# The Dark Side of Android Applications

**Michael Spreitzenbarth**

**Lehrstuhl für Informatik 1**
**Universität Erlangen-Nürnberg**

michael.spreitzenbarth@cs.fau.de

**OWASP**
07.11.2012



**The OWASP Foundation**
http://www.owasp.org

# Agenda

✓ Aktueller Trend im Bereich Android Malware

✓ Malware Analyse-Systeme

✓ Android Werbenetzwerke

✓ Ausblick

# Android Malware

Aktueller Trend und Überblick

# Aktueller Trend im Bereich Android Malware

- stark anwachsendes Bedrohungspotential

- Wachstumsraten von mehr als 3.000% [1]

- ca. 15.000 neue bösartige Apps
  in weniger als 4 Monaten erschienen

- aktuell über 150 Malware-Familien bekannt

# Malware Studie

- Analyse von über 300.000 Apps
  - Google Play und Third-Party Märkte

# Malware Studie - Ergebnisse

- über 8.000 bösartige Apps gefunden

- zu 152 Familien gruppiert

- hauptsächlich Fraud- und Spyware

# Malware Studie - Ergebnisse

- 57% stehlen persöhnliche Daten wie IMEI, GPS-Koordinaten, Adressbucheinträge, usw...

- 45% versenden Premium-SMS

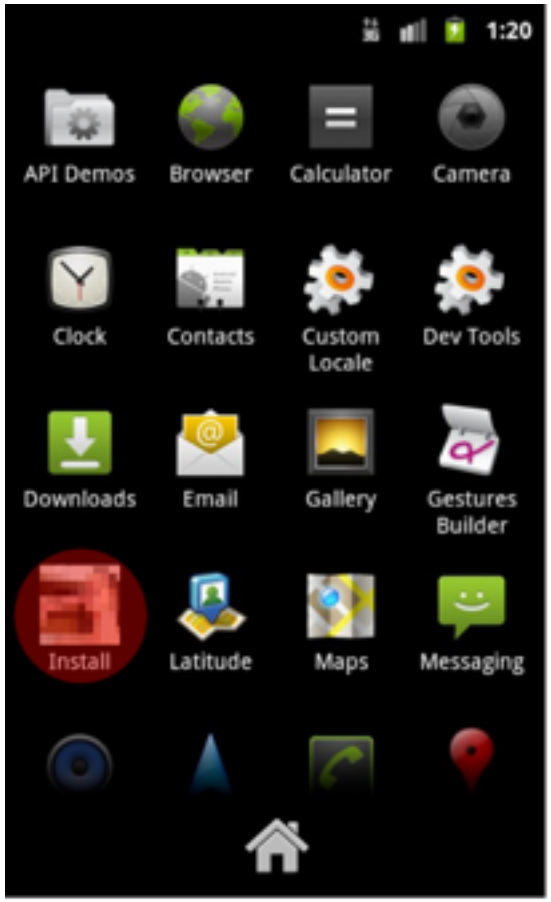
- 20% verbinden sich zu C&C-Servern (Botnets)

