Shell

简介

基于dex加密实现的一个简单加固的插件。

使用步骤

1. 项目根目录下的build.gradle中引入插件:

```
buildscript {

    repositories {
        //...
        maven { url 'https://dl.bintray.com/milovetingting/maven' }
        //...
    }
    dependencies {
        //...
        classpath 'com.wangyz.plugins:ShellPlugin:1.0.0'
        //...
    }
}

allprojects {
    repositories {
        //...
        maven { url 'https://dl.bintray.com/milovetingting/maven' }
        //...
    }
}
```

2. app模块下的build.gradle引入插件及配置插件

```
apply plugin: 'com.wangyz.plugins.ShellPlugin'

//主要注意shellModuleName和shellApplication的配置
shellConfig {
    //壳Module的名称
    shellModuleName = 'shell'
    //壳Module中Application的全类名
    shellApplication = 'com.wangyz.shell.ShellApplication'
    keyStore = 'E:\\Code\\Android\\android.keystore'
    keyStorePassword = 'android'
    keyPassword = 'android'
```

```
    alias = 'android'
}
```

3. 壳Module中增加继承自Application的自定义Application,如这里的ShellApplication,然后重写attachBaseContext方法

```
public class ShellApplication extends Application {

    @Override
    protected void attachBaseContext(Context base) {
        super.attachBaseContext(base);
    }

}
```

4. sync工程

5. 在打包apk前，先执行Build-Clean Project,然后双击gradle面板的app/Tasks/build/assembleRelease，就会在项目根目录/壳Module名称-release/outputs/下生成signed.apk,这个apk就是加固过的apk.

注意事项

1. 这里只是演示加固的思路，对于加密部分，只是用了简单的^操作，具体可以自己换成AES， RSA或者其它加密方式。

2. 插件会用到dx,gralde的命令，因此需要配置这两个的路径

3. 插件会用到ASM，在编译出class后修改class。在生成apk后，没有修改过代码或者没有执行sync的操作后，transform的回调不会走，因此也不会执行修改class的逻辑，因此在每次生成加固apk前，需要执行clean项目的操作。

4. 引入插件后的配置文件一定不能错，重点关注shellModuleName和shellApplication，否则会导致生成的apk无法正常使用。