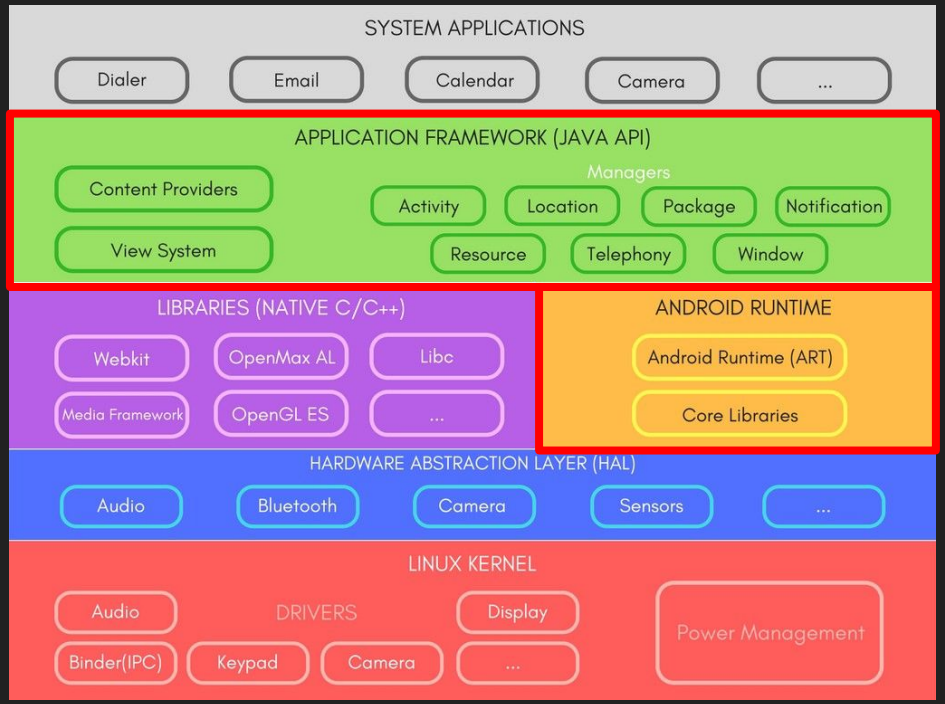
Introduction to Mobile Security

Yue Duan

Introduction to Android

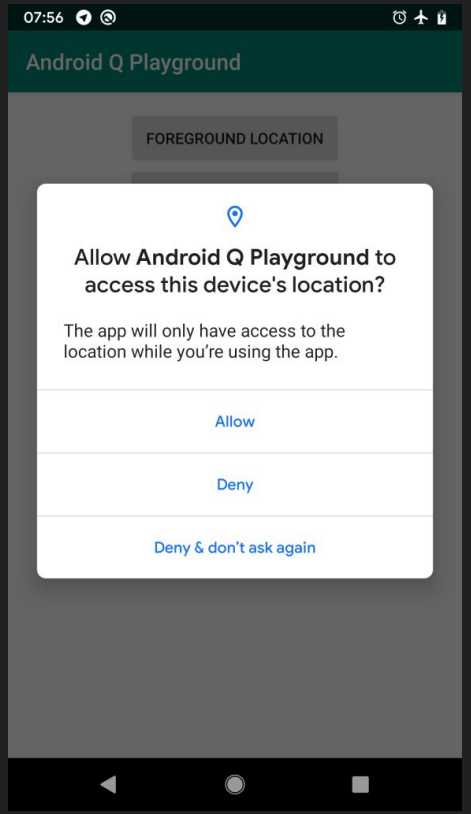
- Kernel layer
  - Linux
  - Binder for IPC
  - Low-level resource management
- Middleware layer
  - Unique to Android
  - Framework
    - high-level resource management
  - Runtime
    - Dalvik Virtual Machine
    - Android Runtime (ART)
- Application layer
  - System applications
  - User applications



Introduction to Android

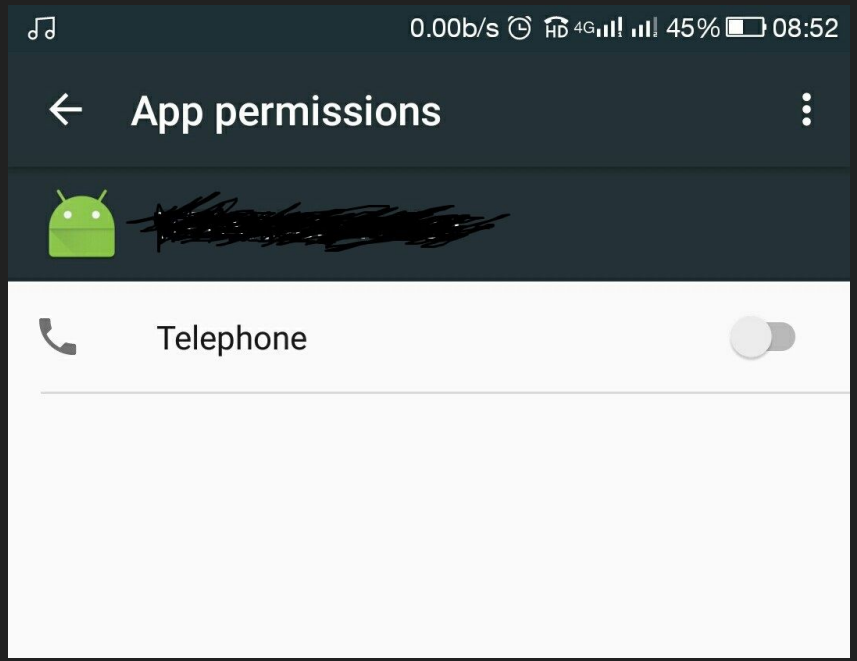
- Android framework
  - Resources management such as location, telephony, etc
  - Largely in Java
  - Access control enforcement: permissions

```
@RequiresPermission(anyOf = {ACCESS_COARSE_LOCATION, ACCESS_FINE_LOCATION})
@Nullable
public Location getLastKnownLocation(@NonNull String provider) {
    checkProvider(provider);
    String packageName = mContext.getPackageName();
    LocationRequest request = LocationRequest.createFromDeprecatedProvider(
            provider, 0, 0, true);

    try {
        return mService.getLastLocation(request, packageName);
    } catch (RemoteException e) {
        throw e.rethrowFromSystemServer();
    }
}
```