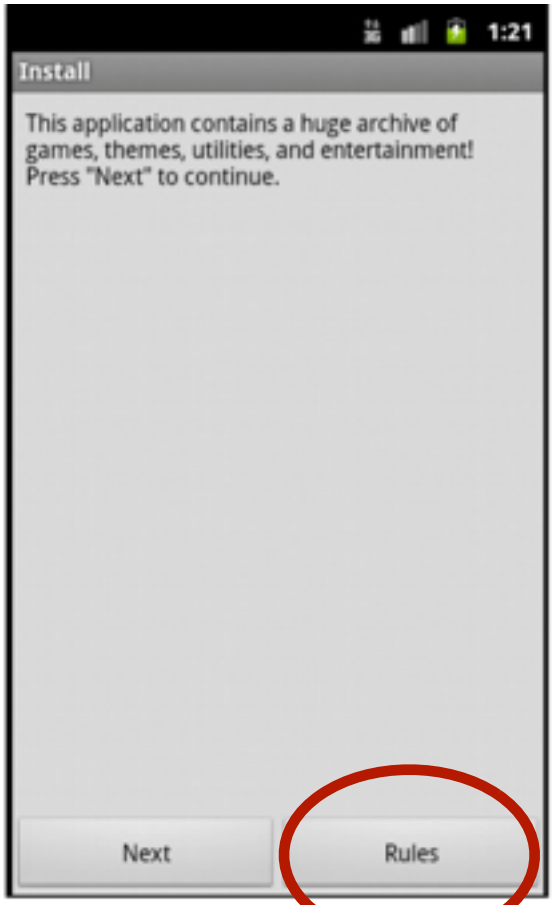
- ca. 10% kommen inkl. Root-Exploit

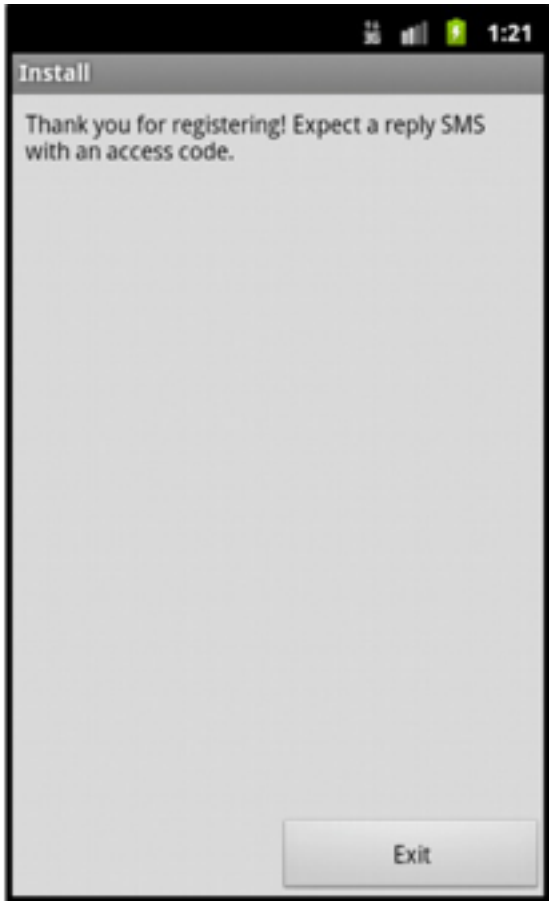
# Beispiel

Android.FakeRegSMS

# FakeRegSMS


1:20
API Demos
Browser
Calculator
Camera
Clock
Contacts
Custom Locale
Dev Tools
Downloads
Email
Gallery
Gestures Builder
Install
Latitude
Maps
Messaging

