目录

CTF-web 之 burp suite 使用 2
WEB 基础知识 2
（1）查看和修改 http 请求头 4
（2）intruder 载荷攻击 4
Burp Intruder 主要有四个选项卡组成: 5
Positions 的四种攻击模式 5
payload 设置方法 6
（3）decoder encode 8
例题(例题来源于其他博主) 9
猫抓老鼠 9
头有点大 10
localhost 允许 10

CTF-web 之 burp suite 使用

burp suite 使用

一般其是作为一个辅助工具，直接使用来解题的部分是少数，我们可以使用它来观察请求和响应，并且可以反复的提交，关键的是他还带有很多其他的功能，在我们做题的过程中，使用的关键点包括：

1. 页面和源码无特殊信息时，可以使用抓包观察

----有无特殊字段，泄露服务器或 flag 等信息

----对提交的 url 和数据进行观察

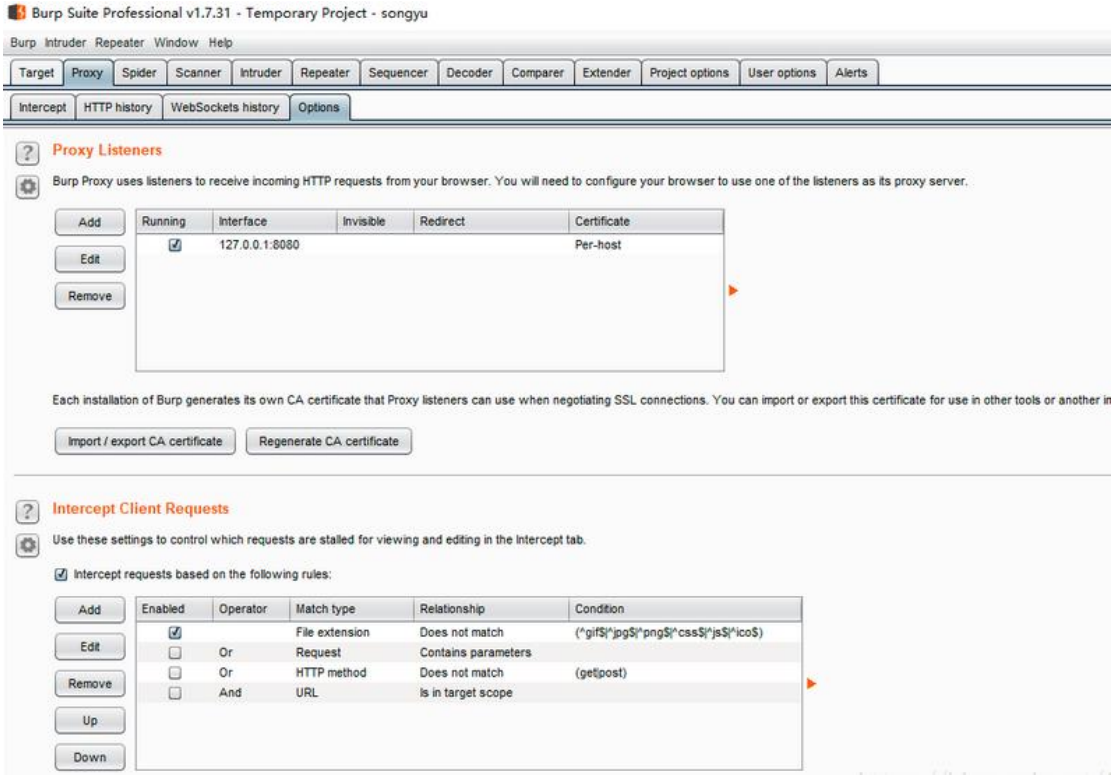
2. 使用 reapter 功能，重复的测试提交的数据，观察响应等
3. intruder 爆破功能，用来进行一些密码，验证码的爆破
4. 使用 request 的编辑功能，编辑头信息以达到题目要求

WEB 基础知识

```
1  HTML超文本标记语言，采用标签的方式分类元素，使用CSS层叠样式控制，另外可以插入js脚本，即<script>标签
2  HTTP超文本传输控制协议 应用层协议在tcp/ip之上，通过多个字段属性控制网页的访问，传输。
3
4  常用请求方式GET--> 参数形式 index.php?a=123&b=431
5  POST--> 附加在字段之后，空一行 然后附加数据
6
7  常见的字段作用就不详细讲解了。
8
9  我们使用Bp观察的目标就是提交的数据（get和post）和返回的响应，对于格式这里也不做详细的介绍，在web基础中对各字段已经做了介绍。
```

