APK 防二次打包解决方案

SPEAKER

何晓杰

为什么要防止二次打包

- 恶意代码或广告注入APK
- 现有防二次打包的方案持续被破解
- Android = Windows?

- 完美？成本？破解成本？

主流防二次打包方案核心原理及问题

沪江 HUJIANG.COM

- 代码和资源混淆和加密
- DEX全盘加密
- 运行时字节码变形
- 签名和关键信息校验

- 兼容性是大问题
- 执行效率会变低
- 占用更多系统资源
- 成熟的方案更容易成为目标

为什么要使用JNI

- JVM层的代码容易被反编译
- JNI执行效率更高
- JNI破解成本更高
- JNI可以访问到一些JVM层无法访问的内容
- 原生代码加壳方式已成熟

JNI的作用

签名及程序
完整性校验

检查具体的服务
是否正常工作

实现
Magic Number

收集破解者/
使用盗版者的信息

警告与崩溃

常见的签名校验

获取签名信息并与
标准字符串比对

获取签名信息并与
服务器端字符串比对

获取签名信息并加密后
与加密字符串比对

- 比对！比对！比对！

破解常见的签名校验

找到关键函数名

```
.text:0050DFB8          MOV     R8, R0
.text:0050DFBC          BL      nativeGetApkSignature
.text:0050DFC0          CMP     R8, #0
```

反编译函数代码

```
memset(&s, 0, 0, 0x7D0u);
v2 = sub_50DE8C(
    v1,
    (int) 'sD1CkCQ8ATLT+XT00WAL7gvnhWgKV3D77PtRdkt8m62/B+
    1040,
    &s);
v3 = (const char*) nativeGetApkSignature();
if ( !v2 || (result = strcmp(v3, &s)) != 0 )
```

修改函数的返回值

```
005046A8    00 00 96 E5 00 10 E0 E3    B4 3F FF EB 04 00 A0 E1    .........?......
005046B8    29 26 00 EB 0C D0 8D E2    F0 80 BD E8 20 B3 6B 00    )&.......?..,.k.
005046C8    E0 87 6C 00 48 CF FF FF    BA 0D 55 00 98 7B 6C 00    ..l.H.....U..{l.
```

较为安全的签名校验

不把签名信息写在连续的内存空间内

```
.     property A0: string read FA0 write FA0;
1145  property A1: string read FA1 write FA1;
.     property A2: string read FA2 write FA2;
.     property A3: string read FA3 write FA3;
.     property A4: string read FA4 write FA4;
.     property A5: string read FA5 write FA5;
```

```
2270  property A1126: string read FA1126 write FA1126;
.     property A1127: string read FA1127 write FA1127;
.     property A1128: string read FA1128 write FA1128;
.     property A1129: string read FA1129 write FA1129;
.     property A1130: string read FA1130 write FA1130;
```

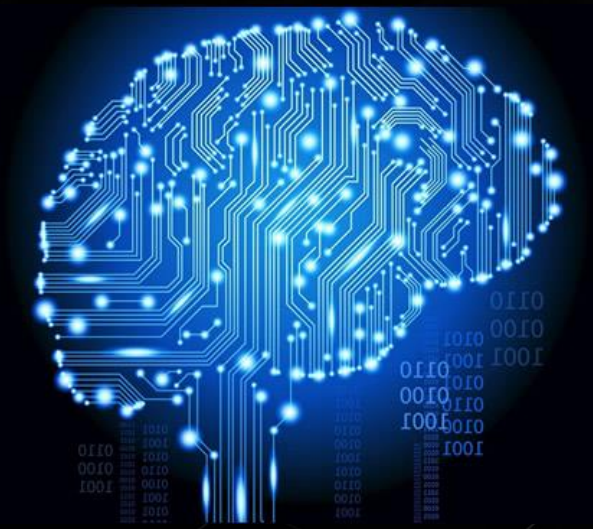
```
.     FA0 := '3';
.     FA1 := '0';
.     FA2 := '8';
.     FA3 := '2';
2285  FA4 := '0';
```

```
.     FA1128 := 'a';
3410  FA1129 := '2';
.     FA1130 := 'a';
.     FA1131 := '8';
```

