ASC

应用安全联盟

2016 移动物联网安全高峰论坛

换个视角攻击 Android App

About Me

曲和

Alipay unLimit Security Team

- 支付宝钱包终端对抗组的负责人
- 多年安全从业经历
- 主要专注在移动端逆向、漏洞挖掘、防护等

支付宝，知托付

目录

Part. 01

APP安全视角

APP安全检测发展

反编译、人工审计

自动化审计

更多

Framework安全现状

A New Vulnerability in the Android Framework: Fragment Injection

December 10, 2013 | By Roee Hay

We have recently disclosed a new vulnerability to the Android Security Team. The vulnerability affected many apps, including *Settings* (the one that is found on every Android device), *Gmail*, *Google Now*, *DropBox* and *Evernote*. To be more accurate, *any* App which extended the *PreferenceActivity* class using an *exported* activity was automatically vulnerable. A patch has been provided in Android KitKat. If you wondered why your code is now broken, it is due to the Android KitKat patch which requires applications to override the new method, *PreferenceActivity.isValidFragment*, which has been added to the Android Framework.

第三方库安全现状


缺陷编号： WooYun-2015-146617
漏洞标题： 百度系应用安卓版远程代码执行漏洞(百度地图/输入法为例)
相关厂商： 百度
漏洞作者： 路人甲
提交时间： 2015-10-14 10:35
公开时间： 2016-01-12 11:21
漏洞类型： 远程代码执行
危害等级： 高
漏洞状态： 厂商已经确认
漏洞来源： http://www.wooyun.org
Exploit
714)