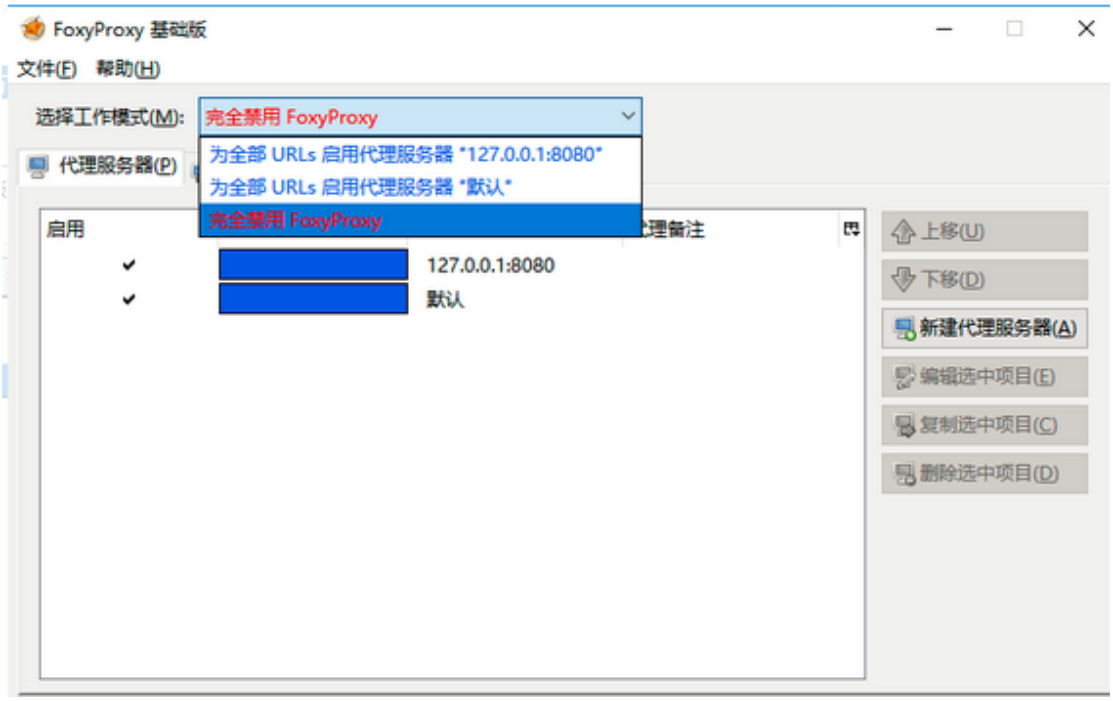
具体的看图文介绍

下面就是它的设置界面，默认的代理是 127.0.0.1 端口 8080

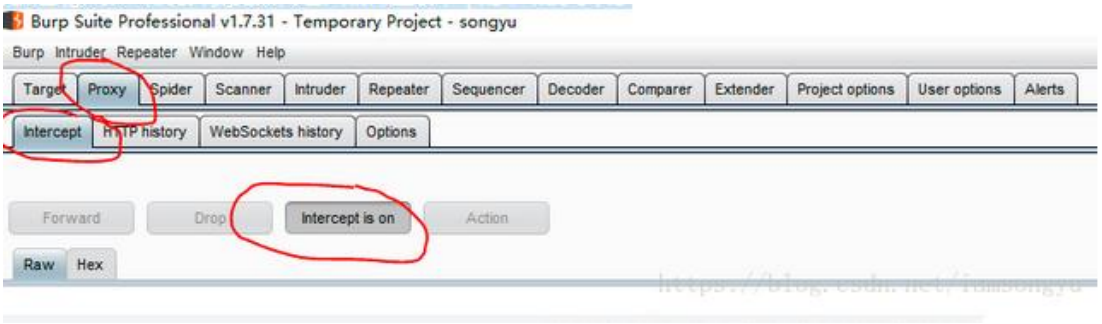


我们在火狐浏览器中添加插件 foxyproxy

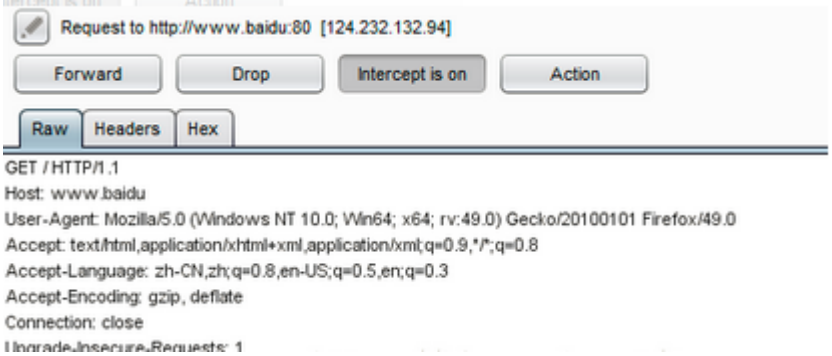
这样我们就可以方便的打开浏览器的代理，当然需要提前新建一下



那么如何开始抓包呢，第一部打开浏览器代理，然后在打开 Bp 的中断就可以了

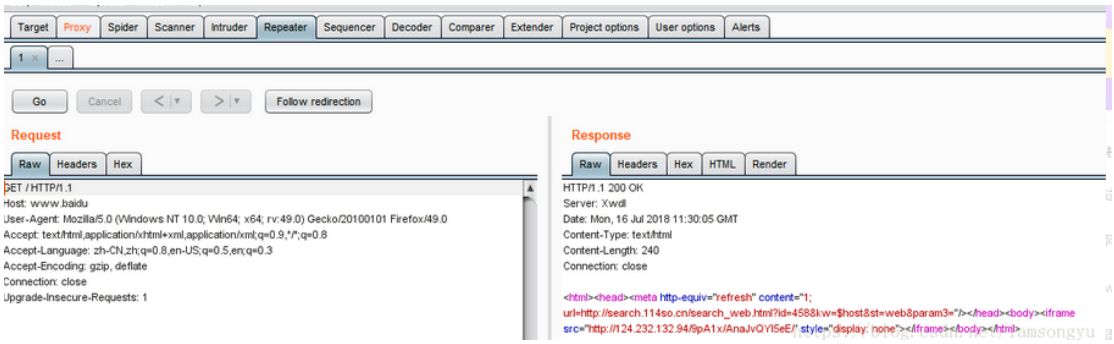


这样我们的发送和接受的数据就都会被拦截



Forward 是发送的意思，Drop 丢弃请求，Action 则是采取其他一些动作，比较实用之一的就是 send to repeater|intruder

允许我们不断修改和重复的一个请求，测试中非常有用。



我们可以实用 16 进制和字符的方式观察请求和响应头，支持随时修改，重复发送，并且返回信息不会进入到浏览器。按钮 go 就是发送信息。

（1）查看和修改 http 请求头

burp suite 配合火狐浏览器使用，将两者全部设为代理并打开断点，即可观察和修改数据

一般用于获取请求和响应中的特殊数据，或用于上传绕过等

使用方法：

（1）运行 Burp site,点击 Proxy 标签，确认 Options 选项卡下，Proxy listeners 的 running 运行正常（勾选状态为运行），如果端口打开失败，可能的原因是有程序

