



作者 east520 (/users/106cd932a72f) 2016.03.23 19:45\*

写了28988字，被1898人关注，获得了6352个喜欢  
(/users/106cd932a72f)

✓ 正在关注 (/users/106cd932a72f/toggle\_like)

# 一步一步实现iOS微信自动抢红包(非越狱)

字数2219 阅读56171 评论329 喜欢731



微信红包

前言：最近笔者在研究iOS逆向工程，顺便拿微信来练手，在非越狱手机上实现了微信自动抢红包的功能。

题外话：此教程是一篇严肃的学术探讨类文章，仅仅用于学习研究，也请读者不要用于商业或其他非法途径上，笔者一概不负责哟~~

好了，接下来可以进入正题了！

## 此教程所需要的工具/文件

- yololib (<https://github.com/KJCracks/yololib>)
- class-dump ([http://stevenygard.com/projects/\\*class-dump\\*/](http://stevenygard.com/projects/*class-dump*/))
- dumpdecrypted (<https://github.com/stefanesser/dumpdecrypted>)
- iOSOpenDev (<http://iosopendev.com/download/>)
- iTools (<http://www.itools.cn/>)
- OpenSSH(Cydia)
- iFile(Cydia)
- Cycrypt(Cydia)
- Command Line Tools
- Xcode
- 苹果开发者证书或企业证书
- 一台越狱的iPhone

是的，想要实现在非越狱iPhone上达到自动抢红包的目的，工具用的可能是有点多（**工欲善其事必先利其器**^^）。不过，没关系，大家可以按照教程的步骤一步一步来执行，不清楚的步骤可以重复实验，毕竟天上不会掉馅饼嘛。

## 解密微信可执行文件(Mach-O)

---

因为从Appstore下载安装的应用都是加密过的，所以我们需要用一些工具来为下载的App解密，俗称砸壳。这样才能便于后面分析App的代码结构。

首先我们需要一台已经越狱的iPhone手机(现在市面上越狱已经很成熟，具体越狱方法这里就不介绍了)。然后进入Cydia，安装**OpenSSH**、**Cycrypt**、**iFile**(调试程序时可以方便地查看日志文件)这三款软件。

PS：笔者的手机是iPhone 6Plus，系统版本为iOS9.1。

在电脑上用iTunes上下载一个最新的微信，笔者当时下载的微信版本为6.3.13。下载完后，iTunes上会显示出已下载的app。



## iTunes

连上iPhone，用iTunes装上刚刚下载的微信应用。

打开Mac的终端，用ssh进入连上的iPhone(确保iPhone和Mac在同一个网段，笔者iPhone的IP地址为**192.168.8.54**)。OpenSSH的root密码默认为**alpine**。

```
~ ➔ ssh root@192.168.8.54
root@192.168.8.54's password:
iPhone:~ root#
```

## ssh

接下来就是需要找到微信的Bundle id了，，这里笔者有一个小技巧，我们可以把iPhone上的所有App都关掉，唯独保留微信，然后输入命令 `ps -e`

```
3923 ??      0:00.12 /System/Library/PrivateFrameworks/SyncedDefaults.framework/Support/syncdefaultsd
3926 ??      0:01.66 /var/mobile/Containers/Bundle/Application/51FDCE6F-C72B-4EDF-A8EC-277C7480135C/WeChat.app/WeChat
3928 ??      0:00.04 /System/Library/Frameworks/UIKit.framework/Support/pasteboardd
```

## 微信bundle id

这样我们就找到了微信的可执行文件Wechat的具体路径了。接下来我们需要用Cycrypt找出微信的Documents的路径，输入命令 `cycrypt -p WeChat`

```
iPhone:~ root# cycrypt -p WeChat
cy# NSSearchPathForDirectoriesInDomains(NSDocumentDirectory, NSUserDomainMask, YES)[0]
@"/var/mobile/Containers/Data/Application/B7D8AA8D-27C7-4C5A-95D8-C399FDABE086/Documents"
cy#
```

