目录

抓包代理利器：mitmproxy

- 最新版本：`v1.9.2`
- 更新时间：`20240714`

简介

介绍主要用于抓包领域的代理工具mitmproxy，尤其是常用的命令行版的mitmdump。先对mitmproxy概述，再介绍mitmdump的下载和安装。包括Mac和Win中如何安装和常见问题。接下来介绍如何使用mitmdump，包括核心的通用逻辑，即先电脑端启动mitmdump代理，再去移动端初始化安装mitmproxy的根证书ssl代理证书文件，其中总结了各种iOS和安卓手机在安装根证书时候的各种坑和问题及解决办法。再去介绍如何给移动端中WiFi去设置代理。总结了常见的问题，比如No module named yaml、if you can see this, traffic is not passing through mitmproxy、Cannot validate certificate hostname without SNI、ssl3_read_bytes sslv3 alert certificate unknown、-s时无法指定python版本、检测到代理显示异常、mitmdump偶尔突然无效、ssl3_get_record wrong version number、SysCallError 10054 WSAECONNRESET等等。还有其他方面，比如用代码控制mitmdump的运行、win中如何给mitmdump的python打包成exe。最后附上mitmdump、mitmproxy和mitmweb的help语法供需要时查阅。

源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

HonKit源码

- crifan/crawler_proxy_tool_mimproxy: 抓包代理利器：mitmproxy

如何使用此HonKit源码去生成发布为电子书

详见：crifan/honkit_template: demo how to use crifan honkit template and demo

在线浏览

- 抓包代理利器：mitmproxy book.crifan.org
- 抓包代理利器：mitmproxy crifan.github.io

离线下载阅读

- 抓包代理利器：mitmproxy PDF
- 抓包代理利器：mitmproxy ePub
- 抓包代理利器：mitmproxy Mobi

版权和用途说明

此电子书教程的全部内容，如无特别说明，均为本人原创。其中部分内容参考自网络，均已备注了出处。如发现有侵权，请通过邮箱联系我 `admin 艾特 crifan.com`，我会尽快删除。谢谢合作。

各种技术类教程，仅作为学习和研究使用。请勿用于任何非法用途。如有非法用途，均与本人无关。

鸣谢

感谢我的老婆**陈雪**的包容理解和悉心照料，才使得我 `crifan` 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

其他

作者的其他电子书

本人 `crifan` 还写了其他 `150+` 本电子书教程，感兴趣可移步至：

crifan/crifan_ebook_readme: Crifan的电子书的使用说明

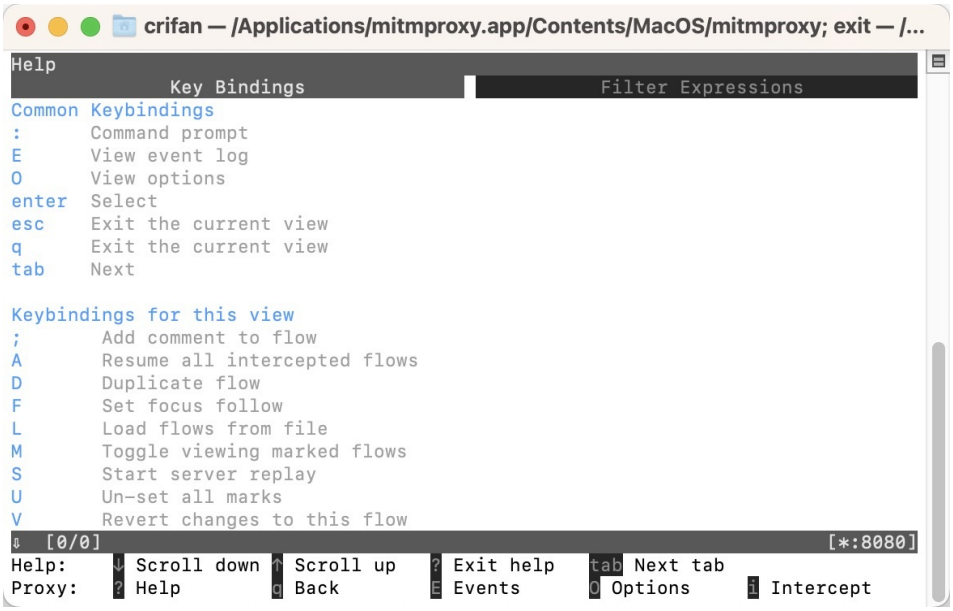
关于作者

关于作者更多介绍，详见：

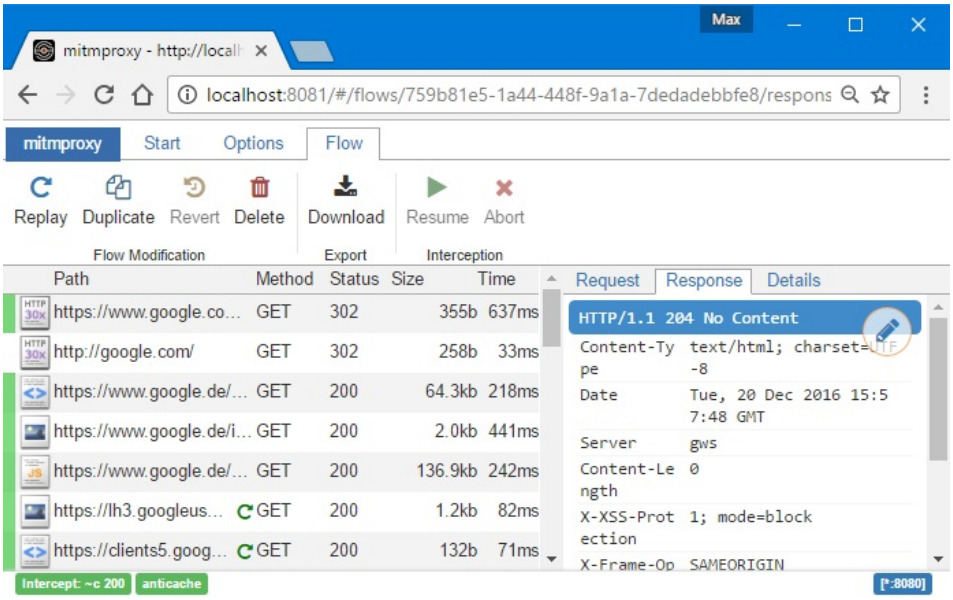
关于CrifanLi李茂 – 在路上

mitmproxy概述

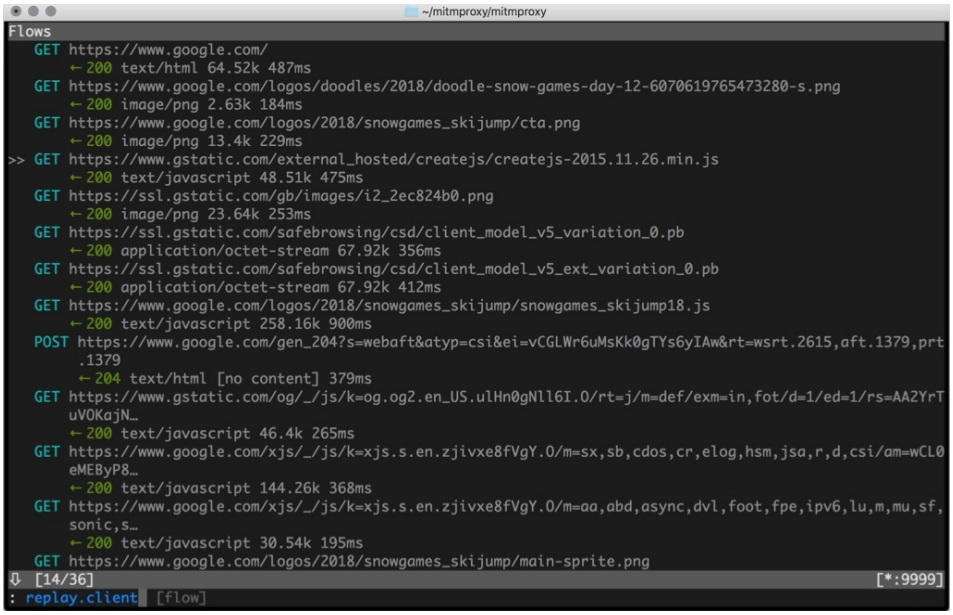
- `mitmproxy`
  - 名词解析
    - `mitmproxy` = `mitm` 的 `proxy`
      - `mitm` = `MITM` = `Man-In-The-Middle`
        - 直译：`人在中间`
          - 在中间 ->
            - 首先要确保原先网络请求能继续
              - 所以就是代理的功能：**正常转发**原有网络请求
            - 但也可以干很多事情
              - 比如
                - **记录**、**保存**网络请求
                - **拦截**（符合特定规则的）网络请求
                - （甚至）**篡改**、**伪造**成新的合法的或不合法的网络请求
        - 相关：往往也被叫做
          - `Man-In-The-Middle attack` = `中间人攻击`
  - `mitmproxy` 是一套工具的总称，包含
    - `mitmproxy`：交互式命令行工具
      - 是什么=一句话概述
        - 英文
          - mitmproxy is an interactive, SSL/TLS-capable intercepting proxy with a console interface for HTTP/1, HTTP/2, and WebSockets
        - 中文
          - mitmproxy是一个代理工具
            - 功能和特点
              - 交互式的
              - 支持https拦截
              - 支持协议：`HTTP/1`、`HTTP/2`、`WebSockets`
                - 产品形态：控制台console中显示交互界面
      - 长什么样=截图
        - Mac



- `mitmweb`：基于命令行的带UI界面
  - 可以理解为：**网页版的mitmproxy**
  - 是什么=一句话描述
    - mitmweb is a web-based interface for mitmproxy
  - 长什么样=截图
    - Win



- `mitmdump`：命令行版本
  - 是什么=一句话描述
    - mitmdump is the command-line version of mitmproxy. Think tcpdump for HTTP
  - 类比
    - `mitmdump` 之于 `mitmproxy`
      - 就像
        - `tcpdump` 之于 `HTTP`
  - 可以理解为
    - 命令行版本的 `Charles` / `Fiddler`
  - 效果截图
    - Mac



- 主要用途：实现对网页、app的抓包
- 网址
  - 官网文档
    - Introduction
      - https://docs.mitmproxy.org/stable/
  - GitHub
    - mitmproxy/mitmproxy: An interactive TLS-capable intercepting HTTP proxy for penetration testers and software developers
      - https://github.com/mitmproxy/mitmproxy

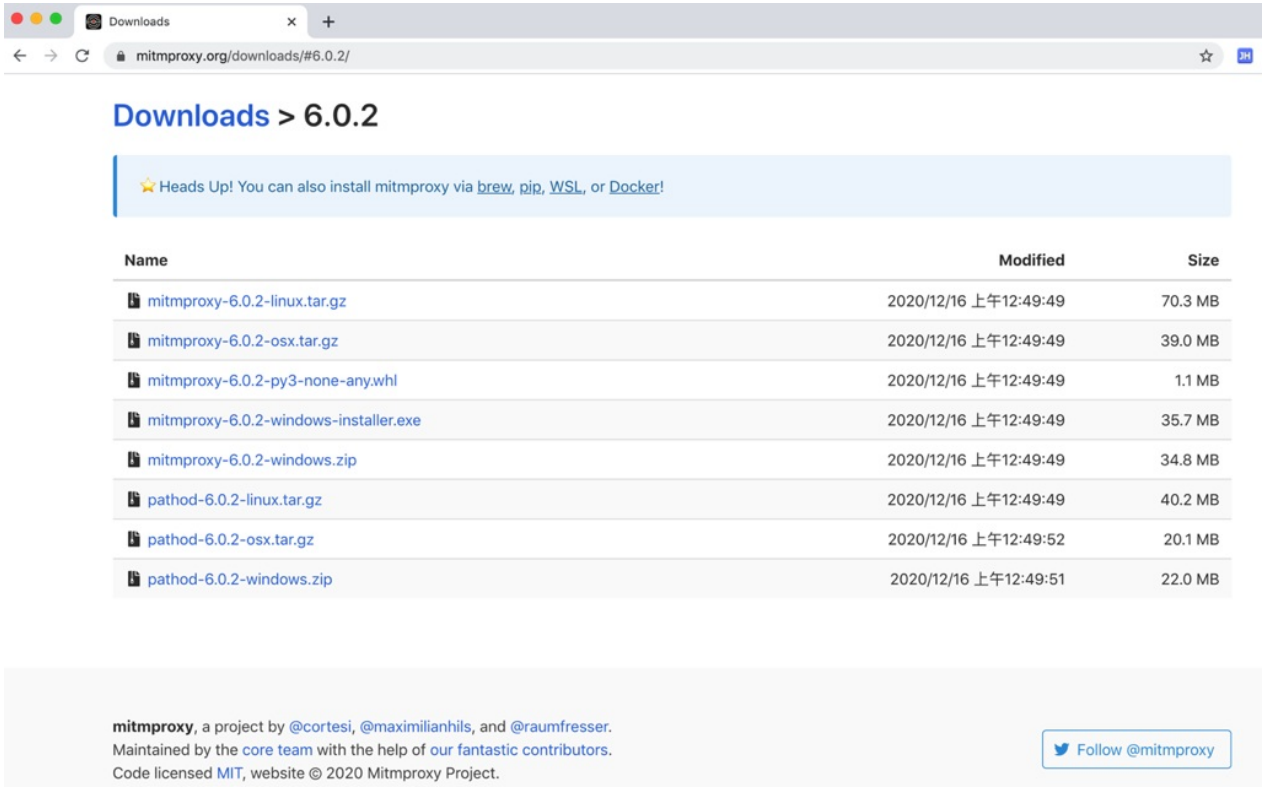
mitmproxy

下面就介绍这个，交互式命令行==基于命令行的图形界面==基于cli的GUI的：`mitmproxy`

下载mitmproxy

官网下载地址：

- Downloads
  - https://mitmproxy.org/downloads/#6.0.2/



即可下载到对应系统的二进制可执行文件：

- Mac
  - https://snapshots.mitmproxy.org/6.0.2/mitmproxy-6.0.2-osx.tar.gz
- Win
  - https://snapshots.mitmproxy.org/6.0.2/mitmproxy-6.0.2-windows-installer.exe
  - https://snapshots.mitmproxy.org/6.0.2/mitmproxy-6.0.2-windows.zip
- Linux
  - https://snapshots.mitmproxy.org/6.0.2/mitmproxy-6.0.2-linux.tar.gz