占用了该端口，点击 edit，在 local listener port:输入框输入一个未占用的端口，点击 update 即可。

（2）然后设置浏览器代理地址为 127.0.0.1，端口为所选端口，设置浏览器开始代理 并打开 burp suite 的 proxy-intercept 的 on 状态

（3）进入上传页面，选择我们的 asp 木马，点击上传就可以看到 burp suite 已经拦截在 proxy-intercept-Raw 就是原始数据 也可以 Hex 观察 16 进制数据

（4）鼠标对着 Raw 的内容右击，最后单击 Send To Repeater（包重放），修改之后点击 go 进行发送。

常见的有：

Referer 来源伪造

X-Forwarded-For：ip 伪造

User-Agent：用户代理（就是用什么浏览器什么的）

Accept-Language：语言 国家要求

Cookie 的修改

（2）intruder 载荷攻击

在我们需要大量构造载荷重复请求时 可以使用该插件，该插件可以定制数据类型，变化范围，以便进行大量的爆破工作，当然我们也可以使用编写脚本

Burp Intruder 主要有四个选项卡组成:

1：Target 用于配置目标服务器进行攻击的详细信息。

2：Positions 设置 Payloads 的插入点以及攻击类型（攻击模式）。

3：Payloads 设置 payload，配置字典

4：Opetions 此选项卡包含了 request headers，request engine，attack results ，grep match，grep_extrack，grep payloads 和 redirections。

