Ministry of Science, Technology & Innovation

Reversing Android Malware

The Honeynet Project 10th Annual Workshop
ESIEA || PARIS || .FR || 2011-03-21

MAHMUD AB RAHMAN

(MyCERT, CyberSecurity Malaysia)

Securing Our Cyberspace

Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

MYSELF

- Mahmud Ab Rahman
- MyCERT, CyberSecurity Malaysia
- Lebahnet(honeynet), Botnet, Malware

Copyright © 2011 CyberSecurity Malaysia

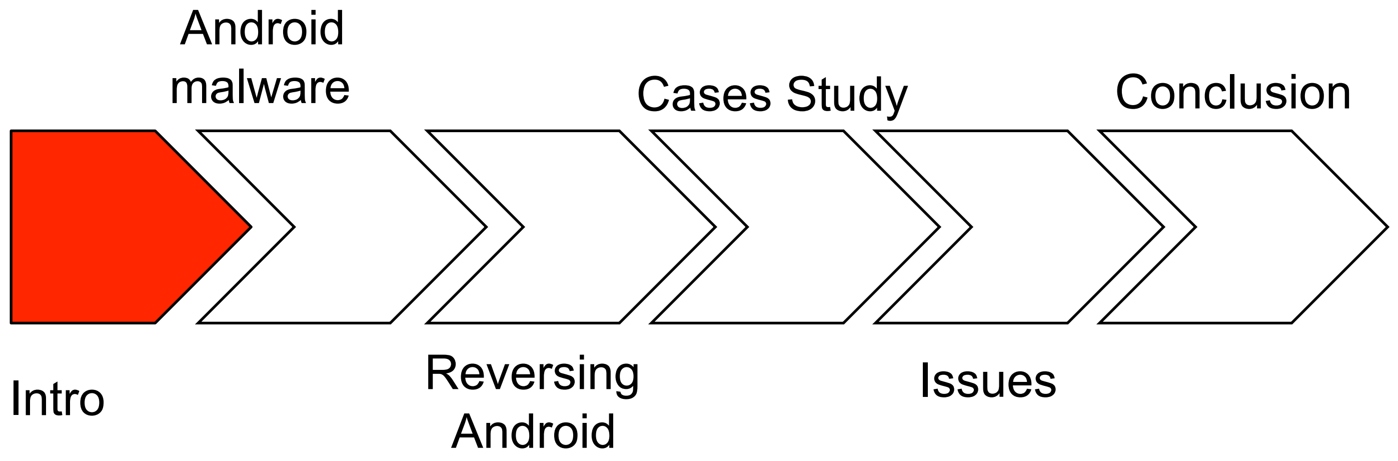
Ministry of Science, Technology & Innovation

Agenda

- Intro
- Malware and Android
- Reversing Android Malware
- Android Malware Cases study:
  - SMS.Trojan
  - Geinimi
  - DreamDroid
- Challenge and Issues
- Outro/Conclusion

