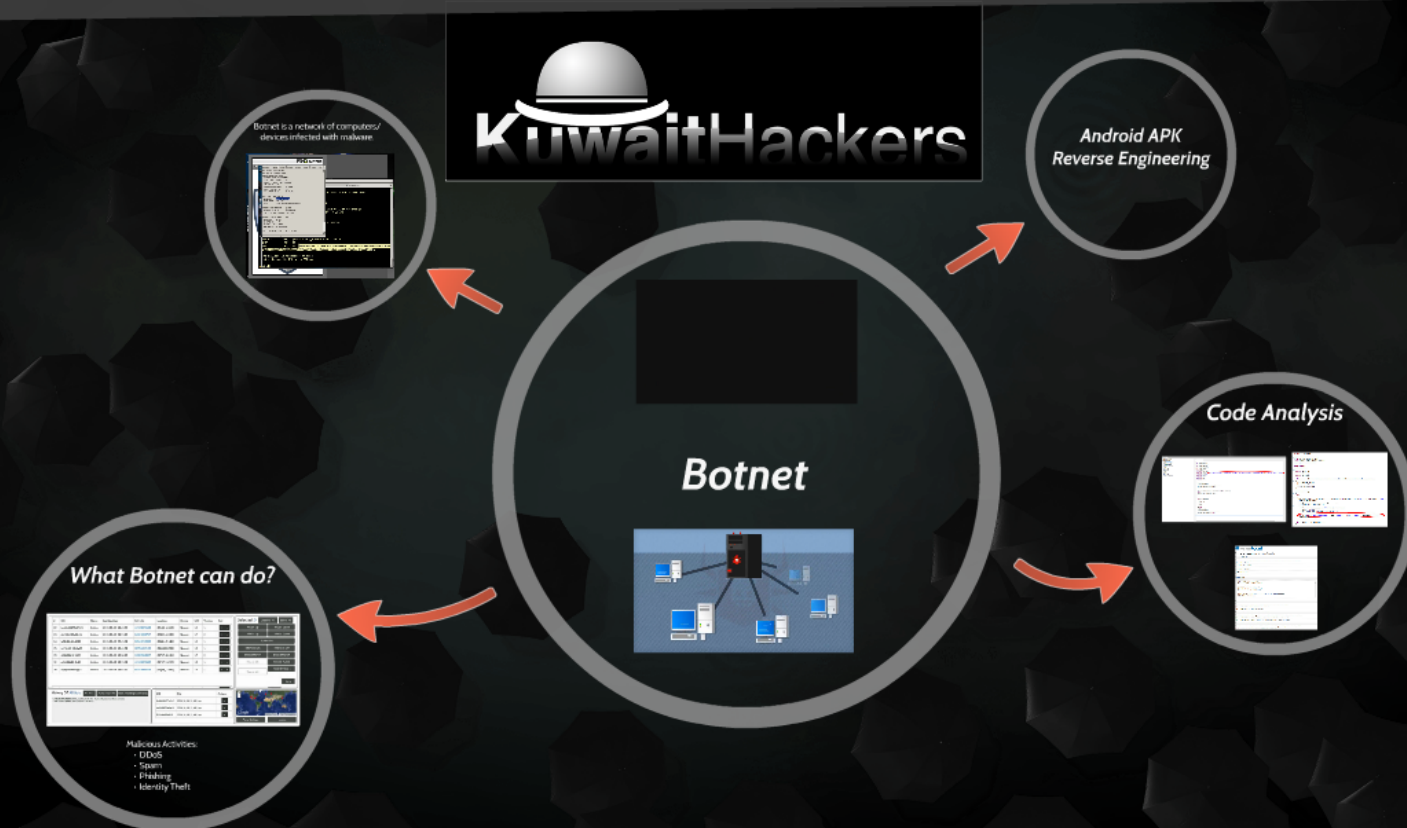
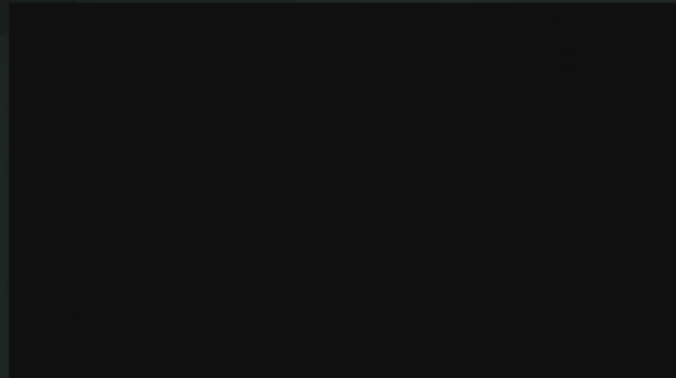