*cycrypt*

---

- 编译dumpdecrypted

先记下刚刚我们获取到的两个路径(Bundle和Documents)，这时候我们就要开始用dumpdecrypted来为微信二进制文件(WeChat)砸壳了。

确保我们从Github上下载了最新的dumpdecrypted源码，进入dumpdecrypted源码的目录，编译dumpdecrypted.dylib，命令如下：

```
~/Work/iOS-Hack/dumpdecrypted ➤ make
`xcrun --sdk iphoneos --find gcc` -Os -Wimplicit -isysroot `xcrun --sdk iphoneos --show-sdk-path` -F`xcrun --sdk iphoneos --show-sdk-path` /System/Library/Frameworks -F`xcrun --sdk iphoneos --show-sdk-path` /System/Library/PrivateFrameworks -arch armv7 -arch armv7s -arch arm64 -dynamiclib -o dumpdecrypted.dylib dumpdecrypted.o
~/Work/iOS-Hack/dumpdecrypted ➤ ls
Makefile      dumpdecrypted.c  dumpdecrypted.o
README        dumpdecrypted.dylib
```

*dumpdecrypted.dylib*

---

这样我们可以看到dumpdecrypted目录下生成了一个dumpdecrypted.dylib的文件。

- scp

拷贝dumpdecrypted.dylib到iPhone上，这里我们用到scp命令。

`scp 源文件路径 目标文件路径` 。具体如下：

```
~/Work/iOS-Hack/dumpdecrypted ➤ scp ./dumpdecrypted.dylib root@192.168.8.54:/var/mobile/Containers/Data/Application/4E996CCD-934E-4310-B7C9-490787016060/Documents/
root@192.168.8.54's password:
dumpdecrypted.dylib                                100% 193KB 192.9KB/s 00:00
```

*scp*

---

- 开始砸壳

dumpdecrypted.dylib的具体用法是：`DYLD_INSERT_LIBRARIES=/PathFrom/dumpdecrypted.dylib`  
`/PathTo`



```
Eastde-iPhone:~ root# DYLD_INSERT_LIBRARIES=/var/mobile/Containers/Data/Application/B7D8AA8D-27C7-4C5A-95D8-C399FDABE086/Documents/dumpdecrypted.dylib /var/mobile/Containers/Bundle/Application/51FDCE6F-C72B-4EDF-A8EC-277C7480135C/WeChat.app/WeChat
mach-o decryption dumper

DISCLAIMER: This tool is only meant for security research purposes, not for application crackers.

[+] detected 64bit ARM binary in memory.
[+] offset to cryptid found: @0x100010ca8(from 0x100010000) = ca8
[+] Found encrypted data at address 00004000 of length 41467904 bytes - type 1.
[+] Opening /private/var/mobile/Containers/Bundle/Application/51FDCE6F-C72B-4EDF-A8EC-277C7480135C/WeChat.app/WeChat for reading.
[+] Reading header
[+] Detecting header type
[+] Executable is a FAT image - searching for right architecture
[+] Correct arch is at offset 45826048 in the file
[+] Opening WeChat.decrypted for writing.
[+] Copying the not encrypted start of the file
[+] Dumping the decrypted data into the file
[+] Copying the not encrypted remainder of the file
[+] Setting the LC_ENCRYPTION_INFO->cryptid to 0 at offset 2bb4ca8
[+] Closing original file
[+] Closing dump file
Eastde-iPhone:~ root#
```

### *dumpdecrypted*

---

这样就代表砸壳成功了，当前目录下会生成砸壳后的文件，即WeChat.decrypted。同样用scp命令把WeChat.decrypted文件拷贝到电脑上,接下来我们要正式dump微信的可执行文件了。