下载后，解压，即可得到：可执行文件

举例说明：

Mac

此处的Mac M2 Max，对应的最新版本是：

https://downloads.mitmproxy.org/10.3.1/mitmproxy-10.3.1-macos-arm64.tar.gz

下载后，解压得到：

`mitmproxy.app`

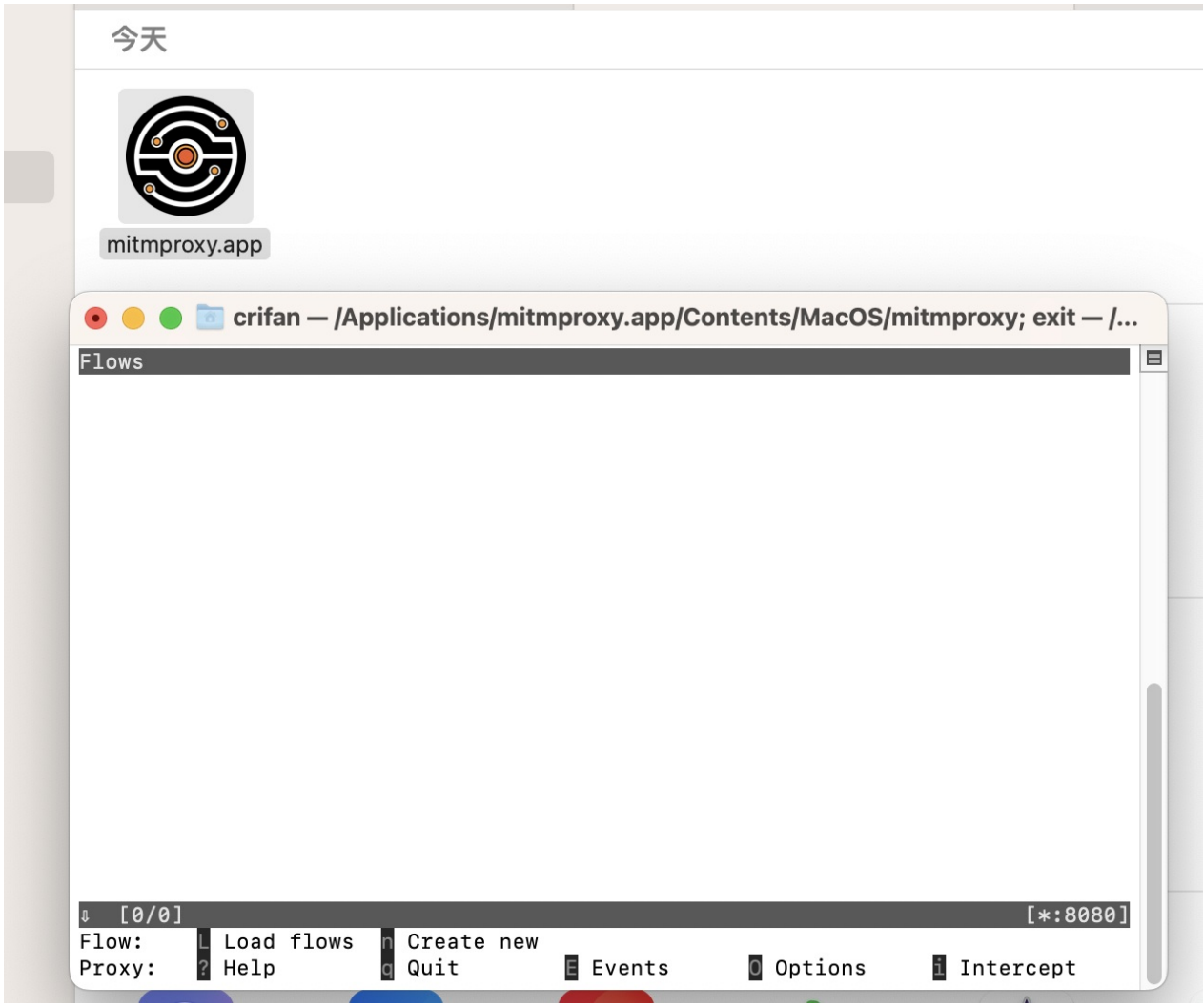
移动到 `应用程序` 中：

使用mitmproxy

Mac

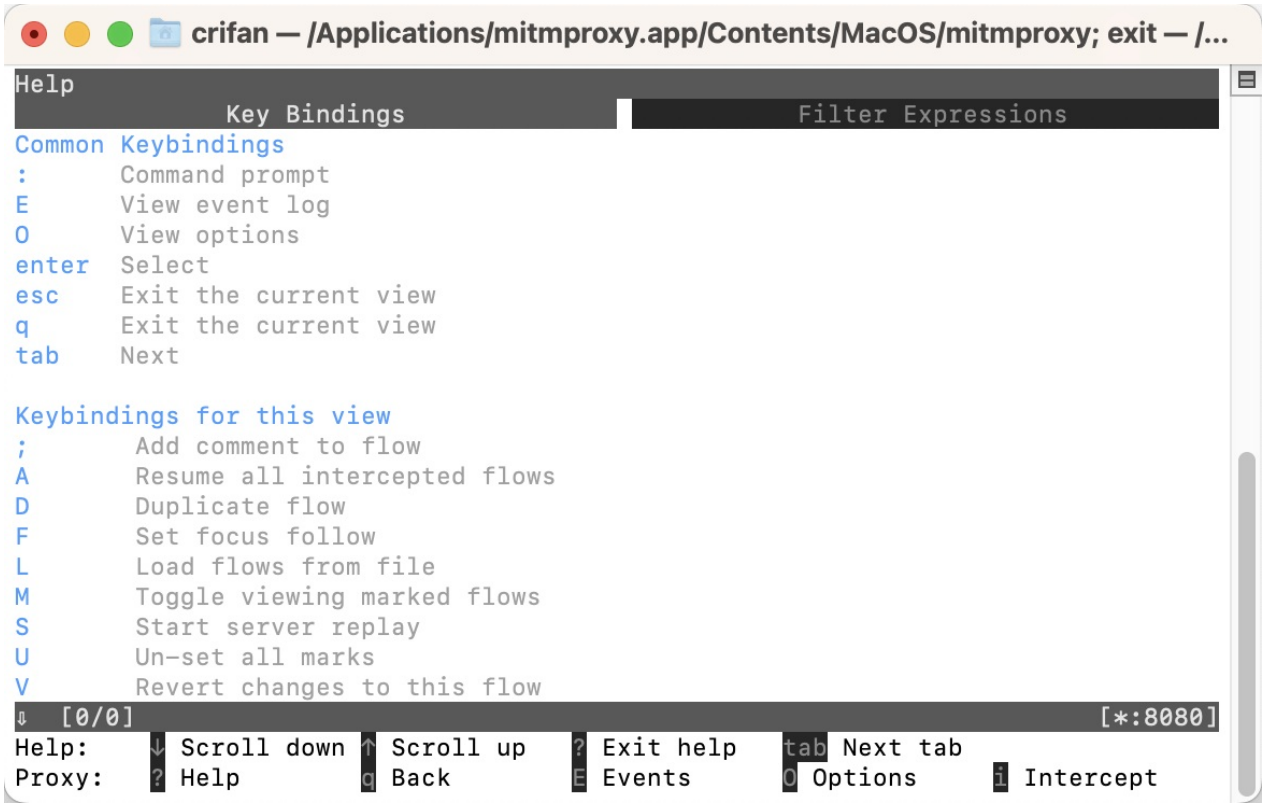
双击已移动到应用程序目录中的 `mitmproxy.app` ，即可启动运行：

打开的是：基于命令行的GUI图形界面：

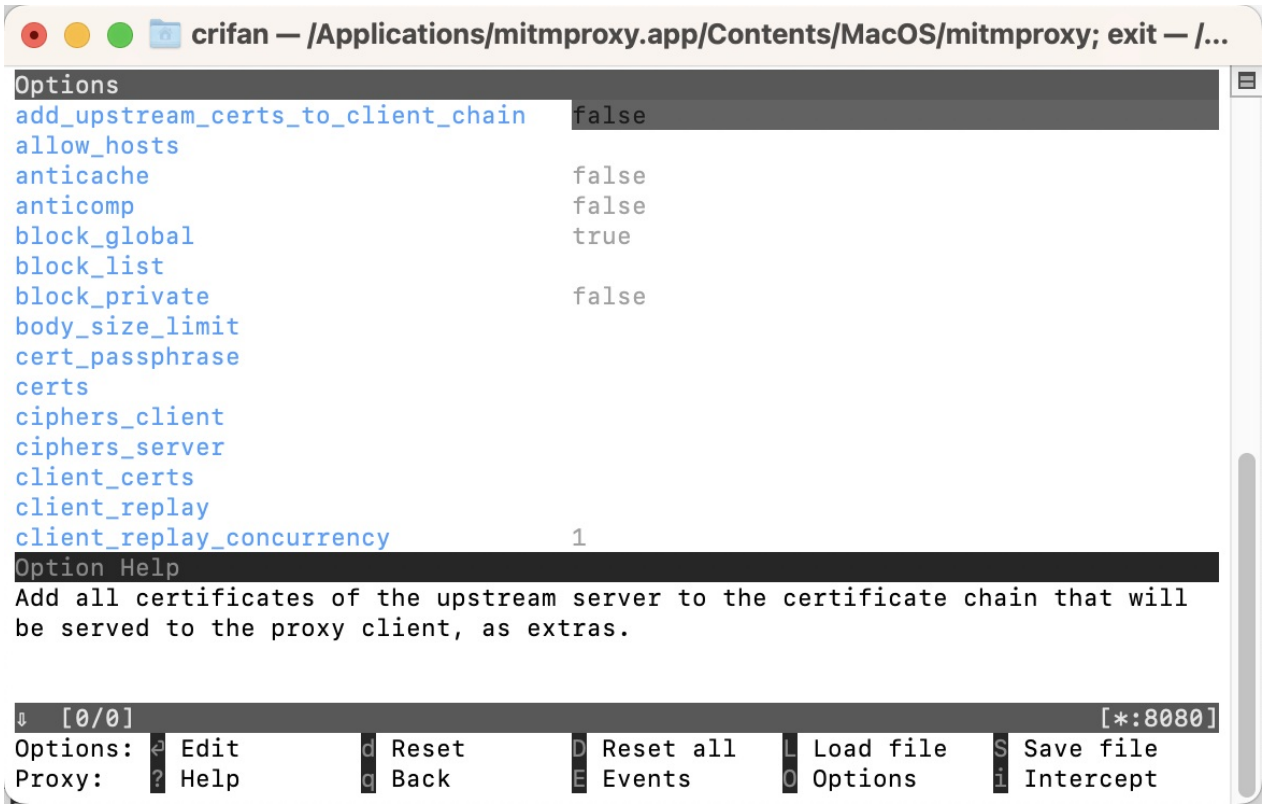


比如：

输入 `?` 可以打开help帮助界面：



输入 `O` = `Options` 可以打开设置选项界面：



mitmdump

`mitmdump` 是 `mitmproxy` 的命令行版本。

`mitmproxy` 和 `mitmdump` 的用法和逻辑基本一致。

此处主要介绍 `mitmdump` 的使用。

安装mitmproxy

下载到二进制文件后，（Mac、Win、Linux等）各个系统中，即可正常安装。

Mac

通过brew安装mitmproxy

Mac中也可以直接用 `brew` 去安装：

```
brew install mitmproxy
```

安装mitmproxy后，就自带了命令行版本的 `mitmdump`

用pip安装Python版本

如果后续需要用到 `mitmdump` 的 `-s` 去加载的 `.py` 的python脚本，那么此处就应该先要通过安装Python中的pip去（给Python环境中）安装mitmproxy

```
pip install mitmproxy
```

安装其他依赖的Python库

举例：

如果后续 `mitmdump` 用到 `-s` 去加载的 `.py` 的python脚本中，用到了 `pyaml` 的话，则记得要先用 `pip` 安装 `pyyaml`：

```
pip instal pyyaml
```

Win

安装遇到的问题

之前在win10中安装mitmproxy，遇到过2个问题：

build _openssl.c error C2065 X509_V_FLAG_CB_ISSUER_CHECK undeclared identifier

```
creating build\temp.win-amd64-3.8\Release\build\temp.win-amd64-3.8\Release
  C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\BIN\x86_amd64\cl.exe /c /nologo /Ox /W3 /GL /DNDEBUG /MD "-Ic:\program files\python38\include" "-Ic:\program files\python38\include" "-IC:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\INCLUDE" "-IC:\Program Files (x86)\Windows Kits\10\include\10.0.10240.0\ucrt" "-IC:\Program Files (x86)\Windows Kits\8.1\include\shared" "-IC:\Program Files (x86)\Windows Kits\8.1\include\um" "-IC:\Program Files (x86)\Windows Kits\8.1\include\winrt" /Tcbuild\temp.win-amd64-3.8\Release\_openssl.c /Fobuild\temp.win-amd64-3.8\Release\build\temp.win-amd64-3.8\Release\_openssl.obj
  _openssl.c
  build\temp.win-amd64-3.8\Release\_openssl.c(1369): warning C4098: 'Cryptography_HMAC_CTX_free': 'void' function returning a value
  build\temp.win-amd64-3.8\Release\_openssl.c(11095): error C2065: 'X509_V_FLAG_CB_ISSUER_CHECK': undeclared identifier
  build\temp.win-amd64-3.8\Release\_openssl.c(11096): error C2065: 'X509_V_FLAG_CB_ISSUER_CHECK': undeclared identifier
  build\temp.win-amd64-3.8\Release\_openssl.c(12429): warning C4013: 'ASN1_STRING_data' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(12429): warning C4047: 'return': 'unsigned char *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(12452): warning C4047: '=': 'unsigned char *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(13565): warning C4090: 'return': different 'const' qualifiers
  build\temp.win-amd64-3.8\Release\_openssl.c(13575): warning C4090: '=': different 'const' qualifiers
  build\temp.win-amd64-3.8\Release\_openssl.c(13589): warning C4090: 'return': different 'const' qualifiers
  build\temp.win-amd64-3.8\Release\_openssl.c(13599): warning C4090: '=': different 'const' qualifiers
  build\temp.win-amd64-3.8\Release\_openssl.c(16441): warning C4013: 'CRYPTO_cleanup_all_ex_data' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(16465): warning C4013: 'CRYPTO_get_locking_callback' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(16465): warning C4047: 'return': 'void (__cdecl *)(int,int,const char *,int)' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(16475): warning C4047: '=': 'void (__cdecl *)(int,int,const char *,int)' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(16575): warning C4013: 'CRYPTO_num_locks' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(16599): warning C4013: 'CRYPTO_set_locking_callback' undefined; assuming extern returning int
```

```
  build\temp.win-amd64-3.8\Release\_openssl.c(19290): warning C4013: 'DTLSv1_client_met
hod' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(19290): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(19300): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(19350): warning C4013: 'DTLSv1_method' un
defined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(19350): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(19360): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(19374): warning C4013: 'DTLSv1_server_met
hod' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(19374): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(19384): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(22275): warning C4013: 'ENGINE_cleanup' u
ndefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(23380): warning C4013: 'ENGINE_load_dynam
ic' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(23404): warning C4013: 'ENGINE_load_opens
sl' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(25957): warning C4013: 'EVP_CIPHER_CTX_cl
eanup' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(26094): warning C4013: 'EVP_CIPHER_CTX_in
it' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(32153): warning C4013: 'OBJ_cleanup' unde
fined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(34129): warning C4090: 'return': differen
t 'const' qualifiers
  build\temp.win-amd64-3.8\Release\_openssl.c(34152): warning C4090: '=': different 'co
nst' qualifiers
  build\temp.win-amd64-3.8\Release\_openssl.c(34609): warning C4013: 'OPENSSL_config' u
ndefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(34709): warning C4013: 'OPENSSL_no_config'
 undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(34793): warning C4013: 'OpenSSL_add_all_a
lgorithms' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(38288): warning C4013: 'RAND_cleanup' und
efined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(46480): warning C4090: 'function': differ
ent 'const' qualifiers
  build\temp.win-amd64-3.8\Release\_openssl.c(46520): warning C4090: 'function': differ
ent 'const' qualifiers
  build\temp.win-amd64-3.8\Release\_openssl.c(46713): warning C4013: 'SSL_library_init'
 undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(46773): warning C4013: 'SSL_load_error_st
rings' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(49146): warning C4013: 'TLSv1_1_client_me
thod' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(49146): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49156): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
```

```
  build\temp.win-amd64-3.8\Release\_openssl.c(49170): warning C4013: 'TLSv1_1_method' u
ndefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(49170): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49180): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49194): warning C4013: 'TLSv1_1_server_me
thod' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(49194): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49204): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49218): warning C4013: 'TLSv1_2_client_me
thod' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(49218): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49228): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49242): warning C4013: 'TLSv1_2_method' u
ndefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(49242): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49252): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49266): warning C4013: 'TLSv1_2_server_me
thod' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(49266): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49276): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49290): warning C4013: 'TLSv1_client_meth
od' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(49290): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49300): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49314): warning C4013: 'TLSv1_method' und
efined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(49314): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49324): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49338): warning C4013: 'TLSv1_server_meth
od' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(49338): warning C4047: 'return': 'const S
SL_METHOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(49348): warning C4047: '=': 'const SSL_ME
THOD *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(50421): warning C4013: 'X509_CRL_get_last
Update' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(50421): warning C4047: 'return': 'ASN1_OC
TET_STRING *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(50444): warning C4047: '=': 'ASN1_OCTET_S
TRING *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(50457): warning C4013: 'X509_CRL_get_next
Update' undefined; assuming extern returning int
```

```
  build\temp.win-amd64-3.8\Release\_openssl.c(50457): warning C4047: 'return': 'ASN1_OC
TET_STRING *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(50480): warning C4047: '=': 'ASN1_OCTET_S
TRING *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(50659): warning C4013: 'X509_CRL_set_last
Update' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(50712): warning C4013: 'X509_CRL_set_next
Update' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(53826): warning C4013: 'X509_STORE_CTX_ge
t_chain' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(53826): warning C4047: 'return': 'Cryptog
raphy_STACK_OF_X509 *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(53849): warning C4047: '=': 'Cryptography
_STACK_OF_X509 *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(54363): warning C4013: 'X509_STORE_CTX_se
t_chain' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(54620): warning C4013: 'X509_STORE_CTX_tr
usted_stack' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(57001): warning C4013: 'X509_get_notAfter'
 undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(57001): warning C4047: 'return': 'ASN1_OC
TET_STRING *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(57024): warning C4047: '=': 'ASN1_OCTET_S
TRING *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(57037): warning C4013: 'X509_get_notBefor
e' undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(57037): warning C4047: 'return': 'ASN1_OC
TET_STRING *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(57060): warning C4047: '=': 'ASN1_OCTET_S
TRING *' differs in levels of indirection from 'int'
  build\temp.win-amd64-3.8\Release\_openssl.c(57500): warning C4013: 'X509_set_notAfter'
 undefined; assuming extern returning int
  build\temp.win-amd64-3.8\Release\_openssl.c(57553): warning C4013: 'X509_set_notBefor
e' undefined; assuming extern returning int
  error: command 'C:\\Program Files (x86)\\Microsoft Visual Studio 14.0\\VC\\BIN\\x86_a
md64\\cl.exe' failed with exit status 2
  ----------------------------------------
  ERROR: Failed building wheel for cryptography
  Running setup.py clean for cryptography
Failed to build cryptography
ERROR: Could not build wheels for cryptography which use PEP 517 and cannot be installe
d directly
```

经过一番折腾，但最后是没解决。

具体过程详见：

- 【未解决】windows中pip安装mitmproxy报错： build _openssl.c error C2065 X509_V_FLAG_CB_ISSUER_CHECK undeclared identifier

