

블로그에서 검색 유입을 판별하는 원리

*Yun @ Baekjoon Best Conference 2023
Winter*

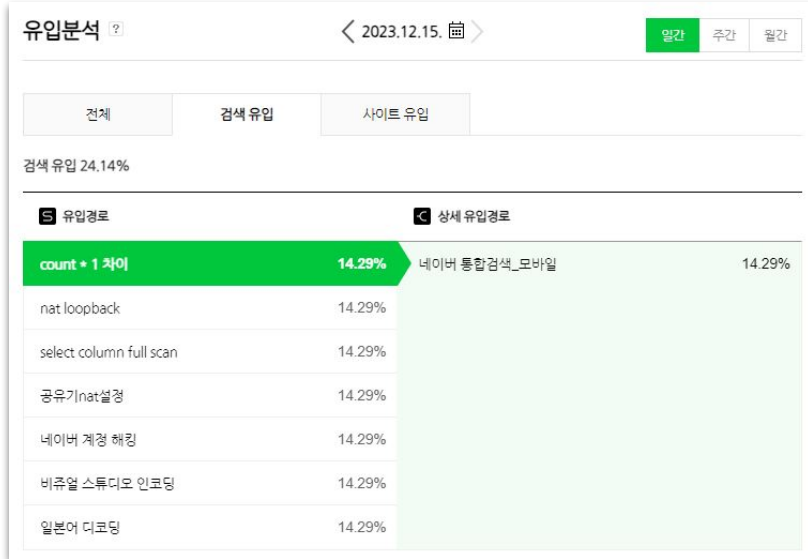
발표자 소개

티스토리 백엔드 4년차 개발자

- Kotlin & Spring Boot 로 일하는 중
- PHP를 혐오함
- 카카오 코딩테스트 문제 검수 담당
- 알고리즘 문제 해결이 취미
- 최근 갑자기 코드포스 오렌지가 되고 싶어서 환장함



블로그 하시는 분?



HTTP 리퍼러

한글 16개 언어 ▾

문서 토론

읽기 편집 역사 보기 도구 ▾

위키백과, 우리 모두의 백과사전.

HTTP 리퍼러(HTTP referer)는 웹 브라우저로 월드 와이드 웹을 서핑할 때, 하이퍼링크를 통해서 각각의 사이트로 방문시 남는 흔적을 말한다.

예를 들어 A라는 웹 페이지에 B 사이트로 이동하는 하이퍼링크가 존재한다고 하자. 이때 웹 사이트 이용자가 이 하이퍼링크를 클릭하게 되면 웹 브라우저에서 B 사이트로 참조 주소(리퍼러)를 전송하게 된다. B 사이트의 관리자는 이 전송된 리퍼러를 보고 방문객이 A 사이트를 통해 자신의 사이트에 방문한 사실을 알 수 있다.

웹 사이트의 서버 관리자가 사이트 방문객이 어떤 경로로 자신의 사이트에 방문했는지 알아볼 때 유용하게 사용된다.

하지만, 리퍼러는 조작또한 가능하기 때문에 리퍼러 정보를 사용할 때에는 보안에 항상 주의해야 한다.

한편, HTTP 리퍼러를 정의한 RFC에서 'referrer'를 'referer'라고 잘못 친 것에서 기인하여 HTTP 리퍼러는 'HTTP referer'라고 불린다.^[1]

각주 [편집]

- ↑ "HTTP:The Definitive Guide" .



지속성 · 압축 · HTTPS

요청 방식

OPTIONS · GET · HEAD · POST · PUT · DELETE · TRACE · CONNECT · PATCH

헤더 필드

쿠키 · ETag · 위치 · 리퍼러 · DNT · XFF

상태 코드

301 Moved Permanently · 302 Found · 303 See Other · 403 Forbidden · 404 Not Found · 451 Unavailable For Legal Reasons

V · T · E

https://en.wikipedia.org/wiki/HTTP_referer

← → ↻ 🏠 🔍 search.naver.com/search.naver?where=nexearch&sm=top_hyty&fbm=0&ie=utf8&query=nat+loopback

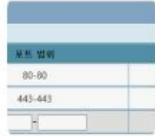
N nat loopback

VIEW 이미지 지식IN 인플루언서 동영상 쇼핑 > ...

blog.naver.com > dsyun96

SK 브로드밴드 공유기의 NAT loopback 문제

2022.04.05. 이는 **NAT loopback** 기능을 공유기가 지원하지 않아서인데, 만약 지원해주면 똑같이 헬로월드가 잘 댄겠지만 SK 브로드밴드 공유기는 이런 기능이 없어서 안 된다고 한다. 참고로 iptime 공유기는 이 기...




포트 범위
80-90
443-443

blog.naver.com > micel79

SK 브로드밴드 Nat LoopBack :: 일명 (헤어핀Nat)

2018.01.15. **NAT(Network Address Tranlation) Loop-Back** occurs when somebody from within the local network tries to access the server computer using the external public IP Address behind **NAT** enabled router. ...



192 + 2011

N nat loopback

- VIEW
- 이미지
- 지식IN
- 인플루언서
- 동영상
- 쇼핑
- >
- ...

- blog.naver.com > dsyun96

SK 브로드밴드 공유기의 NAT loopback 문제

2022.04.05. 이는 NAT loopback 기능을 공유기가 지원하지 않아서인데, 만약 지원해주면 똑같이 헬로월드가 잘 댔겠지만 SK 브로드밴드 공유기는 이런 기능이 없어서 안 된다고 한다. 참고로 iptime 공유기는 이 기...



포트 범위	내부 IP	외부 IP
80-90		
443-443		
- blog.naver.com > micel79

SK 브로드밴드 Nat LoopBack :: 일명 (헤어핀Nat)

2018.01.15. NAT(Network Address Tranlation) Loop-Back occurs when somebody from within the local network tries to access the server computer using the external public IP Address behind NAT enabled router. ...



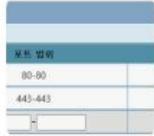
N nat loopback

VIEW 이미지 지식iN 인플루언서 동영상 쇼핑 > ...

blog.naver.com > dsyun96

SK 브로드밴드 공유기의 NAT loopback 문제

2022.04.05. 이는 **NAT loopback** 기능을 공유기가 지원하지 않아서인데, 만약 지원해주면 똑같이 헬로월드가 잘 댄겠지만 SK 브로드밴드 공유기는 이런 기능이 없어서 안 된다고 한다. 참고로 iptime 공유기는 이 기...




포트 범위
80-90
443-443

blog.naver.com > micel79

SK 브로드밴드 Nat LoopBack :: 일명 (헤어핀Nat)

2018.01.15. **NAT**(Network Address Tranlation) **Loop-Back** occurs when somebody from within the local network tries to access the server computer using the external public IP Address behind **NAT** enabled router. ...



