MTP命令行加固工具指引

工具说明

MTP命令行加固工具，可以把加固和签名过程集成到ci流程中，只需一行代码，即可实现加固和签名操作的自动化。

命令行正式版加固功能支持so导出表混淆、AAB加固功能、V1/V2/V3签名、自定义壳版本；

在正式版加固功能的基础上，最新版加固功能支持global-metadata加密、壳对反外挂SDK保护能力、新增检测dex文件以及lib目录下的so文件（该功能前提需要用户选择全校验）；

游戏侧可前往下载命令行加固工具

在使用MTP命令加固工具对游戏进行加壳前，可先浏览开发者建议

特别注意：

1. 本工具支持的游戏版本：Android游戏；
2. 本工具支持的操作系统：Windows、MAC OS、Linux（对于Windows和Linux系统，分别提供了x86版和x64版工具，请先确认电脑的操作系统配置，再选择对应的工具版本）；
3. 本工具运行时会首先对游戏进行鉴权，MTP会为每款游戏颁发cert格式的证书，如果你下载的工具包内没有对应游戏的证书，请联系客服进行获取；
4. 本工具提供两种模式：自定义加固模式、App Bundle加固模式；
5. 在MAC和Linux系统上调用工具前，请首先为工具赋权：在本工具所在目录下执行；

```
cd where_your_mtpclientconsole_dir
chmod -R 777 ./*
```

多渠道注意事项（注：多渠道相关场景可查看游戏多渠道问题操作指引）

1. 如果有需要对多个游戏渠道安装包进行批量加固，可参考文档底部批量处理伪代码进行脚本编写。（强烈建议）
2. 如果有需要对加固后的游戏进行多渠道打包，建议游戏加固时使用自定义加固模式，在配置文件中对渠道修改的文件取消校验；

自定义加固模式

自定义加固模式，支持的该功能更加全面。通过自己填写并调用配置文件，可以自由选择对指定的so文件、dll文件进行加密，可以自由选择对文件进行全校验还是部分校验，默认状态对文件不进行校验，同时还可以配置签名信息，加固后会自动对游戏包进行签名处理。其中对文件进行部分校验时，可先详细阅读部分校验风险说明

调用方法：

```
cd where_your_mtpclientconsole_dir
MTPClientConsole.exe -d gameId apkpath outDir certPath -c configPath
```

参数说明：

参数	含义
-d	标准版（默认选项，不可变更）
gameId	游戏ID（MTP注册）
apkPath	待加固的apk文件完整路径（不要包含中文字符）
outDir	输出加固后apk文件完整路径（不要包含中文字符）
certPath	与游戏ID对应的证书路径（证书名为Game_ID.cert且不要包含中文字符）
-c	调用配置文件（默认选项，不可变更）
configPath	config配置文件的路径（不要包含中文字符），配置文件中成对的标签需放在同一行且文件名称以.xml结尾

App Bundle加固模式

App Bundle加固模式，Android App Bundle是Google最新推出的Apk动态打包，动态组件化的技术，最终打包生成.aab结尾的bundle文件。通过自己填写并调用配置文件，可以自由选择对指定的so文件、dll文件进行加密，客户端App Bundle加固模式不支持部分校验模式，会默认对App Bundle内文件进行校验。（目前支持的App bundle文件大小上限为500M，仅限在64位机器上使用）

调用方法：

```
cd where_your_mtpclientconsole_dir
MTPClientConsole.exe -a gameId aabPath outDir certPath -c configPath
```

参数说明：

参数	含义
-a	AppBundle版（默认选项，不可变更）
gameId	游戏ID（MTP注册）
aabPath	待加固的aab文件完整路径（不要包含中文字符，文件大小不超过500M）
outDir	输出加固后apk文件完整路径（不要包含中文字符）
certPath	与游戏ID对应的证书路径（证书名为Game_ID.cert且不要包含中文字符）
-c	调用配置文件（默认选项，不可变更）
configPath	config配置文件的路径（不要包含中文字符），配置文件中成对的标签需放在同一行且文件名称以.xml结尾

