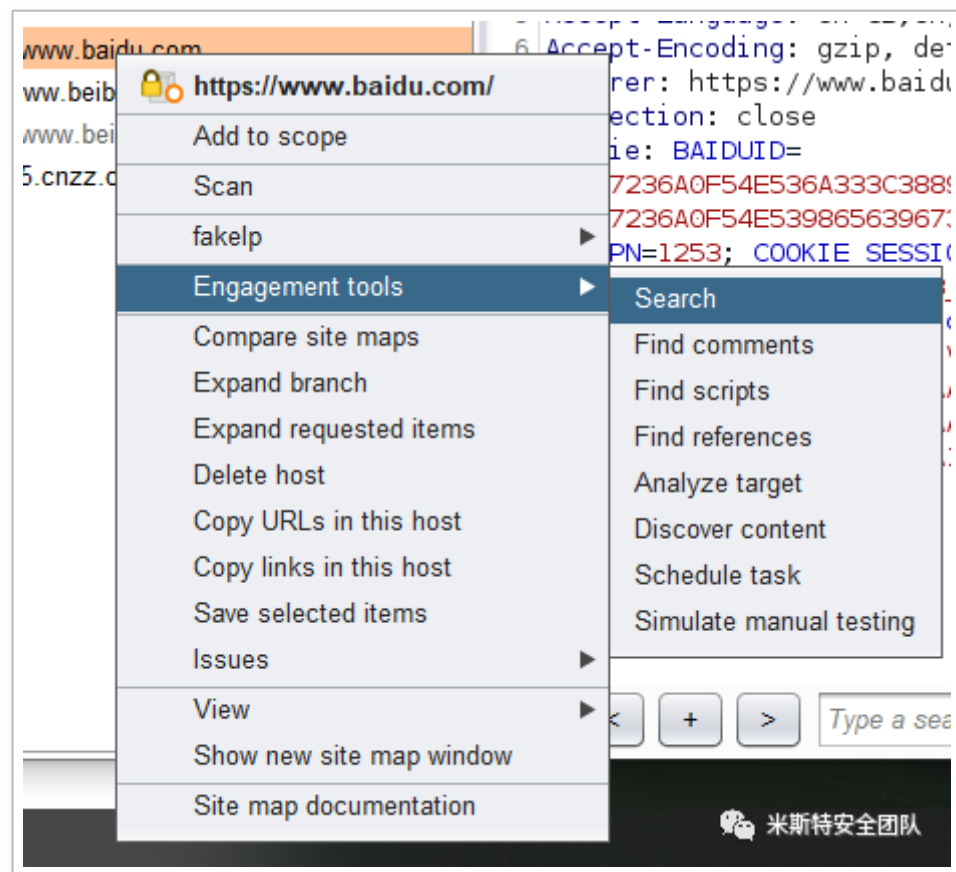


BurpSuite 系列 | 基础技巧（一）

一、浅析 Burpsuite 作战工具

直入 burp 右键菜单中的 Engagement tools，中文翻译为作战工具。



从上往下依次为

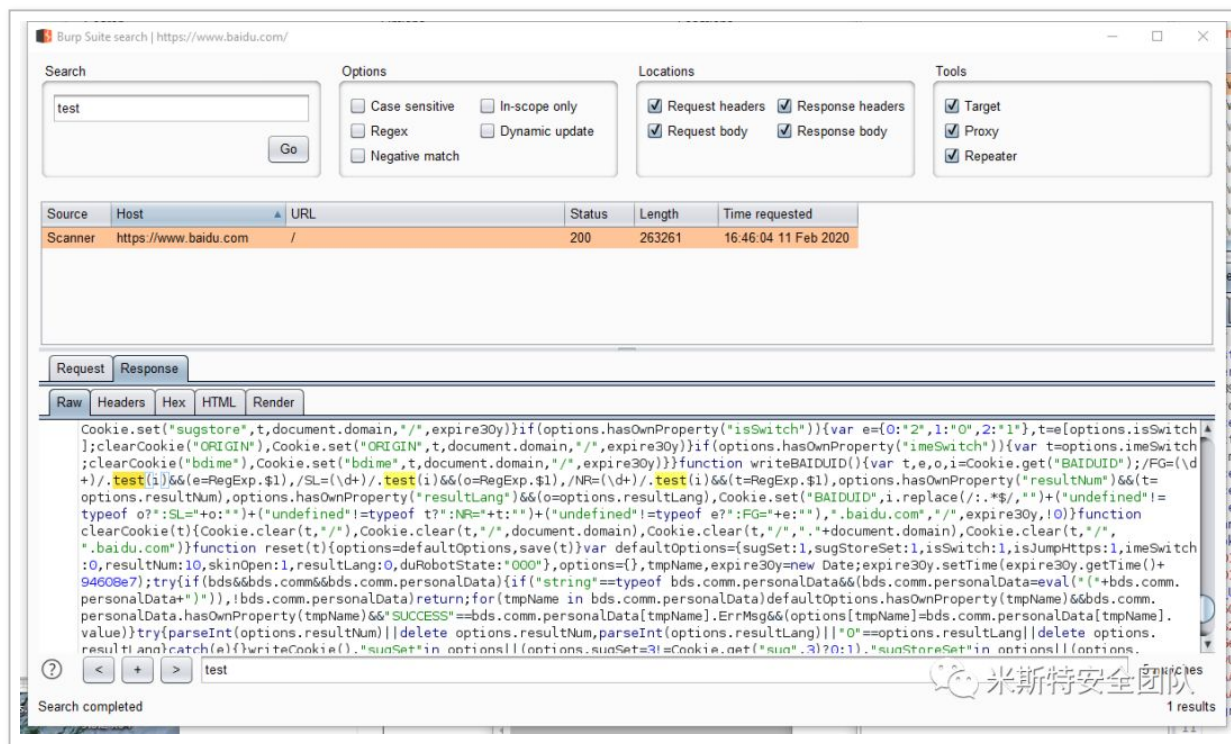
Search	搜索
Find Comments	查找注释
Find scripts	查找 js 代码
Find references	查找引用
Analyze target	分析目标
Discover content	内容勘测
Schedule task	定时任务
Simulate manual testing	人工模拟

以上翻译属于个人理解，非字面意思。如有不同见解，欢迎交流。

依次聊一下个人的使用心得：

0x01 Search 搜索

在选定的上右键打开后默认如下：

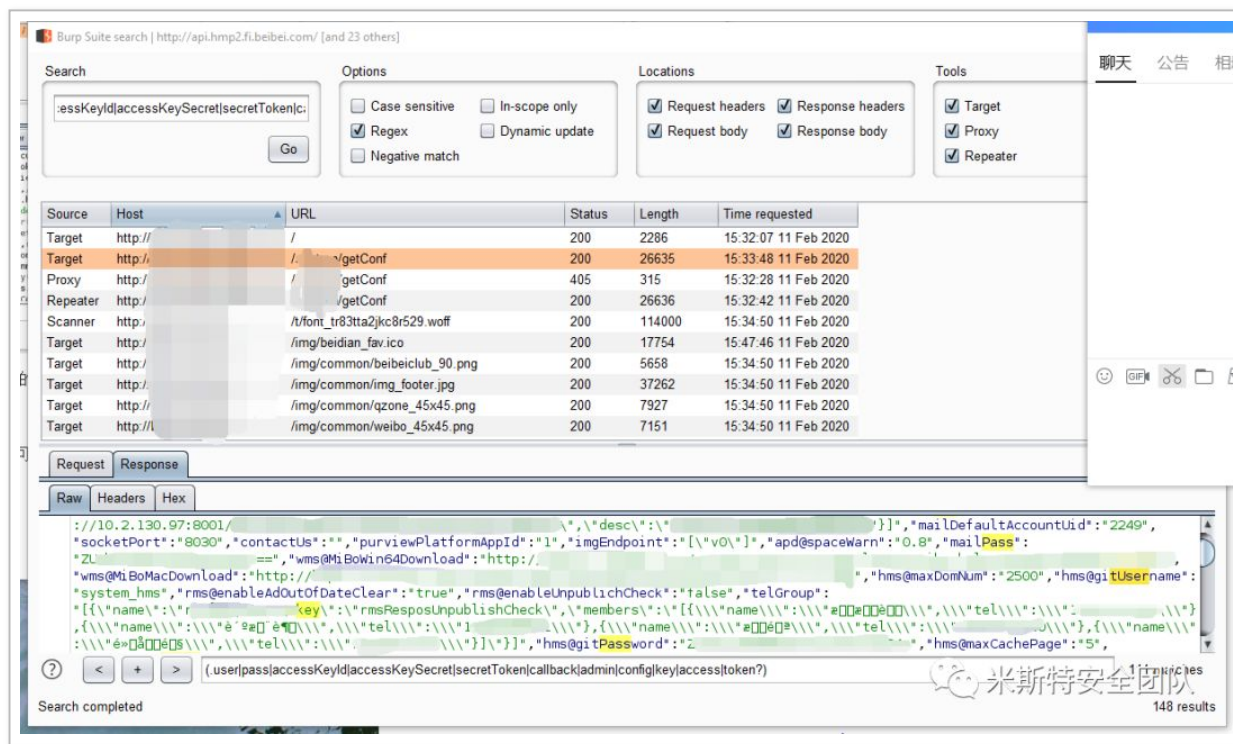


输入字符串即可查找需要的字符串，相当于浏览器 view-source 后再 Ctrl + F 查找字符串。

如果只用到这里的话有点不太黑阔，正则搜索这么强大，不用可惜了。

贴一条自己常用的正则：

```
(.user|pass|accessKeyId|accessKeySecret|secretToken|callback|admin|config|key|access|token?)
```



擅用该功能，在 SRC 捡洞可能可能只是一个习惯而已。（比如每次准备关 Burp 之前考虑全选 site map 中所有的站点，再拿正则去尝试一下。）

0x02 Find Comments 查找注释

选择目标右键打开后如下图：

Comments search | https://www.baidu.com/

Search

☐ Dynamic update

Export comments

Source	Host	URL	Item	Time requested
Scanner	https://www.baidu.com	/	STATUS OK, <div class="...	16:46:04 11 Feb 2020
Scanner	https://www.baidu.com	/s?ie=utf-8&f=8&rsv_bp=1&rsv_idx=2&ch=&t...	STATUS OK, opener.xxx, ...	17:20:17 11 Feb 2020
Scanner	https://www.baidu.com	/s?wd=aliyun+osskey&rsv_spt=1&rsv_iqid=0...	STATUS OK, opener.xxx, ...	17:19:28 11 Feb 2020

Comments

Request

Response

1 STATUS OK

2

3 opener.xxx

4

5 * @description 鍥剧煡base64鐮稿

6 * @author lizhouquan

7

8 base64

9

10 ;

11

12 base64

13

14 list鐮岀殑鐮稿鐮稿load

15

16 dom

?

