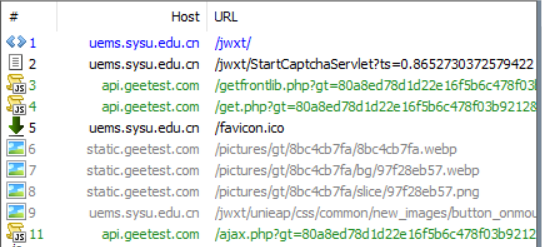
# 中山大学教务系统拼图验证调研文案

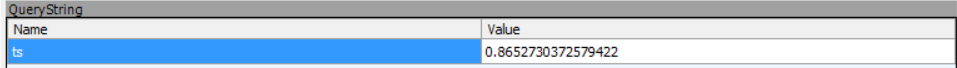
**抓包工具:Fiddler+Chrome**

选择好你的浏览器之后，访问[中山大学教务系统](http://link.zhihu.com/?target=http%3A//uems.sysu.edu.cn/jwxt/)，待首次加载完成后，再开启F12，并从地址栏重新进入网页抓包(非刷新)。这样做可以避免一大堆CSS和图片之类的无用包干扰视线，最后拉动滑块完成验证。于是大概会得到如下的结果：



全程是清一色的GET Request，而且可以注意到，除了被高亮了的四个package，其他的包都是没有带Query String的，所以分析的重点便落在了这四个包上。

首先看1号包，这个包只带有一个参数ts：

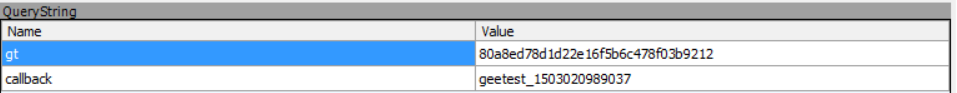


如果重复多次抓包，可以看到这个ts值每次都不一样，初步猜测是随机数，防止浏览器从缓存中直接返回结果。至于Response Body，从Fiddler中看到如下：

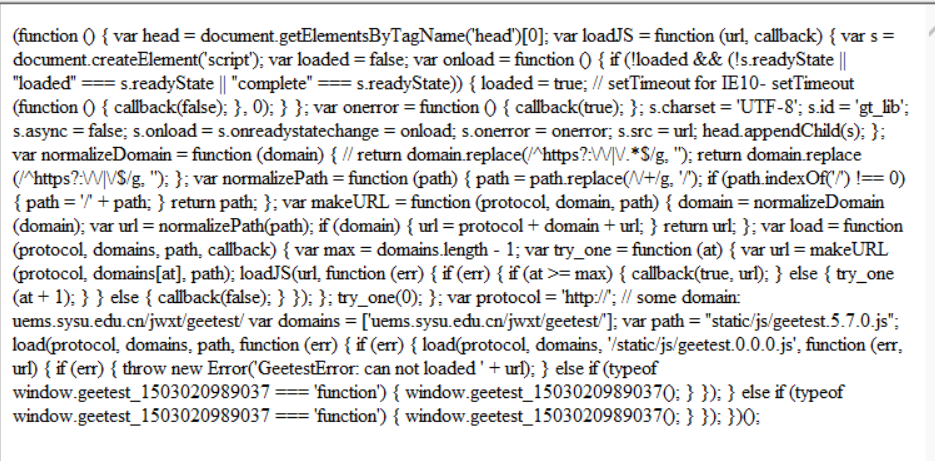


这是一段包含了三个值的JSON，其中的gt和challenge显然会用在后面的请求中。

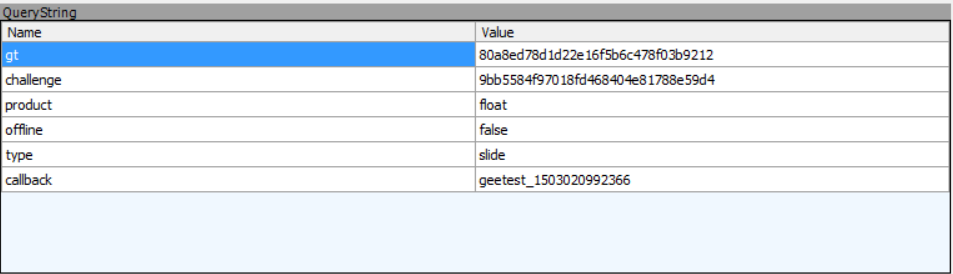
接着是2号包，这个包发送的参数如图：



其中gt便是我们刚才从1号包的回复里找到的值，至于callback，从名字推测是本地生成了名为geetest\_xxxx的函数，待请求完成后会回调这个函数，至于后面的xxxx是怎么来的暂时不清楚。猜测是随机数。