不使用0和1作为返回值

```
.     if (ASig[1] <> FA0) then begin Result := 16003; Exit; end;
.     if (ASig[2] <> FA1) then begin Result := 2301; Exit; end;
3420  if (ASig[3] <> FA2) then begin Result := 9086; Exit; end;
.     if (ASig[4] <> FA3) then begin Result := 872457; Exit; end;
.     if (ASig[5] <> FA4) then begin Result := 981; Exit; end;
```

安利一个强大的开发工具



CodeTyphon Studio

is the FREE and Open Source
Pascal Visual Programming Studio

with Multi-OS, Multi-CPU, Multi-Platform
and Cross-Build abilities.

for REAL

Native Development

高效的、跨平台的、编译简单的、第三方库丰富的

检查程序完整性以及是否工作

- 包名检查
- 关键Activity、Service、Receiver检查

- $ pm enable <package name>/<class name>

实现MAGIC NUMBER

- Magic Number不影响APK的解析与运行
- 写入Magic Number可以作为APK的标志
- Magic Number基本上不能被识别和伪造
- 二次打包后，Magic Number会消失

实现MAGIC NUMBER

定义Magic Number
的写入位置和值

```
type
  TMagicPair = record
    offset: Integer;
    value: Byte;
  end;
  TMagicPairs = array[0..6]of TMagicPair;
  TMagics = array of TMagicPairs;
  PMagicPairs = ^TMagicPairs;

const
  BASE_MAGIC: TMagicPairs = (
    (offset: 4; value: $ff),
    (offset: 5; value: $01),
    (offset: 7; value: $12),
    (offset: 8; value: $80),
    (offset: 9; value: $01),
    (offset: 10; value: $99),
    (offset: 11; value: $0A)
  );
```

实现MAGIC NUMBER

实现多套Magic Number数值

```
function getMagicPairs: TMagics;
var
  i, j: Integer;
begin
  SetLength(Result, 20);
  for i:=0 to Length(Result) - 1 do begin
    for j:= 0 to Length(BASE_MAGIC) - 1 do begin
      Result[i][j].offset := BASE_MAGIC[j].offset;
      Result[i][j].value:= BASE_MAGIC[j].value + (i * 2 or j);
    end;
  end;
end;
```

向APK写入Magic Number

```
function writeMagicPair(filePath: string; magic: TMagicPairs): Boolean;
var
  fs: TFileStream;
  i: Integer;
begin
  Result := True;
  fs := TFileStream.Create(filePath, fmOpenReadWrite);
  for i := 0 to Length(magic) - 1 do begin
    fs.Seek(magic[i].offset, fsFromBeginning);
    if (fs.Write(magic[i].value, 1) <> 1) then begin
      Result := false;
      Break;
    end;
  end;
  fs.Free;
end;
```

实现帐户信息收集

```
clsAccMgr := env^^.FindClass(env, 'android/accounts/AccountManager');
getAccMgr := env^^.GetStaticMethodID(env, clsAccMgr, 'get', '(Landroid/content/Context;)Landroid/accounts/AccountManager;');
objAccMgr := env^^.CallStaticObjectMethodA(env, clsAccMgr, getAccMgr, argsToJValues(env, [CONTEXT]));
getAccountsAccMgr := env^^.GetMethodID(env, clsAccMgr, 'getAccounts', '()[Landroid/accounts/Account;');
objAccounts := env^^.CallObjectMethodA(env, objAccMgr, getAccountsAccMgr, nil);
clsAcc := env^^.FindClass(env, 'android/accounts/Account');
typeAcc := env^^.GetFieldID(env, clsAcc, 'type', 'Ljava/lang/String;');
nameAcc := env^^.GetFieldID(env, clsAcc, 'name', 'Ljava/lang/String;');
if (objAccounts <> nil) then begin
  size := env^^.GetArrayLength(env, objAccounts);
  for i := 0 to size - 1 do begin
    objAcc := env^^.GetObjectArrayElement(env, objAccounts, i);
    Result += Format('%s:%s|', [
      jstringToString(env, env^^.GetObjectField(env, objAcc, typeAcc)),
      jstringToString(env, env^^.GetObjectField(env, objAcc, nameAcc))]);
    env^^.DeleteLocalRef(env, objAcc);
  end;
end;
```

```
procedure TAccountThread.Execute;
var
  AParam: TStringList;
begin
  AParam := TStringList.Create;
  AParam.Add('pkg=' + PKG_NAME);
  AParam.Add('accs=' + Faccs);
  HttpPost(ACC_URL, AParam);
  AParam.Free;
end;
```

