Lab HTTP Request Smuggling

Motivation

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.

Learning outcomes

After completing this lab students should be able to:

- Understand HTTP messages
- Explain HTTP headers
- Describe how request smuggling work
- Detect HTTP request smuggling
- Execute HTTP request smuggling
  Explain how HTTP request smuggling can be prevented

How HTTP works

HTTP (Hypertext Transfer Protocol) is a simple, text-based, request/response protocol. A client (browser, script, scanner) opens a TCP connection to a server, sends a request, and the server replies with a response.

What does a http request look like?

An HTTP request has:

1. a request line
2. headers
3. a blank line
4. an optional message body

```
GET /index.html HTTP/1.1
Host: www.example.re
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.1)
Accept: text/html
Accept-Language: en-US, en; q=0.5
Accept-Encoding: gzip, deflate

q=smuggling
```

What are http headers?

Http headers pass additional information with a request/ response message.

- General headers: apply to both requests and responses (e.g., Connection, Date)
- Request headers: sent by the client (e.g.: Host, User-Agents, Cookie, Content-Length)
- Response headers: sent by the server (e.g., Server, Location)
- Entity/ Representation headers: describe the message body (e.g., Content-Type, Content-Length, Content-Encoding)

Question: What headers play an important role in HTTP request smuggling and what is their function? Provide an example on how an HTTP request with those headers looks like.

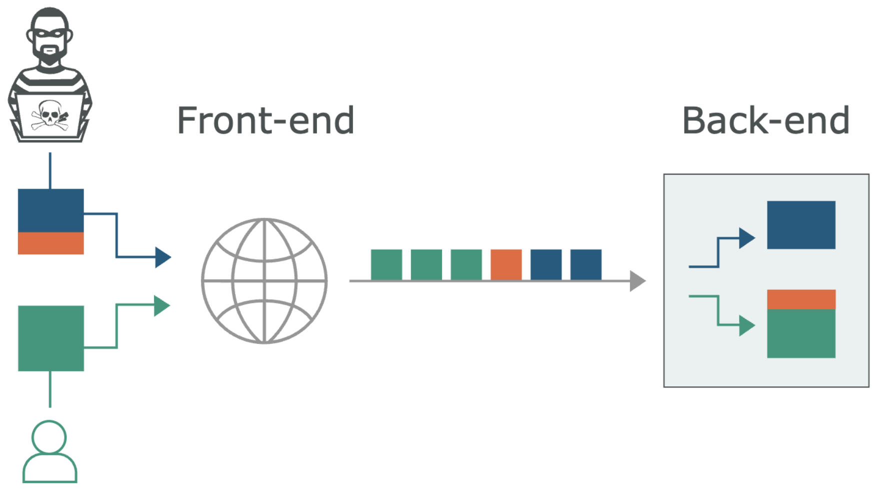
Task 1: Redirect victim to another page

Craft a raw HTTP request from the attacker to the front-end that exploits a Transfer-Encoding / Content-Length parsing mismatch so the victim gets redirected to /admin.

HTTP Request Smuggling

HTTP request smuggling (HRS) is an exploit that takes advantage of inconsistencies in HTTP request parsing between intermediary components (reverse proxies & load balancers) and back-end servers. The vulnerability typically arises because the HTTP/1 specification provides two different ways to specify where a request ends, those being the Content-Length header and the Transfer-Encoding header.

By constructing requests that different components interpret differently, an attacker can desynchronise the request stream. This makes it possible to inject hidden requests into the back-end connection, which leads to cache poisoning, credential hijacking, bypass of access controls, or direct compromise of other users' sessions. Request smuggling is primarily associated with HTTP/1 requests. However, a website supporting HTTP/2 may be vulnerable, depending on their back-end architecture.



Question: Explain HTTP request smuggling in your own words

Task 2: Delete the victim account

Craft a raw HTTP request from the attacker to the front-end that exploits a Transfer-Encoding / Content-Length parsing mismatch so the victim deletes his account by redirecting him to the /deleteAccount page.

Question: What different HTTP request smuggling attack types exist? Explain them shortly.

Question: Craft an HTTP request smuggling request in an attack type of your choice

Task 3: Leak the HTTP request headers

Craft a raw HTTP request from the attacker to the front-end that exploits a Transfer-Encoding / Content-Length parsing mismatch to leak sensitive headers by redirecting the user to the /postComment page.

Question: Name and explain 3 defense types against HTTP request smuggling