Hacking Team

how they infected your Android device by 0days

Attila Marosi
Senior Threat Researcher
OSCE, OSCP, ECSA, CEH

SOPHOS

20-22 OCTOBER 2015
11 years
HACK.LU

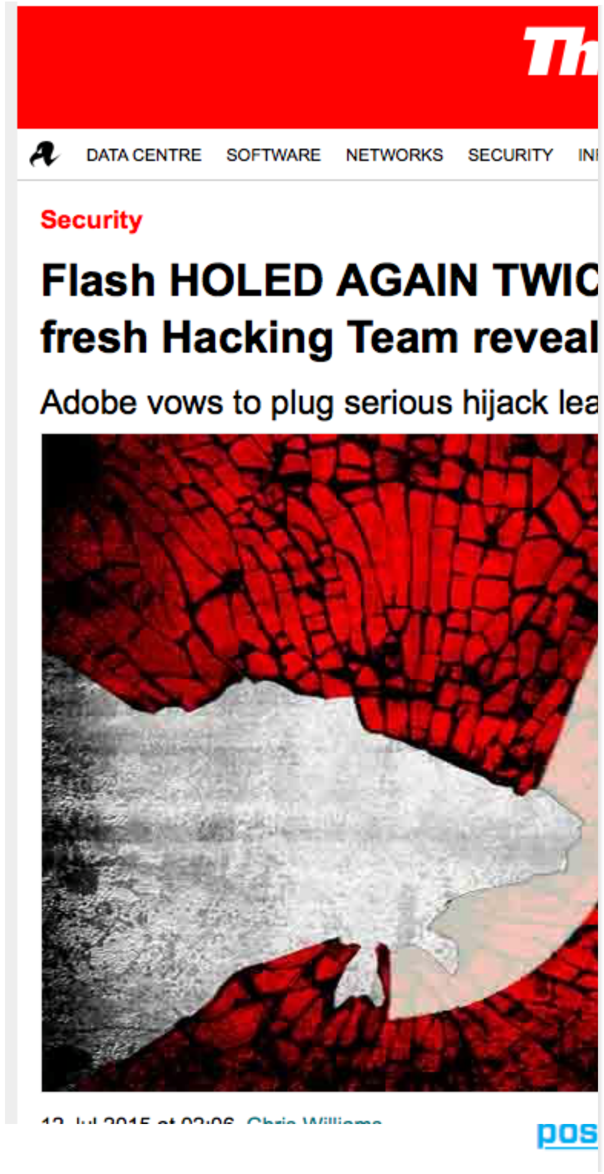
Agenda

- Hacking Team - Remote Control System (RCS)
- Leak –]Hacked Team[
- WebView exploit for Android
- DEMO
- Avoiding techniques they used
- Conclusion

Hacking Team – Remote Control System

- Product for Law-enforcement agencies (only)
  - Flagship - Remote Control System (RCS)
    - Windows, OSX, Linux,
    - Android, iOS, Blackberry, Windows Phones, Symbian
  - remote exploits (many 0days)
  - UEFI BIOS rootkit
  - remote injectors for ISP side
- civil right activist – “Enemy of the Internet”
  - Some of their customers - non-democratic countries
  - Using this tool against journalists and protesters