<

+

>

Type a search term

0 matches

Search completed

米斯特安全团队

5 results

Comments search | https://www.baidu.com/

Search ☐ Dynamic update Export comments

Source	Host	URL	Item	Time requested
Scanner	https://www.baidu.com	/	STATUS OK, <div class="...	16:46:04 11 Feb 2020
Scanner	https://www.baidu.com	/s?ie=utf-8&f=8&rsv_bp=1&rsv_idx=2&ch=&t...	STATUS OK, opener.xxx, ...	17:20:17 11 Feb 2020
Scanner	https://www.baidu.com	/s?wd=aliyun+osskey&rsv_spt=1&rsv_iqid=0...	STATUS OK, opener.xxx, ...	17:19:28 11 Feb 2020

Comments Request Response

Raw Headers Hex HTML Render

```
16 Traceid: 1581412820043964698617197811250965361531
17 Vary: Accept-Encoding
18 X-Ua-Compatible: IE=Edge,chrome=1
19 Connection: close
20 Content-Length: 280704
21
22 <!DOCTYPE html>
23 <!--STATUS OK-->
24
25
26
27
28
29
```

81 highlights

Search completed

米斯特安全团队

可以理解为 Search 功能的正则写为:

```
<!--(.*?)-->
```

不同语言的注释不一样，burp 内置的正则更为全面。测试过程中适当看一下注释内容，没准一个未授权就出来了。

0x03 Find scripts 查找 js 代码

选择目标右键打开后如下图：

Scripts search | http://api.hmp2.fi.beibei.com/ [and 27 others]

Search

☐ Dynamic update

Export scripts

Source	Host	URL	Item	Time requested
Scanner	https://[REDACTED].com	/kg/m-base/2.0.0/index.js	!function(e){function t(){var ...	17:19:44 11 Feb 20...
Scanner	https://[REDACTED].com	/kissy/kimi/6.0.1/kimi-min.js	/*! Copyright 2015, KIMI v6...	17:19:44 11 Feb 20...
Target	https://[REDACTED].com	/secdev/entry/index.js?t=219640	(function(a,e,t,i,n,r,o){if(e._...	17:19:45 11 Feb 20...
Target	https://[REDACTED].com	/secdev/nsw/1.0.64/ns_c_75_3_f.js	!function(){function e(e,a){f...	17:19:46 11 Feb 20...
Target	https://[REDACTED].com	/secdev/sufei_data/3.8.3/index.js	!function(n,t,r,i,a,o,e,c,u,f,...	17:19:46 11 Feb 20...
Scanner	https://[REDACTED].com	/	<ul style="text-indent: {{le...	15:56:51 11 Feb 20...
Scanner	https://[REDACTED].com	/system/resources/morning-ui.js	(function webpackUniversa...	15:56:52 11 Feb 20...
Scanner	https://m.aliyun.com	/yunqi/zt/491675	var _czc = _czc []; _czc...	17:19:42 11 Feb 20...
Target	http://[REDACTED].com	/assets/libs/common-v1.0.0.js	/* Zepto 1.2.0 - zepto even...	15:47:46 11 Feb 20...
Target	http://[REDACTED].com	/script/production/lib.js	"object"!typeof ISON&&/	15:34:50 11 Feb 20...

Scripts

Request

Response

1

2

3

4

5

6

7

```
var _czc = _czc || [];  
_czc.push(["_setAccount", "1256835944"]);  
_czc.push(["_setCustomVar", "穗垮三鐵ヲ善四5", 2]);
```

?

<

+

>

Type a search term

0 matches

Search completed

米斯特安全团队

26 results

Scripts search | http://api.hmp2.fi.beibei.com/ [and 27 others]

Search ☐ Dynamic update Export scripts

Source	Host	URL	Item	Time requested
Scanner	https://	/kg/m-base/2.0.0/index.js	!function(e){function t(){var ...	17:19:44 11 Feb 20...
Scanner	https://	/kissy/kimi/6.0.1/kimi-min.js	/*! Copyright 2015, KIMI v6...	17:19:44 11 Feb 20...
Target	https://	/secdev/entry/index.js?t=219640	(function(a,e,t,i,n,r,o){if(e._...	17:19:45 11 Feb 20...
Target	https://	/secdev/nsw/1.0.64/ns_c_75_3_f.js	!function(){function e(e,a){f...	17:19:46 11 Feb 20...
Target	https://	/secdev/sufei_data/3.8.3/index.js	!function(n,t,r,i,a,o,e,c,u,f,...	17:19:46 11 Feb 20...
Scanner	https://	/	<ul style="text-indent: {{le...	15:56:51 11 Feb 20...
Scanner	https://	/system/resources/morning-ui.js	(function webpackUniversa...	15:56:52 11 Feb 20...
Scanner	https://	/yunqi/zt/491675	var _czc = _czc []; _czc...	17:19:42 11 Feb 20...
Target	http://	/assets/libs/common-v1.0.0.js	/* Zepto 1.2.0 - zepto even...	15:47:46 11 Feb 20...
Target	http://	/script/production/lib.js	"object"!function ISOM&&f...	15:34:50 11 Feb 20...

Scripts Request Response

Raw Headers Hex HTML Render

```
src="//g.alicdn.com/??aliyun/dblx/0.0.4/es5-shim.min.js,aliyun/dblx/0.0.4/es5-sham.min.js"></scri
pt>
43 <![endif]-->
44 <link rel="stylesheet" href="//g.alicdn.com/aliyun/m-aliyun-yunqi/1.0.52/css/aggregation.css"
charset="utf-8" />
45 <script nonce="FVQ69B5EE2">
46   var _czc = _czc || [];
47   _czc.push(["_setAccount", "1256835944"]);
48   _czc.push(["_setCustomVar", "访客来源H5", 2]);
49 </script>
50 <body data-spm="11156470"><script type="text/javascript"
51 id="beacon-aplus"
52 src="//a.alicdn.com/alilog/mlog/aplus v2.is"

2 highlights
```

Search completed

米斯特安全团队

26 results

该功能处理.js 后缀的文件之外，页面中 script 标签的内容也会自动匹配出来。等同于右键看完当前页面的js 逻辑后在接着 Ctrl F 搜.js 然后继续看代码，在 Burp 里相对方便得多。

0x04 Find references 查找引用

选择目标右键打开后如下图：

Source	Host	URL	Status	Length	Time requested
Scanner	https://www.baidu.com	/	200	263261	16:46:04 11 Feb 2020
Scanner	https://www.baidu.com	/s?ie=utf-8&f=8&rsv_bp=1&rsv_idx=2&ch=&t...	200	281335	17:20:17 11 Feb 2020
Scanner	https://www.baidu.com	/s?wd=aliyun+osskey&rsv_spt=1&rsv_iqid=0...	200	275645	17:19:28 11 Feb 2020

Request Response

Raw Headers Hex HTML Render

```
100     <meta http-equiv="content-type" content="text/html; charset=utf-8">
101     <meta content="always" name="referrer">
102     <meta name="theme-color" content="#2932e1">
103     <link rel="shortcut icon" href="/favicon.ico" type="image/x-icon" />
104     <link rel="icon" sizes="any" mask href="
//www.baidu.com/img/baidu_85beaf5496f291521eb75ba38eacbd87.svg">
105     <link rel="search" type="application/opensearchdescription+xml" href="
/content-search.xml" title="百度搜索">
106
107
108 <title>accessKeyId. accessKeySecret. secretToken. expireTime. callback 百度搜索</title>
```