윤준

활동일지

프로필 >

+ 이웃추가

카테고리

전체보기 (1414)

잡담 (405)

일기 (67)

【블로그】 주간일기 캘린더

Language (24)

C/C++ (12)

Python (2)

Java (3)

Kotlin (5)

JavaScript (1)

PHP (1)

Database (3)

N/A (1)

Oracle (2)

MySQL (0)

Algorithm (7)

PS (432)

N/A (20)

BOJ (265)

CodeUp (8)

ALGOSPOT (14)

doyleat (16)

LeetCode (11)

Project Euler (20)

hacker.org (35)

Codeforces (31)

AtCoder (12)

Settings (6)

공부 (178)

코딩대회 (26)

이것저것 (82)

일러스트 (79)

게임 (105)

RSS 2.0 | RSS 1.0 | ATOM 1.0

🔍

Setting

SK broadband 공유기의 NAT loopback 문제

윤준 2022. 4. 5. 23:15

URL 복사 +이웃추가

현재 내 공유기가 사용 중인 공인 IP는 110.14.2.9이다.

여기서 그랜 노트북은 LAN으로, 핸드폰과 맥북과 라즈베리파이는 와이파이로 연결되어 있으며 서버 IP로 192.168.35.xxx를 사용한다.

나는 라즈베리파이를 통해 서버를 띄웠는데, 외부에서 110.14.2.9 아이피를 통해 80포트로 들어오면 이걸 라즈베리파이가 연결된 192.168.35.xxx 아이피의 80포트로 들어가게끔 포트포워딩을 해준 상태이다.

포트 번호	외부 IP	포트 번호	외부 IP	연결	비고
192.168.35.252	192.168.35.252	80-80	80	raspberrypi-nigin	서버
192.168.35.252	192.168.35.252	443-443	443	raspberrypi-nigin	서버
	192.168.35.252	80			9.9.9

따라서, 다른 사람들은 110.14.2.9 아이피로 접속하게 되면 문제없이 내 블로그를 볼 수 있다.

하지만 내 컴퓨터와 핸드폰은 상황이 다르다. 요 녀석들은 같은 내부망을 사용 중이기 때문에 192.168.35.xxx로 접속하면 블로그를 볼 수 있지만 다른 사람처럼 110.14.2.9 아이피로 접속하면 연결조차 되지 않는다.

← → ↻ 🏠 주의 요함 | <https://192.168.35.252>

Hello, world!

← → ↻ 🏠 110.14.2.9



사이트에 연결할 수 없음

Network Performance Memory Application Security Lighthouse Recorder

Filter: Invert Hide data URLs Hide extension URLs

Blocked response cookies Blocked requests 3rd-party requests

Name: **rdm=1&px=363&py=220&xs...**

General

- Request URL: <https://blog.naver.com/dsyun96/222692746029>
- Request Method: GET
- Status Code: 200 OK
- Remote Address: 23.201.36.184:443
- Referrer Policy: unsafe-url

Response Headers

- Cache-Control: no-cache
- Connection: keep-alive
- Content-Encoding: gzip
- Content-Length: 1280
- Content-Type: text/html;charset=UTF-8
- Date: Sun, 17 Dec 2023 03:12:54 GMT
- Expires: Thu, 01 Jan 1970 00:00:00 GMT
- Referrer-Policy: unsafe-url
- Server: rfront
- Set-Cookie: JSESSIONID=CD9C262688E1DC24C149E39AB6C3237B:jm1; Path=/; Secure; HttpOnly
- Vary: Accept-Encoding

Request Headers

- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
- Accept-Encoding: gzip, deflate, br
- Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7,jav;q=0.6
- Connection: keep-alive
- Cookie: NNB=UPVL54VKQFZGK; BA_DEVICE=bb4c0098-0221-4912-b006-aedb4b45aa11; NSCS=1; _ga_8P4PY65Y2Z=GS1.1.1702096252.1.0.1702096252.60.0.0; _ga_GN4BHvX9DS=GS1.1.1702096252.1.0.1702096252.60.0.0; _ga=GA.2817269547.1702096253; rx_xsl=2; page_uid=iUHDXqVOZCsnFPB8sssssvf-024440; JSESSIONID=4AF3AC3B0F49CE71E1266F783A80A3A3:jm1
- Dnt: 1
- Host: blog.naver.com
- Referer: **https://search.naver.com/search.naver?where=nxsearch&sm=top_hiy&fbm=0&ie=utf8&query=nat-loopback**
- Sec-CH-UA: not_A brand v=8, Chromium v=120, Google Chrome v=120
- Sec-CH-UA-Mobile: 70
- Sec-CH-UA-Platform: "Windows"
- Sec-Fetch-Dest: document
- Sec-Fetch-Mode: navigate
- Sec-Fetch-Site: same-site
- Sec-Fetch-User: 71
- Upgrade-Insecure-Requests: 1
- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36

- 윤준
- 필요할때
- 프로필
- + 이웃추가
- 카테고리
- 전체보기 (1414)
- 잡담 (405)
- 일기 (67)
- 【분류】주간일기
편지
- Language (24)
- C/C++ (12)
- Python (2)
- Java (3)
- Kotlin (5)
- JavaScript (1)
- PHP (1)
- Database (3)
- N/A (1)
- Oracle (2)
- MySQL (0)
- Algorithm (7)
- PS (432)
- N/A (20)
- BOJ (265)
- CodeUp (8)
- ALGOSPOT (14)
- dovelet (16)
- LeetCode (11)
- Project Euler (20)
- hacker.org (35)
- Codeforces (31)
- AlCoder (12)
- Settings (6)
- 공부 (178)
- 코딩대회 (26)
- 이것저것 (82)
- 블로그 (79)
- 즐거움 (105)

Setting

SK broadband 공유기의 NAT loopback 문제

윤준 2022. 4. 5. 23:15

URL 복사 +이웃추가

현재 내 공유기가 사용 중인 공인 IP는 110.14.2.9이다.
 여기서 그랜 노트북은 LAN으로, 핸드폰과 맥북과 라즈베리파이는 와이파이로 연결되어 있으며 사설 IP로
 192.168.35.xxx를 사용한다.

검색 유입 24.14%

유입경로	상세 유입경로
count * 1 차이	네이버 통합검색_PC
nat loopback	+1
select column full scan	
공유기 nat설정	
네이버 계정 해킹	
비주얼 스튜디오 인코딩	
일본어 디코딩	



Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder

Blocked response cookies Blocked requests 3rd-party requests

Network

rdm=1&px=363&py=220&sx=222692746029

Request URL: https://blog.naver.com/dsyun96/222692746029
 Request Method: GET
 Status Code: 200 OK
 Remote Address: 23.201.36.184:443
 Referrer Policy: unsafe-url

Response Headers

Cache-Control: no-cache
 Connection: keep-alive
 Content-Encoding: gzip
 Content-Length: 1280
 Content-Type: text/html; charset=UTF-8
 Date: Sun, 17 Dec 2023 03:12:54 GMT
 Expires: Thu, 01 Jan 1970 00:00:00 GMT
 Referrer-Policy: unsafe-url
 Server: rfront
 Set-Cookie: JSESSIONID=CD9C262688E1DC24C149E39A86C3237B:jm1; Path=/; Secure; HttpOnly
 Vary: Accept-Encoding

Request Headers

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
 Accept-Encoding: gzip, deflate, br
 Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7,jav;q=0.6
 Connection: keep-alive
 Cookie: NNB=UPVL54VKQFZGK; BA_DEVICE=bb4c0098-0221-4912-b006-aedb4b45aa11; NSCS=1; _ga_8P4PY65Y2Z=GS1.1.1702096252.1.0.1702096252.60.0.0; _ga_GN4BHvX9DS=GS1.1.1702096252.1.0.1702096252.60.0.0; _ga=GA1.287269547.1702096253; nx_xsl=2; page_uid=iUHDXqVOZCsnFPpB8ossssvf-024440; JSESSIONID=4AF3AC3B0F49CE71E1266F783A80A3A3:jm1

Dnt: 1
 Host: blog.naver.com

Referer: https://search.naver.com/search.naver?where=nxsearch&sm=top_hiy&fbm=0&ie=utf8&query=nat+loopback

Sec-CH-UA: not_A brand v=8, Chromium v=120, Google Chrome v=120
 Sec-Ch-UA-Mobile: 70
 Sec-Ch-UA-Platform: "Windows"
 Sec-Fetch-Dest: document
 Sec-Fetch-Mode: navigate
 Sec-Fetch-Site: same-site
 Sec-Fetch-User: 71
 Upgrade-Insecure-Requests: 1
 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36



사이트에 연결할 수 없음

원리는 이게 전부입니다. 쉽죠?