Positions 的四种攻击模式

Sniper：这个模式会使用单一的 payload【就是导入字典的 payload】组。它会针对每个 position 中$$位置设置 payload。这种攻击类型适合对常见漏洞中的请求参数单独地进行测试。攻击中的请求总数应该是 position 数量和 payload 数量的乘积。

【一组 payload 独立测试每个位置，互相没关系】

Battering ram – 这一模式是使用单一的 payload 组。它会重复 payload 并且一次把所有相同的 payload 放入指定的位置中。这种攻击适合那种需要在请求中把相同的输入放到多个位置的情况。请求的总数是 payload 组中 payload 的总数。简单说就是一个 playload 字典同时应用到多个 position 中。

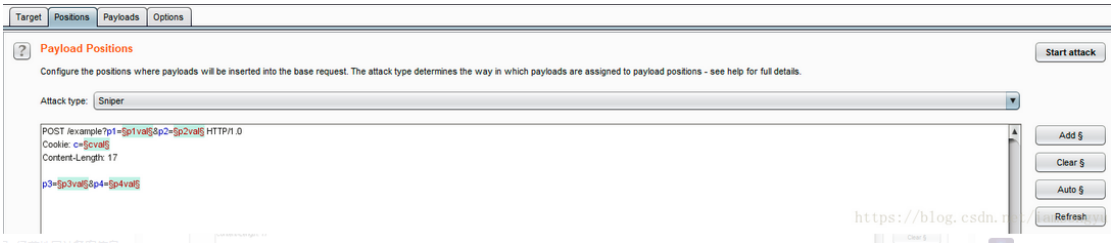
【一组 payload 同步测试所有位置（每个位置都填相同的）】

Pitchfork – 这一模式是使用多个 payload 组。对于定义的位置可以使用不同的 payload 组。攻击会同步迭代所有的 payload 组，把 payload 放入每个定义的位置中。

【多位置，每个位置的 payload 是一对一的，即两组 payload 的序号是同步增加】

Cluster bomb – 这种模式会使用多个 payload 组。每个定义的位置中有不同的 payload 组。攻击会迭代每个 payload 组，每种 payload 组合都会被测试一遍。