## dump微信可执行文件


















---

- 从Github上下载最新的class-dump源代码，然后用Xcode编译即可生成class-dump(这里比较简单，笔者就不详细说明了)。

- 导出微信的头文件

使用class-dump命令,把刚刚砸壳后的WeChat.decrypted,导出其中的头文件。 ./class-dump

-s -S -H ./WeChat.decrypted -o ./header6.3-arm64

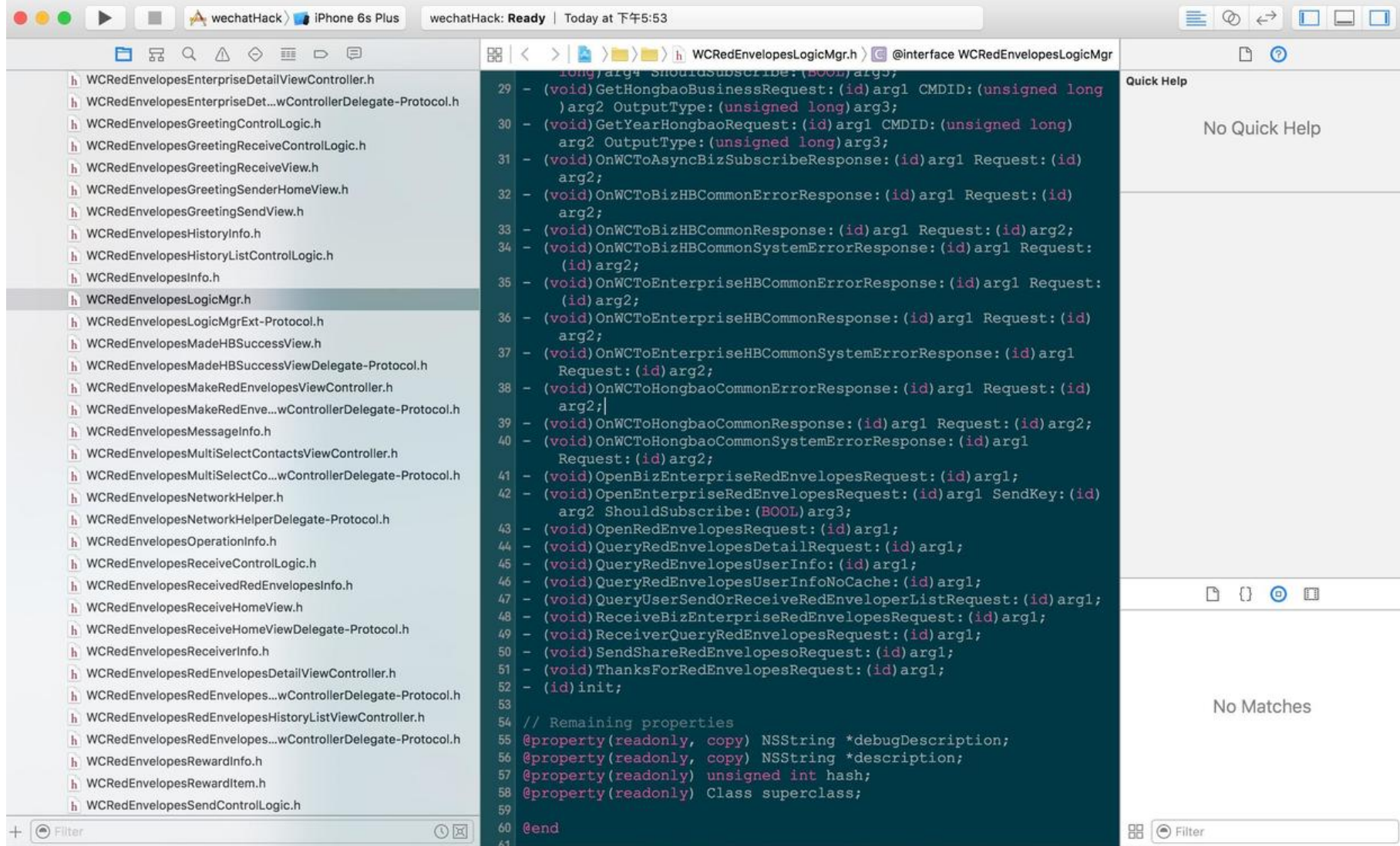
 \_\_BlueChannel.h  
 \_\_BlueDevice.h  
 \_\_CThreadWarp.h  
 \_\_DeviceInfo.h  
 \_\_ThreadWarp.h  
 \_\_WCDevice...ndDataTask.h  
 ABNewPerso...te-Protocol.h  
 ABPeoplePic...te-Protocol.h  
 ABtestCase.h  
 ABTestItem.h  
 ABtestMgr.h  
 ABtestPoint.h  
 ABtestPointPeriod.h  
 AccelerometerFilter.h  
 AcceptCardItemRequest.h  
 AcceptCardItemResponse.h  
 AcceptCardL...ppRequest.h