좀 더 심화된 내용으로
들어가봅시다.



2023.07.21 09:19 · 질문/기타

조회수 179 · 댓글 2

와.... 이젠 블로그 통계까지 엉망이네요 ^{hot}

정상적으로 검색포털 통해 유입된 페이지뷰를,
기타유입으로 잡아버리네요.

이제 광고 강제 끼워넣기에 이어
모든 티스토리 블로그를 저품질 블로그로 만드는게 목표인지...
힘빠집니다 참;

[신고](#)



2023.07.21 09:28 · 신고

뭔가 달라지는거 같긴하네요. ...



2023.07.22 03:23 · 신고

어제 어느 분께서 네이버 검색어 없이 유입 잡힌다고 하셨는데, 다음
도 마찬가지네요.

<http://search.daum.net/> 이렇게만 뜹니다. 검색어까지 뜨는건
간혹 가다 있고요.

내용을 입력하세요
하루에 10개까지 작성 가능합니다

0/500

완료

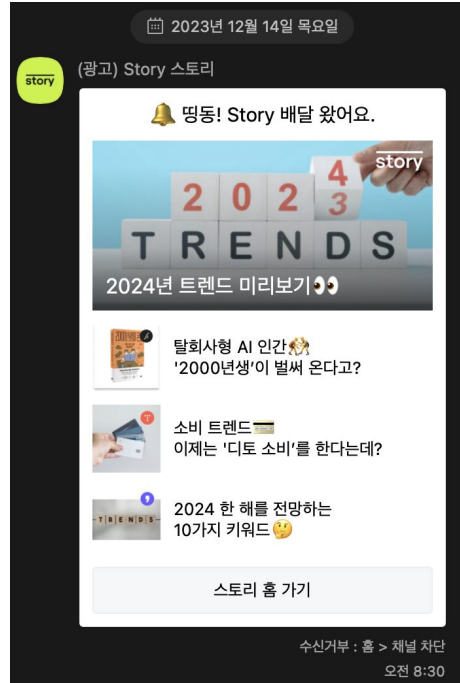
원인은 바로 며칠 전에 추가된
자동로그인 프로세스

자동로그인 탄생 배경

티스토리, 브런치(現 브런치스토리), 카카오토리가 묶여서 [스토리홈](#) 탄생

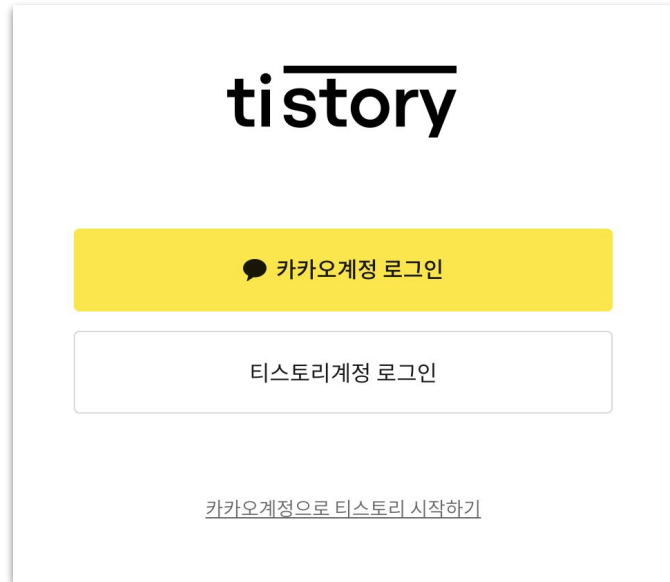
자동로그인 탄생 배경

티스토리, 브런치(現 브런치스토리), 카카오스토리가 묶여서 스토리홈 탄생
카카오톡에 스토리 채널이 생겨서 카카오톡 → 티스토리 유입 동선 발생



자동로그인 탄생 배경

티스토리, 브런치(現 브런치스토리), 카카오토리가 묶여서 [스토리홈](#) 탄생
카카오톡에 스토리 채널이 생겨서 카카오톡 → 티스토리 유입 동선 발생
티스토리는 카카오 계정으로 로그인 가능한 상황



자동로그인 탄생 배경

티스토리, 브런치(現 브런치스토리), 카카오스토리가 묶여서 [스토리홈](#) 탄생
카카오톡에 스토리 채널이 생겨서 카카오톡 → 티스토리 유입 동선 발생
티스토리는 카카오 계정으로 로그인 가능한 상황

카카오톡에서는 어차피 로그인이 되어있으니까,
티스토리로 유입될 때 자동으로 로그인이 되면 편하지 않을까?!

자동로그인 탄생 배경

티스토리, 브런치(現 브런치스토리), 카카오토리가 묶여서 [스토리홈](#) 탄생
카카오톡에 스토리 채널이 생겨서 카카오톡 → 티스토리 유입 동선 발생
티스토리는 카카오 계정으로 로그인 가능한 상황

카카오톡에서는 어차피 로그인이 되어있으니까,
티스토리로 유입될 때 자동으로 로그인이 되면 편하지 않을까?!