【多位置，对于两个位置的 payload，迭代所有的可能组合】

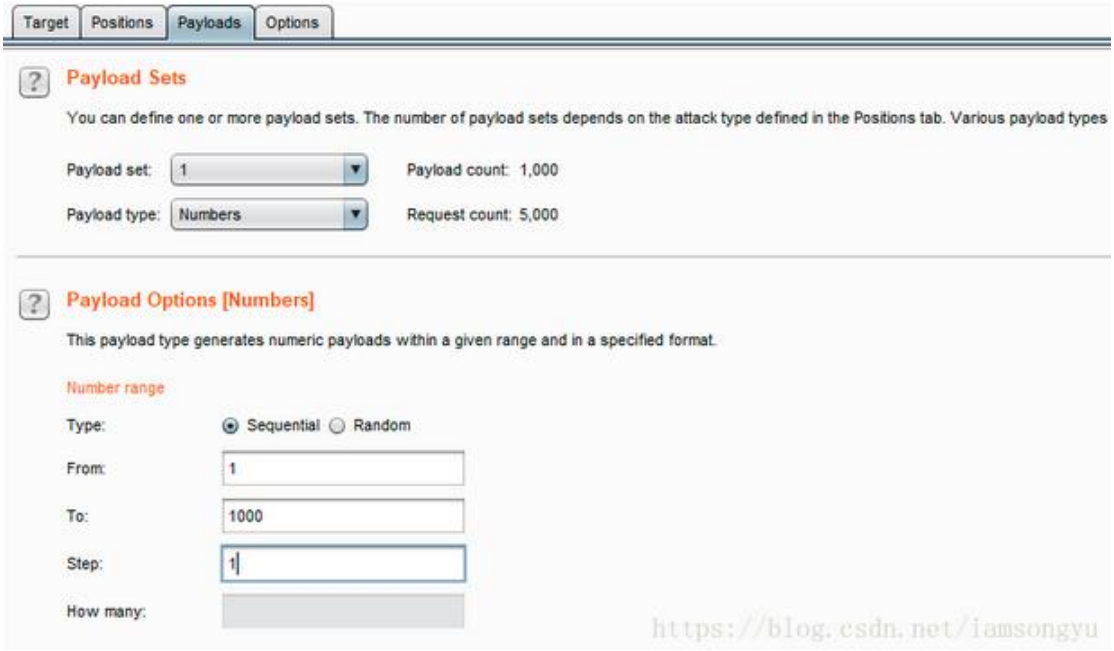


其中$$之间所夹的信息就是在测试中不断被替换的信息，我们可以编辑需要测试的位置，右侧有四个按钮，也可以手动敲上去或者删除。

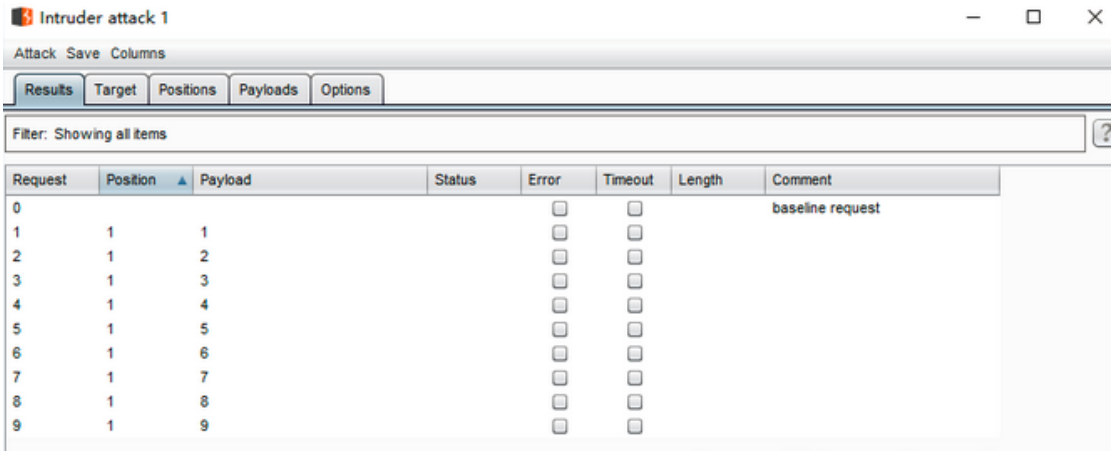
payload 设置方法

关于载荷的设置分为多种不同类型的数据，而后可以选择的范围也会随之变化，初始设置为 Payload set 和 Payload type