Part. 02

防不胜防

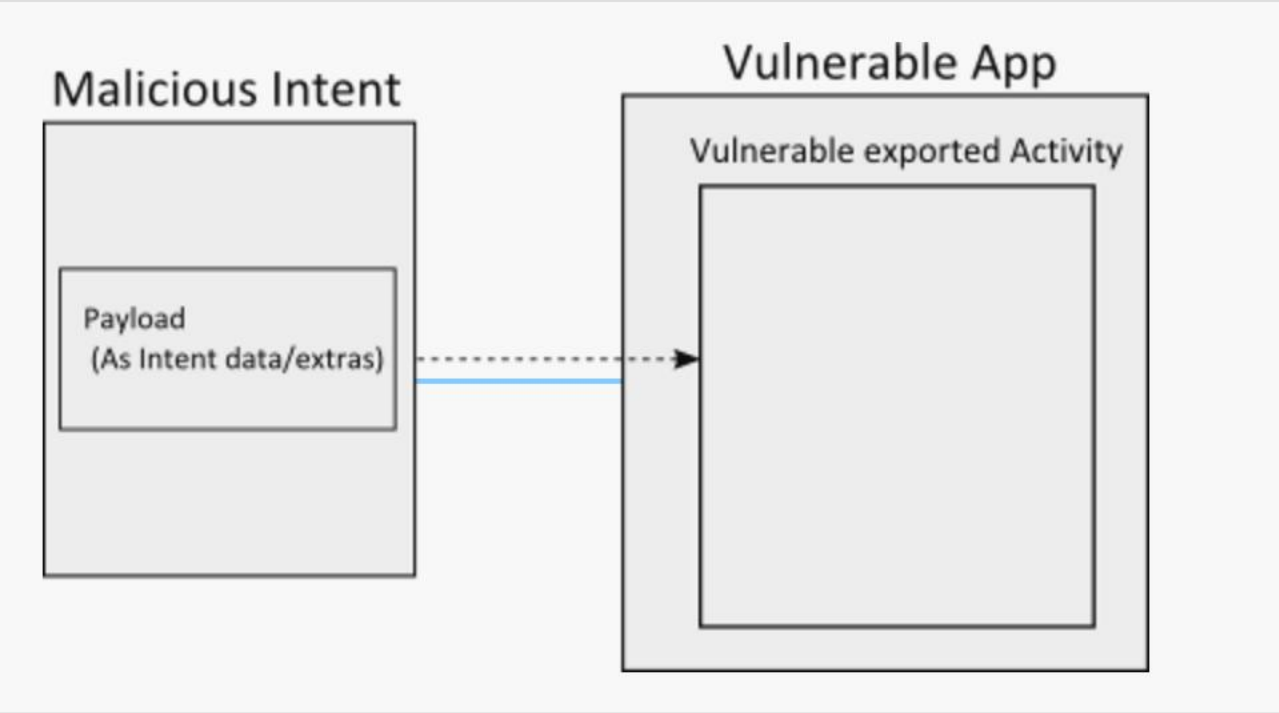
Framework角度攻击

fragment注入

- Activity继承PreferenceActivity类并且被声明成export=true
- 绕过安全限制

Framework角度攻击

fragment注入



攻击Android第三方库

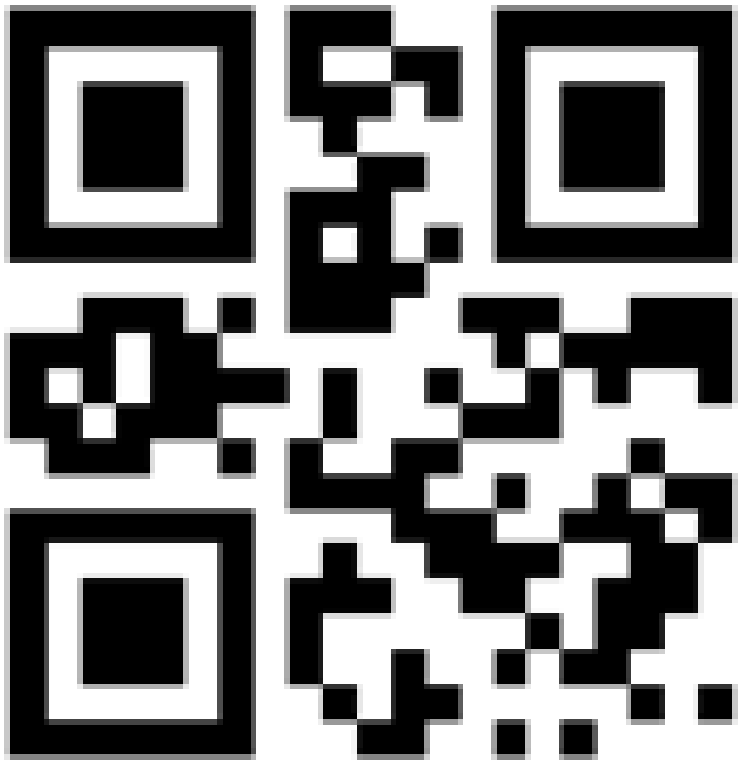
zxing

- 二维码解析库

- 几乎所有App都有扫码功能，攻击范围大

攻击Android第三方库

zxing



攻击Android第三方库

libjhead

- 操作图像文件的exif信息的库

- 社交类软件更易攻击

攻击Android第三方库

libjhead

ARM在处理VLDR指令的时候，地址需要是一个对齐的，jhead最初没考虑到这个，
所以某些图片EXIF头内的float或者double存放在一个不对齐的地址中，就会导致crash

```
native crash happen:SIGBUS
        #00  pc 00002090  /system/lib/libexif.so (ConvertAnyFormat)
        #00  lr 000026d5  /system/lib/libexif.so
java:
java.lang.Thread.getStackTrace(Thread.java:579)
android.media.ExifInterface.getAttributesNative(Native Method)
android.media.ExifInterface.loadAttributes(ExifInterface.java:207)
android.media.ExifInterface.<init>(ExifInterface.java:124)
```

