burp 日志插件从原理到实践

> 本文作者：**鹅**（首次投稿）

Logger++ 是 nccgroup 开源的一个 burp 扩展，主要功能是记录经过 Burp Suite 的所有 HTTP 请求 和 HTTP 响应。

相较于 Burp 自带的 Proxy 组件中的 HTTP History， logger++ 的优势是记录了更完整的流量，并且支持对这些流量进行基于正则表达式的简易分析，对相关流量记录进行着色展示，将流量导入到 elasticsearch 平台等。Burp 基本组件 Proxy 中的 HTTP History 则只记录经过代理的 HTTP 流量，对于 Repeater, Scanner, Intruder 等组件的流量，并不会在它的标签中展示。

对于笔者而言，常用的功能主要有两个：

1、基于正则表达式的简易 HTTP 流量分析

2、记录流量日志 (导出 csv 便于后续代码分析)

代码简要分析

> https://github.com/nccgroup/BurpSuiteLoggerPlusPlus/blob/master/src/main/java/burp/BurpExtender.java

从入口 `src/main/java/burp/BurpExtender.java` 开始看，一开始只是继承 loggerplusplus.LoggerPlusPlus 类。

```
package burp;

import loggerplusplus.LoggerPlusPlus;


public class BurpExtender extends LoggerPlusPlus
{
  public static void main(String [] args){
    System.out.println("You have built the Logger++. You shall play with the jar file now!");
  }
}
```

切换到 `src/main/java/loggerplusplus/LoggerPlusPlus.java` 与记录 HTTP 日志有关的语句是 `logManager = new LogManager(loggerPreferences);` 。

```
    @Override
    public void registerExtenderCallbacks(final IBurpExtenderCallbacks callbacks)
{
        //Burp Specific
        LoggerPlusPlus.callbacks = callbacks;
        LoggerPlusPlus.instance = this;
        LoggerPlusPlus.contextMenuFactory = new LoggerContextMenuFactory();

        callbacks.setExtensionName("Logger++");

        filterListeners = new ArrayList<>();
        loggerPreferences = new LoggerPreferences(LoggerPlusPlus.this);
        logManager = new LogManager(loggerPreferences);
        elasticSearchLogger = new ElasticSearchLogger(logManager, loggerPreferences);

        if(!callbacks.isExtensionBapp() && loggerPreferences.checkUpdatesOnStartup()){
            MoreHelp.checkForUpdate(false);
        }

        buildUI();
    }
```

```
17
18
19
20
21
```

继续跟踪到 `src/main/java/loggerplusplus/LogManager.java` 。

基本思路就是继承和实现 Burp 提供的 IHttpListener 接口 和 IProxyListener 接口，重写 `processHttpMessage` 和 `processProxyMessage` 方法，存储流经的 HTTP 流量。还有一些对 HTTP 请求 / 响应 的处理细节可以在 `src/main/java/loggerplusplus/LogEntry.java` 中的 `processRequest` 和 `processResponse` 中找到。（其实大部分情况下 Burp 自带的 `LoggerPlusPlus.getCallbacks().getHelpers().analyzeRequest(requestResponse)` 已经帮我们把需要的字段解析完成了。）

关键代码如下：

```
1       @Override
        public void processHttpMessage(final int toolFlag, final boolean messageIsRequest, final
2           // Only process scanner messages which contain the request and response.
            if(!messageIsRequest) {
3               final LogEntry logEntry = new LogEntry();
```

```
            processHttpMessage(logEntry, toolFlag, requestResponse);
        }
    }

    // Wrapper to allow a custom LogEntry to be passed as a parameter
    // Custom LogEntry used when importing proxy history.
    // messageIsRequest is removed as not needed.
    public void processHttpMessage(final LogEntry logEntry, final int toolFlag, final IHttpRe
        executorService.submit(new Runnable() {
            @Override
            public void run() {
                if(toolFlag != IBurpExtenderCallbacks.TOOL_PROXY || logEntry.isImported){
                    if(requestResponse == null || !prefs.isEnabled()) return;

                    IRequestInfo analyzedReq = LoggerPlusPlus.getCallbacks().getHelpers().ana
                    URL uUrl = analyzedReq.getUrl();

                    if (isValidTool(toolFlag) && (!prefs.isRestrictedToScope() || LoggerPlusP
                        // We will not need to change messageInfo so save to temp file
                        IHttpRequestResponse savedReqResp = LoggerPlusPlus.getCallbacks().sav
                        logEntry.processRequest(toolFlag, savedReqResp, uUrl, analyzedReq, nu
                        if(requestResponse.getResponse() != null) logEntry.processResponse(sa
                        // Check entry against colorfilters.
                        for (ColorFilter colorFilter : prefs.getColorFilters().values()) {
                            logEntry.testColorFilter(colorFilter, false);
                        }

                        addNewRequest(logEntry, true); // Complete Request and Response Added
                        for (LogEntryListener logEntryListener : logEntryListeners) {
```

```
                        logEntryListener.onResponseUpdated(logEntry);
                    }
                }
            }
        }
    });
}

@Override
public void processProxyMessage(final boolean messageIsRequest, final IInterceptedProxyMe
    //REQUEST AND RESPONSE SEPARATE
    final LogEntry.PendingRequestEntry logEntry;
    if(messageIsRequest){
        logEntry = new LogEntry.PendingRequestEntry();
    }else{
        synchronized (pendingRequests) {
            logEntry = pendingRequests.remove(proxyMessage.getMessageReference());
        }
    }
    executorService.submit(new Runnable() {
        @Override
        public void run() {
            if(proxyMessage == null || !prefs.isEnabled()) return;
            IHttpRequestResponse requestResponse = proxyMessage.getMessageInfo();
            IRequestInfo analyzedReq = LoggerPlusPlus.getCallbacks().getHelpers().analyze
            URL uUrl = analyzedReq.getUrl();
            int toolFlag = LoggerPlusPlus.getCallbacks().TOOL_PROXY;
            if (isValidTool(toolFlag) && (!prefs.isRestrictedToScope() || LoggerPlusPlus.
                if(messageIsRequest){
```

```
                    //New Proxy Request
                    //We need to change messageInfo when we get a response so do not save
                    logEntry.processRequest(toolFlag, requestResponse, uUrl, analyzedReq,
                    for (ColorFilter colorFilter : prefs.getColorFilters().values()) {
                        logEntry.testColorFilter(colorFilter, false);
                    }
                    synchronized (pendingRequests) {
                        pendingRequests.put(proxyMessage.getMessageReference(), logEntry)
                    }
                    addNewRequest(logEntry, false); // Request added without response
                }else{
                    // Existing Proxy Request, update existing
                    if (logEntry != null) {
                        updatePendingRequest(logEntry, requestResponse);
                    } else {
                        lateResponses++;
                        if(totalRequests > 100 && ((float)lateResponses)/totalRequests >
                            MoreHelp.showWarningMessage(lateResponses +
                              " responses have been delivered after the Logger++ timeout.
                            //Reset late responses to prevent message being displayed aga
                            lateResponses = 0;
                        }
                    }
                }
            }
        }
    });
}