LEAK/HACK – STOLEN DATA



Th

DATA CENTRE SOFTWARE NETWORKS SECURITY INI

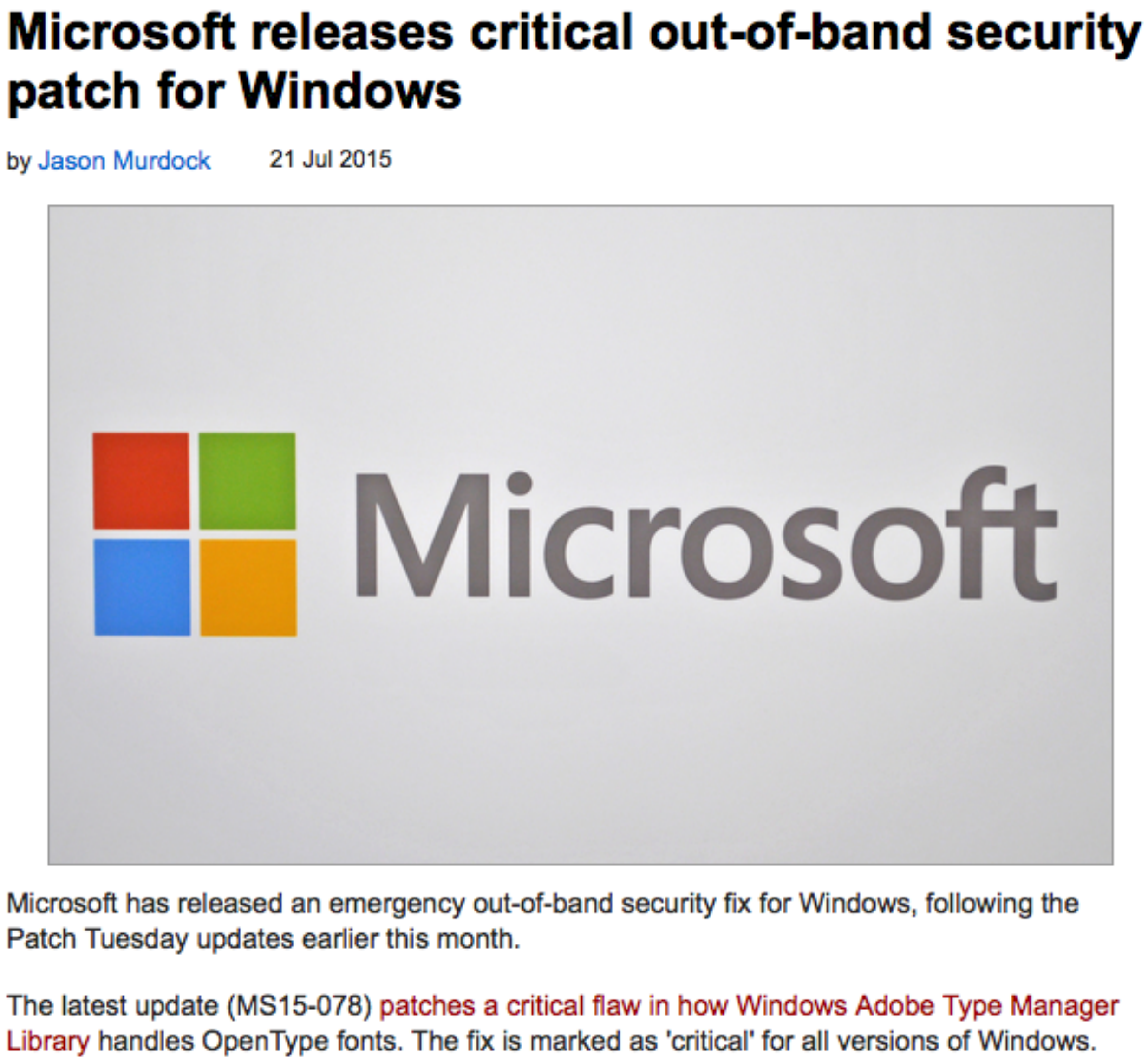
Security

Flash HOLED AGAIN TWIC fresh Hacking Team reveal

Adobe vows to plug serious hijack lea

12 Jul 2015 at 02:06, Chris Williams

pos



Microsoft releases critical out-of-band security patch for Windows

by Jason Murdock 21 Jul 2015

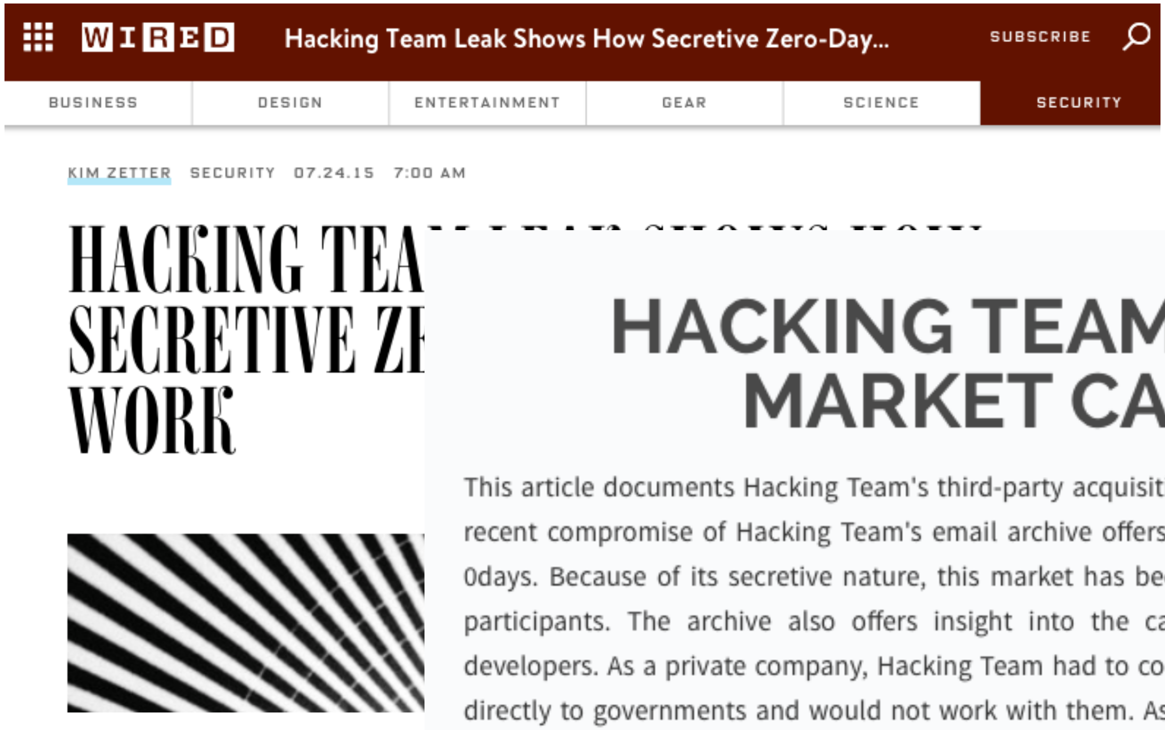
Microsoft has released an emergency out-of-band security fix for Windows, following the Patch Tuesday updates earlier this month.

The latest update (MS15-078) patches a critical flaw in how Windows Adobe Type Manager Library handles OpenType fonts. The fix is marked as 'critical' for all versions of Windows.

LEAK/HACK – STOLEN DATA

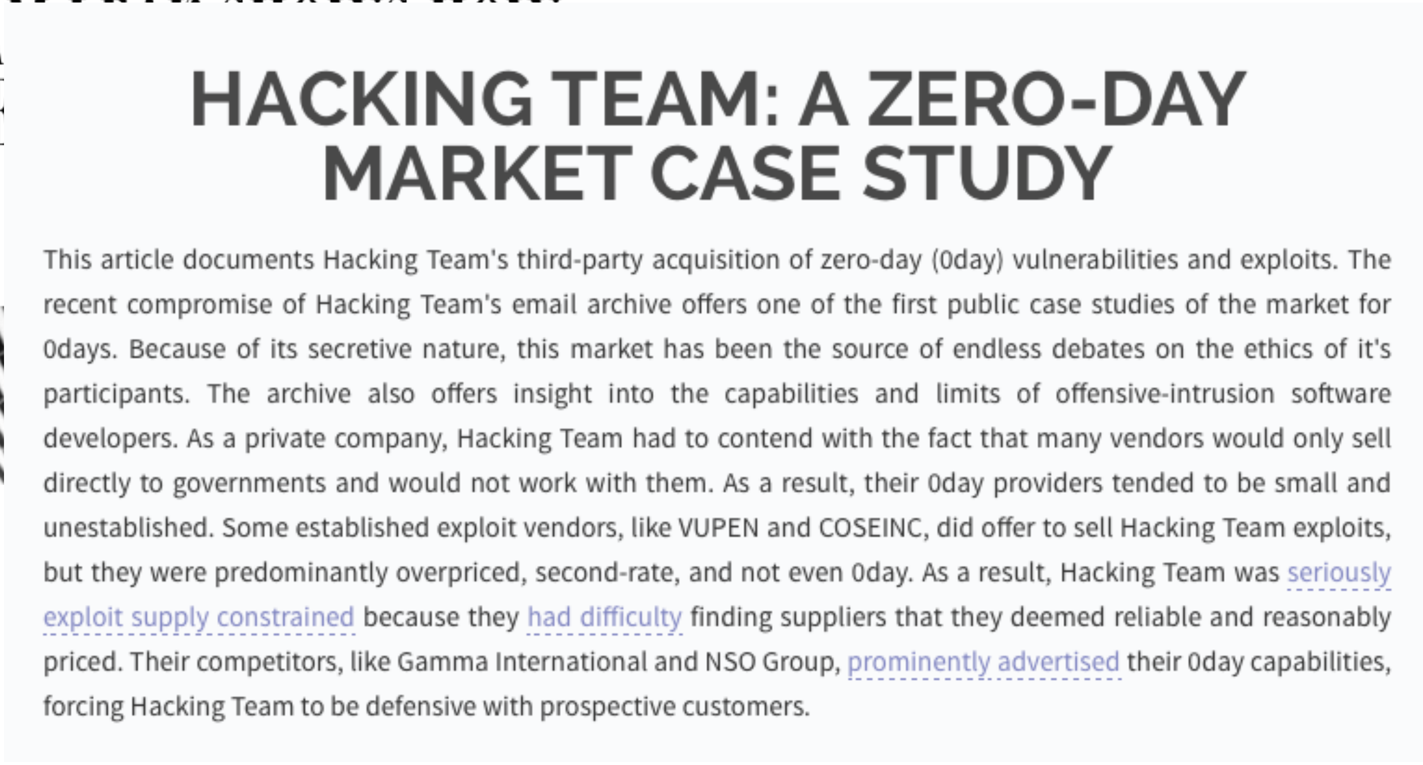
- July 5th of 2015
- still no information who did it
- what was stolen = “almost” everything (400GB)
  - all source codes
    - full git repositories (53 repos.)
  - 6 0day exploits
    - CVE-2015-{5119, 2387, 5122, 5123, 2425, 2426}
  - release brochures, product documentation
  - company emails

LEAK/HACK – Market of 0day



KIM ZETTER SECURITY 07.24.15 7:00 AM

HACKING TEA
SECRETIVE ZI
WORK



HACKING TEAM: A ZERO-DAY MARKET CASE STUDY

This article documents Hacking Team's third-party acquisition of zero-day (0day) vulnerabilities and exploits. The recent compromise of Hacking Team's email archive offers one of the first public case studies of the market for 0days. Because of its secretive nature, this market has been the source of endless debates on the ethics of it's participants. The archive also offers insight into the capabilities and limits of offensive-intrusion software developers. As a private company, Hacking Team had to contend with the fact that many vendors would only sell directly to governments and would not work with them. As a result, their 0day providers tended to be small and unestablished. Some established exploit vendors, like VUPEN and COSEINC, did offer to sell Hacking Team exploits, but they were predominantly overpriced, second-rate, and not even 0day. As a result, Hacking Team was seriously exploit supply constrained because they had difficulty finding suppliers that they deemed reliable and reasonably priced. Their competitors, like Gamma International and NSO Group, prominently advertised their 0day capabilities, forcing Hacking Team to be defensive with prospective customers.

https://tsyrklevich.net/2015/07/22/hacking-team-0day-market/

INFRASTRUCTURE

1. HT RCS – RAT agent to monitor everything in all interesting platforms
2. Infection
    - Melting tool
    - Exploit Delivery Network (Windows / Android)
    - Remote Mobile Infection (vector-rim – crafted MMS)
    - Injection Proxy Appliance (vector-ipa)
        - Inject malicious contents
        - Melt on-the-fly
    - Offline infection (with bootable devices)
3. Control
    - proxy chain by Anonymizer
    - Fancy control panel for agents

EXPLOIT DELIVERY NETWORK

SOPHOS

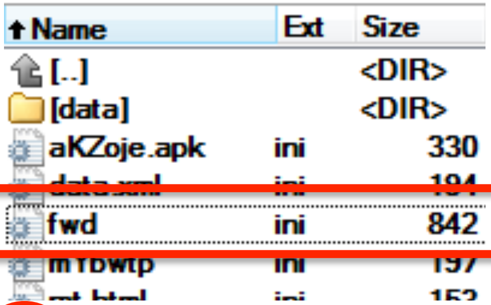
20-22 OCTOBER 2015
11 years
HACK.LU

EXPLOIT DELIVERY NETWORK

- Separated systems for Windows and Android
- They were “**one-shot**” exploits – just a very limited time they were available
- Each customer (victims ☺) had a dedicated folder – URLs and place for logs
- httpd, URL rewrite, one file – download.php
- all resources had a .ini file – how it has to be handled
  - static or dynamic values,
  - how long could it be accessible