Introduction to Android

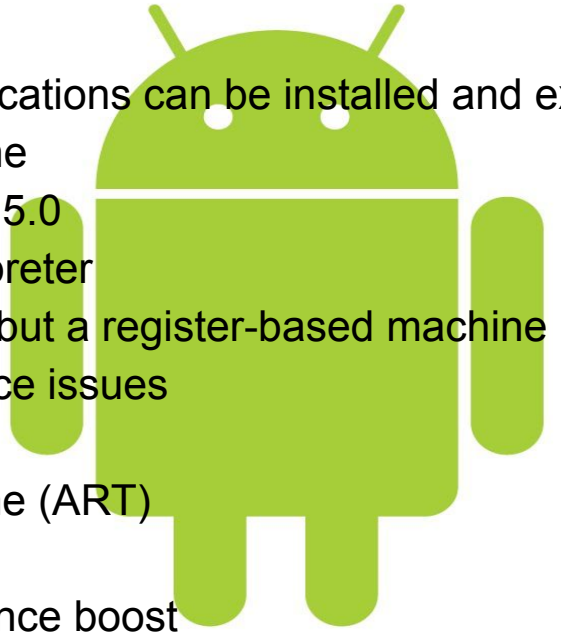
- Android framework



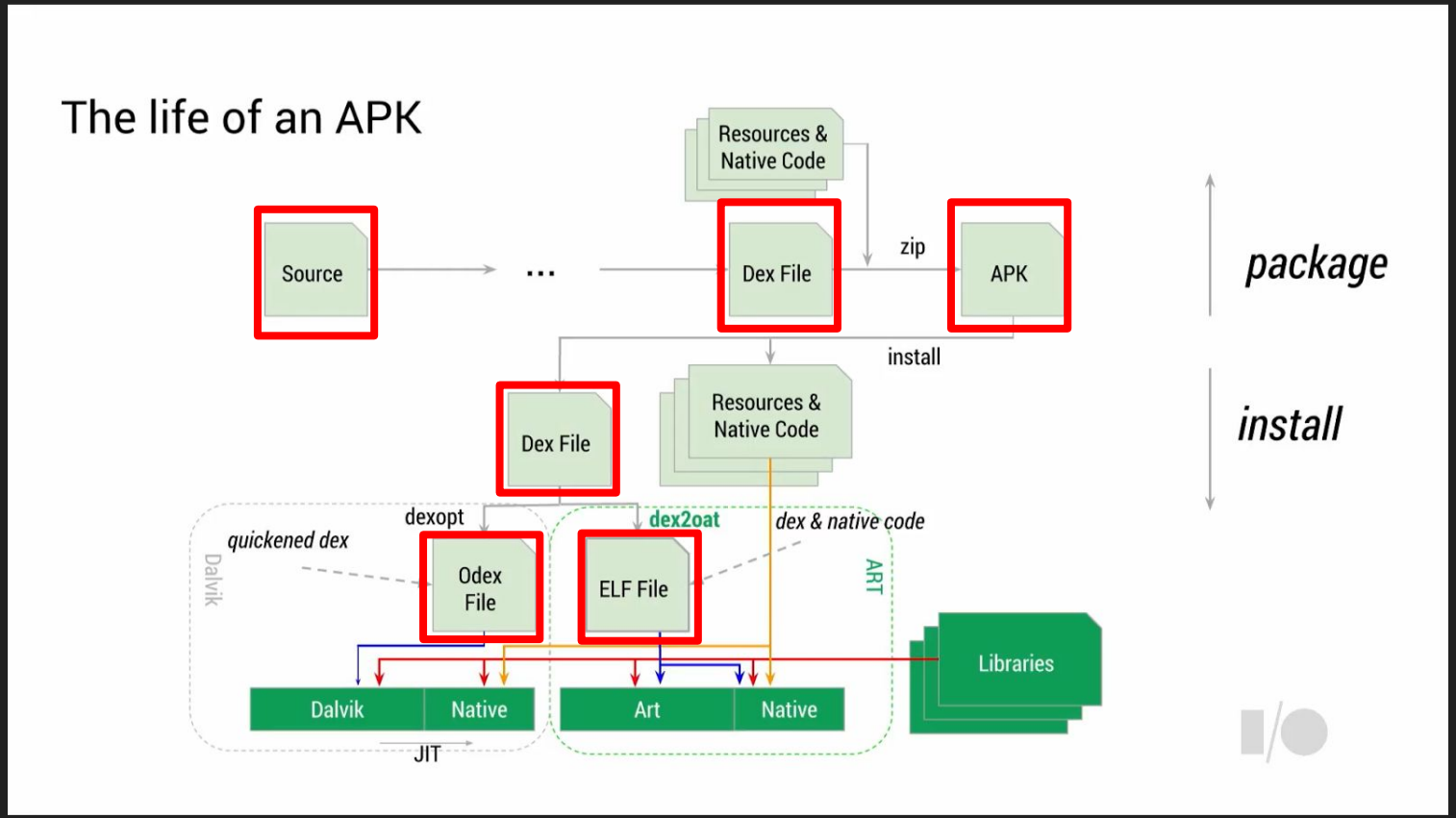

```
/** @hide */
@SystemApi
@RequiresPermission(android.Manifest.permission.MODIFY_PHONE_STATE)
public boolean disableDataConnectivity() {
    try {
        ITelephony telephony = getITelephony();
        if (telephony != null)
            return telephony.disableDataConnectivity();
    } catch (RemoteException e) {
        Log.e(TAG, "Error calling ITelephony#disableDataConnectivity", e);
    }
    return false;
}
```