```

```
84
85
86
87
88
89
90
```

功能说明

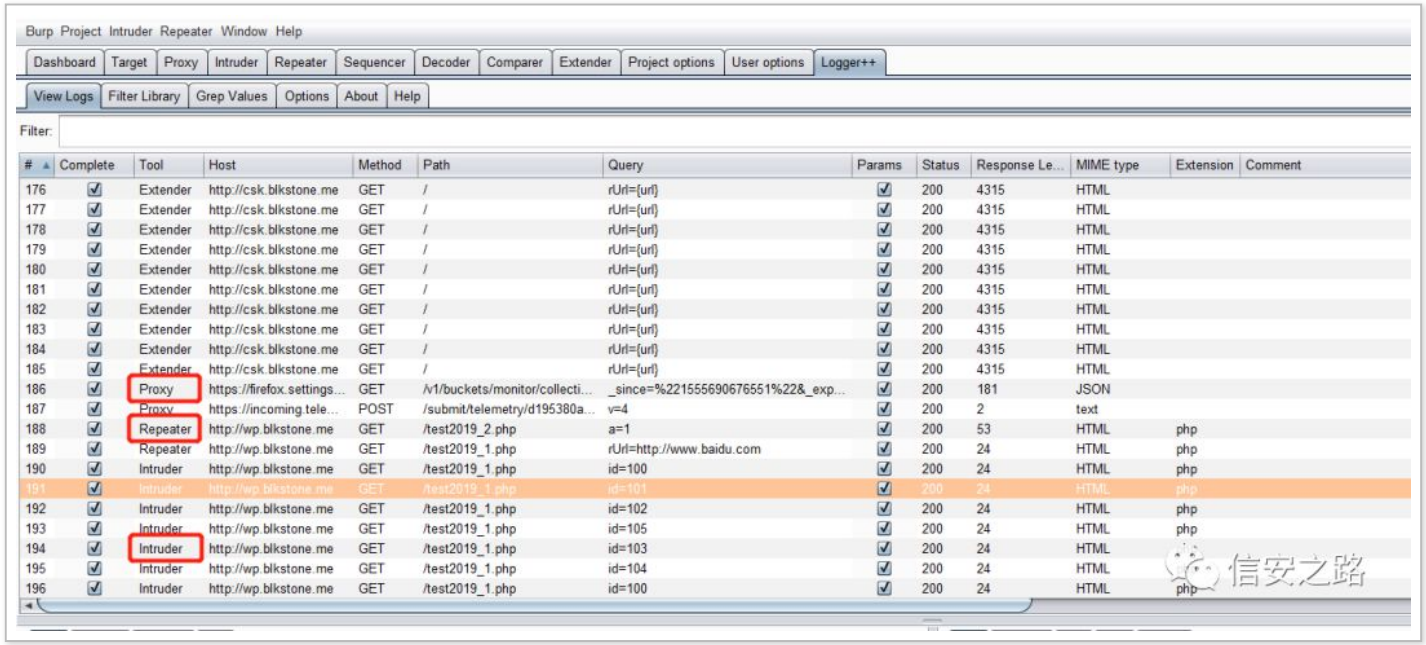
主界面

先看下 Logger++ 的基本界面，其实和 Proxy 中的 HTTP History 基本一致，稍微新增了一些字段。

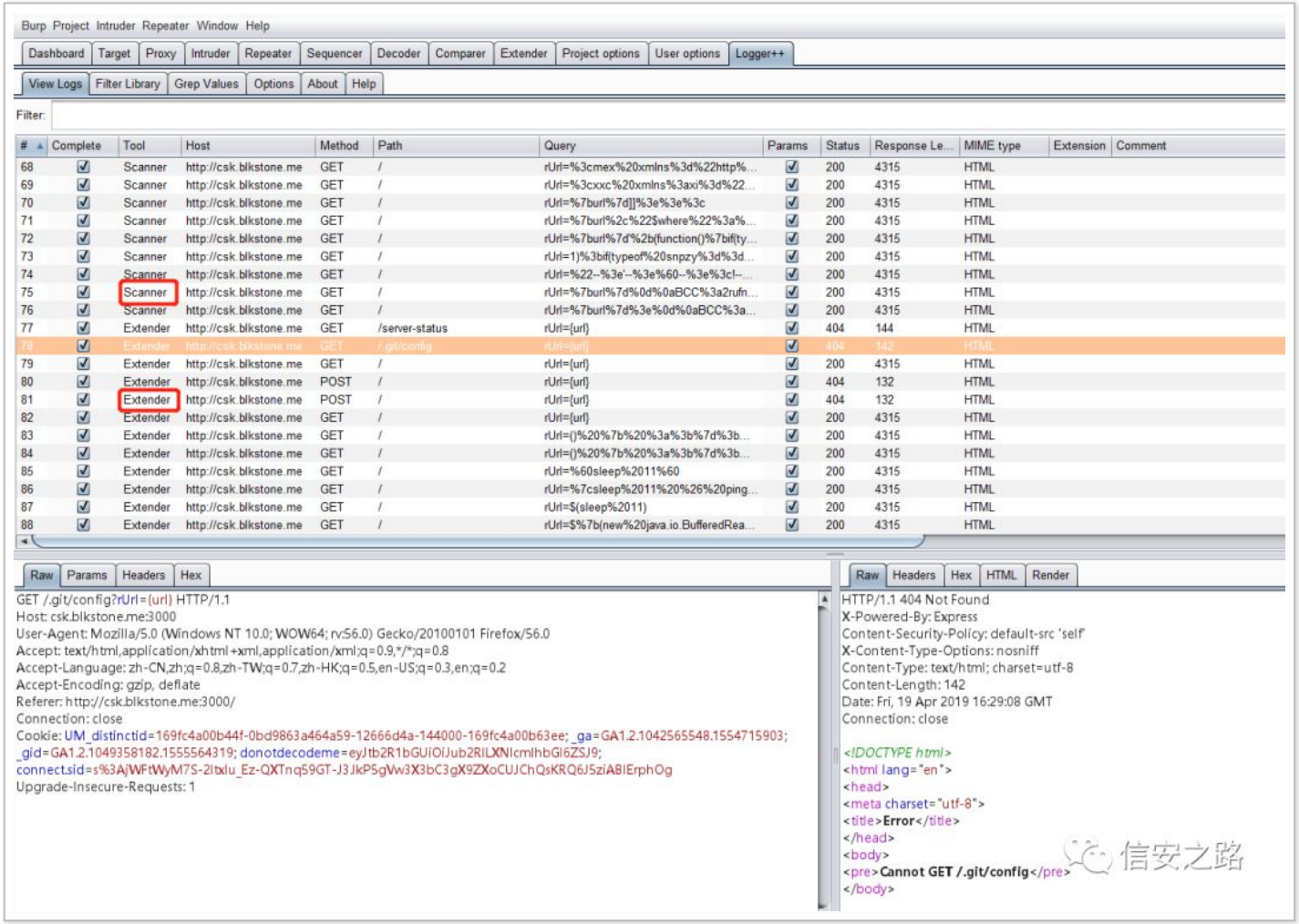
正常经过代理的流量 Tool 字段都是 Proxy；Repeater, Intruder 等工具发出的请求也会进行记录。

如果 Tool 为 Scanner 是 Burp 自带的扫描器发送的请求；Tool 为 Extender 是其他 Burp 插件发送的请求。在这个场景插件发出的请求都是来自 Active Scan++，**利用这个特性你可以使用 Logger++ 以 "黑盒" 的方式分析 Burp 一些的扫描能力增强插件的原理 / payload** 。

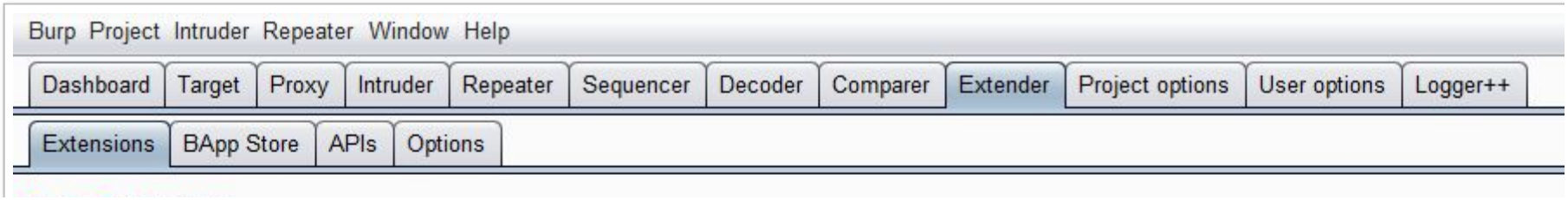
Proxy / Repeater / Intruder 案例:



Scanner / Extender 案例:


Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Logger++
View Logs Filter Library Grep Values Options About Help
Filter:
Complete Tool Host Method Path Query Params Status Response Le... MIME type Extension Comment
68 Scanner http://csk.blkstone.me GET / rUrl=%3cmex%20xmlns%3d%22http%... 200 4315 HTML
69 Scanner http://csk.blkstone.me GET / rUrl=%3cxxc%20xmlns%3axi%3d%22... 200 4315 HTML
70 Scanner http://csk.blkstone.me GET / rUrl=%7burl%7d]]%3e%3e%3c 200 4315 HTML
71 Scanner http://csk.blkstone.me GET / rUrl=%7burl%2c%22$where%22%3a%... 200 4315 HTML
72 Scanner http://csk.blkstone.me GET / rUrl=%7burl%7d%2b(function()%7bif(ty... 200 4315 HTML
73 Scanner http://csk.blkstone.me GET / rUrl=1)%3bif(typeof%20snpzy%3d%3d... 200 4315 HTML
74 Scanner http://csk.blkstone.me GET / rUrl=%22--%3e'--%3e%60--%3e%3c!--... 200 4315 HTML
75 Scanner http://csk.blkstone.me GET / rUrl=%7burl%7d%0d%0aBCC%3a2rufn... 200 4315 HTML
76 Scanner http://csk.blkstone.me GET / rUrl=%7burl%7d%3e%0d%0aBCC%3a... 200 4315 HTML
77 Extender http://csk.blkstone.me GET /server-status rUrl={url} 404 144 HTML
78 Extender http://csk.blkstone.me GET /.git/config rUrl={url} 404 142 HTML
79 Extender http://csk.blkstone.me GET / rUrl={url} 200 4315 HTML
80 Extender http://csk.blkstone.me POST / rUrl={url} 404 132 HTML
81 Extender http://csk.blkstone.me POST / rUrl={url} 404 132 HTML
82 Extender http://csk.blkstone.me GET / rUrl={url} 200 4315 HTML
83 Extender http://csk.blkstone.me GET / rUrl=()%20%7b%20%3a%3b%7d%3b... 200 4315 HTML
84 Extender http://csk.blkstone.me GET / rUrl=()%20%7b%20%3a%3b%7d%3b... 200 4315 HTML
85 Extender http://csk.blkstone.me GET / rUrl=%60sleep%2011%60 200 4315 HTML
86 Extender http://csk.blkstone.me GET / rUrl=%7csleep%2011%20%26%20ping... 200 4315 HTML
87 Extender http://csk.blkstone.me GET / rUrl=$(sleep%2011) 200 4315 HTML
88 Extender http://csk.blkstone.me GET / rUrl=$%7b(new%20java.io.BufferedRea... 200 4315 HTML
Raw Params Headers Hex
GET /.git/config?rUrl={url} HTTP/1.1
Host: csk.blkstone.me:3000
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://csk.blkstone.me:3000/
Connection: close
Cookie: UM_distinctid=169fc4a00b44f-0bd9863a464a59-12666d4a-144000-169fc4a00b63ee; _ga=GA1.2.1042565548.1554715903; _gid=GA1.2.1049358182.1555564319; donotdecodeme=eyJtb2R1bGUiOiJub2RlLXNlcmlhbGl6ZSJ9; connect.sid=s%3AjWFtWyM7S-2IbxIu_Ez-QXTnqS9GT-J3.JkP5gVw3X3bC3gX9ZXoCUJChQsKRQ6J5ziABIErphOg
Upgrade-Insecure-Requests: 1
Raw Headers Hex HTML Render
HTTP/1.1 404 Not Found
X-Powered-By: Express
Content-Security-Policy: default-src 'self'
X-Content-Type-Options: nosniff
Content-Type: text/html; charset=utf-8
Content-Length: 142
Date: Fri, 19 Apr 2019 16:29:08 GMT
Connection: close
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Error</title>
</head>
<body>
<pre>Cannot GET /.git/config</pre>
</body>
信安之路

记录上图时的插件情况


Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Logger++
Extensions BApp Store APIs Options

Burp Extensions

Extensions let you customize Burp's behavior using your own or third-party code.

Add | Remove | Up | Down

Loaded	Type	Name
☑	Python	Active Scan++
☐	Java	J2EEScan
☐	Java	Backslash Powered Scanner
☑	Java	Logger++
☐	Java	Copy As Python-Requests
☐	Java	LogRequestsToSQLite
☐	Java	JSON Beautifier
☐	Java	CO2
☐	Java	Turbo Intruder
☐	Python	SQLiPy Sqlmap Integration
☐	Python	Intruder Time Payloads
☐	Python	Random JSESSIONID Generator
☐	Python	Unix timestamp Intruder Generator
☐	Java	reCAPTCHA v0.8 by bit4
☐	Java	CVSS Calculator
☐	Java	Knife v0.8 by bit4woo
☐	Java	Chunked coding converter 0.1
☐	Java	Image Location & Privacy Scanner
☐	Java	Java Deserialization Scanner
☐	Python	Intelligent analysis
☐	Python	Burp Collector

Details | Output | Errors

☑ Extension loaded

Name: Active Scan++

Item	Detail
Extension type	Python
Filename	bapps\3123d5b5f25c4128894d97ea1acc4976\activeScan++.py

基于正则表达式的简易流量分析

过滤器也是 Logger++ 中一个很方便的功能。笔者通常会使用这个功能，来寻找一些敏感信息泄露和潜在的漏洞请求（这里提供的规则并不是 100% 能确定漏洞的，只是作为一个辅助手段，还需要进一步手工验证）。在该工具的 Filter Library 里也有几个不错的过滤器示例可以参考。

关于过滤器支持的其他字段可以在 Logger++ 的项目 wiki 里查到，链接如下

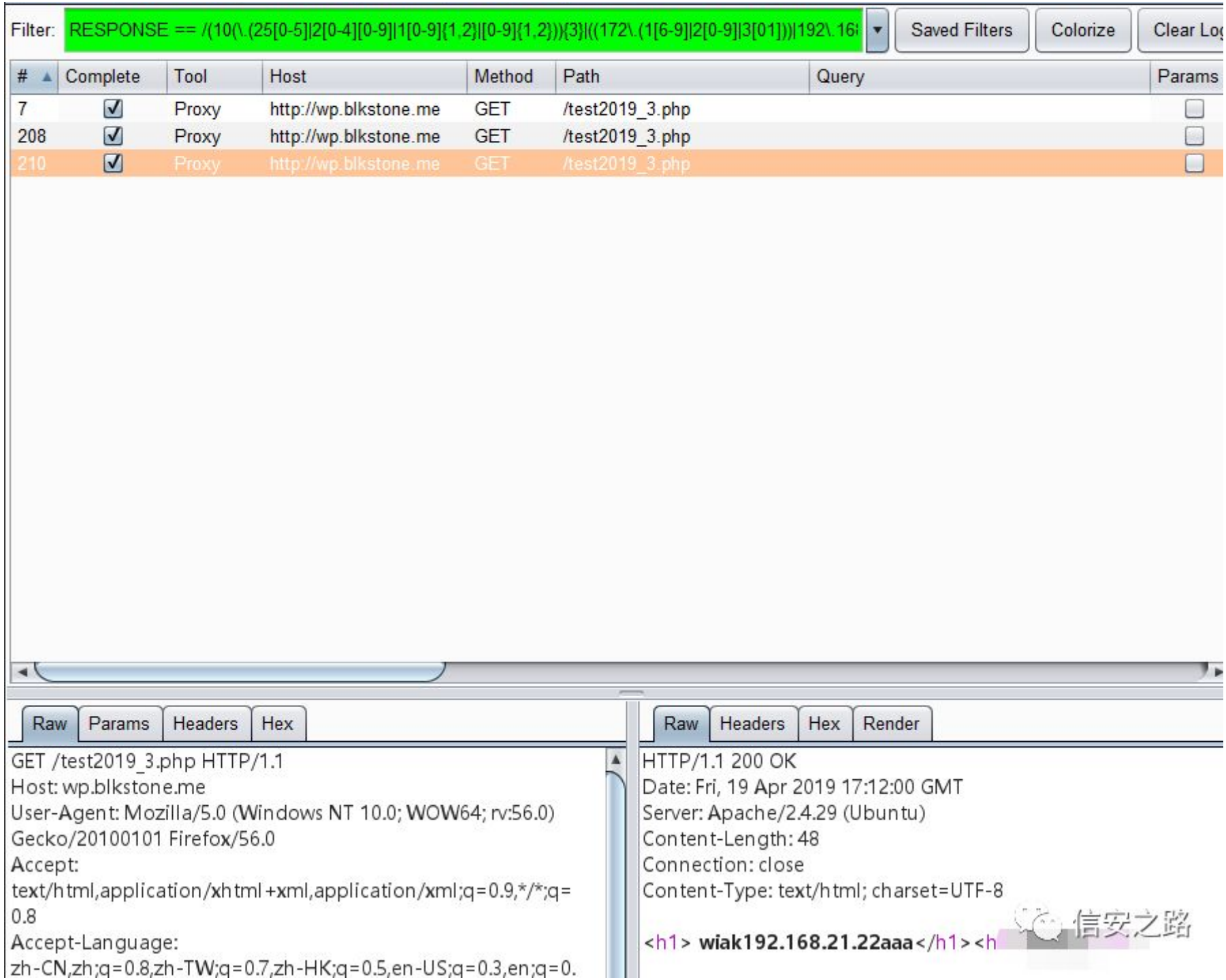
> https://github.com/nccgroup/BurpSuiteLoggerPlusPlus/wiki/Filter-Fields