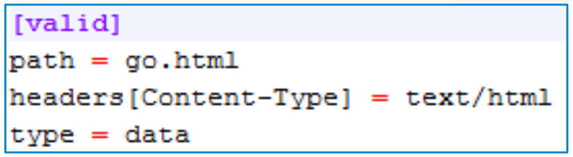
EXPLOIT DELIVERY NETWORK

1



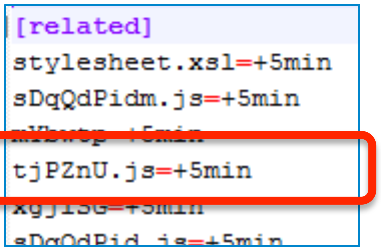
Name	Ext	Size
[..]		<DIR>
[data]		<DIR>
aKZoje.apk	ini	330
data.xml	ini	194
fwd	ini	842
mYbwtp	ini	197
mt.html	ini	153

http://...../docs/BR2u9Z/fwd



```
[valid]
path = go.html
headers[Content-Type] = text/html
type = data
```

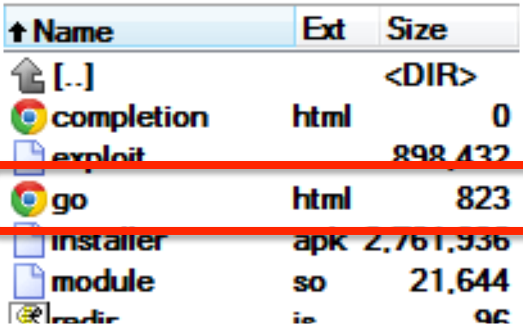
2



```
[related]
stylesheet.xsl=+5min
sDqQdPidm.js=+5min
mYbwtp=+5min
tjPZnU.js=+5min
xgjISG=+5min
sDqQdPid.js=+5min
```

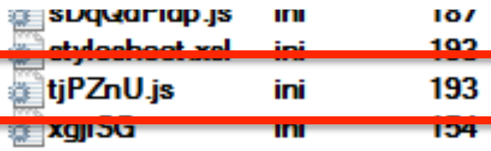
Expiry value will be updated

3



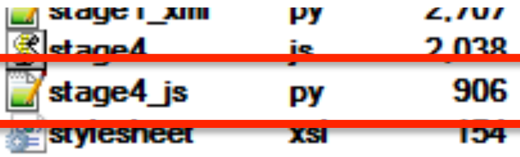
Name	Ext	Size
[..]		<DIR>
completion	html	0
exploit		898,432
go	html	823
installer	apk	2,761,936
module	so	21,644
redir	js	96

4

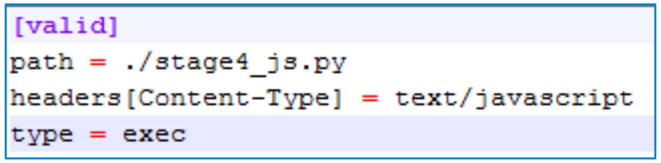


Name	Ext	Size
sDqQdPidp.js	ini	187
stylesheet.xsl	ini	193
tjPZnU.js	ini	193
xgjISG	ini	154

http://...../docs/BR2u9Z/tjPZnU



Name	Ext	Size
stage1_xml	py	2,707
stage4	js	2,038
stage4_js	py	906
stylesheet	xsl	154



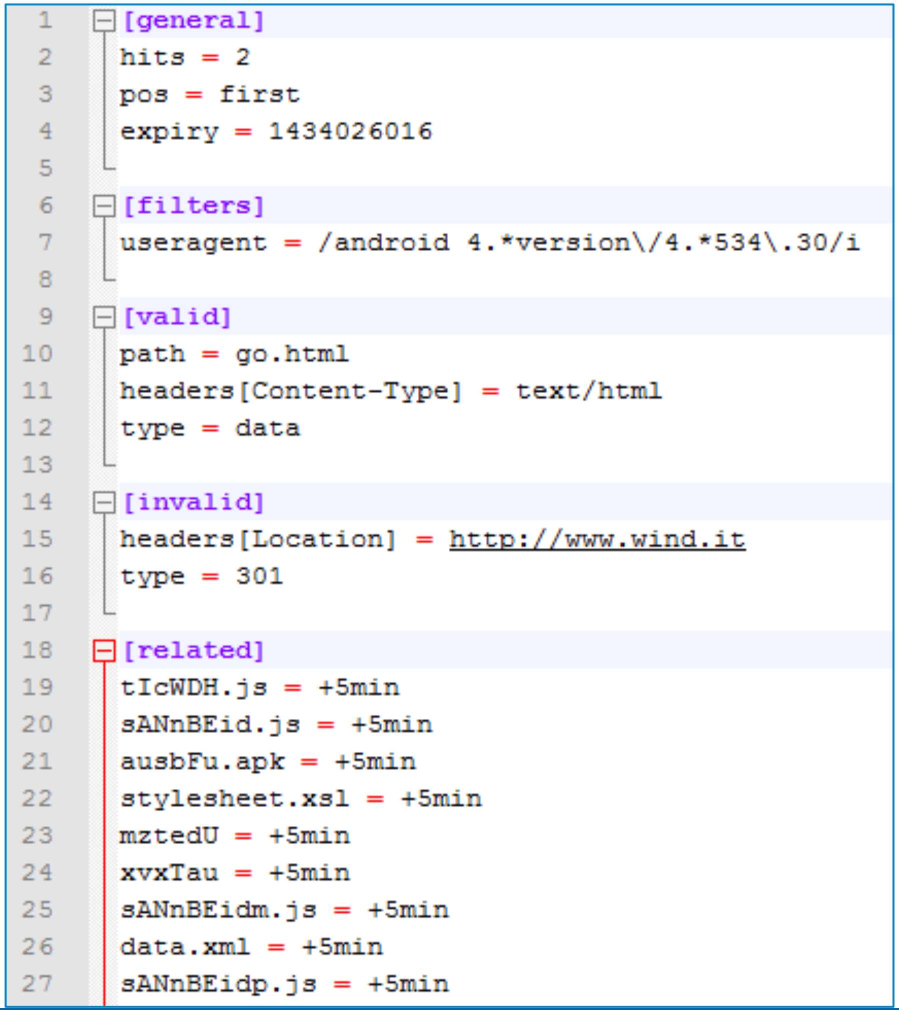
```
[valid]
path = ./stage4_js.py
headers[Content-Type] = text/javascript
type = exec
```

EXPLOIT DELIVERY NETWORK

http://exploit_delivery_network/docs/**BR2u9z**/**fwd**

Customer = victim ID (BR2u9z)

fwd = fwd.ini → go.html



```
[general]
hits = 2
pos = first
expiry = 1434026016

[filters]
useragent = /android 4.*version\/4.*534\.30/i

[valid]
path = go.html
headers[Content-Type] = text/html
type = data

[invalid]
headers[Location] = http://www.wind.it
type = 301

[related]
tIcWDH.js = +5min
sANnBEid.js = +5min
ausbFu.apk = +5min
stylesheet.xsl = +5min
mztedU = +5min
xvxTau = +5min
sANnBEidm.js = +5min
data.xml = +5min
sANnBEidp.js = +5min
```

hits = hits left

expiry = reachable until this time

useragent = User-Agent header must contains this string

The content of the go.html file was sent back as a result of the call

HTTP Redirection to a harmless website

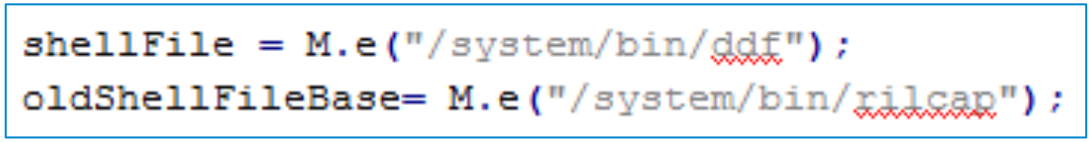
[**related**] If the call was right all related files' configs were updated – within 5 mins they were accessible