对于常用的数字型，我们可以选择起始 From 和终止 To 数字，步长 Step 等,而后 number format 的 Base 会选择进制和 interger digits 整数位数和 fraction digits 小数位数等

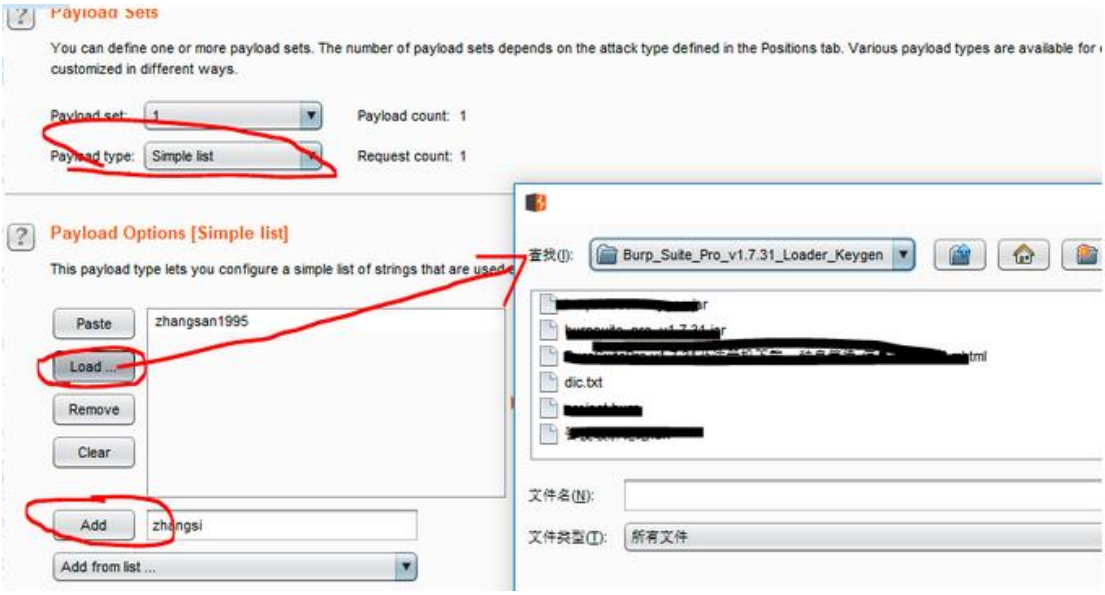


点击右上角的 start attack 之后，程序就开始运行了，中间我们可以观察每次相应的部分信息，点击可以查看详细的信息。一般我们通过观察长度判断是否达到了目的，因为此时的长度与其他的不同。（下图只是示意，并不是实战中的）