这里提供一些过滤器的示例作为抛砖引玉

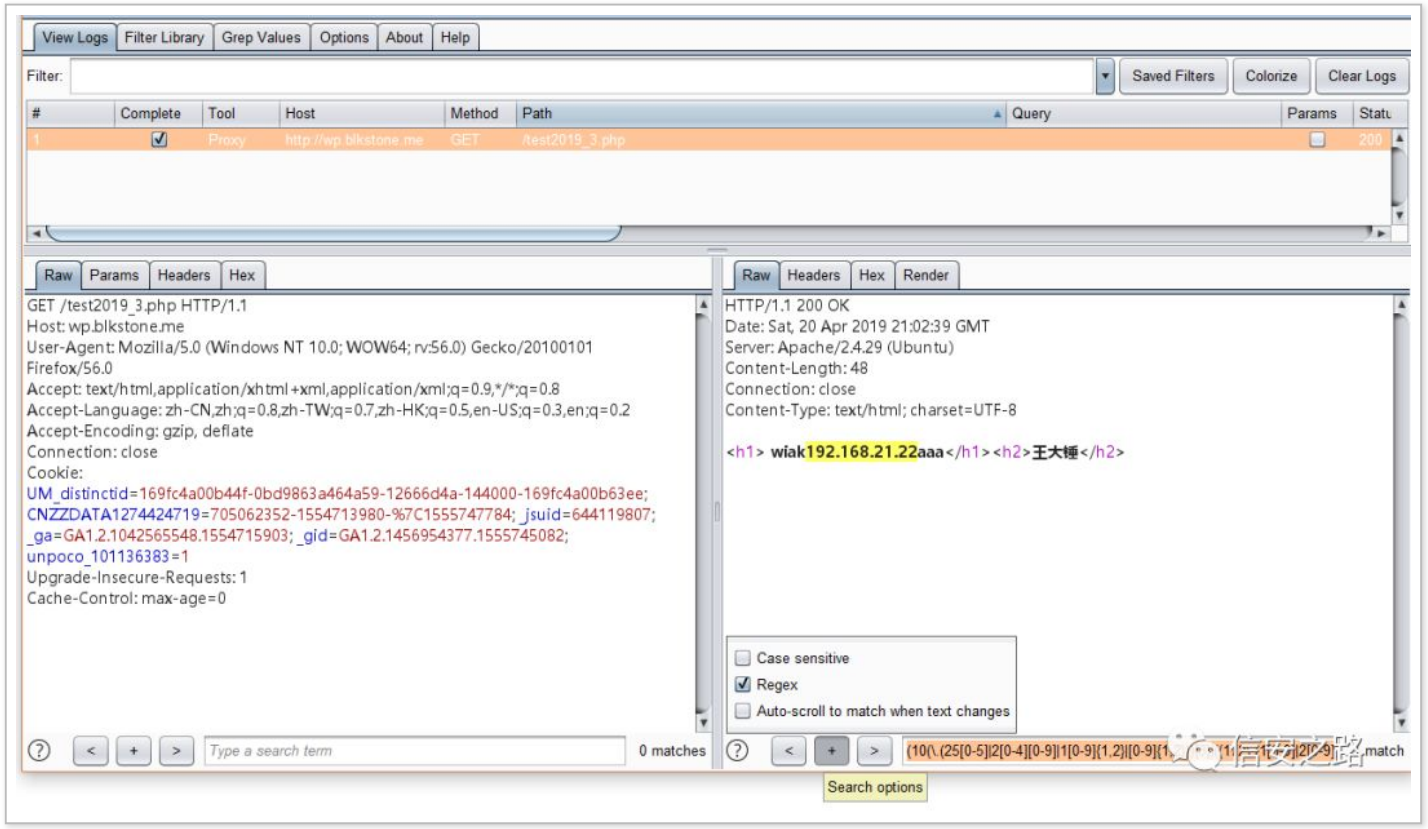
信息泄露（内网 IP）规则 #1

Internal IP Address

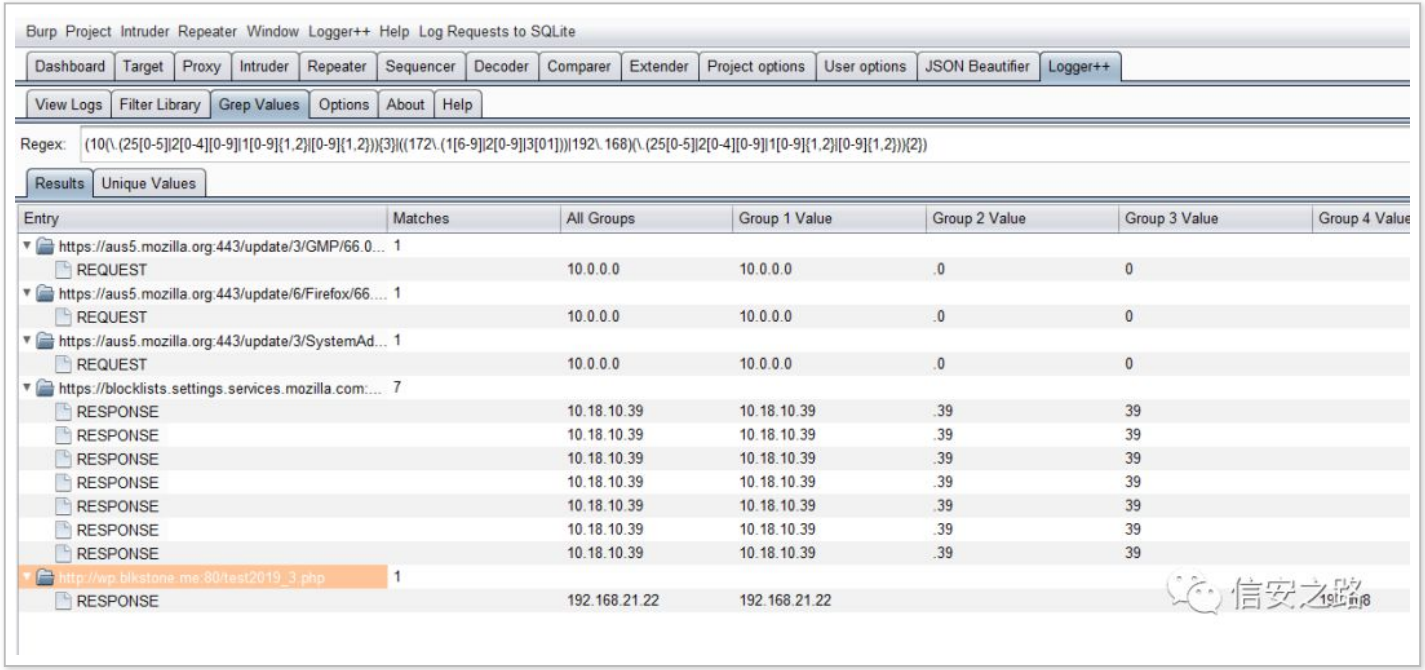
```
RESPONSE == /(10(\.(25[0-5]|2[0-4][0-9]|1[0-9]{1,2}|[0-9]{1,2})){3}|((172\.(1[6-9]|2[0-9]|3[01]))|192\.168)(\.(25[0-5]|2[0-4][0-9]|1[0-9]{1,2}|[0-9]{1,2})){2})/
```


Filter: RESPONSE == /(10(\.(25[0-5]|2[0-4][0-9]|1[0-9]{1,2}|[0-9]{1,2})){3}|((172\.(1[6-9]|2[0-9]|3[01]))|192\.16
Saved Filters
Colorize
Clear Log
Complete Tool Host Method Path Query Params
7 Proxy http://wp.blkstone.me GET /test2019_3.php
208 Proxy http://wp.blkstone.me GET /test2019_3.php
210 Proxy http://wp.blkstone.me GET /test2019_3.php
Raw Params Headers Hex
GET /test2019_3.php HTTP/1.1
Host: wp.blkstone.me
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=
0.8
Accept-Language:
zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.
Raw Headers Hex Render
HTTP/1.1 200 OK
Date: Fri, 19 Apr 2019 17:12:00 GMT
Server: Apache/2.4.29 (Ubuntu)
Content-Length: 48
Connection: close
Content-Type: text/html; charset=UTF-8
<h1> wiak192.168.21.22aaa</h1><h
信安之路