ERROR: Could not build wheels for cryptography which use PEP 517 and cannot be installed directly

```
> pip install mitmproxy
......
Collecting pycparser
  Using cached https://files.pythonhosted.org/packages/68/9e/49196946aee219aead1290e00d
1e7fdeab8567783e83e1b9ab5585e6206a/pycparser-2.19.tar.gz
Building wheels for collected packages: cryptography
  Building wheel for cryptography (PEP 517) ... error
  ERROR: Command errored out with exit status 1:
   command: 'c:\program files\python38\python.exe' 'c:\program files\python38\lib\site-
packages\pip\_vendor\pep517\_in_process.py' build_wheel 'C:\Users\xxx~1\AppData\Local\T
emp\tmp38qbzrlq'
       cwd: C:\Users\xxx\AppData\Local\Temp\pip-install-x4h_xxr8\cryptography
  Complete output (112 lines):
  running bdist_wheel
  running build
  running build_py
  creating build
  creating build\lib.win-amd64-3.8
  creating build\lib.win-amd64-3.8\cryptography
  copying src\cryptography\exceptions.py -> build\lib.win-amd64-3.8\cryptography
  copying src\cryptography\fernet.py -> build\lib.win-amd64-3.8\cryptography
  copying src\cryptography\utils.py -> build\lib.win-amd64-3.8\cryptography
  copying src\cryptography\__about__.py -> build\lib.win-amd64-3.8\cryptography
  copying src\cryptography\__init__.py -> build\lib.win-amd64-3.8\cryptography
  creating build\lib.win-amd64-3.8\cryptography\hazmat
  copying src\cryptography\hazmat\_oid.py -> build\lib.win-amd64-3.8\cryptography\hazma
t
  copying src\cryptography\hazmat\__init__.py -> build\lib.win-amd64-3.8\cryptography\h
azmat
  creating build\lib.win-amd64-3.8\cryptography\x509
  copying src\cryptography\x509\base.py -> build\lib.win-amd64-3.8\cryptography\x509
  copying src\cryptography\x509\certificate_transparency.py -> build\lib.win-amd64-3.8\
cryptography\x509
  copying src\cryptography\x509\extensions.py -> build\lib.win-amd64-3.8\cryptography\x
509
  copying src\cryptography\x509\general_name.py -> build\lib.win-amd64-3.8\cryptography\
x509
  copying src\cryptography\x509\name.py -> build\lib.win-amd64-3.8\cryptography\x509
  copying src\cryptography\x509\ocsp.py -> build\lib.win-amd64-3.8\cryptography\x509
  copying src\cryptography\x509\oid.py -> build\lib.win-amd64-3.8\cryptography\x509
  copying src\cryptography\x509\__init__.py -> build\lib.win-amd64-3.8\cryptography\x50
9
  creating build\lib.win-amd64-3.8\cryptography\hazmat\backends
  copying src\cryptography\hazmat\backends\interfaces.py -> build\lib.win-amd64-3.8\cry
ptography\hazmat\backends
  copying src\cryptography\hazmat\backends\__init__.py -> build\lib.win-amd64-3.8\crypt
ography\hazmat\backends
  creating build\lib.win-amd64-3.8\cryptography\hazmat\bindings
  copying src\cryptography\hazmat\bindings\__init__.py -> build\lib.win-amd64-3.8\crypt
ography\hazmat\bindings
  creating build\lib.win-amd64-3.8\cryptography\hazmat\primitives
  copying src\cryptography\hazmat\primitives\cmac.py -> build\lib.win-amd64-3.8\cryptog
raphy\hazmat\primitives
  copying src\cryptography\hazmat\primitives\constant_time.py -> build\lib.win-amd64-3.8
\cryptography\hazmat\primitives
```

```
  copying src\cryptography\hazmat\primitives\hashes.py -> build\lib.win-amd64-3.8\crypt
ography\hazmat\primitives
  copying src\cryptography\hazmat\primitives\hmac.py -> build\lib.win-amd64-3.8\cryptog
raphy\hazmat\primitives
  copying src\cryptography\hazmat\primitives\keywrap.py -> build\lib.win-amd64-3.8\cryp
tography\hazmat\primitives
  copying src\cryptography\hazmat\primitives\mac.py -> build\lib.win-amd64-3.8\cryptogr
aphy\hazmat\primitives
  copying src\cryptography\hazmat\primitives\padding.py -> build\lib.win-amd64-3.8\cryp
tography\hazmat\primitives
  copying src\cryptography\hazmat\primitives\serialization.py -> build\lib.win-amd64-3.8
\cryptography\hazmat\primitives
  copying src\cryptography\hazmat\primitives\__init__.py -> build\lib.win-amd64-3.8\cry
ptography\hazmat\primitives
  creating build\lib.win-amd64-3.8\cryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\aead.py -> build\lib.win-amd64-3.8\c
ryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\backend.py -> build\lib.win-amd64-3.8
\cryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\ciphers.py -> build\lib.win-amd64-3.8
\cryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\cmac.py -> build\lib.win-amd64-3.8\c
ryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\decode_asn1.py -> build\lib.win-amd6
4-3.8\cryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\dh.py -> build\lib.win-amd64-3.8\cry
ptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\dsa.py -> build\lib.win-amd64-3.8\cr
yptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\ec.py -> build\lib.win-amd64-3.8\cry
ptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\encode_asn1.py -> build\lib.win-amd6
4-3.8\cryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\hashes.py -> build\lib.win-amd64-3.8\
cryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\hmac.py -> build\lib.win-amd64-3.8\c
ryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\ocsp.py -> build\lib.win-amd64-3.8\c
ryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\rsa.py -> build\lib.win-amd64-3.8\cr
yptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\utils.py -> build\lib.win-amd64-3.8\
cryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\x25519.py -> build\lib.win-amd64-3.8\
cryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\x509.py -> build\lib.win-amd64-3.8\c
ryptography\hazmat\backends\openssl
  copying src\cryptography\hazmat\backends\openssl\__init__.py -> build\lib.win-amd64-3
.8\cryptography\hazmat\backends\openssl
  creating build\lib.win-amd64-3.8\cryptography\hazmat\bindings\openssl
  copying src\cryptography\hazmat\bindings\openssl\binding.py -> build\lib.win-amd64-3.8
\cryptography\hazmat\bindings\openssl
  copying src\cryptography\hazmat\bindings\openssl\_conditional.py -> build\lib.win-amd
64-3.8\cryptography\hazmat\bindings\openssl
  copying src\cryptography\hazmat\bindings\openssl\__init__.py -> build\lib.win-amd64-3
.8\cryptography\hazmat\bindings\openssl
```

```
  creating build\lib.win-amd64-3.8\cryptography\hazmat\primitives\asymmetric
  copying src\cryptography\hazmat\primitives\asymmetric\dh.py -> build\lib.win-amd64-3.8
\cryptography\hazmat\primitives\asymmetric
  copying src\cryptography\hazmat\primitives\asymmetric\dsa.py -> build\lib.win-amd64-3
.8\cryptography\hazmat\primitives\asymmetric
  copying src\cryptography\hazmat\primitives\asymmetric\ec.py -> build\lib.win-amd64-3.8
\cryptography\hazmat\primitives\asymmetric
  copying src\cryptography\hazmat\primitives\asymmetric\padding.py -> build\lib.win-amd
64-3.8\cryptography\hazmat\primitives\asymmetric
  copying src\cryptography\hazmat\primitives\asymmetric\rsa.py -> build\lib.win-amd64-3
.8\cryptography\hazmat\primitives\asymmetric
  copying src\cryptography\hazmat\primitives\asymmetric\utils.py -> build\lib.win-amd64
-3.8\cryptography\hazmat\primitives\asymmetric
  copying src\cryptography\hazmat\primitives\asymmetric\x25519.py -> build\lib.win-amd6
4-3.8\cryptography\hazmat\primitives\asymmetric
  copying src\cryptography\hazmat\primitives\asymmetric\__init__.py -> build\lib.win-am
d64-3.8\cryptography\hazmat\primitives\asymmetric
  creating build\lib.win-amd64-3.8\cryptography\hazmat\primitives\ciphers
  copying src\cryptography\hazmat\primitives\ciphers\aead.py -> build\lib.win-amd64-3.8\
cryptography\hazmat\primitives\ciphers
  copying src\cryptography\hazmat\primitives\ciphers\algorithms.py -> build\lib.win-amd
64-3.8\cryptography\hazmat\primitives\ciphers
  copying src\cryptography\hazmat\primitives\ciphers\base.py -> build\lib.win-amd64-3.8\
cryptography\hazmat\primitives\ciphers
  copying src\cryptography\hazmat\primitives\ciphers\modes.py -> build\lib.win-amd64-3.8
\cryptography\hazmat\primitives\ciphers
  copying src\cryptography\hazmat\primitives\ciphers\__init__.py -> build\lib.win-amd64
-3.8\cryptography\hazmat\primitives\ciphers
  creating build\lib.win-amd64-3.8\cryptography\hazmat\primitives\kdf
  copying src\cryptography\hazmat\primitives\kdf\concatkdf.py -> build\lib.win-amd64-3.8
\cryptography\hazmat\primitives\kdf
  copying src\cryptography\hazmat\primitives\kdf\hkdf.py -> build\lib.win-amd64-3.8\cry
ptography\hazmat\primitives\kdf
  copying src\cryptography\hazmat\primitives\kdf\kbkdf.py -> build\lib.win-amd64-3.8\cr
yptography\hazmat\primitives\kdf
  copying src\cryptography\hazmat\primitives\kdf\pbkdf2.py -> build\lib.win-amd64-3.8\c
ryptography\hazmat\primitives\kdf
  copying src\cryptography\hazmat\primitives\kdf\scrypt.py -> build\lib.win-amd64-3.8\c
ryptography\hazmat\primitives\kdf
  copying src\cryptography\hazmat\primitives\kdf\x963kdf.py -> build\lib.win-amd64-3.8\
cryptography\hazmat\primitives\kdf
  copying src\cryptography\hazmat\primitives\kdf\__init__.py -> build\lib.win-amd64-3.8\
cryptography\hazmat\primitives\kdf
  creating build\lib.win-amd64-3.8\cryptography\hazmat\primitives\twofactor
  copying src\cryptography\hazmat\primitives\twofactor\hotp.py -> build\lib.win-amd64-3
.8\cryptography\hazmat\primitives\twofactor
  copying src\cryptography\hazmat\primitives\twofactor\totp.py -> build\lib.win-amd64-3
.8\cryptography\hazmat\primitives\twofactor
  copying src\cryptography\hazmat\primitives\twofactor\utils.py -> build\lib.win-amd64-
3.8\cryptography\hazmat\primitives\twofactor
  copying src\cryptography\hazmat\primitives\twofactor\__init__.py -> build\lib.win-amd
64-3.8\cryptography\hazmat\primitives\twofactor
  running egg_info
  writing src\cryptography.egg-info\PKG-INFO
  writing dependency_links to src\cryptography.egg-info\dependency_links.txt
  writing requirements to src\cryptography.egg-info\requires.txt
```

```
  writing top-level names to src\cryptography.egg-info\top_level.txt
  reading manifest file 'src\cryptography.egg-info\SOURCES.txt'
  reading manifest template 'MANIFEST.in'
  no previously-included directories found matching 'docs\_build'
  warning: no previously-included files matching '*' found under directory 'vectors'
  writing manifest file 'src\cryptography.egg-info\SOURCES.txt'
  running build_ext
  generating cffi module 'build\\temp.win-amd64-3.8\\Release\\_padding.c'
  creating build\temp.win-amd64-3.8
  creating build\temp.win-amd64-3.8\Release
  generating cffi module 'build\\temp.win-amd64-3.8\\Release\\_constant_time.c'
  generating cffi module 'build\\temp.win-amd64-3.8\\Release\\_openssl.c'
  building '_openssl' extension
  creating build\temp.win-amd64-3.8\Release\build
  creating build\temp.win-amd64-3.8\Release\build\temp.win-amd64-3.8
  creating build\temp.win-amd64-3.8\Release\build\temp.win-amd64-3.8\Release
  C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\BIN\x86_amd64\cl.exe /c /nologo /Ox /W3 /GL /DNDEBUG /MD "-Ic:\program files\python38\include" "-Ic:\program files\python38\include" "-IC:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\INCLUDE" "-IC:\Program Files (x86)\Windows Kits\10\include\10.0.10240.0\ucrt" "-IC:\Program Files (x86)\Windows Kits\8.1\include\shared" "-IC:\Program Files (x86)\Windows Kits\8.1\include\um" "-IC:\Program Files (x86)\Windows Kits\8.1\include\winrt" /Tcbuild\temp.win-amd64-3.8\Release\_openssl.c /Fobuild\temp.win-amd64-3.8\Release\build\temp.win-amd64-3.8\Release\_openssl.obj
  _openssl.c
  build\temp.win-amd64-3.8\Release\_openssl.c(498): fatal error C1083: Cannot open include file: 'openssl/opensslv.h': No such file or directory
  error: command 'C:\\Program Files (x86)\\Microsoft Visual Studio 14.0\\VC\\BIN\\x86_amd64\\cl.exe' failed with exit status 2
  ----------------------------------------
  ERROR: Failed building wheel for cryptography
  Running setup.py clean for cryptography
Failed to build cryptography
ERROR: Could not build wheels for cryptography which use PEP 517 and cannot be installed directly
```

也试了：

```
pip install cryptography
```

但问题依旧。

以及：

```
python -m pip install --no-use-pep517 mitmproxy
```

但报其他错误：

```
    copying src\cryptography\hazmat\primitives\twofactor\__init__.py -> build\lib.win-amd64-3.8\cryptography\hazmat\primitives\twofactor
    running egg_info
    writing src\cryptography.egg-info\PKG-INFO
```

```
    writing dependency_links to src\cryptography.egg-info\dependency_links.txt
    writing requirements to src\cryptography.egg-info\requires.txt
    writing top-level names to src\cryptography.egg-info\top_level.txt
    reading manifest file 'src\cryptography.egg-info\SOURCES.txt'
    reading manifest template 'MANIFEST.in'
    no previously-included directories found matching 'docs\_build'
    warning: no previously-included files matching '*' found under directory 'vectors'
    writing manifest file 'src\cryptography.egg-info\SOURCES.txt'
    running build_ext
    generating cffi module 'build\\temp.win-amd64-3.8\\Release\\_padding.c'
    creating build\temp.win-amd64-3.8
    creating build\temp.win-amd64-3.8\Release
    generating cffi module 'build\\temp.win-amd64-3.8\\Release\\_constant_time.c'
    generating cffi module 'build\\temp.win-amd64-3.8\\Release\\_openssl.c'
    building '_openssl' extension
    creating build\temp.win-amd64-3.8\Release\build
    creating build\temp.win-amd64-3.8\Release\build\temp.win-amd64-3.8
    creating build\temp.win-amd64-3.8\Release\build\temp.win-amd64-3.8\Release
    C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\BIN\x86_amd64\cl.exe /c /nol
ogo /Ox /W3 /GL /DNDEBUG /MD "-IC:\Program Files\Python38\include" "-IC:\Program Files\
Python38\include" "-IC:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\INCLUDE" "-
IC:\Program Files (x86)\Windows Kits\10\include\10.0.10240.0\ucrt" "-IC:\Program Files
(x86)\Windows Kits\8.1\include\shared" "-IC:\Program Files (x86)\Windows Kits\8.1\inclu
de\um" "-IC:\Program Files (x86)\Windows Kits\8.1\include\winrt" /Tcbuild\temp.win-amd6
4-3.8\Release\_openssl.c /Fobuild\temp.win-amd64-3.8\Release\build\temp.win-amd64-3.8\R
elease\_openssl.obj
    _openssl.c
    build\temp.win-amd64-3.8\Release\_openssl.c(498): fatal error C1083: Cannot open in
clude file: 'openssl/opensslv.h': No such file or directory
    error: command 'C:\\Program Files (x86)\\Microsoft Visual Studio 14.0\\VC\\BIN\\x86
_amd64\\cl.exe' failed with exit status 2
    ----------------------------------------
  Rolling back uninstall of cryptography
  Moving to c:\program files\python38\lib\site-packages\cryptography-2.8.dist-info\
   from C:\Program Files\Python38\Lib\site-packages\~ryptography-2.8.dist-info
  Moving to c:\program files\python38\lib\site-packages\cryptography\
   from C:\Program Files\Python38\Lib\site-packages\~ryptography
ERROR: Command errored out with exit status 1: 'C:\Program Files\Python38\python.exe' -u
 -c 'import sys, setuptools, tokenize; sys.argv[0] = '"'"'C:\\Users\\xxx\\AppData\\Local
\\Temp\\pip-install-wkntcchc\\cryptography\\setup.py'"'"'; __file__='"'"'C:\\Users\\xxx\
\AppData\\Local\\Temp\\pip-install-wkntcchc\\cryptography\\setup.py'"'"';f=getattr(toke
nize, '"'"'open'"'"', open)(__file__);code=f.read().replace('"'"'\r\n'"'"', '"'"'\n'"'"
');f.close();exec(compile(code, __file__, '"'"'exec'"'"'))' install --record 'C:\Users\
xxx\AppData\Local\Temp\pip-record-dqgnm_3i\install-record.txt' --single-version-externa
lly-managed --compile Check the logs for full command output.
```

以及其他折腾。

但最后是没解决。

具体过程详见：

- 【未解决】windows中pip install mitmproxy失败：ERROR Could not build wheels for cryptography which use PEP 517 and cannot be installed directly

如何使用

在（Mac、Win等）PC端安装了mitmproxy后，自带mitmdump。

此处介绍如何去使用mitmdump。

通用逻辑

核心的通用逻辑是：

- 电脑端
  - 启动 `mitmdump` 代理
- 移动端
  - 给WiFi设置（PC端的mitmdump的）代理
  - （初始化，只需第一次）安装 `mitmproxy` 的**根证书**

即可正常使用代理，实现给移动端抓包等功能。

电脑端启动mitmdump代理

比如，PC端运行对应命令：

```
mitmdump -k -p 8081 -s middleware/Save1.py
```

即可。

接下来分不同平台详细介绍具体细节。

Mac

接着介绍，如何在 `Mac` 中使用 `mitmdump`

举例：

`Mac` 中终端去运行：

```
mitmdump -k -p 8081 -s saveUrl.py
```

启动mitmdump的代理

然后给手机端加上此处Mac的mitmdump的代理

即可实现：脚本 `saveUrl.py` 把手机端发出的所有的 `url` = `请求` = `链接地址` （还可以根据自己需要做一定过滤处理后再）保存起来（比如保存到一个文件中），供后续使用。

说明

先要手动创建一个： `output` 文件夹，其中新建一个空文件 `SavedUrl.txt` ，供后续保存url到其中。

此处的saveUrl.py是个python脚本

具体内容：

```
# Function: using mitmproxy to save url
# Author: Crifan Li
# Date: 20240711

import re
import os

class Saver:
    def __init__(self):
      self.Allurls = set()
      self.DataFilePath = self.initOutputFile()

    def initOutputFile(self):
      saveFile = os.path.join("output", "SavedUrl.txt")
      return saveFile

    def get_ContentType(self, headers):
      ContentType = "None"
      patten = "b'Accept', b'(.*?)'"
      result = re.search(patten, headers)
      if result:
        ContentType = result.group(1)
        ContentType = ContentType.split(",")[0]
      return ContentType if not "*" in ContentType else "None"

    def request(self, flow):
```

```
        url = flow.request.url
        ContentType = self.get_ContentType(str(flow.request.headers))
        if not url in self.Allurls:
          self.Allurls.add(url)
          print(url)
          with open(self.DataFilePath, "a", encoding="utf-8") as f:
            f.write(url + "|" + ContentType)
            f.write('\n')

addons = [Saver()]
```