攻击Android第三方库

sqlcipher

- 第三方透明加密数据库组件
- sqlcipher编译时没移除load extension
- sql注入配合load_extension进行漏洞利用

攻击Android第三方库

sqlcipher

```
query("2 or load_extension(\'/data/data/com.testsqliteloadext/lib/libSqliteLoadExtTest.so\')
```

```
public void query(String sql)
{

    try{
    String str =  "select * from person where id=";
    Log.i("testsqliteloadext", str+sql);
    Cursor cursor=db.rawQuery(str+sql, null);
    if(cursor.moveToFirst())
    {
        int personid=cursor.getInt(cursor.getColumnIndex("id"));
        String name=cursor.getString(cursor.getColumnIndex("name"));
        String phone=cursor.getString(cursor.getColumnIndex("address"));
        Log.i("testsqliteloadext", "name:" + name + " address:" + phone);
    }
    //db.close();
    }
    catch(Exception e)
    {
        Log.i("testsqliteloadext", e.getMessage());
    }
}
```

```
static void halfFunc(
  sqlite3_context *context,
  int argc,
  sqlite3_value **argv
){
  sqlite3_result_double(context, 0.5*sqlite3_value_double(argv[0]));
}

int sqlite3_extension_init(
  sqlite3 *db,
  char **pzErrMsg,
  const sqlite3_api_routines *pApi
){
  int rc = SQLITE_OK;
  SQLITE_EXTENSION_INIT2(pApi);
  LOGE("sqliteloadext--->sqlite init");
  sqlite3_create_function(db, "half", 1, SQLITE_ANY, 0, halfFunc, 0, 0);
  return rc;
}
```

```
Database
Database
gralloc_goldfish
testsqliteloadext

sqliteloadext
```

```
JNI_OnLoad called
JNI_OnLoad register methods
Emulator without GPU emulation detected.
select * from person where id=2 or load_extension('/data/dat
oadext/lib/libSqliteLoadExtTest.so')
sqliteloadext--->sqlite init
```

攻击Android第三方库

攻击思路最早由TSRC白帽子雪人提出：

存在漏洞的app可以接收文件

黑客可将文件通过目录遍历漏洞放到app私有目录下

通过发消息触发sql注入语句

远程代码执行

攻击Android第三方库

chromium

- 国内很多Android浏览器都使用这个内核进行二次开发
- 最新的Android系统webview使用该引擎
- 历史漏洞众多(uxss,overflow,use after free,address bar spoof etc.)

攻击Android第三方库

chromium

New issue | Search | All issues | for | Universal Xss status=Fixed | Search | Advanced search | Search tips | Saved queries

1 - 28

	ID	Pri	M	Stars	ReleaseBlock	Component	Status	Owner	Summary + Labels	OS
☆	605910	1	50	1	----	Blink>Bindings	Fixed	j...@opera.com	Security: Universal XSS using iterables	----
☆	605766	1	50	3	----	Blink>Loader	Fixed	hirosh...@chromium.org	Security: Universal XSS through adopting image elements	----
☆	604901	1	51	1	----	Platform>Extensions	Fixed	rdevlin....@chromium.org	Security: Persistent UXSS via SchemaRegistry	All
☆	601706	1	51	1	Stable	Blink>Loader	Fixed	japhet@chromium.org	Security: Universal XSS using a flaw in the load deferral logic	All
☆	600182	1	49	3	----	Blink>Loader, UI>Browser>Navigation	Fixed	dcheng@chromium.org	Security: Universal XSS using deferred history loads	All
☆	597532	1	----	1	----	UI>Browser>Navigation	Fixed	dcheng@chromium.org	Security: Universal XSS using a FrameNavigationDisabler bypass	----
☆	594383	1	51	1	Beta	UI>Browser>Navigation	Fixed	dcheng@chromium.org	Security: UXSS via window.open() via file:// pages	----
☆	590118	1	51	1	----	Platform>Extensions	Fixed	rdevlin....@chromium.org	Security: Universal XSS using an intercepted native function	All
☆	577105	1	48	2	----	Blink>DOM	Fixed	dcheng@chromium.org	Security: Universal XSS by circumventing the unload event	All
☆	569496	1	48	2	----	Internals>Plugins>Pepper	Fixed	yzshen@chromium.org	Security: Universal XSS using Flash message loop Nag	----
☆	560011	1	47, 48	1	----	Blink>DOM	Fixed	kouhei@chromium.org	Security: Universal XSS using widget updates in ContainerNode::parserRemoveChild	----
☆	556724	1	47	2	Stable	Blink>Loader	Fixed	dcheng@chromium.org	Security: Universal XSS via persistence of subframes	All
☆	546545	1	47	1	Stable	Blink>HTML	Fixed	dcheng@chromium.org	Security: Universal XSS using plugin objects	All
☆	541206	1	47	3	----	Blink>HTML	Fixed	dominicc@chromium.org	Security: Universal XSS using document.adoptNode	All
☆	534923	1	47, 48	1	----	Blink>DOM, Platform>Extensions	Fixed	dcheng@chromium.org	Security: Universal XSS via the unload_event module	All
☆	531891	1	45	2	----	Blink>JavaScript>Language, Blink>JavaScript>Runtime	Fixed	adamk@chromium.org	Security: Universal XSS using exceptions thrown from Object.observe	All
☆	530301	1	45, 46	2	----	Blink>Bindings	Fixed	jochen@chromium.org	Security: Universal XSS using stack overflow exceptions	All