Introduction to Android

- Runtime
  - Where Android applications can be installed and executed
  - Dalvik virtual machine
    - Before Android 5.0
    - Bytecode interpreter
    - Similar to JVM but a register-based machine
    - Has performance issues
  - After Android 5.0
    - Android Runtime (ART)
    - Native code
    - Huge performance boost

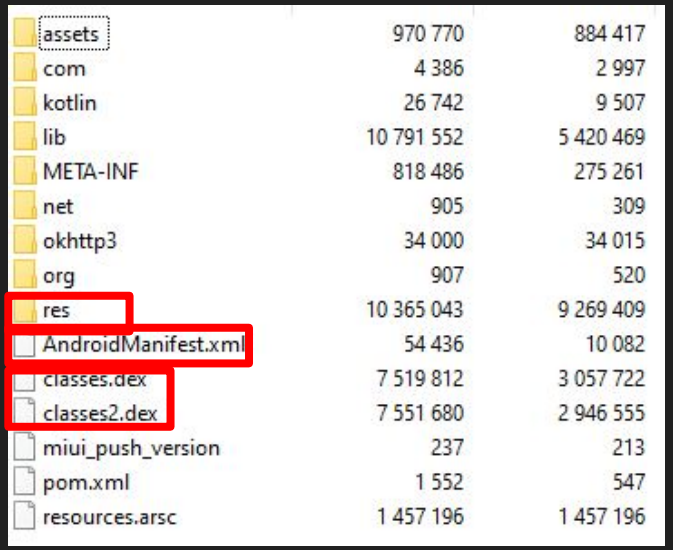


Introduction to Android



Introduction to Android

- Android applications
  - Resource files
  - Dex files
  - Android Manifest file



assets	970 770	884 417
com	4 386	2 997
kotlin	26 742	9 507
lib	10 791 552	5 420 469
META-INF	818 486	275 261
net	905	309
okhttp3	34 000	34 015
org	907	520
res	10 365 043	9 269 409
AndroidManifest.xml	54 436	10 082
classes.dex	7 519 812	3 057 722
classes2.dex	7 551 680	2 946 555
miui_push_version	237	213
pom.xml	1 552	547
resources.arsc	1 457 196	1 457 196

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.homeandlearn.ken.twoactivities">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="TwoActivities"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".SecondActivity"></activity>
    </application>

</manifest>
```

Introduction to Android

- Android applications
  - Can be written in Java or C++ (most likely in Java)
  - Java compiled into Dex (Dalvik Executable) bytecode

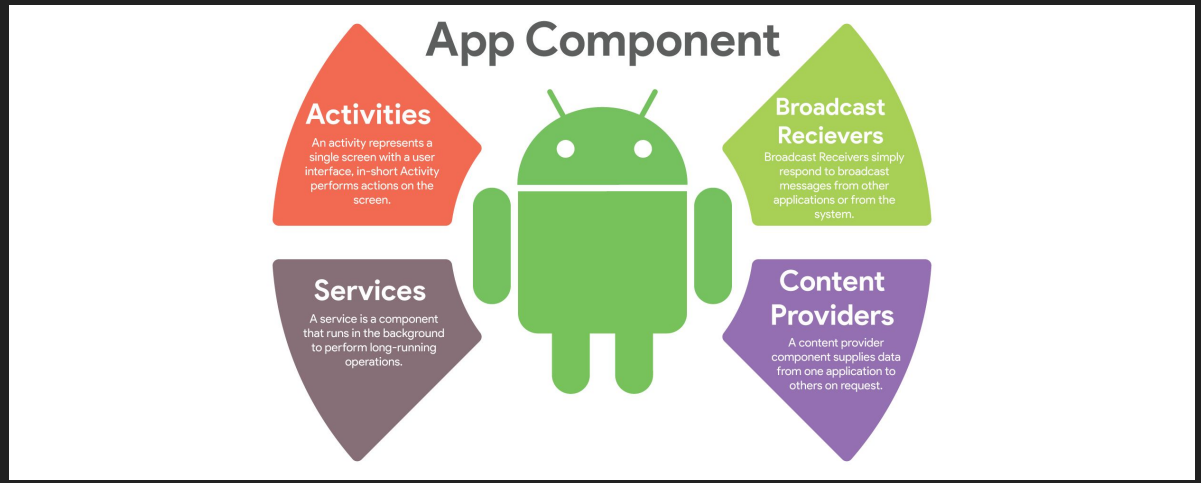
```
public boolean offer(E e) {
        checkNotNull(e);
        final ReentrantLock lock = this.lock;
        lock.lock();
        try {
            if (count == items.length)
                return false;
            else {
                enqueue(e);
                return true;
            }
        } finally {
            lock.unlock();
        }
    }
```