RCS FOR ANDROIDS

RCS agent for Android

- Install custom root service (ddf / rilcap) - instead of using ‘su’
- Modify permissions of APK
- Install itself as an administrator application
- hooking into the MediaServer system service to intercept all audio content – all calls (regardless of the app) can be motorized by this technique
- Traditional evidence gathering features
    - Take screenshot, monitoring clipboard, location tracking
    - Contact and messages for these apps
    - FaceBook, Viber, Skype, wechat, whatsapp, snapchat, gtalk, bbm, build in mail app & contacts

RCS Android root tool (ddf / rilcap)



```
shellFile = M.e("/system/bin/ddf");
oldShellFileBase= M.e("/system/bin/rilcap");
```

```
Usage:
  fb     try to capture a screen snapshot
  vol    kill VOLD twice
  reb    reboot the phone
  blr    mount /system in READ_ONLY
  blw    mount /system in READ_WRITE
  rt     install the root shell in /system/bin/rilcap
  ru     remove the root shell from /system/bin/rilcap
  rf     <mntpoint> <file> - remove <file> from <mntpoint>
  sd     mount /sdcard
  air    check if the shell has root privileges
  qzx    "command" - execute the given commandline
  fhc    <src> <dest> - copy <src> to <dst>
  fhs    <mntpoint> <src> <dest> - copy
         <src> to <dst> on mountpoint <mntpoint>
  fho    <user> <group> <file> - chown <file> to <user>:<group>
  pzm    <newmode> <file> - chmod <file> to <newmode>
  adm    <package name/receiver>
  qzs    start a root shell
  lid    <proc> <dest file> - return process id for <proc>
         write it to <dest file>
  ape    <content> <dest file> - append text
         <content> to <dest files> if not yet present
  srh    <content> <file> - search for <content> in <file>
```

SOPHOS

Exploit for Android

Exploit for Android

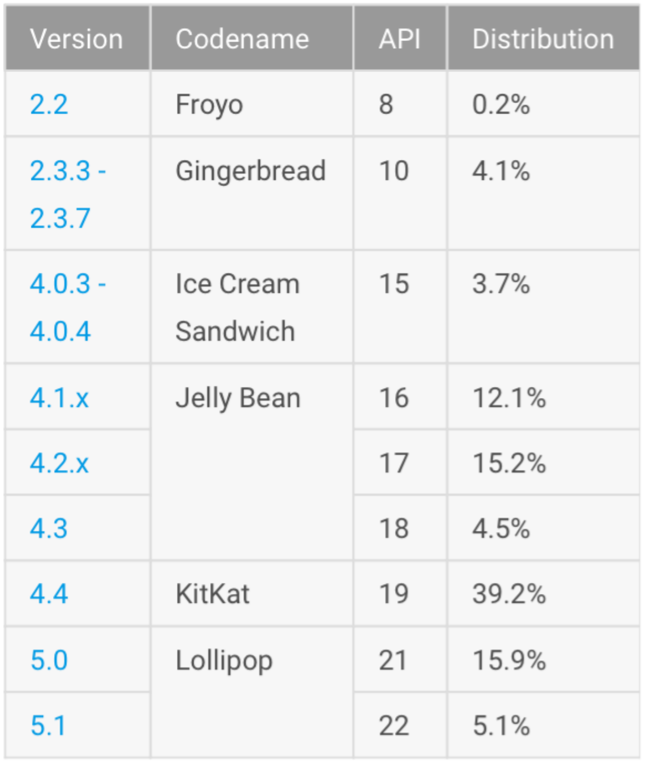
- Remote code execution (webview)
  - They joined 3 vulnerabilities to create this exploit
  - For code execution 4 stages
  - The most stages are encrypted or obfuscated
  - Information leakage vulnerability helped them to bypass ASLR
  - They used ROP gadgets to bypass NX
- Local root exploit
  - exynos exploit (Samsung)
  - CVE-2013-6282 - get_user and (2) put_user
  - CVE-2014-3153 - futex_requeue (TowelRoot)

Vulnerabilities they joined together (webview)

- **Information Leak (CVE-2011-1202)**
  - "The xsltGenerateIdFunction function in functions.c in libxslt 1.1.26 and earlier, as used in Google Chrome before **10.0.648.127** and other products, allows remote attackers to obtain potentially sensitive information about heap memory addresses via an XML document containing a call to the XSLT generate-id XPath function."
  - "combined information leakage vulnerability CVE-2011-1202, to obtain the base address and then get libwebcore.so libc.so base address."
- **Arbitrary Memory Read (CVE-2012-2825)**
  - "The XSL implementation in Google Chrome before **20.0.1132.43** allows remote attackers to cause a denial of service (incorrect read operation) via unspecified vectors."
- **Heap-Buffer-overflow (CVE-2012-2871)**
  - "libxml2 2.9.0-rc1 and earlier, as used in Google Chrome before **21.0.1180.89**, does not properly support a cast of an unspecified variable during handling of XSL transforms, which allows remote attackers to cause a denial of service or possibly have unknown other impact via a crafted document, related to the _xmlNs data structure in include/libxml/tree.h."

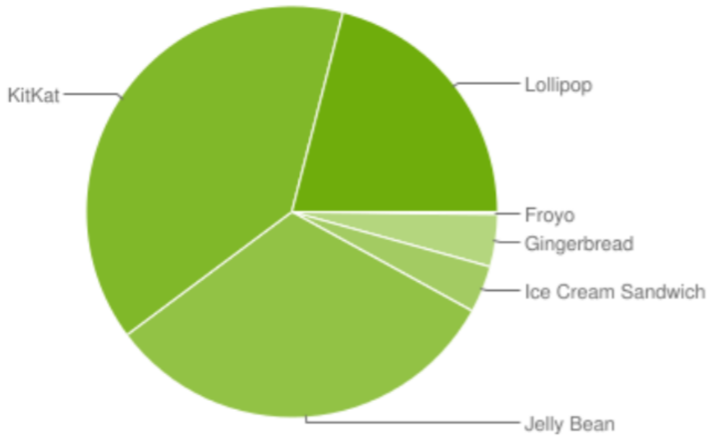
- https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-2825
- http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-2871
- http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-1202

HT WebView exploit



Version	Codename	API	Distribution
2.2	Froyo	8	0.2%
2.3.3 - 2.3.7	Gingerbread	10	4.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	3.7%
4.1.x	Jelly Bean	16	12.1%
4.2.x		17	15.2%
4.3		18	4.5%
4.4	KitKat	19	39.2%
5.0	Lollipop	21	15.9%
5.1		22	5.1%

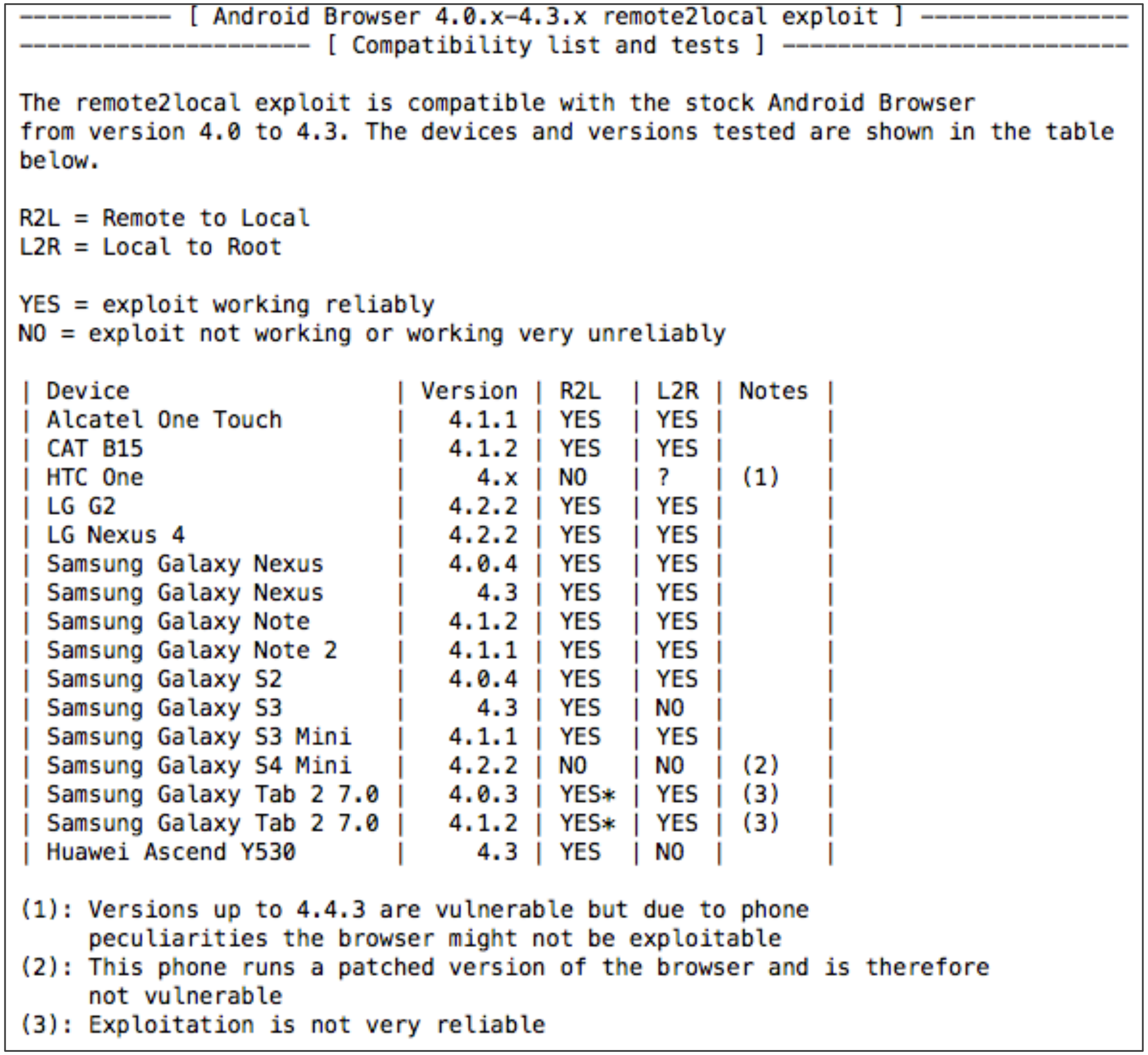
Data collected during a 7-day period ending on September 7, 2015.