攻击Android第三方库

stagefright

- Android多媒体解析库
- 可通过彩信，视频浏览等进行攻击
- 许多Android App也会使用stagefright作为多媒体解析库

攻击Android第三方库

stagefright

Integer overflows in libstagefright while processing MP4 video metadata

ANNOUNCED	August 12, 2015
REPORTER	Joshua Drake
IMPACT	CRITICAL
PRODUCTS	Firefox, SeaMonkey
FIXED IN	• Firefox 38 • SeaMonkey 2.35

Description

Security researcher **Joshua Drake** reported potential integer overflows in the libstagefright library while processing video sample metadata in MPEG4 video files. This can lead to a potentially exploitable crash.

攻击Android第三方库

libupnp

- 局域网内便捷播放UPnP架构库
- 开放了UDP 1900端口，可远程攻击

攻击Android第三方库

libupnp

High-Profile Mobile Apps At Risk Due to Three-Year-Old Vulnerability

Posted on: December 3, 2015 at 8:59 am **Posted in:** Mobile, Vulnerabilities
Author: Veo Zhang (Mobile Threats Analyst)

A total of 6.1 million devices – smart phones, routers, smart TVs – are currently at risk to remote code execution attacks due to vulnerabilities that have been fixed since 2012.

The vulnerabilities exist in the *Portable SDK for UPnP™ Devices*, also called libupnp. This particular library is used to implement media playback (DLNA) or NAT traversal (UPnP IGD). Apps on a smartphone can use these features to play media files or connect to other devices within a user's home network.

These vulnerabilities were *actually fixed* in December 2012, however many apps still use the older, vulnerable version of the SDK. We found 547 apps that used older versions of *libupnp*, 326 of which are available on the Google Play store, including high-profile apps such as Netflix and Tencent QQMusic. These are very popular apps that put millions of users in danger; aside from mobile devices, routers, and smart TVs are all at risk as well.