자동로그인 탄생 🎉



자동로그인 동작 방식

1. 유저가 개별 유저블로그에 접근

자동로그인 동작 방식

1. 유저가 개별 유저블로그에 접근
2. 서버에서 자동로그인을 시켜야 하는 요청인지 판단

자동로그인 동작 방식

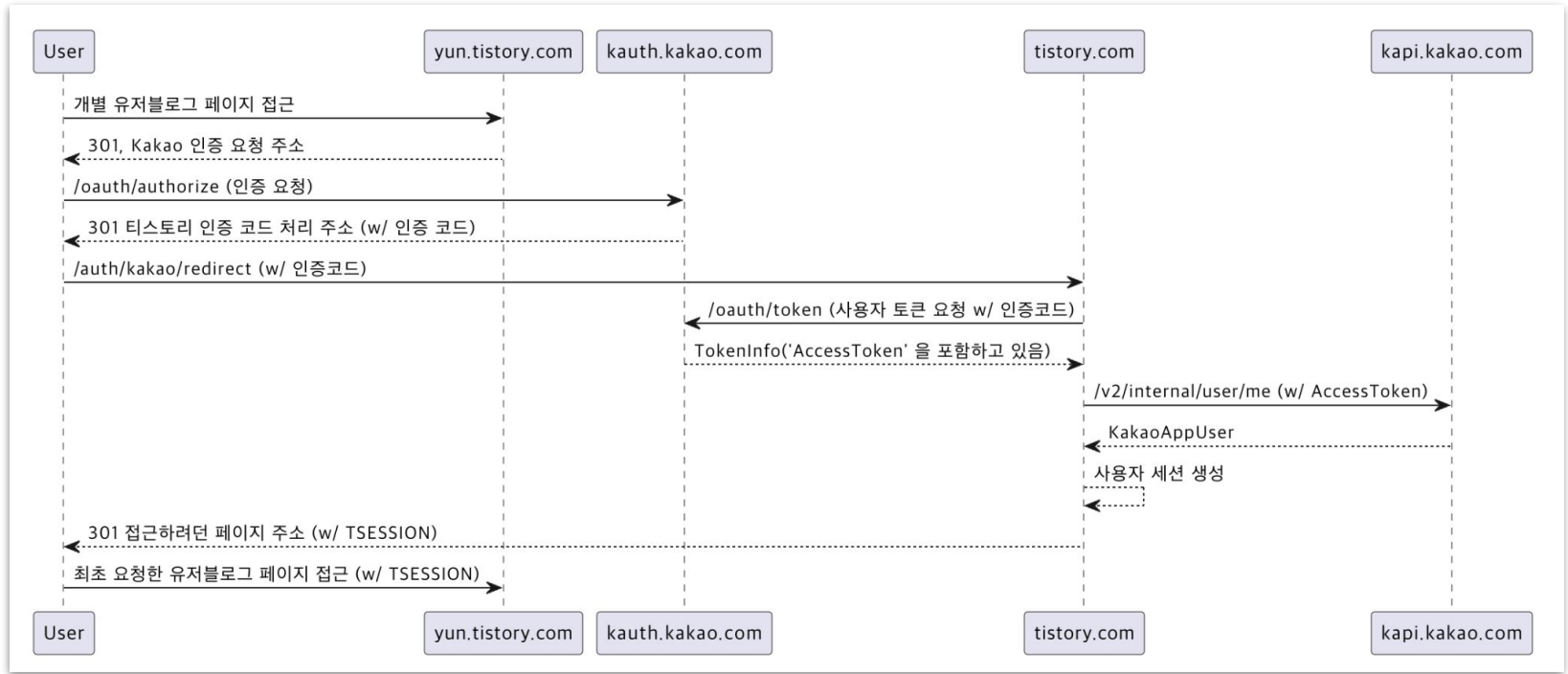
1. 유저가 개별 유저블로그에 접근
2. 서버에서 자동로그인을 시켜야 하는 요청인지 판단
 - 자동로그인 기능을 켜둔 상태인지
 - 자동로그인 방지를 위한 쿠키가 없는지
 - 티스토리 세션이 없는지
 - 봇 요청이 아닌지 등등


자동로그인 동작 방식

1. 유저가 개별 유저블로그에 접근
2. 서버에서 자동로그인을 시켜야 하는 요청인지 판단
 - 자동로그인 기능을 켜둔 상태인지
 - 자동로그인 방지를 위한 쿠키가 없는지
 - 티스토리 세션이 없는지
 - 봇 요청이 아닌지 등등
3. 자동로그인을 시켜야 한다면, 인증서버인 `kauth.kakao.com`으로 리다이렉트하여 자동로그인 요청

자동로그인 동작 방식


1. 유저가 개별 유저블로그에 접근
2. 서버에서 자동로그인을 시켜야 하는 요청인지 판단
 - 자동로그인 기능을 켜둔 상태인지
 - 자동로그인 방지를 위한 쿠키가 없는지
 - 티스토리 세션이 없는지
 - 봇 요청이 아닌지 등등
3. 자동로그인을 시켜야 한다면, 인증서버인 `kauth.kakao.com`으로 리다이렉트하여 자동로그인 요청
4. 성공 여부에 따라 티스토리 세션을 만들고 1번에서 요청한 url으로 리다이렉트




 와쌌_ · 2023.07.21 09:19 · 질문/기타 조회수 179 · 댓글 2

와... 이젠 블로그 통계까지 엉망이네요 ^{hot}
정상적으로 검색포털 통해 유입된 페이지뷰를,
기타유입으로 잡아버리네요.

이제 광고 강제 끼워넣기에 이어
모든 티스토리 블로그를 저품질 블로그로 만드는게 목표인지...
힘빠집니다 참; 신고

 아빠약사김피 · 2023.07.21 09:28 · 신고
뭔가 달라지는거 같긴하네요. ...

 바로코 Barroco · 2023.07.22 03:23 · 신고
어제 어느 분께서 네이버 검색어 없이 유입 잡힌다고 하셨는데, 다음
도 마찬가지네요.
<http://search.daum.net/> 이렇게만 뜹니다. 검색어까지 뜨는건
간혹 가다 있고요.

내용을 입력하세요
하루에 10개까지 작성 가능합니다

0/500 완료

자동로그인이랑 검색어 유실이랑 무슨 상관일까?

원인 분석을 위한 삽질의 시작

테스트 상황 #1

step 0. 최초 상황

현재 유저가 아래의 페이지에 머무르고 있는 상황을 가정.

원래라면 유저블로그에 접근했을 때 이미 자동로그인 프로세스를 거치고 '자동로그인 방지 쿠키'가 만들어졌겠지만,

테스트를 위해 이 페이지에서 쿠키를 모두 지우고 진행.



테스트 상황 #1

step 1. 다른 게시글 클릭

121번 게시글에서 79번 게시글을 누르자, 정상적으로 자동로그인을 타는 모습.

Name	Status	Type
79	302	document / Redirect
authorize?client_id=4000dedcb7...	302	document / Redirect
redirect?error_description=user...	302	document / Redirect
79	200	document
api	200	document

테스트 상황 #1

step 1. 다른 게시물 클릭

121번 게시물에서 79번 게시물을 누르자, 정상적으로 자동로그인을 타는 모습.

The screenshot shows the 'Headers' tab in a web browser's developer tools. The left sidebar lists the page title '79' and the URL 'https://code.dev.tistory.com/79'. The main panel displays the following information:

- General:**
 - Request URL: `https://code.dev.tistory.com/79`
 - Request Method: `GET`
 - Status Code: `302`
 - Remote Address: `10.202.148.173:443`
 - Referrer Policy: `unsafe-url`
- Response Headers:**
 - Cache-Control: `no-cache, no-store, max-age=0, must-revalidate`
 - Content-Length: `0`
 - Date: `Wed, 26 Jul 2023 02:42:12 GMT`
 - Expires: `0`
 - Location: `https://sandbox-kauth.kakao.com/oauth/authorize?client_id=4000dedcb73719ace32d8bba342395e4&redirect_uri=https%3A%2F%2Fwww.dev.tistory.com/79`
 - Pragma: `no-cache`
 - Strict-Transport-Security: `max-age=31536000 ; includeSubDomains`
 - Vary: `Access-Control-Request-Headers`
 - Vary: `Access-Control-Request-Method`
 - Vary: `Origin`
 - X-Content-Type-Options: `nosniff`
 - X-Xss-Protection: `1; mode=block`
- Request Headers:**
 - Accept: `text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8`
 - Accept-Encoding: `gzip, deflate, br`
 - Accept-Language: `ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7`
 - Cache-Control: `no-cache`
 - Connection: `keep-alive`
 - Host: `code.dev.tistory.com`
 - Pragma: `no-cache`
 - Referer: `https://code.dev.tistory.com/121?q=123`**
 - Sec-Ch-Ua: `"Not/A Brand";v="88", "Google Chrome";v="115", "Chromium";v="115"`
 - Sec-Ch-Ua-Mobile: `?0`

테스트 상황 #1

step 2. kauth로 로그인 요청

kauth로 redirect 될 때 역시 마찬가지로 referer는 그대로 잘 넘어가는 중.

