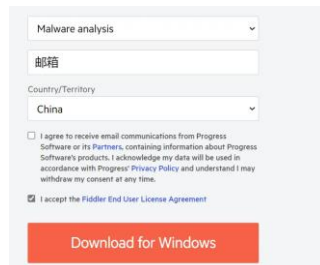


1 安装 fiddler

安装包下载链接: <https://www.telerik.com/download/fiddler>

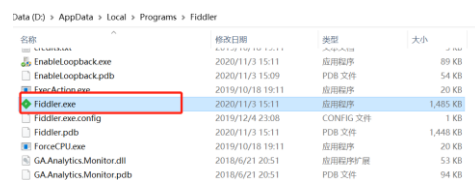
随便选个用途, 填写邮箱, 地区选择 China, 勾选“I accept the Fiddler End User License Agreement”, 点击“Download for windows”, 下载。



The image shows the Fiddler download form. It includes a dropdown menu for 'Malware analysis', an email input field labeled '邮箱', a dropdown menu for 'Country/Territory' set to 'China', a checkbox for 'I agree to receive email communications from Progress Software or its Partners...', and a checked checkbox for 'I accept the Fiddler End User License Agreement'. A red 'Download for Windows' button is at the bottom.

双击 FiddlerSetup.exe 安装 fiddler, 可以选择常用的、不那么深的一个路径。

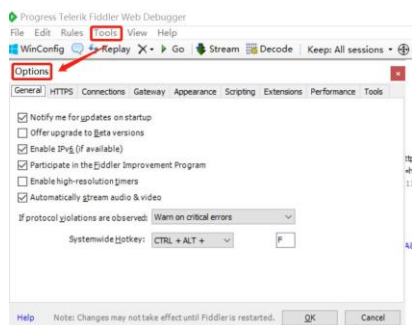
在安装路径下, 双击 Fiddler.exe, 能打开, 说明安装成功, 可以给 Fiddler.exe 创建一个桌面快捷方式。



2.fiddler 配置

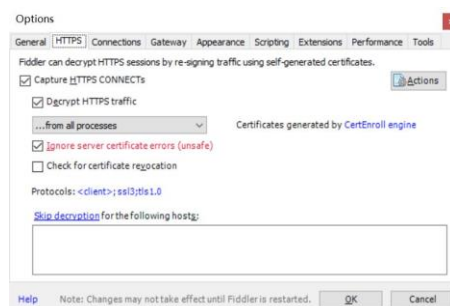
双击 Fiddler.exe, 弹出“AppContainer Configuration”对话框, 点击“cancel”就行。

Progress Telerik Fiddler Web Debugger 菜单栏中, Tools——Options。



HTTPS 的配置如下:

勾选 Capture HTTPS CONNECTs、Decrypt HTTPS traffic、Ignore server certificate errors(unsafe)。



点击 OK 保存。

弹出对话框“SCARY TEXT AHEAD: Read Carefully! ”，点击 YES。

弹出对话框“安全警告”，询问是否安装证书，点击是。

弹出对话框“Add certificate to the Machine Root List? ”，点击 YES。

弹出对话框“TrustCert Success”，点击确定。

再点击一下 options 中的 ok，以防忘记保存配置。

Decrypt HTTPS traffic 中的选项说明：

from all processes :

抓取所有的 https 程序，包括电脑程序和手机 APP。

from browsers only :

只抓取浏览器中的 https 请求。

from non-browsers only :

只抓取除了浏览器之外的所有 https 请求。

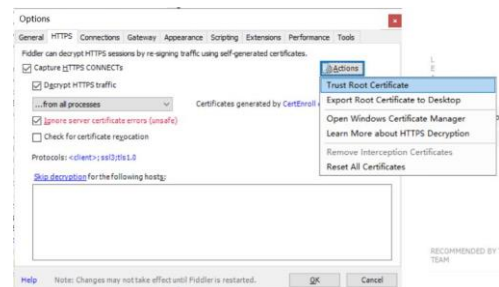
from remote clients only:

