Frida官方文档-快速入门指南

原创

helloworddm 于 2020-03-23 22:57:59 发布 2085 收藏 1

分类专栏： Android平台 Frida系列

本文链接：https://blog.csdn.net/helloworlddm/article/details/105059996

Android平台 同时被 2 个专栏收录

60 篇文章 5 订阅

订阅专栏

Frida系列

74 篇文章 11 订阅

订阅专栏

对于不耐烦的人， 这里是使用Frida进行跟踪的方法

```
~ $ pip install frida-tools
~ $ frida-trace -i "recv*" -i "read*" *twitter*
recv: Auto-generated handler: …/recv.js
#(snip)
recvfrom: Auto-generated handler: …/recvfrom.js
Started tracing 21 functions. Press Ctrl+C to stop.
    39 ms recv()
   112 ms recvfrom()
   128 ms recvfrom()
   129 ms recvfrom()
```

如您所见，Frida将自己注入Twitter，枚举已加载的共享库，并钩住名称以recv或read开头的所有函数。 它还生成了一些样板脚本，用于在函数调用发生时检查它们。 现在，这些脚本只是您要进行编辑以使其具有品味的示例，随着它们在文件系统上的更改，它们将自动重新加载。 默认情况下，它们仅打印函数的名称，如您在上面的输出中所见

现在，让我们看一下生成的recvfrom.js:

```
/*
 * Auto-generated by Frida. Please modify to match the
 * signature of recvfrom.
 *
 * This stub is somewhat dumb. Future verions of Frida
 * could auto-generate based on OS API references, manpages,
 * etc. (Pull-requests appreciated!)
 *
 * For full API reference, see:
 * http://www.frida.re/docs/javascript-api/
 */

{
    /**
     * Called synchronously when about to call recvfrom.
     *
     * @this {object} - Object allowing you to store state for
     * use in onLeave.
     * @param {function} log - Call this function with a string
     * to be presented to the user.
     * @param {array} args - Function arguments represented as
     * an array of NativePointer objects.
     * For example use args[0].readUtf8String() if the first
     * argument is a pointer to a C string encoded as UTF-8.
     * It is also possible to modify arguments by assigning a
     * NativePointer object to an element of this array.
     * @param {object} state - Object allowing you to keep
     * state across function calls.
     * Only one JavaScript function will execute at a time, so
     * do not worry about race-conditions. However, do not use
     * this to store function arguments across onEnter/onLeave,
     * but instead use "this" which is an object for keeping
     * state local to an invocation.
     */
    onEnter: function onEnter(log, args, state) {
        log("recvfrom()");
    },

    /**
     * Called synchronously when about to return from recvfrom.
     *
     * See onEnter for details.
     *
     * @this {object} - Object allowing you to access state
     * stored in onEnter.
     * @param {function} log - Call this function with a string
     * to be presented to the user.
     * @param {NativePointer} retval - Return value represented
     * as a NativePointer object.
     * @param {object} state - Object allowing you to keep
     * state across function calls.
     */
    onLeave: function onLeave(log, retval, state) {
    }
}
```

现在，使用下面的代码代替log()

```
log("recvfrom(socket=" + args[0].toInt32()
    + ", buffer=" + args[1]
    + ", length=" + args[2].toInt32()
    + ", flags=" + args[3]
    + ", address=" + args[4]
    + ", address_len=" + args[5].readPointer().toInt32()
    + ")");
```

Save the file (it will be reloaded automatically) and perform some action in your Twitter application to trigger some network activity. You should now see something along the lines of:
保存文件（该文件将自动重新加载），然后在您的Twitter应用程序中执行某些操作以触发某些网络活动。 现在，您应该可以看到以下内容：

```
  8098 ms recvfrom(socket=70,
                        buffer=0x32cc018, length=65536,
                        flags=0x0,
                        address=0xb0420bd8, address_len=16)
```

That's nothing, though. The real magic happens when you start building your own tools using the Python API that frida-trace is built on top of.
没什么 当您开始使用在frida-trace构建于Python之上的Python API来构建自己的工具时，真正的魔力就会发生。

这里保留英文，总是不能明确的翻译出意思来。

Q&A

一个小问题的，在使用USb连接手机的时候执行 frida-trace -i “recv*” -i “read*” *twitter*会出现如下的错误，需要用下面的命令执行
Failed to attach: unable to find process with pid 3308

```
frida-trace -i "open"  -i "read*"  -U tv.danmaku.bili
```

修改log()之后出现下面的错误：（不清楚是否和模拟器有关）
{‘type’: ‘error’, ‘description’: ‘Error: access violation accessing 0x0’, ‘stack’: ‘Error: access violation accessing 0x0\n at frida/runtime/core.js:132\n at [anon] (input:28)\n at invokeCallback (/tracer.js:49)\n at /tracer.js:55’, ‘fileName’: ‘frida/runtime/core.js’, ‘lineNumber’: 132, ‘columnNumber’: 1}