The screenshot displays the Headers tab in a web browser's developer tools. The 'Request Headers' section is expanded, showing the following details:

- Request Method:** GET
- Status Code:** 302 Found
- Remote Address:** 121.53.217.36:443
- Referrer Policy:** unsafe-url

The 'Response Headers' section is also expanded, showing the following details:

- Access-Control-Allow-Headers:** Authorization, KA, Origin, X-Requested-With, Content-Type, Accept
- Access-Control-Allow-Methods:** GET, POST, OPTIONS
- Access-Control-Allow-Origin:** *
- Cache-Control:** no-cache, no-store, max-age=0, must-revalidate
- Connection:** keep-alive
- Content-Length:** 0
- Date:** Wed, 26 Jul 2023 02:42:13 GMT
- Expires:** 0
- Kakao:** Talk
- Location:** https://www.dev.tistory.com/auth/kakao/redirect?error_description=user%20authentication%20required.&state=aHR0cHM6Ly9jb2RlLmRldi50aXN0b3J5LmNvbS83Ox8xVVVPRTE9HSU4=
- Pragma:** no-cache
- Referrer-Policy:** strict-origin-when-cross-origin
- X-Content-Type-Options:** nosniff
- X-Frame-Options:** ALLOW-FROM https://code.dev.tistory.com
- X-Xss-Protection:** 1; mode=block

The 'Request Headers' section is expanded, showing the following details:

- Accept:** text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
- Accept-Encoding:** gzip, deflate, br
- Accept-Language:** ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
- Cache-Control:** no-cache
- Connection:** keep-alive
- Cookie:** webid=76a8d9f8b00b497dbc61a20a23a93916; webid_ts=1690338992849
- Host:** sandbox-kauth.kakao.com
- Pragma:** no-cache
- Referer:** https://code.dev.tistory.com/121?q=123
- Sec-Ch-Ua:** "Not/A Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"
- Sec-Ch-Ua-Mobile:** ?0

테스트 상황 #1

step 3. 티스토리 TOP callback

이 부분이 문제의 핵심!

kauth에서 TOP으로 redirect 될 때,
갑자기 path와 query string 정보가
유실되는 것을 확인.

The screenshot displays the Headers tab of a web browser's developer tools. The left sidebar shows a list of network requests, with the selected request being a GET request to `https://www.dev.tistory.com/auth/kakao/redirect?error_description=user%20authentication%20required.&state=aHR0cHM6Ly9jb2RlMmRldi50aXNlD&error=login_required`. The main pane shows the following headers:

- Request Headers:**
 - `:authority:` www.dev.tistory.com
 - `:method:` GET
 - `:path:` /auth/kakao/redirect?error_description=user%20authentication%20required.&state=aHR0cHM6Ly9jb2RlMmRldi50aXNlD&error=login_required
 - `:scheme:` https
 - `Accept:` text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
 - `Accept-Encoding:` gzip, deflate, br
 - `Accept-Language:` ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
 - `Cache-Control:` no-cache
 - `Cookie:` __T__=1; TOP-XSRF-TOKEN=d44f7438-a300-4a02-b039-2e11983ad45b; __T__=1; _T_ANO=FW01wNTPHoj6WDX4eviNvZJT+IDVKERFXi0Fq4E7h/JCO+RqB55ZWvFF0qnFGk+Aq/qtovzpTBGLvM408a aAKz5cvlvKTC5XV13yCX7VYyW6glVVFzHbirDpGyoyUaoymwOsdjmbAZgqFP0jdq0ZB75WS1quqMzn9UANfWuRfT3+ubOFVyz98wg5MpZkc9SfVwWgkPcUaD+Z60dAy8tMyXTj94Hhwfgvg==
 - Pragma:** no-cache
 - Referer:** https://code.dev.tistory.com/
 - `Sec-CH-UA:` "Not(A)B, and", "00", "Google Chrome";v="115", "Chromium";v="115"
 - `Sec-Ch-UA-Mobile:` ?0

테스트 상황 #1

step 4. 이동하고자 했던 url로 최종 도착

79번 게시글로 이동하는 게 목적이었으니, 최종적으로 79번 게시글에 접근 완료. 하지만 step 3에서 이미 referer에 path와 query string 정보가 사라진 뒤이므로, 여기서도 referer에는 도메인 정보만 남음.

The screenshot shows the 'Headers' tab of a browser's developer tools. The 'Request Headers' section is expanded, and the 'Referer' header is highlighted with a red box. The 'Referer' value is 'https://code.dev.tistory.com/'. Other visible headers include 'Accept', 'Accept-Encoding', 'Accept-Language', 'Cache-Control', 'Connection', 'Cookie', 'Host', 'Pragma', 'Sec-Ch-Ua', and 'Sec-Ch-Ua-Mobile'. The 'Response Headers' section is also visible, showing 'Cache-Control', 'Content-Encoding', 'Content-Type', 'Date', 'Expires', 'Pragma', 'Set-Cookie', 'Strict-Transport-Security', 'T_userid', 'Transfer-Encoding', 'Vary', and 'X-Xss-Protection'.

Header Name	Value
Request URL:	https://code.dev.tistory.com/79
Request Method:	GET
Status Code:	200
Remote Address:	10.202.148.173:443
Referrer Policy:	unsafe-url
Cache-Control:	no-cache, no-store, max-age=0, must-revalidate
Content-Encoding:	gzip
Content-Type:	text/html; charset=UTF-8
Date:	Wed, 26 Jul 2023 02:42:13 GMT
Expires:	0
Pragma:	no-cache
Set-Cookie:	REACTION_GUEST=9ae39bab1d67fe4ebac075badc75bddecc1151562
Strict-Transport-Security:	max-age=31536000 ; includeSubDomains
T_userid:	8d85a98aeba2af3603f8b75006aa9d46be227a1d
Transfer-Encoding:	chunked
Vary:	Accept-Encoding
Vary:	Access-Control-Request-Headers
Vary:	Access-Control-Request-Method
Vary:	Origin
X-Content-Type-Options:	nosniff
X-Xss-Protection:	1; mode=block
Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cache-Control:	no-cache
Connection:	keep-alive
Cookie:	_T_ANO=FW01wNTPHoj6WDX4eviNvZJT+IDVKERFXi0Fq4E7h/JCO+RqB55ZWvFF0qnFGk+Aq/qtovzpTBaAKz5cvlvKTC5XV13yCX7VyyW6glvVFzHbirfDpGyoyUaoymwOsdTjmbAZgqFP0jdjq0ZBr75WS1quqMzn9L+ubOFvyz98wg5MpZkc9SFVlwWgkPcUaD+Z60dAy8tMyXTJ94Hhwfgvg==; TSAL=1
Host:	code.dev.tistory.com
Pragma:	no-cache
Referer:	https://code.dev.tistory.com/
Sec-Ch-Ua:	"Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"
Sec-Ch-Ua-Mobile:	?0

재현은 성공했고...
그럼 무엇이 문제인가?