#### 导出的头文件

---

这里我们可以新建一个Xcode项目，把刚刚导出的头文件加到新建的项目中，这样便于查找微信的相关代码。


















## 微信的头文件

找到**CMessageMgr.h**和**WCRRedEnvelopesLogicMgr.h**这两文件，其中我们注意到有这两个方法： - (void)AsyncOnAddMsg:(id) arg1 MsgWrap:(id) arg2; ， - (void)OpenRedEnvelopesRequest:(id) arg1; 。没错，接下来我们就是要利用这两个方法来实现微信自动抢红包功能。其实现原理是，通过hook微信的新消息函数，我们判断是否为红包消息，如果是，我们就调用微信的打开红包方法。这样就能达到自动抢红包的目的了。哈哈，是不是很简单，我们一起来看看具体是怎么实现的吧。

- 新建一个dylib工程，因为Xcode默认不支持生成dylib，所以我们需要下载iOSOpenDev，安装完成后(Xcode7环境会提示安装iOSOpenDev失败，请参考iOSOpenDev安装问题(<http://www.tqcto.com/article/software/14553.html>)), 重新打开Xcode，在新建项目的选项中即可看到iOSOpenDev选项了。

Choose a template for your new project:

iOS Application Framework & Library iOSOpenDev	 Action Menu Plugin	 Activator Listener	 Assistant Extensions Extension	 CaptainHook Tweak
watchOS Application Framework & Library	 Cocoa Touch Library	 Command-line Tool	 Empty Project	 Logos Tweak
tvOS Application Framework & Library	 NotificationCent er Widget	 PreferenceLoad er Bundle	 SBSettings Toggle	 XPC Service
OS X Application Framework & Library System Plug-in	 Cocoa Touch Library This template builds a dynamic or static library that links against the Foundation and UIKit frameworks.			
Other				

Cancel

Previous

Next

iOSOpenDev

- dylib代码

选择 Cocoa Touch Library，这样我们就新建了一个 dylib 工程了，我们命名为 autoGetRedEnv。

删除 autoGetRedEnv.h 文件，修改 autoGetRedEnv.m 为 autoGetRedEnv.mm，然后在项目中加入 CaptainHook.h (<https://github.com/rpetrich/CaptainHook>)

因为微信不会主动来加载我们的 hook 代码，所以我们需要把 hook 逻辑写到构造函数中。

```
__attribute__((constructor)) static void entry()
{
    //具体hook方法
}
```

hook 微信的 AsyncOnAddMsg: MsgWrap: 方法，实现方法如下：



```

//声明CMessageMgr类
CHDeclareClass(CMessageMgr);
CHMethod(2, void, CMessageMgr, AsyncOnAddMsg, id, arg1, MsgWrap, id, arg2)
{
    //调用原来的AsyncOnAddMsg:MsgWrap:方法
    CHSuper(2, CMessageMgr, AsyncOnAddMsg, arg1, MsgWrap, arg2);
    //具体抢红包逻辑
    //...
    //调用原生的打开红包的方法
    //注意这里必须为给objc_msgSend的第三个参数声明为NSMutableDictionary,不然调用objc_msgSend时, 不会
    ((void (*)(id, SEL, NSMutableDictionary*))objc_msgSend)(logicMgr, @selector(OpenRedEnv)
}
__attribute__((constructor)) static void entry()
{
    //加载CMessageMgr类
    CHLoadLateClass(CMessageMgr);
    //hook AsyncOnAddMsg:MsgWrap:方法
    CHClassHook(2, CMessageMgr, AsyncOnAddMsg, MsgWrap);
}

```

项目的全部代码，笔者已放入Github (<https://github.com/east520/AutoGetRedEnv>)中。

完成好具体实现逻辑后，就可以顺利生成dylib了。

## 重新打包微信App

- 为微信可执行文件注入dylib

要想微信应用运行后，能执行我们的代码，首先需要微信加入我们的dylib，这里我们用到一个dylib注入神器:yololib (<https://github.com/KJCracks/yololib>)，从网上下载源代码，编译后得到yololib。

使用yololib简单的执行下面一句就可以成功完成注入。注入之前我们先把之前保存的WeChat.decrypted重命名为WeChat，即已砸完壳的可执行文件。

```
./yololib 目标可执行文件 需注入的dylib
```