Type a search term

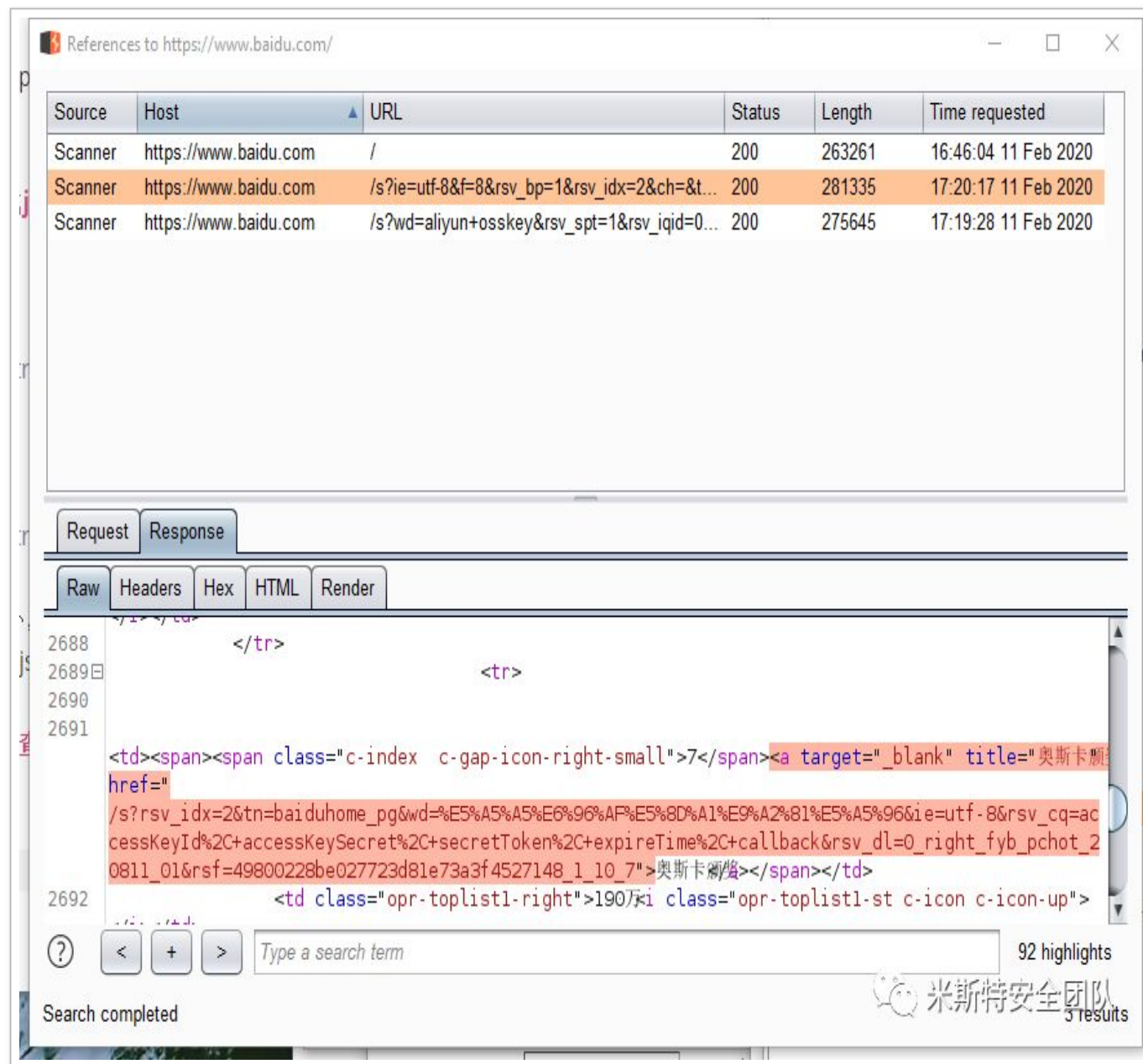
92 highlights

Search completed



米斯特安全团队

5 results



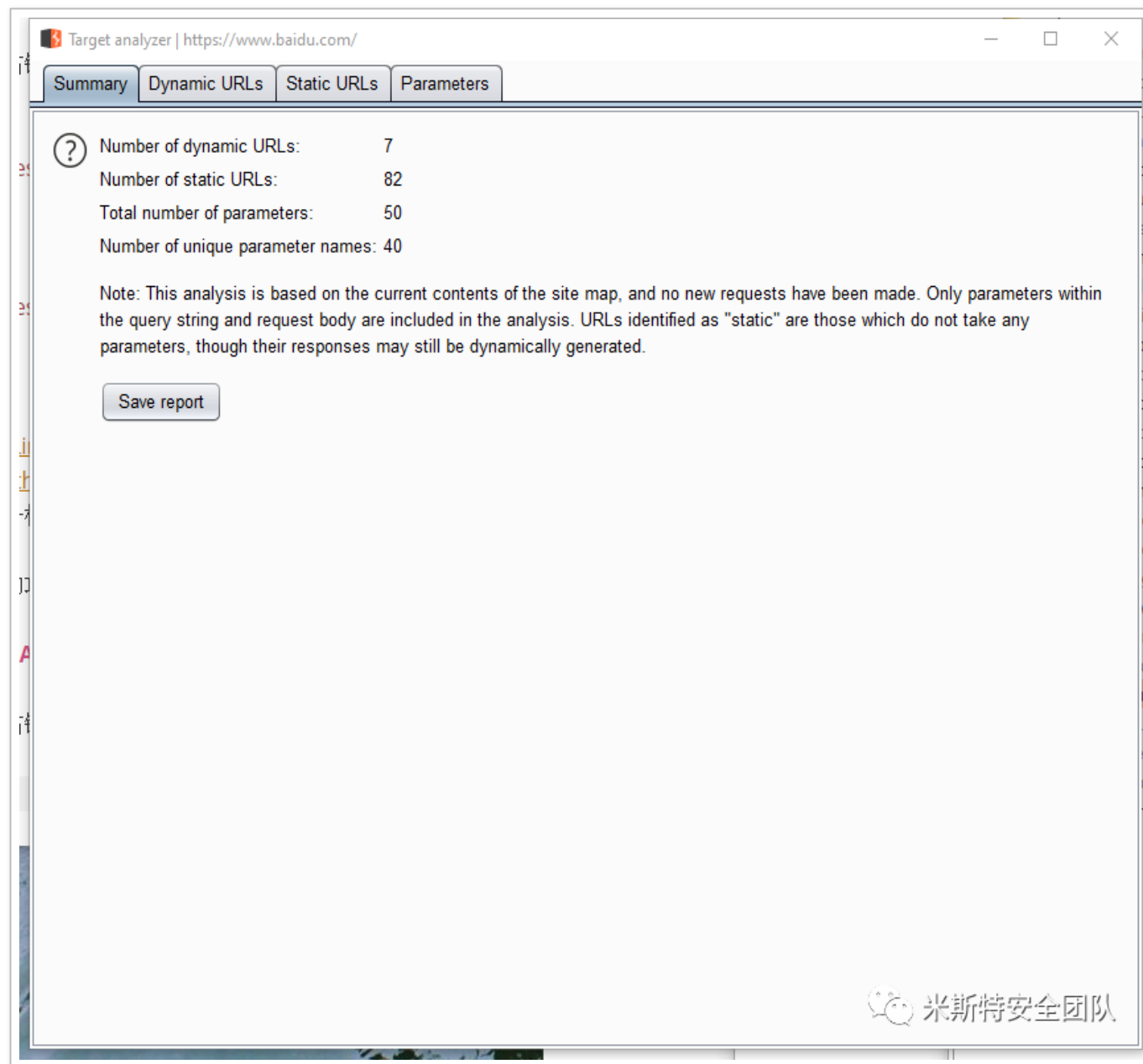
如果用过 LinksDumper 的话会发现 Burp 这个功能和它一模一样，都是匹配出页面中存在的各类链接。

默认自带的功能还有不少人是去用插件或者用独立的程序跑。burp 自带的它不香咩？

0x05 Analyze target 分析目标

选择目标右键打开后总共有四个页面

1. 总结页面：



假设拿到 100 个目标，筛选软柿子的时候通过总结页面判断一下，寻找交互点多的站点 / 链接进行单点突破，未尝不是一个高效的选择。

2. 动态 Urls:

The screenshot shows the 'Target analyzer' window for the URL <https://www.baidu.com/>. The 'Dynamic URLs' tab is selected, displaying a table of dynamic URLs found on the page. A red arrow points to the 'Params' column header.