若想要后台运行，则后面加 `&`

```
mitmdump -k -p 8081 -s saveUrl.py &
```

移动端给WiFi设置代理

之后再去给移动端的WiFi设置（PC端的mitmdump的）代理（信息）。

- 概述
  - 找到电脑端（Win/Mac等）中的WiFi地址（比如 `192.168.1.10` ）
  - 然后去手机移动端（Android、iPhone等），给WiFi设置代理地址（比如 `192.168.1.10` ）和端口（之前电脑端启动mitmproxy所设置的端口，一般常用 `8081` ），代理地址为电脑端的WiFi地址
- 细节详见
  - 如何添加代理 移动端 · 网络中转站：代理技术

此处简单举例如下：

Android

Google Pixel 5


zhizhiwuwu003
手动
浏览器会使用 HTTP 代理，但其他应用可能不会使用。
代理主机名
192.168.1.10
代理端口
8081
对以下网址不使用代理
example.com,mycomp.test.com,loc
取消
保存

iOS

iPhone

移动端安装mitmproxy根证书

通用逻辑

即给移动端手机中安装 `mitmproxy` 的

`SSL代理证书` = `ssl证书` = `根证书` = `root CA`

核心逻辑：

- 手机中浏览器中打开 http://mitm.it
- 然后根据提示去下载（ `.pem` / `.crt` / `.cer` 等）证书（文件）
- 点击安装证书文件

iOS

此处整理iOS的iPhone手机中，安装mitmproxy的根证书的详细情况：

- iOS
  - iPhone
    - 详见
      - 【已解决】给iPhone添加mitmproxy的mitmdump代理用于保存抓包链接到文件
      - 【已解决】iPhone8P中安装mitmproxy的CA的ssl证书

Android

此处整理安卓手机中，安装mitmproxy的根证书，对于不同手机的详细情况：

概述：

- 前提
  - Mac中的已运行了mitmdump的代理
    - `mitmdump -k -p 8081 -s saveUrl.py`
- 步骤
  - Pixel5中用Chrome浏览器打开链接：
    - http://mitm.it
  - 下载得到证书文件：`mitmproxy-ca-cert.cer`
  - 然后去安装证书文件：
    - 设置-》安全-》更多安全设置-》加密与凭证-》安装证书-》CA证书-》点选mitmproxy-ca-cert.cer去安装-》正常会提示：已安装CA证书

详见：

- Android
  - Google
    - Pixel5
      - 【已解决】Mac M2 Max中初始化mitmproxy安卓手机Pixel5的抓包环境
      - 【已解决】安卓手机Pixel5中安装mitmproxy的根证书
      - 【已解决】安卓手机Pixel5中用证书安装程序安装证书报错：必须在“设置”中安装这个由null提供的证书
  - 华为
    - 荣耀
      - 【记录】给安卓手机中安装mitmproxy代理的SSL证书
      - 【记录】给自动抓包工具的安卓手机设置mitmproxy代理用于能抓包到链接地址
  - 小米
    - 小米9
      - 相关
        - 【已解决】安卓手机小米9中安装mitmproxy的SSL代理证书
          - 【无需解决】小米9中WLAN或WAPI证书中找不到mitmproxy的SSL的pem证书文件
          - 【无法解决】小米9中用ES文件管理器安装pem证书
    - 红米Note8Pro
      - 问题
        - 用微信或小米浏览器无法下载pem证书文件
      - 解决办法：
        - 换QQ浏览器就可以正常下载pem证书文件 mitmproxy-ca-cert.pem
          - 细节
            - 不能用：
              - 微信
                - 点击Android无反应

                - 小米浏览器
                    - 点击Android，弹框下载：perm.crt
                        - 而不是希望的：mitmproxy-ca-cert.pem
                        - 关键是：始终无法下载成功
                - 只能用：QQ浏览器
                    - 点击Android，可以弹框下载：mitmproxy-ca-cert.pem
                        - 是我们希望的pem证书
                        - 也可以正常（瞬间）下载完毕
        - 详见
            - 【无法解决】红米Note8Pro中用微信或小米浏览器下载mitmproxy的SSL代理证书
            - 【已解决】红米Note8Pro中用QQ浏览器下载mitmproxy的Android的SSL代理证书
        - 相关
            - 【已解决】红米Note8Pro中安装mitmproxy的SSL代理证书
    - 红米10X
        - 问题：下载证书失败
            - 自带小米浏览器
                - 可弹框下载pem.crt，但下载失败
            - QQ浏览器
                - 可弹框下载mitmproxy-ca-cert.pem，但下载失败
                - 偶然甚至会提示：
                    - if you can see this, traffic is not passing through mitmproxy
            - UC浏览器
                - 可弹框下载mitmproxy-ca-cert.pem，但下载失败
                - 偶然甚至会提示：
                    - if you can see this, traffic is not passing through mitmproxy
        - 解决办法：
            - 试了多次，最后终于：
                - UC浏览器
                    - 可弹框并成功下载mitmproxy-ca-cert.pem
        - 详见：
            - 【已解决】红米10X安卓手机中无法下载mitmproxy的证书文件
- Vivo
    - iQOO U1x
        - 用QQ浏览器无法下载pem文件，提示下载失败
            - 解决办法：换Vivo的内置浏览器，即可下载 mitmproxy-ca-cert.pem
        - 直接点击pem证书文件，无法安装：未找到证书文件
            - 问题现象
                - QQ浏览器下载到mitmproxy-ca-cert.pem，直接点击提示：找不到对应程序打开该文件
                - 更多安全设置-》从手机存储和SD卡安装，点击提示：未找到证书文件
                - 从文件管理中点击已下载的mitmproxy-ca-cert.pem，选 证书安装程序，也提示：未找到证书文件
            - 原因：Vivo不支持pem证书文件，只支持crt证书文件
            - 解决办法：把文件pem后缀改为crt
                - 点击即可正常安装
        - 详见：

- 【已解决】给安卓手机ViVo的iQOO U1x下载和安装mitmproxy的SSL代理证书
- 【已解决】安卓手机Vivo的iQOO U1x中手动安装mitmproxy-ca-cert.pem证书文件
- 【已解决】安卓手机Vivo的iQOO U1x中点击安装mitmproxy的pem证书报错：未找到证书文件
- 【未解决】给安卓手机Vivo的iQOO U1x初始化mitmdump的代理环境

常见问题

ModuleNotFoundError No module named

现象

Mac中用brew安装了mitmproxy，然后去运行：

```
mitmdump -p 8081 -s middleware/Save1.py
```

但是报错：

- 现象1

  ```
  ModuleNotFoundError: No module named 'yaml'
  ```

- 现象2

  ```
  ➜  mitmproxy mitmdump -k -p 8081 -s saveUrl.py
  [16:21:46.972] Loading script saveUrl.py
  [16:21:46.973] error in script saveUrl.py
  Traceback (most recent call last):
    File "saveUrl.py", line 10, in <module>
      import tldextract
  ModuleNotFoundError: No module named 'tldextract'␣
  Note that mitmproxy's binaries include their own Python environment. If your addon
  requires the installation of additional dependencies, please install mitmproxy from
   PyPI (https://docs.mitmproxy.org/stable/overview-installation/#installation-from-t
  he-python-package-index-pypi).
  ```

原因

- 概述：mitmdump（即mitmproxy）中的Python环境中没有安装该Python库
- 详解
  - Mac中通过 `brew` 安装的 `mitmproxy` ，会调用自己内部安装的python（此处是 `3.7.5` ）
  - 而不是Mac中自己python（ `2.7` 或 `3.8` ）
    - 注：此时 `Mac` 中所有版本的 `Python` 中都安装过 `yaml` 了
  - 而 `mitmproxy` 中python，没有安装过 `yaml` ，所以上述脚本会报错。

解决办法

解决办法1

不要用brew安装，而是用系统中的python的pip去安装mitmproxy

```
pip install mitmproxy
```

注：系统中的python是,此处是用的3.8，用pyenv设置全局为3.8

另外此处2.7的python中，pip安装mitmproxy会失败。

之后即可正常调用

```
mitmdump -p 8081 -s middleware/Save1.py
```

其中python解析器用的是此处系统的python了，因此可以正常找到（系统中python中已安装的）yaml，而不会报错了。

具体细节详见：

- 【基本解决】Mac中mitmdump运行命令报错：in script py No module named yaml

解决办法2

如果上述办法不生效，则：

根本原因

mitmdump（即mitmproxy）内部自带了一个自己的Python环境

即：mitmdump（即mitmproxy）中的Python，和系统的Python，不是同一个

根本解决办法

给mitmproxy中能识别第三方Python库==给mitmproxy中安装第三方Python库

具体步骤

可以用官网推荐的pipx

（1）安装pipx

```
brew install pipx
pipx ensurepath
sudo pipx ensurepath --global
```

（2）用pipx安装mitmproxy

```
pipx install mitmproxy
```

（3）给mitmproxy中安装（插入）第三方Python库

```
mitmproxy pipx inject mitmproxy tldextract
```

（4）【可选，但很重要】确保此时你的终端=环境变量中，第一个找到的=实际生效的mitmproxy，是pipx所安装的

此处的意思是

前面通过pipx安装的mitmproxy后

```
pipx install mitmproxy
```

会提示你实际生效的位置，mitmproxy所在的PATH是哪个

此处是提示：

```
➜  mitmproxy pipx install mitmproxy
⚠  Note: mitmdump was already on your PATH at /usr/local/bin/mitmdump
⚠  Note: mitmproxy was already on your PATH at /usr/local/bin/mitmproxy
⚠  Note: mitmweb was already on your PATH at /usr/local/bin/mitmweb
  installed package mitmproxy 10.3.1, installed using Python 3.12.4
  These apps are now globally available
    - mitmdump
    - mitmproxy
    - mitmweb
⚠  Note: '/Users/crifan/.local/bin' is not on your PATH environment variable. These ap
ps will not be globally accessible until your PATH is updated. Run `pipx
    ensurepath` to automatically add it, or manually modify your PATH in your shell's c
onfig file (e.g. ~/.bashrc).
done
```

中的

`/Users/crifan/.local/bin`

所以你后续要确保：

后续终端命令行中，所找到的mitmproxy，是我们希望的：

用pipx安装的，插入安装了第三方Python库的）mitmproxy

而实现这个目的的典型方式是：

- 把此处对应的路径 `/Users/crifan/.local/bin` 加到环境变量PATH中去
- 且确保在前面（如果你本身PATH中其他的在前面的路径已经包含了，另外别的方式安装的mitmproxy）
  - 就像我此处：之前已经通过brew和pip安装了mitmproxy了，对应别的路径（ `/usr/local/bin/` 、 `/opt/homebrew/bin/` ）就会包含mitmproxy

此处具体处理方式就是：

编辑启动脚本（此处是：~/.zshrc）

```
vim ~/.zshrc
```

把 `/Users/crifan/.local/bin` 加到PATH的最开始

```
# Created by `pipx` on 2024-07-14 08:54:02
#export PATH="$PATH:/Users/crifan/.local/bin"
export PATH="/Users/crifan/.local/bin:$PATH"
```

[可选]再去让当前终端生效

```
source ~/.zshrc
```

详见：

Installation (mitmproxy.org)

后记：

然后再去查看对应的mitmdump，可以看到，有多个路径，且第一个是我们希望的版本：

```
➜  mitmproxy where mitmdump
/Users/crifan/.local/bin/mitmdump
/usr/local/bin/mitmdump
/opt/homebrew/bin/mitmdump
/usr/local/bin/mitmdump
/Users/crifan/.local/bin/mitmdump
/Users/crifan/.local/bin/mitmdump
```

mitmdump即可正常运行：

```
mitmdump -k -p 8081 -s saveUrl.py
```

而不报错了

if you can see this, traffic is not passing through mitmproxy

- **现象**

手机中浏览器打开 http://mitm.it 后，看到页面提示：

```
if you can see this, traffic is not passing through mitmproxy
```

- **原因**

需要你手机中WiFi加上PC端的mitmproxy（mitmdump）的代理后，打开 http://mitm.it 后才能正常显示页面

- **解决办法**
  - PC端（Mac）中启动mitmproxy的代理
    - 举例
      - `mitmdump -k -p 8081 -s middleware/Save1.py`
  - 然后再给手机端的当前WiFi中加上对应的mitmdump的代理

细节详见：

【已解决】红米Note8Pro中去下载mitmproxy证书提示：if you can see this, traffic is not passing through mitmproxy

Cannot validate certificate hostname without SNI

此处通过：

```
mitmdump -k -p 8081 -s middleware/Save1.py
```

加上了

- -k == --ssl-insecure
  - Do not verify upstream server SSL/TLS certificates

从而规避了：

```
TlsException('Cannot validate certificate hostname without SNI
```

的问题。

- 细节详见
  - 【已解决】mitmproxy代理报错：Cannot establish TLS with 443 sni None TlsException Cannot validate certificate hostname without SNI
  - 【未解决】安卓抓包mitmproxy报错：TlsException SSL handshake Error routines ssl3_get_record wrong version number

sslv3 alert certificate unknown

mitmdump访问部分url出错：

比如：

某次报错：

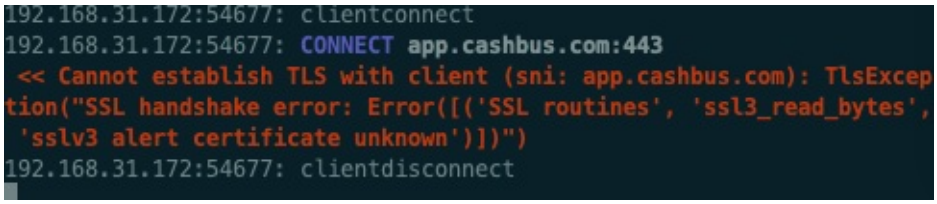
```
192.168.31.177:50670: CONNECT 6ib5h.com:443
<< Cannot establish TLS with client (sni: 6ib5h.com): TlsException("SSL handshake error
: Error([('SSL routines', 'ssl3_read_bytes', 'sslv3 alert certificate unknown')])")
```

和

抓包安卓app `现金巴士` 时：

对应mitmdump的log：

```
<< Cannot establish TLS with client (sni: app.cashbus.com): TlsException("SSL handshake
 error: Error([('SSL routines', 'ssl3_read_bytes', 'sslv3 alert certificate unknown')])"
)
```



- **原因**

app内部做了certificate pinning 证书固定 的技术

app内部给证书做了指纹，只允许来自服务器的证书，匹配后才认为是合法的有效的，否则就拒绝

即拒绝那些指纹不匹配的证书

- **如何解决**
  - 分2种情况：
    - 用 `tls_passthrough.py` 实现部分解决
    - 无法解决

用 `tls_passthrough.py` 实现部分解决

借用别人的脚本：

- tls_passthrough.py

去：

- 要么直接利用：
  - `mitmproxy -s tls_passthrough.py`
- 要么整理到自己的脚本中：
  - `mitmdump -p 8081 -s Save1.py`

其中：`Save1.py`

```python
# _*_ coding: utf-8 _*_

import json
import re
import os
import sys
print("sys.executable=%s" % sys.executable)

class Saver:

    def __init__(self):
        '''

    def request(self, flow):
```

```
        curReq = flow.request
    url = curReq.url
        headers = curReq.headers
        print("url=%s, headers=%s" % (url, headers))
    # do what you want
    # eg: save something to some file

addons = [Saver()]

"""
This inline script allows conditional TLS Interception based
on a user-defined strategy.
Example:
    > mitmdump -s tls_passthrough.py
    1. curl --proxy http://localhost:8080 https://example.com --insecure
    // works - we'll also see the contents in mitmproxy
    2. curl --proxy http://localhost:8080 https://example.com --insecure
    // still works - we'll also see the contents in mitmproxy
    3. curl --proxy http://localhost:8080 https://example.com
    // fails with a certificate error, which we will also see in mitmproxy
    4. curl --proxy http://localhost:8080 https://example.com
    // works again, but mitmproxy does not intercept and we do *not* see the contents
Authors: Maximilian Hils, Matthew Tuusberg
"""
import collections
import random

from enum import Enum

import mitmproxy
from mitmproxy import ctx
from mitmproxy.exceptions import TlsProtocolException
from mitmproxy.proxy.protocol import TlsLayer, RawTCPLayer

class InterceptionResult(Enum):
    success = True
    failure = False
    skipped = None

class _TlsStrategy:
    """
    Abstract base class for interception strategies.
    """

    def __init__(self):
        # A server_address -> interception results mapping
        self.history = collections.defaultdict(lambda: collections.deque(maxlen=200))

    def should_intercept(self, server_address):
        """
        Returns:
            True, if we should attempt to intercept the connection.
            False, if we want to employ pass-through instead.
        """
        raise NotImplementedError()
```

```
    def record_success(self, server_address):
        self.history[server_address].append(InterceptionResult.success)

    def record_failure(self, server_address):
        self.history[server_address].append(InterceptionResult.failure)

    def record_skipped(self, server_address):
        self.history[server_address].append(InterceptionResult.skipped)

class ConservativeStrategy(_TlsStrategy):
    """
    Conservative Interception Strategy - only intercept if there haven't been any faile
d attempts
    in the history.
    """

    def should_intercept(self, server_address):
        if InterceptionResult.failure in self.history[server_address]:
            return False
        return True

class ProbabilisticStrategy(_TlsStrategy):
    """
    Fixed probability that we intercept a given connection.
    """

    def __init__(self, p):
        self.p = p
        super(ProbabilisticStrategy, self).__init__()

    def should_intercept(self, server_address):
        return random.uniform(0, 1) < self.p

class TlsFeedback(TlsLayer):
    """
    Monkey-patch _establish_tls_with_client to get feedback if TLS could be established
    successfully on the client connection (which may fail due to cert pinning).
    """

    def _establish_tls_with_client(self):
        server_address = self.server_conn.address

        try:
            super(TlsFeedback, self)._establish_tls_with_client()
        except TlsProtocolException as e:
            tls_strategy.record_failure(server_address)
            raise e
        else:
            tls_strategy.record_success(server_address)

# inline script hooks below.

tls_strategy = None

def load(l):
    l.add_option(
```

```
        "tlsstrat", int, 0, "TLS passthrough strategy (0-100)",
    )

def configure(updated):
    global tls_strategy
    if ctx.options.tlsstrat > 0:
        tls_strategy = ProbabilisticStrategy(float(ctx.options.tlsstrat) / 100.0)
    else:
        tls_strategy = ConservativeStrategy()

def next_layer(next_layer):
    """
    This hook does the actual magic - if the next layer is planned to be a TLS layer,
    we check if we want to enter pass-through mode instead.
    """
    if isinstance(next_layer, TlsLayer) and next_layer._client_tls:
        server_address = next_layer.server_conn.address

        if tls_strategy.should_intercept(server_address):
            # We try to intercept.
            # Monkey-Patch the layer to get feedback from the TLSLayer if interception
worked.
            next_layer.__class__ = TlsFeedback
        else:
            # We don't intercept - reply with a pass-through layer and add a "skipped"
entry.
            mitmproxy.ctx.log("TLS passthrough for %s" % repr(next_layer.server_conn.ad
dress), "info")
            next_layer_replacement = RawTCPLayer(next_layer.ctx, ignore=True)
            next_layer.reply.send(next_layer_replacement)
            tls_strategy.record_skipped(server_address)
```