Android version	WebKit version
Android 4.0.1	534.30
Android 4.0.2	534.30
Android 4.0.3	534.30
Android 4.0.4	534.30
Android 4.1.1	534.30
Android 4.1.2	534.30
Android 4.2	534.30
Android 4.2.1	534.30
Android 4.2.2	534.30
Android 4.3	534.30

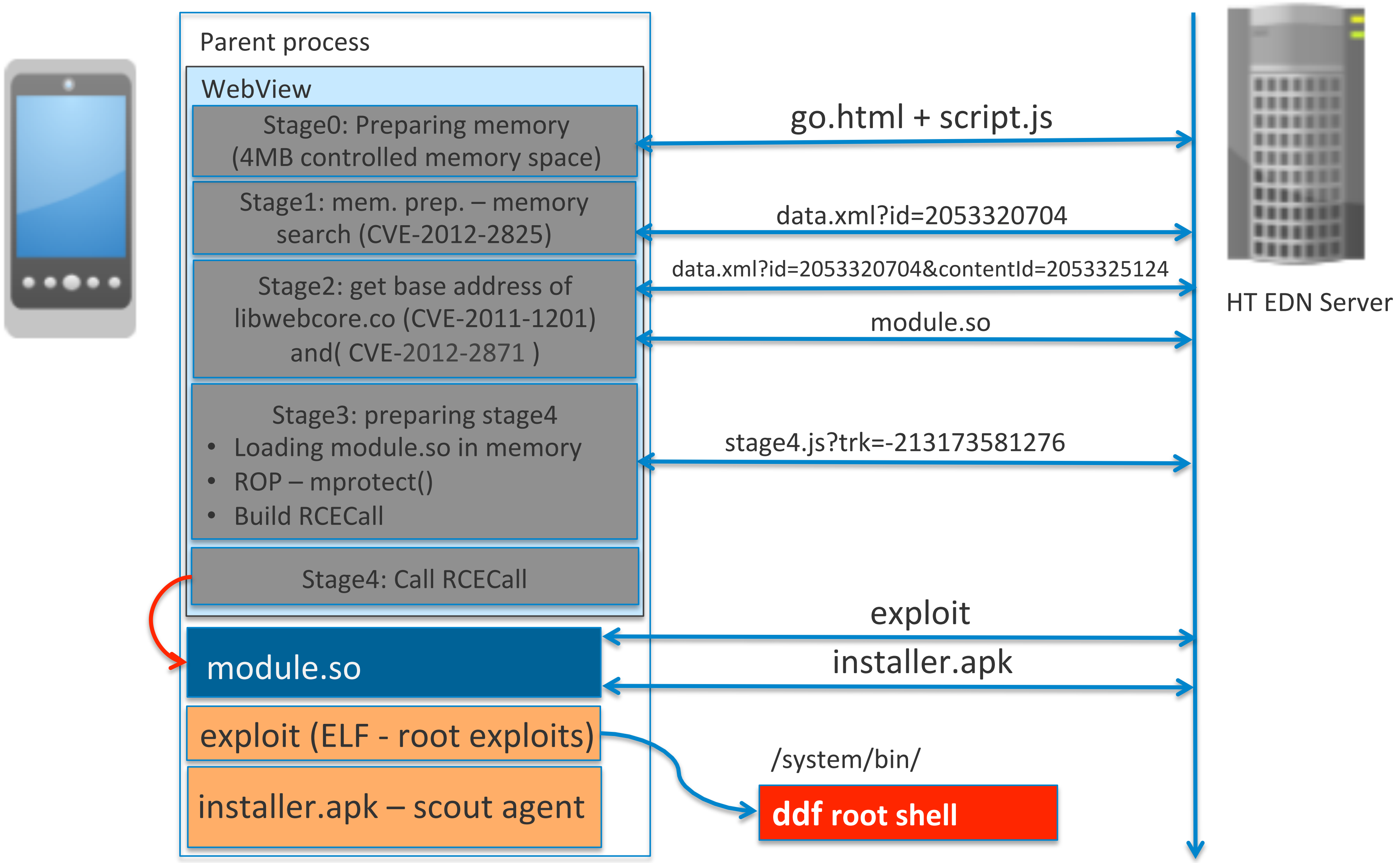
35.5% are still vulnerable

HT WebView exploit



```
----------- [ Android Browser 4.0.x-4.3.x remote2local exploit ] ---------------
-------------------- [ Compatibility list and tests ] -------------------------

The remote2local exploit is compatible with the stock Android Browser
from version 4.0 to 4.3. The devices and versions tested are shown in the table
below.

R2L = Remote to Local
L2R = Local to Root

YES = exploit working reliably
NO = exploit not working or working very unreliably

| Device                     | Version | R2L  | L2R | Notes |
| Alcatel One Touch          |   4.1.1 | YES  | YES |       |
| CAT B15                    |   4.1.2 | YES  | YES |       |
| HTC One                    |     4.x | NO   | ?   | (1)   |
| LG G2                      |   4.2.2 | YES  | YES |       |
| LG Nexus 4                 |   4.2.2 | YES  | YES |       |
| Samsung Galaxy Nexus       |   4.0.4 | YES  | YES |       |
| Samsung Galaxy Nexus       |     4.3 | YES  | YES |       |
| Samsung Galaxy Note        |   4.1.2 | YES  | YES |       |
| Samsung Galaxy Note 2      |   4.1.1 | YES  | YES |       |
| Samsung Galaxy S2          |   4.0.4 | YES  | YES |       |
| Samsung Galaxy S3          |     4.3 | YES  | NO  |       |
| Samsung Galaxy S3 Mini     |   4.1.1 | YES  | YES |       |
| Samsung Galaxy S4 Mini     |   4.2.2 | NO   | NO  | (2)   |
| Samsung Galaxy Tab 2 7.0   |   4.0.3 | YES* | YES | (3)   |
| Samsung Galaxy Tab 2 7.0   |   4.1.2 | YES* | YES | (3)   |
| Huawei Ascend Y530         |     4.3 | YES  | NO  |       |

(1): Versions up to 4.4.3 are vulnerable but due to phone
     peculiarities the browser might not be exploitable
(2): This phone runs a patched version of the browser and is therefore
     not vulnerable
(3): Exploitation is not very reliable
```