# Android Botnet Analysis

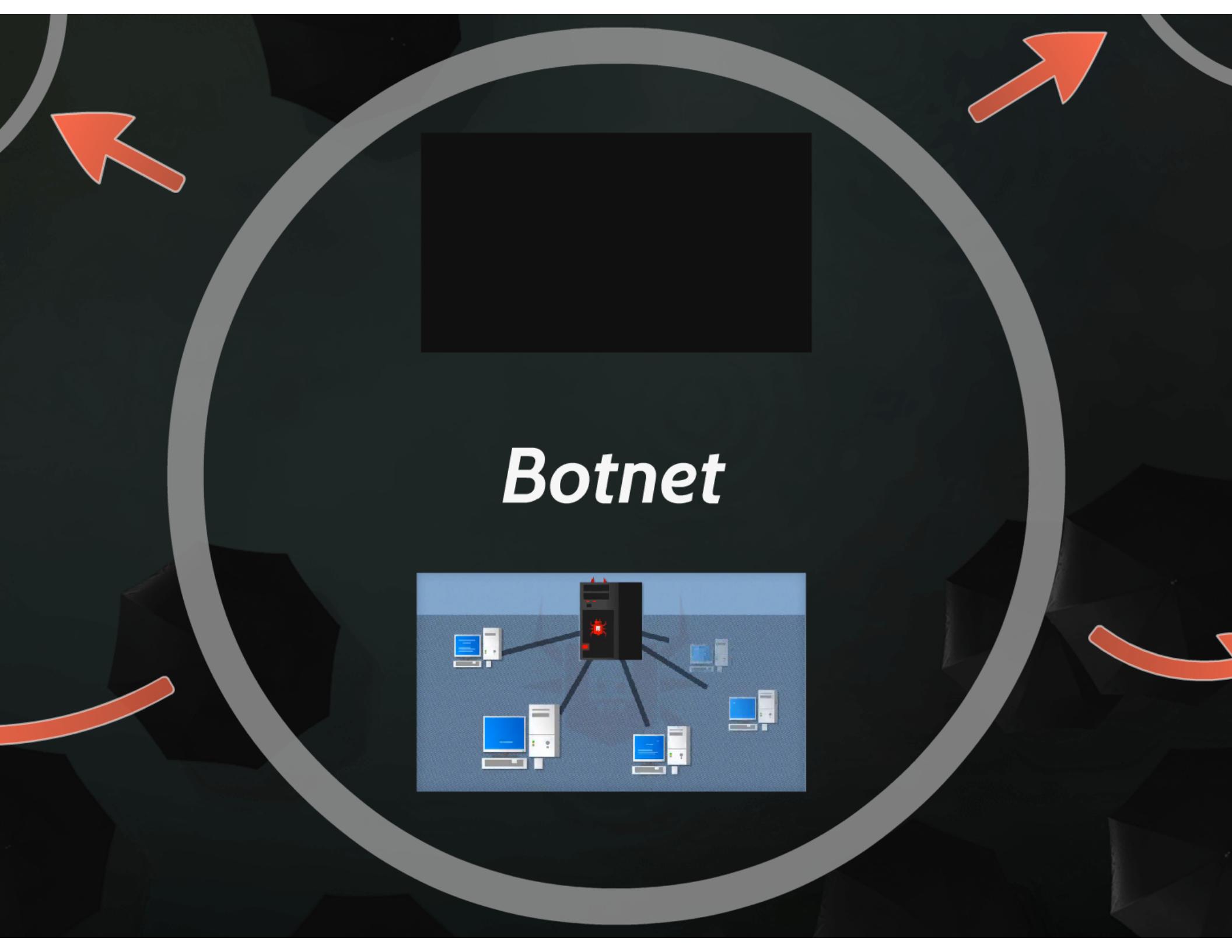
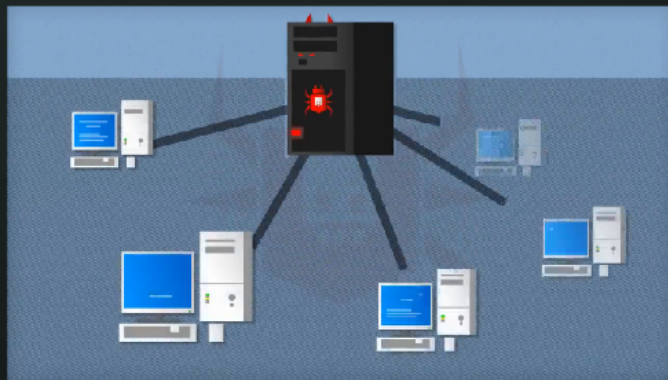