- **效果**：至少不报错了

mitmproxy的log会显示相关的 `TLS passthrough` ：

```
TLS passthrough for ('app.cashbus.com', 443)
```

其他（https的？）资源（图片等）类的文件可以正常加载，页面可以显示（图片）等内容了：

- 细节详见
  - 【已解决】提取自动抓包工具中的mitmdump自动保存代理抓包出来的url链接保存到文件
  - 【已解决】mitmproxy代理抓包安卓app数据访问出错：Cannot establish TLS with client sni TlsException

无法解决

- **彻底的解决办法**：修改app的逻辑和规则，允许你（的非法）的证书。
  - 很明显：是别人的app，自己无法修改。所以此处实际上无解
    - 除非你能破解app，重新编译和运行破解后的app，把证书的限制去掉。

另外：此处被测app是一个安卓游戏app，也没有时间去折腾破解app

也没必要：因为最终方案是希望支持无限多的安卓游戏app，所以一个个破解，也不现实不可行。

总之：无解，且放弃

- 相关

    - 安卓破解 Certificate pinning
        - 作者提到了一些关于逆向工程安卓app方面的资料
            - 需要给app打包，用于跳过证书验证，换成自己证书
        - 相关资料：
            - Android Security: SSL Pinning. Using SSL in an Android app is easy… | by Matthew Dolan | Medium
            - Bypassing Certificate Pinning on Android for fun and profit | by Felipe Lima | Medium
            - ->
            - Bypassing SSL Pinning on Android via Reverse Engineering.pdf
                - https://dl.packetstormsecurity.net/papers/general/android-sslpinning.pdf
- 细节详见
    - 【无法解决】安卓游戏加了代理后支付页面时mitmdump报错：TlsException SSL handshake error Error SSL routines ssl3_read_bytes sslv3 alert certificate unknown

`-s` 时无法指定python版本

此处

```
mitmdump -s xxx.py
```

中，无法指定加载 `xxx.py` 脚本时，所用的 `Python的版本`

-> 会导致，导入一些python库时，即使你Python环境已安装了该库（比如 `pyyaml` ），仍会报错

-> 因为其只用调用（mitmdump所）内置的Python的版本，无法换成当前（Mac）系统中的某个版本的Pyton

- 细节详见
  - 【无法解决】Mac中mitmdump通过-s加载python脚本时指定Python版本
  - 【已解决】mac中Python2和Python3都已安装了yaml但mitmdump -s加载python脚本中导入yaml还是报错
  - 【已解决】Mac中让mitmdump解析python脚本不用自己内置Python而是用系统Python
  - 【已解决】Mac中运行mitmdump再次报错：Failed to import yaml
  - 【已解决】Mac中mitmdump运行命令报错：in script py No module named yaml

检测到代理显示异常

有些app、网页等，技术做的相对先进，能自动检测是否有代理：

- 如果检测到有代理
  - 会出现警告
  - 甚至不显示内容
  - 或者显示内容异常
  - 等等
- 对应解决办法：去掉代理
  - 比如切换到没有代理的WiFi

下面总结一下这些异常情况：

安全警告 该网站的安全证书存在问题

安卓手机中安装了mitmproxy代理后，部分app页面（首次打开）会弹框提示证书问题

比如 安卓中的微信的某些页面，第一次访问某些其他网站时，会提示证书问题：

- 【已解决】自动抓包工具适配iOS：安全警告弹框提示该网站的安全证书存在问题

- vivo应用市场登录后，偶尔也有同样弹框

- 。

点击了继续后，后续就不会再提示。

类似问题：

当你Wifi代理有变动，比如：

- 去掉WiFi的mitmproxy的代理后，重新加上
- 换了一个WiFi，重新加上代理

等等情况，则会被视为第一次使用代理，第一次打开页面时，就仍会出现上述弹框提示。同理，点击继续后，之后不会再提示。

显示空白页面或者只显示部分内容

另外，有些安卓游戏，加了代理后，会导致游戏中和安全相关的，尤其是支付相关页面，会无法正常显示内容。

比如空白页面：

或者是 只能显示部分内容：

底部支付方式没显示：

后来经过多次点击，偶尔才能完整显示内容：

网络异常，请检查网络设置

iOS的app 斑马AI课 会提示：

> 网络异常，请检查网络设置
>
> 请检查您当前的网络环境，如果其他App可以正常使用，请到设置-斑马AI课-无线数据中允许斑马AI课访问网络。检查后，点击重试按钮。

无法连接服务器，请退出重试

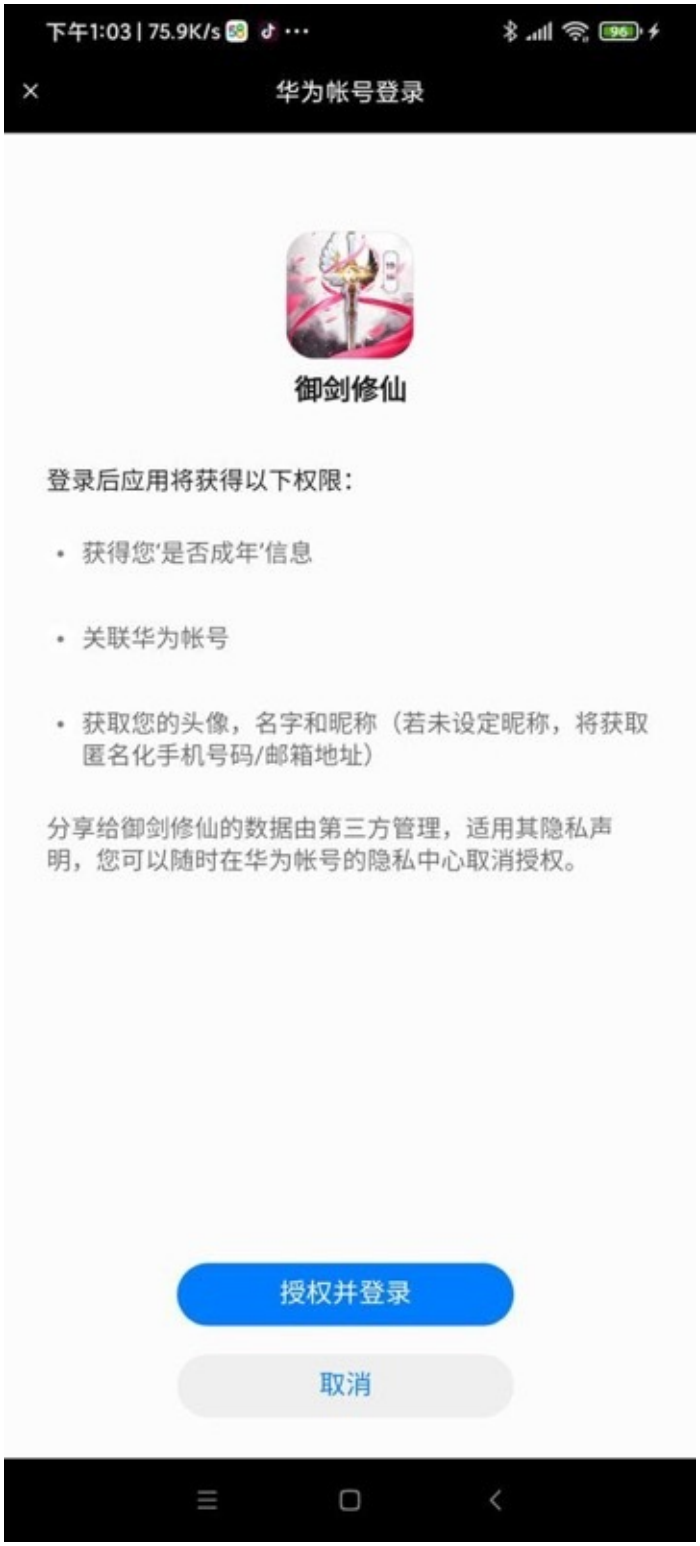
对于来自华为应用市场的游戏app，在登录时需要先登录华为应用市场。

当华为应用市场检测到有代理时，就会无法显示，报错：

无法连接服务器，请退出重试

解决办法：去掉代理

才能正常加载授权页面：

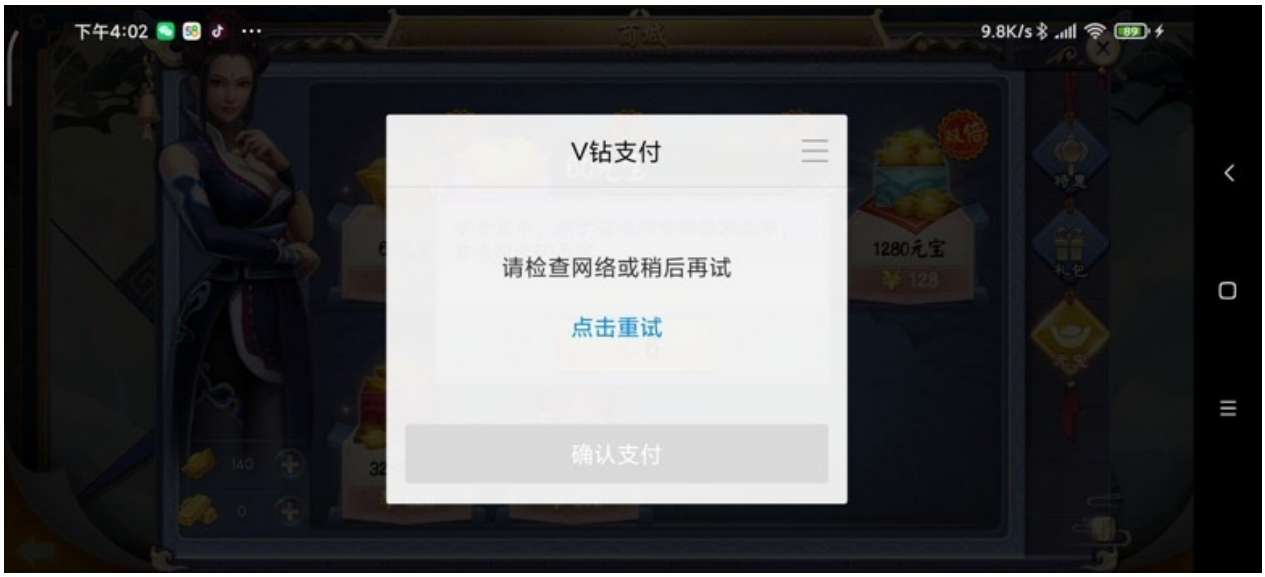

下午1:03 | 75.9K/s
华为帐号登录
御剑修仙
登录后应用将获得以下权限：
• 获得您'是否成年'信息
• 关联华为帐号
• 获取您的头像，名字和昵称（若未设定昵称，将获取匿名化手机号码/邮箱地址）
分享给御剑修仙的数据由第三方管理，适用其隐私声明，您可以随时在华为帐号的隐私中心取消授权。
授权并登录
取消