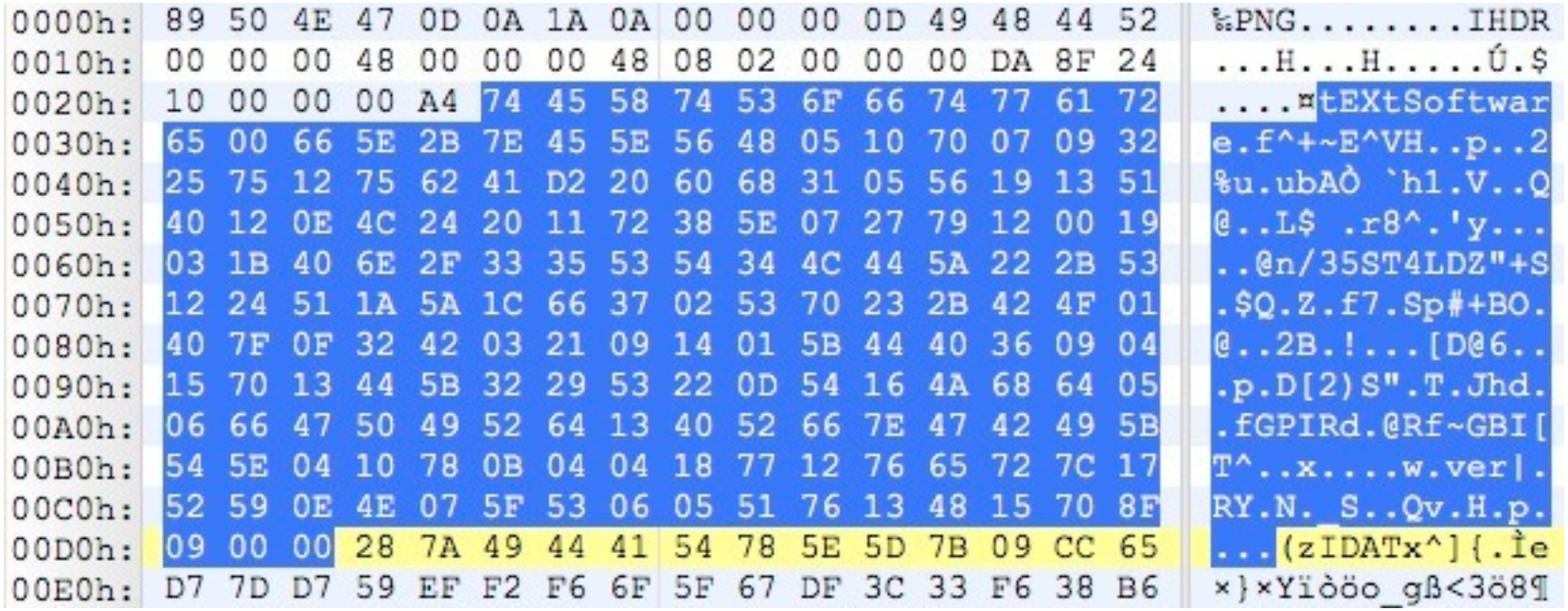

1:21
Install
This application contains a huge archive of games, themes, utilities, and entertainment! Press "Next" to continue.
Next
Rules


1:21
Install
Thank you for registering! Expect a reply SMS with an access code.
Exit

# FakeRegSMS

```
i f ( k < 0)
    throw new IOException ( ”Chank tEXt not found in png” ) ;
```

# FakeRegSMS



```
0000h: 89 50 4E 47 0D 0A 1A 0A 00 00 00 0D 49 48 44 52  ‰PNG........IHDR
0010h: 00 00 00 48 00 00 00 48 08 02 00 00 00 DA 8F 24  ...H...H.....Ú.$
0020h: 10 00 00 00 A4 74 45 58 74 53 6F 66 74 77 61 72  ....¤tEXtSoftwar
0030h: 65 00 66 5E 2B 7E 45 5E 56 48 05 10 70 07 09 32  e.f^+~E^VH..p..2
0040h: 25 75 12 75 62 41 D2 20 60 68 31 05 56 19 13 51  %u.ubAÒ `h1.V..Q
0050h: 40 12 0E 4C 24 20 11 72 38 5E 07 27 79 12 00 19  @..L$ .r8^.'y...
0060h: 03 1B 40 6E 2F 33 35 53 54 34 4C 44 5A 22 2B 53  ..@n/35ST4LDZ"+S
0070h: 12 24 51 1A 5A 1C 66 37 02 53 70 23 2B 42 4F 01  .$Q.Z.f7.Sp#+BO.
0080h: 40 7F 0F 32 42 03 21 09 14 01 5B 44 40 36 09 04  @..2B.!...[D@6..
0090h: 15 70 13 44 5B 32 29 53 22 0D 54 16 4A 68 64 05  .p.D[2)S".T.Jhd.
00A0h: 06 66 47 50 49 52 64 13 40 52 66 7E 47 42 49 5B  .fGPIRd.@Rf~GBI[
00B0h: 54 5E 04 10 78 0B 04 04 18 77 12 76 65 72 7C 17  T^..x....w.ver|.
00C0h: 52 59 0E 4E 07 5F 53 06 05 51 76 13 48 15 70 8F  RY.N._S..Qv.H.p.
00D0h: 09 00 00 28 7A 49 44 41 54 78 5E 5D 7B 09 CC 65  ...(zIDATx^]{.Ìe
00E0h: D7 7D D7 59 EF F2 F6 6F 5F 67 DF 3C 33 F6 38 B6  ×}×Yïòöo_gß<3ö8¶
```