注入成功后即可见到如下信息：

```
~/Work/iOS-Hack/Tools ./yololib WeChat libautoGetRedEnv.dylib
2016-03-23 18:43:51.155 yololib[64525:1993014] dylib path @executable_path/libautoGetRedEnv.dylib
2016-03-23 18:43:51.156 yololib[64525:1993014] dylib path @executable_path/libautoGetRedEnv.dylib
Reading binary: WeChat

2016-03-23 18:43:51.157 yololib[64525:1993014] FAT binary!
2016-03-23 18:43:51.157 yololib[64525:1993014] Injecting to arch 9
2016-03-23 18:43:51.157 yololib[64525:1993014] Patching mach_header..
2016-03-23 18:43:51.157 yololib[64525:1993014] Attaching dylib..

2016-03-23 18:43:51.158 yololib[64525:1993014] Injecting to arch 0
2016-03-23 18:43:51.158 yololib[64525:1993014] 64bit arch wow
2016-03-23 18:43:51.158 yololib[64525:1993014] dylib size wow 64
2016-03-23 18:43:51.158 yololib[64525:1993014] mach.ncmds 83
2016-03-23 18:43:51.158 yololib[64525:1993014] mach.ncmds 84
2016-03-23 18:43:51.158 yololib[64525:1993014] Patching mach_header..
2016-03-23 18:43:51.159 yololib[64525:1993014] Attaching dylib..

2016-03-23 18:43:51.159 yololib[64525:1993014] size 63
2016-03-23 18:43:51.159 yololib[64525:1993014] complete!
~/Work/iOS-Hack/Tools
```

## dylib注入

- 新建Entitlements.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
  <key>application-identifier</key>
  <string>123456.com.autogetredenv.demo</string>
  <key>com.apple.developer.team-identifier</key>
  <string>123456</string>
  <key>get-task-allow</key>
  <true/>
  <key>keychain-access-groups</key>
  <array>
    <string>123456.com.autogetredenv.demo</string>
  </array>
</dict>
</plist>
```

这里大家也许不清楚自己的证书Teamid及其他信息，没关系，笔者这里有一个小窍门，大家可以找到之前用开发者证书或企业证书打包过的App(例如叫Demo)，然后在终端中输入以下命令即可找到相关信息，命令如下：

```
./ldid -e ./Demo.app/demo
```

- 给微信重新签名

接下来把我们生成的 **dylib(libautoGetRedEnv.dylib)**、刚刚注入 dylib 的 **WeChat**、以及 **embedded.mobileprovision** 文件(可以在之前打包过的App中找到)拷贝到 **WeChat.app** 中。

命令格式: `codesign -f -s 证书名字 目标文件`

PS:证书名字可以在钥匙串中找到

分别用codesign命令来为微信中的相关文件签名,具体实现如下:

```
~/Work/iOS-Hack/Tools ➤ codesign -f -s "iPhone Developer: // / (8C4GZ26J43)" WeChat.app/libautoGetRedEnv.dylib
WeChat.app/libautoGetRedEnv.dylib: replacing existing signature
~/Work/iOS-Hack/Tools ➤ codesign -f -s "iPhone Developer: // / (8C4GZ26J43)" WeChat.app/Watch/WeChatWatchNative.app/PlugIns/WeChatWatchNativeExtension.appex
WeChat.app/Watch/WeChatWatchNative.app/PlugIns/WeChatWatchNativeExtension.appex: replacing existing signature
~/Work/iOS-Hack/Tools ➤ codesign -f -s "iPhone Developer: // / (8C4GZ26J43)" WeChat.app/Watch/WeChatWatchNative.app
WeChat.app/Watch/WeChatWatchNative.app: replacing existing signature
~/Work/iOS-Hack/Tools ➤ codesign -f -s "iPhone Developer: // / (8C4GZ26J43)" WeChat.app/PlugIns/WeChatShareExtensionNew.appex
WeChat.app/PlugIns/WeChatShareExtensionNew.appex: replacing existing signature
~/Work/iOS-Hack/Tools ➤ codesign -f -s "iPhone Developer: // / (8C4GZ26J43)" --entitlements Entitlements.plist WeChat.app
WeChat.app: replacing existing signature
~/Work/iOS-Hack/Tools ➤
```

重新签名

- 打包成ipa

给微信重新签名后,我们就可以用xcrun来生成ipa了,具体实现如下:

```
xcrun -sdk iphoneos PackageApplication -v WeChat.app -o ~/WeChat.ipa
```

## 安装拥有抢红包功能的微信

以上步骤如果都成功实现的话,那么真的就是万事俱备,只欠东风了~~~

我们可以使用iTools工具,来为iPhone(此iPhone Device id需加入证书中)安装改良过的微信了。



名称	版本	应用大小	文档大小	操作
 微信	6.3.13 6.3.15 ↑	134.85 MB	3.60 MB	
 壁纸	2.0	26.96 MB	4.00 KB	
 	1.0	148.00 KB	4.00 KB	
 	2.0.2	1.48 MB	4.00 KB	
 	2.0.1	1.59 MB	88.00 KB	

当前进度：

安装复制...

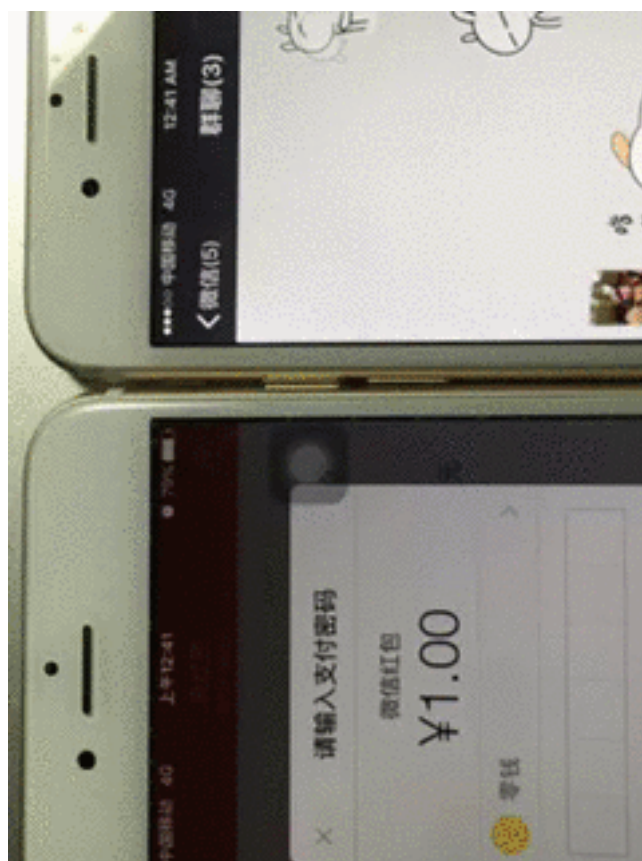
总进度：

Wechat.ipa

*iTools*

**大工告成！！**

好了，我们可以看看hook过的微信抢红包效果了~



## 自动抢红包

哈哈，是不是觉得很爽啊，"妈妈再也不用担心我抢红包了。"。大家如果有兴趣可以继续hook微信的其他函数，这样既加强了学习，又满足了自己的特(zhuang)殊(bi)需求嘛。

教程中所涉及到的工具及源代码笔者都上传到Github上。

Github地址 (<https://github.com/east520/AutoGetRedEnv>)

特别鸣谢:

- 1.iOS冰与火之歌(作者:蒸米) (<http://drops.wooyun.org/papers/12803>)
- 2.iOS 应用逆向工程 ([http://www.amazon.cn/iOS%E5%BA%94%E7%94%A8%E9%80%86%E5%90%91%E5%B7%A5%E7%A8%8B-%E6%B2%99%E6%A2%93%E7%A4%BE/dp/B00VFDVY7E/ref=sr\\_1\\_1?s=books&ie=UTF8&qid=1458733194&sr=1-1&keywords=ios%E5%BA%94%E7%94%A8%E9%80%86%E5%90%91%E5%B7%A5%E7%A8%8B](http://www.amazon.cn/iOS%E5%BA%94%E7%94%A8%E9%80%86%E5%90%91%E5%B7%A5%E7%A8%8B-%E6%B2%99%E6%A2%93%E7%A4%BE/dp/B00VFDVY7E/ref=sr_1_1?s=books&ie=UTF8&qid=1458733194&sr=1-1&keywords=ios%E5%BA%94%E7%94%A8%E9%80%86%E5%90%91%E5%B7%A5%E7%A8%8B))

- 50部高智商烧脑电影推荐(上) - 简书 (<http://www.jianshu.com/p/133d84d419fe>)  
east520 (/users/106cd932a72f) · [www.jianshu.com](http://www.jianshu.com) → (<http://www.jianshu.com/p/133d84d419fe>)
- 一步一步构建iOS持续集成:Jenkins+GitLab+蒲公英+FTP - 简书 (<http://www.jianshu.com/p/c69deb29720d>)  
east520 (/users/106cd932a72f) · [www.jianshu.com](http://www.jianshu.com) → (<http://www.jianshu.com/p/c69deb29720d>)
- 一步一步实现无线安装iOS应用(内网OTA) - 简书 (<http://www.jianshu.com/p/35ca63ec0d8e>)  
east520 (/users/106cd932a72f) · [www.jianshu.com](http://www.jianshu.com) → (<http://www.jianshu.com/p/35ca63ec0d8e>)

🔗 推荐拓展阅读 📄 举报文章    © 著作权归作者所有

如果觉得我的文章对您有用，请随意打赏。您的支持将鼓励我继续创作！


¥ 打赏支持




(/users/e9c00ea53446)

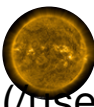
👍 喜欢

   
更多分享 ▼


 添加新评论

 我是年糕先生 (/users/cc9688452d55)  
3楼 · 2016.03.23 21:30 (/p/189afbe3b429/comments/1810217#comment-1810217)  
(/users/cc9688452d55)  