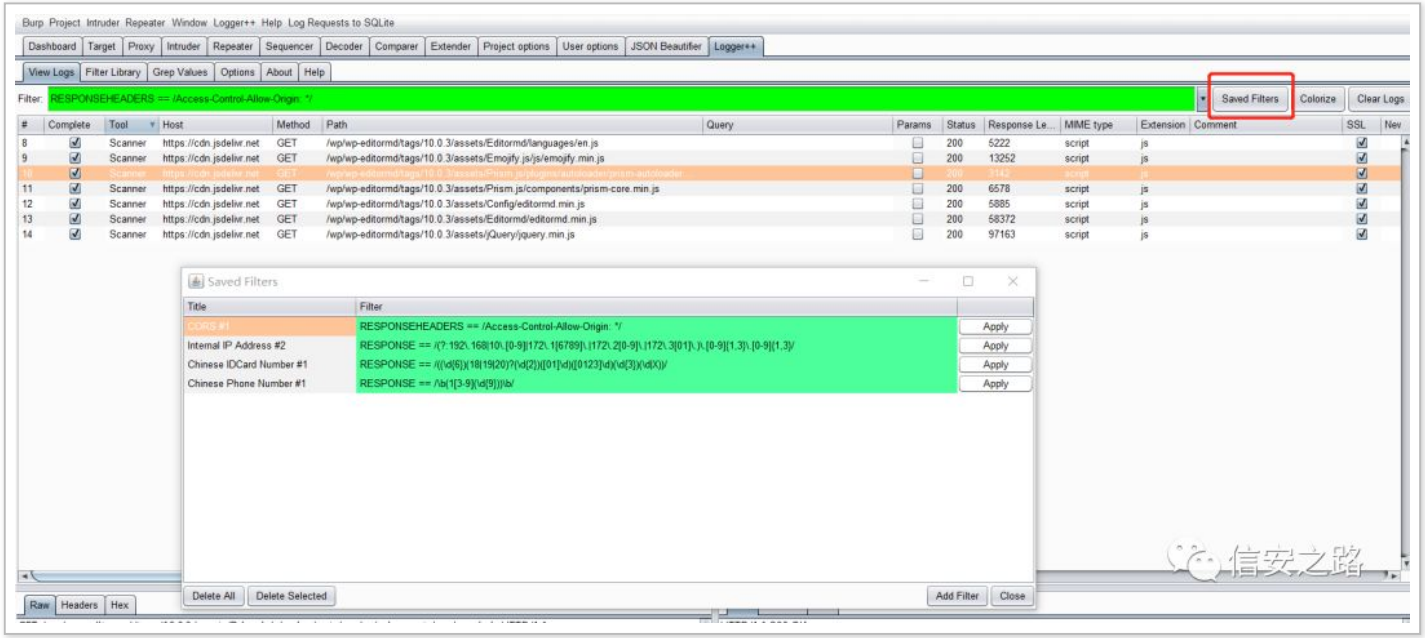
如果你需要确定具体匹配的值，可以在 Message Editor 里用正则表达式搜索。



也可以使用 Grep Values 标签来汇总所有请求中符合过滤器筛选内容的字符串值。



如果要新增自定义过滤器，只要点击 Saved Filters，然后点击 Add Filter 即可，另外在 Options 选项卡中可以批量导入 / 导出过滤器设置。



信息泄露（身份证号）#2

```
RESPONSE == /((\d{6})(18|19|20)?(\d{2})([01]\d)([0123]\d)(\d{3})(\d|X))/
```

信息泄露（电子邮件）#3

匹配所有邮箱

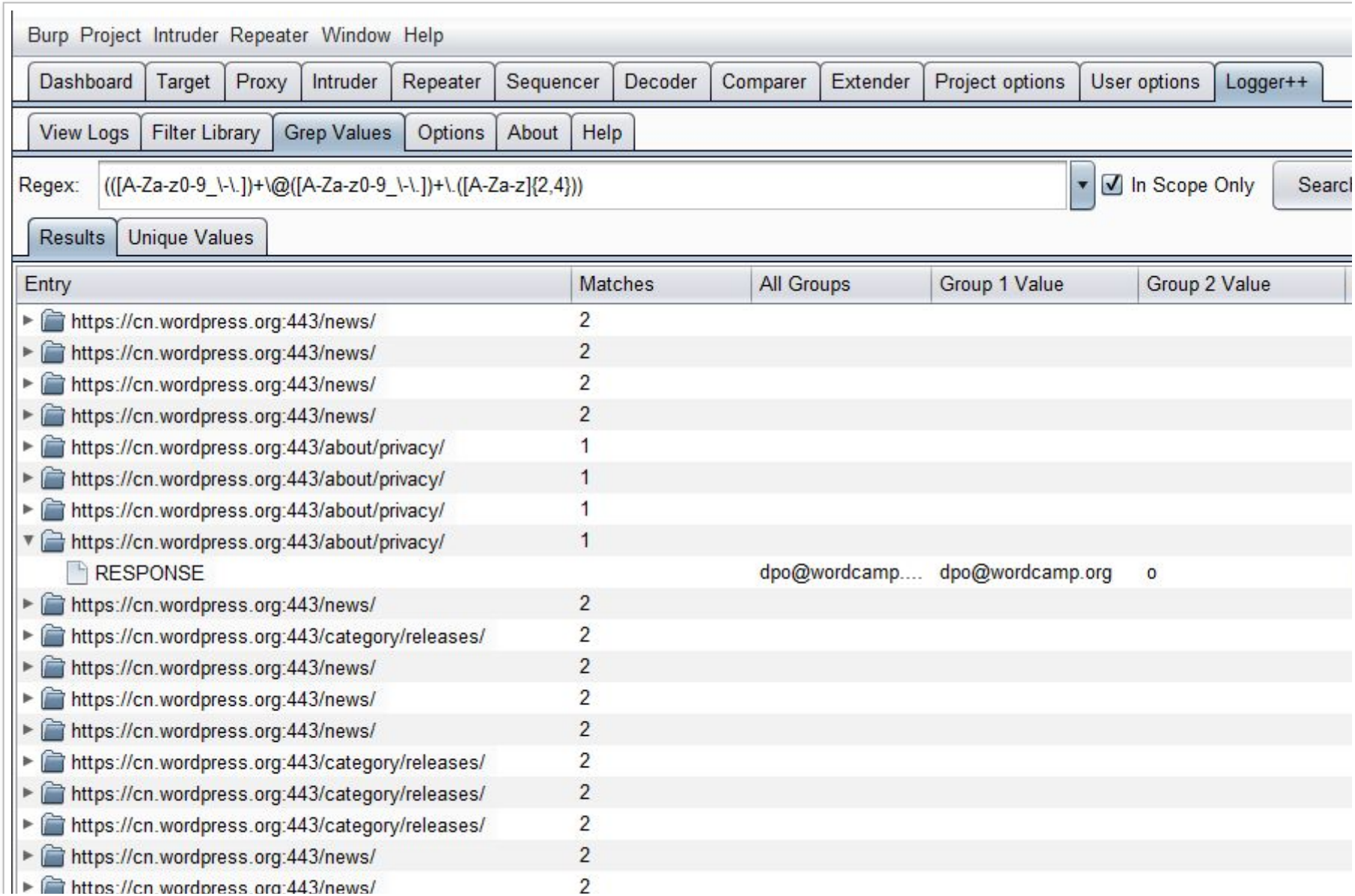
```
RESPONSE == /(([A-Za-z0-9_\-\.])+\@([A-Za-z0-9_\-\.])+\.([A-Za-z]{2,4}))/
```

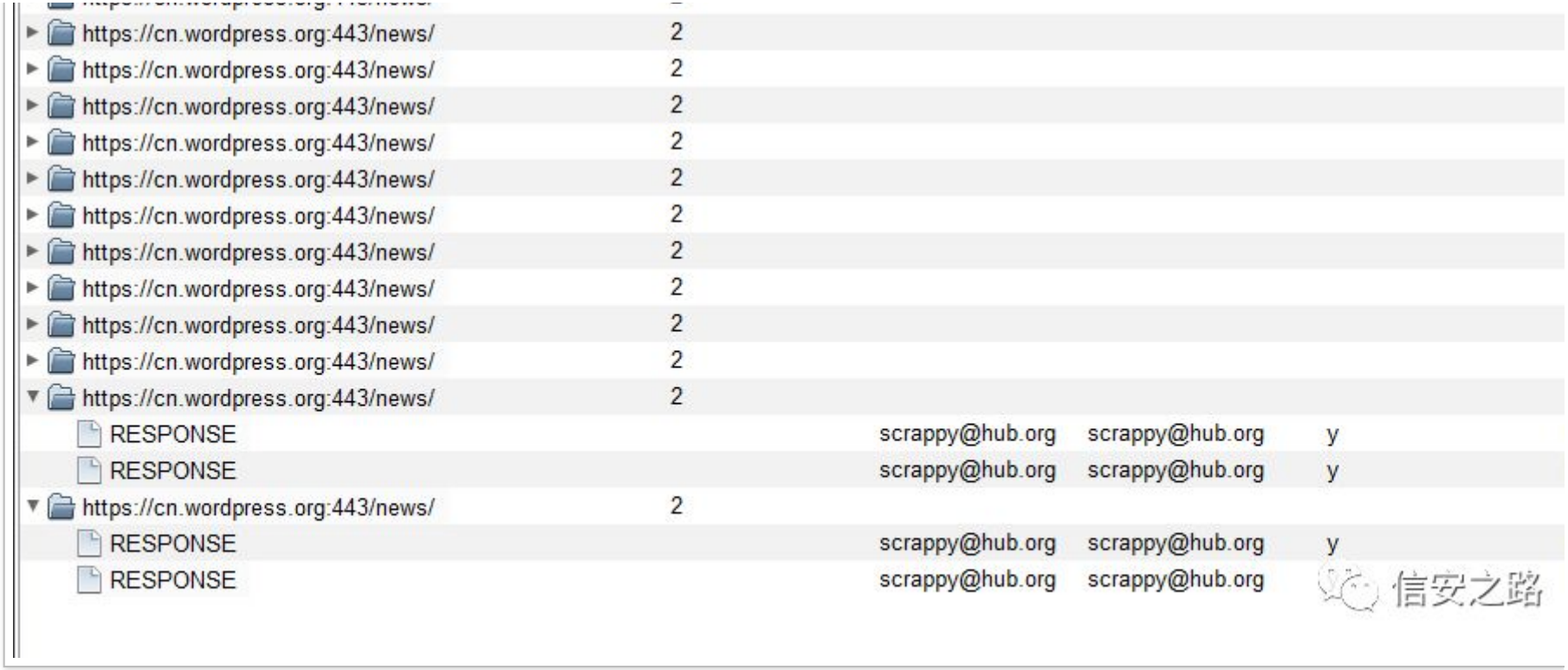
匹配特定邮箱

```
RESPONSE == /(([A-Za-z0-9_\-\.])+\@test.com)/
```

