ABUSING WEB APIS THROUGH SCRIPTED ANDROID APPLICATIONS
DANIEL PECK

- Barracuda Labs Principle Researcher
- Studying Malicious Messaging (Email/Social Networks/etc)
- Data/Trend Analysis in Security

@ramblinpeck

@barracudalabs

Past Lives

- SCADA, Snort Jockey, Reverser (not so past?), Assessment Work
SESSION ROADMAP

- Brief overview of android/dalvik vm
- Reversing an apk
- Disassembly and static analysis
- Dynamic Analysis
- Control/scripting for our own usage

*Do the dumb thing first and build on the work of smarter people.*
Hot social app that I want to be a part of
Great web interface, great api once we have a few hundred thousand accounts, but protected
SOLUTION

- People are too worried about “friction” to put many safeguard/throttling into mobile apps

- Create our own client that mimics mobile app for api purposes.

- Lets target android
ASSUMPTIONS AND HOPES

- Twacebook has a well documented API that's protected using OAuth
- We'll probably need to extract some keys
- They probably use their own API for Android app
BUILD ON EXISTING TOOLS
INTERCEPTING APP COMMUNICATIONS

- Need to MitM to be able to view tx/rx
- Proxydroid
  - [https://github.com/madeye/proxydroid](https://github.com/madeye/proxydroid)
  - Run all/some of android traffic through our proxy
- SSL
  - The developers at Twacebook aren't idiots
• Create and add a cert to your testing device

  • Easy, and writeups all over so won't detail, basics for 2.x devices:

    $ adb pull /system/etc/security/cacerts.bks
    $ keytool ...
    $ adb push cacerts.bks /system/etc/security

  
  • Gotchas

  • Make sure you have the right version of bouncycastle otherwise things break in not-fun ways
  • Different/easier procedures on Android 4.0+ devices
BURP PROXY

- Invisible proxying, generates cert on demand, but you have to provide hostname
- Look at dns requests/guess hostnames to tell burp to use for generated certs
- Done automatically in 1.4.12 release

http://releases.portswigger.net/2012/08/v1412.html
(http://releases.portswigger.net/2012/08/v1412.html)
INTERCEPTED TRAFFIC

POST /create_account HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: 296
Accept-Encoding: gzip, deflate
User-Agent: TwacebookAndroidApp(build 6294, v1.8.64)
Host: mobileapi.twacebook.com
Connection: Keep-Alive
Cache-Control: no-cache

auth_consumer_key=40iq0gCcXqfwq0a02D7nQ
oauth_nonce=0437A32D733151CABA3A06A12243CD0A
oauth_signature_method=HMAC-SHA1
oauth_timestamp=1340141019
oauth_version=1.0
x_auth_mode=client_auth
x_auth_password=f00bar%24
x_auth_username=jimbo
oauth_signature=v%2FVnCJrssg9D07Zdy%2F8dPSapv8s%3D
OAUTH

- Consumers requests a consumer key and consumer secret from provider
- End users allow provider to grant a token and token secret to consumer to make requests on their behalf
- Signs requests (HMAC-SHA1 usually) with consumer secret & token secret
MORE OAUTH

- Users don't have to give their password to third party apps

  That's good

- Providers get to restrict apps accessing their api to only (honest) approved ones, essentially DRM

  That's bad

- Designed and works well for server ← → server

  That's good
• Used extensively for mobile/desktop apps

Thats just everyone fooling themselves

- Decodes apks
- Nice wrapper for smali/baksmali
- In theory should allow for some nice debugging..