브라우저의 보안정책?
kauth의 서버 설정?
redirect를 여러 번 거쳐서?

자동로그인을 적용한 브런치는
이런 문제가 없을까?

테스트 상황 #2

step 0. 최초 상황






query string이 referer에 잘 남는지 확인하기 위해 적당히 세팅.
아까와 마찬가지로 이 상태에서 마찬가지로 쿠키는 모두 삭제.



테스트 상황 #2

step 1. 브런치 홈으로 접근

정상적으로 자동로그인을 타는 것을 확인할 수 있음.

Name	Status	Type
 brunch.co.kr	302	document / Redirect
 kakao?url=https%3A%2F%2Fbrunch....	302	document / Redirect
 authorize?client_id=e0201caea90cafb...	302	document / Redirect
 kakao?error_description=NOT_CONN...	302	document / Redirect
 brunch.co.kr	200	document

테스트 상황 #2

step 1. 브런치 홈으로 접근

정상적으로 자동로그인을 타는 것을 확인할 수 있음.
referer에도 query string이 잘 남아있는 상태.

The screenshot displays the 'Headers' tab of a browser's developer tools. The 'Request Headers' section is highlighted with a red box, showing the following details:

- Referer: `https://brunch.co.kr/?q=123`

Other visible headers include:

- authority: `brunch.co.kr`
- method: `GET`
- path: `/`
- scheme: `https`
- Accept: `text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7`
- Accept-Encoding: `gzip, deflate, br`
- Accept-Language: `ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7,ja;q=0.6`
- Cache-Control: `no-cache`
- Dnt: `1`
- Pragma: `no-cache`
- Sec-Ch-Ua: `"Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"`
- Sec-Ch-Ua-Mobile: `?0`

테스트 상황 #2

step 2. 브런치의 auth api로 redirect

정상.

The screenshot shows the 'Headers' tab in a browser's developer tools. The request is a GET to `https://brunch.co.kr/auth/kakao?url=https%3A%2F%2Fbrunch.co.kr%2F&auto_login=true`. The response status is 302. The 'Response Headers' section shows a 'Location' header pointing to the authorization page. The 'Request Headers' section shows various headers, with the 'Referer' header highlighted in a red box, containing the URL `https://brunch.co.kr/?q=123`.

Name	Value
General	
Request URL:	https://brunch.co.kr/auth/kakao?url=https%3A%2F%2Fbrunch.co.kr%2F&auto_login=true
Request Method:	GET
Status Code:	302
Remote Address:	211.249.249.13:443
Referrer Policy:	unsafe-url
Response Headers	
Content-Length:	0
Date:	Wed, 26 Jul 2023 04:42:26 GMT
Location:	https://kauth.kakao.com/oauth/authorize?client_id=e0201caaa90cafb237e250f63a519b5&response_type=code&auto_login=true&redirect_uri=https%3A%2F%2Fbrunch.co.kr%2Fcallback%2Fauth%2Fkakao&scope=&state=aHR0cHM6Ly9icnVuY2guY28ua3lv&grant_type=authorization_code
Set-Cookie:	bid=""; Domain=.brunch.co.kr; Expires=Thu, 01-Jan-1970 00:00:10 GMT; Path=/; Secure
Strict-Transport-Security:	max-age=15724800; includeSubDomains
Request Headers	
:authority:	brunch.co.kr
:method:	GET
:path:	/auth/kakao?url=https%3A%2F%2Fbrunch.co.kr%2F&auto_login=true
:scheme:	https
Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7,ja;q=0.6
Cache-Control:	no-cache
Cookie:	b_s_a_l=1
Dnt:	1
Pragman:	no-cache
Referer:	https://brunch.co.kr/?q=123
Sec-Ch-Ua:	"Not A Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"
Sec-Ch-Ua-Mobile:	?0

테스트 상황 #2

step 3. kauth로 로그인 요청

여기까지도 정상.

과연 이 다음에는 그대로 남을까?

The screenshot displays the 'Headers' tab in a web browser's developer tools. The left pane shows a list of requests, with the third request, 'authorize?client_id=e0201caea...', selected. The right pane shows the details of this request:

- Request URL:** `https://kauth.kakao.com/oauth/authorize?client_id=e0201caea90cafb237e250f63a519b5&response_type=code&auto_login=true&direct_url=https%3A%2F%2Fbranch.co.kr%2Fcallback%2Fauth%2Fkakao&scope=&state=aHR0cHM6Ly9lcnuY2guY28ua3lv&_type=authorization_code`
- Request Method:** GET
- Status Code:** 302 Found
- Remote Address:** 27.0.237.15:443
- Referrer Policy:** unsafe-url

The 'Response Headers' section is expanded, showing various headers such as 'Access-Control-Allow-Headers', 'Access-Control-Allow-Methods', 'Cache-Control', 'Date', 'Expires', 'Kakao: Talk', 'Location', 'Pragma', 'Referrer-Policy', 'X-Content-Type-Options', 'X-Frame-Options', and 'X-Xss-Protection'. The 'Request Headers' section is also expanded, showing 'Accept', 'Accept-Encoding', 'Accept-Language', 'Cache-Control', 'Connection', 'Dnt', 'Host', and 'Referer'. The 'Referer' header is highlighted with a red box and has the value `https://branch.co.kr/?q=123`.

At the bottom of the developer tools, a status bar indicates '5 / 162 requests | 149 kB / 9.6 MB'.

테스트 상황 #2

step 4. 브런치 callback

마찬가지로 query string이
잘 남아있다!!

The screenshot displays the 'Headers' tab in a web browser's developer tools. The left sidebar shows a list of tabs, with the current page being 'brunch.co.kr'. The main area is divided into sections: 'General', 'Response Headers', and 'Request Headers'. The 'Request Headers' section is expanded, showing various headers. The 'Referer' header is highlighted with a red box, and its value is 'https://brunch.co.kr/?q=123'. Other headers include 'authority', 'method', 'path', 'scheme', 'Accept', 'Accept-Encoding', 'Accept-Language', 'Cache-Control', 'Cookie', 'Dnt', 'Pragma', and 'Sec-Ch-Ua-Mobile'.

Header Name	Value
Request URL:	https://brunch.co.kr/callback/auth/kakao?error_description=NOT_CONNECTED_USER&state=aHR0cHM6Ly9icmVudG91Y2guYm91bnR1eS5jb20=
Request Method:	GET
Status Code:	302
Remote Address:	211.249.249.13:443
Referrer Policy:	unsafe-url
Content-Language:	ko-KR
Content-Length:	0
Date:	Wed, 26 Jul 2023 04:42:26 GMT
Location:	https://brunch.co.kr/
Set-Cookie:	bid=""; Domain=.brunch.co.kr; Expires=Thu, 01-Jan-1970 00:00:10 GMT; Path=/; Secure
Strict-Transport-Security:	max-age=15724800; includeSubDomains
authority:	brunch.co.kr
method:	GET
path:	/callback/auth/kakao?error_description=NOT_CONNECTED_USER&state=aHR0cHM6Ly9icmVudG91Y2guYm91bnR1eS5jb20=
scheme:	https
Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7,ja;q=0.6
Cache-Control:	no-cache
Cookie:	b_s_a_l=1
Dnt:	1
Pragma:	no-cache
Referer:	https://brunch.co.kr/?q=123
Sec-Ch-Ua:	"Not A Brand";v="99", "Chromium";v="115", "Chrome";v="115", "Chromium";v="115"
Sec-Ch-Ua-Mobile:	?0

테스트 상황 #2

step 5. 이동하고자 했던 url로
최종 도착

결국 마지막까지 최초 referer
정보가 query string까지 유지하며
요청이 잘 보내졌음.