WebView exploit



Parent process
WebView
Stage0: Preparing memory (4MB controlled memory space)
Stage1: mem. prep. – memory search (CVE-2012-2825)
Stage2: get base address of libwebcore.co (CVE-2011-1201) and(CVE-2012-2871)
Stage3: preparing stage4
• Loading module.so in memory
• ROP – mprotect()
• Build RCECall
Stage4: Call RCECall
module.so
exploit (ELF - root exploits)
installer.apk – scout agent
go.html + script.js
data.xml?id=2053320704
data.xml?id=2053320704&contentId=2053325124
module.so
stage4.js?trk=-213173581276
exploit
installer.apk
/system/bin/
ddf root shell
HT EDN Server

DEMO

SOPHOS

Plan A:

- **Email with malicious link**
- **Click on it to trigger the exploit**

20-22 OCTOBER 2015 11 years HACK.LU

BORING

DEMO

SOPHOS

Plan B:

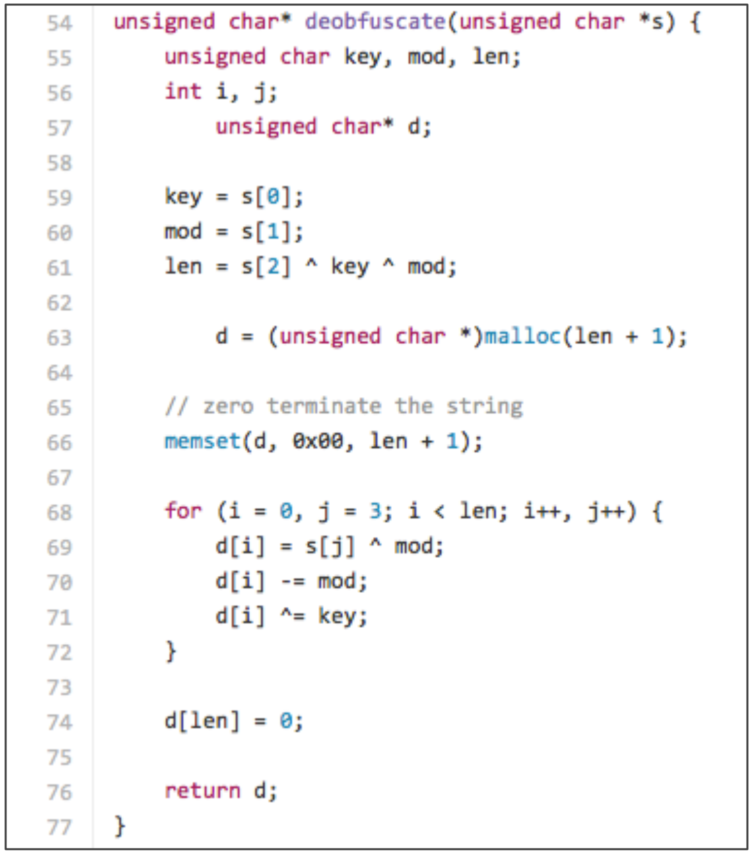
- hijack network flow (free wifi, ISP)
- inject malicious content on-the-fly
- exploit any app which uses webview

20-22 OCTOBER 2015
11 years
HACK.LU

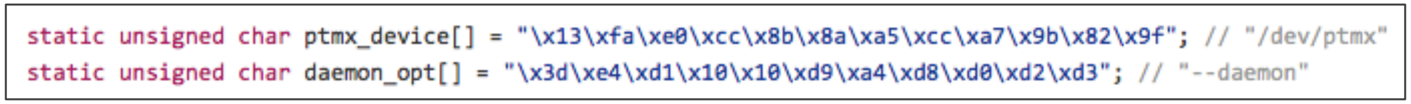
HOW THEY FLEW UNDER THE RADAR

Code protection

- Source code obfuscation & code protection
  - Obfuscation on all levels
  - Own packer
  - Melt with legal app (all platforms)
  - VMProtect for Windows
  - ProGard for Android
  - ELF string obfuscator
- Other solutions
  - Different version of the tool (Scout, Soldier, Elite)
  - blacklisted applications



```c
unsigned char* deobfuscate(unsigned char *s) {
    unsigned char key, mod, len;
    int i, j;
        unsigned char* d;

    key = s[0];
    mod = s[1];
    len = s[2] ^ key ^ mod;

        d = (unsigned char *)malloc(len + 1);

    // zero terminate the string
    memset(d, 0x00, len + 1);

    for (i = 0, j = 3; i < len; i++, j++) {
        d[i] = s[j] ^ mod;
        d[i] -= mod;
        d[i] ^= key;
    }

    d[len] = 0;

    return d;
}
```

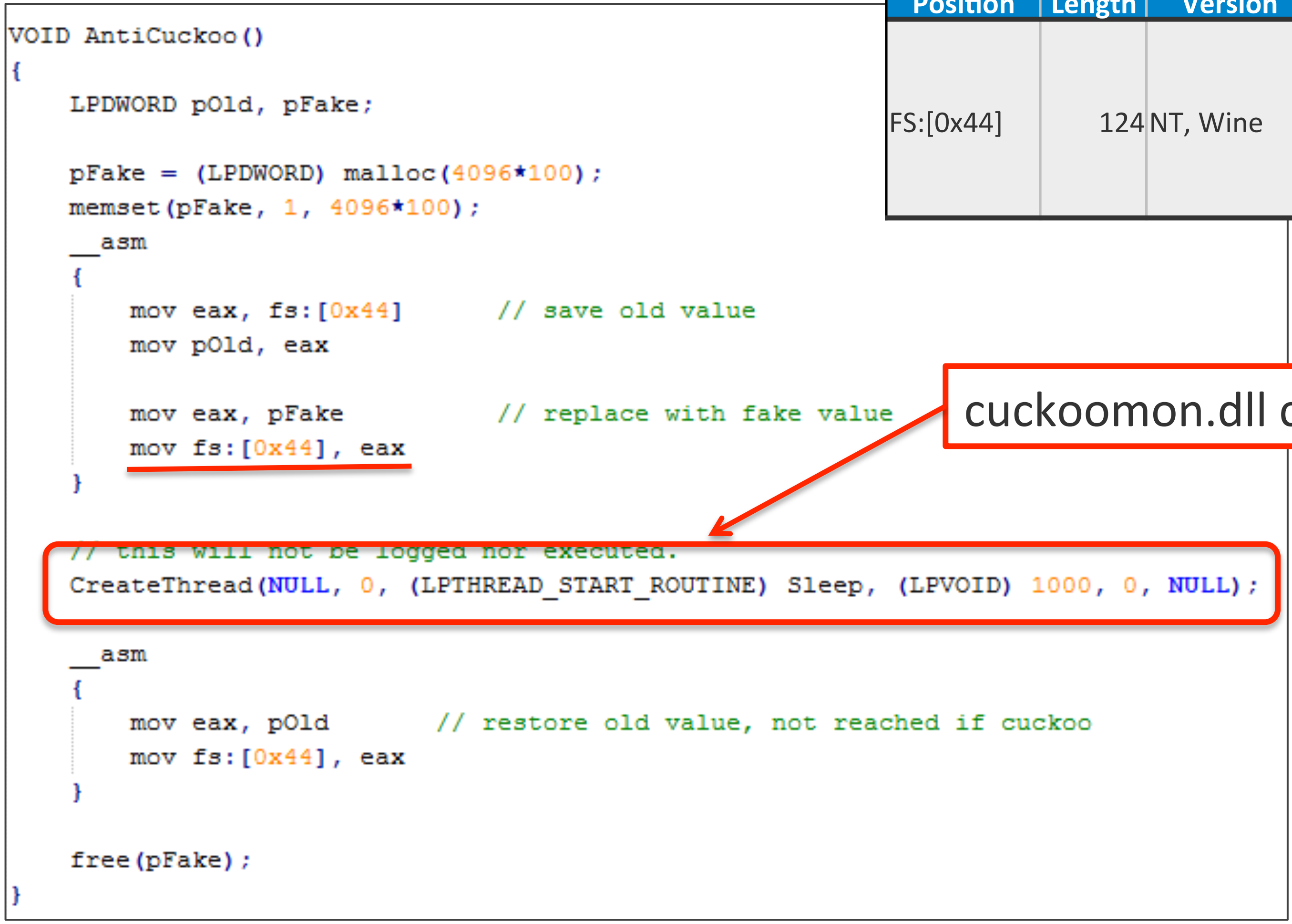


```c
static unsigned char ptmx_device[] = "\x13\xfa\xe0\xcc\x8b\x8a\xa5\xcc\xa7\x9b\x82\x9f"; // "/dev/ptmx"
static unsigned char daemon_opt[] = "\x3d\xe4\xd1\x10\x10\xd9\xa4\xd8\xd0\xd2\xd3"; // "--daemon"
```