Verschlüsseltes Code-Fragment im App-Icon

# FakeRegSMS

```
#!/usr/bin/python
key = "f_+wqlfh4_@312!@#DSAD_fh8w3hf43f@#$!_r43"
length = len(key)
obfuscatedData =
    "\x66\x5E\x2B\x7E\x45\x5E\x56\x48\x05\x10\x70\x07\x09\x32\x25\x75\x12\x75
\x62\x41\xD2\x20\x60\x68\x31\x05\x56\x19\x13\x51\x40\x12\x0E\x4C\x24\x20\x11\x72
\x38\x5E\x07\x27\x79\x12\x00\x19\x03\x1B\x40\x6E\x2F\x33\x35\x53\x54\x34\x4C\x44
\x5A\x22\x2B\x53\x12\x24\x51\x1A\x5A\x1C\x66\x37\x02\x53\x70\x23\x2B\x42\x4F\x01
\x40\x7F\x0F\x32\x42\x03\x21\x09\x14\x01\x5B\x44\x40\x36\x09\x04\x15\x70\x13\x44
\x5B\x32\x29\x53\x22\x0D\x54\x16\x4A\x68\x64\x05\x06\x66\x47\x50\x49\x52\x64\x13
\x40\x52\x66\x7E\x47\x42\x49\x5B\x54\x5E\x04\x10\x78\x0B\x04\x04\x18\x77\x12\x76
\x65\x72\x7C\x17\x52\x59\x0E\x4E\x07\x5F\x53\x06\x05\x51\x76\x13\x48\x15\x70\x8F
\x09"
unObfuscatedData = ""
for x, y in enumerate(obfuscatedData):
    keyIndex = x % length
    unObfuscatedData = unObfuscatedData + chr(ord(y) ^ ord(key[keyIndex]))
print "unobfuscated_data:_" + unObfuscatedData
```

Entschlüsselung des Code-Fragmentes mit dem key aus dem Quellcode der App

# FakeRegSMS

Entschlüsseltes Codefragment:

420 1004851XX? requestNo1 maxRequestNoauto costLimit150 costLimitPeriod8640 smsDelay15 sms-Data!l5872600885697126387416947526760l4P?=

# FakeRegSMS

```
D/SMS     (  161): SMS send size=0time=1328880622092
D/RILJ    (  161): [0077]> SEND_SMS
D/RIL     (   32): onRequest: SEND_SMS
D/AT      (   32): AT> AT+CMGS=51
D/AT      (   32): AT< >
D/AT      (   32): AT>
    000100048115XX00002f34190c1483c1683810bb86bbc96c30180e57b3e
    56e31996d86bbd162b61ced5693d96e36181b1683c100^Z
D/AT      (   32): AT< +CMGS: 0
D/AT      (   32): AT< OK
D/RILJ    (  161): [0077]< SEND_SMS { messageRef = 0, errorCode = 0, ackPdu =
    null}
D/SMS     (  161): SMS send complete. Broadcasting intent: null
```

Log-Eintrag des Telefons

# FakeRegSMS

- erstmals Ende 2011 gefunden

- versendet Premium-SMS im Hintergrund

- setzt Steganographie zum Verschleiern ein

- versteckte Codefragmente im App-Icon

# Android Analyse-Systeme

Andrubis & Mobile-Sandbox

# Andrubis [2]

- setzt auf DroidBox [7] für Android 2.2

- statische Analyse mit Hilfe von Androguard [8]

- Überwachung von Netzwerkverkehr

- Überwachung der Aufrufe von nativen Bibliotheken

# Mobile-Sandbox [3]


- setzt auf modifiziertes DroidBox [7]

- unterstützt Android 2.3.7 und Android 4.1

- statische Analyse mit Hilfe von Code-Review

- Überwachung von Netzwerkverkehr

- Überwachung von nativen Bibliotheken mit ltrace

- Anbindung an VirusTotal [9]

# Statische Analyse

- betrachtet Permissions und Intents

- häufig Code-Review um potentiell gefährliche Methoden oder Funktions-Aufrufe zu finden

- erkennt über-/unterprivilegierte Apps

- setzt häufig auf Muster- und Signaturerkennung

# Dynamische Analyse

- setzt auf Taint-Tracking

- interagiert mit der App in einer abgeschotteten Umgebung = Sandbox

- überwacht Daten, die das Gerät verlassen

- basiert meist auf TaintDroid [4]