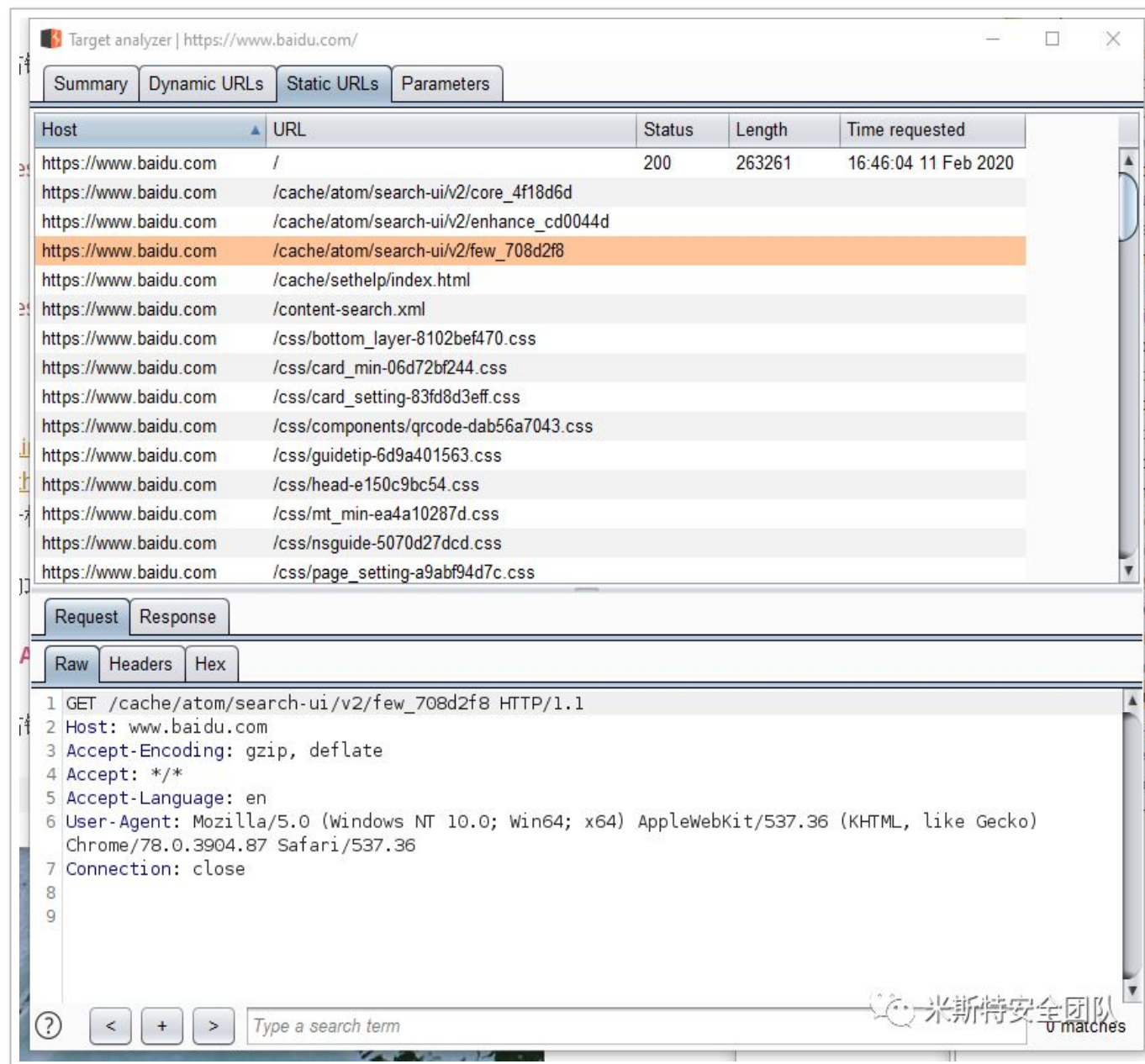
Host	URL	Method	Params
https://www.baidu.com	/s	GET	30
https://www.baidu.com	/sf/vsearch		11
https://www.baidu.com	/home/page/data/pageserver	GET	3
https://www.baidu.com	/page/data/pageserver		2
https://www.baidu.com	/ups/submit/addtips/		2
https://www.baidu.com	/link	GET	1
https://www.baidu.com	/v.gif		1

Below the table, the 'Request' tab is selected, showing a 'Raw' view of the request. The content is currently empty.

At the bottom of the window, there is a search bar with the placeholder text 'Type a search term' and a search button. To the right of the search bar, there is a watermark logo and the text '米斯特安全团队' (Mist Security Team) and '0 matches'.

该功能可以快速筛选出带外部参数的 url，并统计了参数的数量，秉着交互点越多可能存在的问题就越多的原则，筛选一下，效率 ++。

3. 其他 Url:



该功能可以理解为 Find references 功能遍历出来所有的连接后再去除 Dynamic URLs 中带交互点的 Url 的结果。

4. 参数统计:

Target analyzer | https://www.baidu.com/

Summary Dynamic URLs Static URLs Parameters

Name	Number of URLs
rsv_bp	2
rsv_pq	2
rsv_spt	2
rsv_t	2
tn	2
wd	2
_t	1
_t1581410765412	1
bar	1
bsst	1

Host	URL	Method	Params	Value [rsv_bp]
https://www.baidu.com	/s	GET	30	0
https://www.baidu.com	/sf/vsearch		11	1

Request Response Parameters

Raw Params Headers Hex

```
1 GET /sf/vsearch?pd=video&tn=vsearch&lid=e3f83a77000c8a1a&ie=utf-8&wd=%E9%98%BF%E9%87%8C%E4%BA%91+osskey&rsv_spt=7&rsv_bp=1&f=8&oq=%E9%98%BF%E9%87%8C%E4%BA%91+osskey&rsv_pq=e3f83a77000c8a1a&rsv_t=5c95mZ4CjMQ3zoiBOITfOum4MJSDkkVbK9n2PEL7HILjtXwg2MawMmR4rawJ4I6sLLTj HTTP/1.1
2 Host: www.baidu.com
```

米斯特安全团队

首先是统计了出现过那些参数，其次是这些参数在 Url 里面出现过次数。有了这个功能在瓶颈渗透提取本站存在的参数时根本不需要第三方插件工具辅助，直接复制 name 字段即可；某个参数存在漏洞后也可通过该功能查找其他可能存在相同问题的 Url。

0x06 Discover content 内容勘测

该功能秒杀一切目录扫描工具。

下面是关于如何巧用它去递归扫描站点备份的：

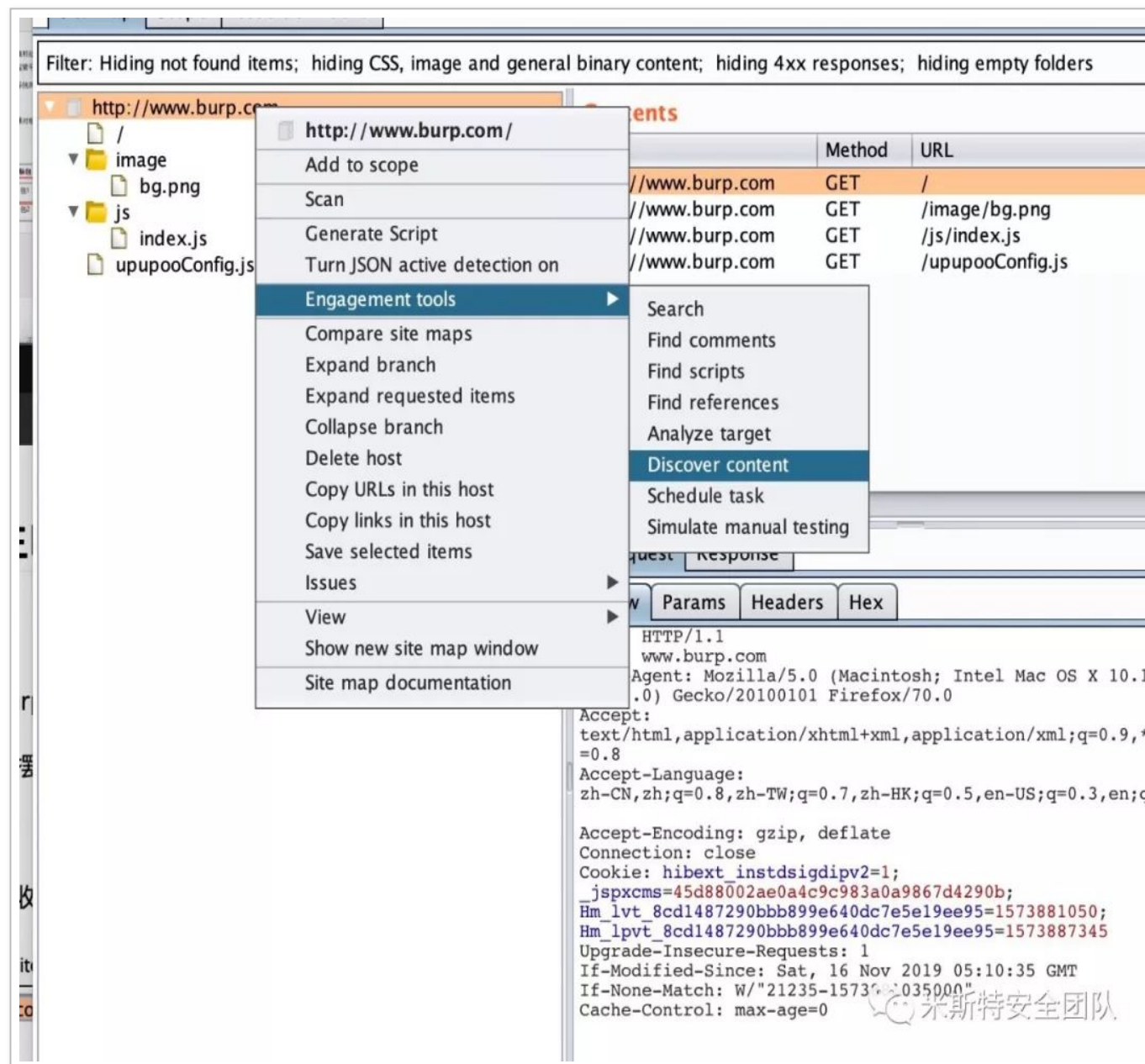
此前在米斯特 `BurpSuite` 插件生态开发组 里有提到想开发一款主动的备份扫描插件，寄生于 `BurpSuite` 强大的爬虫功能对每个目录 / 文件进行备份扫描。写一半发现 `BurpSuite` 本来就有这样的功能，只是很少有人用到罢了，本文仅对相关功能进行简要的分析以及优化。