略屌


♡ 喜欢(0) 回复

 我纯洁 (/users/d778e83d24f8)  
4楼 · 2016.03.23 22:07 (/p/189afbe3b429/comments/1810625#comment-1810625)  
(/users/d778e83d24f8)  
6

♡ 喜欢(0) 回复

 姜流儿96 (/users/26987a323850)  
5楼 · 2016.03.24 10:43 (/p/189afbe3b429/comments/1814765#comment-1814765)  
(/users/26987a323850)  
666


♡ 喜欢(0) 回复


 化缘11 (/users/a85ca445bc09)  
6楼 · 2016.03.24 16:53 (/p/189afbe3b429/comments/1818460#comment-1818460)  
(/users/a85ca445bc09)  
大部分看了蒸米的教程😂😂

♡ 喜欢(0) 回复

east520 (/users/106cd932a72f): @化缘11 (/users/a85ca445bc09) 是的，站在巨人的肩膀上办事才不累😁  
2016.03.24 17:24 (/p/189afbe3b429/comments/1818709#comment-1818709)

回复


 添加新回复

 代码描绘人生 (/users/4588e4274830)  
7楼 · 2016.03.24 17:00 (/p/189afbe3b429/comments/1818503#comment-1818503)  
(/users/4588e4274830)



♡ 喜欢(0)


回复

 MarkTang (/users/4d2f03256949)  
8 楼 · 2016-03-24 17:00 (/p/189afbe3b429/comments/1818510#comment-1818510)

略屌略屌 🍊

♡ 喜欢(0)

回复

 杰米 (/users/eb00acb403aa)  
9 楼 · 2016-03-24 17:10 (/p/189afbe3b429/comments/1818588#comment-1818588)


卧槽，膜拜大神


♡ 喜欢(0)

回复

east520 (/users/106cd932a72f): @杰米 (/users/eb00acb403aa) 过奖了  
2016.03.24 23:54 (/p/189afbe3b429/comments/1822779#comment-1822779)

回复


 添加新回复

 mingjiameng (/users/d68767f43bf1)  
10 楼 · 2016-03-24 17:23 (/p/189afbe3b429/comments/1818694#comment-1818694)

niubility

♡ 喜欢(0)


回复

 Eric\_ (/users/58ac08fbf7da)  
11 楼 · 2016-03-24 18:00 (/p/189afbe3b429/comments/1818987#comment-1818987)

强大

♡ 喜欢(0)

回复

 Z先森的花小喵 (/users/38ecbde68ec8)  
12 楼 · 2016-03-24 19:03 (/p/189afbe3b429/comments/1819440#comment-1819440)

改天试试，

♡ 喜欢(0)

回复

xiaoxiaoc (/users/9f311a2edde9): @Z先森的花小喵 (/users/38ecbde68ec8) 改天以后又有改天  
2016.03.24 19:09 (/p/189afbe3b429/comments/1819507#comment-1819507)

回复


Z先森的花小喵 (/users/38ecbde68ec8): @xiaoxiaoc (/users/9f311a2edde9) 目测会，最近催活儿太紧，可是明日复明日。。。。  
2016.03.24 19:26 (/p/189afbe3b429/comments/1819697#comment-1819697)

回复

east520 (/users/106cd932a72f): @Z先森的花小喵 (/users/38ecbde68ec8) 加油，少说多做，多动手🤪  
2016.03.25 00:03 (/p/189afbe3b429/comments/1822882#comment-1822882)

回复

✎ 添加新回复

 九月未央 (/users/91e1f9ef8da4)  