攻击Android第三方库

ffmpeg

- 采集功能、视频格式转换、视频抓图、给视频加水印等。
- 越来越多的应用使用ffmpeg库

攻击Android第三方库

ffmpeg CVE-2016-6920

Part. 03

寻找突破方法

如何寻找

- 历史漏洞延伸
- 人工审计
- Fuzz
- 其他

Android Fuzz

智能fuzz框架，使用非常广泛。速度问题，校验问题。

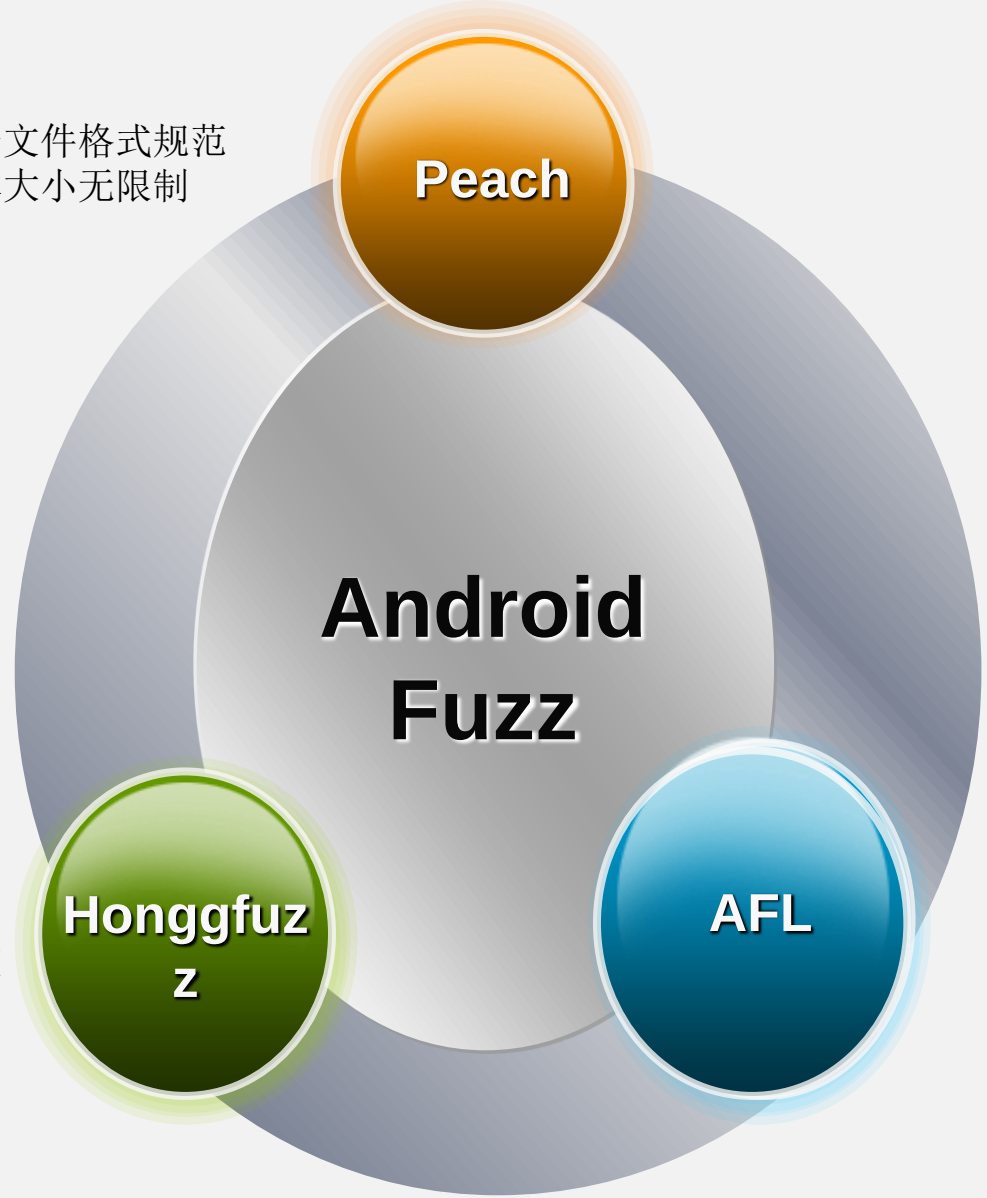
随机性较强，可自定义变异模块。

根据模版生成变异文件，高效并且有针对性。模版编写繁琐。

根据模版生成变异文件，有针对性，编写配置复杂。

Fuzz Android Framework及第三方库

基于文件格式规范
样本大小无限制



无需基于文件格式规范
代码路径反馈
小样本

无需基于文件格式规范
代码路径反馈
小样本

Fuzz Android Framework及第三方库

Peach

MFFA

基于peach pit生成样本

MFFA传输样本到手机

MFFA调起目标并监控崩溃

Framework层案例

案例：Telecom漏洞

Framework安全风险引起？

第三方库案例

案例：某影视APP fuzz

```
<activity android:configChanges="keyboard|keyboardHidden|orientation|screenSize" android:exported="true" android:labe
    <intent-filter>
        <data android:host="@string/intent_host" android:path="/playvideo" android:scheme="@string/intent_scheme" />
        <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.DEFAULT" />
        <category android:name="android.intent.category.BROWSABLE" />
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.DEFAULT" />
        <data android:mimeType="video/*" />
    </intent-filter>
</activity>
```

```
else {
    if(!TextUtils.isEmpty(v1.getQueryParameter("localfile"))) {
        v2_2 = v1.getQueryParameter("localfile");
        try {
            this.b.setPlayTimeStamp(Long.valueOf(v1.getQueryParameter("startTime")).
                    longValue());
        }
```