config配置文件的写法：

```
<?xml version='1.0' encoding='utf-8'?>
<Config>
    <!--Server Tools Version, please use default value-->
    <TPVersion>default value</TPVersion>
    <!--ToolPath: mtp tools location,
     default MTPClientConsole/tools or cut this attribute-->
    <ToolPath>./tools</ToolPath>
    <!--EncSo: so fileName that you want to encrypt,
     less than 5 files for performance-->
    <EncSo>libmono.so</EncSo>
    <EncSo>libGameCore.so</EncSo>
    <!--EncDll: dll fileName that you want to encrypt,
     less that 5 files for performance-->
    <EncDll>Assembly-CSharp.dll</EncDll>
    <!--Sign: information of keystore when using local sign-->
    <Sign>
        <sign-argument name='keystorePath' value='your_keystore_absolute_path'/>
        <sign-argument name='keypass' value='123456'/>
        <sign-argument name='storepass' value='123456'/>
        <sign-argument name='alianame' value='test'/>
        <!--Application Signing, Please read the Google official documentation-->
        <sign-argument name="v1SigningEnabled" value="true"/>
        <sign-argument name="v2SigningEnabled" value="false"/>
        <sign-argument name="v3SigningEnabled" value="false"/>
    </Sign>

    <!--Encryption configuration for globalmetadata-->
    <extparams>{"enable_globalmetadata_enc":true}</extparams>

    <!--FileCheck: verify all file in the installation package-->
    <!--selector: Optional file verification -->
    <FileCheck name="root" action="+">
        <!-- The following is an example configuration, please edit as needed -->
        <!-- Note：The content of the name key-value pair can't start or end with "/" --
        <!-- <selector name="assets/bin" action="-">
             <selector name="assets/bin/Data/Managed" action="+">
                 <selector name="assets/bin/Data/Managed/UnityEngine.dll" action="-"/>
                 <selector name="assets/bin/Data/Managed/System.dll" action="-"/>
             </selector>
             <selector name="assets/bin/Data/Resources" action="+"/>
         </selector>
         <selector name="res/layout" action="-">
             <selector name="res/layout/check_permission_layout.xml" action="+"/>
         </selector> -->
    </FileCheck>
</Config>
```

config配置文件说明：

配置项	含义
TPVersion	必填项，指定壳版本号
ToolPath	MTP命令行加固工具的运行，会依赖工具包内的tools目录信息，请填写该tools目录的完整绝对路径（不带引号）
EncSo	加密so文件名列表，建议只选择主游戏逻辑so，加密so的数量过多会影响游戏性能（不带引号）

配置项	含义
EncDll	加密dll文件名列表，选择Assembly.dll等游戏主逻辑脚本（不带引号）
keystorePath	密钥文件地址，字符串类型（请填写绝对路径）
keyPass	密钥库口令，字符串类型
storePass	存储库口令，字符串类型
aliaName	别名，字符串类型
v1SigningEnabled	使用v1签名，字符串类型，只能是“true”或“false”
v2SigningEnabled	使用v2签名，字符串类型，只能是“true”或“false”
v3SigningEnabled	使用v3签名，字符串类型，只能是“true”或“false"
extparams	开启global-metada加密功能
action	部分校验文件操作，“+”为勾选，“-”为不勾选
selector name	部分校验文件路径，其路径不能以/开头或结尾，且需严格按照xml层次关系

特别注意：

***关于TPVersion**：建议使用命令行工具自带的默认版本，如需更换版本，请联系我们

***关于签名：** 使用本地签名，证书和证书信息无需上传，keystorePath、storePass、keyPass、aliaName、v1SigningEnabled、v2SigningEnabled、v3SigningEnabled不填或者填错，则默认不签名，加固后的游戏需要手动签名。 无论是使用v1、v2、v3签名，由开发者决定，请在使用前，阅读Google官方相关文档，一般而言，aab仅使用v1签名。

***关于加密文件列表：** 若EncSo、EncDll为空列表或不填写，默认对 libmono.so文件、libil2cpp.so文件、Assembly-CSharp.dll文件、Assembly-CSharp-firstpass.dll文件等进行加密

***关于global-metadata.dat加密** extparams字段是加密global-metadata使用的配置，若不需要该功能，可直接将extparams配置直接删除即可。仅在最新加固版本支持global-metadata加密功能且默认开启。

***关于资源文件列表：** 自定义加固模式配置文件中若FileCheck不填写，默认对资源文件不进行校验 App Bundle加固模式不支持部分校验模式，默认全部文件进行校验

***关于极速上传：** 本版本的命令行加固工具已经集成了极速上传功能，精简网络传输内容，提升传输速度,且无需上传完整包。

***关于校验文件选择：** FileCheck name中，root节点为默认动作，“+”为勾选，“-”为不勾选，当需要从节点中执行相反动作时，可添加到selector attribute中，如配置文件中的举例，assets/bin目录不勾选，assets/bin/Data/Managed目录勾选时，则将目录下的所有选项进行勾选，再将其目录下的UnityEngine.dll文件与System.dll文件取消勾选，在勾选assets/bin/Data/Resourses及其目录下的子文件，不勾选res/layout及其目录下的全部子文件，选择其子文件中的check_permission_layout.xml进行勾选。（可等价如下web加固中的操作）

常见返回结果

参数	含义
The shelling task was finished	加固任务完成
Shell failed for time out	加固超时失败
Configuration error	配置解析错误
Command line parameter error	命令行参数错误
Shelling error	加壳错误
Local client error	本地客户端错误
Server error	服务器错误

部分校验风险说明

MTP壳对未选择校验的文件不进行保护，一旦被玩家修改，MTP壳不做处理；相关情况说明：

1. 如果加固时未对游戏逻辑相关二进制文件进行校验，存在玩家对其进行修改替换生成破解版功能的安装包；
2. 如果加固时未对class.dex文件进行校验，存在玩家删除一些java相关的代码重打包后导致一些功能无法使用情况；
3. 如果加固时未对class.dex文件进行校验，存在玩家修改dex来加载新增加的so文件，从而修改游戏逻辑生成破解版功能的安装包；