JD-GUI [http://java.decompiler.free.fr/?q=jdgui](http://java.decompiler.free.fr/?q=jdgui)

- dex2jar first
- not compilable source, sometimes misleading, good for general idea
ABOUT ANDROID

Runs within a Dalvik application virtual machine
DALVIK

- Register based machine
- Optimized for low memory environments
- Runs dex files
  - Deduped
  - Dalvik instruction set instead of standard JVM
- Smali bytecode
class public final Lcd;
super Ljava/lang/Object;
# static fields
.field public static final a:Lcd;

.method constructor
init
()V
.locals 2
const/4 v1, 0x0
const/4 v0, 0x0
invoke-direct {p0, v1, v0, v1}, Lcd;-
init
(Laa;ILjava/lang/String;)V
return-void
.end method
DECIPHERING SMALI

- Register based machine
  - Parameters are stored in p0...pX
  - Local registers v0...vY where
  - Last X local registers are identical to parameter registers
- Registers store 32-bit values
  - 64-bit values (J, long, and D, double primitives) are stored in 2 registers
PRIMITIVES

void - can only be used for return types

boolean
byte
short
char
int
Jlong (64 bits)

Ffloat

Ddouble (64 bits)

Lobjects. You'll see in the form of “Lpackage/name/ObjectName”
FUNCTION DECLARATIONS

method private static a
( Lorg/apache/http/client/methods/HttpBase;
La;
J
Ljava/lang/String;
Ljava/lang/String;
)Ljava/lang/String;
FUNCTION DECLARATIONS

method private static a #name and type
(
Lorg/apache/http/client/methods/HttpRequestBase; #p0
Laa; #p1
J #p2 + #p3
Ljava/lang/String; #p4
Ljava/lang/String; #p5
)Ljava/lang/String; #return type
OPCOCDES

move-result vx

return-object vx

invoke-direct parameters , methodtocallo

invoke-static parameters , methodtocallo

...
Many more, great reference:

http://pallergabor.uw.hu/androidblog/dalvik_opcodes.html

(http://pallergabor.uw.hu/androidblog/dalvik_opcodes.html)
const-string p1, "OAuth realm="%s",
oauth_version="%s", oauth_nonce="%s",
oauth_timestamp="%s", oauth_signature="%s",
oauth_consumer_key="%s", oauth_signature_method="%s"
new-array p3, p3, [Ljava/lang/Object;
...
const/4 p2, 0x4
aput-object p0, p3, p2
const/4 p0, 0x5
aput-object p4, p3, p0
...
invoke-static {p1, p3}, Ljava/lang/String;->format(Ljava/lang/String;
[Ljava/lang/Object;)Ljava/lang/String;
move-result-object p0
const-string p1, "OAuth realm="%s",
oauth_version="%s", oauth_nonce="%s",
oauth_timestamp="%s", oauth_signature="%s",
oauth_consumer_key="%s", oauth_signature_method="%s"
new-array p3, p3, [Ljava/lang/Object; #create array