只抓取远程的客户端的 https 请求，就是只抓取手机 APP 上的 https 请求。

注意事项：

如果 HTTPS 请求出问题，例如，浏览器提示“您的链接不是私密链接”等，一般都是证书安装有问题，重新安装一遍证书，重复一遍 HTTPS 配置即可。

Options——HTTPS——Actions——Trust Root Certificate。

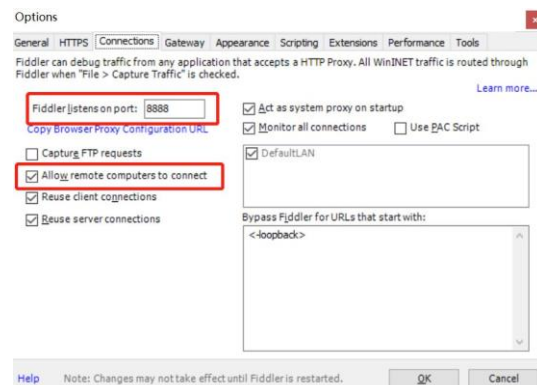


Connections 配置

Fiddler listens on port, 确保 fiddler 的端口为 8888。

勾选 Allow remote computers to connect。

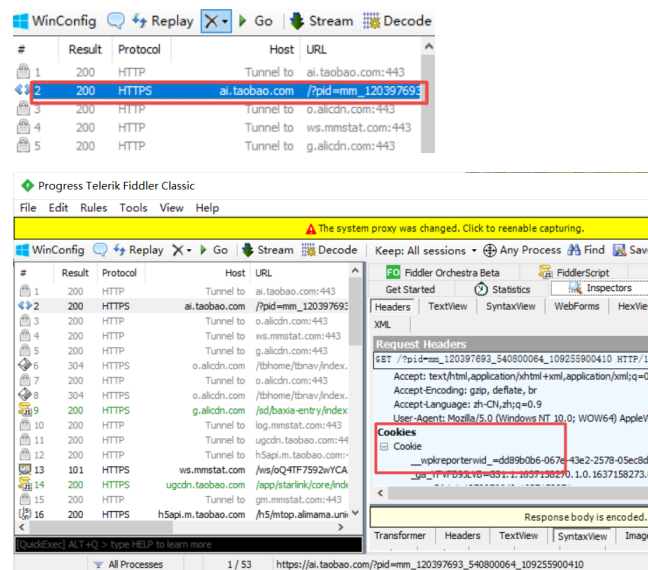
弹出对话框“Enabling Remote Access”对话框，点击确定。



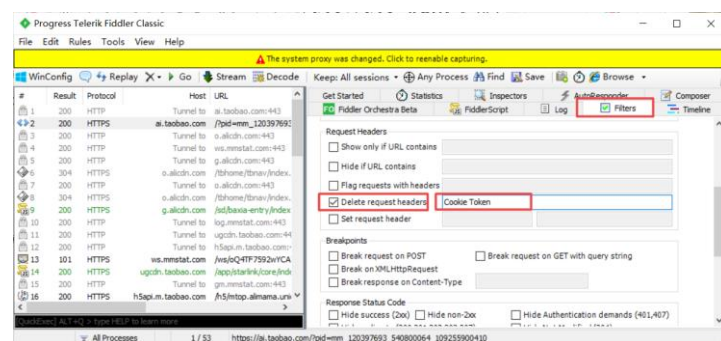
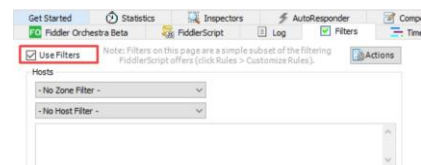
点击 OK。