Copyright © 2011 CyberSecurity Malaysia

INTRO



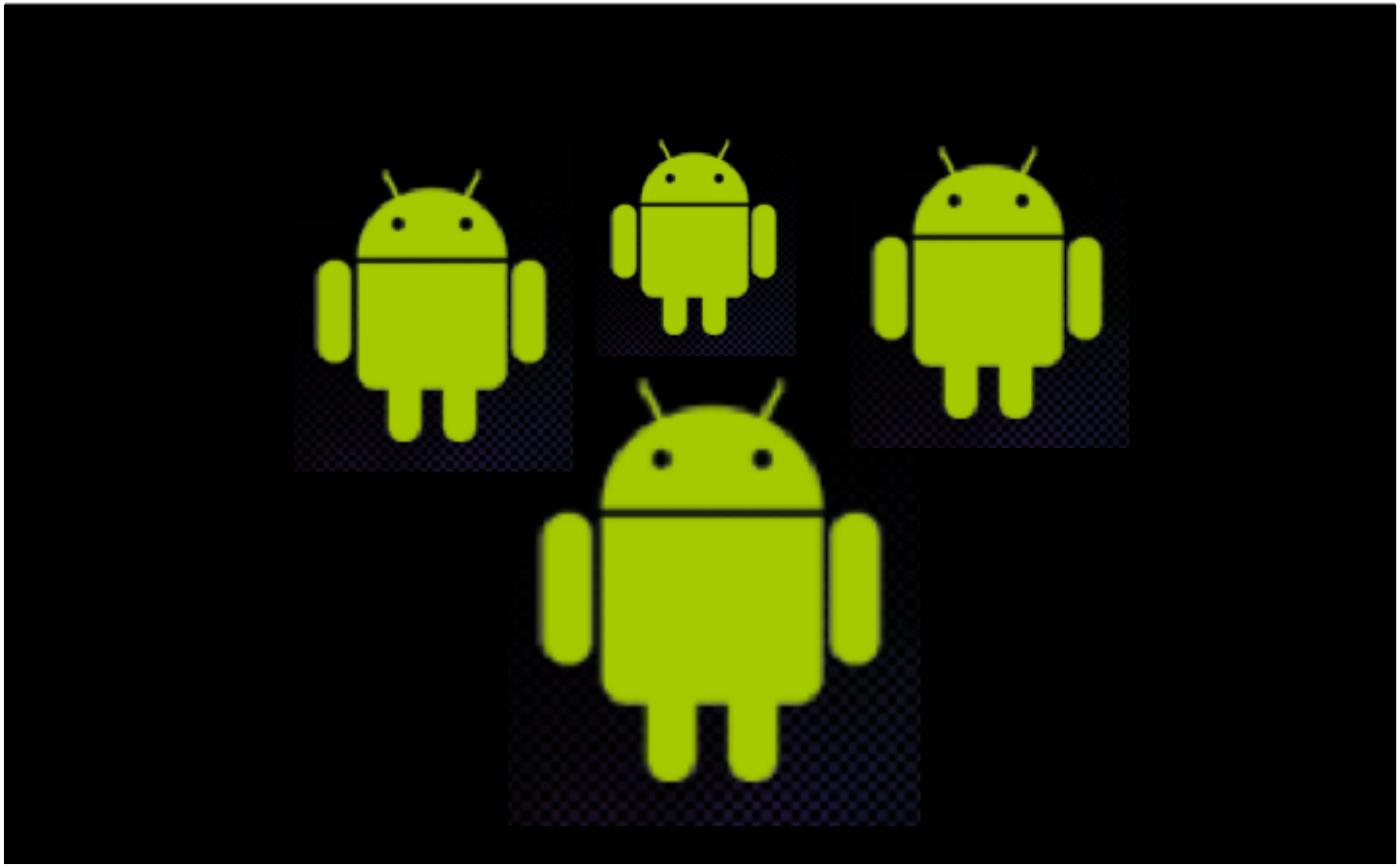
Securing Our Cyberspace

Copyright © 2011 CyberSecurity Malaysia

CyberSecurity MALAYSIA

MOSTI Ministry of Science, Technology & Innovation

INTRO : Android 101



INTRO : Android 101

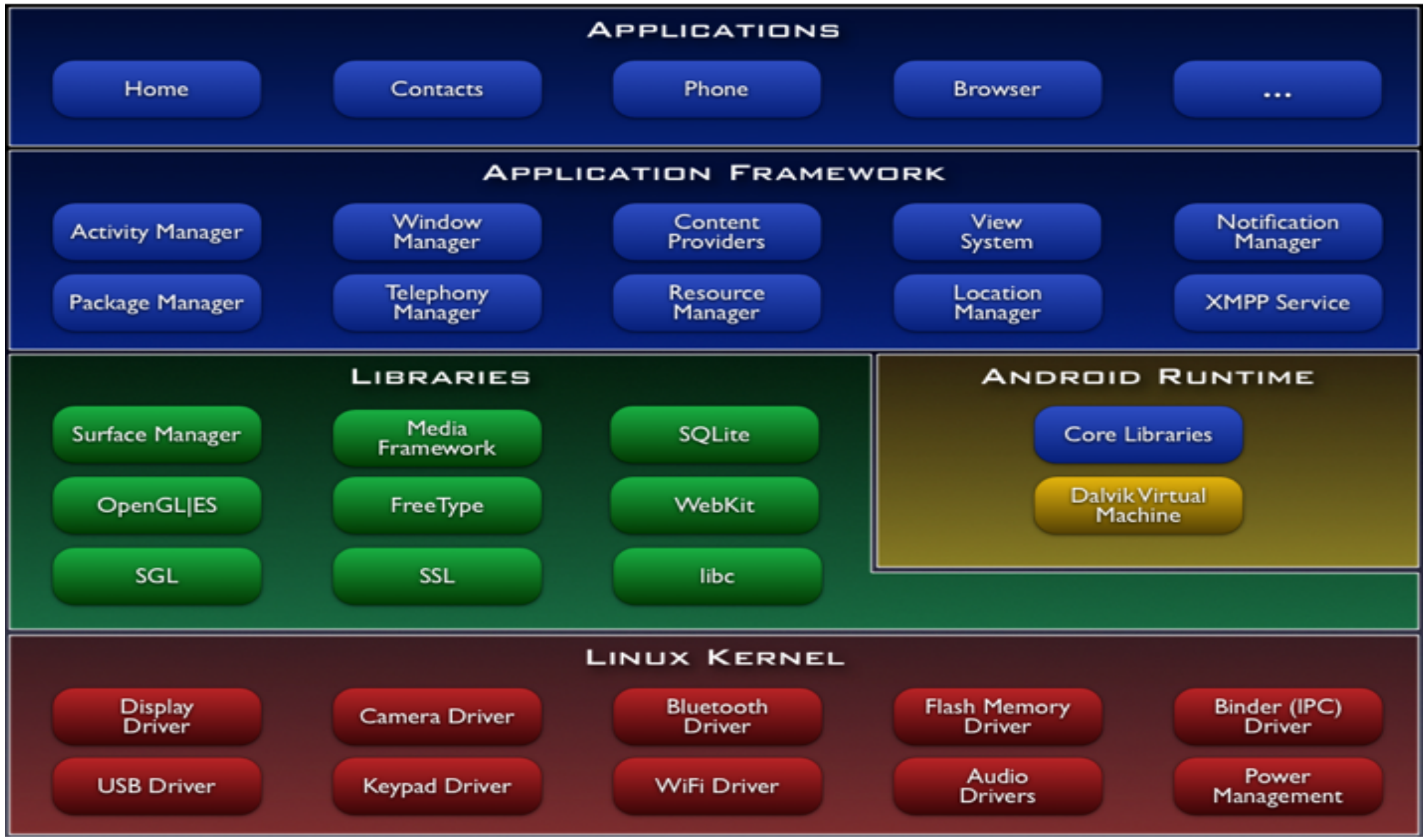
- Android architecture:
  - Run on top of Linux kernel
  - Use proprietary VM (Dalvik VM) as oppose to Java VM
  - Mutiple layers for different purpose
    - Application layer
    - Driver layer
    - Kernel layer

Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

INTRO : Android 101

- Android architecture



Securing Our Cyberspace

Copyright © 2011 CyberSecurity Malaysia

INTRO : APK 101

- Android package format
  - Bundle a few files into a file (.apk)
  - Just a zip file
  - Classes.dex is core file – compiled java classes.

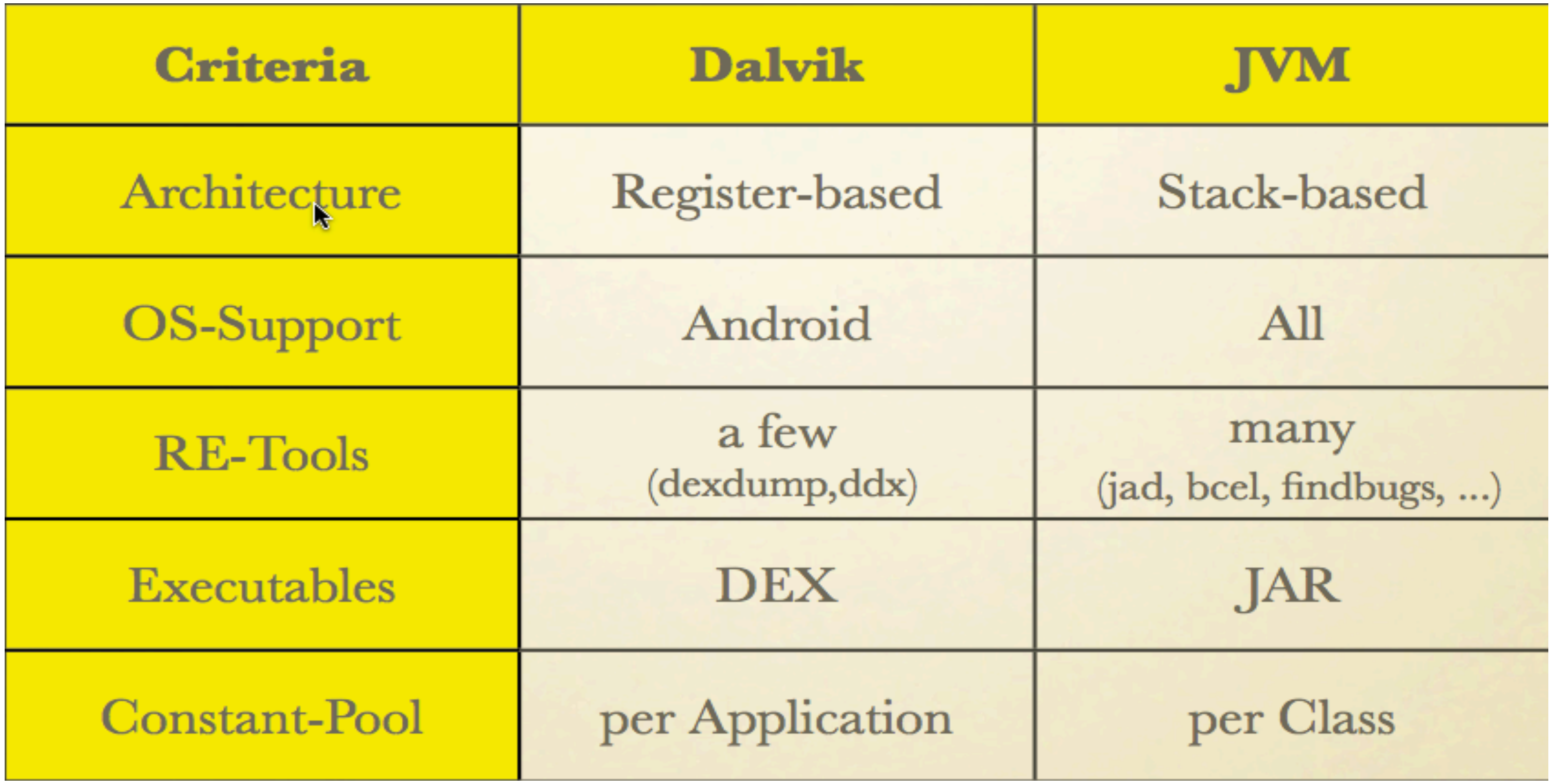
INTRO : Dalvik VM 101

- Run userspace Android applications
- Designed by Dan Bornstein
- Register based:
  - Faster than stack based register
- Run dalvik bytecode instead of Java bytecode
- Use "DX" tool to convert Java *.class to Dalvik bytecode

http://android.git.kernel.org/?p=platform/dalvik.git;a=tree.