无法加载，请点击重试

偶尔vivo的支付弹框，也会出现，检测到代理后，无法正常显示，提示：

请检查网络或稍后再试

点击重试



点击一下，即可正常显示：

或者类似的：

加载失败

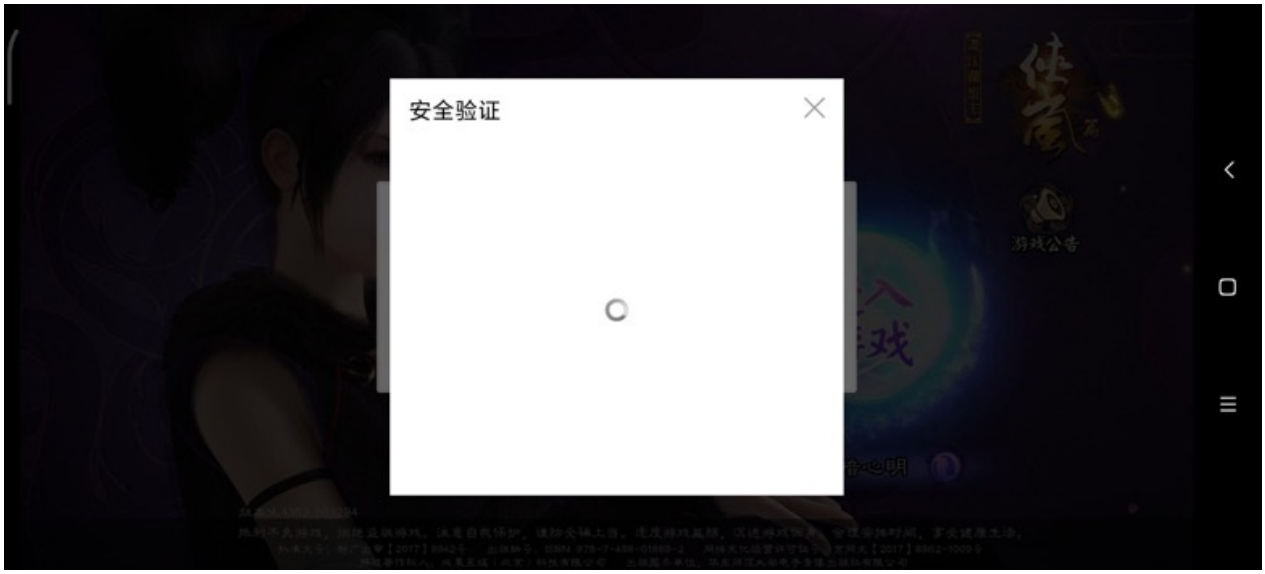
点击重试

点击一下，即可正常显示支付：

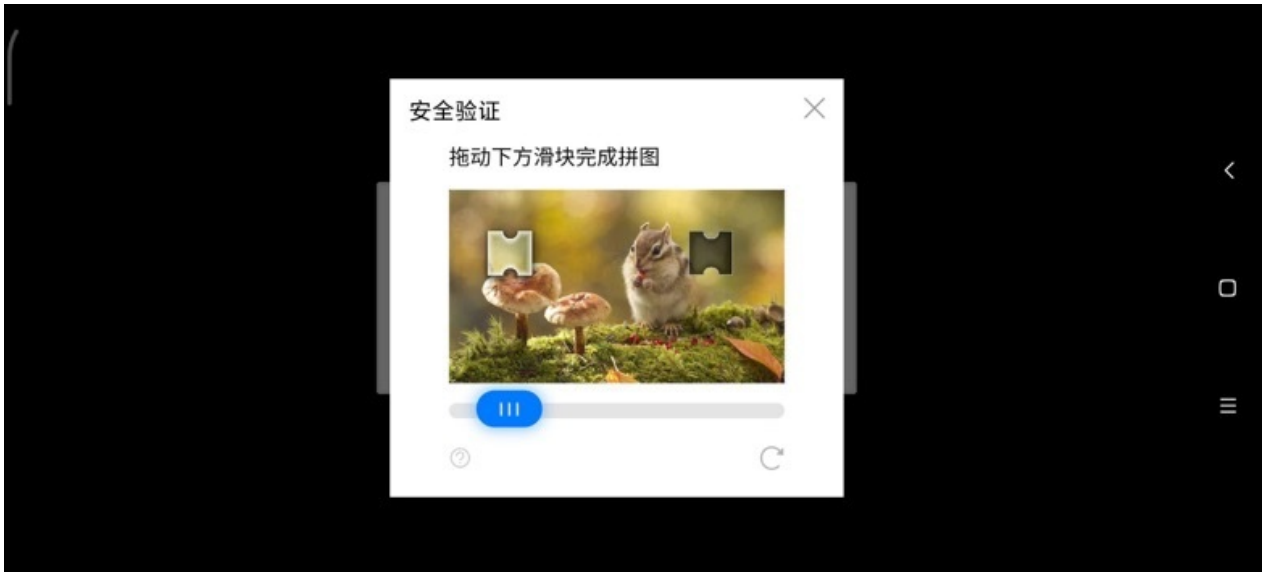
页面卡死在加载中

游戏app发行到vivo应用市场后，登录时往往要登录vivo账号。

其中一种登录弹框时，先显示 滑动补全缺口的图的验证码 的弹框，当vivo发现有代理时，则验证码图片弹框完全就无法显示，且卡死在验证码图片加载页面：



（通过切换WiFi而）去掉代理后，验证码图片才能正常显示：



支付方式弹框不显示，显示等待中

某游戏检测到有了代理后，支付弹框不显示，只提示：正在等待支付

请检查网络后，刷新重试！

小米应用市场发行的游戏登录时也需要登录小米账号授权，检测到代理后，会报错：

请检查网络后，刷新重试！

mitmdump偶尔突然无效

- **现象**

Mac中正常启动mitmdump：

```
mitmdump -p 8081 -s middleware/Save1.py
```

作为手机中WiFi的代理，一直可以正常工作。

但是最近遇到几次了：

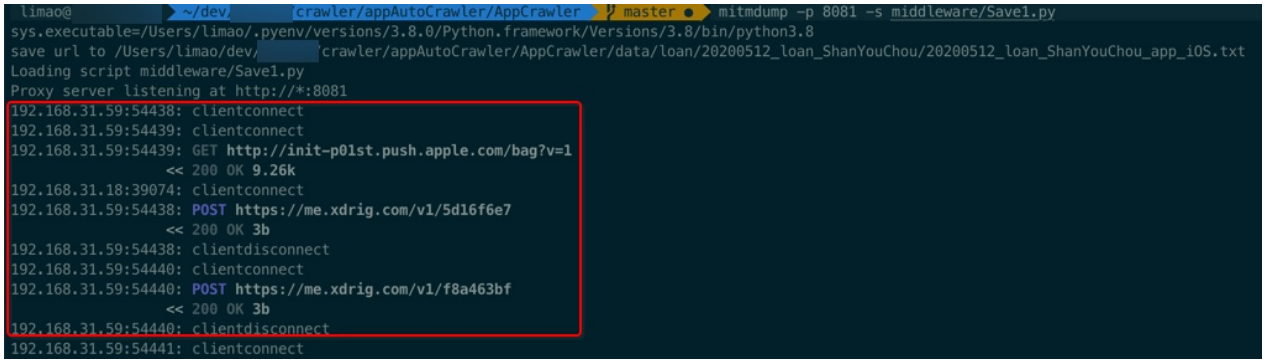
突然，不知道什么原因，mitmdump，就无效了。

导致手机中虽然设置了mitmdump的代理，但是传入mitmdump的脚本无效，无法过滤出url，保存到文件中了。

- **解决办法**：切换WiFi

无意间发现：切换WiFi，比如从 `xxx_guest` 切换到 `xxx_guest_5G`：

即可解决此问题，mitmdump又重新正常工作了，手机端代理就生效了：



后记：

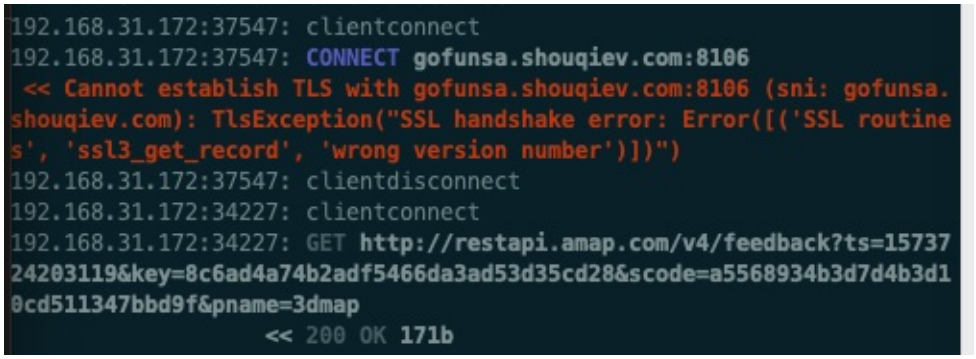
- 手机中：切换不同的WiFi（再加上代理）
  - 多试试几次，也可以规避此 代理不工作 的问题

ssl3_get_record wrong version number

- **现象**

自动抓包工具抓取安卓app `GoFun出行` 期间，mitmproxy的log中一直出现：

```
192.168.31.172:37547: CONNECT gofunsa.shouqiev.com:8106
 << Cannot establish TLS with gofunsa.shouqiev.com:8106 (sni: gofunsa.shouqiev.com): TlsException("SSL handshake error: Error([('SSL routines', 'ssl3_get_record', 'wrong version number')])")
```



- **尝试解决**

```
openssl s_client -debug -connect gofunsa.shouqiev.com:8106
```

输出有：

```
SSL-Session:
    Protocol  : TLSv1.2
    Cipher    : 0000
    Session-ID:
    Session-ID-ctx:
    Master-Key:
    Start Time: 1573725685
    Timeout   : 7200 (sec)
    Verify return code: 0 (ok)
```

看起来版本没问题，是：`TLS v1.2`

后来注意到上面输出了：

```
4574582380:error:1400410B:SSL routines:CONNECT_CR_SRVR_HELLO:wrong version number:/BuildRoot/Library/Caches/com.apple.xbs/Sources/libressl/libressl-22.260.1/libressl-2.6/ssl/ssl_pkt.c:386:
```

也说是：

- wrong version number
  - and then try adding flags from this set: `-no_ssl2` , `-no_ssl3` and `-no_tls1` (consult the

s_client(1) manual page for more details) to work out which version of SSL/TLS has to be enabled for the connection to succeed.

另外通过：

```
brew info openssl
openssl: stable 1.0.2t (bottled) [keg-only]
SSL/TLS cryptography library
https://openssl.org/
Not installed
From: https://github.com/Homebrew/homebrew-core/blob/master/Formula/openssl.rb
==> Caveats
A CA file has been bootstrapped using certificates from the SystemRoots
keychain. To add additional certificates (e.g. the certificates added in
the System keychain), place .pem files in
  /usr/local/etc/openssl/certs

and run
  /usr/local/opt/openssl/bin/c_rehash

openssl is keg-only, which means it was not symlinked into /usr/local,
because Apple has deprecated use of OpenSSL in favor of its own TLS and crypto librarie
s.

==> Analytics
install: 145,281 (30 days), 708,633 (90 days), 5,855,621 (365 days)
install_on_request: 61,384 (30 days), 181,370 (90 days), 877,804 (365 days)
build_error: 0 (30 days)
```

可以看出此处mac安装的openssl的库的版本，是1.0.2t的?

后来：

```
mitmdump --help
```

其中有：

- --ssl-insecure, -k
  - Do not verify upstream server SSL/TLS certificates

有空再试试能否解决此问题。

- 细节详见 *【未解决】安卓抓包mitmproxy报错：TlsException SSL handshake Error routines ssl3_get_record wrong version number

SysCallError 10054 WSAECONNRESET

- **背景**

部分 https=tls=ssl 的url抓包，Mac中正常，但是Windows报错：

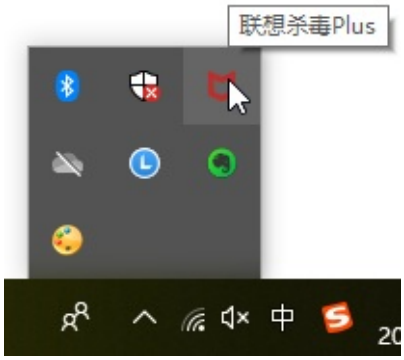
```
192.168.31.172:55087: CONNECT store1.hispace.hicloud.com:443
<< Cannot establish TLS with client (sni: store1.hispace.hicloud.com): TlsException("SSL handshake error: SysCallError(10054, 'WSAECONNRESET')")
TLS passthrough for ('tmge.alicdn.com', 443)
192.168.31.172:56214: CONNECT dnkeeper.hicloud.com:443
<< Cannot establish TLS with client (sni: dnkeeper.hicloud.com): TlsException("SSL handshake error: SysCallError(10054, 'WSAECONNRESET')")
TLS passthrough for ('zconfig.alibabausercontent.com', 443)

192.168.31.172:54363: CONNECT lf.snssdk.com:443
<< Cannot establish TLS with client (sni: lf.snssdk.com): TlsException("SSL handshake error: Error([('SSL routines', 'ssl3_read_bytes', 'sslv3 alert certificate unknown')])")
192.168.31.172:52353: CONNECT gecko-hl.snssdk.com:443
<< Cannot establish TLS with client (sni: gecko-hl.snssdk.com): TlsException("SSL handshake error: Error([('SSL routines', 'ssl3_read_bytes', 'sslv3 alert certificate unknown')])")
192.168.31.172:42426: CONNECT sf3-ttcdn-tos.pstatp.com:443
<< Cannot establish TLS with client (sni: sf3-ttcdn-tos.pstatp.com): TlsException("SSL handshake error: Error([('SSL routines', 'ssl3_read_bytes', 'sslv3 alert certificate unknown')])")
192.168.31.172:36743: CONNECT webcast-hl.amemv.com:443
<< Cannot establish TLS with client (sni: webcast-hl.amemv.com): TlsException("SSL handshake error: Error([('SSL routines', 'ssl3_read_bytes', 'sslv3 alert certificate unknown')])")
```

以及后来的游戏中点击付费按钮，产生的付费链接：

```
Loading script  mitmdumpUrlSaver.py
Proxy server listening at http://*:8081

192.168.31.172:57502: CONNECT hm.baidu.com:443
<< Cannot establish TLS with client (sni: hm.baidu.com): TlsException("(-1, 'Unexpected EOF')")

192.168.31.172:40054: CONNECT apiouterh5.37.com:443
<< Cannot establish TLS with client (sni: apiouterh5.37.com): TlsException("SSL handshake error: Error([('SSL routines', 'ssl3_read_bytes', 'sslv3 alert certificate unknown')])")

192.168.31.172:44871: CONNECT apipayh5.37.com:443<< Cannot establish TLS with client (sni: apipayh5.37.com): TlsException("SSL handshake error: SysCallError(10054, 'WSAECONNRESET')")

TLS passthrough for ('apipayh5.37.com', 443)192.168.31.172:36750: clientdisconnect

192.168.31.172:57503: CONNECT h5.37.com:443<< Cannot establish TLS with client (sni: h5
```

```
.37.com): TlsException("(-1, 'Unexpected EOF')")
TLS passthrough for ('h5.37.com', 443)

TLS passthrough for ('hm.baidu.com', 443)192.168.31.172:57505: CONNECT paysdk.37.com:44
3
<< Cannot establish TLS with client (sni: paysdk.37.com): TlsException("(10054, 'WSAECO
NNRESET')")

TLS passthrough for ('paysdk.37.com', 443)
```

即，出现问题：

- 都是TLS passthrough掉了，保存的txt文件中，没有这些https的url
- 或者是部分https游戏付费链接 都是SSL handshake error
- **可能的原因**
  - windows中此处mitmproxy本身有问题?
  - windows中此处ssl底层库有问题?
  - windows中此处杀毒软件有问题?
    - 因为后来看到有个 联想杀毒plus
    - 

    - 不知道对此是否有影响?

其他

此处整理一些和 `mitmdump` 相关的其他内容。

代码调用

- **背景需求**

Mac中想要用Python代码去控制 `mitmdump` ，即可以启动和停止 `mitmdump`

问题就转化为，Mac中如何写Python代码，能够检测到mitmdump的进程状态，如何解析具体信息，如何杀死对应，mitmdump进程等过程。

- **最后代码**

```
def stopExistingMitmproxy(curDevId):
    logging.info("curDevId=%s", curDevId)
    curDevIdInt = int(curDevId)
    isCheckCmdRunOk, mitmdumpInfoList = checkMitmdumpStatus()
    logging.info("isCheckCmdRunOk=%s, mitmdumpInfoList=%s", isCheckCmdRunOk, mitmdumpIn
foList)
    isRunning = bool(mitmdumpInfoList)
    logging.info("isRunning=%s", isRunning)

    if isCheckCmdRunOk and isRunning:
        foundExistedDevId = False
        existedPid = None

        for eachMitmdumpInfo in mitmdumpInfoList:
            eachDevId = eachMitmdumpInfo["devId"]
            if eachDevId == curDevIdInt:
                foundExistedDevId = True
                existedPid = eachMitmdumpInfo["pid"]
                break

        if foundExistedDevId:
            killOK, errCode = utils.killProcess(existedPid)
            logging.info("killOK=%s, errCode=%s", killOK, errCode)

def debugStartProxy():
    stopExistingMitmproxy(gCurDevId)

    taskFileFullPath = "/Users/limao/dev/xxx/crawler/appAutoCrawler/AppCrawler/task/191
115_card_DongKaKongJian/191115_card_DongKaKongJian_wexin.txt"
    taskId = "5e9552d1c5c2eb3ccdf777bc"
    startTaskProxy(taskId, gCurDevId, taskFileFullPath)

    time.sleep(2)

    isCheckCmdRunOk, mitmdumpInfoList = checkMitmdumpStatus()
    logging.info("isCheckCmdRunOk=%s, mitmdumpInfoList=%s", isCheckCmdRunOk, mitmdumpIn
foList)

def checkMitmdumpStatus():
    # check mitmdump is indeed running
    isCheckCmdRunOk, mitmdumpInfoList = False, []
    checkMitmdumpCmd = "ps aux | grep mitmdump"
```

```
    isCheckCmdRunOk, cmdResult = utils.getCommandOutput(checkMitmdumpCmd)
    logging.info("isCheckCmdRunOk=%s, cmdResult=%s", isCheckCmdRunOk, cmdResult)
    if isCheckCmdRunOk:
        # resultList = cmdResult.split("\n")
        resultList = cmdResult.split(os.linesep)
        logging.info("resultList=%s", resultList)
        # limao            56562   0.0  0.0  4267948    664 s006  R+    5:53下午   0:00.
00 grep mitmdump
        # limao            56560   0.0  0.0  4268636   1112 s006  S+    5:53下午   0:00.
00 /bin/sh -c ps aux | grep mitmdump
        # limao            55396   0.0  0.1  4381268  11568 s011  S+    5:19下午   0:05.
04 /Users/limao/.pyenv/versions/3.8.0/Python.framework/Versions/3.8/Resources/Python.ap
p/Contents/MacOS/Python /Users/limao/.pyenv/versions/3.8.0/bin/mitmdump -p 8081 -s midd
leware/Save1.py
        if resultList:
            for eachLine in resultList:
                logging.info("eachLine=%s", eachLine)
                mitmdumpP = "^\s*(?P<username>\w+)\s+(?P<pid>\d+)\s+.+?mitmdump\s+-p\s+
(?P<port>\d+)\s+-s\s+(?P<scriptFile>middleware/Save(?P<devId>\d+)\.py)\s*$"
                foundMitmdump = re.search(mitmdumpP, eachLine)
                logging.info("foundMitmdump=%s", foundMitmdump)
                if foundMitmdump:
                    username = foundMitmdump.group("username")
                    pid = foundMitmdump.group("pid")
                    port = foundMitmdump.group("port")
                    devId = foundMitmdump.group("devId")
                    scriptFile = foundMitmdump.group("scriptFile")
                    logging.info("username=%s, pid=%s, port=%s, scriptFile=%s, devId=%s"
, username, pid, port, scriptFile, devId)
                    curMitmdumpDict = {
                        "username": username,
                        "pid": int(pid),
                        "port": int(port),
                        "scriptFile": scriptFile,
                        "devId": int(devId),
                    }
                    mitmdumpInfoList.append(curMitmdumpDict)
    logging.info("mitmdumpInfoList=%s", mitmdumpInfoList)
    return isCheckCmdRunOk, mitmdumpInfoList

def killProcess(pid):
    """Kill process by pid

    Args:
        pid (id): process ID
    Returns:
    Raises:
    """
    isKillOk, errCode = False, 0
    pidInt = int(pid)
    killCmd = "kill -9 %s" % pidInt
    returnCode = os.system(killCmd)
    logging.debug("Command: %s -> returnCode=%s", killCmd, returnCode)
    RETURN_CODE_OK = 0
    if returnCode == RETURN_CODE_OK:
        isKillOk = True
```

```
    else:
        errCode = returnCode
    return isKillOk, errCode
```

基本完成了想要的功能：

- 在启动任务前，启动mitmproxy
- 如果之前已有当前设备id的mitmdump在运行，就kill掉
  - 因为很可能是之前的旧的task的对应的代理，所以要关闭掉，再重新启动，才能传递当前task的data文件
- 然后再去启动mitmproxy，之后再去检测看看是否的确已启动

后续优化版本

全局定义：

```
MitmdumpPortBase = 8080
curDevId = 1
RunProxyShellFilename = "runProxy.sh"
```

生成mitmproxy命令

```

#---------- generate start service command ----------

def generateMitmproxyStartCommand(curDevId):
    curMitmdumpPort = MitmdumpPortBase + int(curDevId)
    # mitmproxyStartCommand = "mitmdump -p %d -s middleware/Save%d.py" % (curMitmdumpPo
rt, curDevId)
    mitmproxyStartCommand = "mitmdump -k -p %d -s middleware/Save%d.py" % (curMitmdumpP
ort, curDevId)
    logging.debug("mitmproxyStartCommand=%s", mitmproxyStartCommand)
    # mitmdump -k -p 8081 -s middleware/Save1.py
    mitmproxyCommandList = [
        # "cd /Users/limao/dev/xxx/crawler/appAutoCrawler/AppCrawler",
        "cd %s" % AppCralwerFolder,
        "pwd",
        mitmproxyStartCommand,
    ]
    logging.debug("mitmproxyCommandList=%s", mitmproxyCommandList)
    # ['cd /Users/limao/dev/xxx/crawler/appAutoCrawler/AppCrawler', 'pwd', 'mitmdump -k
 -p 8081 -s middleware/Save1.py']
    # mitmproxyCommandStr = ";".join(mitmproxyCommandList)
    # mitmproxyCommandStr = "; ".join(mitmproxyCommandList)
    mitmproxyCommandStr = "\n".join(mitmproxyCommandList)
    # cd /Users/limao/dev/xxx/crawler/appAutoCrawler/AppCrawler
    # pwd
    # mitmdump -k -p 8081 -s middleware/Save1.py
    logging.debug("mitmproxyCommandStr=%s", mitmproxyCommandStr)
    return mitmproxyCommandStr
```