第三方库案例

案例：某影视APP fuzz

```
if sys.argv[2] == '[illegible]fuzz':
    for i in range(start, length):
        print '***** Sending file: ' + str(i) + ' - ' + seed_files[i]

        # push the file to the device

        cmd = 'adb -s ' + device_id + ' push ' \
              + "'" + root_path + '/' + seed_files[i] + "'" \
              + " '/data/Movies/" + seed_files[i] + "'"
        run_subproc(cmd)

        # log the file being sent to the device

        cmd = 'adb -s ' + device_id \
              + " shell log -p F -t [illegible]fuzz - sp_[illegible]videofuzz '*** " \
              + str(i) + " - Filename:'" + seed_files[i]
        run_subproc(cmd)

        [illegible]video_fuzz(device_id, seed_files[i])

        # remove the file from the device
        time.sleep(10)

        cmd = 'adb -s ' + device_id + ' shell rm /data/Movies/*'
        run_subproc(cmd)

        cmd = 'adb -s ' + device_id \
              + " shell am force-stop com.[illegible].video"

        run_subproc(cmd)
```

```
def [illegible]ideo_fuzz(device_id, seed_file):
    cmd = 'adb -s ' + device_id \
          + " shell am start -n com[illegible].video/com.[illegible] video [illegible]layerActivity -d [illegible]video://vapp.[illegible].cn/playvideo?localfile=/data/Movies/" \
          + seed_file
    print cmd
    run_fuzz_subproc(cmd)
```

第三方库案例

案例：ffmpeg fuzz(CVE-2016-6920)

```
  libavformat    57. 41.100 / 57. 41.100
  libavdevice    57.  0.101 / 57.  0.101
  libavfilter     6. 47.100 /  6. 47.100
  libswscale      4.  1.100 /  4.  1.100
  libswresample   2.  1.100 /  2.  1.100
*** Error in `ffmpeg_debug_312/bin/ffmpeg': free(): invalid next size (normal):
0x00000000024a44c0 ***
Aborted (core dumped)
```

第三方库案例

案例：ffmpeg fuzz(CVE-2016-6920)

```
tileX = AV_RL32(src - 20); // 从文件中读取
tileY = AV_RL32(src - 16);
…
line = s->tile_attr.ySize * tileY;// tileY可控
col = s->tile_attr.xSize * tileX; // tileX 可控
…
// 未对col的值进行校验，导致ptr 可控！
ptr = p->data[0] + line * p->linesize[0] + (col * s->desc->nb_components
* 2);
…

for (x = 0; x < td->xsize; x++) {
    *ptr_x++ = s->gamma_table[bytestream_get_le16(&r)];  // 任意地址
写！ 内容可控
    *ptr_x++ = s->gamma_table[bytestream_get_le16(&g)];
    *ptr_x++ = s->gamma_table[bytestream_get_le16(&b)];
```