0x06_1 简介

正常访问时爬虫被动收集到的目录：

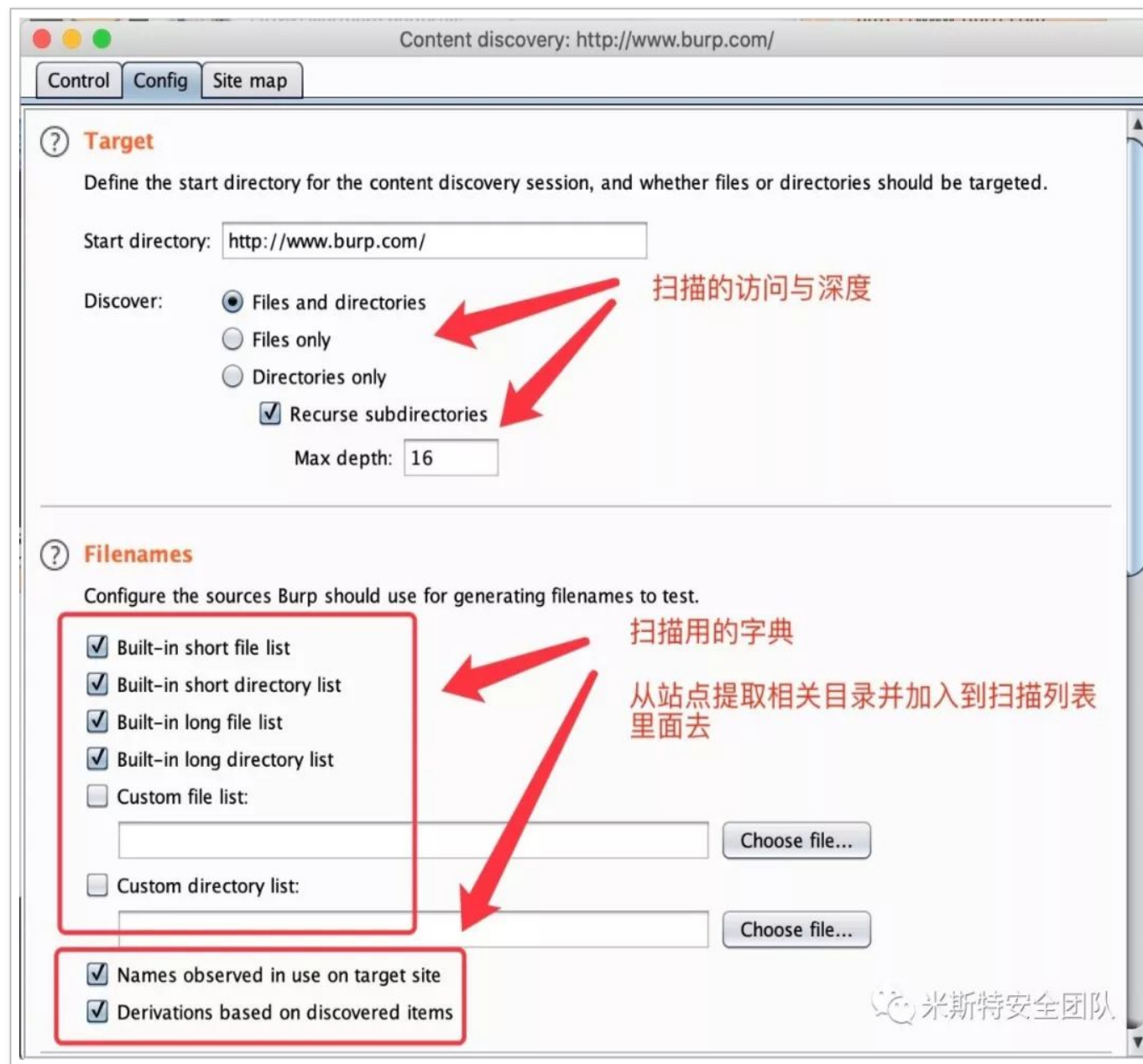


下面是本文的主角：

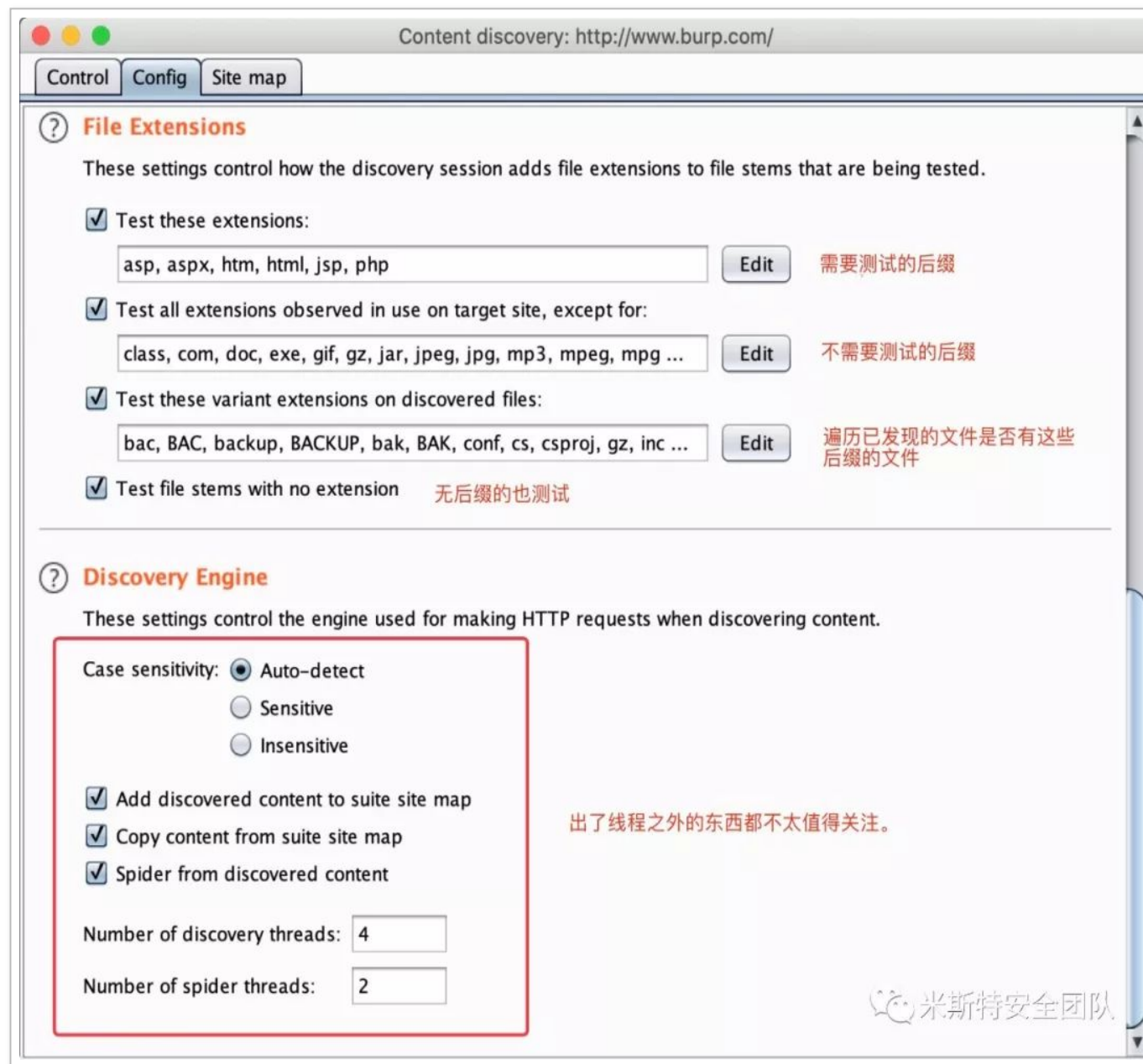


位于 Engagement tools ==> Discover content

看看默认配置：



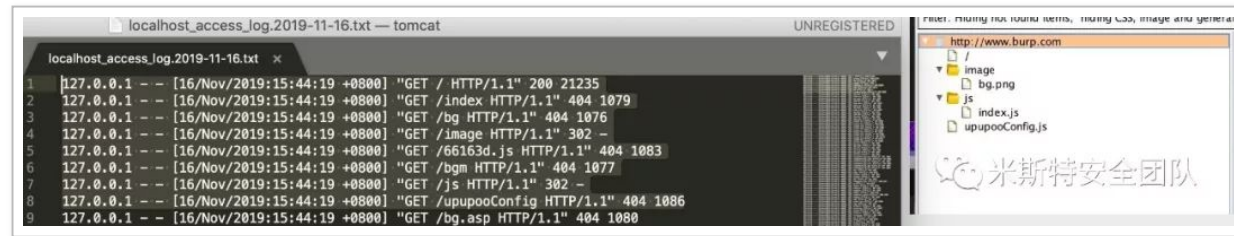
图中圈起来的第一部分默认勾选了 burp 内置的四个字典, 第二个圈也是默认配置, 其作用是从站点中提取各种目录并递归扫描, 前面的配置可根据情况修改, 后面不建议动它。



默认配置就是这四项了，下面具体分析一下他扫描的流量，取其精华。

0x06_2 扫描分析

流程 1:



先去除所有文件的后缀以及提取目录名当作文件扫一遍。

流程 2:

```
8 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /upupooConfig HTTP/1.1" 404 1086
9 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bg.asp HTTP/1.1" 404 1080
10 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bgm.asp HTTP/1.1" 404 1081
11 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bg.aspx HTTP/1.1" 404 1081
12 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bg.htm HTTP/1.1" 404 1080
13 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bgm.aspx HTTP/1.1" 404 1082
14 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bgm.htm HTTP/1.1" 404 1081
15 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bg.html HTTP/1.1" 404 1081
16 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bgm.html HTTP/1.1" 404 1082
17 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bg.jsp HTTP/1.1" 404 1080
18 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bgm.jsp HTTP/1.1" 404 1081
19 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bg.php HTTP/1.1" 404 1080
20 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bgm.php HTTP/1.1" 404 1081
21 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /image.asp HTTP/1.1" 404 1083
22 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /index.asp HTTP/1.1" 404 1083
23 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /image.aspx HTTP/1.1" 404 1084
24 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /index.aspx HTTP/1.1" 404 1084
25 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /image.htm HTTP/1.1" 404 1083
26 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /index.htm HTTP/1.1" 404 1083
27 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /image.html HTTP/1.1" 404 1084
28 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /index.html HTTP/1.1" 200 21235
29 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /image.jsp HTTP/1.1" 404 1083
30 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /image.php HTTP/1.1" 404 1083
31 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /js.asp HTTP/1.1" 404 1080
32 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /js.aspx HTTP/1.1" 404 1081
33 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /js.htm HTTP/1.1" 404 1080
34 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /js.html HTTP/1.1" 404 1081
35 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /js.jsp HTTP/1.1" 404 1080
36 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /js.php HTTP/1.1" 404 1080
37 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /upupooConfig.asp HTTP/1.1" 404 1090
38 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /upupooConfig.aspx HTTP/1.1" 404 1091
39 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /upupooConfig.htm HTTP/1.1" 404 1090
40 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /upupooConfig.html HTTP/1.1" 404 1091
41 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /upupooConfig.jsp HTTP/1.1" 404 1090
42 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /upupooConfig.php HTTP/1.1" 404 1090
43 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bg/ HTTP/1.1" 404 1081
```