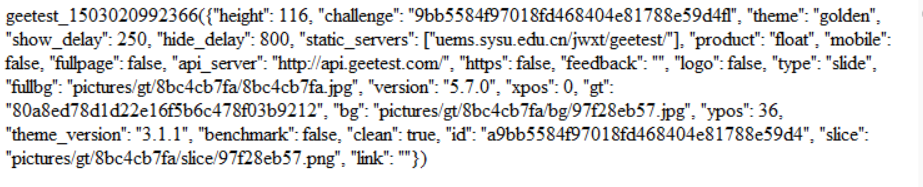
上图是Response Body部分，是一段没有被混淆的明文JavaScript。浏览后发现这段JavaScript是在执行一些初始化工作，并发出后面的请求。代码比较长这里不贴出来了，这也是这次破解里能见到的最后一段没有被混淆的JavaScript代码了

然后是3号包。刚才看的2号包实际上没有太多作用，返回的JavaScript中除了每次会更改要调用的回调函数名之外没有改变，并没有包含有效信息。3号包则不一样，其参数为：



callback应该也是回调函数的名称。challenge和gt是之前1号包得到的值。offline并不知道是什么，不过看上去并不重要，应该设置为false就好了吧。至于product，从官网上可以了解到它的验证码有多种嵌入模式，而中山大学教务系统采用了float的模式。所以product设置为float就错不了。

至于返回值，则包含了重要的信息：

这是在以这段JSON为参数，调用本地的回调函数。这段JSON里包含了三个图片地址：fullbg、bg和slice。分表访问其URL，得到的图片如下：

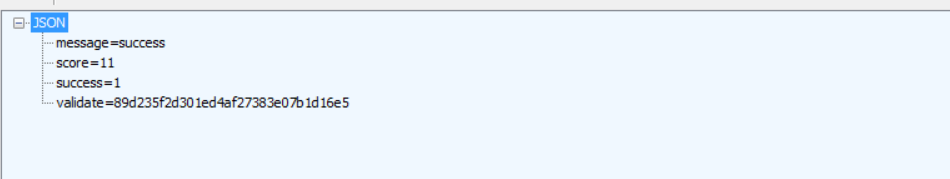






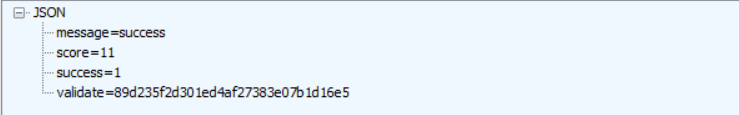
看得出来，slice并不在我们的兴趣范围内。fullbg和bg则分别是完整的背景和缺块的背景。这两张图片被打乱的方法是相同的，可见可以通过同一个算法还原。

另外，返回值里的staticservers和https可以帮我们还原出完整的图片地址。而剩余的对我们来说并没有太大意义。

最后是4号包，4号包明显是我们最后拖动滑块后触发的一个包，先看返回值：

总共4个返回值，其中score是用以显示“您击败了xx%的用户”的分数(这风格好像360)。而validate相信是在网页的实际登录过程中的一个参数。

接下来看参数：



这里总共有多达7个参数。其中callback、challenge和gt的含义我们都已经了解。

a是一段看上去毫无意义的乱码，但是想必是用以防破解的Hash之类。经后续调研结果，这里的a为鼠标轨迹数组加密后结果。

imgload，这个参数的含义暂时不清楚。

passtime，从名字便可以看出是指用户拉动滑块后用了多少时间完成，这也是返回值里score的来源。

userresponse，这个参数里包含了用户拉动滑块的数据。估计是用滑动距离加密得出。