调用：

```
mitmproxyCmdStr = generateMitmproxyStartCommand("1")
```

和此处的：

```
#---------- generate shell file ----------
def generateRunProxyShell(devId, taskId):
    mitmproxyCmdStr = generateMitmproxyStartCommand(devId)
    logging.debug("mitmproxyCmdStr=%s", mitmproxyCmdStr)
    return generateShellFile(mitmproxyCmdStr, RunProxyShellFilename, taskId)
```

停止当前正在运行的mitmdump

```
def stopExistingMitmproxy(curDevId):
    logging.debug("curDevId=%s", curDevId)
    curDevIdInt = int(curDevId)
    isCheckOk, isRunning, mitmdumpInfoList = detectMitmdumpStatus()
    logging.debug("isCheckOk=%s, isRunning=%s, mitmdumpInfoList=%s", isCheckOk, isRunni
ng, mitmdumpInfoList)

    if isCheckOk and isRunning:
        foundExistedDevId = False
        existedPidInt = None

        for eachMitmdumpInfo in mitmdumpInfoList:
            eachDevIdStr = eachMitmdumpInfo["devId"]
            eachDevIdInt = int(eachDevIdStr)
            if eachDevIdInt == curDevIdInt:
                foundExistedDevId = True
                existedPidStr = eachMitmdumpInfo["pid"]
                existedPidInt = int(existedPidStr)
                break

        if foundExistedDevId:
            killOK, errCode = utils.killProcess(existedPidInt)
            logging.debug("killOK=%s, errCode=%s", killOK, errCode)

            logging.info("%s to stopped mitmproxy", killOK)
```

调用：

```
devId="1"
stopExistingMitmproxy(devId)
```

确保mitmdump已运行

```
CheckServiceRunningInterval = 2.0

def makesureProxyingRunning(devId, taskId):
```

```
    def checkProxyStatus():
        isCheckOk, isRunning, infoList = detectMitmdumpStatus()
        return isCheckOk and isRunning

    def startCurTaskProxy():
        startTaskProxy(devId, taskId)

    makesureServiceRunning(checkProxyStatus, startCurTaskProxy, "Proxy")

def detectMitmdumpStatus():
    # crifanli 9428 0.0 0.6 4341956 19792 s006 S+ 9:16上午 0:23.78 /Users/crifanli/.pyen
v/versions/3.8.3/bin/python3.8 /Users/crifanli/.pyenv/versions/3.8.3/bin/mitmdump -k -p
 8081 -s middleware/Save1.py
    # crifanli 10982 0.0 0.0 4268032 776 s005 S+ 1:51下午 0:00.00 grep mitmdump
    # crifanli 10980 0.0 0.0 4278852 1116 s005 S+ 1:51下午 0:00.01 /bin/sh -c ps aux | g
rep mitmdump
    # mitmdumpP = "^\s*(?P<username>\w+)\s+(?P<pid>\d+)\s+.+?mitmdump\s+-p\s+(?P<port>\
d+)\s+-s\s+(?P<scriptFile>middleware/Save(?P<devId>\d+)\.py)\s*$"
    mitmdumpP = "^\s*(?P<username>\w+)\s+(?P<pid>\d+)\s+.+?mitmdump\s+(-k\s+)?-p\s+(?P<
port>\d+)\s+-s\s+(?P<scriptFile>middleware/Save(?P<devId>\d+)\.py)\s*$"
    return utils.grepProcessStatus("mitmdump", mitmdumpP)

def makesureServiceRunning(checkStatusCallback, startServiceCallback, serviceName=""):
    isRunning = False
    while not isRunning:
        # isRunning = eval(checkStatusCallback)
        isRunning = checkStatusCallback()
        logging.debug("isRunning=%s", isRunning)
        if isRunning:
            break
        else:
            logging.info("%s not running, try to start", serviceName)
            # eval(startServiceCallback)
            startServiceCallback()

        logging.info("Wait %d seconds", CheckServiceRunningInterval)
        time.sleep(CheckServiceRunningInterval)

    logging.info("%s is running", serviceName)
```

Mac中调用Terminal终端去启动mitmdump

```
# CurFilePath = __file__
CurFilePath = os.path.abspath(__file__)
print("CurFilePath=%s" % CurFilePath)
PlatformIntegrationFolder = os.path.dirname(CurFilePath)
print("PlatformIntegrationFolder=%s" % PlatformIntegrationFolder)

OutputFolderName = "output"
OutputRootFolder = os.path.join(PlatformIntegrationFolder, OutputFolderName)

def getTaskRootFolder(taskId):
    taskIdStr = str(taskId)
```

```
    taskFolder = os.path.join(OutputRootFolder, "tasks", taskIdStr)
    return taskFolder

def getTaskShellFolder(taskId):
    taskRootFolder = getTaskRootFolder(taskId)
    taskShellFolder = os.path.join(taskRootFolder, "shell")
    return taskShellFolder

def startTaskProxy(devId, taskId):
    logging.info("Start proxy for: devId=%s, taskId=%s", devId, taskId)
    proxyShellFile = generateRunProxyShell(devId, taskId)
    logging.debug("proxyShellFile=%s", proxyShellFile)
    utils.launchTerminalRunShellCommand(proxyShellFile)

def generateRunProxyShell(devId, taskId):
    mitmproxyCmdStr = generateMitmproxyStartCommand(devId)
    logging.debug("mitmproxyCmdStr=%s", mitmproxyCmdStr)
    return generateShellFile(mitmproxyCmdStr, RunProxyShellFilename, taskId)

def generateShellFile(fileContentStr, shellFilename, taskId=None):
    """Generate shell file, which is used to run command
        such as
            mitmdump proxy
            crawlerStart.py
            USB port forward
            wda server(xcodebuild/XCode)
    """
    logging.debug("fileContentStr=%s, shellFilename=%s, taskId=%s", fileContentStr, she
llFilename, taskId)
    if taskId:
        shellFolder = getTaskShellFolder(taskId)
        # /Users/limao/dev/xxx/crawler/appAutoCrawler/AppCrawler/platformIntegration/ou
tput/tasks/5e9552d1c5c2eb3ccdf777bc/shell
    else:
        shellFolder = OutputRootFolder
    logging.debug("shellFolder=%s", shellFolder)
    shellFullPath = os.path.join(shellFolder, shellFilename)
    logging.debug("shellFullPath=%s", shellFullPath)
    # /Users/limao/dev/xxx/crawler/appAutoCrawler/AppCrawler/platformIntegration/output
/tasks/5e9552d1c5c2eb3ccdf777bc/shell/runProxy.sh
    shellAbsFullPath = os.path.abspath(shellFullPath)
    logging.debug("shellAbsFullPath=%s", shellAbsFullPath)
    respShellFullPath = shellAbsFullPath
    utils.saveTextToFile(respShellFullPath, fileContentStr)
    utils.chmodAddX(shellFullPath, isOnlySelf=False)
    # utils.chmodAddX(respShellFullPath)
    logging.debug("respShellFullPath=%s", respShellFullPath)
    # /Users/limao/dev/xx/crawler/appAutoCrawler/AppCrawler/platformIntegration/output/
tasks/5e9552d1c5c2eb3ccdf777bc/shell/runProxy.sh
    return respShellFullPath
```

调用到的相关的库函数：

`other/common/libs/utils.py`

```
import re

#-------------------------------------------------------------------------------
# Process
#-------------------------------------------------------------------------------

def runCommand(consoleCommand):
    """run command using subprocess call"""
    isRunCmdOk = False
    errMsg = "Unknown Error"

    try:
        resultCode = subprocess.check_call(consoleCommand, shell=True)
        if resultCode == 0:
            isRunCmdOk = True
            errMsg = ""
        else:
            isRunCmdOk = False
            errMsg = "%s return code %s" % (consoleCommand, resultCode)
    except subprocess.CalledProcessError as callProcessErr:
        isRunCmdOk = False
        errMsg = str(callProcessErr)
        # "Command 'ffmpeg -y -i /Users/crifan/.../debug/extractAudio/show_112233_video
.mp4 -ss 00:00:05.359 -to 00:00:06.763 -b:a 128k /.../show_112233_video_000005359_00000
6763.mp3 2> /dev/null' returned non-zero exit status 1."

    return isRunCmdOk, errMsg

def getCommandOutput(consoleCommand, consoleOutputEncoding="utf-8"):
    """
        get command output from terminal
    """
    # print("getCommandOutput: consoleCommand=%s" % consoleCommand)
    isRunCmdOk = False
    consoleOutput = ""
    try:
        # consoleOutputByte = subprocess.check_output(consoleCommand)

        consoleOutputByte = subprocess.check_output(consoleCommand, shell=True)

        # commandPartList = consoleCommand.split(" ")
        # print("commandPartList=%s" % commandPartList)
        # consoleOutputByte = subprocess.check_output(commandPartList)
        # print("type(consoleOutputByte)=%s" % type(consoleOutputByte)) # <class 'bytes
'>
        # print("consoleOutputByte=%s" % consoleOutputByte) # b'640x360\n'

        consoleOutput = consoleOutputByte.decode(consoleOutputEncoding) # '640x360\n'
        consoleOutput = consoleOutput.strip() # '640x360'
        isRunCmdOk = True
    except subprocess.CalledProcessError as callProcessErr:
        cmdErrStr = str(callProcessErr)
        print("Error %s for run command %s" % (cmdErrStr, consoleCommand))

    # print("isRunCmdOk=%s, consoleOutput=%s" % (isRunCmdOk, consoleOutput))
```

```
    return isRunCmdOk, consoleOutput

def launchTerminalRunShellCommand(shellFile, isForceNewInstance=True, isUseiTerm2=False)
:
    """in Mac, Launch terminal(Mac Terminal / iTerm2) and execute shell file, which con
tain command to run

    Args:
        shellFile (str): shell file full path
        isUseiTerm2 (bool): True to use iTerm2, False to use Mac builtin Terminal
        isForceNewInstance (bool): whether pase -n to open, which means: Open a new ins
tance of the application even if one is already running
    Returns:
    Raises:
    """
    logging.debug("shellFile=%s, isForceNewInstance=%s, isUseiTerm2=%s", shellFile, isF
orceNewInstance, isUseiTerm2)

    TerminalApp_iTerm2 = '/Applications/iTerm.app'
    TerminalApp_Terminal = 'Terminal'
    if isUseiTerm2:
        terminalApp = TerminalApp_iTerm2
    else:
        terminalApp = TerminalApp_Terminal

    cmdList = [
        "/usr/bin/open",
    ]

    if isForceNewInstance:
        cmdList.append("-n")

    extarArgs = shellFile
    restCmdList = [
        "-a",
        terminalApp,
        '--args',
        extarArgs,
    ]
    cmdList.extend(restCmdList)
    logging.debug("cmdList=%s" % cmdList)

    curProcess = subprocess.Popen(cmdList, stdin=subprocess.PIPE, stdout=subprocess.PIPE
)
    logging.debug("curProcess=%s" % curProcess)

    returnCode = None
    while True:
        returnCode = curProcess.poll()
        logging.debug("returnCode=%s", returnCode)
        if returnCode is not None:
            logging.debug("subprocess end: returnCode=%s", returnCode)
            break
        time.sleep(0.5)

    logging.debug("Final returnCode=%s", returnCode)
```

```
    logging.debug("Complete launch %s and run shell %s", terminalApp, shellFile)

def killProcess(pid):
    """Kill process by pid

    Args:
        pid (id): process ID
    Returns:
    Raises:
    """
    isKillOk, errCode = False, 0
    pidInt = int(pid)
    killCmd = "kill -9 %s" % pidInt
    returnCode = os.system(killCmd)
    logging.debug("Command: %s -> returnCode=%s", killCmd, returnCode)
    RETURN_CODE_OK = 0
    if returnCode == RETURN_CODE_OK:
        isKillOk = True
    else:
        errCode = returnCode
    return isKillOk, errCode

def grepProcessStatus(processFile, singleLinePattern, psCmd="ps aux"):
    """grep process info status from ps output

    Args:
        processFile (str): process file name
        singleLinePattern (str): single process line search pattern
        psCmd (str): ps command, default: ps aux
    Returns:
    Raises:
    Examples:
        input: "crawlerStart.py", "^\s*(?P<username>\w+)\s+(?P<pid>\d+)\s+.+?python\s+c
rawlerStart\.py\s+-task\s+(?P<taskFile>\S+)\s+-id\s+(?P<curDevId>\d+)$"
        output: [{'username': 'limao', 'pid': '64320', 'taskFile': '/Users/limao/dev/xx
x/crawler/appAutoCrawler/AppCrawler/task/191115_card_DongKaKongJian/191115_card_DongKaK
ongJian_wexin.txt', 'curDevId': '1'}]
    """
    logging.debug("processFile=%s, singleLinePattern=%s", processFile, singleLinePattern
)
    isCheckCmdRunOk, isRunning, processInfoList = False, False, []

    groupNameList = re.findall("\(\?P<(\w+)>", singleLinePattern)
    logging.debug("groupNameList=%s", groupNameList)
    # groupNameList=['username', 'pid', 'port', 'scriptFile', 'devId']
    grepProcessCmd = "%s | grep %s" % (psCmd, processFile)
    logging.debug("grepProcessCmd=%s", grepProcessCmd)
    isCheckCmdRunOk, cmdResult = getCommandOutput(grepProcessCmd)
    logging.debug("isCheckCmdRunOk=%s, cmdResult=%s", isCheckCmdRunOk, cmdResult)
    if isCheckCmdRunOk:
        # lineSeparator = "\n"
        lineSeparator = os.linesep
        resultList = cmdResult.split(lineSeparator)
        logging.debug("resultList=%s", resultList)
        # limao            56562   0.0  0.0  4267948    664 s006  R+    5:53下午   0:00.
00 grep mitmdump
```

```
        # limao           56560   0.0  0.0  4268636   1112 s006  S+    5:53下午   0:00.
00 /bin/sh -c ps aux | grep mitmdump
        # limao           55396   0.0  0.1  4381268  11568 s011  S+    5:19下午   0:05.
04 /Users/limao/.pyenv/versions/3.8.0/Python.framework/Versions/3.8/Resources/Python.ap
p/Contents/MacOS/Python /Users/limao/.pyenv/versions/3.8.0/bin/mitmdump -p 8081 -s midd
leware/Save1.py
        if resultList:
            for eachLine in resultList:
                logging.debug("eachLine=%s", eachLine)
                foundProcess = re.search(singleLinePattern, eachLine)
                logging.debug("foundProcess=%s", foundProcess)
                if foundProcess:
                    curProcessInfoDict = {}
                    for eachKey in groupNameList:
                        curValue = foundProcess.group(eachKey)
                        curProcessInfoDict[eachKey] = curValue
                    logging.debug("curProcessInfoDict=%s", curProcessInfoDict)
                    processInfoList.append(curProcessInfoDict)

    isRunning = bool(processInfoList)
    logging.debug("isRunning=%s, processInfoList=%s", isRunning, processInfoList)
    return isCheckCmdRunOk, isRunning, processInfoList
```

注：

相关库文件的最新版，详见：

- https://github.com/crifan/crifanLibPython/blob/master/python3/crifanLib/crifanSystem.py
  - `grepProcessStatus`
  - `killProcess`
  - `launchTerminalRunShellCommand`
  - `getCommandOutput`
  - `runCommand`

打包exe

`windows` 中用 `PyInstaller` 打包 `python` 脚本为exe文件

其中python脚本调用到 `mitmdump`

可以理解为：打包mitmdump的Python为exe

核心命令：

```
pyinstaller pymitmdump\mitmdumpStartApi.py --distpath pymitmdumpstartdist --add-data "pymitmdump\mitmdump_executable;mitmdump_executable" --add-data "pymitmdump\mitmdumpUrlSaver.py;."

pyinstaller pymitmdump\mitmdumpOtherApi.py --distpath pymitmdumpotherdist
```