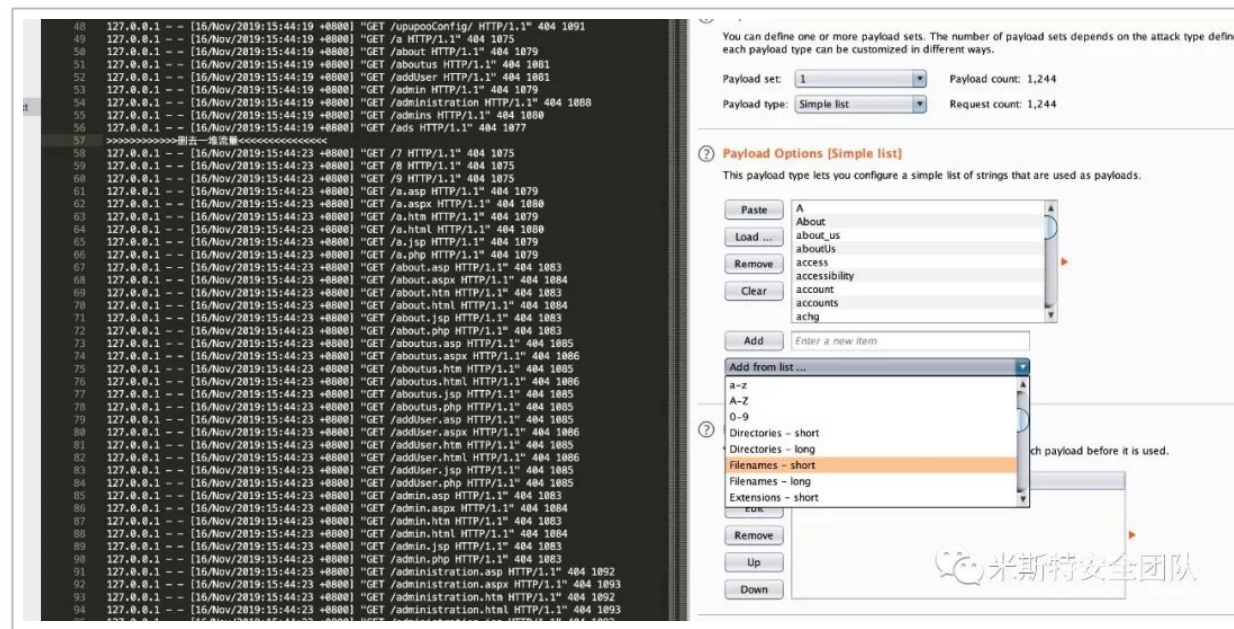
在所有去了后缀的文件名后面加上 Test these extentsions 配置中的所有后缀并进行扫描，在已知站点语言的情况下此处明显多了很多无效流量。

流程 3:

```
43 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bg/ HTTP/1.1" 404 1081
44 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /bgm/ HTTP/1.1" 404 1082
45 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /image/ HTTP/1.1" 404 1084
46 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /index/ HTTP/1.1" 404 1084
47 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /js/ HTTP/1.1" 404 1080
48 127.0.0.1 -- [16/Nov/2019:15:44:19 +0800] "GET /upupooConfig/ HTTP/1.1" 404 1091
```

将流程 1 中的所有文件名当作目录扫一遍。

流程 4:



按照配置中 Filenames 中的字典顺序依次遍历并结合流程 2 的方式遍历文件。

流程 5:

依次在每层目录执行一遍流程 1-4，如果新目录或文件发现就添加到列队里面。

0x06_3 优化扫描

? Discovery Session Status

Use these settings to monitor and control the discovery session.

Session is running

Requests made:	14,048
Bytes transferred:	21,350,214
Errors:	0
Tasks queued:	0
Spider requests queued:	0
Responses queued for analysis:	0

米斯特安全团队

缺陷很明显，仅仅三个目录三个文件总共请求了 14048 次，流量超过 20M。

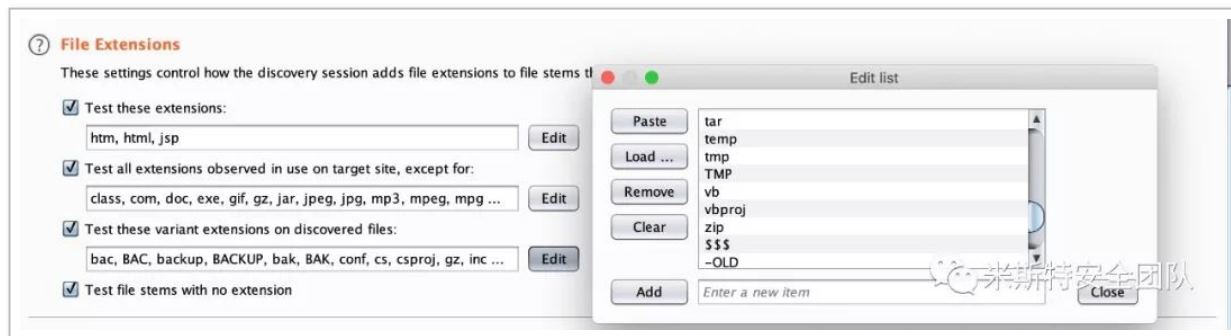
递归扫描产生的无效流量太多了。

首要原因出现在：



大多数时候，已知语言情况下完全没必要吧所有后缀都跑一遍，留特定语言以及相关后缀。如 st2 时加上. action/.do, 流量去掉大半。

其次出现在:



此处主要针对扫备份，原生字典长这样:


bac
BAC
backup
BACKUP
bak
BAK
conf
cs
csproj
gz
inc
INC

ini
java
log
lst
old
OLD
orig
ORIG
sav
save
tar
temp
tmp
TMP
vb
vbproj
zip
\$\$\$
-OLD
-old
0
1
~1
~bk

主要针对国外的站吧。这里提供一份。

txt
md
xml
db
7z
rar
zip
gz
tar
tar.gz
sql
bak
swp
old
properties
inc
ini
mdb
mdf
conf
config
log
rar

最后是字典的选择：

 **Filenames**

Configure the sources Burp should use for generating filenames to test.


- ☒ Built-in short file list
- ☒ Built-in short directory list
- ☒ Built-in long file list
- ☒ Built-in long directory list

☐ Custom file list:

☐ Custom directory list:

☒ Names observed in use on target site

☒ Derivations based on discovered items

 米斯特安全团队

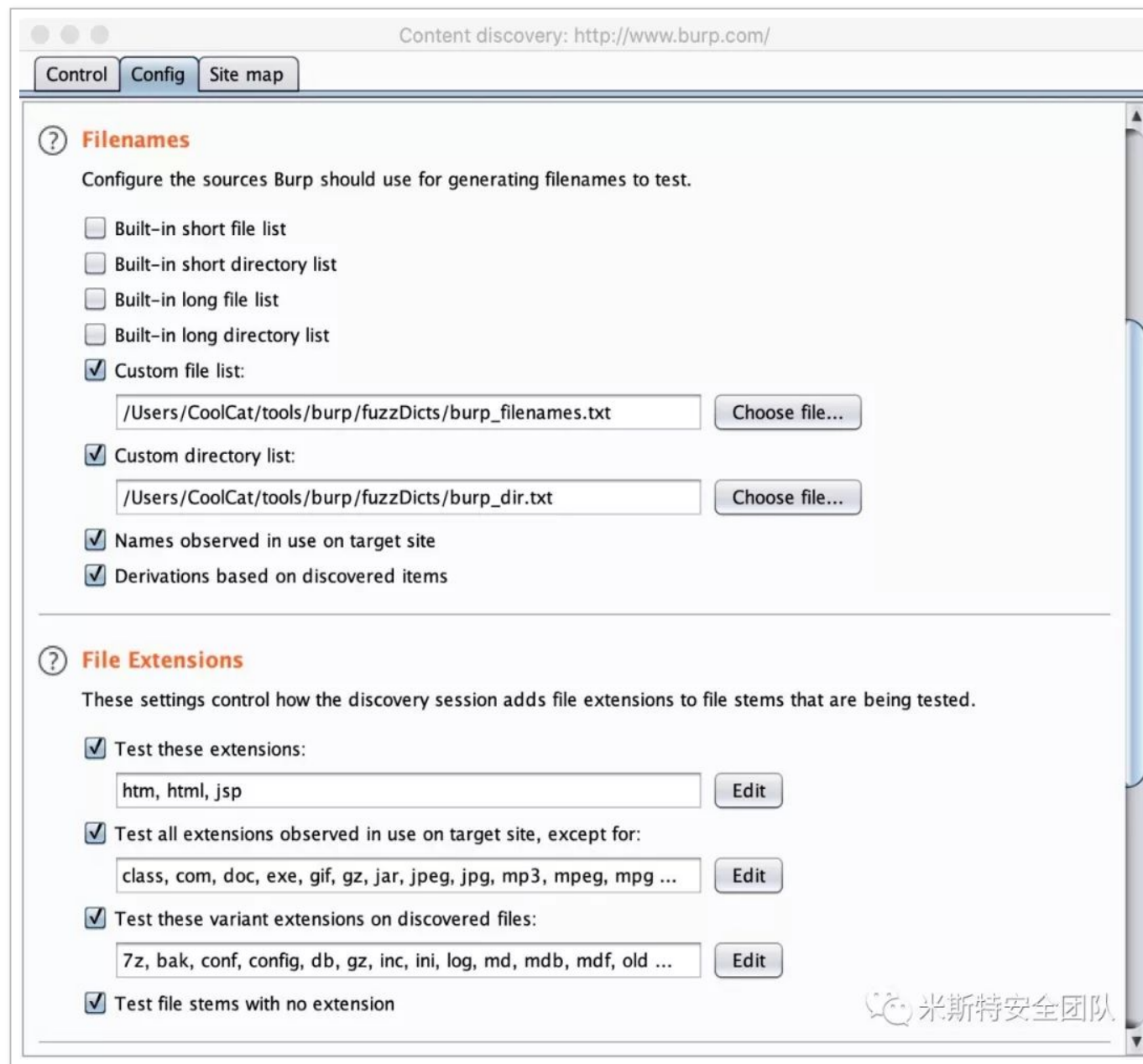
burp 内置的 4 个字典内容重复率极高，比如 about 这个词在四个字典里都分别出现了... 四个字典共 3266 个，去重后 2265 个。

