Python, Android, Java, 0day, 0mai

TLP-RED


UR COMPUTER
HAZ MEOWARE

Targeted Attack Vectors

- Win32 Executable in archive (zip/rar)
- Java RAT as .jar attachment or in archive
- CVE-2012-0158/CVE-2014-1761 RTF attachment
- Link to Win32 executable in archive (zip/rar)
- CVE-2014-4114 PPSX attachment
- Malicious APK attachment/link
- Links to exploit kits

Python is so Hot

Snake In The Grass: Python-based Malware Used For Targeted Attacks

Snorre Fagerland, Jonathan Camp, Morten Kråkvik - June 10, 2014

Researchers at Blue Coat Systems have identified an intelligence-gathering campaign related to the Hangover operation detailed in 2013. The targets of this operation appear to be Pakistani and presumably represent military interests.

Why is it hot?

```
def getserver():
    if not path.exists(dir2+"ip.txt"):
        encryption = "jGKfohkJpJ9KGS+rT+1glg=="    # server name
        enc = open(dir2+"ip.txt",'wb')
        enc.write(encryption)
        enc.close()
    else:
        enc = open(dir2+"ip.txt",'rb')
        encryption = enc.read()
        enc.close()
    PADDING = '{'
    DecodeAES = lambda c, e: c.decrypt(b64decode(e)).rstrip(PADDING)
    key = "123456789!@#$%^&"      # key
    cipher = AES.new(key)
    decoded = DecodeAES(cipher, encryption)
    return decoded
```

PyInstaller Based Malware

- PyInstaller is an open source tool for turning your python code into a hugely bloated Win32 executable

About

PyInstaller is a program that converts (packages) Python programs into stand-alone executables, under Windows, Linux, Mac OS X, Solaris and AIX. Many packages are supported "out of the box". Check our compatibility list of Supported Packages. For more details about PyInstaller see About page.

Feel free to join us in the effort. Please consult our Milestones to check our plans. Also usage reports are welcomed: let us know if PyInstaller works for you and how, or what problems you found in using it.

Check our list of Projects Using PyInstaller.

Who’s using pyinstaller?

- Turkish based cybercriminals
  - aka BePush
    - previously used AutoHotKey
- APT actors Several python implants
  - downloaders, RATS, validators
- Volatility, Dropbox and other legit uses

```
mjr-mbp:appin2 mjr$ python /data/tools/pyinstaller/utils/archive_viewer.py propstar.exe
fatal: Not a git repository (or any of the parent directories): .git
 pos, length, uncompressed, iscompressed, type, name
[(0, 1579314, 1579314, 0, 'z', 'out00-PYZ.pyz'),
 (1579314, 170, 234, 1, 'm', 'struct'),
 (1579484, 1147, 2627, 1, 'm', 'pyi_os_path'),
 (1580631, 4787, 12757, 1, 'm', 'pyi_archive'),
 (1585418, 3967, 13558, 1, 'm', 'pyi_importers'),
 (1589385, 1800, 4228, 1, 's', '_pyi_bootstrap'),
 (1591185, 4173, 13142, 1, 's', 'pyi_carchive'),
 (1595358, 2364, 8360, 1, 's', 'fresh'),
 (1597722, 602, 1857, 1, 'b', 'Microsoft.VC90.CRT.manifest'),
 (1598324, 317595, 655872, 1, 'b', 'msvcr90.dll'),
 (1915919, 155722, 568832, 1, 'b', 'msvcp90.dll'),
 (2071641, 66835, 224768, 1, 'b', 'msvcm90.dll'),
 (2138476, 1134500, 2454016, 1, 'b', 'python27.dll'),
 (3272976, 16883, 29184, 1, 'b', 'Crypto.Cipher._AES.pyd'),
 (3289859, 10623, 24064, 1, 'b', 'win32pipe.pyd'),
 (3300482, 17395, 36352, 1, 'b', '_psutil_mswindows.pyd'),
 (3317877, 5956, 11776, 1, 'b', 'select.pyd'),
 (3323833, 258305, 688128, 1, 'b', 'unicodedata.pyd'),
 (3582138, 11443, 25088, 1, 'b', 'win32wnet.pyd'),
 (3593581, 137063, 287232, 1, 'b', '_hashlib.pyd'),
 (3730644, 36825, 71680, 1, 'b', 'bz2.pyd'),
 (3767469, 354339, 721408, 1, 'b', '_ssl.pyd'),
 (4121808, 35537, 73216, 1, 'b', '_ctypes.pyd'),
 (4157345, 40160, 100352, 1, 'b', 'win32api.pyd'),
 (4197505, 19324, 40960, 1, 'b', '_socket.pyd'),
 (4216829, 42781, 110080, 1, 'b', 'pywintypes27.dll'),
 (4259610, 321, 730, 1, 'b', 'fresh.exe.manifest')]
?
```