INTRO : Dalvik VM 101

- Dalvik VM vs Java VM



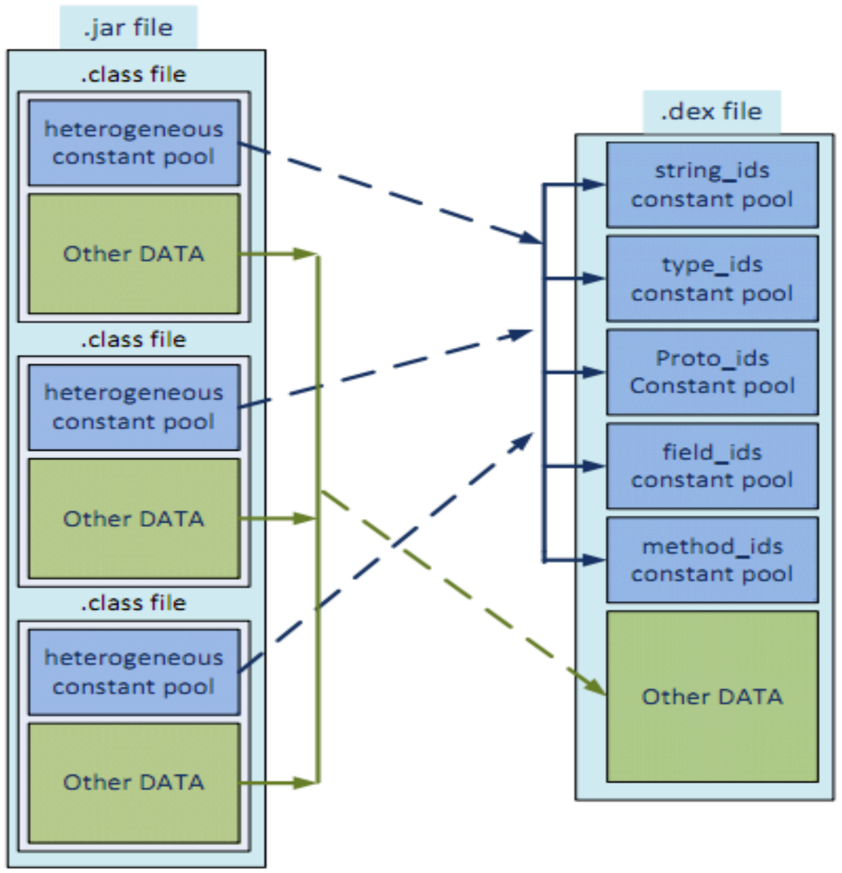
Criteria	Dalvik	JVM
Architecture	Register-based	Stack-based
OS-Support	Android	All
RE-Tools	a few (dexdump,ddx)	many (jad, bcel, findbugs, ...)
Executables	DEX	JAR
Constant-Pool	per Application	per Class

(Mark schoenefeld,2009)

Copyright © 2011 CyberSecurity Malaysia

INTRO : Dalvik VM 101

- Java *.classes to .dex file

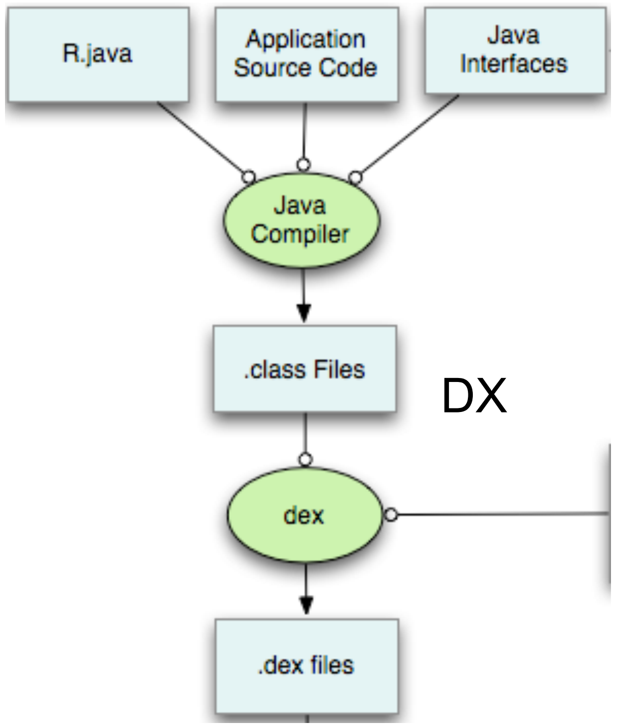


Securing Our Cyberspace

Copyright © 2011 CyberSecurity Malaysia

INTRO : DEX 101

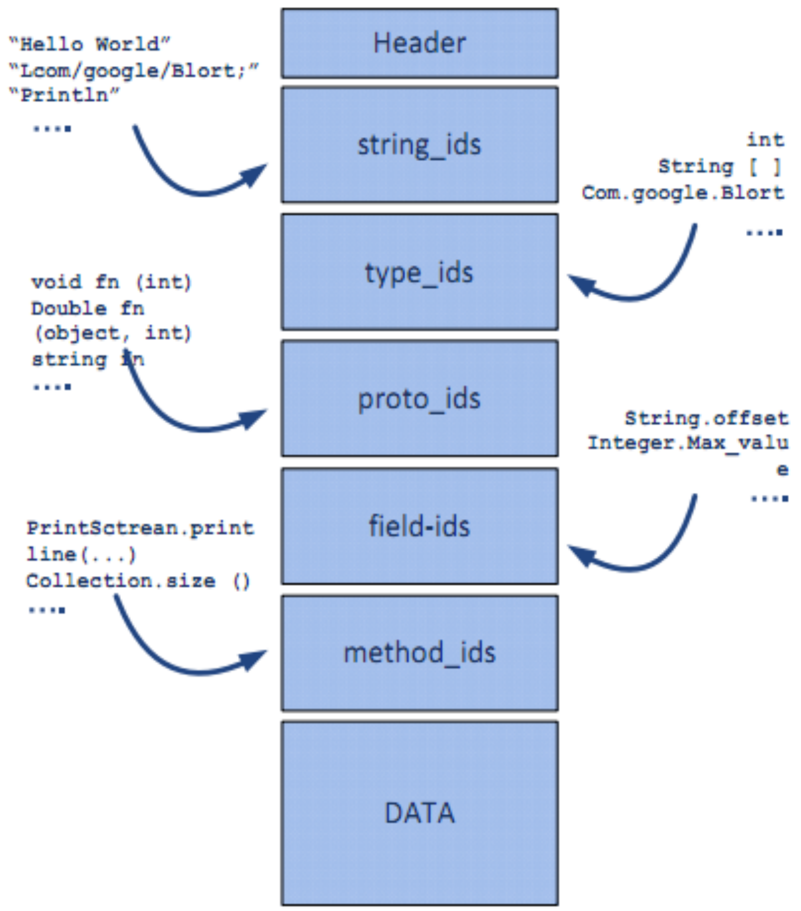
- Executable format for Android platform
- DEX process flow