可以生成2个exe文件。

- 细节详见：【已解决】windows中用PyInstaller打包mitmdump的Python脚本为exe

附录

下面列出相关参考资料。

help语法

mitmdump --help

```
╭ mitmdump --help
usage: mitmdump [options] [filter]

positional arguments:
  filter_args           Filter expression, equivalent to setting both the view_filter a
nd save_stream_filter options.

optional arguments:
  -h, --help            show this help message and exit
  --version             show version number and exit
  --options             Show all options and their default values
  --commands            Show all commands and their signatures
  --set option[=value]  Set an option. When the value is omitted, booleans are set to t
rue, strings and integers are set to None (if permitted),
                        and sequences are emptied. Boolean values can be true, false or
 toggle.
  -q, --quiet           Quiet.
  -v, --verbose         Increase log verbosity.
  --mode MODE, -m MODE  Mode can be "regular", "transparent", "socks5", "reverse:SPEC",
 or "upstream:SPEC". For reverse and upstream proxy modes,
                        SPEC is host specification in the form of "http[s]://host[:port
]".
  --no-anticache
  --anticache           Strip out request headers that might cause the server to return
304-not-modified.
  --no-showhost
  --showhost            Use the Host header to construct URLs for display.
  --rfile PATH, -r PATH
                        Read flows from file.
  --scripts SCRIPT, -s SCRIPT
                        Execute a script. May be passed multiple times.
  --stickycookie FILTER
                        Set sticky cookie filter. Matched against requests.
  --stickyauth FILTER   Set sticky auth filter. Matched against requests.
  --save-stream-file PATH, -w PATH
                        Stream flows to file as they arrive. Prefix path with + to appe
nd.
  --no-anticomp
  --anticomp            Try to convince servers to send us un-compressed data.
  --flow-detail LEVEL   The display detail level for flows in mitmdump: 0 (almost quiet)
 to 3 (very verbose). 0: shortened request URL, response
                        status code, WebSocket and TCP message notifications. 1: full r
equest URL with response status code 2: 1 + HTTP headers 3:
                        2 + full response content, content of WebSocket and TCP message
s.

Proxy Options:
  --listen-host HOST    Address to bind proxy to.
```

```
  --listen-port PORT, -p PORT
                        Proxy service port.
  --no-server, -n
  --server              Start a proxy server. Enabled by default.
  --ignore-hosts HOST   Ignore host and forward all traffic without processing it. In t
ransparent mode, it is recommended to use an IP address
                        (range), not the hostname. In regular mode, only SSL traffic is
 ignored and the hostname should be used. The supplied
                        value is interpreted as a regular expression and matched on the
ip or the hostname. May be passed multiple times.
  --allow-hosts HOST    Opposite of --ignore-hosts. May be passed multiple times.
  --tcp-hosts HOST      Generic TCP SSL proxy mode for all hosts that match the pattern
. Similar to --ignore, but SSL connections are intercepted.
                        The communication contents are printed to the log in verbose mo
de. May be passed multiple times.
  --upstream-auth USER:PASS
                        Add HTTP Basic authentication to upstream proxy and reverse pro
xy requests. Format: username:password.
  --proxyauth SPEC      Require proxy authentication. Format: "username:pass", "any" to
 accept any user/pass combination, "@path" to use an Apache
                        htpasswd file, or "ldap[s]:url_server_ldap:dn_auth:password:dn_
subtree" for LDAP authentication.
  --no-rawtcp
  --rawtcp              Enable/disable experimental raw TCP support. TCP connections st
arting with non-ascii bytes are treated as if they would
                        match tcp_hosts. The heuristic is very rough, use with caution.
 Disabled by default.
  --no-http2
  --http2               Enable/disable HTTP/2 support. HTTP/2 support is enabled by def
ault.

SSL:
  --certs SPEC          SSL certificates of the form "[domain=]path". The domain may in
clude a wildcard, and is equal to "*" if not specified. The
                        file at path is a certificate in PEM format. If a private key i
s included in the PEM, it is used, else the default key in
                        the conf dir is used. The PEM file should contain the full cert
ificate chain, with the leaf certificate as the first
                        entry. May be passed multiple times.
  --no-ssl-insecure
  --ssl-insecure, -k    Do not verify upstream server SSL/TLS certificates.
  --key-size KEY_SIZE   TLS key size for certificates and CA.

Client Replay:
  --client-replay PATH, -C PATH
                        Replay client requests from a saved file. May be passed multipl
e times.

Server Replay:
  --server-replay PATH, -S PATH
                        Replay server responses from a saved file. May be passed multip
le times.
  --no-server-replay-kill-extra
  --server-replay-kill-extra
                        Kill extra requests during replay.
  --no-server-replay-nopop
```

```
  --server-replay-nopop
                        Don't remove flows from server replay state after use. This mak
es it possible to replay same response multiple times.
  --no-server-replay-refresh
  --server-replay-refresh
                        Refresh server replay responses by adjusting date, expires and
last-modified headers, as well as adjusting cookie
                        expiration.

Replacements:
  --replacements PATTERN, -R PATTERN
                        Replacement patterns of the form "/pattern/regex/replacement",
where the separator can be any character. May be passed
                        multiple times.

Set Headers:
  --setheaders PATTERN, -H PATTERN
                        Header set pattern of the form "/pattern/header/value", where t
he separator can be any character. May be passed multiple
                        times.
```

mitmproxy --help

```
mitmproxy --help
usage: mitmproxy [options]

optional arguments:
  -h, --help            show this help message and exit
  --version             show version number and exit
  --options             Show all options and their default values
  --commands            Show all commands and their signatures
  --set option[ value]  Set an option. When the value is omitted, booleans are set to t
rue, strings and integers are set to None (if permitted),
                        and sequences are emptied. Boolean values can be true, false or
 toggle.
  -q, --quiet           Quiet.
  -v, --verbose         Increase log verbosity.
  --mode MODE, -m MODE  Mode can be "regular", "transparent", "socks5", "reverse:SPEC",
 or "upstream:SPEC". For reverse and upstream proxy modes,
                        SPEC is host specification in the form of "http[s]://host[:port
]".
  --no-anticache
  --anticache           Strip out request headers that might cause the server to return
304-not-modified.
  --no-showhost
  --showhost            Use the Host header to construct URLs for display.
  --rfile PATH, -r PATH
                        Read flows from file.
  --scripts SCRIPT, -s SCRIPT
                        Execute a script. May be passed multiple times.
  --stickycookie FILTER
                        Set sticky cookie filter. Matched against requests.
  --stickyauth FILTER   Set sticky auth filter. Matched against requests.
```

```
  --save-stream-file PATH, -w PATH
                        Stream flows to file as they arrive. Prefix path with + to appe
nd.
  --no-anticomp
  --anticomp            Try to convince servers to send us un-compressed data.
  --console-layout {horizontal,single,vertical}
                        Console layout.
  --no-console-layout-headers
  --console-layout-headers
                        Show layout component headers

Proxy Options:
  --listen-host HOST    Address to bind proxy to.
  --listen-port PORT, -p PORT
                        Proxy service port.
  --no-server, -n
  --server              Start a proxy server. Enabled by default.
  --ignore-hosts HOST   Ignore host and forward all traffic without processing it. In t
ransparent mode, it is recommended to use an IP address
                        (range), not the hostname. In regular mode, only SSL traffic is
 ignored and the hostname should be used. The supplied
                        value is interpreted as a regular expression and matched on the
ip or the hostname. May be passed multiple times.
  --allow-hosts HOST    Opposite of --ignore-hosts. May be passed multiple times.
  --tcp-hosts HOST      Generic TCP SSL proxy mode for all hosts that match the pattern
. Similar to --ignore, but SSL connections are intercepted.
                        The communication contents are printed to the log in verbose mo
de. May be passed multiple times.
  --upstream-auth USER:PASS
                        Add HTTP Basic authentication to upstream proxy and reverse pro
xy requests. Format: username:password.
  --proxyauth SPEC      Require proxy authentication. Format: "username:pass", "any" to
 accept any user/pass combination, "@path" to use an Apache
                        htpasswd file, or "ldap[s]:url_server_ldap:dn_auth:password:dn_
subtree" for LDAP authentication.
  --no-rawtcp
  --rawtcp              Enable/disable experimental raw TCP support. TCP connections st
arting with non-ascii bytes are treated as if they would
                        match tcp_hosts. The heuristic is very rough, use with caution.
 Disabled by default.
  --no-http2
  --http2               Enable/disable HTTP/2 support. HTTP/2 support is enabled by def
ault.

SSL:
  --certs SPEC          SSL certificates of the form "[domain=]path". The domain may in
clude a wildcard, and is equal to "*" if not specified. The
                        file at path is a certificate in PEM format. If a private key i
s included in the PEM, it is used, else the default key in
                        the conf dir is used. The PEM file should contain the full cert
ificate chain, with the leaf certificate as the first
                        entry. May be passed multiple times.
  --no-ssl-insecure
  --ssl-insecure, -k    Do not verify upstream server SSL/TLS certificates.
  --key-size KEY_SIZE   TLS key size for certificates and CA.
```

```
Client Replay:
  --client-replay PATH, -C PATH
                        Replay client requests from a saved file. May be passed multipl
e times.

Server Replay:
  --server-replay PATH, -S PATH
                        Replay server responses from a saved file. May be passed multip
le times.
  --no-server-replay-kill-extra
  --server-replay-kill-extra
                        Kill extra requests during replay.
  --no-server-replay-nopop
  --server-replay-nopop
                        Don't remove flows from server replay state after use. This mak
es it possible to replay same response multiple times.
  --no-server-replay-refresh
  --server-replay-refresh
                        Refresh server replay responses by adjusting date, expires and
last-modified headers, as well as adjusting cookie
                        expiration.

Replacements:
  --replacements PATTERN, -R PATTERN
                        Replacement patterns of the form "/pattern/regex/replacement",
where the separator can be any character. May be passed
                        multiple times.

Set Headers:
  --setheaders PATTERN, -H PATTERN
                        Header set pattern of the form "/pattern/header/value", where t
he separator can be any character. May be passed multiple
                        times.

Filters:
  See help in mitmproxy for filter expression syntax.

  --intercept FILTER    Intercept filter expression.
  --view-filter FILTER  Limit the view to matching flows.
```

mitmweb --help

```
~ mitmweb --help
usage: mitmweb [options]

optional arguments:
  -h, --help            show this help message and exit
  --version             show version number and exit
  --options             Show all options and their default values
  --commands            Show all commands and their signatures
  --set option[=value]  Set an option. When the value is omitted, booleans are set to t
rue, strings and integers are set to None (if permitted),
                        and sequences are emptied. Boolean values can be true, false or
```

```
 toggle.
  -q, --quiet           Quiet.
  -v, --verbose         Increase log verbosity.
  --mode MODE, -m MODE  Mode can be "regular", "transparent", "socks5", "reverse:SPEC",
 or "upstream:SPEC". For reverse and upstream proxy modes,
                        SPEC is host specification in the form of "http[s]://host[:port
]".
  --no-anticache
  --anticache           Strip out request headers that might cause the server to return
304-not-modified.
  --no-showhost
  --showhost            Use the Host header to construct URLs for display.
  --rfile PATH, -r PATH
                        Read flows from file.
  --scripts SCRIPT, -s SCRIPT
                        Execute a script. May be passed multiple times.
  --stickycookie FILTER
                        Set sticky cookie filter. Matched against requests.
  --stickyauth FILTER   Set sticky auth filter. Matched against requests.
  --save-stream-file PATH, -w PATH
                        Stream flows to file as they arrive. Prefix path with + to appe
nd.
  --no-anticomp
  --anticomp            Try to convince servers to send us un-compressed data.

Mitmweb:
  --no-web-open-browser
  --web-open-browser    Start a browser.
  --web-port PORT       Web UI port.
  --web-iface INTERFACE
                        Web UI interface.

Proxy Options:
  --listen-host HOST    Address to bind proxy to.
  --listen-port PORT, -p PORT
                        Proxy service port.
  --no-server, -n
  --server              Start a proxy server. Enabled by default.
  --ignore-hosts HOST   Ignore host and forward all traffic without processing it. In t
ransparent mode, it is recommended to use an IP address
                        (range), not the hostname. In regular mode, only SSL traffic is
 ignored and the hostname should be used. The supplied
                        value is interpreted as a regular expression and matched on the
ip or the hostname. May be passed multiple times.
  --allow-hosts HOST    Opposite of --ignore-hosts. May be passed multiple times.
  --tcp-hosts HOST      Generic TCP SSL proxy mode for all hosts that match the pattern
. Similar to --ignore, but SSL connections are intercepted.
                        The communication contents are printed to the log in verbose mo
de. May be passed multiple times.
  --upstream-auth USER:PASS
                        Add HTTP Basic authentication to upstream proxy and reverse pro
xy requests. Format: username:password.
  --proxyauth SPEC      Require proxy authentication. Format: "username:pass", "any" to
 accept any user/pass combination, "@path" to use an Apache
                        htpasswd file, or "ldap[s]:url_server_ldap:dn_auth:password:dn_
subtree" for LDAP authentication.
```

```
  --no-rawtcp
  --rawtcp              Enable/disable experimental raw TCP support. TCP connections st
arting with non-ascii bytes are treated as if they would
                        match tcp_hosts. The heuristic is very rough, use with caution.
 Disabled by default.
  --no-http2
  --http2               Enable/disable HTTP/2 support. HTTP/2 support is enabled by def
ault.

SSL:
  --certs SPEC          SSL certificates of the form "[domain=]path". The domain may in
clude a wildcard, and is equal to "*" if not specified. The
                        file at path is a certificate in PEM format. If a private key i
s included in the PEM, it is used, else the default key in
                        the conf dir is used. The PEM file should contain the full cert
ificate chain, with the leaf certificate as the first
                        entry. May be passed multiple times.
  --no-ssl-insecure
  --ssl-insecure, -k    Do not verify upstream server SSL/TLS certificates.
  --key-size KEY_SIZE   TLS key size for certificates and CA.

Client Replay:
  --client-replay PATH, -C PATH
                        Replay client requests from a saved file. May be passed multipl
e times.

Server Replay:
  --server-replay PATH, -S PATH
                        Replay server responses from a saved file. May be passed multip
le times.
  --no-server-replay-kill-extra
  --server-replay-kill-extra
                        Kill extra requests during replay.
  --no-server-replay-nopop
  --server-replay-nopop
                        Don't remove flows from server replay state after use. This mak
es it possible to replay same response multiple times.
  --no-server-replay-refresh
  --server-replay-refresh
                        Refresh server replay responses by adjusting date, expires and
last-modified headers, as well as adjusting cookie
                        expiration.

Replacements:
  --replacements PATTERN, -R PATTERN
                        Replacement patterns of the form "/pattern/regex/replacement",
where the separator can be any character. May be passed
                        multiple times.

Set Headers:
  --setheaders PATTERN, -H PATTERN
                        Header set pattern of the form "/pattern/header/value", where t
he separator can be any character. May be passed multiple
                        times.

Filters:
```

```
    See help in mitmproxy for filter expression syntax.

    --intercept FILTER    Intercept filter expression.
```

参考资料

- 【记录】Mac M2 Max中安装mitmproxy
-
- 【已解决】Mac中安装Mitmdump和启动服务
- 【未解决】windows中pip安装mitmproxy报错： build _openssl.c error C2065 X509_V_FLAG_CB_ISSUER_CHECK undeclared identifier
- 【未解决】windows中pip install mitmproxy失败：ERROR Could not build wheels for cryptography which use PEP 517 and cannot be installed directly
- 【已解决】iPhone8P中安装mitmproxy的CA的ssl证书
- 【已解决】给iPhone添加mitmproxy的mitmdump代理用于保存抓包链接到文件
- 【记录】给安卓手机中安装mitmproxy代理的SSL证书
- 【记录】给自动抓包工具的安卓手机设置mitmproxy代理用于能抓包到链接地址
- 【已解决】安卓手机小米9中安装mitmproxy的SSL代理证书
- 【无需解决】小米9中WLAN或WAPI证书中找不到mitmproxy的SSL的pem证书文件
- 【无法解决】小米9中用ES文件管理器安装pem证书
- 【无法解决】红米Note8Pro中用微信或小米浏览器下载mitmproxy的SSL代理证书
- 【已解决】红米Note8Pro中用QQ浏览器下载mitmproxy的Android的SSL代理证书
- 【已解决】红米10X安卓手机中无法下载mitmproxy的证书文件
- 【已解决】给安卓手机ViVo的iQOO U1x下载和安装mitmproxy的SSL代理证书
- 【已解决】安卓手机Vivo的iQOO U1x中手动安装mitmproxy-ca-cert.pem证书文件
- 【已解决】安卓手机Vivo的iQOO U1x中点击安装mitmproxy的pem证书报错：未找到证书文件
- 【未解决】给安卓手机Vivo的iQOO U1x初始化mitmdump的代理环境
- 【已解决】给VMWare中macOS中抓包项目开启mitmdump代理
- 【已解决】红米Note8Pro中去下载mitmproxy证书提示：if you can see this, traffic is not passing through mitmproxy
- 【已解决】windows中用PyInstaller打包mitmdump的Python脚本为exe
- 【已解决】自动抓包平台化：Python调用命令行启动mitmproxy代理
- 【已解决】用自动处理任务脚本启动自动测试工具测试自动化安卓游戏
- 【已解决】Mac中用Python检测mitmdump进程状态和杀死原有进程
- 【已解决】Python中实现检测mitmdump进程服务的状态
- 【已解决】mitmproxy代理报错：Cannot establish TLS with 443 sni None TlsException Cannot validate certificate hostname without SNI
- 【无法解决】安卓游戏加了代理后支付页面时mitmdump报错：TlsException SSL handshake error Error SSL routines ssl3_read_bytes sslv3 alert certificate unknown
- 【未解决】windows中用mitmproxy无法抓包部分http付费链接
- 【未解决】安卓抓包mitmproxy报错：TlsException SSL handshake Error routines ssl3_get_record wrong version number
- 【无法解决】Mac中mitmdump通过-s加载python脚本时指定Python版本
- 【已解决】mac中Python2和Python3都已安装了yaml但mitmdump -s加载python脚本中导入yaml还是报错
- 【已解决】Mac中让mitmdump解析python脚本不用自己内置Python而是用系统Python
- 【已解决】Mac中运行mitmdump再次报错：Failed to import yaml
- 【已解决】Mac中mitmdump运行命令报错：in script py No module named yaml

- 【已解决】自动化测试安卓游戏烈焰龙城：从主页到带支付的真正支付页面
- 【已解决】自动化测试安卓游戏烈焰龙城：优化是否是支付页面以及点击支付出现支付弹框的逻辑
- 【已解决】给VMWare中macOS中抓包项目开启mitmdump代理
- 【已解决】提取自动抓包工具中的mitmdump自动保存代理抓包出来的url链接保存到文件
- 【已解决】mitmdump中运行python的script报错：ModuleNotFoundError No module named tldextract
-
- Android Security: SSL Pinning. Using SSL in an Android app is easy… | by Matthew Dolan | Medium
- Bypassing Certificate Pinning on Android for fun and profit | by Felipe Lima | Medium
- Bypassing SSL Pinning on Android via Reverse Engineering.pdf
- https://dl.packetstormsecurity.net/papers/general/android-sslpinning.pdf
- mitmproxy/tls_passthrough.py at master · mitmproxy/mitmproxy
- Mitmproxy教程 - zha0gongz1 - 博客园
- Installation (mitmproxy.org)
-