- 将收集到的信息发送到服务器, 为了避免耗时操作引起ANR, 需要将此操作放入线程内执行

实现警告和崩溃

```
    if (isOnTop(env)) then begin
      getUserSerial(env);
      pkgName:= getPackageName(env);
      if (USER_SERIAL = '') then begin
        stdlib.system(PChar(Format('am start -a action.fake.%s &', [pkgName])));
      end else begin
        stdlib.system(PChar(Format('am start --user %s -a action.fake.%s &' , [USER_SERIAL, pkgName])));
      end;
      Result := True;
    end else begin
      Result := True;
    end;
  except
  end;
```

- 在JNI内执行am或pm操作，必须在命令尾部加入 & 符号以表示不阻塞执行，否则很可能因此产生ANR

保护JNI

- 防止JNI被无效
- 防止JNI被破解和恶意修改

- JVM和JNI双向保护
- 对JNI库进行加壳
- 向程序中其他JNI注入代码，以检查当前JNI
- 将写在JNI中的业务代码与当前JNI融合

写一个工具来注入SMALI代码

- 注入的代码强调随机、隐蔽，不可轻易识别
- 注入的代码要可以经常的对JNI进行检查
- 永远不要注入两段相同的代码
- 注入一些能让反编译器出错的代码

```
.       Insert(idx, '    new-instance v1, Landroid/content/Intent;');
380     idx += 1;
.       Insert(idx, '    move-object v0, p0');
.       idx += 1;
.       Insert(idx, '    check-cast v0, Landroid/content/Context;');
.       idx += 1;
385     Insert(idx, '    const-class v2, Lcom/hujiang/gl/ClassService;');
.       idx += 1;
.       Insert(idx, '    invoke-direct {v1, v0, v2}, Landroid/content/Intent;-><init>(Landroid/content/Context;Ljava/lang/Class;)V');
.       idx += 1;
.       Insert(idx, format('    invoke-virtual {p0, v1}, %s->startService(Landroid/content/Intent;)Landroid/content/ComponentName;', [clsSmali]));
390     idx += 1;
.       Insert(idx, Format('    invoke-virtual {p0}, %s->_____req()V', [clsSmali]));
```

让反编译器出错

- 注入 Kotlin 框架, 使用嵌套的try语句

```
    public g() {
        /* JADX: method processing error */
/*
Error: jadx.core.utils.exceptions.DecodeException: Load method exception in method: com.hu
     at jadx.core.dex.nodes.MethodNode.load(MethodNode.java:116)
     at jadx.core.dex.nodes.ClassNode.load(ClassNode.java:249)
     at jadx.core.ProcessClass.process(ProcessClass.java:34)
     at jadx.api.JadxDecompiler.processClass(JadxDecompiler.java:306)
     at jadx.api.JavaClass.decompile(JavaClass.java:62)
Caused by: java.lang.NullPointerException
     at jadx.core.dex.nodes.MethodNode.addJump(MethodNode.java:370)
     at jadx.core.dex.nodes.MethodNode.initJumps(MethodNode.java:360)
     at jadx.core.dex.nodes.MethodNode.load(MethodNode.java:106)
     ... 4 more
*/
```

- 使用 Companion 来隐藏真实变量和方法

```
@Metadata(bv = {1, 0, 0}, d1 = {"\u0000 \n\u0002\u0018\u0002\n\
/* compiled from: ClassService.kt */
public final class ClassService extends Service {
    @NotNull
    private static final String A = A;
    @NotNull
    private static final String C = C;
    public static final Companion Companion = new Companion();
    @NotNull
    private static final String D = D;
    @NotNull
    private static final String L = L;
```