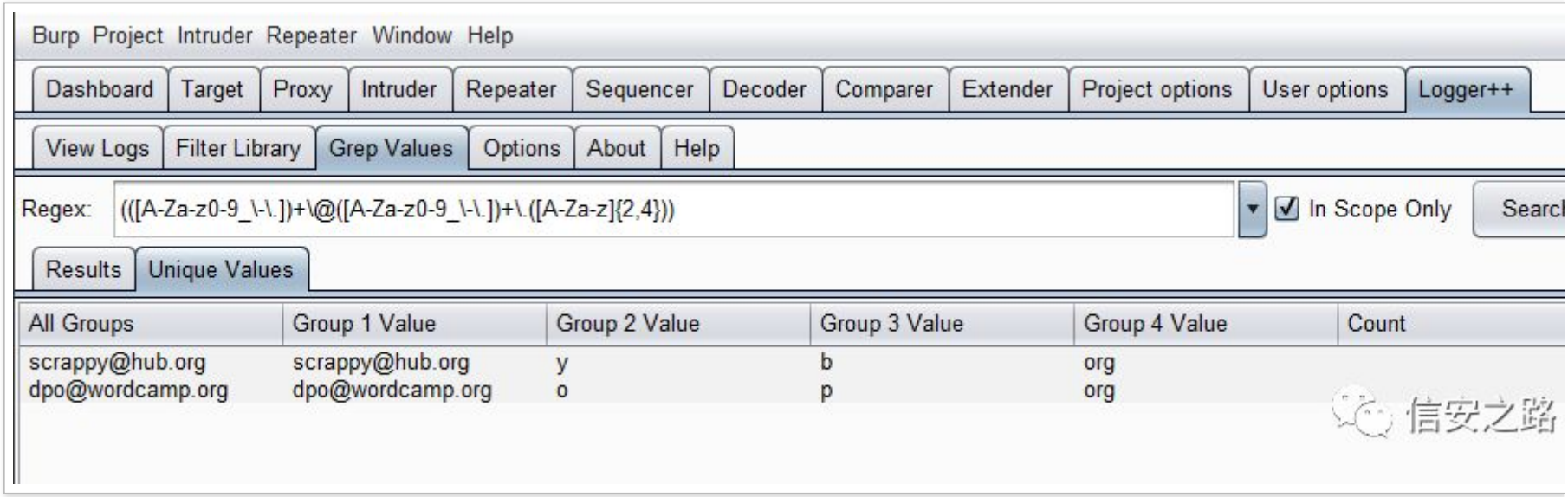
模糊关键字匹配邮箱

```
RESPONSE == /(([A-Za-z0-9_\-\.])+\@(.*)test(.*))/
```




https://cn.wordpress.org:443/news/ 2
RESPONSE scrappy@hub.org scrappy@hub.org y
信安之路

对结果进行去重



潜在的 CORS 配置不当 #4

```
RESPONSEHEADERS == /Access-Control-Allow-Origin: null/
```

```
RESPONSEHEADERS == /Access-Control-Allow-Origin: \*/
```

寻找潜在的 SSRF / 开放重定向 #5

开放重定向

根据响应头

```
ResponseHeaders == /(Location)/
```

根据参数名称

```
QUERY == /(url(.*)=)/ || REQUEST == /(url(.*)=)/

QUERY == /(uri(.*)=)/ || REQUEST == /(uri(.*)=)/

QUERY == /(path(.*)=)/ || REQUEST == /(path(.*)=)/

QUERY == /(href(.*)=)/ || REQUEST == /(href(.*)=)/

QUERY == /(redirect(.*)=)/ || REQUEST == /(redirect(.*)=)/
```

寻找参数中的图片

```
QUERY == /(img(.*)=)/ || REQUEST == /(img(.*)=)/

QUERY == /(pic(.*)=)/ || REQUEST == /(pic(.*)=)/

QUERY == /(\.png)/ || REQUEST == /(\.png)/

QUERY == /(\.jpg)/ || REQUEST == /(\.jpg)/

QUERY == /(\.gif)/ || REQUEST == /(\.gif)/
```

寻找潜在的 JSONP 调用 #6

基于参数

```
REQUEST == /(callback(.*)=)/ || QUERY == /(callback(.*)=)/
```

基于响应特征

```
RESPONSE == /(.+\(\[(.*)\]\))/ && RESPONSEHEADERS == /application\/json/
```

寻找潜在的越权漏洞 #7

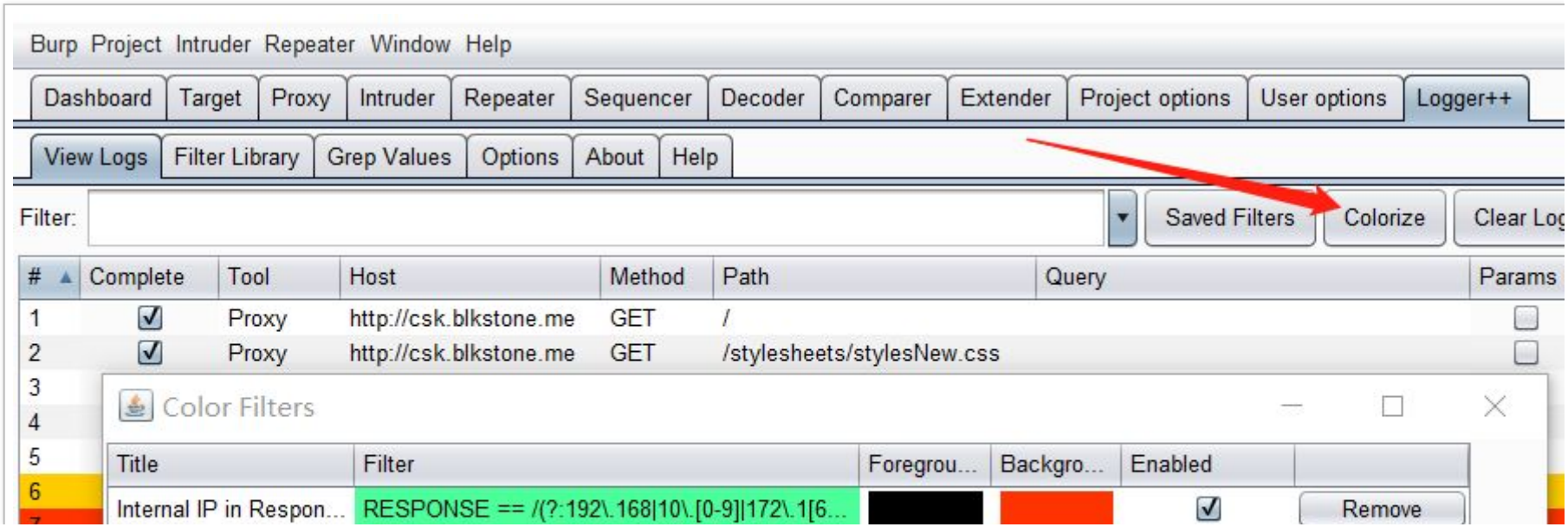
```
REQUEST == /(id(.*)=)/ || QUERY == /(id(.*)=)/
```