# Android Botnet Analysis

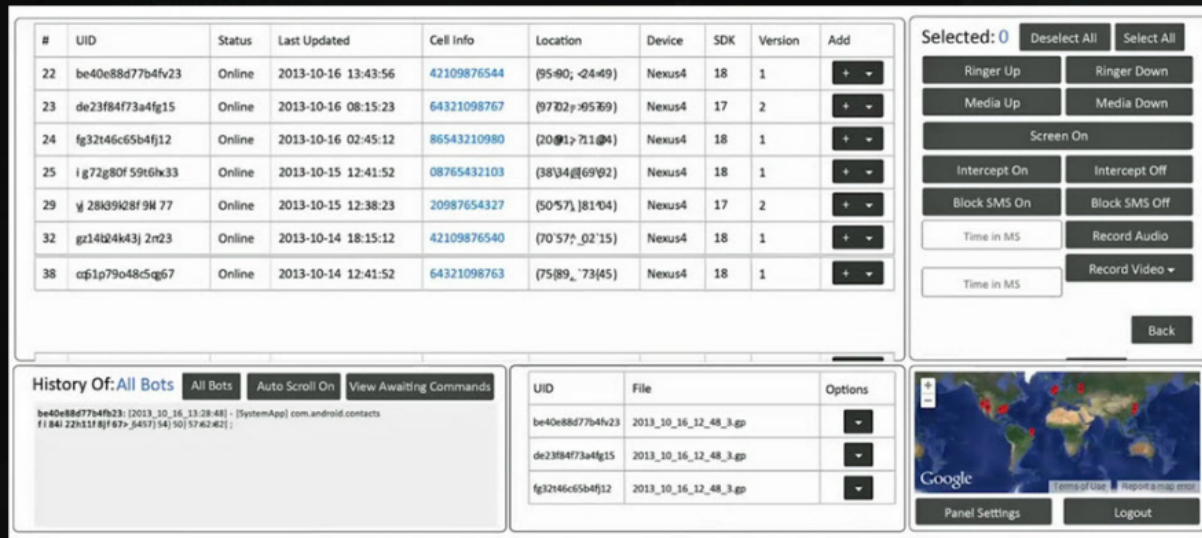




# *Botnet*



# What Botnet can do?



The screenshot displays a web-based interface for managing a botnet. It features a main table of bots with columns for ID, UID, Status, Last Updated, Cell Info, Location, Device, SDK, and Version. Below the table are several control panels: a 'Selected: 0' panel with buttons for 'Deselect All' and 'Select All', and a 'Control' panel with buttons for 'Ringer Up/Down', 'Media Up/Down', 'Screen On', 'Intercept On/Off', 'Block SMS On/Off', 'Time in MS', 'Record Audio', and 'Record Video'. A 'History Of: All Bots' panel shows a list of bots with their UID, File, and Options. A map of the world is also visible, showing the locations of the bots. A large red arrow points to the right from the map area.