安利一个强大的语言

简洁的、安全的、图灵完备的、JVM通用的、独立的

MULTIDEX

- 当注入方法时，单个DEX内方法数超过65535会引起出错
- 需要设计算法，在注入代码之前找出Application、MainActivity 和直接被它们引用的类，除此之外的其的类在DEX方法数超出时，可进行移动操作

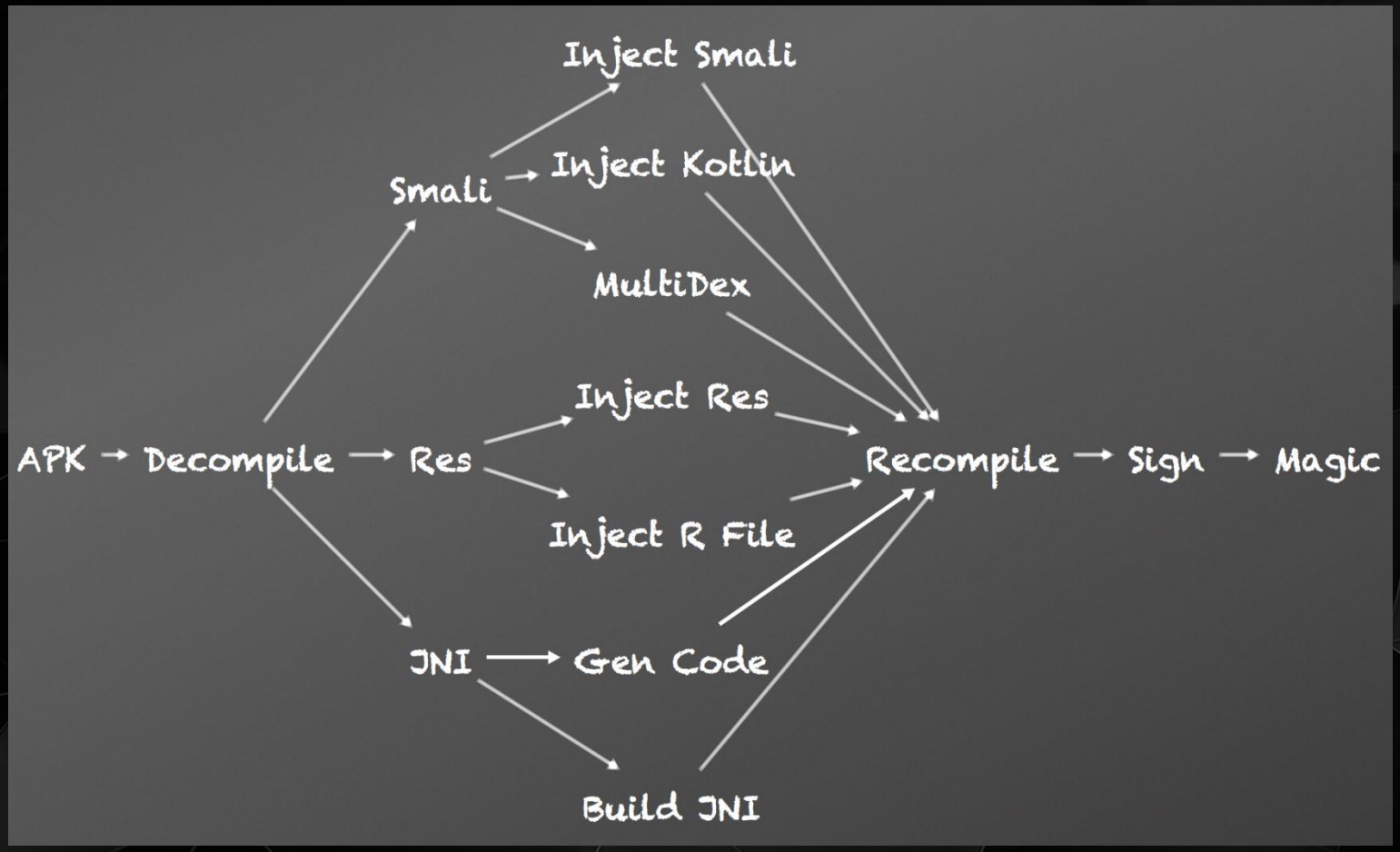
```
  if (idx = -1) then begin
    Result := 'smali';
80  end else begin
    Result := Format('smali_classes%d', [idx]);
    methodCount:= getMethodCount(outPath + Result);
    if (methodCount + GUARD_COUNT > 60000) then begin
      idx += 1;
85    Result := Format('smali_classes%d', [idx]);
      ForceDirectories(outPath + Result);
    end;
  end;
```