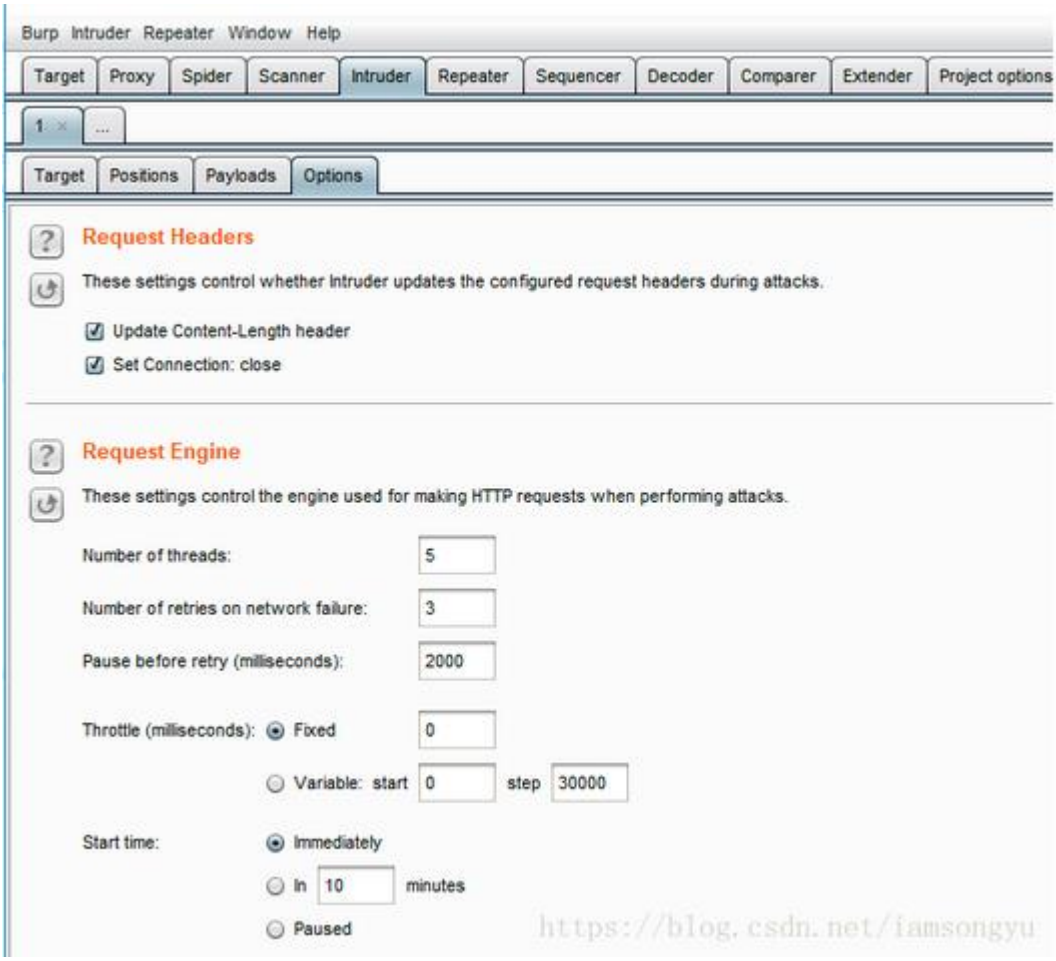
2018.10.24 添加

还有一个常用的破解弱口令之类的字典方法

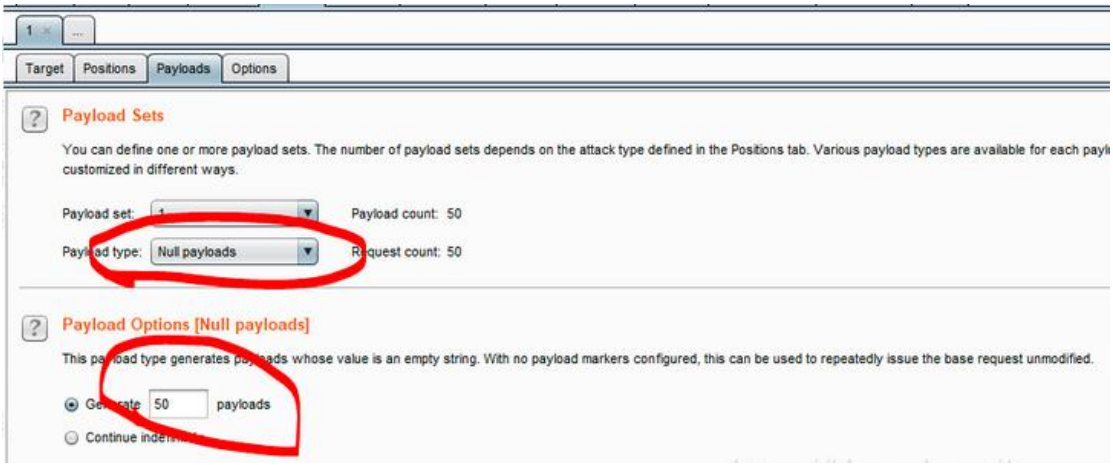


我们可以通过 ADD 增加自己编辑的字符串，可以通过 paste 粘贴复制的字符串，可以通过 load 来读取字典，这时候就需要网上的强大的字典了 dic.txt.

option 选项卡包含线程设置，可用于条件竞争



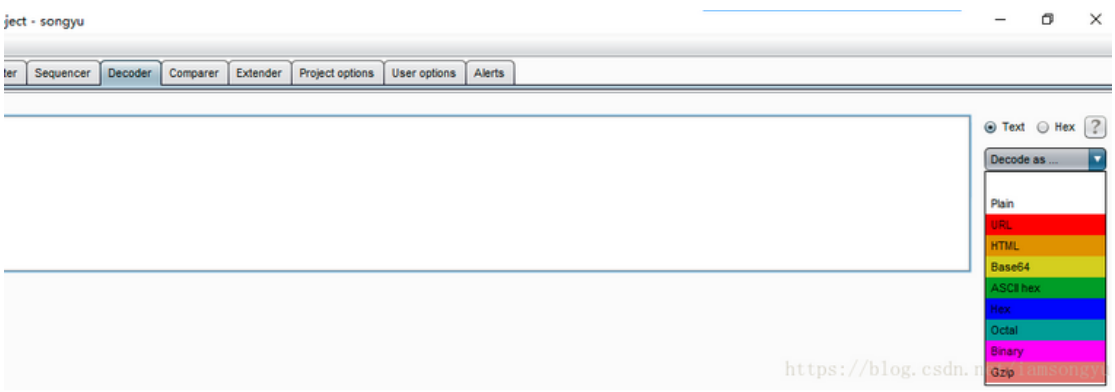
numbers of thread 设置线程数目，一般条件竞争时候我们是用不上 payload 的，此时一般设置为