#	UID	Status	Last Updated	Cell Info	Location	Device	SDK	Version	Add
22	be40e88d77b4fv23	Online	2013-10-16 13:43:56	42109876544	(95°00', 42°49')	Nexus4	18	1	+ -
23	de23f84f73a4fg15	Online	2013-10-16 08:15:23	64321098767	(97°02', 95°09')	Nexus4	17	2	+ -
24	fg32t46c65b4fg12	Online	2013-10-16 02:45:12	86543210980	(20°01', 71°04')	Nexus4	18	1	+ -
25	ig72g80f59t6hx33	Online	2013-10-15 12:41:52	08765432103	(38°34', 69°02')	Nexus4	18	1	+ -
29	y28k99k28f9h77	Online	2013-10-15 12:38:23	20987654327	(50°57', 181°04')	Nexus4	17	2	+ -
32	gz14t24k43j2n23	Online	2013-10-14 18:15:12	42109876540	(70°57', 02°15')	Nexus4	18	1	+ -
38	cf1p79o48c5q67	Online	2013-10-14 12:41:52	64321098763	(75°09', 73°45')	Nexus4	18	1	+ -

Selected: 0 Deselect All Select All

Ringer Up Ringer Down  
Media Up Media Down  
Screen On  
Intercept On Intercept Off  
Block SMS On Block SMS Off  
Time in MS Record Audio  
Time in MS Record Video  
Back

History Of: All Bots All Bots Auto Scroll On View Awaiting Commands

be40e88d77b4fv23: [2013\_10\_16\_13:28:48] - [SystemApp] com.android.contacts  
f184122b51f8f87>3451754]50]57&2&2):

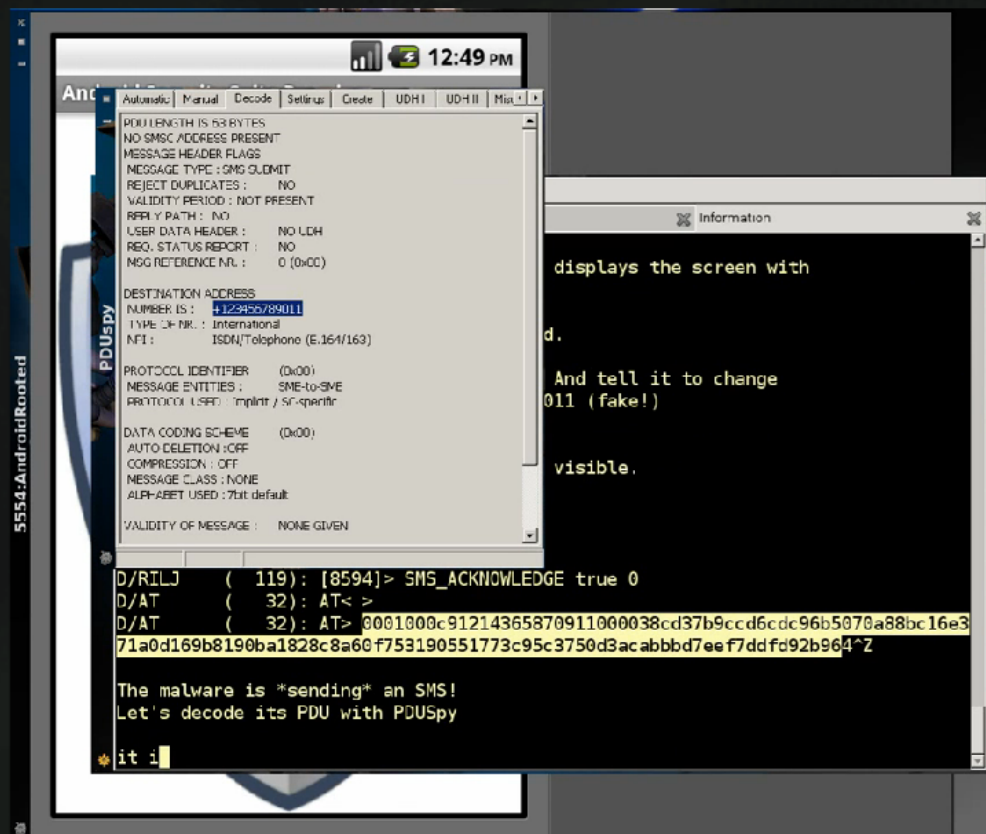
UID	File	Options
be40e88d77b4fv23	2013_10_16_12_48_3_gp	-
de23f84f73a4fg15	2013_10_16_12_48_3_gp	-
fg32t46c65b4fg12	2013_10_16_12_48_3_gp	-