“I will now do a demo”

–mjr

Detection

```
rule pyinstaller_magic
{
meta:
  description = "the pyinstaller magic
signature is located 88 bytes from the end of
the file"
strings:
  $a = { 4d 45 49 0c 0b 0a 0b 0e }
condition:
  $a at filesize-88
}
```

Family Detection

- Each entry in the pyinstaller archive has a table of contents entry like this one for downloader. The format is ENTRYSTRUCT = '!iiiibc' # (structlen, dpos, dlen, ulen, flag, typcd) followed by name.

- { 00 00 00 20 [12] 01 73 name 00 } where name is the hex encoded name, assuming strlen < 15.

  - These signatures could be a little flaky if used with a common name. Might want to use the manifest name toc entry or some library imports to make it more resilient.

```
00600e40 00 20 00 15 c8 ef 00 00 01 c2 00 00 03 5c 01 73 |. ............\.s|

00600e50 64 6f 77 6e 6c 6f 61 64 65 72 00 00 00 00 00 00 |downloader......|
```

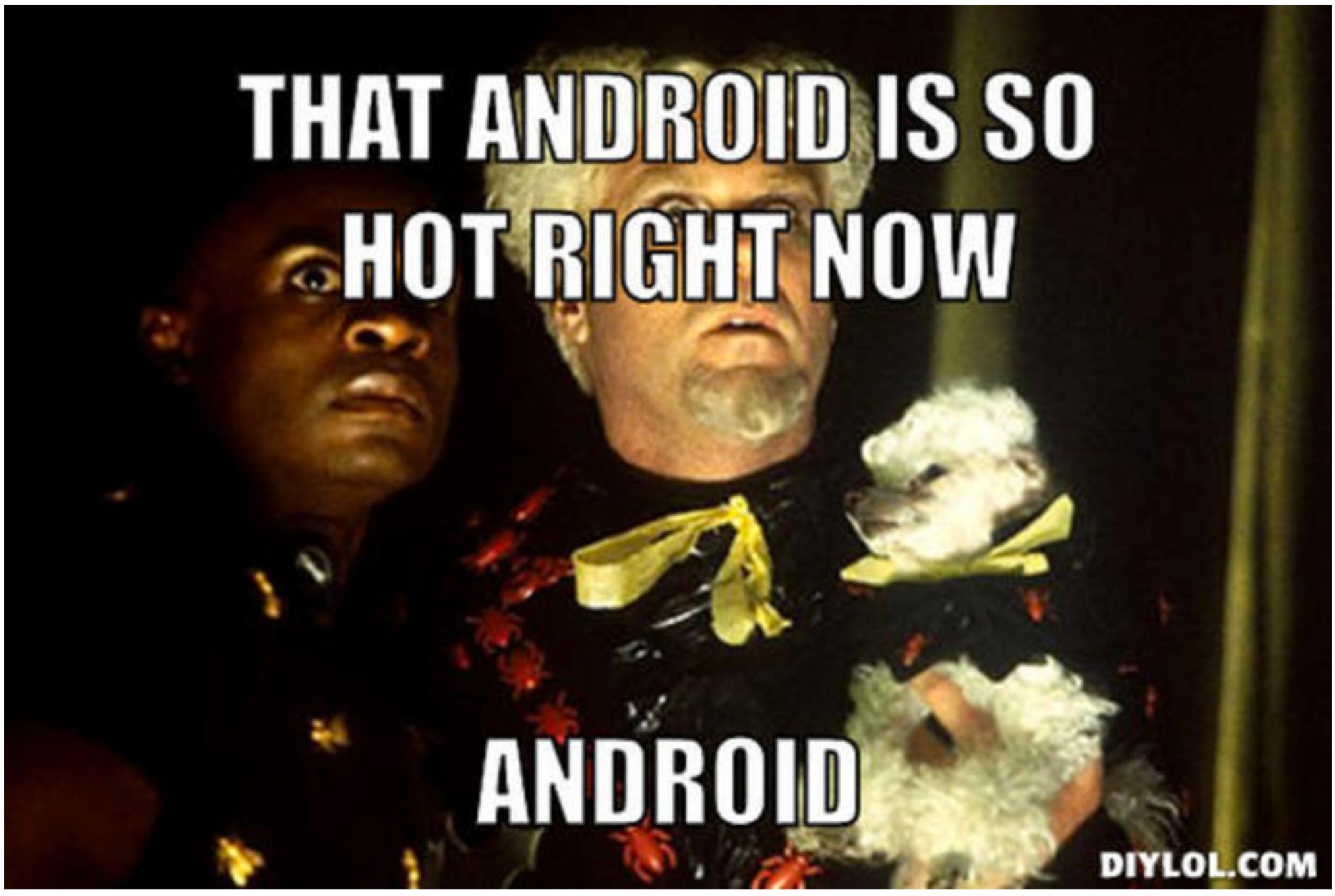
Family Detection

```
rule turkish_malware_pyinstaller
{
strings:
  $magic = { 4d 45 49 0c 0b 0a 0b 0e }
  // downloader toc entry
  $filename_downloader = { 00 00 00 20 [12] 01 73 64 6f 77 6e 6c 6f 61 64 65 72 00 }
  // servant toc entry
  $filename_servant = { 00 00 00 20 [12] 01 73 73 65 72 76 61 6e 74 00 }
condition:
  $magic at filesize-88 and any of ($filename*)
}
```