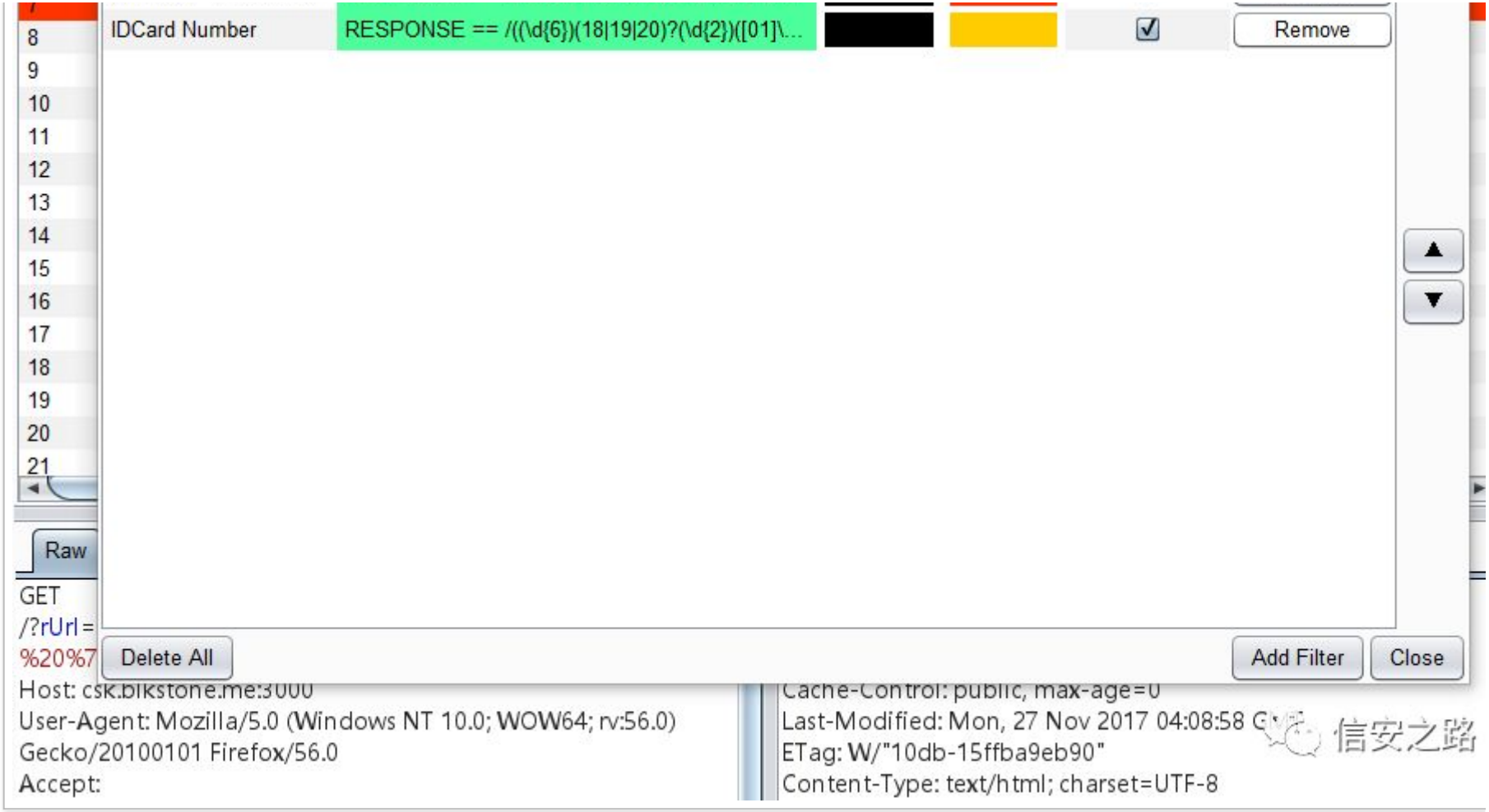
着色过滤器 (color filter)

举一个 内网 IP 泄露检测规则 和 身份证号检测规则 配置的例子

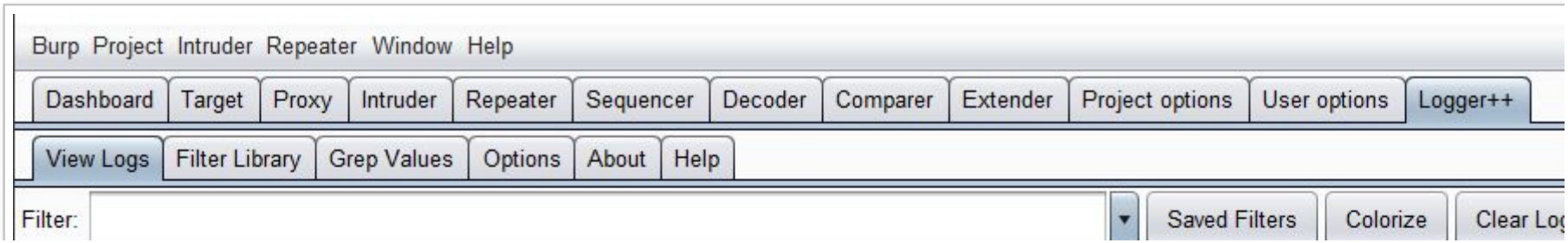
> Logger++ => View Logs => Colorize

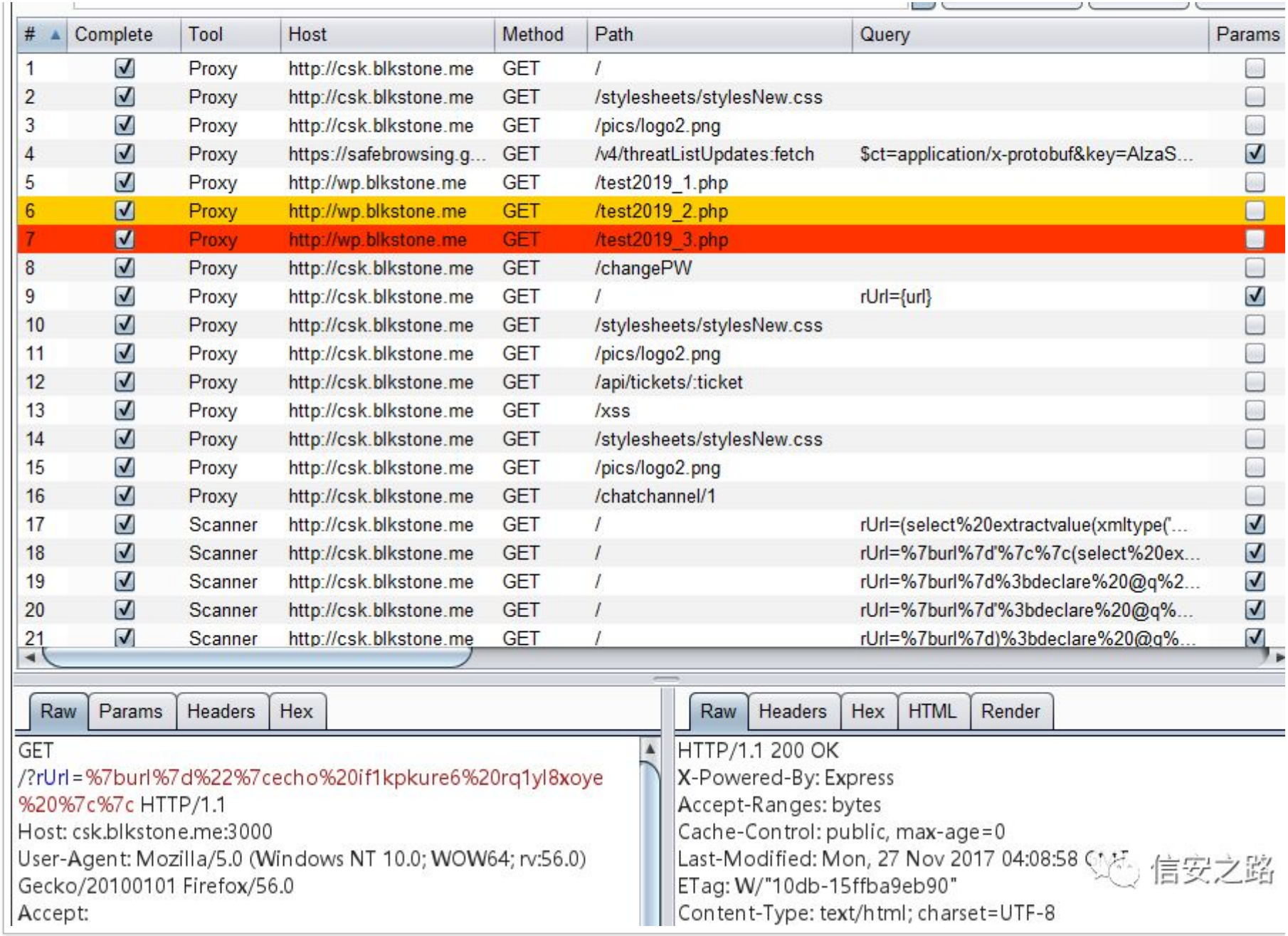
着色过滤器配置也可以在 Options 中配置批量导入 / 导出。




IDCard Number
RESPONSE == /((\d{6})(18|19|20)?(\d{2})([01]\...
Remove
Raw
GET
/?rUrl=
%20%7
Delete All
Add Filter
Close
Host: csk.blkstone.me:3000
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept:
Cache-Control: public, max-age=0
Last-Modified: Mon, 27 Nov 2017 04:08:58 G
ETag: W/"10db-15ffba9eb90"
Content-Type: text/html; charset=UTF-8
信安之路