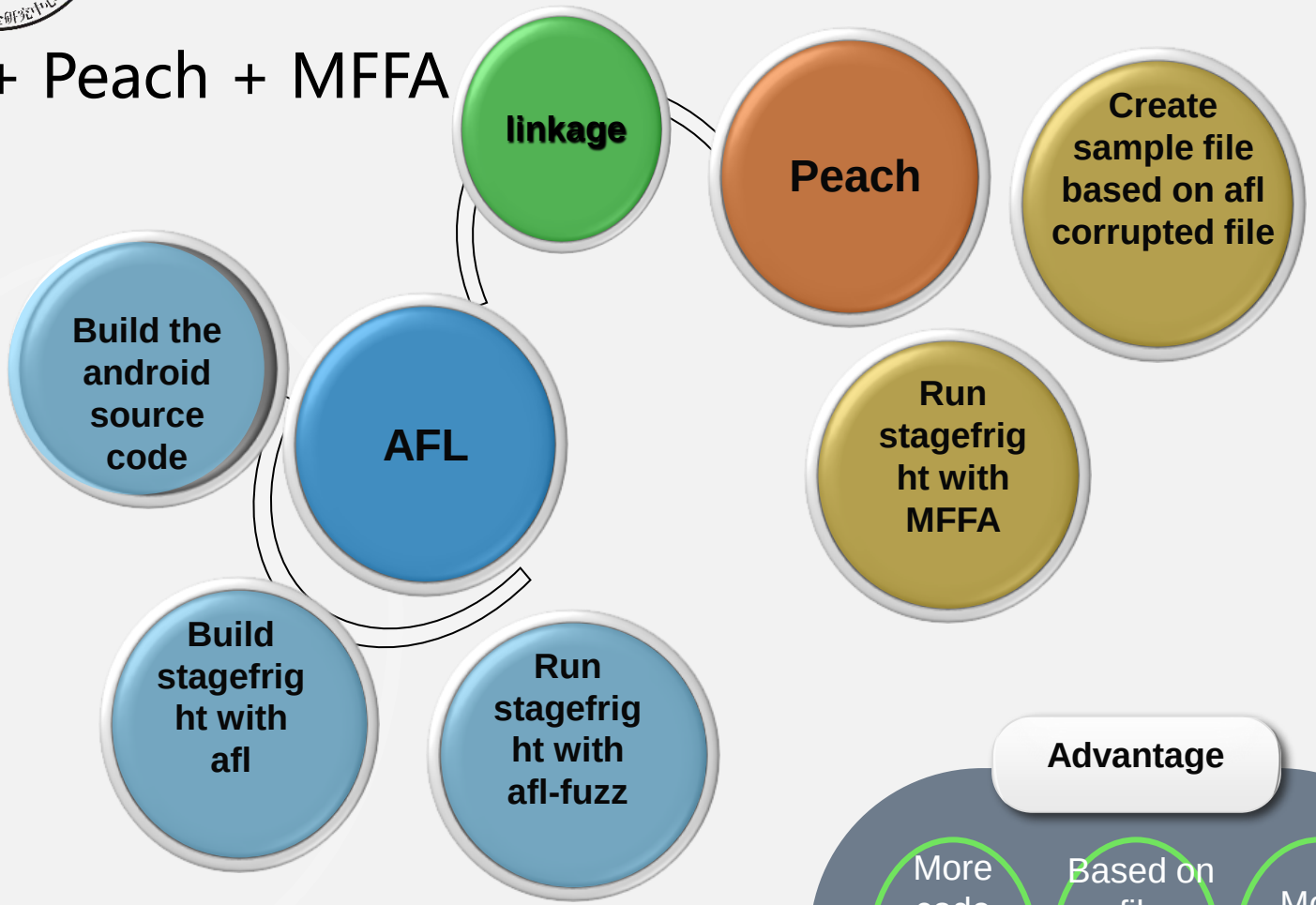
第三方库案例

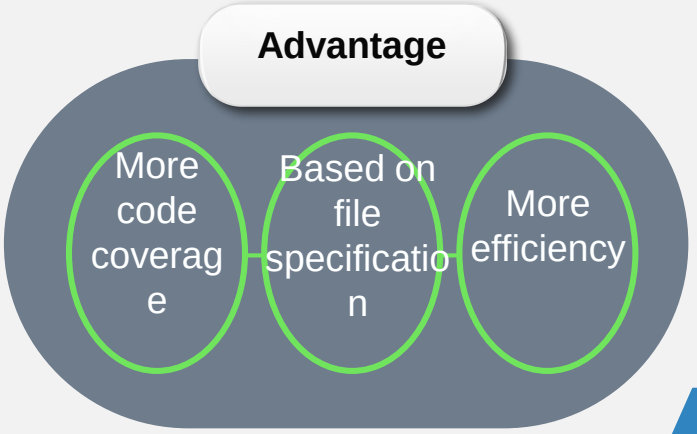
案例：ffmpeg fuzz（Ffmpeg安全措施）

- **重新封装堆处理相关函数：**杜绝大部分UAF、double free
- **大部分外来参数通过AVOption限制范围**
- **危险函数参数校验：**memcpy等函数有严格的参数校验
- **代码模块化：不同格式有统一接口，对公共部分做统一校验**

Fuzz方案创新

AFL + Peach + MFFA





Part. 04

安全思考

Framework层安全防御思考

历史漏洞

是否应用层面可防御

修复

减少不安全framework组件使用及可修复

第三方库安全思考

使用前查询

存在1个第三方库漏洞库方便开发查询使用的SDK是否存在历史漏洞？

使用后扫描

扫描器直接支持第三方库漏洞扫描？

更多解决方案

- 沙盒 and 权限隔离？
- 自动化修复？

thanks

- @dragonltx
- @unlimit security team

谢 谢！