Copyright © 2011 CyberSecurity Malaysia

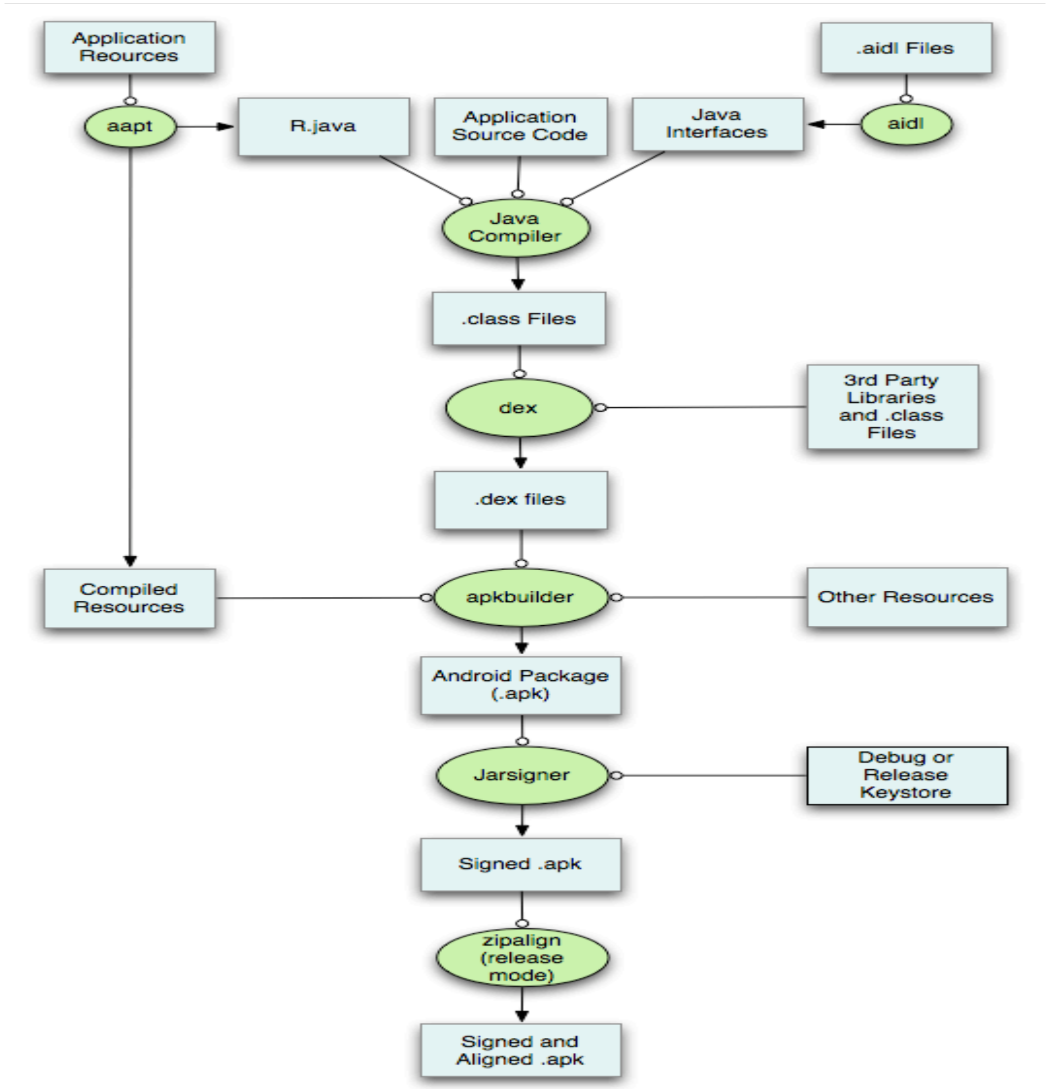
INTRO : DEX 101

- DEX file format
- *.odex
  - Optimized DEX



Copyright © 2011 CyberSecurity Malaysia

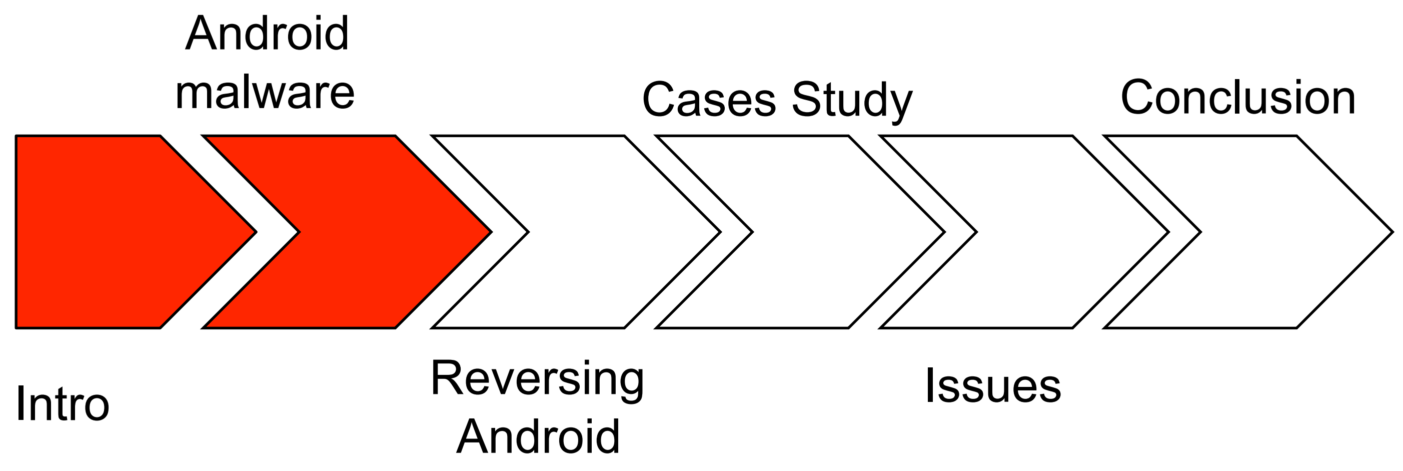
INTRO :Android Apps Building Process



Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

ANDROID MALWARE



Copyright © 2011 CyberSecurity Malaysia

CyberSecurity MALAYSIA

MOSTI Ministry of Science, Technology & Innovation

Android Malware

Copyright © 2011 CyberSecurity Malaysia

Android Malware

- Malicious piece of codes.
- Infection methods:
  - Infecting legitimate apps
    - Mod app with malicious codes (Geinimi, DreamDroid,ADDR)
    - Upload to “Market” or 3rd party hosting
  - Exploiting Android’s (core/apps) bugs
  - Fake apps
    - DreamDroid’s removal tool

Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

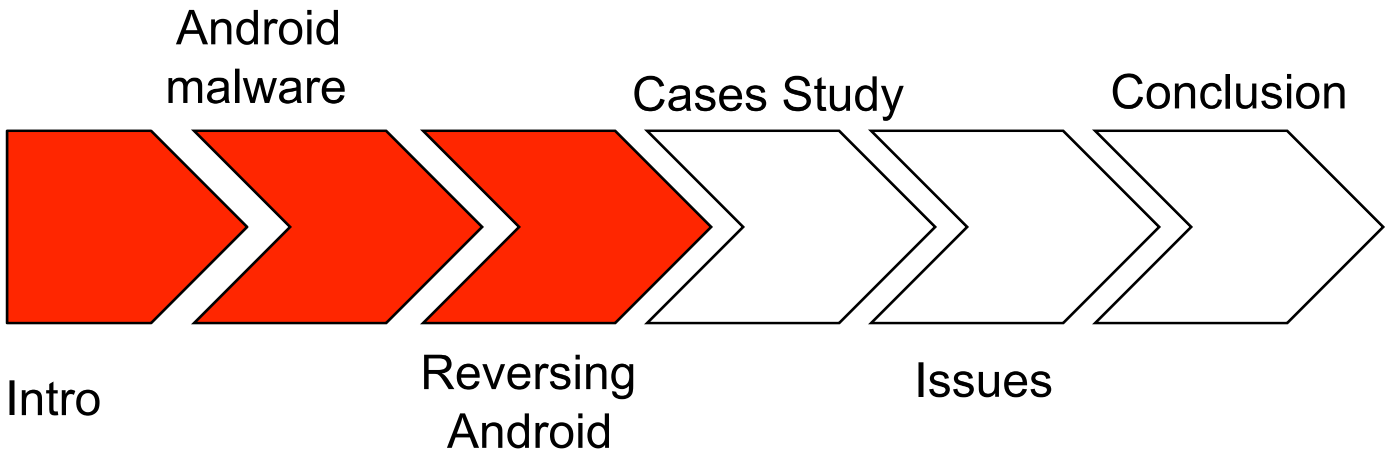
Android Malware

- Infection methods (cont):
  - Remote install?.
    - Victim’s gmail credential is required
    - Browse “Market” and pass gmail info
    - “Market” will install app into victim’s phone REMOTELY

http://www.net-security.org/article.php?id=1556

Copyright © 2011 CyberSecurity Malaysia

REVERSING ANDROID MALWARE



Copyright © 2011 CyberSecurity Malaysia

Reversing Android Malware

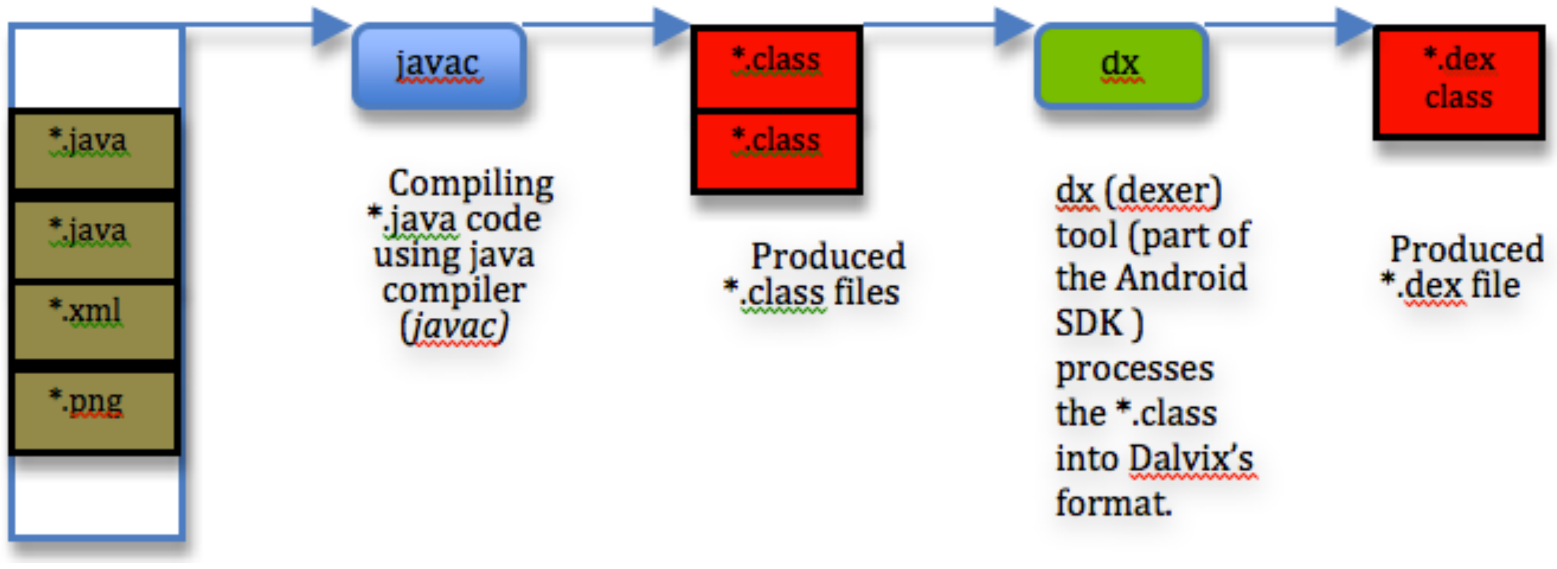
Reversing Android Malware

- Source Of Files
  - APK file
    - Can extract .DEX file
    - Reversing and interactive debugging is possible
      - ADB
  - DEX file
    - Only reversing is possible
    - Files for “res” + “asset” + etc are missing.