Avoiding Emulation (Windows)

- virtualization / sandbox detection
  `(scout-win-master/core-scout-win32/antivm.cpp)`
  - AntiVMWare() - VMWare
    - WMI query “`SELECT SerialNumber FROM Win32_Bios`”
  - AntiVBox() - VirtualBox
    - WMI query “`SELECT DeviceId FROM Win32_PnPEntity`”
    - Seeking for this value:
      “`PCI\\VEN_80EE&DEV_CAFE`”

Cuckoo avoiding (Windows)

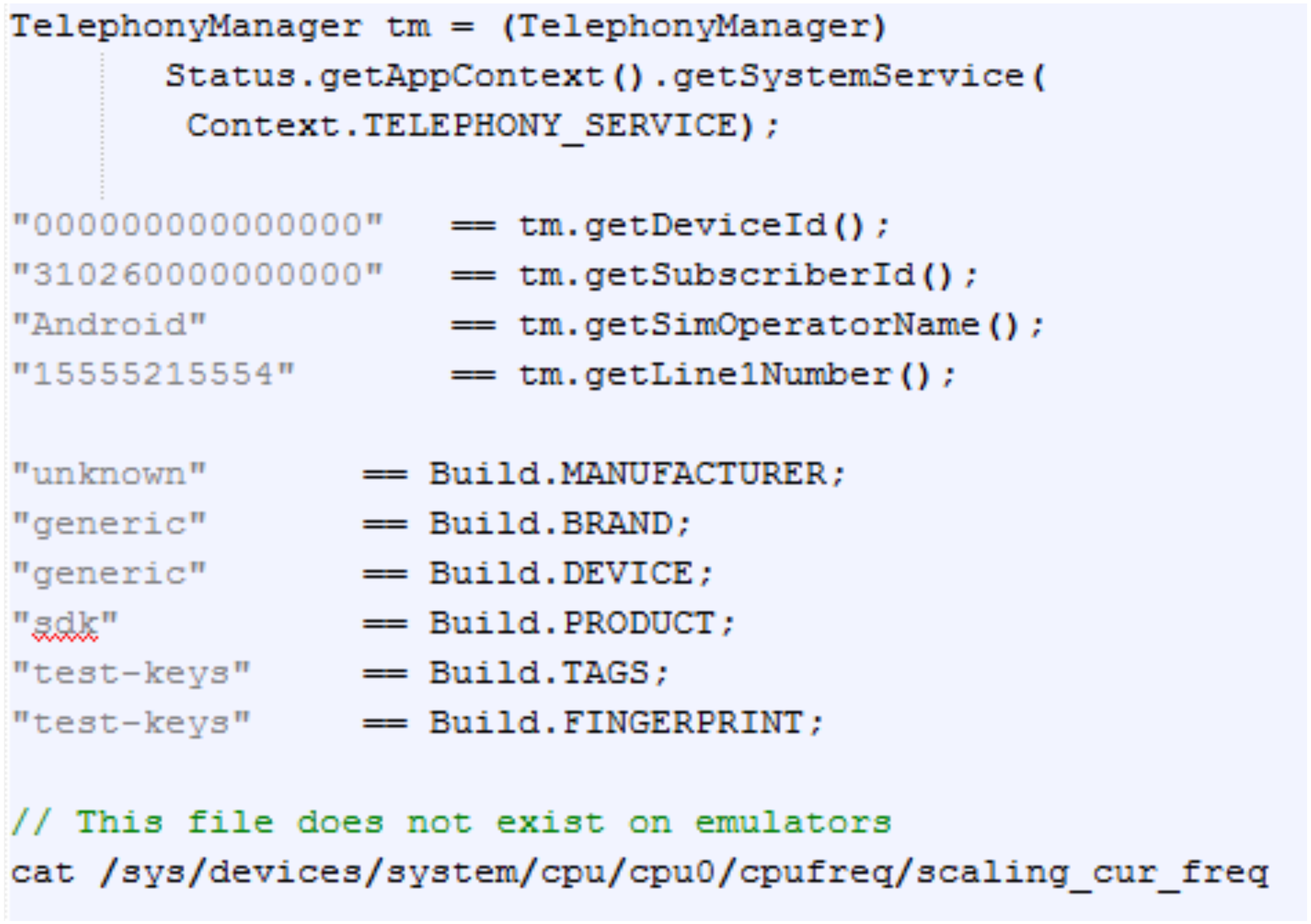


```
VOID AntiCuckoo()
{
    LPDWORD pOld, pFake;

    pFake = (LPDWORD) malloc(4096*100);
    memset(pFake, 1, 4096*100);
    __asm
    {
        mov eax, fs:[0x44]      // save old value
        mov pOld, eax

        mov eax, pFake          // replace with fake value
        mov fs:[0x44], eax
    }

    // this will not be logged nor executed.
    CreateThread(NULL, 0, (LPTHREAD_START_ROUTINE) Sleep, (LPVOID) 1000, 0, NULL);

    __asm
    {
        mov eax, pOld       // restore old value, not reached if cuckoo
        mov fs:[0x44], eax
    }

    free(pFake);
}
```

cuckoomon.dll crash here

Position	Length	Version	Description
FS:[0x44]	124	NT, Wine	**Win32 client information (NT)**, user32 private data (Wine), 0x60 = LastError (Win95), 0x74 = LastError (WinME)

Avoiding Emulation (Android)

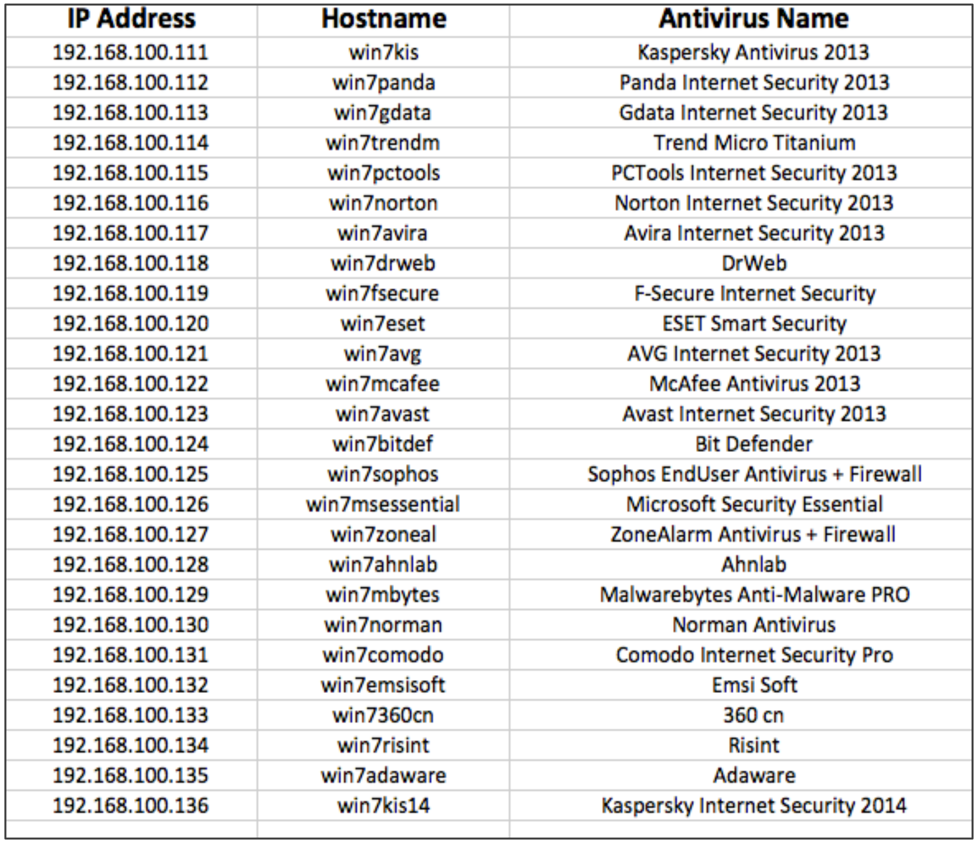


```
TelephonyManager tm = (TelephonyManager)
        Status.getAppContext().getSystemService(
         Context.TELEPHONY_SERVICE);

"000000000000000"   == tm.getDeviceId();
"310260000000000"   == tm.getSubscriberId();
"Android"           == tm.getSimOperatorName();
"15555215554"       == tm.getLine1Number();

"unknown"       == Build.MANUFACTURER;
"generic"       == Build.BRAND;
"generic"       == Build.DEVICE;
"sdk"           == Build.PRODUCT;
"test-keys"     == Build.TAGS;
"test-keys"     == Build.FINGERPRINT;

// This file does not exist on emulators
cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_cur_freq
```