...
const/4 p2, 0x4
aput-object p0, p3, p2 #filling array
const/4 p0, 0x5
aput-object p4, p3, p0
...
invoke-static {p1, p3}, Ljava/lang/String;->format(Ljava/lang/String;
[Ljava/lang/Object;)Ljava/lang/String; #filling in string
move-result-object p0
invoke-static {p0, p5, v0}, Lcd;-> a(Ljava/lang/String;
Ljava/lang/String;
Ljava/lang/String;)Ljava/lang/String;
move-result-object p0
invoke-virtual {v0, v1}, Ljava/lang/String;->getBytes(Ljava/lang/String;)Ljava/lang/String;[B
    move-result-object v0
    new-instance v1, Ljavax/crypto/spec/SecretKeySpec;
    const-string v2, "HmacSHA1"
    invoke-direct {v1, v0, v2},
    Ljavax/crypto/spec/SecretKeySpec;-><init>([BLjava/lang/String;])V
    invoke-static {v0}, Ljavax/crypto/Mac;->getInstance(Ljava/lang/String;)Ljavax/crypto/Mac;
...
    invoke-virtual {v0, v1}, Ljavax/crypto/Mac;->init(Ljava/security/Key;)V
    const-string v1, "UTF8"
    invoke-virtual {p0, v1}, Ljava/lang/String;->getBytes(Ljava/lang/String;)Ljava/lang/String;[B
    move-result-object v1
    invoke-virtual {v0, v1}, Ljavax/crypto/Mac;->
>doFinal([B][B
    move-result-object v0
private static String a(String paramString1, String paramString2, String paramString3)
{
    if (paramString3 == null);
    while (true)
    {
        try
        {
            str1 = "";
            SecretKeySpec localSecretKeySpec = new SecretKeySpec((ch.a(paramString2) + "&" + ch.a(str1)).getBytes("UTF8"), "HmacSHA1");
            Mac localMac = Mac.getInstance("HmacSHA1");
            localMac.init(localSecretKeySpec);
            String str3 = ch.a(new String(cc.a(localMac.doFinal(paramString1.getBytes("UTF8")))),}
"UTF8"));
    str2 = str3;
    return str2;
}
catch (InvalidKeyException localInvalidKeyException)
{
    str2 = "";
    continue;
}
catch (NoSuchAlgorithmException localNoSuchAlgorithmException)
{
    str2 = "";
    continue;
}
catch (UnsupportedEncodingException localUnsupportedEncodingException)
{
    String str2 = "";
    continue;
}
String str1 = paramString3;
public String computeHmac(String baseString, String key)
   throws NoSuchAlgorithmException, InvalidKeyException, 
   IllegalStateException, 
   UnsupportedEncodingException
{
    Mac mac = Mac.getInstance("HmacSHA1");
    SecretKeySpec secret = new SecretKeySpec(key.getBytes(), mac.getAlgorithm());
    mac.init(secret);
    byte[] digest = mac.doFinal(baseString.getBytes());
    return Base64.encode(digest);
}
AGAIN, DUMB THING FIRST