Panel Settings Logout

## Malicious Activities:

- DDoS
- Spam
- Phishing
- Identity Theft

Botnet is a network of computers/  
devices infected with malware.



The screenshot shows an Android terminal window with a title bar that includes 'Android', 'Automatic', 'Manual', 'Decode', 'Settings', 'Create', 'UDH1', 'UDH2', and 'Misc'. The terminal output is as follows:

```
5554:AndroidRooted
PDUspy
PDU LENGTH IS 83 BYTES
NO SMSC ADDRESS PRESENT
MESSAGE HEADER FLAGS
MESSAGE TYPE : SMS QUDMIT
REJECT DUPLICATES : NO
VALIDITY PERIOD : NOT PRESENT
REPLY PATH : NO
USER DATA HEADER : NO LDH
REQ. STATUS REPORT : NO
MOG REFERENCE N.L. : 0 (0x0C)

DESTINATION ADDRESS
NUMBER IS : +12955789011
TYPE OF NR. : International
NFI : ISDN/Telephone (E.164/163)

PROTOCOL IDENTIFIER : (0x00)
MESSAGE ENTITIES : SME-to-SME
PROTOCOL USER : Implrtr / SCS-qrtrtr

DATA CODING SCHEME : (0x00)
AUTO DELETION : OFF
COMPRESSION : OFF
MESSAGE CLASS : NONE
ALPHABET USED : 7bit default


VALIDITY OF MESSAGE : NONE GIVEN

D/RILJ ( 119): [8594]> SMS_ACKNOWLEDGE true 0
D/AT ( 32): AT< >
D/AT ( 32): AT> 0001000c91214365870911000038cd37b9ccd6cdc96b5070a88bc16e3
71a0d169b8190ba1828c8a60f753190551773c95c3750d3acabbbd7ee7ddfc92b964^Z

The malware is *sending* an SMS!
Let's decode its PDU with PDUspy
it :
```

Information  
displays the screen with  
d.  
And tell it to change  
011 (fake!)  
visible.