# Android Werbenetzwerke

Überblick und Gefahren

# Überblick über bekannte Werbenetzwerke

| Ad Library (version) | INTERNET | ACCESS_NETWORK_STATE | READ_PHONE_STATE | ACCESS_LOCATION | CAMERA | CALL_PHONE | WRITE_EXTERNAL_STORAGE | READ_CALENDAR | WRITE_CALENDAR | READ_CONTACTS | WRITE_CONTACTS | SEND_SMS | RECEIVE_BOOT_COMPLETE | GET_ACCOUNTS | READ_LOGS | ACCESS_WIFI_STATE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| adfonic (1.1.4) | R | R | | R | | | | | | | | | | | | |
| admob (4.3.1) | R | R | | | | | | | | | | | | | | |
| airpush (2-2012) | R | R | R | O | | | | | | | | | R | | | |
| buzzcity (1.0.5) | R | | R | | | R | | | | | | | | | | |
| greystripe (1.6.1) | R | R | R | | | | | | | | | | | | | |
| inmobi (3.0.1) | R | O | | O | | O | | X | X | | | | | | | |
| jumptap (2.3) | R | R | R | O | | | | | | | | | | | | |
| millennialmedia (4.5.1) | R | R | R | | O | | R | | | | | | | | | |
| mobclix (3.2.0) | R | O | R | X | X | | | X | X | X | X | | | X | | |
| mOcean (2.9.1) | R | R | R | O | O | O | O | O | O | | | O | | | O | |
| smaato (2.5.4) | R | R | R | O | | | | | | | | | | | | |
| vdopia (2.0.1) | R | R | | | | | | | | | | | | | | |
| youmi (3.05) | R | R | R | R | | | R | | | | | | | | | X |

[5]

# mOcean

- Zugriff auf aktuelle Ortsdaten

- Zugriff auf Kamera, Kalender und SD-Karte

- darf SMS versenden und Anrufe tätigen

# Probleme durch Werbenetzwerke

- oft sehr aggressiv und datenhungrig

- verwenden viele Permissions

- sammeln persönliche Daten und senden diese an Werbenetz-Betreiber

# Folgen für Entwickler

=> dadurch entsteht Verhalten welches oft identisch zu bekannter Spyware ist

=> App läuft Gefahr als Malware klassifiziert zu werden

# Zusammenfassung

Schutzmaßnahmen und Ausblick

# Infektionswege

- meist durch Third-Party Märkte und kompromittierte Webseiten

- legitime Apps mit Malware neu gepackt

- bösartige Updates legitimer Apps durch gestohlene Zertifikate

- übliche Phishing-Techniken

# Schutzmaßnahmen für den Nutzer

- nach Möglichkeit nur Apps aus dem Google Play Markt installieren

- AV-Lösung einsetzen

- vorsichtiger Umgang mit NFC und QR-Codes

# Schutzmaßnahmen für den Entwickler

- Vorsicht bei dem Einbinden von Werbenetzwerken

- Permissions sparsam verwenden

- Google-Richtlinien zu Obfuscation beachten

# Ausblick

- Obfuscation kommt so langsam auch bei Android
  - erschwert die statische Analyse

- Malware tarnt sich immer besser
  - voll funktionsfähige Apps mit bösartigem Verhalten im Hintergrund
  - Versand von Premium-SMS nur in geringer Menge

- Werbenetzwerke sammeln immer mehr persönliche Daten
  - zusätzliches Generieren von Einnahmen

# Quellen


[1] Juniper Networks Inc.
2011 Mobile Threat Report

[2] International Secure System Labs
http://anubis.iseclab.org

[3] Universität Erlangen-Nürnberg
http://www.mobile-sandbox.com

[4] W. Enck et al.
Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones

[5] R. Stevens et al.
Investigating User Privacy in Android Ad Libraries

[6] A. Desnos et al.
Droidbox: An Android Application sAndbox for dynamic Analysis

# Quellen


[7] Droidbox
http://code.google.com/p/droidbox/

[8] Androguard
http://code.google.com/p/androguard/

[9] VirusTotal API
https://www.virustotal.com/documentation/public-api/

# Vielen Dank für Ihre Aufmerksamkeit!

**Michael Spreitzenbarth**

**Lehrstuhl für Informatik 1**
**Universität Erlangen-Nürnberg**

michael.spreitzenbarth@cs.fau.de
https://twitter.com/m_spreitz

**OWASP**
07.11.2012