Printf debugging

const-string v2, "SECRETKEY , v0"
invoke-static {v2, v0}, Landroid/util/Log;
>d(Ljava/lang/String;Ljava/lang/String;)
invoke-virtual {v0, v1}, Ljava/lang/String;
>getBytes(Ljava/lang/String;)[B
move-result-object v0
new-instance v1, Ljavax/crypto/spec/SecretKeySpec;
const-string v2, "HmacSHA1"
invoke-direct {v1, v0, v2},
Ljavax/crypto/spec/SecretKeySpec;-init
  ([BLjava/lang/String;)V
Rebuild the apk and run it

$ apktool b twacebook.apk twacebook_new.apk
EXAMING THE LOGS

$ adb shell
$ adb logcat

"SECRETKEY, v0 - I7PW5lgEkgMrqP0dxIj1o6llAbFdXHhVjFnvUsg1g"
SUCESS?
ERROR, INVALID SIGNATURE

Sadness → Confusion → Realization

Twacebook devs have been especially sneaky, passing the returned signature to another method

Custom hash/encoding? No clue but its ugly
.method public final a([BIILjava/io/OutputStream;])I
  .locals 9

  const/4 v8, 0x0

  rem-int/lit8 v0, p3, 0x3

  sub-int v1, p3, v0

  move v2, v8

  :goto_0
  add-int/lit8 v3, v1, 0x0

  if-ge v2, v3, :cond_0

  aget-byte v3, p1, v2
and-int/lit16 v3, v3, 0xff
add-int/lit8 v4, v2, 0x1
aget-byte v4, p1, v4
and-int/lit16 v4, v4, 0xff
add-int/lit8 v5, v2, 0x2
aget-byte v5, p1, v5
and-int/lit16 v5, v5, 0xff
iget-object v6, p0, Ll;->a:[B
ushr-int/lit8 v7, v3, 0x2
and-int/lit8 v7, v7, 0x3f
aget-byte v6, v6, v7
invoke-virtual {p4, v6}, Ljava/io/OutputStream;->write(I)V
iget-object v6, p0, Ll;->a:[B
shl-int/lit8 v3, v3, 0x4
ushr-int/lit8 v7, v4, 0x4
or-int/2addr v3, v7
and-int/lit8 v3, v3, 0x3f
aget-byte v3, v6, v3
invoke-virtual {p4, v3}, Ljava/io/OutputStream;->write(I)V
iget-object v3, p0, Ll;->a:[B
shl-int/lit8 v4, v4, 0x2
ushr-int/lit8 v6, v5, 0x6
or-int/2addr v4, v6
and-int/lit8 v4, v4, 0x3f
aget-byte v3, v3, v4
invoke-virtual {p4, v3},Ljava/io/OutputStream;->write(I)V
iget-object v3, p0, Ll;->a:[B
and-int/lit8 v4, v5, 0x3f
aget-byte v3, v3, v4
invoke-virtual {p4, v3},Ljava/io/OutputStream;->write(I)V
add-int/lit8 v2, v2, 0x3
goto :goto_0
:cond_0
packed-switch v0, :pswitch_data_0
:goto_1
:pswitch_0
div-int/lit8 v1, v1, 0x3

mul-int/lit8 v1, v1, 0x4

if-nez v0, :cond_1

move v0, v8

:goto_2
add-int/2addr v0, v1

return v0

:pswitch_1
add-int/lit8 v2, v1, 0x0

aget-byte v2, p1, v2

and-int/lit16 v2, v2, 0xff
ushr-int/lit8 v3, v2, 0x2
and-int/lit8 v3, v3, 0x3f
shl-int/lit8 v2, v2, 0x4
and-int/lit8 v2, v2, 0x3f
iget-object v4, p0, Ll;->a:[B
aget-byte v3, v4, v3
invoke-virtual {p4, v3}, Ljava/io/OutputStream;->write(I)V
iget-object v3, p0, Ll;->a:[B
aget-byte v2, v3, v2
invoke-virtual {p4, v2}, Ljava/io/OutputStream;->write(I)V
iget-byte v2, p0, Ll;->b:B
invoke-virtual {p4, v2}, Ljava/io/OutputStream;->write(I)V

iget-byte v2, p0, Ll;->b:B

invoke-virtual {p4, v2}, Ljava/io/OutputStream;->write(I)V

goto :goto_1

:pswitch_2
add-int/lit8 v2, v1, 0x0

aget-byte v2, p1, v2

and-int/lit16 v2, v2, 0xff

add-int/lit8 v3, v1, 0x0

add-int/lit8 v3, v3, 0x1

aget-byte v3, p1, v3
and-int/lit16 v3, v3, 0xff
ushr-int/lit8 v4, v2, 0x2
and-int/lit8 v4, v4, 0x3f
shl-int/lit8 v2, v2, 0x4
ushr-int/lit8 v5, v3, 0x4
or-int/2addr v2, v5
and-int/lit8 v2, v2, 0x3f
shl-int/lit8 v3, v3, 0x2
and-int/lit8 v3, v3, 0x3f
iget-object v5, p0, Ll;->a:[B
aget-byte v4, v5, v4
invoke-virtual {p4, v4}, Ljava/io/OutputStream;
>write(I)V