Copyright © 2011 CyberSecurity Malaysia

Reversing Android Malware

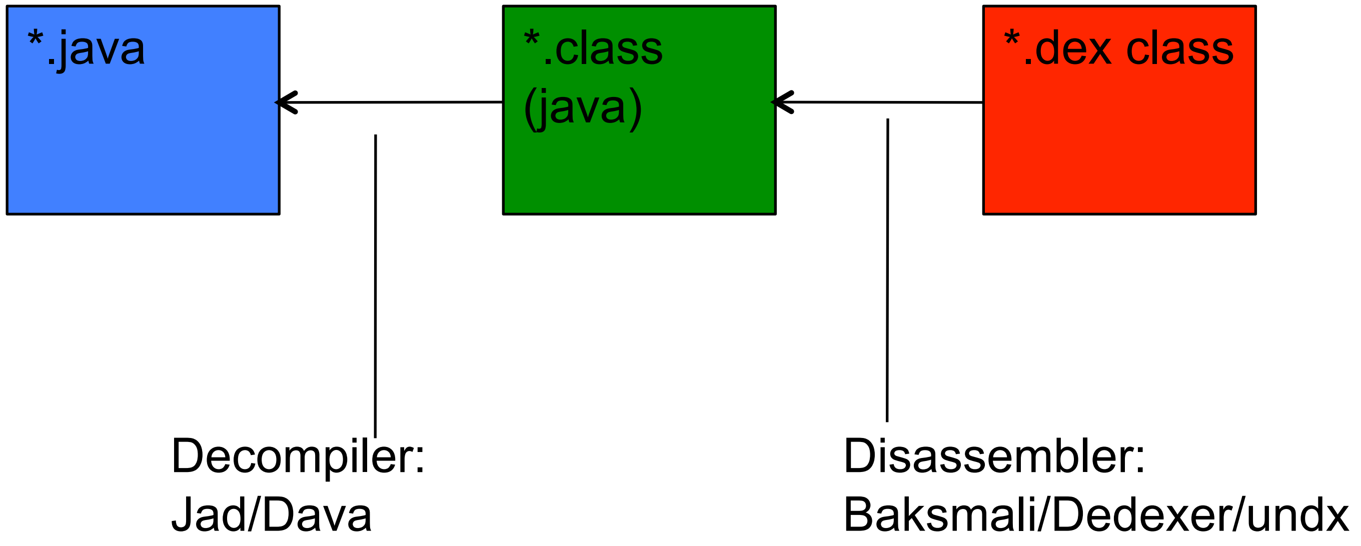
- Building Process


*.java
*.java
*.xml
*.png
javac
Compiling *.java code using java compiler (javac)
*.class
*.class
Produced *.class files
dx
dx (dexer) tool (part of the Android SDK) processes the *.class into Dalvix's format.
*.dex class
Produced *.dex file

Copyright © 2011 CyberSecurity Malaysia

Reversing Android Malware

- Reversing Process



Copyright © 2011 CyberSecurity Malaysia

Reversing Android Malware

- Tools
  - Disassembler- to dump Dalvik VM bytecode to assembly-like syntax
    - Dedexer
    - Baksmali
    - Undx
    - Dexdump – dumping *.dex file (from Android SDK)
  - Assembler- to convert to original Dalvik VM bytecode
    - Smali

Copyright © 2011 CyberSecurity Malaysia

Reversing Android Malware

- Tools (cont)
  - Text Editor – viewing the code
    - Use a decent one with baksmali/dedexer output highlighter
      - UltraEdit
      - Emacs
    - Notepad is fine. :-)
  - dex2jar
    - If you prefer Java than assembly-like output
    - Easy way to avoid complexity of Dalvik VM bytecode
    - May have errors interpreting Dalvik VM bytecode

Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

Reversing Android Malware

- Check on AndroidManifest.XML
  - Permission request
  - Entry point
- RE is solving a puzzle
  - Start with "names/strings"
    - "NET"
    - "CRYPTO"
    - "SERVER"
    - "IO"
  - Check on suspicious Android API
    - Location API, SMS API, Phone API, Mail API, Network API

Securing Our Cyberspace

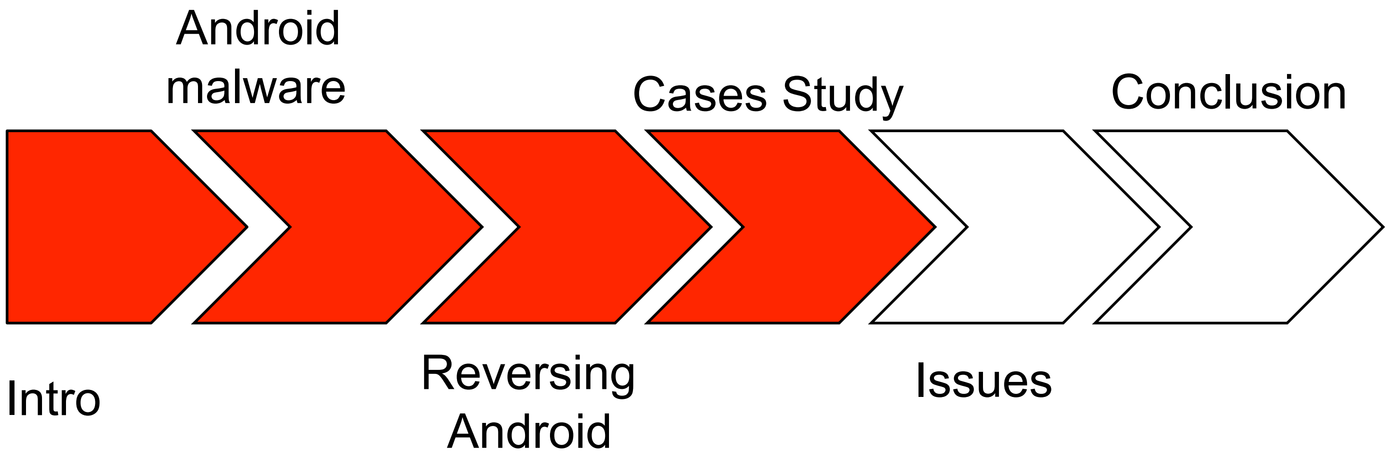
Copyright © 2011 CyberSecurity Malaysia

Reversing Android Malware

- Tracing function calls:
  - Browsing the codes and trace function call chains ("*XREF*")

Copyright © 2011 CyberSecurity Malaysia

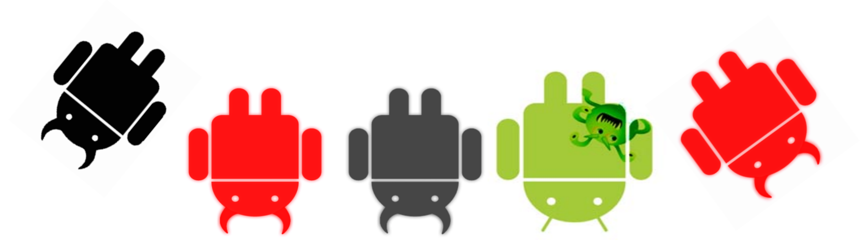
CASE STUDY



Copyright © 2011 CyberSecurity Malaysia

CASE STUDY

ANDROID MALWARE HAPPY FAMILY



Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

CASE #1: SMS.Trojan

- Oldest android malware (public)
- Very simple (follow HelloWorld Android SDK)
- Social engineering. It's by nature is malicious app
- Perform toll SMS fraud to Russia (premium shortcode)
  - Toll charges to enduser
  - Send to shartcode number "3353" and "3354"
- A good reason for AV on mobile ;)