3.开始抓包

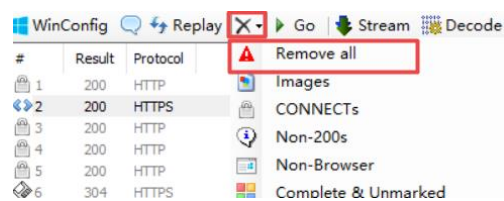
登录淘宝网，可以看到抓到的一些请求包，双击打开一个



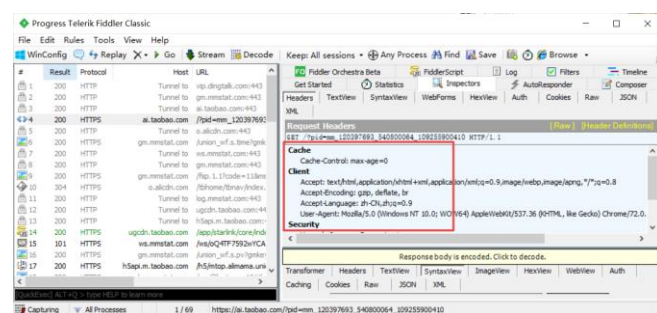
看到这里 cookie 敏感信息，将其过滤



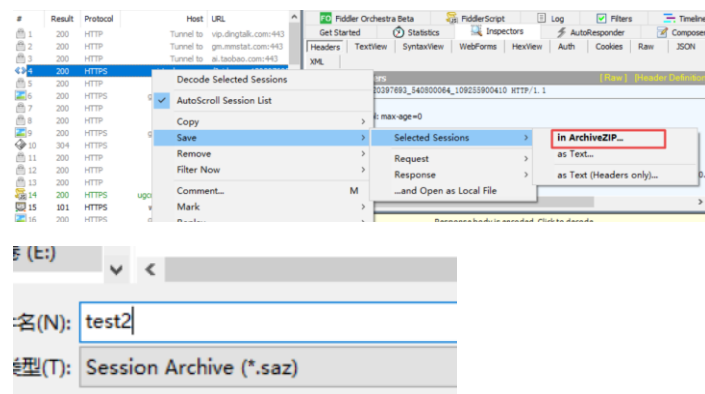
清除记录，再次访问该网站



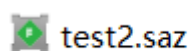
可以看到 cookie 已经没有了出现了



将这个请求包保存下来
右键点击

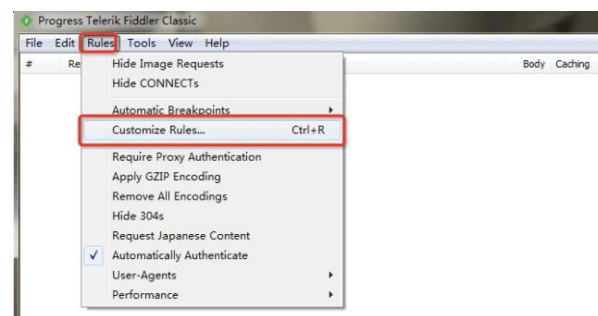


对应文件夹下面出现该包

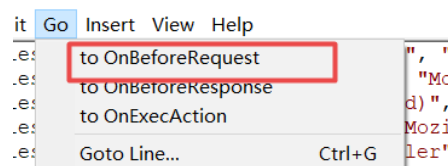


双击可以用 fiddler 打开

4 通过配置文件来删除 cookie



找到请求方法



配置

```
static function OnBeforeRequest(oSession: Session) {
    //路径中含有/?pid=mm_120397693_5..., 可以自己设置
    if(oSession.fullUrl.Contains("/?pid=mm_120397693_540800064_109255900410")){
        var fso;
        var file;
        fso = new ActiveXObject("Scripting.FileSystemObject");
        //文件保存路径, 可自定义 但是前提是已经创建了这个文件
        file = fso.OpenTextFile("F:\\zhuabaoRuannian\\myBao\\test.txt", 8, true, true);
        // 删除所有的cookie
        oSession.oRequest.headers.Remove("Cookie");
        file.WriteLine("Request url: " + oSession.url);
        file.WriteLine("Request header: " + "\n" + oSession.oRequest.headers);
        file.WriteLine("Request body: " + oSession.GetRequestBodyAsString());
        file.WriteLine("\n");
        file.close();
    }
    // Sample Rule: Color ASPX requests in RED
    // if (oSession.uri.Contains(".aspx")) { oSession["ui-color"] = "red"; }
```

同样没有 cookie,可以使用配置的 test.txt,也可以手动保存 saz 文件。
还有很多函数可以自行修改配置

一些常用的 oSession 函数和方法

```
// 请求 host
oSession.host == "my.test.com";
// 请求 host 之后的 url 是否包含
oSession.url.Contains("/feed") ;
// 获取响应内容的字符串
var logContent = oSession.GetResponseBodyAsString();
// 创建写入流
var sw : System.IO.StreamWriter;
if (System.IO.File.Exists(filename)){ //是否有该文件夹
    sw = System.IO.File.AppendText(filename); //有添加
    sw.Write(logContent);
}
else{
    sw = System.IO.File.CreateText(filename); //没有创建
    sw.Write(logContent);
}
sw.Close(); //关闭写入流
sw.Dispose(); //销毁写入流

// 修改 session 中的显示样式
oSession["ui-color"] = "orange";
// 移除 http 头部中的MQB-X5-Referer 字段
oSession.oRequest.headers.Remove("MQB-X5-Referer");
// 修改 http 头部中的Cache-Control 字段
oSession.oRequest["Cache-Control"] = "no-cache";
// 修改 host
oSession.host = "example.domain";
// 修改 Origin 字段
oSession.oRequest["Origin"] = "http://domain";
// 删除所有的 cookie
oSession.oRequest.headers.Remove("Cookie");
// 新建 cookie
oSession.oRequest.headers.Add("Cookie",
"username=cookienam;");
// 修改 Referer 字段
oSession.oRequest["Referer"] = "https://yoururl";

// 获取 Request 中的 body 字符串
var strBody=oSession.GetRequestBodyAsString();
// 用正则表达式或者 replace 方法去修改 string
strBody=strBody.replace("aaaa","bbbbbb");
// 将修改后的 body，重新写回 Request 中
```

```
oSession.utilSetRequestBody(strBody);  
// 判断连接中是否包含字符串 str  
oSession.uriContains(str)  
// 给连接请求添加一个字段 TEST  
oSession.oRequest["TEST"]="TEST NEW Request";
```