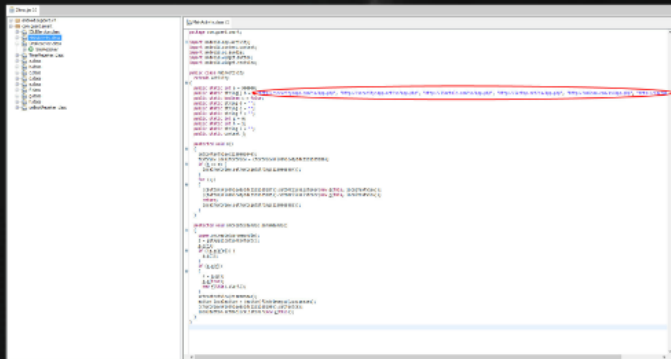




***Android APK  
Reverse Engineering***



# Code Analysis



```
package com.guard.start;

import android.content.Context;
import android.os.PowerManager;
import android.os.PowerManager.WakeLock;
import android.util.Log;

public class g
    extends Thread
{
    private String h;
    private String i;
    private Context j;

    public g(Context paramContext, Context paramContext, String paramString, String paramString)
    {
        this.g = paramContext;
        this.h = paramContext;
        this.i = paramString;
        this.j = paramContext;
    }

    public void run()
    {
        PowerManager.WakeLock localWakeLock = ((PowerManager)this.g.getSystemService("power")).newWakeLock(2, "g");
        localWakeLock.acquire();
        String url = URLDecoder.decode(this.i, "utf-8");
        if (localWakeLock.isAcquired() != 0)
        {
            String str = Integer.toHexString(1 * 10 * this.h + "0" + str);
            int localh = str.length();
            String str2 = String.format("%0" + localh + "d", new Object[] { Integer.parseInt(str) });
            Log.v("g", this.g, "services android" + str1 + "sender" + this.g + "biops" + "downloaded" + "k");
            localWakeLock.release();
        }
        catch (Exception localException) {}
    }
}
```


```
package com.guard.start;

import android.content.Context;
import android.os.PowerManager;
import android.os.PowerManager.WakeLock;
import android.util.Log;

public class g
    extends Thread
{
    private String h;
    private String i;
    private Context j;

    public g(Context paramContext, Context paramContext, String paramString, String paramString)
    {
        this.g = paramContext;
        this.h = paramContext;
        this.i = paramString;
        this.j = paramContext;
    }

    public void run()
    {
        try
        {
            PowerManager.WakeLock localWakeLock = ((PowerManager)this.g.getSystemService("power")).newWakeLock(2, "g");
            localWakeLock.acquire();
            String url = URLDecoder.decode(this.i, "utf-8");
            if (localWakeLock.isAcquired() != 0)
            {
                String str = Integer.toHexString(1 * 10 * this.h + "0" + str);
                int localh = str.length();
                String str2 = String.format("%0" + localh + "d", new Object[] { Integer.parseInt(str) });
                Log.v("g", this.g, "services android" + str1 + "sender" + this.g + "biops" + "downloaded" + "k");
                localWakeLock.release();
            }
            catch (Exception localException) {}
        }
    }
}
```



**virustotal**  
securitysapp.com domain information

File hashes (SHA-256) of the address: securitysapp.com. This domain has been seen to resolve to the following IP addresses:

IP Address	First Seen	Last Seen
207.172.112.113	2015-04-28	2015-04-28
207.172.112.114	2015-04-28	2015-04-28
207.172.112.115	2015-04-28	2015-04-28
207.172.112.116	2015-04-28	2015-04-28
207.172.112.117	2015-04-28	2015-04-28
207.172.112.118	2015-04-28	2015-04-28

**Whois lookup**

```
Domain Name: SECURITYSAPP.COM
Registrar: GODADDY (GODADDY.COM, LLC)
Creation Date: 2015-04-28 00:00:00
Expiration Date: 2016-04-28 00:00:00
DNS Servers: NS1.SECURITYSAPP.COM
NS2.SECURITYSAPP.COM
NS3.SECURITYSAPP.COM
NS4.SECURITYSAPP.COM
Name Server: NS1.SECURITYSAPP.COM
Status: clientTransferProhibited https://www.icann.org/epp/

Last updated on 2015-04-28 00:00:00 UTC
```

**Link Analysis**

Link	First Seen	Last Seen
http://www.securitysapp.com	2015-04-28	2015-04-28
http://www.securitysapp.com	2015-04-28	2015-04-28
http://www.securitysapp.com	2015-04-28	2015-04-28

**Link Analysis**

Link	First Seen	Last Seen
http://www.securitysapp.com	2015-04-28	2015-04-28
http://www.securitysapp.com	2015-04-28	2015-04-28
http://www.securitysapp.com	2015-04-28	2015-04-28



# Android Botnet Analysis