Copyright © 2011 CyberSecurity Malaysia

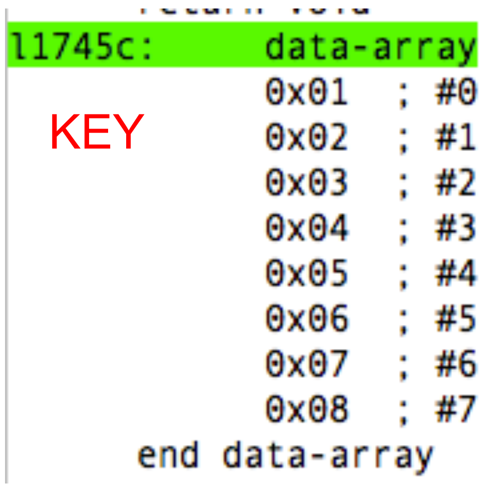
CASE #2: Geinimi

- Nice way to celebrate new year
  - Discovered close to new year eve!
- Modus Operandi
  - Infecting legitimate software
    - MonkeyJump2, Sex Positions, etc.etc
- Features:
  - Encryption- DES
  - C&C Servers
  - Info stealer
  - Bot capability
  - Encrypted communication

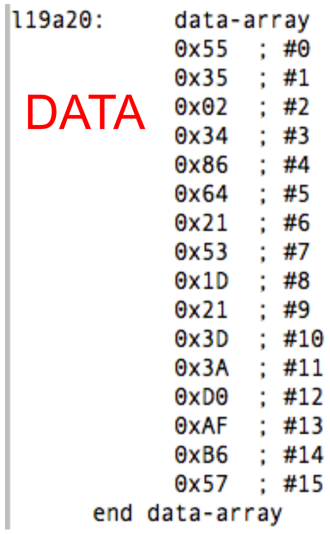
Copyright © 2011 CyberSecurity Malaysia

CASE #2: Geinimi (cont)

- Encryption- DES
  - Encrypted C&C and Data
  - DES with “01020304050608” key



```
11745c:     data-array
            0x01   ; #0
            0x02   ; #1
            0x03   ; #2
            0x04   ; #3
            0x05   ; #4
            0x06   ; #5
            0x07   ; #6
            0x08   ; #7
        end data-array
```



```
119a20:     data-array
            0x55   ; #0
            0x35   ; #1
            0x02   ; #2
            0x34   ; #3
            0x86   ; #4
            0x64   ; #5
            0x21   ; #6
            0x53   ; #7
            0x1D   ; #8
            0x21   ; #9
            0x3D   ; #10
            0x3A   ; #11
            0xD0   ; #12
            0xAF   ; #13
            0xB6   ; #14
            0x57   ; #15
        end data-array
```

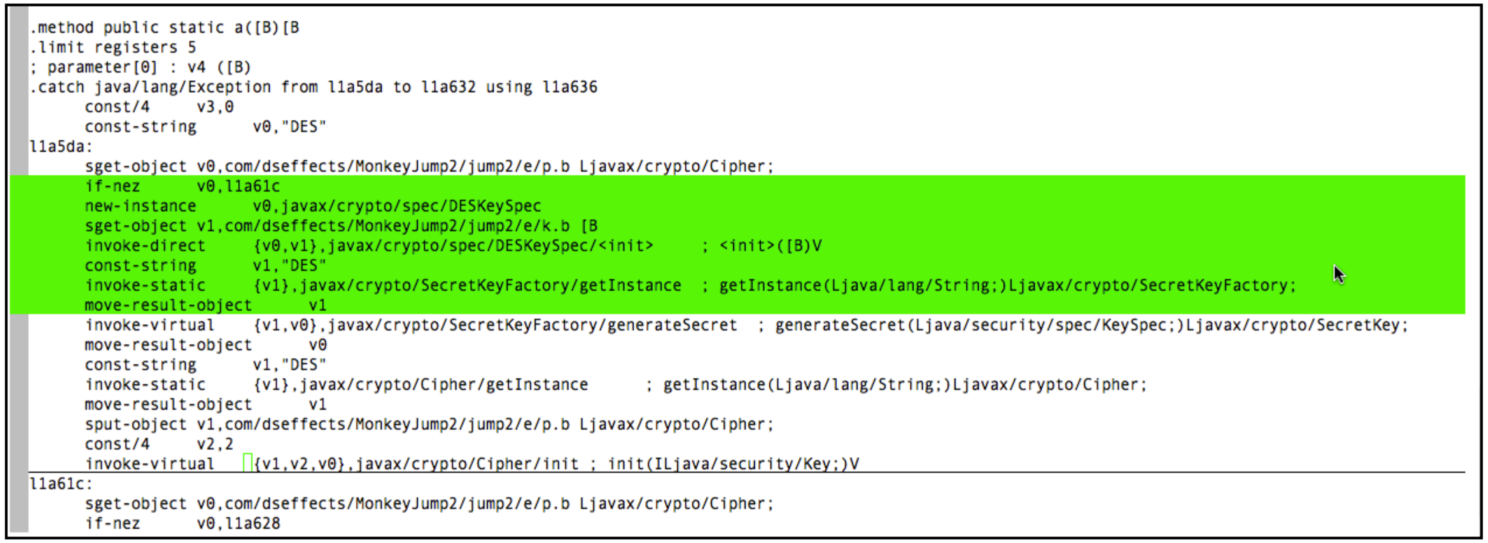
Copyright © 2011 CyberSecurity Malaysia

CyberSecurity MALAYSIA

MOSTI Ministry of Science, Technology & Innovation

CASE #2: Geinimi (cont)

- Encryption- DES
  - Encrypted C&C and Data
  - DES with “01234568” key

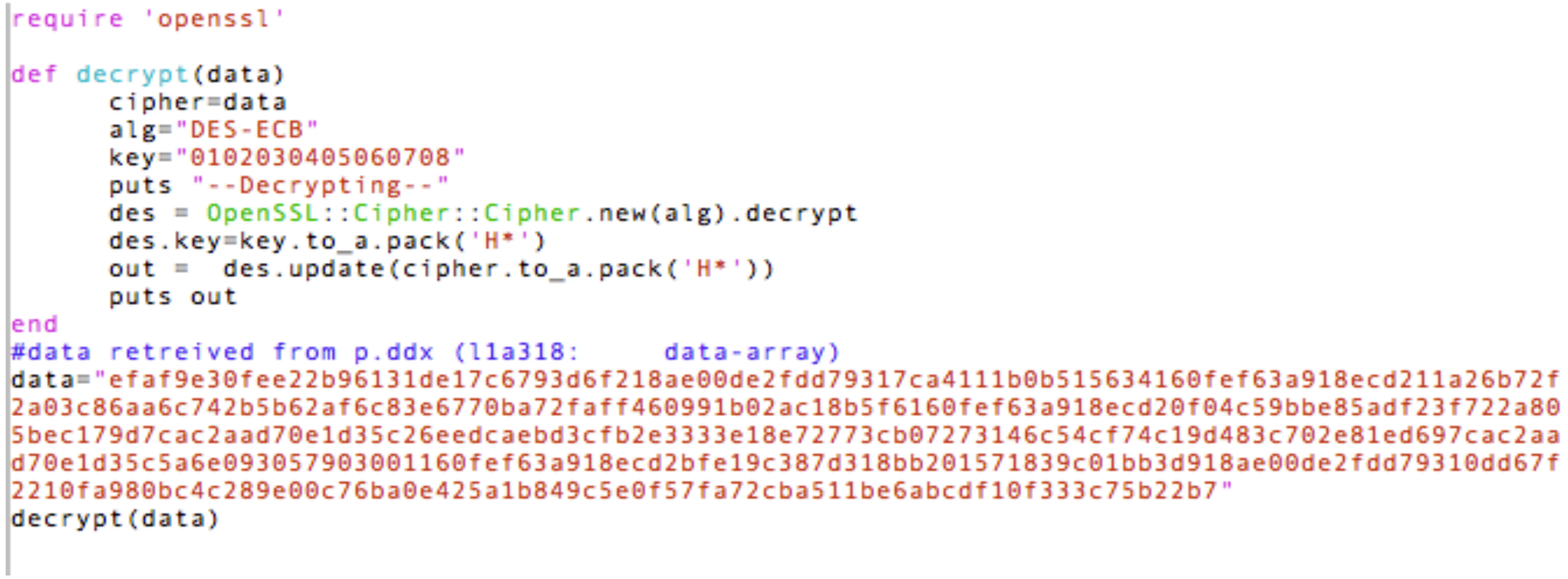


```
.method public static a([B)[B
.limit registers 5
; parameter[0] : v4 ([B)
.catch java/lang/Exception from l1a5da to l1a632 using l1a636
      const/4     v3,0
      const-string      v0,"DES"
l1a5da:
      sget-object v0,com/dseffects/MonkeyJump2/jump2/e/p.b Ljavax/crypto/Cipher;
      if-nez      v0,l1a61c
      new-instance      v0,javax/crypto/spec/DESKeySpec
      sget-object v1,com/dseffects/MonkeyJump2/jump2/e/k.b [B
      invoke-direct     {v0,v1},javax/crypto/spec/DESKeySpec/<init>     ; <init>([B)V
      const-string      v1,"DES"
      invoke-static     {v1},javax/crypto/SecretKeyFactory/getInstance   ; getInstance(Ljava/lang/String;)Ljavax/crypto/SecretKeyFactory;
      move-result-object      v1
      invoke-virtual    {v1,v0},javax/crypto/SecretKeyFactory/generateSecret   ; generateSecret(Ljava/security/spec/KeySpec;)Ljavax/crypto/SecretKey;
      move-result-object      v0
      const-string      v1,"DES"
      invoke-static     {v1},javax/crypto/Cipher/getInstance      ; getInstance(Ljava/lang/String;)Ljavax/crypto/Cipher;
      move-result-object      v1
      sput-object v1,com/dseffects/MonkeyJump2/jump2/e/p.b Ljavax/crypto/Cipher;
      const/4     v2,2
      invoke-virtual    {v1,v2,v0},javax/crypto/Cipher/init ; init(ILjava/security/Key;)V
l1a61c:
      sget-object v0,com/dseffects/MonkeyJump2/jump2/e/p.b Ljavax/crypto/Cipher;
      if-nez      v0,l1a628
```