开发者建议

结合目前使用的加壳问题和Google开发者规范，提供以下建议：

1. 尽量保证各arch下的so文件一致，如果某文件在某arch下不存在，则选择删除该arch文件夹，或者补齐。
2. 尽量保证dex方法数远低于Google规范限制的65535个，如果数量处于边缘，选择MultiDex分包处理。
3. 尽量保证加固方案不要混用，如果存在冲突，请屏蔽方案进行排查。
4. 尽量不要随意升级minSdkVersion和targetSdkVersion，升级过程，需参照Google官方文档相关指引。
5. 尽量不要随意替换签名证书，存在覆盖安装不上的问题。
6. 尽量不要上传debug版本的so进行加壳或者发布，带符号的文件存在代码泄露的风险。
7. 尽量不要选择过多的so加壳或者dll加壳（建议不超过5个），避免造成因加壳引起的性能影响。
8. 尽量保证应用中声明的application放在主dex下，如果是采用分包，则可以keep文件让它保留在主dex中。
9. so文件保护功能要求被加固的so文件含有.init_array节，若不包含则联系MTP相关人员。
10. UE4游戏如果使用lld链接器出现加固失败情况，参见lld链接器

加固效果

MTP壳对游戏资源会进行全面保护，外挂作者对游戏任意资源进行增、删、改等操作，MTP壳都会立即退出游戏。若游戏同时使用MTP反外挂和MTP加固方案，默认开启反外挂闪退保护功能，而闪退请求需要获取openid，即游戏侧调用setuserinfo接口才会生效，若游戏长时间未登录，考虑到不同游戏注册时长，在未登录的情况下预计5分钟左右闪退。apktool下载地址

下列步骤可验证加固效果：

```
1. 判断下/data/app/包名/lib目录下是否存有libtprt.so文件，存在表明该游戏已加壳
2. apktool d apkPath，使用apktool解压apk文件
3. apktool b apkPath，使用apktool重打包apk文件
4. 运行游戏，MTP壳都会立即退出游戏
```

lld链接器解决方案

UE4游戏如果使用了lld链接器，若出现加固失败则联系MTP相关人员，同时游戏侧需要修改GameActivity.java.template，将System.loadLibrary("UE4")放到static块最后一行，文件如下所示：

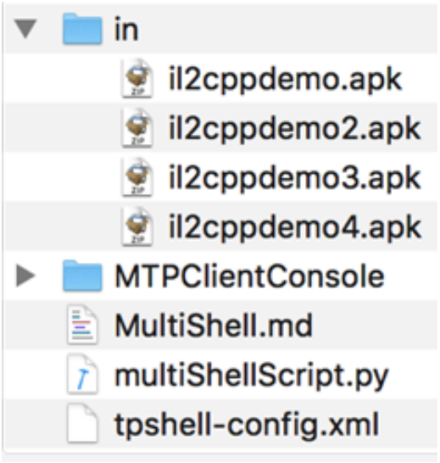
```
static{
            //$${soLoadLibrary}$$
        System.loadLibrary("c++_shared");
        System.loadLibrary("UE4");
}
```

批量处理伪代码

注意

由于加壳涉及很多io操作，考虑到机器cpu，网络带宽，建议同时开启的线程数不超过10个（根据机器性能调整），通过线程池分批次对大量的安装包进行处理

目录结构：




```python
#/usr/local/bin/python3
import os
import threading
import time
import shutil

from contextlib import contextmanager

g_temp_out = os.path.realpath("./temp")
g_threads = []

@contextmanager
def _cwd(path):
    #change workspace into multichannel
    saved_dir = os.getcwd()
    if path is not None:
        os.chdir(path)
    try:
        yield
    finally:
        os.chdir(saved_dir)

def get_timestamp():
    now = int(round(time.time()*1000))
    now = time.strftime('%Y%m%d%H%M%S',time.localtime(now/1000))
    return now

def shell_sub_main(apk_absolute_path):
    channal_name = get_timestamp() + "_" +
                   os.path.basename(apk_absolute_path).replace(".apk", "")
    workspace = os.path.join(g_temp_out, channal_name)
    if os.path.isdir(workspace):
        shutil.rmtree(workspace)
    os.makedirs(workspace)
    with _cwd(workspace):
        '''
         here run console command
         note:
         use Absolute Path in config!!!
         use Absolute Path in config!!!
         use Absolute Path in config!!!
        '''
        cmd = "mtpclient_absolute_path -d gameid" +
              "apk_absolute_path ./ cert_absolute_path -c config"
        os.system(cmd)

def shell_main():
    inDir = os.path.realpath("./in")
```

```
    files = os.listdir(inDir)
    for fi in files:
        path = "%s/%s"%(inDir, fi)
        if fi.endswith(".apk"):
            t = threading.Thread(target=shell_sub_main,args=(path,))
            g_threads.append(t)
    for t in g_threads:
        #sleep 2s here, very important
        time.sleep(2)
        t.setDaemon(True)
        t.start()
    t.join()
    print("all over")

if __name__ == '__main__':
    shell_main()
```