The screenshot shows the 'Headers' tab in a browser's developer tools. The left pane lists several URLs, with 'brunch.co.kr' selected. The right pane displays the following information:

- General:**
 - Request URL: https://brunch.co.kr/
 - Request Method: GET
 - Status Code: 200
 - Remote Address: 211.249.249.13:443
 - Referrer Policy: unsafe-url
- Response Headers:**
 - Content-Language: ko-KR
 - Content-Type: text/html;charset=utf-8
 - Date: Wed, 26 Jul 2023 04:42:26 GMT
 - Strict-Transport-Security: max-age=15724800; includeSubDomains
- Request Headers:**
 - :authority: brunch.co.kr
 - :method: GET
 - :path: /
 - :scheme: https
 - Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/svg+xml,*/*;q=0.8
 - Accept-Encoding: gzip, deflate, br
 - Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7,ja;q=0.6
 - Cache-Control: no-cache
 - Cookie: b_s_a_l=1
 - Dnt: 1
 - Referer: https://brunch.co.kr/?q=123
 - Sec-Ch-Ua: "Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"
 - Sec-Ch-Ua-Mobile: ?0

왜 브런치는 되고, 티스토리는 안 될까?



우선 눈에 띄는 차이점...

브런치는 다음과 같은 redirect를 가진다.

brunch → kauth → brunch

티스토리는 다음과 같은 redirect를 가진다.

userblog → kauth → TOP → userblog

우선 눈에 띄는 차이점...

브런치는 다음과 같은 redirect를 가진다.

brunch → kauth → brunch

티스토리는 다음과 같은 redirect를 가진다.

userblog → kauth → TOP → userblog

그렇다면, userblog → kauth → userblog로 redirect가 이루어지면 괜찮을까?

테스트 상황 #3

step 0. 최초 상황

로컬에서 빌드하였고 테스트를 진행(kauth에서 바로 userblog로 받으려면 새로운 test api가 필요).
마찬가지로 현재 사용자는 최초에 이 페이지에서 시작.



테스트 상황 #3

step 1. 다른 게시물 클릭

The screenshot shows the 'Headers' tab of a browser's developer tools. The left sidebar lists network requests, with '79' selected. The main pane displays the details for this request:

- General:**
 - Request URL: `https://code.yuni.dev.tistory.com/79`
 - Request Method: `GET`
 - Status Code: `302`
 - Remote Address: `127.0.0.1:443`
 - Referrer Policy: `unsafe-url`
- Response Headers (14):** (Collapsed)
- Request Headers:**
 - Raw
 - Accept: `text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp exchange;v=b3;q=0.7`
 - Accept-Encoding: `gzip, deflate, br`
 - Accept-Language: `ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7`
 - Cache-Control: `no-cache`
 - Connection: `keep-alive`
 - Host: `code.yuni.dev.tistory.com`
 - Pragma: `no-cache`
 - Referer: `https://code.yuni.dev.tistory.com/121?q=123`** (highlighted with a red box)
 - Sec-Ch-Ua: `"Not(A)Brand";v="99", "Google Chrome";v="111", "Chromium";v="115"`
 - Sec-Ch-Ua-Mobile: `?0`

테스트 상황 #3

step 2. kauth로 로그인 요청

The screenshot shows the Headers tab of a web browser's developer tools. The request is to `https://sandbox-kauth.kakao.com/oauth/authorize?client_id=4000dedcb73719ace32yuni.dev.tistory.com%2Ftest&response_type=code&prompt=none&state=aHR0cHM6E9MT0dJTg==`. The status code is 302 Found. The Referer header is highlighted with a red box and contains the URL `https://code.yuni.dev.tistory.com/121?q=123`. Other headers include Accept, Accept-Encoding, Accept-Language, Cache-Control, Connection, Host, and Sec-Ch-Ua-Mobile.

Name	Value
Request URL:	https://sandbox-kauth.kakao.com/oauth/authorize?client_id=4000dedcb73719ace32yuni.dev.tistory.com%2Ftest&response_type=code&prompt=none&state=aHR0cHM6E9MT0dJTg==
Request Method:	GET
Status Code:	302 Found
Remote Address:	121.53.217.36:443
Referrer Policy:	unsafe-url
Response Headers (15)	
Request Headers	<input type="checkbox"/> Raw
Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/exchange;v=b3;q=0.7
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cache-Control:	no-cache
Connection:	keep-alive
Host:	sandbox-kauth.kakao.com
Pragma:	no-cache
Referer:	https://code.yuni.dev.tistory.com/121?q=123
Sec-Ch-Ua:	"Not(A)Brand";v="99", "Google Chrome";v="116", "Chromium";v="115"
Sec-Ch-Ua-Mobile:	?0

테스트 상황 #3

step 3. /test callback

The image shows a browser's developer tools network tab. The left sidebar lists network requests, with the selected request being a GET to `https://code.yuni.dev.tistory.com/test?error_description=user%20authentication%20failed`. The main pane shows the 'Headers' tab for this request. The 'Request Headers' section is expanded, and the 'Referer' header is highlighted with a red box. The 'Referer' value is `https://code.yuni.dev.tistory.com/121?q=123`. Other visible headers include 'Accept', 'Accept-Encoding', 'Accept-Language', 'Cache-Control', 'Connection', 'Host', 'Pragma', 'Sec-Ch-Ua', and 'Sec-Ch-Ua-Mobile'.

Name	Value
Request URL:	https://code.yuni.dev.tistory.com/test?error_description=user%20authentication%20failed
Request Method:	GET
Status Code:	302
Remote Address:	127.0.0.1:443
Referrer Policy:	unsafe-url
Request Headers:	
Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/svg+xml,*/*;q=0.8
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cache-Control:	no-cache
Connection:	keep-alive
Host:	code.yuni.dev.tistory.com
Pragma:	no-cache
Referer:	https://code.yuni.dev.tistory.com/121?q=123
Sec-Ch-Ua:	"Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"
Sec-Ch-Ua-Mobile:	?0

테스트 상황 #3

step 4. 이동하고자 했던 url로 최종 도착

The screenshot shows the 'Headers' tab in a browser's developer tools. The left sidebar lists several headers, with the second one selected. The main panel displays the details for this header, including the Request URL, Request Method, Status Code, Remote Address, and Referrer Policy. Below these are sections for Response Headers (16) and Request Headers. The 'Request Headers' section is expanded, showing various headers such as Accept, Accept-Encoding, Accept-Language, Cache-Control, Connection, Cookie, Host, and Referer. The 'Referer' header is highlighted with a red box, showing its value as 'https://code.yuni.dev.tistory.com/121?q=123'. Other headers include 'Sec-Ch-Ua' and 'Sec-Ch-Ua-Mobile'.