Copyright © 2011 CyberSecurity Malaysia

CASE #2: Geinimi (cont)

- Reversing DES encryption



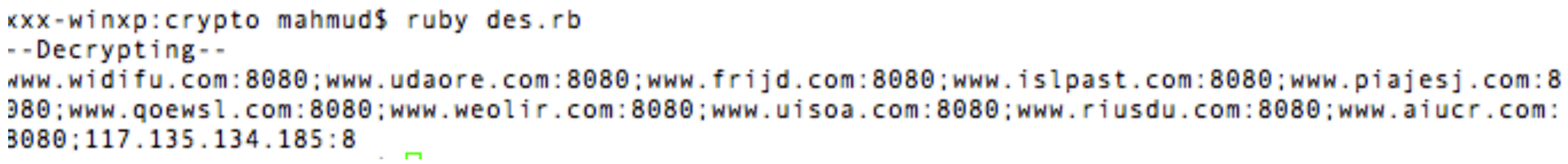
```
require 'openssl'

def decrypt(data)
        cipher=data
        alg="DES-ECB"
        key="0102030405060708"
        puts "--Decrypting--"
        des = OpenSSL::Cipher::Cipher.new(alg).decrypt
        des.key=key.to_a.pack('H*')
        out =  des.update(cipher.to_a.pack('H*'))
        puts out
end
#data retreived from p.ddx (l1a318:      data-array)
data="efaf9e30fee22b96131de17c6793d6f218ae00de2fdd79317ca4111b0b515634160fef63a918ecd211a26b72f
2a03c86aa6c742b5b62af6c83e6770ba72faff460991b02ac18b5f6160fef63a918ecd20f04c59bbe85adf23f722a80
5bec179d7cac2aad70e1d35c26eedcaebd3cfb2e3333e18e72773cb07273146c54cf74c19d483c702e81ed697cac2aa
d70e1d35c5a6e093057903001160fef63a918ecd2bfe19c387d318bb201571839c01bb3d918ae00de2fdd79310dd67f
2210fa980bc4c289e00c76ba0e425a1b849c5e0f57fa72cba511be6abcdf10f333c75b22b7"
decrypt(data)
```

Copyright © 2011 CyberSecurity Malaysia

CASE #2: Geinimi (cont)

- Encryption- DES (result))



```
xxx-winxp:crypto mahmud$ ruby des.rb
--Decrypting--
www.widifu.com:8080;www.udaore.com:8080;www.frijd.com:8080;www.islpast.com:8080;www.piajesj.com:8
080;www.qoewsl.com:8080;www.weolir.com:8080;www.uisoa.com:8080;www.riusdu.com:8080;www.aiucr.com:
8080;117.135.134.185:8
```

DECRYPTED DATA

Copyright © 2011 CyberSecurity Malaysia

CASE #2: Geinimi (cont)

- Info stealer
  - Steal info and pass to C&C Server
  - Encrypted data
  - Steal data:
    - IMEI,IMSI
    - GEOLocation (lat,long)
    - SMS
    - Contacts list
    - Installed apps list

Copyright © 2011 CyberSecurity Malaysia

CASE #2: Geinimi (cont)

- Bot capability
  - Received commands from C&C server
    - dsms – Delete SMS(es)
    - Smsrecord – steal sms record and pass to C&C
    - showurl – Open browser with URL
    - Call – make a call to number
    - Install – install apps
  - State for bot
    - Start, download,parse, transact,Idle

Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

CASE #2: Geinimi (cont)

- Encrypted communication
  - Every data receive/sent are encrypted
  - Embeded into "*params*" parameter for sending encrypted data

Copyright © 2011 CyberSecurity Malaysia

CASE #2: Geinimi (cont)

- Backdoor
  - TCP socket on ports 5432, 4501 or 6543
  - Another back door on port 8791
    - Send a "hi,xiaolu” response message to listener
    - Send a “hi,liqian” response message to request
    - Run at loopback interface. ;)

Copyright © 2011 CyberSecurity Malaysia

CASE #3: ADDR

- Tagged Image File Format (abbreviated TIFF)
- file format for storing images
- it is under the control of Ad(0day)be Systems (2009)
- widely supported by image-manipulation application

Copyright © 2011 CyberSecurity Malaysia

CASE #3: DreamDroid

- Latest addition to android malware family
- Modus Operandi
  - Infecting legitimate software
  - Hosted at “Market”
  - 53 software infected
- Bundled with exploits to “root” the Android
  - Exploid (CVE-2009-1185)
  - Rageagaintsthecage (CVE-2010-EASY)
- Bot capability

Copyright © 2011 CyberSecurity Malaysia

CASE #3: DreamDroid (cont)

- Features:
  - Encrypted communication (XOR)
  - Encrypted data
  - Bot capability
  - Two stage payloads
    - 1st Payload - Infected app
      - Rooted device
      - Install 2nd payload (DownloadProviderManager)
    - 2nd Payload - DownloadProviderManager
      - Sqllite.db (original filename)
      - Receive instructions from C&C
      - Send info to C&C
      - Silently install itself (copy to */system/app* directory)