```
.METHOD offer : boolean
        .PARAM java.lang.Object
.MODIFIERS public
.REGISTERS 5
.ANNOTATION dalvik.annotation.Signature
        value=["(TE;)Z"]
.CODE
              1190288  invoke-static {v4}, meth@12229
              1190294  iget-object v0, v3 field@4169
              1190298  invoke-virtual {v0}, meth@14543
                        .TRY #0
              1190304  iget v1, v3 field@4166
              1190308  iget-object v2, v3 field@4167
              1190312  array-length v2, v2
                        .CATCH
                                ALL address:1190344
              1190314  if-ne v1, v2, 7
              1190318  const/4 v1, #0
              1190320  invoke-virtual {v0}, meth@14549
              1190326  return v1
                        .TRY #1
              1190328  invoke-direct {v3, v4}, meth@12236
                        .CATCH
                                ALL address:1190344
              1190334  const/4 v1, #1
              1190336  invoke-virtual {v0}, meth@14549
              1190342  goto -8
              1190344  move-exception v1
              1190346  invoke-virtual {v0}, meth@14549
              1190352  throw v1
```

Introduction to Android

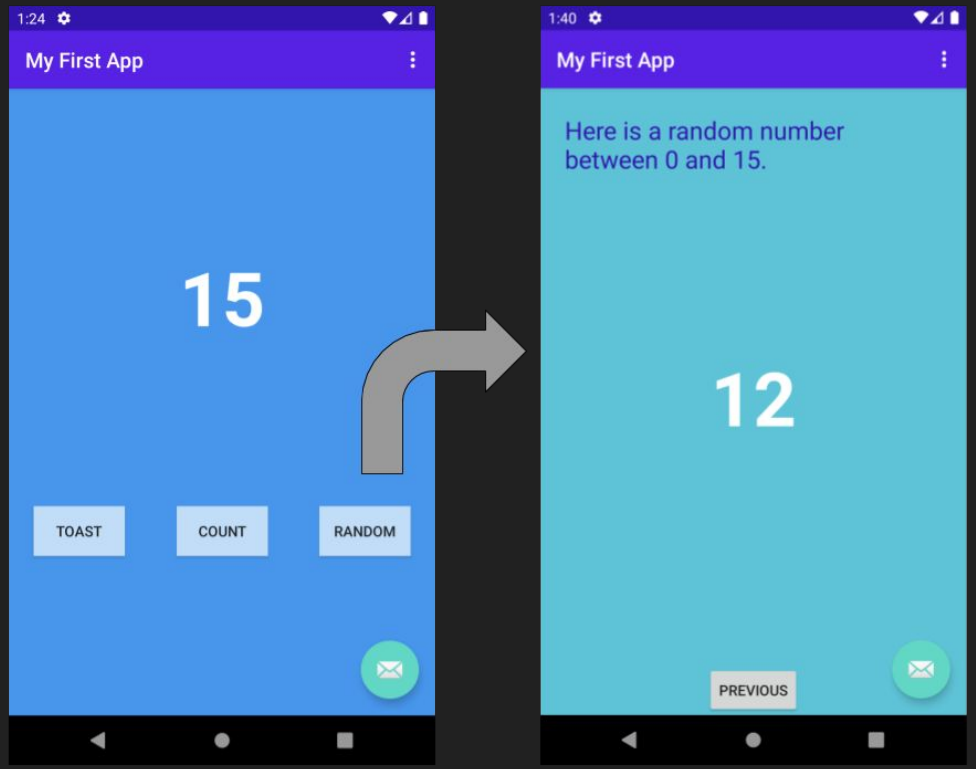
- Android applications
  - Four major components
    - Activity
    - Service
    - Content provider
    - Broadcast receiver

- Intent
  - Used as inter-component signaling
  - Example:
    - start an activity
    - Query a content provider



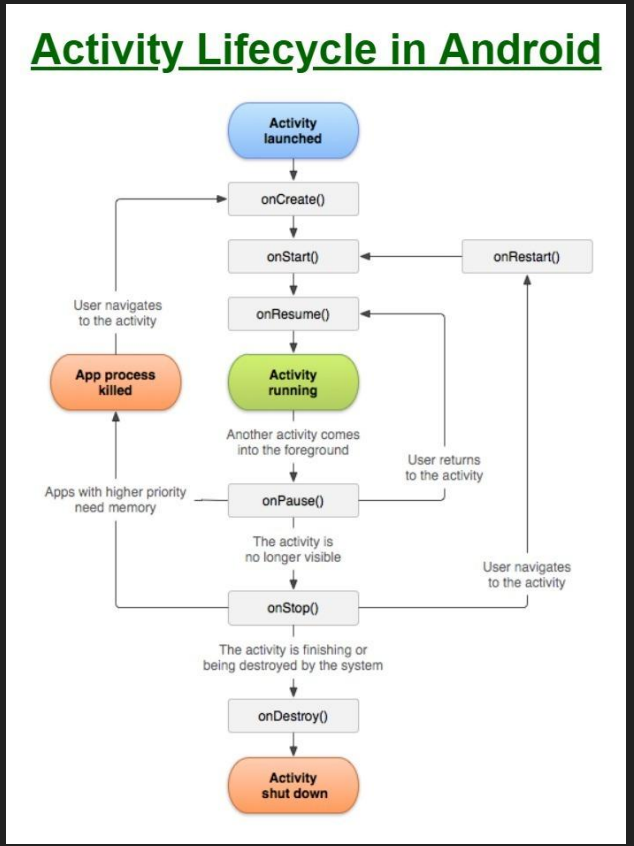
Introduction to Android

- Activity
  - Each activity is a ‘screen’ in app
  - One app can have multiple activities
  - Intent is used to launch an activity
  - Can be invisible/transparent
    - Security consequences!



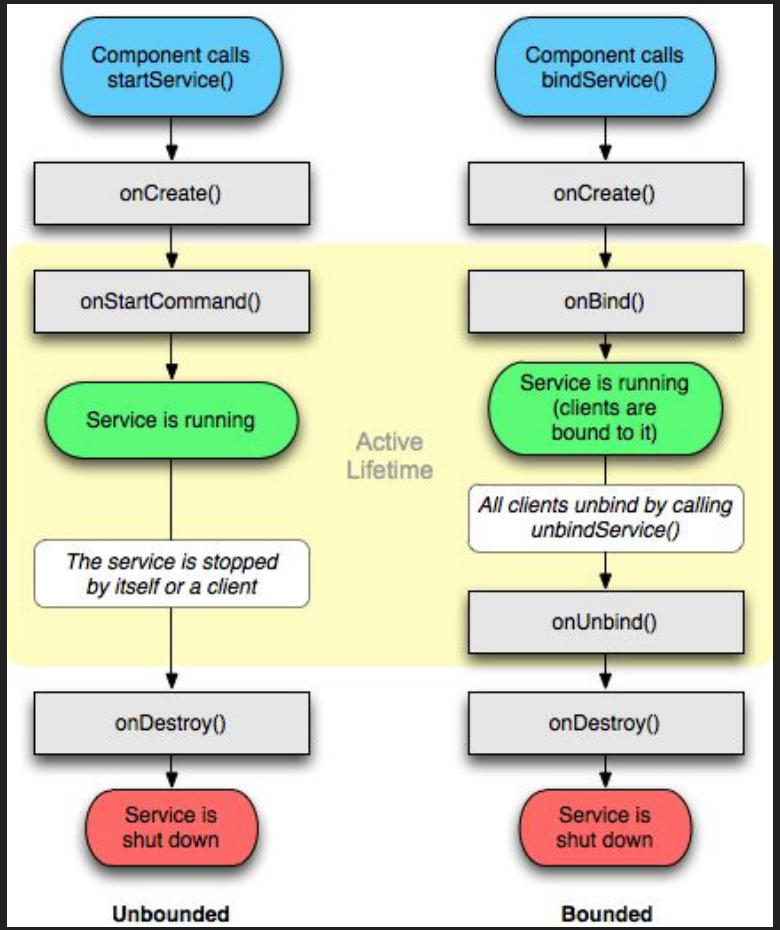
Introduction to Android

- Activity
  - Lifecycle
  - Multiple entry points
  - No explicit control flow within Android apps
  - Make program analysis harder



Introduction to Android

- Service Component
  - Background processing
    - Download a file
    - Play music
  - Multiple interfaces
    - Control: start, stop
    - Method invocation: bind
  - Service lifecycle
    - Similar to activity lifecycle



Introduction to Android

- Permission system
  - Used for access control to sensitive APIs
  - Sensitive APIs:
    - Send text message
    - Retrieve location
    - Access your contacts
    - etc
  - Android apps need to request permissions at installation time