12楼 · 2016.03.24 19:20 (/p/189afbe3b429/comments/1819635#comment-1819635)  
(/users/91e1f9ef8da4)  
这装逼的门槛有点高 😊


♡ 喜欢(0)

回复

east520 (/users/106cd932a72f): @九月未央 (/users/91e1f9ef8da4) 是要费点功夫  
2016.03.24 23:55 (/p/189afbe3b429/comments/1822782#comment-1822782)


回复

✎ 添加新回复

 奴良 (/users/a02909a8a93b)  
14楼 · 2016.03.24 19:31 (/p/189afbe3b429/comments/1819738#comment-1819738)  
(/users/a02909a8a93b)  
明儿试试

♡ 喜欢(0)

回复

 vvvei (/users/96b5fd3d2097)  
15楼 · 2016.03.24 20:10 (/p/189afbe3b429/comments/1820136#comment-1820136)  
(/users/96b5fd3d2097)  
这篇文章都是体力活，干货是怎么定位出是哪两个类，作者没有说，嘿嘿

♡ 喜欢(0)

回复

east520 (/users/106cd932a72f): @cntrump (/users/96b5fd3d2097) 那又涉及到另一个块了, IDA+LLDB, 这篇只讲实现方法, 不说为啥这样 🤔

2016.03.24 23:58 (/p/189afbe3b429/comments/1822812#comment-1822812)

回复

vvvei (/users/96b5fd3d2097): @east520 (/users/106cd932a72f) ida对arm64没有f5不好使了, 还是动态调戏方便。 😊

2016.03.25 03:28 (/p/189afbe3b429/comments/1823471#comment-1823471)

回复

公爵海恩庭斯 (/users/8b55fe53a3e8): @cntrump (/users/96b5fd3d2097) 赞

2016.03.25 10:49 (/p/189afbe3b429/comments/1825405#comment-1825405)

回复

✎ 添加新回复



山雨欲来风 (/users/c242e086ad21)

16 楼 · 2016.03.24 20:21 (/p/189afbe3b429/comments/1820242#comment-1820242)

这种行为了算静态劫持吧?

♡ 喜欢(0)

回复

east520 (/users/106cd932a72f): @山雨欲来风 (/users/c242e086ad21) 动态劫持, 一有新消息都能截获到

2016.03.25 00:02 (/p/189afbe3b429/comments/1822863#comment-1822863)

回复

✎ 添加新回复



Macin (/users/1f96794c21bb)

17 楼 · 2016.03.24 20:35 (/p/189afbe3b429/comments/1820369#comment-1820369)

小白照着做一遍看看

♡ 喜欢(0)

回复

加载更多 ↓ (/notes/3323649/comments?max\_id=5665832&order=asc&page=2)



写下你的评论...

发表



⌘+Return 发表

被以下专题收入，发现更多相似内容：



程序员 (/collection/NEt52a)

如果你是程序员，或者有一颗喜欢写程序的心，喜欢分享技术干货、项目经验、程序员日常囧事等等，欢迎投稿《程序员》专题。专题主编：小...  
✓ | 正在关注 (/collections/16/unsubscribe)

27614篇文章 (/collection/NEt52a) · 226421人关注



首页投稿 (/collection/bDHhpK)

玩转简书的第一步，从这个专题开始。想上首页热门榜么？好内容想被更多人看到么？来投稿  
+ | 添加关注 (/collections/47/subscribe)

116059篇文章 (/collection/bDHhpK) · 140154人关注



iOS Developer (/collection/3233d1a249ca)

分享 iOS 开发的知识，解决大家遇到的问题，讨论iOS开发的前沿，欢迎大家投稿  
+ | 添加关注 (/collections/1276/subscribe)

13399篇文章 (/collection/3233d1a249ca) · 28310人关注