USER FLOW

STEP 1: Analyze Source Code

- Participants are provided with the full source code of the application
- They need to understand the system architecture, including the frontend, backend, caching mechanism, and the security measures(like CSP)

STEP 2: Discover Vulnerability

- Through code analysis, participants identify the HTTP request smuggling vulnerability
- They recognize the mismatch between HTTP/2 (frontend) and HTTP/1.1 (backend)

STEP 3: Craft Malicious Payload

- Participants need to create a payload that can bypass the Content Security Policy (CSP)
- They must find a way to inject a script, either by generating a valid nonce or exploiting a CSP bypass

STEP 4: Cache Poisoning Attack

- Using the HTTP request smuggling vulnerability, participants inject their malicious payload into the blog post
- The poisoned content gets cached by the system

STEP 5: Collect JWT Tokens

- We report the blog and get the admin see this blog.
- As users (including admins) view the poisoned blog post, the malicious script steals their JWT tokens
- Participants set up a collection point for these stolen tokens

STEP 6: Bypass Custom Headers

- Analyze the 'X-Service-Auth' header generation mechanism
- Create a valid 'X-Service-Auth' header for admin requests

STEP 7: Access Admin Dashboard

- Using the admin JWT and custom headers to access the admin dashboard
- Navigate through any additional security measures

STEP 8: Retrieve and Submit Flag

- Once in the admin dashboard, locate and retrieve the flag
- Submit the flag to complete the challenge