```
<uses-permission android:name="android.permission.INTERNET"></uses-permission>
```

  - Security:
    - Over-privilege issue
    - Hard to understand
    - Repackaged apps

Android Security Analysis

- Android application analysis
  - Vulnerability analysis
    - Component hijacking vulnerability
    - Information leakage
    - Collusion attacks
    - etc
  - Malware analysis
    - packing techniques

- Android framework analysis
  - Inconsistent security policy
  - Implicit control flow transitions

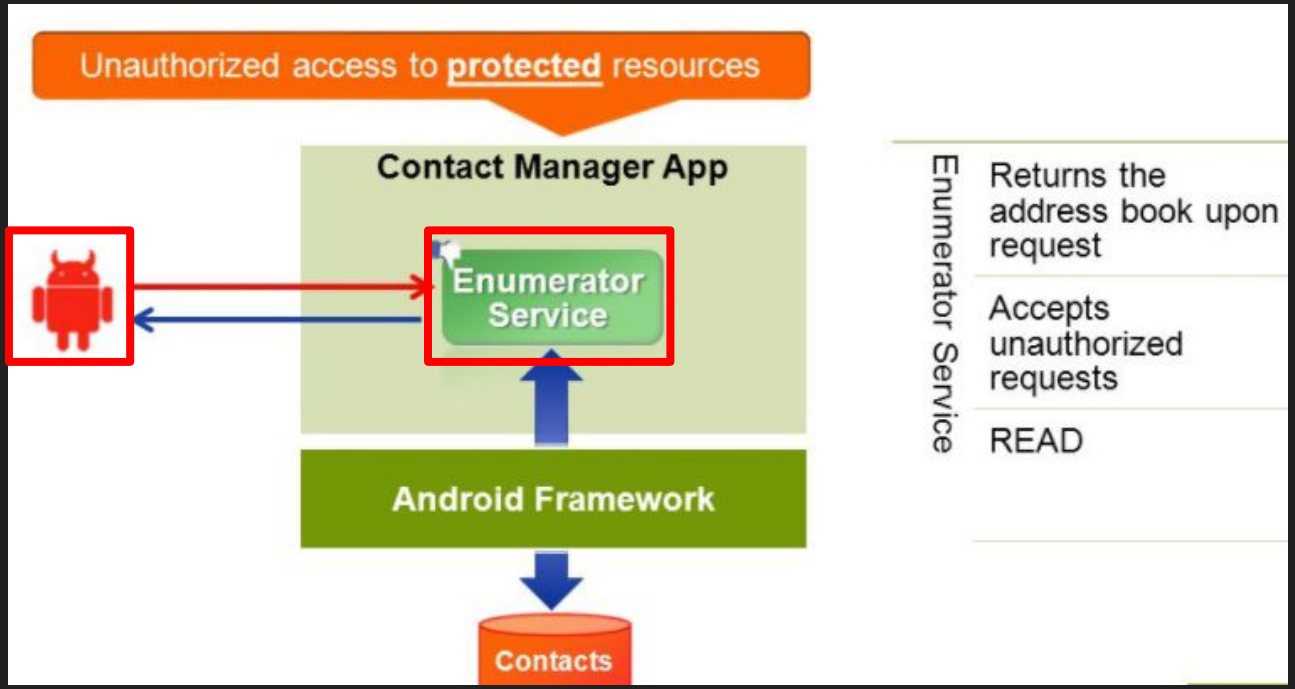
Android Security Analysis

- Component hijacking vulnerability
  - Export components
    - Publicly available
    - Can be launched by other components from a different app
    - Accidentally share permissions

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          xmlns:tools="http://schemas.android.com/tools">

    <application tools:ignore="GoogleAppIndexingWarning">

        <activity
                android:name="com.samples.medium.SecondaryActivity"
                android:exported="true"
                android:screenOrientation="sensorLandscape"/>

    </application>

</manifest>
```