着色效果如下


Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Logger++
View Logs Filter Library Grep Values Options About Help
Filter:
Saved Filters
Colorize
Clear Log



#	Complete	Tool	Host	Method	Path	Query	Params
1	☑	Proxy	http://csk.blkstone.me	GET	/		☐
2	☑	Proxy	http://csk.blkstone.me	GET	/stylesheets/stylesNew.css		☐
3	☑	Proxy	http://csk.blkstone.me	GET	/pics/logo2.png		☐
4	☑	Proxy	https://safebrowsing.g...	GET	/v4/threatListUpdates:fetch	$ct=application/x-protobuf&key=AIzaS...	☑
5	☑	Proxy	http://wp.blkstone.me	GET	/test2019_1.php		☐
6	☑	Proxy	http://wp.blkstone.me	GET	/test2019_2.php		☐
7	☑	Proxy	http://wp.blkstone.me	GET	/test2019_3.php		☐
8	☑	Proxy	http://csk.blkstone.me	GET	/changePW		☐
9	☑	Proxy	http://csk.blkstone.me	GET	/	rUrl={url}	☑
10	☑	Proxy	http://csk.blkstone.me	GET	/stylesheets/stylesNew.css		☐
11	☑	Proxy	http://csk.blkstone.me	GET	/pics/logo2.png		☐
12	☑	Proxy	http://csk.blkstone.me	GET	/api/tickets/:ticket		☐
13	☑	Proxy	http://csk.blkstone.me	GET	/xss		☐
14	☑	Proxy	http://csk.blkstone.me	GET	/stylesheets/stylesNew.css		☐
15	☑	Proxy	http://csk.blkstone.me	GET	/pics/logo2.png		☐
16	☑	Proxy	http://csk.blkstone.me	GET	/chatchannel/1		☐
17	☑	Scanner	http://csk.blkstone.me	GET	/	rUrl=(select%20extractvalue(xmltype('...	☑
18	☑	Scanner	http://csk.blkstone.me	GET	/	rUrl=%7burl%7d'%7c%7c(select%20ex...	☑
19	☑	Scanner	http://csk.blkstone.me	GET	/	rUrl=%7burl%7d%3bdeclare%20@q%2...	☑
20	☑	Scanner	http://csk.blkstone.me	GET	/	rUrl=%7burl%7d'%3bdeclare%20@q%...	☑
21	☑	Scanner	http://csk.blkstone.me	GET	/	rUrl=%7burl%7d)%3bdeclare%20@q%...	☑

Raw | Params | Headers | Hex

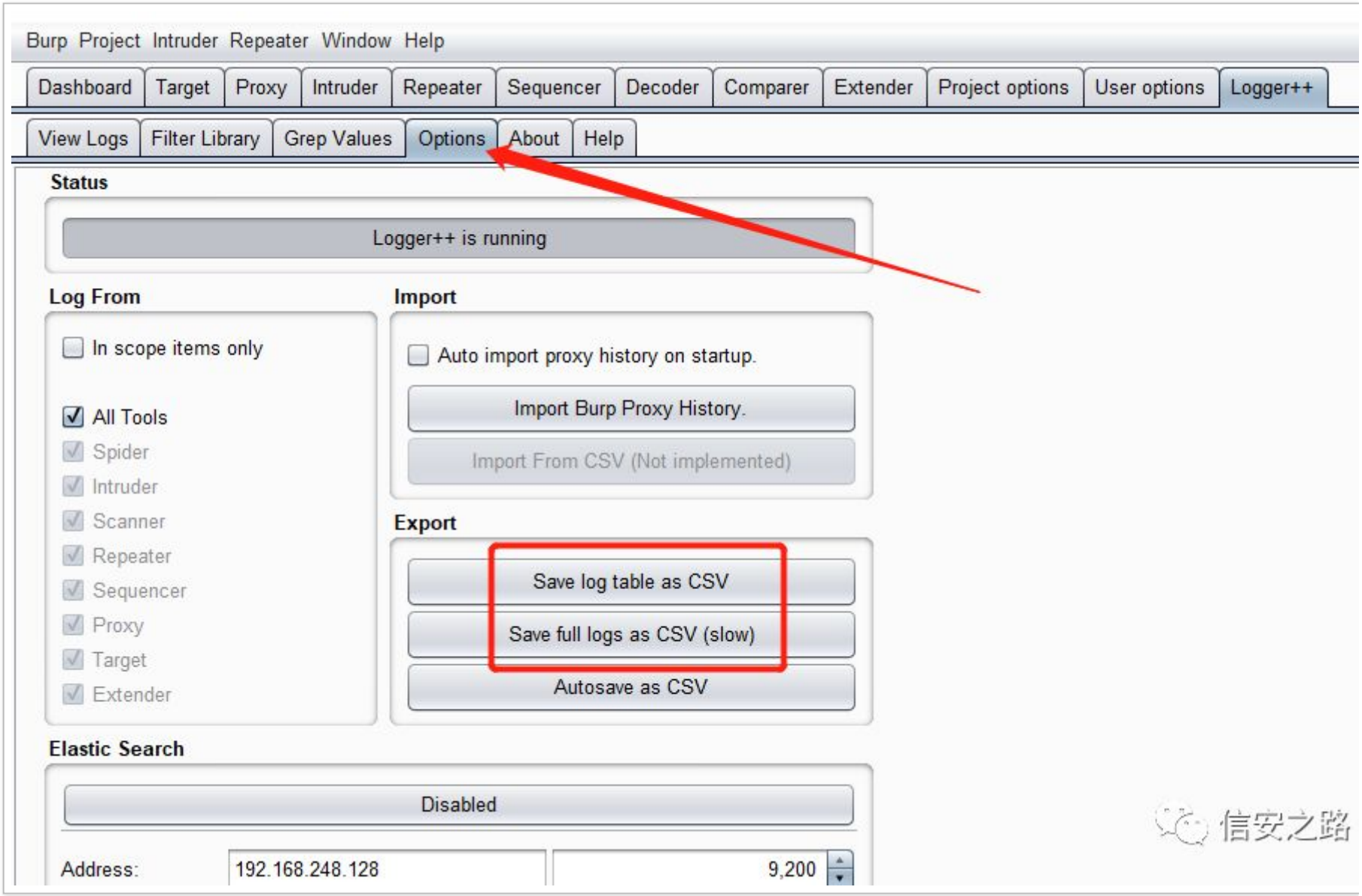
```
GET
/?rUrl=%7burl%7d%22%7cecho%20if1kpkure6%20rq1yl8xoye
%20%7c%7c HTTP/1.1
Host: csk.blkstone.me:3000
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept:
```

Raw | Headers | Hex | HTML | Render

```
HTTP/1.1 200 OK
X-Powered-By: Express
Accept-Ranges: bytes
Cache-Control: public, max-age=0
Last-Modified: Mon, 27 Nov 2017 04:08:58
ETag: W/"10db-15ffba9eb90"
Content-Type: text/html; charset=UTF-8
```


记录流量日志

> Logger++ => Options => Export => Save log table as csv.



结语

充分利用 Logger++ 插件的过滤器，能帮助你在海量 HTTP 请求中更方便快捷地定位到某些脆弱的 HTTP 请求。如果你有其他不错的过滤器规则，也欢迎在下面留言。

参考资料

BurpSuiteLoggerPlusPlus Wiki

> https://github.com/nccgroup/BurpSuiteLoggerPlusPlus/wiki

Logger++ 示例过滤器

> https://github.com/nccgroup/BurpSuiteLoggerPlusPlus/wiki/Example-Filters

Burp Suite 存储日志分析

> https://zhuanlan.zhihu.com/p/28284124

Burp Suite 神器的日志存储

> https://zhuanlan.zhihu.com/p/28231222

经验分享 Burpsuite 插件的使用

> https://cloud.tencent.com/developer/article/1015187