结果保存在：

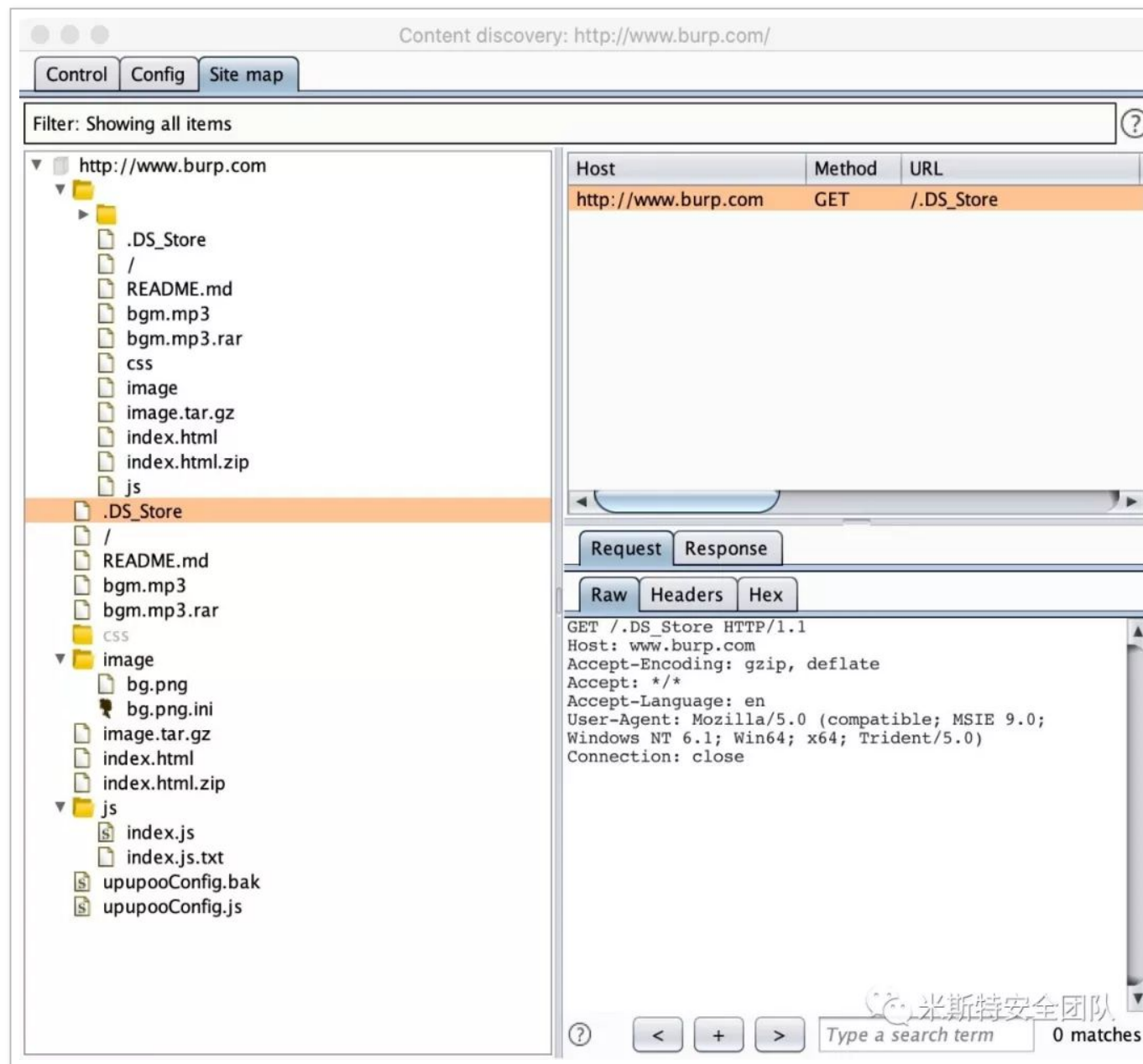
https://raw.githubusercontent.com/TheKingOfDuck/myScripts/master/burp_dir.txt

文件名方面推荐：

https://raw.githubusercontent.com/TheKingOfDuck/myScripts/master/burp_filenames.txt

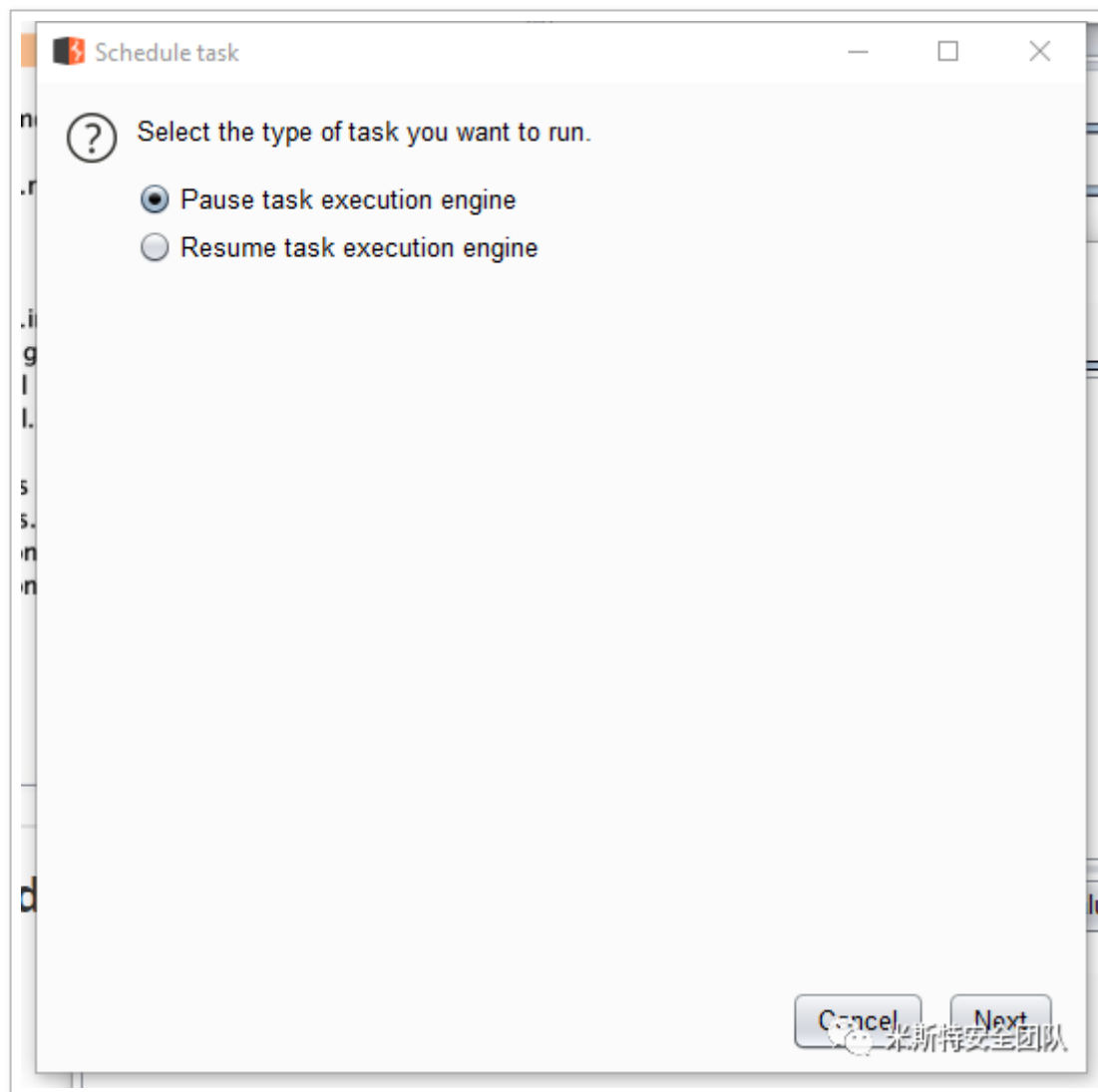


如此一来在扫描目录的过程也可把备份一并扫了。



0x07 Schedule task 定时任务

选择目标右键打开后如下图：



Enter the details of when the task should run. The task can be scheduled to run once at a specific time, or to repeat at a defined interval.