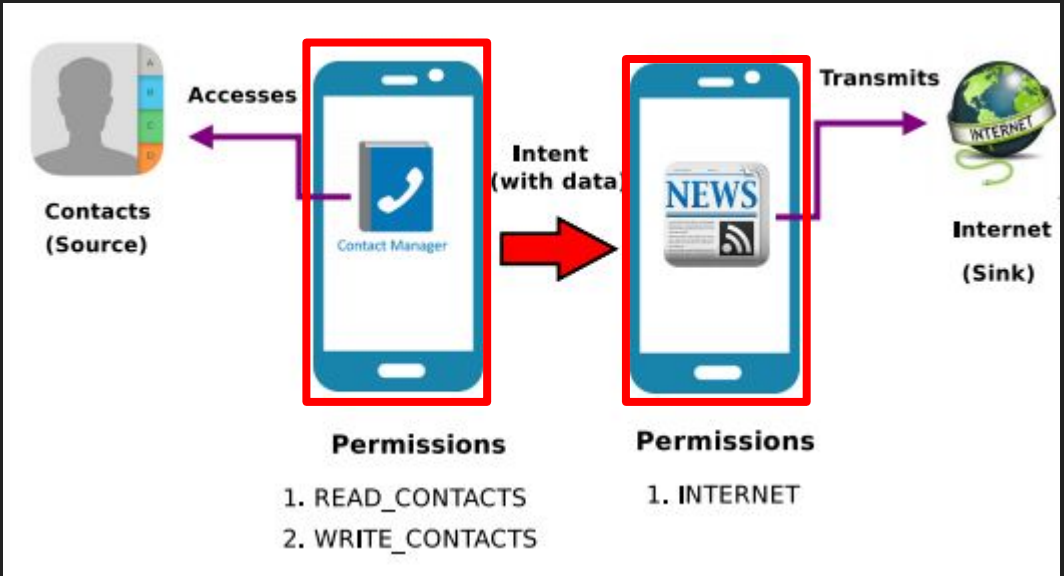
Android Security Analysis

- Component hijacking vulnerability



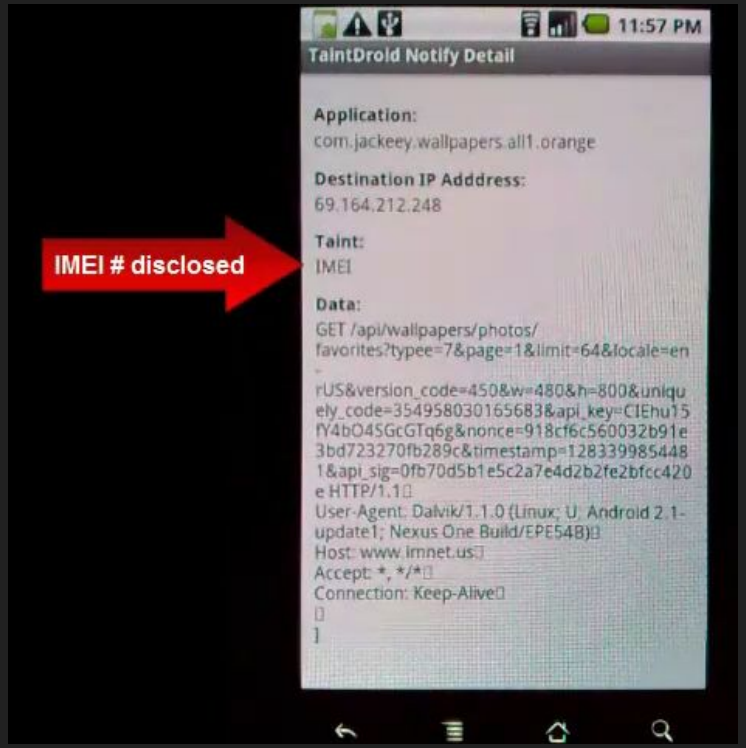
Android Security Analysis

- Collusion attack
  - Multiple apps work together
  - Communicate via intent
  - stealthier

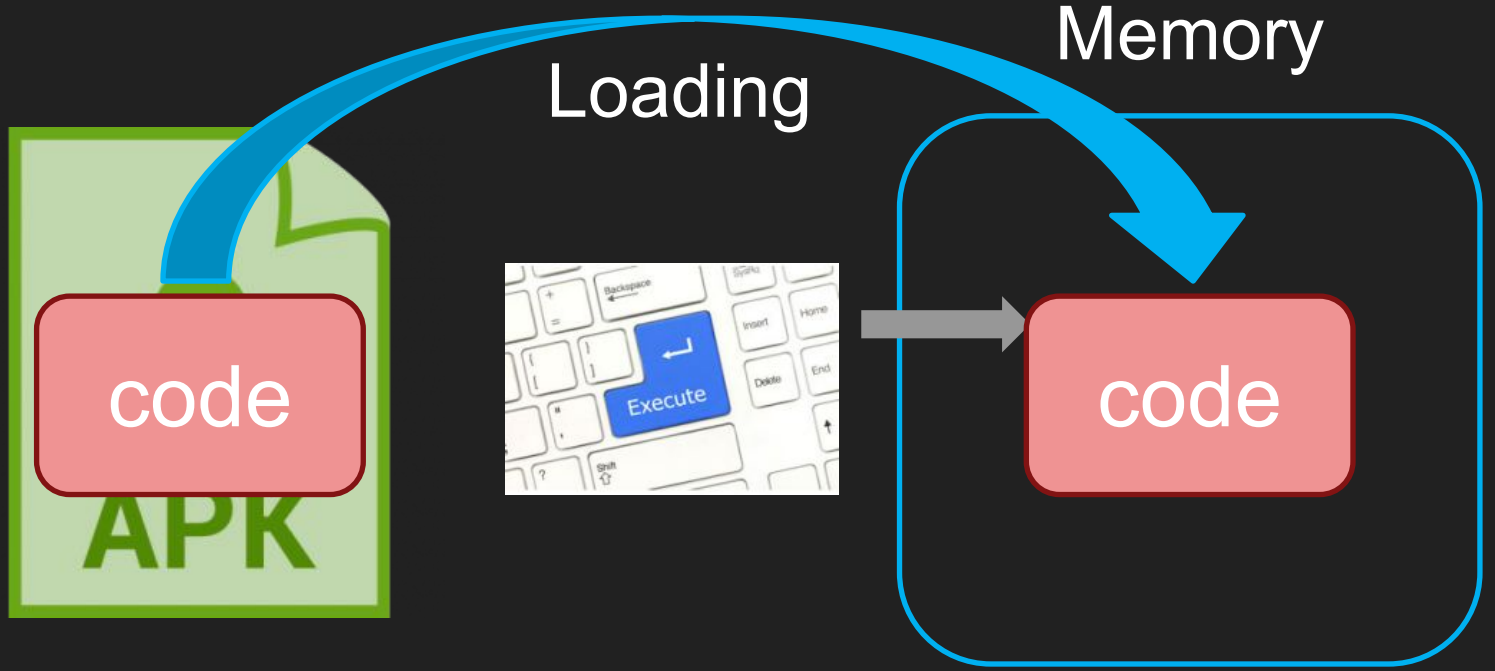


Android Security Analysis

- Information leakage



Packing techniques



Packing techniques

Android App

code

packed App

decrypt.so

encrypted code

Loading

Memory

decrypt.so

encrypted code

decrypt & load

Packing techniques

4.1 MB

```
.
./apktool.yml
./AndroidManifest.xml
./smali
./smali/com
./smali/com/example
./smali/com/example/hellojni
./smali/com/example/hellojni/R$color.smali
./smali/com/example/hellojni/R$layout.smali
./smali/com/example/hellojni/R$string.smali
./smali/com/example/hellojni/HelloJni.smali
./smali/com/example/hellojni/R$dimen.smali
./smali/com/example/hellojni/R$mipmap.smali
./smali/com/example/hellojni/R$integer.smali
./smali/com/example/hellojni/R.smali
./smali/com/example/hellojni/R$style.smali
./smali/com/example/hellojni/R$id.smali
./smali/com/example/hellojni/R$bool.smali
./smali/com/example/hellojni/R$anim.smali
./smali/com/example/hellojni/R$styleable.smali
./smali/com/example/hellojni/R$drawable.smali
./smali/com/example/hellojni/R$attr.smali
./smali/com/example/hellojni/BuildConfig.smali
./original
./original/META-INF
./original/META-INF/ALIAS_NA.SF
./original/META-INF/MANIFEST.MF
./original/META-INF/ALIAS_NA.RSA
./original/AndroidManifest.xml
./lib
./lib/armeabi-v7a
./lib/armeabi-v7a/libhello-jni.so
```