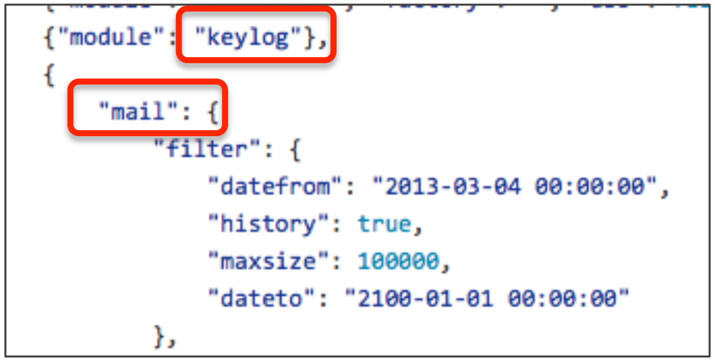
AntiVirus testing environment

- They had a dedicated VM cluster for testing all their product against many AV products to be sure they are still undetectable
- Emulating critical events
  - Screenshots
  - Evidence gathering (email, messages, files)
  - Communications, hooks
- It was a QA process before release

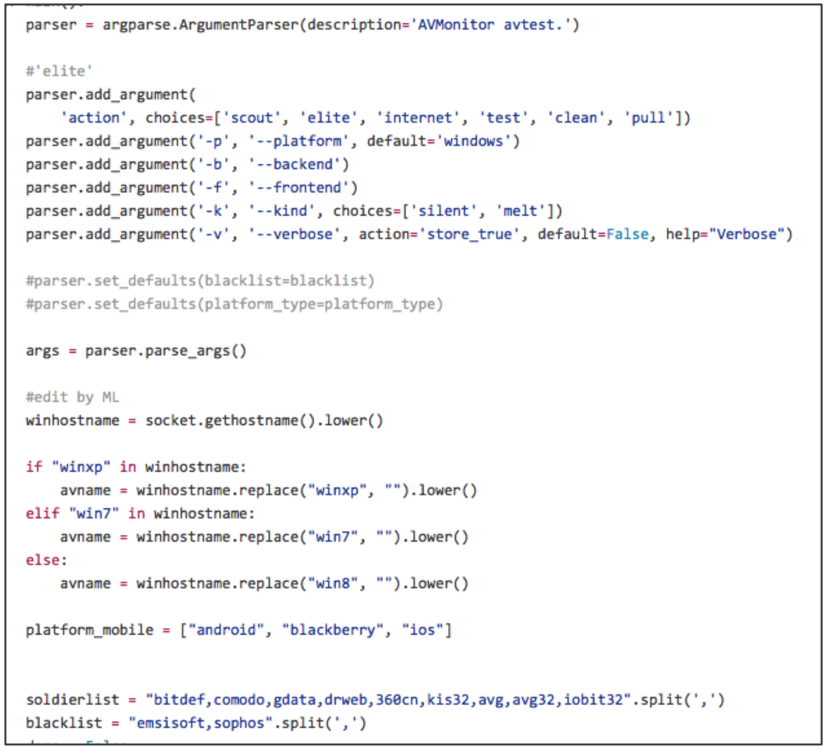
AntiVirus testing environment



IP Address	Hostname	Antivirus Name
192.168.100.111	win7kis	Kaspersky Antivirus 2013
192.168.100.112	win7panda	Panda Internet Security 2013
192.168.100.113	win7gdata	Gdata Internet Security 2013
192.168.100.114	win7trendm	Trend Micro Titanium
192.168.100.115	win7pctools	PCTools Internet Security 2013
192.168.100.116	win7norton	Norton Internet Security 2013
192.168.100.117	win7avira	Avira Internet Security 2013
192.168.100.118	win7drweb	DrWeb
192.168.100.119	win7fsecure	F-Secure Internet Security
192.168.100.120	win7eset	ESET Smart Security
192.168.100.121	win7avg	AVG Internet Security 2013
192.168.100.122	win7mcafee	McAfee Antivirus 2013
192.168.100.123	win7avast	Avast Internet Security 2013
192.168.100.124	win7bitdef	Bit Defender
192.168.100.125	win7sophos	Sophos EndUser Antivirus + Firewall
192.168.100.126	win7msessential	Microsoft Security Essential
192.168.100.127	win7zoneal	ZoneAlarm Antivirus + Firewall
192.168.100.128	win7ahnlab	Ahnlab
192.168.100.129	win7mbytes	Malwarebytes Anti-Malware PRO
192.168.100.130	win7norman	Norman Antivirus
192.168.100.131	win7comodo	Comodo Internet Security Pro
192.168.100.132	win7emsisoft	Emsi Soft
192.168.100.133	win7360cn	360 cn
192.168.100.134	win7risint	Risint
192.168.100.135	win7adaware	Adaware
192.168.100.136	win7kis14	Kaspersky Internet Security 2014



```
{"module": "keylog"},
{
  "mail": {
      "filter": {
          "datefrom": "2013-03-04 00:00:00",
          "history": true,
          "maxsize": 100000,
          "dateto": "2100-01-01 00:00:00"
      },
```



```
parser = argparse.ArgumentParser(description='AVMonitor avtest.')

#'elite'
parser.add_argument(
    'action', choices=['scout', 'elite', 'internet', 'test', 'clean', 'pull'])
parser.add_argument('-p', '--platform', default='windows')
parser.add_argument('-b', '--backend')
parser.add_argument('-f', '--frontend')
parser.add_argument('-k', '--kind', choices=['silent', 'melt'])
parser.add_argument('-v', '--verbose', action='store_true', default=False, help="Verbose")

#parser.set_defaults(blacklist=blacklist)
#parser.set_defaults(platform_type=platform_type)

args = parser.parse_args()

#edit by ML
winhostname = socket.gethostname().lower()

if "winxp" in winhostname:
    avname = winhostname.replace("winxp", "").lower()
elif "win7" in winhostname:
    avname = winhostname.replace("win7", "").lower()
else:
    avname = winhostname.replace("win8", "").lower()

platform_mobile = ["android", "blackberry", "ios"]


soldierlist = "bitdef,comodo,gdata,drweb,360cn,kis32,avg,avg32,iobit32".split(',')
blacklist = "emsisoft,sophos".split(',')
```

CONCLUSION

SOPHOS

Conclusion

- About HT and their stuff
  - it was well designed
    (leaked but never reverse engineered fully)
- The Android exploit (webview)
  - This is a quite good exploit and now it is freely available for anyone – for criminals as well
  - There are still millions of vulnerable devices
    (4.0 Ice Cream Sandwich - 4.3 Jelly Bean)
  - There are many devices in use which can not be updated
    - no official way to patch this vulnerability

References

- http://www.wired.com/2015/07/hacking-team-leak-shows-secretive-zero-day-exploit-sales-work/
- http://blog.trendmicro.com/trendlabs-security-intelligence/hacking-team-uses-uefi-bios-rootkit-to-keep-rcs-9-agent-in-target-systems/
- http://blog.trendmicro.com/trendlabs-security-intelligence/hacking-team-rcsandroid-spying-tool-listens-to-calls-roots-devices-to-get-in/
- http://blog.azimuthsecurity.com/2013/02/re-visiting-exynos-memory-mapping-bug.html
- http://blog.nativeflow.com/the-futex-vulnerability
- https://translate.google.com/translate?sl=auto&tl=en&js=y&prev=_t&hl=hu&ie=UTF-8&u=http%3A%2F%2Fsecurity.tencent.com%2Findex.php%2Fblog%2Fmsg%2F87&edit-text=
- https://www.4armed.com/blog/hacking-team-rcs-analysis-hacked-team/
- http://www.slideshare.net/jiahongfang5/mosec2015-jfang

Questions?

SOPHOS

attila.marosi@sophos.com

attila.marosi@gmail.com

PGP ID: 3782A65A

PGP FP.: 4D49 1447 A4E1 F016 F833
8700 8853 60A7 3782 A65A

20-22 OCTOBER 2015
11 years
HACK.LU