一体化JNI保护平台的设计

- 对APK处理的步骤较多，无法每次手工处理
- 手动处理容易出错，特别是手工混淆代码时
- 手写的随机数往往带有主观规律因素
- 手动处理不易做知识传递，对处理者要求高

- **开发一个自动处理的平台来搞定一切吧!**

完整的处理步骤梳理

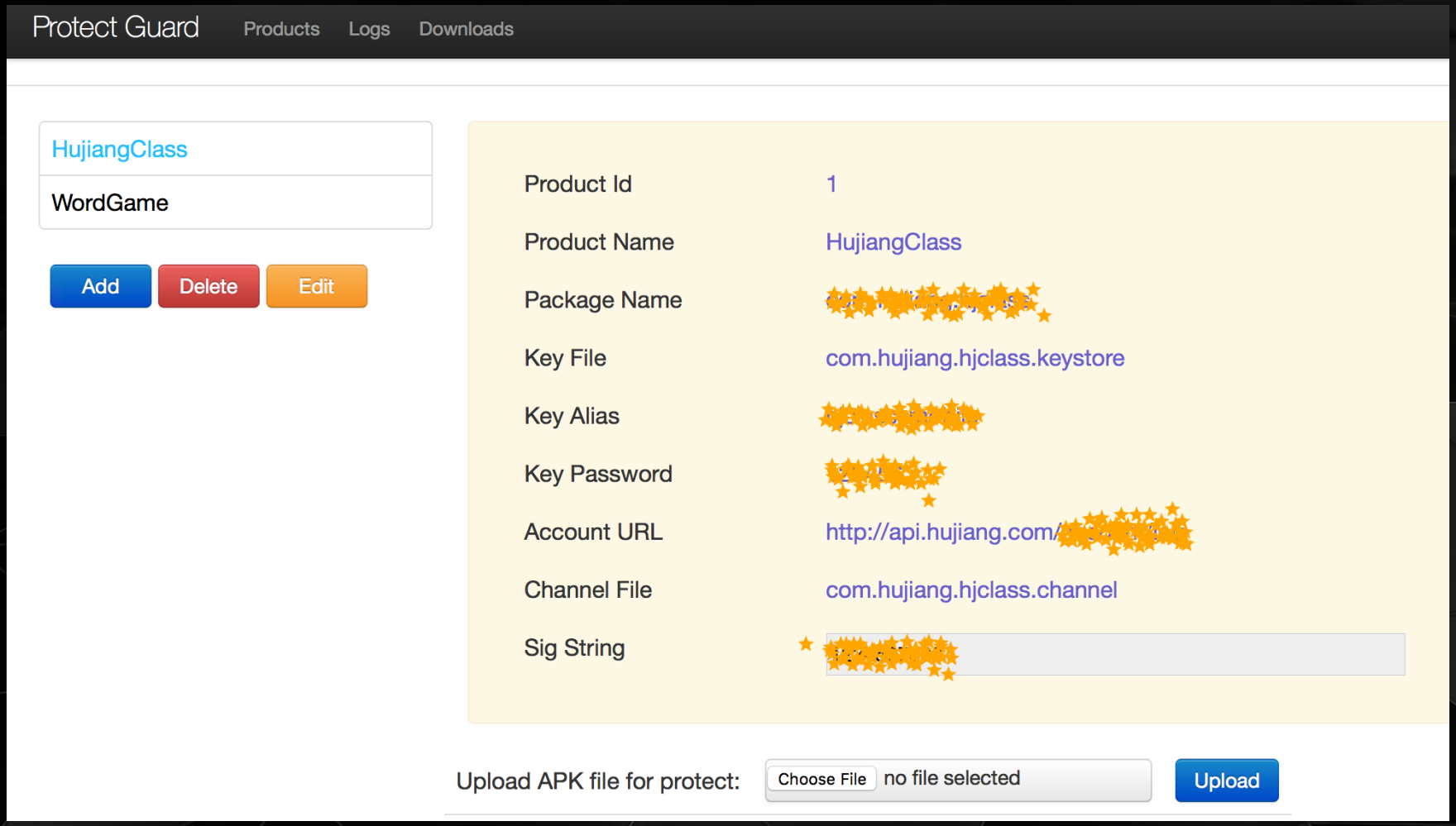

Inject Smali
Smali
Inject Kotlin
MultiDex
Inject Res
APK
Decompile
Res
Recompile
Sign
Magic
Inject R File
JNI
Gen Code
Build JNI

一个简单的工具

```
  // mkcfg
  ret := RunCommand(cmdMkCfg, [pkgName, keyFile, keyAlias, keyPass], outputStr, [poWaitOnExit, poUsePipes, poStderrToOutPut]);
  if (not ret) then begin
    WriteLn(Format('[mkcfg][%s][error] %s', [FormatDateTime(DATE_FMT, Now), outputStr]));
    Exit;
  end else begin
    WriteLn(Format('[mkcfg][%s][done]', [FormatDateTime(DATE_FMT, Now)]));
  end;
  // mkso
  ret := RunCommand(cmdMkSo, [pkgName, keyAlias, pkgAccUrl], outputStr, [poWaitOnExit, poUsePipes, poStderrToOutPut]);
  if (not ret) then begin
    WriteLn(Format('[mkso][%s][error] %s', [FormatDateTime(DATE_FMT, Now), outputStr]));
    Exit;
  end else begin
    WriteLn(Format('[mkso][%s][done]', [FormatDateTime(DATE_FMT, Now)]));
  end;
  // batch protect
  ret := RunCommand(cmdProtect, [IfThen(keepTemp, '-k', '-c'), pkgChannel, apkPath, pkgAccUrl], outputStr,
    [poWaitOnExit, poUsePipes, poStderrToOutPut]);
  if (not ret) then begin
    WriteLn(Format('[batch protect][%s][%s][error] %s', [FormatDateTime(DATE_FMT, Now), pkgName, outputStr]));
  end else begin
    WriteLn(Format('[batch protect][%s][%s][done] %s', [FormatDateTime(DATE_FMT, Now), pkgName, outputStr]));
  end;
```

```
rarnu-mac:release rarnu$ ./batch
usage: batch <key file> <key alias> <key password> <package name> <channel file> <account url> <keep temp> <apk path>
```