After packing

1 KB

```
.
./apktool.yml
./AndroidManifest.xml
./smali
./smali/com
./smali/com/ali
./smali/com/ali/fixHelper.smali
./smali/com/example
./smali/com/example/hellojni
./smali/com/example/hellojni/R$color.smali
./smali/com/example/hellojni/R$layout.smali
./smali/com/example/hellojni/R$string.smali
./smali/com/example/hellojni/HelloJni.smali
./smali/com/example/hellojni/R$dimen.smali
./smali/com/example/hellojni/R$mipmap.smali
./smali/com/example/hellojni/R$integer.smali
./smali/com/example/hellojni/R.smali
./smali/com/example/hellojni/R$style.smali
./smali/com/example/hellojni/R$id.smali
./smali/com/example/hellojni/R$bool.smali
./smali/com/example/hellojni/R$anim.smali
./smali/com/example/hellojni/R$styleable.smali
./smali/com/example/hellojni/R$drawable.smali
./smali/com/example/hellojni/R$attr.smali
./smali/com/example/hellojni/BuildConfig.smali
./original
./original/META-INF
./original/META-INF/ALIAS_NA.SF
./original/META-INF/MANIFEST.MF
./original/META-INF/ALIAS_NA.RSA
./original/AndroidManifest.xml
./lib
./lib/armeabi-v7a
./lib/armeabi-v7a/libpreverify1.so
./lib/armeabi-v7a/libdemolishdata
./lib/armeabi-v7a/libdemolish.so
./lib/armeabi-v7a/libdemolishdata.so
./lib/armeabi-v7a/libhello-jni.so
```

Framework analysis

- Framework
  - Sensitive resources protection
  - Even experts can make mistakes
  - Severe consequences

```
/**
 * Used by device administration to set the maximum screen off timeout.
 *
 * This method must only be called by the device administration policy manager
 */
@Override // Binder call
public void setMaximumScreenOffTimeoutFromDeviceAdmin(int timeMs) {
    final long ident = Binder.clearCallingIdentity();
    try {
        setMaximumScreenOffTimeoutFromDeviceAdminInternal(timeMs);
    } finally {
        Binder.restoreCallingIdentity(ident);
    }
}
```

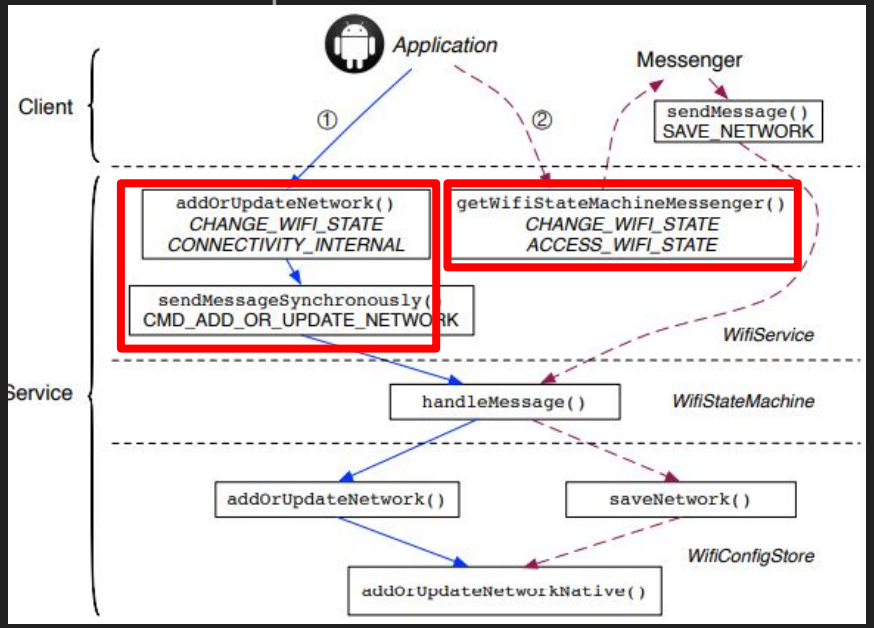
They know the use of this method should be restricted but did not apply any security checks

```
@Override
public boolean havePassword(int userId) throws RemoteException {
    // Do we need a permissions check here?
    return new File(getLockPasswordFilename(userId)).length() > 0;
}

@Override
public boolean havePattern(int userId) throws RemoteException {
    // Do we need a permissions check here?
    return new File(getLockPatternFilename(userId)).length() > 0;
}
```

Android framework developers lack knowledge of security policies that should be enforced

Framework analysis

- Security protection inconsistency
  - An app can use either of the two interfaces to update configs
  - Two interfaces enforce different permissions



Summary

- Understand Android system design
  - Uniqueness of Android
  - Framework
  - Android runtime
- Introduce basics of Android applications
  - Four components
  - Permission system
- Present Android security problems
  - Application vulnerabilities
  - Framework issues
  - Packing techniques

Thank you!!
Questions?