Some other things

- Imphash of exe loader
- Rich headers of exe loader
- Create a fuzzy hash of libraries
- Metadata. All. The. Things. Coming. Soon.

Actually, Android is so Hot



Why Android?

Alternative Distribution Options

As an open platform, Android offers choice. You can distribute your Android apps to users in any way you want, using any distribution approach or combination of approaches that meets your needs. From publishing in an app marketplace to serving your apps from a web site or emailing them directly users, you're never locked into any particular distribution platform.

The process for building and packaging your apps for distribution is the same, regardless of how you distribute them. This saves you time and lets you automate parts of the process as needed. You can read Preparing for Release for more information.

The sections below highlight some of the alternatives for distributing your apps.

Android Tools - TMP_

Wireless Keys(ROOT) | Chrome Password Recovery(ROOT) | Wireless AP | Whatsapp (ROOT)

Pages viewed | Microphone | Location | Screen capture (ROOT) | Take photo | VNC Viewer Password(ROOT)

Installed Apps | Calls | Contacts | SMS Manager | Funny options | Advanced Info | Wallpaper

Name	...
Contactos	com.android.contacts
Teléfono	com.android.dialer
Mensajería	com.android.mms
Configuración	com.android.settings
Navegador	com.android.browser
File Manager	com.cyanogenmod.filemanager
Calculadora	com.android.calculator2
Calendario	com.android.calendar
Reloj	com.android.deskclock
Música	com.cyanogenmod.eleven
Correo	com.android.email
Galería	com.android.gallery3d
Grabadora de sonido	com.android.soundrecorder
Terminal	com.android.terminal
Google Play Music	com.google.android.music
Google Play Películas	com.google.android.videos
Google Play Services	com.google.android.gms
Google+	com.google.android.apps.plus
Gmail	com.google.android.gm
YouTube	com.google.android.youtube
Hangouts	com.google.android.talk
Google Play Store	com.android.vending
Chrome	com.android.chrome

Android Malware ITW

- AlienSpy Android Pro (???)
- Hangover custom
- APT written implants
- Dendroid

Android Distribution

- Google Play Store
- APK Attachments
- Links to APKs on (Dropbox|custom site|internetz)
- Repack existing apps / shell apps

Running Strings on APK

Show apps

A tool for reverse engineering Android apk files

```
$ apktool d test.apk
I: Using Apktool 2.0.0 on test.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: 1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
$ apktool b test
I: Using Apktool 2.0.0 on test
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
$
```

“I will now do a demo”

–mjr

Detection

- APK is:
  - resource files (images, xml, etc)
  - classes.dex (Dalvik compiled Java bytecode)
  - manifest
  - a zip file