Start at: 18 26

On: 11 February

Repeat every: (Leave blank to run once only)

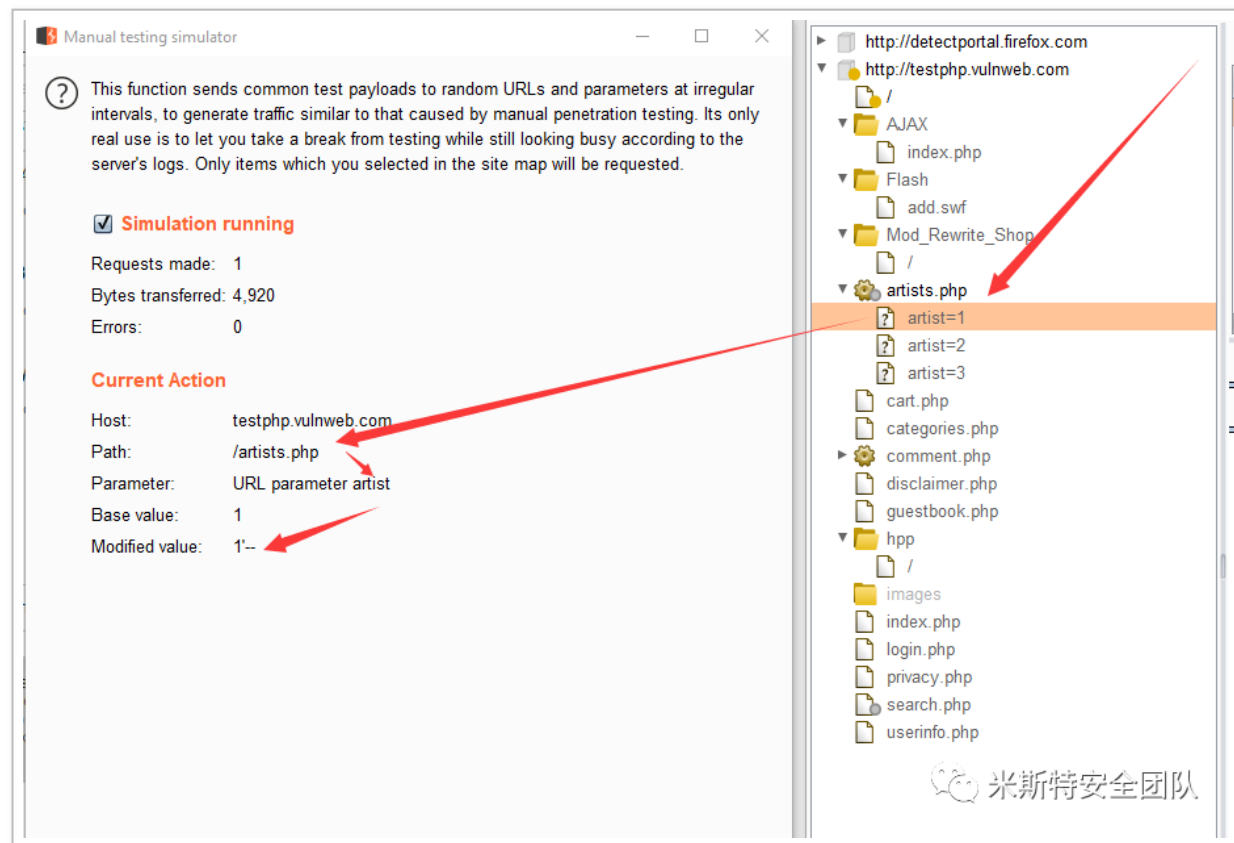
Back Finish

定时任务，没啥特别的值得说。

0x08 Simulate manual testing 人工模拟

“This function sends common test payloads to random URL s and parameters at irregular intervals, to generate traffic similar to that caused by manual penetration testing. Its only real use is to let you take a break from testing while still looking busy according to the server's logs. Only items which you selected in the site map will be requested.”

正如功能描述中所说的，这是一个懒人挖洞选项。



缺陷很明显，不会自动寻找并提交表单，只能依托于已有的表单 / 参数的基础上进行探测，相当于一款被动扫描器。结合 Discover content 使用，法力无边。

0x09 总结

熟练使用这些作战功能，扫描器从此是路人。

二、配置 Intruder 预定义 Payload 列表

预定义 Payload 列表在这里可以看见：

1 x ...

TargetPositionsPayloadsOptions

?

Payload Sets

You can define one or more payload sets. The number of payload sets c
available for each payload set, and each payload type can be customized

Payload set: 1Payload count: 0

Payload type: Simple listRequest count: 0

?

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are use

Paste

Load ...

Remove

Clear

Add

Enter a new item

?

Add from list ...

Add from list ...

Fuzzing - quick

Fuzzing - full

Usernames

Passwords

Short words

a-z

A-Z

ach paylo

米斯特安全团队

在 Tab 栏中有选项可以配置 - Configure predefined payload lists:

Intruder Repeater Window Help

Start attack

Open saved attack

Scan defined insertion points

Send to Repeater

Save attack config ▶

Load attack config ▶

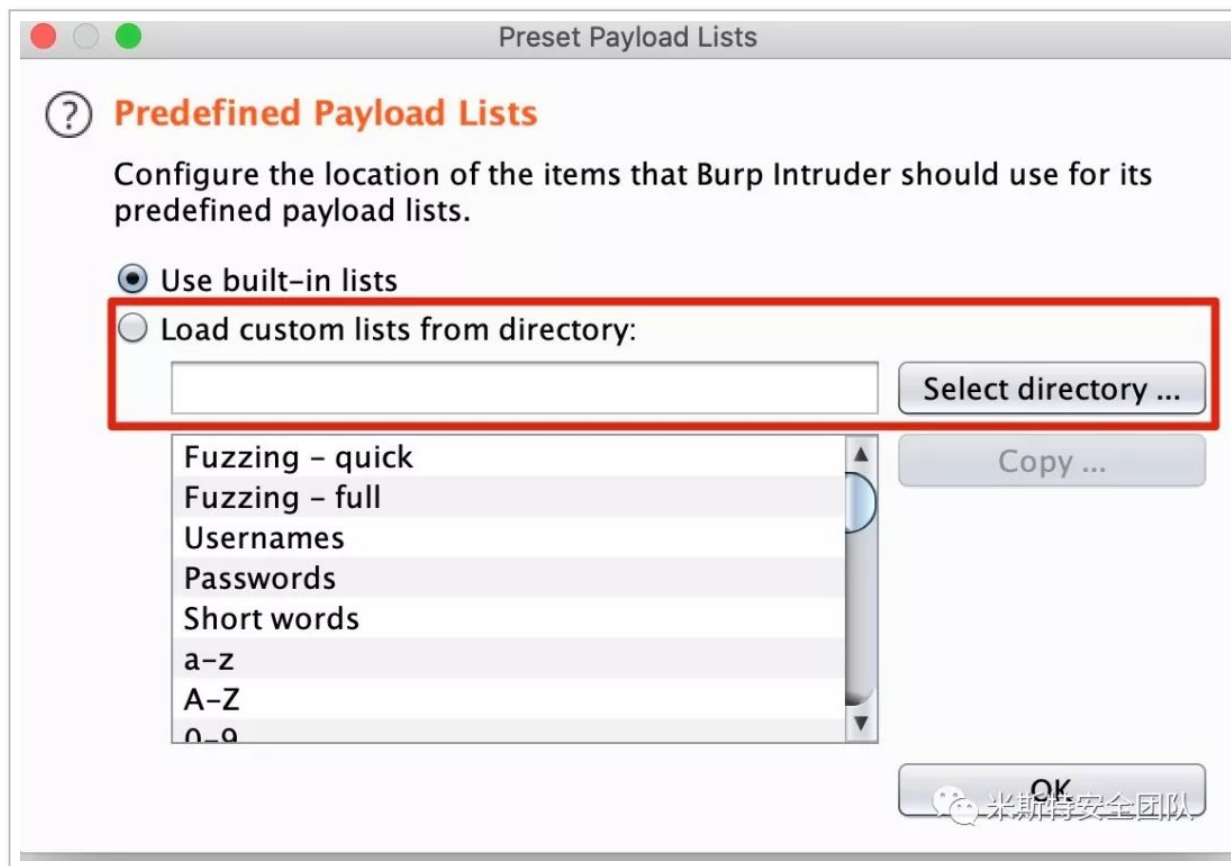
Copy attack config ▶

New tab behavior ▶

Automatic payload positions ▶

Configure predefined payload lists 聚斯特安全团队

选择 Payload 列表目录（注：需将常用的字典都放在一个目录里面）



三、自定义插入点进行扫描

将请求报文转发至 Intruder，选择扫描的插入点： Add\$

Attack type: **Sniper**

```
POST /example?p1=$p1val$&p2=p2val HTTP/1.0
```

```
Cookie: c=cval
```

```
Content-Length: 17
```

```
p3=p3val&p4=p4val
```

米斯特安全团队

右键选择 Scan defined insertion points

Attack type: **Sniper**

```
POST /example?p1=$p1val$&p2=p2val HTTP/1.0
```

```
Cookie: c=cval
```

```
Content-Length: 17
```

```
p3=p3val&p4=p4val
```

Send to Repeater

⌘+^+R

Scan defined insertion points

Send request(s) to Authz

Send to SQLMapper

Send to Laudanum

米斯特安全团队