Copyright © 2011 CyberSecurity Malaysia

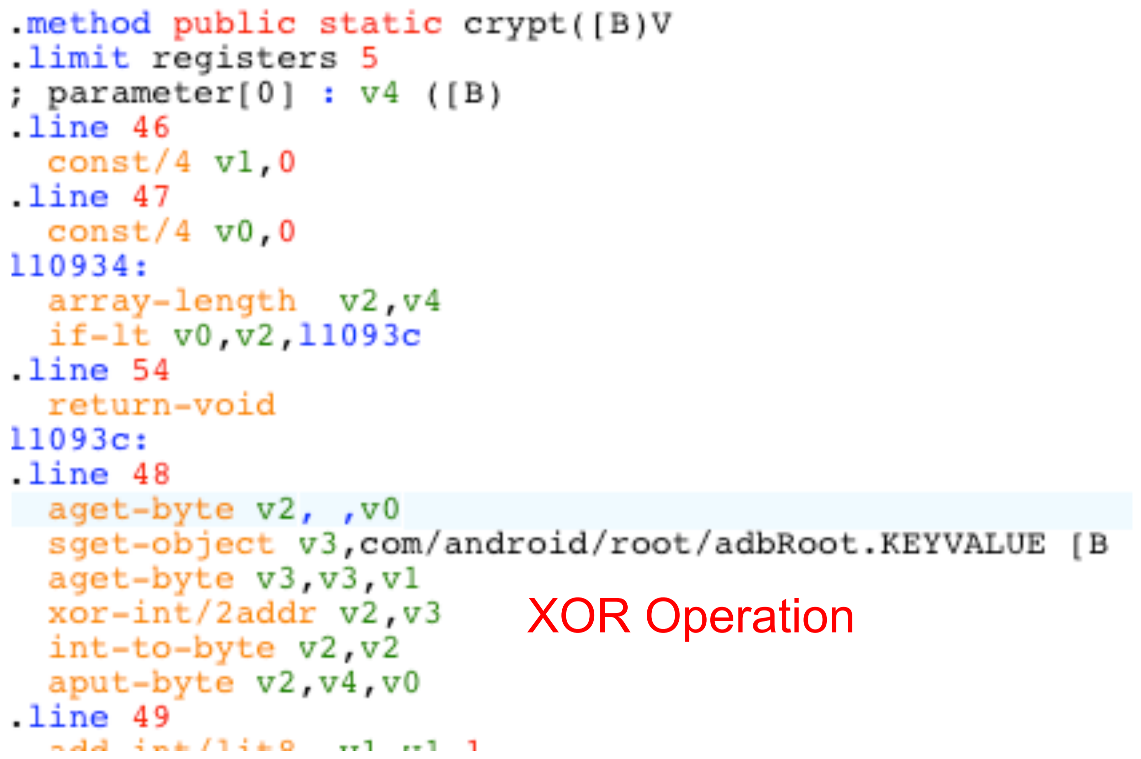
CASE #3: DreamDroid (cont)

- Encryption
  - XOR operation
    - KEY=“6^)(9-p35a%3#4S!4S0)$Yt%^&5(j.g^&o(*0)$Yv!#O@6GpG@=+3j.&6^)(0-=1”.*getBytes()*
    - DATA= “9442938832952138511219112519102302419997621102222611139125244801090511910011960487794252”
  - Revealed C&C server
    - http://184.105.245.17:8080/GMServer/GMServlet
- Send IMEI,IMSI, Device Model, SDK Version to C&C server

Copyright © 2011 CyberSecurity Malaysia

CASE #3: DreamDroid (cont)

- Encryption

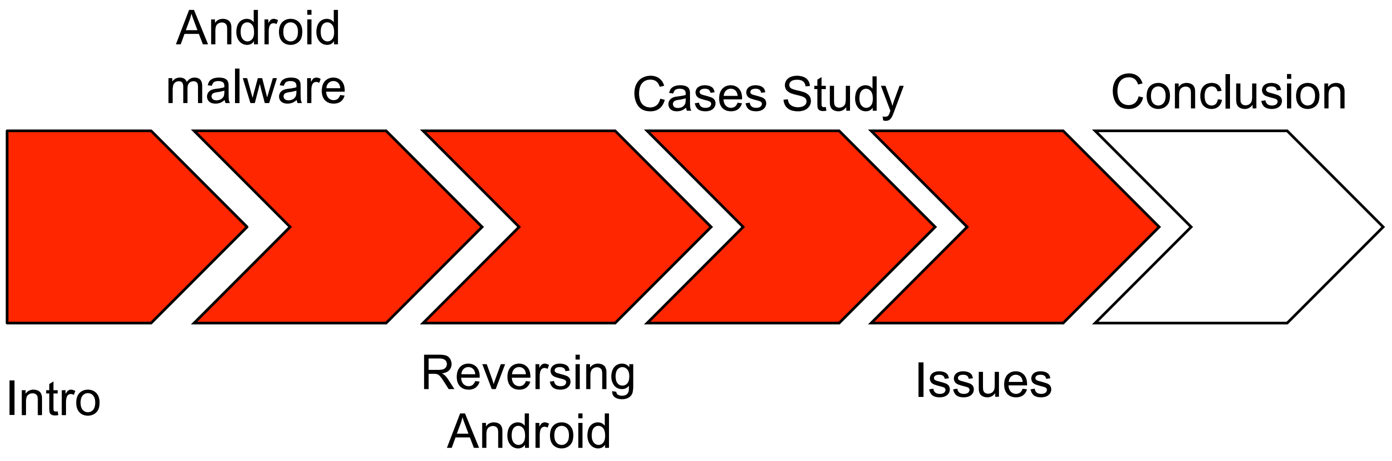


```
.method public static crypt([B)V
.limit registers 5
; parameter[0] : v4 ([B)
.line 46
  const/4 v1,0
.line 47
  const/4 v0,0
l10934:
  array-length  v2,v4
  if-lt v0,v2,l1093c
.line 54
  return-void
l1093c:
.line 48
  aget-byte v2, ,v0
  sget-object v3,com/android/root/adbRoot.KEYVALUE [B
  aget-byte v3,v3,v1
  xor-int/2addr v2,v3
  int-to-byte v2,v2
  aput-byte v2,v4,v0
.line 49
  add-int/lit8  v1,v1,1
```

Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

CHALLENGES AND ISSUES



Copyright © 2011 CyberSecurity Malaysia

Challenges and Issues

- Typical Reverse engineering challenges
  - Code obfuscation
    - Obfuscation on data
  - Encryption
    - Make it harder
    - Eventually will be broken (as for current sample)
  - Code optimizing
    - Code for device, painful for RE
- Tools is not yet mature
  - IDA PRO like RE suite
  - XREF

Securing Our Cyberspace

Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

Challenges and Issues

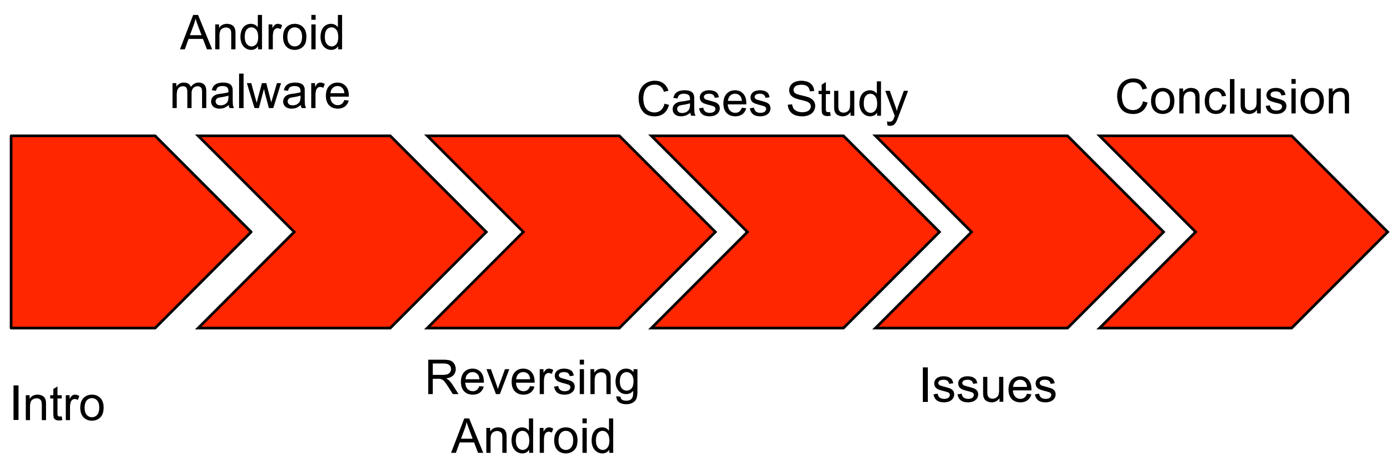
- Spotting the malicious apps
  - Not RE problem but how do you spot the malicious app?.
- Remote Install via “Market” would be interesting to observe

Copyright © 2011 CyberSecurity Malaysia

CyberSecurity MALAYSIA

MOSTI
Ministry of Science, Technology & Innovation

CONCLUSION



Copyright © 2011 CyberSecurity Malaysia

Conclusion

- Android malware is interesting topic
  - More complex android malware are expected
  - More exploits on Android platform are expected
  - More powerful hardware will change the landscape!
- It is possible to reverse engineering Android malware
  - A lot of free tools to reverse engineering android apps/malware
  - Solving a puzzle. PERIOD
- Reversing tools are there, but yet to mature

Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

Q&A

Copyright © 2011 CyberSecurity Malaysia

Ministry of Science, Technology & Innovation

THANKS

Email: mahmud@cybersecurity.my
Web: http://www.cybersecurity.my
Web: http://www.mycert.org.my
Web: www.cybersafe.my
Report Incident: mycert@mycert.org.my

Copyright © 2011 CyberSecurity Malaysia