选择 null payloads 方式，此时由于无法依靠 payload 数目来定义请求数，需要手动输入 generate 来生成指定的数目的请求

（3）decoder encode

加密解密用，各种各样。



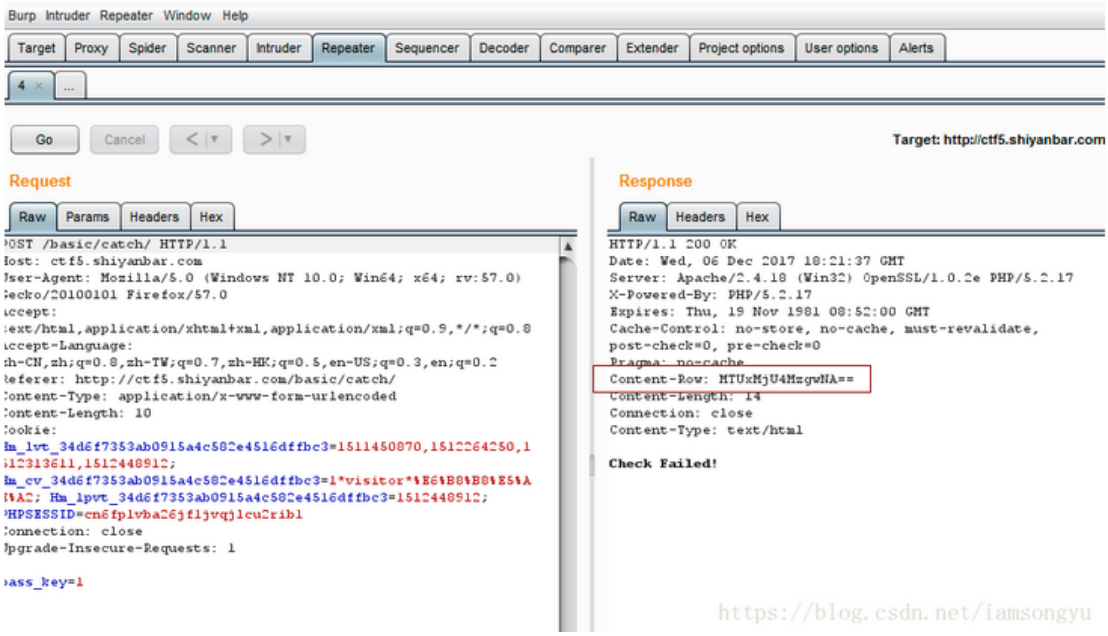
下面随意举两个例题，大多数的 Bp 的使用都是很多题中的一个小小的步骤，在这里就不详细的讲解了，在看其他题目的时候也会涉及不少的 Bp 的使用

例题(例题来源于其他博主)

猫抓老鼠

http://ctf5.shiyanbar.com/basic/catch/

网页有一个输入框，直接让你输入 key，源码没什么情况，抓包可以看到许多提交的数据

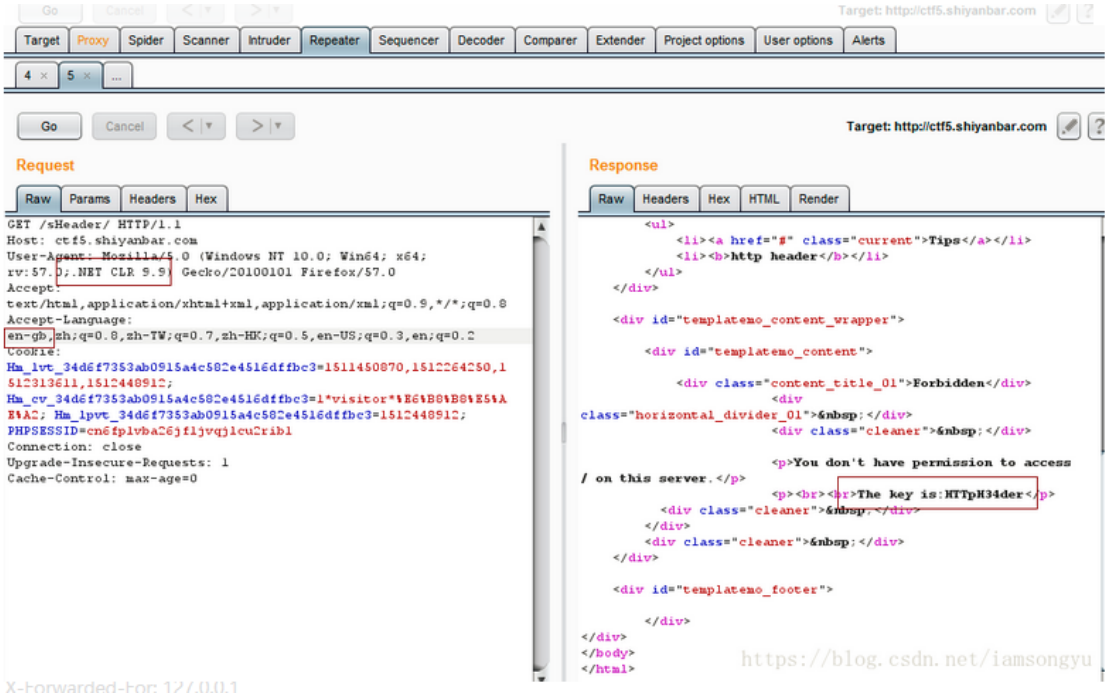


比较引人瞩目的就是提交的 passkey=1，响应中有 Content-Row: MTUxMjU4MzgwNA==，替换 passkey 的值，request 后得到 flag。

头有点大

http://ctf5.shiyanbar.com/sHeader/

要求我们使用 framework 9.9 才可以访问，使用 bp 修改头部信息，



得到 flag（.NET CLR 9.9）

localhost 允许

X-Forwarded-For: 127.0.0.1