iget-object v4, p0, Ll;->a:[B

aget-byte v2, v4, v2

invoke-virtual {p4, v2}, Ljava/io/OutputStream;
>write(I)V

iget-object v2, p0, Ll;->a:[B

aget-byte v2, v2, v3

invoke-virtual {p4, v2}, Ljava/io/OutputStream;
>write(I)V

iget-byte v2, p0, Ll;->b:B

invoke-virtual {p4, v2}, Ljava/io/OutputStream;
>write(I)V

goto :goto_1
:cond_1
const/4 v0, 0x4
goto :goto_2

:pswitch_data_0
.packed-switch 0x0
  :pswitch_0
  :pswitch_1
  :pswitch_2
.end packed-switch
.end method
public final int a(byte[] paramArrayOfByte, int paramInt1, int paramInt2, OutputStream paramOutputStream) {
    int i = paramInt2 % 3;
    int j = paramInt2 - i;
    for (int k = 0; k < j + 0; k += 3) {
        int i9 = 0xFF & paramArrayOfByte[k];
        int i10 = 0xFF & paramArrayOfByte[(k + 1)];
        int i11 = 0xFF & paramArrayOfByte[(k + 2)];
        paramOutputStream.write(this.a[(0x3F & i9 >>> 2)]);
        paramOutputStream.write(this.a[(0x3F & (i9 << 4 | i10 >>> 4))]);
        paramOutputStream.write(this.a[(0x3F & (i10 << 2 | i11 >>> 6))]);
        paramOutputStream.write(this.a[(i11 & 0x3F)]);
    }
}
int i4;
switch (i)
{
case 0:    
default:
    i4 = 4 * (j / 3);
    if (i != 0)
       break;
case 1:    
case 2:
}
for (int i5 = 0; ; i5 = 4)
{
    return i5 + i4;
    int i6 = 0xFF & paramArrayOfByte[(j + 0)];
    int i7 = 0x3F & i6 >> 2;
    int i8 = 0x3F & i6 << 4;
    paramOutputStream.write(this.a[i7]);
    paramOutputStream.write(this.a[i8]);
    paramOutputStream.write(this.b);
    paramOutputStream.write(this.b);
    break;
int m = 0xFF & paramArrayOfByte[(j + 0)];
int n = 0xFF & paramArrayOfByte[(1 + (j + 0))];
int i1 = 0x3F & m >> 2;
int i2 = 0x3F & (m << 4 | n >> 4);
int i3 = 0x3F & n << 2;
paramOutputStream.write(this.a[i1]);
paramOutputStream.write(this.a[i2]);
paramOutputStream.write(this.a[i3]);
paramOutputStream.write(this.b);
break;
}
BUT WAIT, JRUBY?

Ruby interpreter implemented in Java

Allows calling java functions/libraries from ruby

And thankfully, dex are just another kind of jar

$ unzip twacebook.apk
$ d2j-dex2jar.sh classes.dex -o twacebook.jar
require 'java'
require './jars/twacebook.jar'
require './jars/android.jar'

java_import 'cc' do | clasname |
  "Obfuscater"
end

obs_arr = Obfuscater.a(byte_arr)
signature = String.from_java_bytes(obs_arr)
ITERATING UP

```
require 'java'
require './jars/twacebook.jar'
require './jars/android.jar'

java_import 'ab' do |clasname|
  "User"
end

java_import 'cc' do |clasname|
  "ApiFactory"
end

social_bot = ApiFactory.register_new_user(<name>, <email>)

social_bot.post_update("Posting from a JRUBY")
```
BUT HOW TO GET REALISTIC SOCIAL BOTS?

Stereotyping
BUILD ON OPEN DATA SOURCES

- US Census data
  - Last Name -> Ethnicity Mapping
- Facebook Data Dump circa 2010
  - Profile links -> pictures
  - Names to mix and match
- Mash up with scripts
REALISTIC INTERESTS

Pick a random sample of suggested users to follow from the services.

Get "interest" areas from there.

Services give you the corpus of for your own filtering.
EARNING REPORT OF SELLING FAKE FOLLOWERS BUSINESS

Forget malware distributing and spam

20k Followers sell for $30-$80
A few thousand puppet accounts closer to an advertorial social world...
EXPANDING

Opens up reuse of APK code for scripting

Testing frameworks for android apps in ruby?

Great for dynamic analysis during reversing, easily test assumptions with rapid smali->build->run.

And of course bypassing anything you don't want to deal with...

Almost certainly some bugs/inconsistencies. Find them. Have fun
Thank you Blackhat USA

Your ideas, thoughts and questions

peck@barracuda.com (mailto:dpeck@barracuda.com)
peck@danielpeck.com (mailto:peck@danielpeck.com)

@ramblinpeck

@barracudalabs