Java RATs

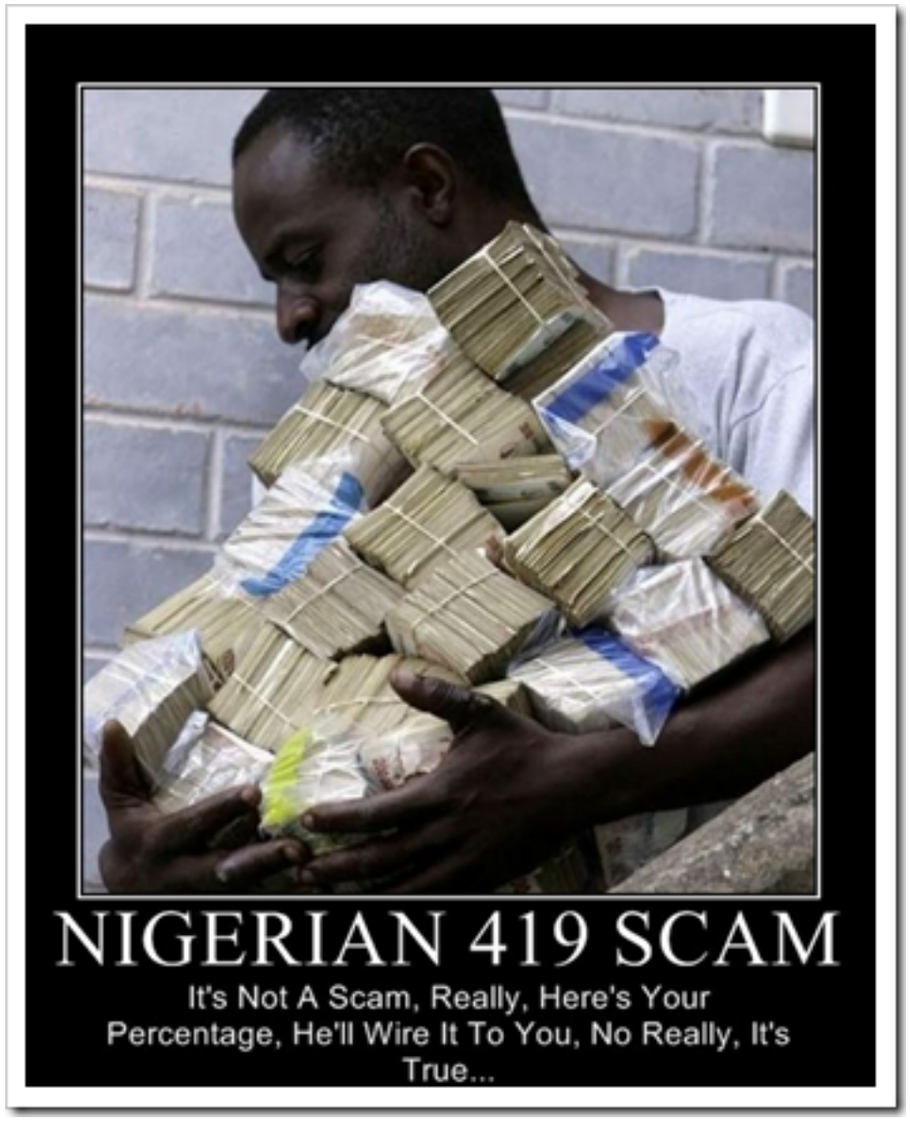
- AlienSpy
- Adwind
- Unrecom
- JRAT
- Custom downloaders

Why Java?

- Non-mal-mis-hack-tribution
- Cross Platform
- Executes with user privs
- On Windows you can download/execute 1-click
- Not monitored by a lot of tools
  - CB/Bit9/etc
- Robust ecosystem
- Easy to get low detection (lack of AV emulator support)

Who’s sending Java?

- 419 / ACH Scammers
- Greek cybercrime
- APT
- A lot of stuff for which I have no idea
- Me



Detection

```
bad_zipfile_names = '|'.join([
    '\"com/java/Main.class\"',
    '\"com/java/Manifest.mf\"',
    '\"key.dat\"',
    '\"enc.dat\"',
    '\"jcrypt/Decrypter.class\"',
    '\"MemoryLoader.class\"',
    '\"blaz42/',
    '\"plugins/Server.class\"',
    '\"e\"',
    '\"k\"',
    ])
```

```
rule jrat_zip
{
  meta:
    author = "mjr@fb.com"
    share_level = "GREEN"
    description = "looks for jrat artifacts in zi
    source = "facebook"
  strings:
    $a = { 50 4b 03 04 }
    $b1 = "Decryptor.classPK"
    $b2 = "MemoryLoader.classPK"
    $b3 = "enc.datPK"
    $b4 = "key.datPK"
    $b5 = "Config.classPK"
    $b6 = "jcrypt/Decrypter.classPK"
  condition:
    $a at 0 and 2 of ($b*)
}
```