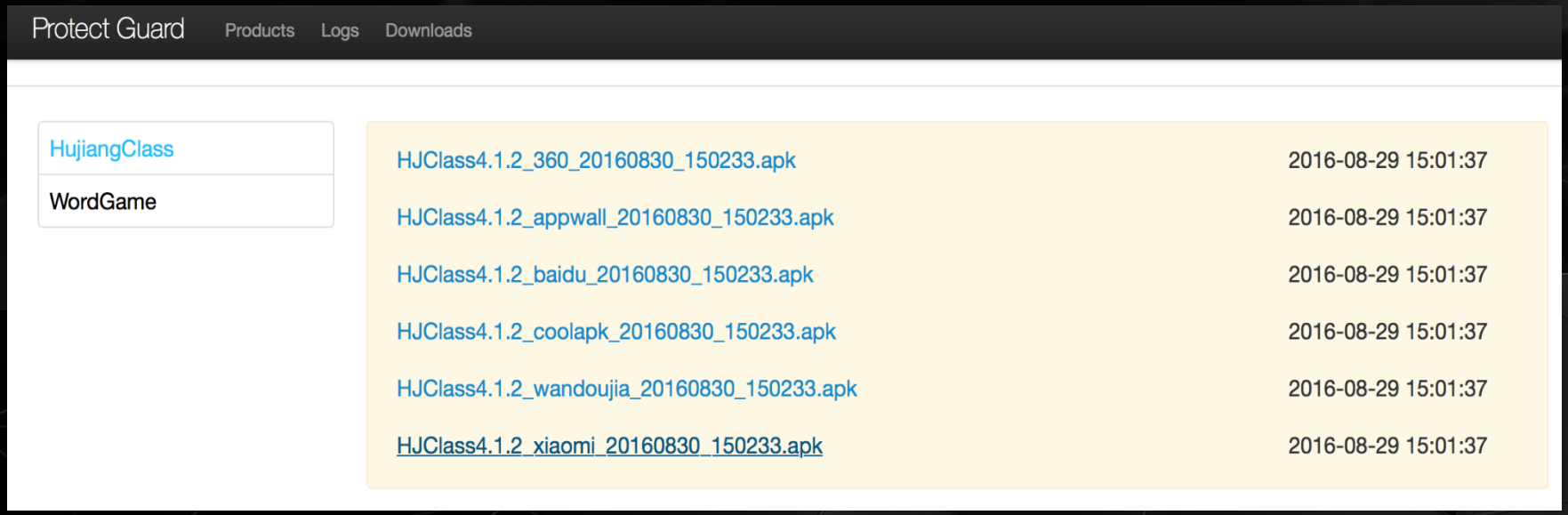
实现对用户友好的平台


Protect Guard
Products
Logs
Downloads
HujiangClass
WordGame
Add
Delete
Edit
Product Id
1
Product Name
HujiangClass
Package Name
Key File
com.hujiang.hjclass.keystore
Key Alias
Key Password
Account URL
http://api.hujiang.com/
Channel File
com.hujiang.hjclass.channel
Sig String
Upload APK file for protect:
Choose File
no file selected
Upload

实现一个打包的Linux服务

```
160   lockFile := ExtractFilePath(ParamStr(0)) + '../lock/protect';
      while True do begin
        Sleep(1000);
        if FileExists(lockFile) then begin
          // read lock file
165       with TStringList.Create do begin
            LoadFromFile(lockFile);
            txtLock := Text;
            Free;
          end;
170       // extract id and apkname in lock file
          with TStringList.Create do begin
            Delimiter := '|';
            DelimitedText := txtLock;
            appId := StrToIntDef(Strings[0], -1);
175         apkName := Strings[1];
            Free;
          end;
          apkPath := ExtractFilePath(ParamStr(0)) + '../apk/' + apkName;
          if (appId <> -1) and (apkName <> '') and (FileExists(apkPath)) then begin
180         // do protect apk
            batchProtect(appId, apkPath);
          end;
          // protect done, delete the lock file, the website will not show "Protecting" then.
          DeleteFile(lockFile);
185     end;
      end;
```

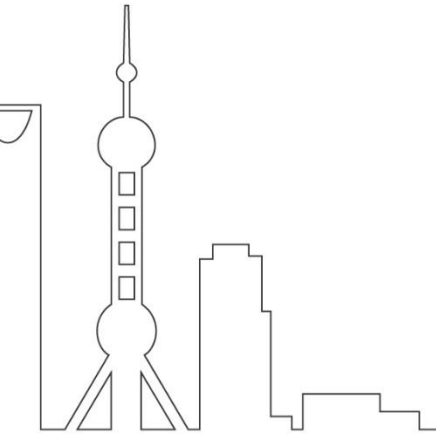
实现对用户友好的平台


Protect Guard
Products
Logs
Downloads
HujiangClass
WordGame
HJClass4.1.2_360_20160830_150233.apk
2016-08-29 15:01:37
HJClass4.1.2_appwall_20160830_150233.apk
2016-08-29 15:01:37
HJClass4.1.2_baidu_20160830_150233.apk
2016-08-29 15:01:37
HJClass4.1.2_coolapk_20160830_150233.apk
2016-08-29 15:01:37
HJClass4.1.2_wandoujia_20160830_150233.apk
2016-08-29 15:01:37
HJClass4.1.2_xiaomi_20160830_150233.apk
2016-08-29 15:01:37

写在最后

- 没有绝对的安全，只有增加的成本
- 每一处出人意料的改动，都是巨大的破解成本
- 相互保护是有必要的，全家桶有的时候并非坏事
- 每个公司都应当有自己的保护方案，并保持更新

沪江



Thanks!

International Software Development Conference

沪江技术学院

主办方 Geekbang. 极客邦科技 InfoQ