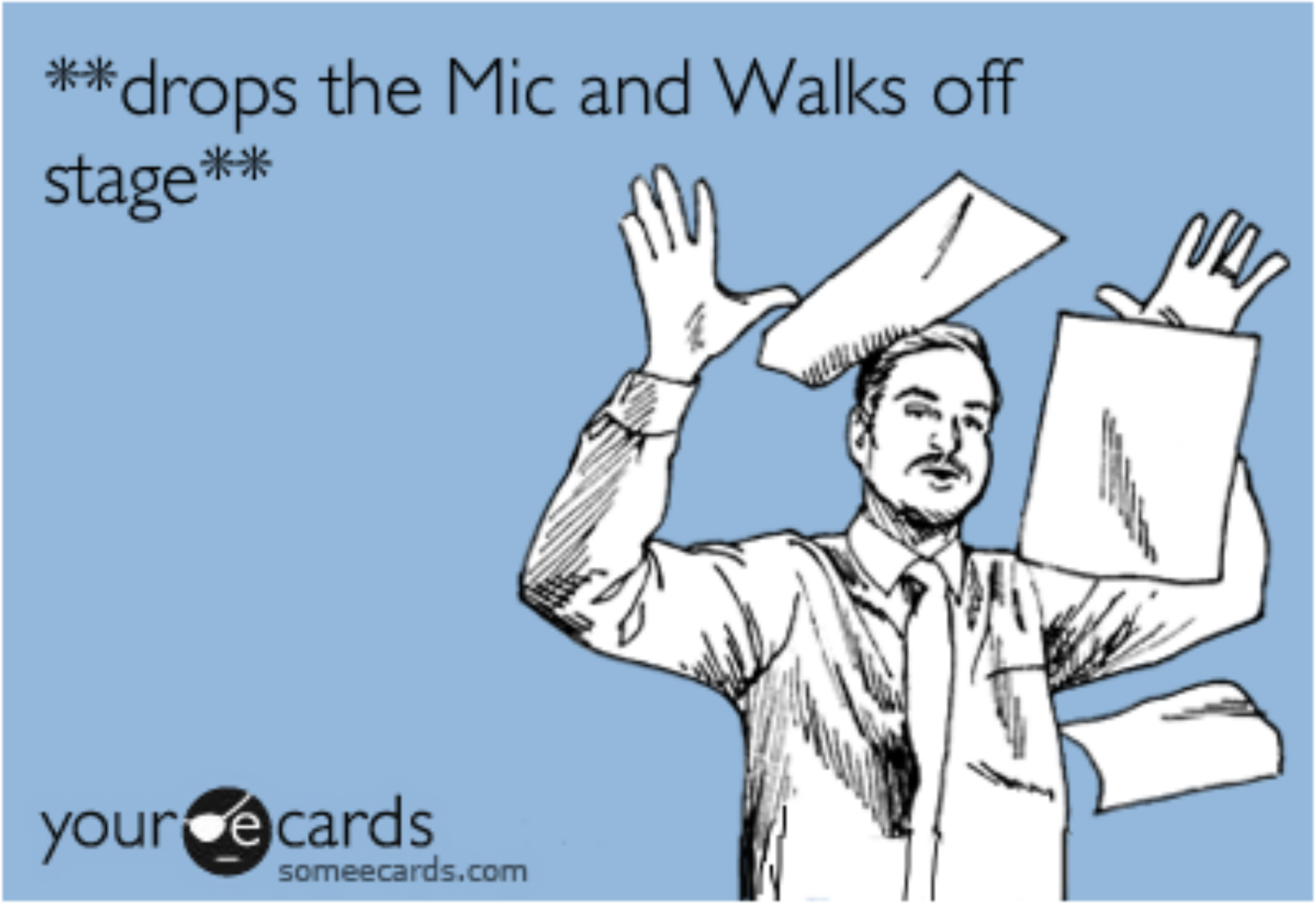

Sometimes..
It just works.
and its because of you
Create something at LiveLuvCreate.com

Detection

- ActiveX objects for heapspray / payload
- EMET

“In my best declarative voice I will now say something unsubstantiated and crazy. There will be a demo.”

–mjr


drops the Mic and Walks off stage
yourecards
someecards.com