Name	Value
79	authorize?client_id=4000d...
79	test?error_description=us...
79	79
api	api

General	
Request URL:	https://code.yuni.dev.tistory.com/79
Request Method:	GET
Status Code:	200
Remote Address:	127.0.0.1:443
Referrer Policy:	unsafe-url

Response Headers (16)	
Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/exchange;v=b3;q=0.7
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cache-Control:	no-cache
Connection:	keep-alive
Cookie:	TSAL=1
Host:	code.yuni.dev.tistory.com
Pragmat:	no-cache
Referer:	https://code.yuni.dev.tistory.com/121?q=123
Sec-Ch-Ua:	"Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"
Sec-Ch-Ua-Mobile:	?0

테스트 상황 #3

step 4. 이동하고자 했던 url로 최종 도착

와!! path와 query string 정보가 남았습니다!!

Name	Value
Request URL:	https://code.yuni.dev.tistory.com/79
Request Method:	GET
Status Code:	200
Remote Address:	127.0.0.1
Accept:	application/xml;q=0.9,image/avif,image/webp,image/svg+xml;q=0.8,en;q=0.7
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR;q=0.9,en;q=0.7
Cache-Control:	no-cache
Connection:	keep-alive
Cookie:	TSAL=1
Host:	code.yuni.dev.tistory.com
Pragma:	no-cache
Referer:	https://code.yuni.dev.tistory.com/121?q=123
Sec-Ch-Ua:	"Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"
Sec-Ch-Ua-Mobile:	?0

성공은 했지만, 사실 이걸 반쪽짜리 테스트

실제 검색유입에서 들어오는 요청의 경우는
userblog에서 userblog로 요청하는 게 아니라
daum이나 naver 등 **다른 도메인에서 들어오는 요청**이기 때문

테스트 상황 #4

step 0. 최초 상황

이번에는 dev 도메인 121번 게시글에서, local 도메인 79번 게시글로 요청하는 상황으로 테스트.



테스트 상황 #4

step 1. local로 향하는 url 클릭

The image shows a screenshot of a web browser's developer tools, specifically the 'Headers' tab. The left sidebar shows a list of requests, with the first one selected: '121?w=321'. The main panel displays the details for this request. The 'General' section shows the Request URL as 'https://code.yuni.dev.tistory.com/121?w=321', the Request Method as 'GET', the Status Code as '302', the Remote Address as '127.0.0.1:443', and the Referrer Policy as 'unsafe-url'. The 'Response Headers (14)' section is collapsed. The 'Request Headers' section is expanded, showing various headers. The 'Referer' header is highlighted with a red box and has the value 'https://code.dev.tistory.com/121?q=123'. Other visible headers include 'Accept', 'Accept-Encoding', 'Accept-Language', 'Cache-Control', 'Connection', 'Host', 'Pragma', 'Sec-Ch-Ua', and 'Sec-Ch-Ua-Mobile'.

Name	Value
Request URL:	https://code.yuni.dev.tistory.com/121?w=321
Request Method:	GET
Status Code:	302
Remote Address:	127.0.0.1:443
Referrer Policy:	unsafe-url
Response Headers (14)	
Request Headers	<input type="checkbox"/> Raw
Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cache-Control:	no-cache
Connection:	keep-alive
Host:	code.yuni.dev.tistory.com
Pragma:	no-cache
Referer:	https://code.dev.tistory.com/121?q=123
Sec-Ch-Ua:	"Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"
Sec-Ch-Ua-Mobile:	?0

테스트 상황 #4

step 2. kauth로 로그인 요청

The screenshot shows the Headers tab in a web browser's developer tools. The left sidebar lists several requests, with 'authorize?client_id=4000de...' selected. The main panel displays the following information:

- General**
 - Request URL: `https://sandbox-kauth.kakao.com/oauth/authorize?client_id=4000dedcb73719ace3tory.com%2Ftest&response_type=code&prompt=none&state=aHR0cHM6Ly9jb2RlSU4=`
 - Request Method: GET
 - Status Code: 302 Found
 - Remote Address: 121.53.217.36:443
 - Referrer Policy: unsafe-url
- Response Headers (15)**
- Request Headers** (with a 'Raw' checkbox)
 - Accept: `text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,imag`
 - Accept-Encoding: `gzip, deflate, br`
 - Accept-Language: `ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7`
 - Cache-Control: `no-cache`
 - Connection: `keep-alive`
 - Host: `sandbox-kauth.kakao.com`
 - Pragma: `no-cache`
 - Referer: `https://code.dev.tistory.com/121?q=123`** (highlighted with a red box)
 - Sec-Ch-Ua: `"Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"`
 - Sec-Ch-Ua-Mobile: `?0`

테스트 상황 #4

step 3. /test callback

The screenshot displays the 'Headers' tab in a web browser's developer tools. The left sidebar shows a list of requests, with the third request, 'test?error_description=user...', selected. The main panel shows the details for this request, including the Request URL, Request Method (GET), Status Code (302), Remote Address (127.0.0.1:443), and Referrer Policy (unsafe-url). The 'Request Headers' section is expanded, showing various headers such as Accept, Accept-Encoding, Accept-Language, Cache-Control, Connection, Cookie, Host, and Referer. The 'Referer' header is highlighted with a red box, and its value is 'https://code.dev.tistory.com/'.

Name	Value
Request URL:	https://code.yuni.dev.tistory.com/test?error_description=user%20authentication%20re... uY29tLzEyMT93PTMyMXxBVVRPTE9HSU4%3D&error=login_required
Request Method:	GET
Status Code:	302
Remote Address:	127.0.0.1:443
Referrer Policy:	unsafe-url
Response Headers (15)	
Request Headers	<input type="checkbox"/> Raw
Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/a...
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cache-Control:	no-cache
Connection:	keep-alive
Cookie:	_T_ANO=X6BWUZRc6z/3ewNkklF0ciHE3BqlafzQPrdhizI/s9zrqkdMFdZNO+WerS9lc... 8s5BAojSOKk2iMhOacHxl2+xUSv7eqWbZvavygjjp1nM9GZxw94+N6ukfKZsv2PMmc3... P63QAFveTpwV0o2K+EU7v3879M1a5KSTeUfEd9addwNi5HP8qR5Jx6eWLQ4I2zhww...
Host:	code.yuni.dev.tistory.com
Pragmat:	no-cache
Referer:	https://code.dev.tistory.com/
Sec-Ch-Ua:	"Not/A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"
Sec-Ch-Ua-Mobile:	?0

테스트 상황 #4

step 3. /test callback



Name	Headers	Payload	Preview	Response	Initiator	Timing	Cookies
121?w=321	General						
authorize?client_id=4000de...	Request URL:			https://code.yuni.dev.tistory.com/test?error_description=user%20authentication%20re...			
test?error_description=user...	Request Method:			GET			
121?w=321	Status Code:			302			
api	Remote Address:			127.0.0.1:443			
	Referrer Policy:			unsafe-url			
	Response Headers (15)						
	Request Headers	<input type="checkbox"/> Raw					
	Accept:			text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/a...			
	Accept-Encoding:			gzip, deflate, br			
	Accept-Language:			ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7			
	Cache-Control:			no-cache			
	Connection:			keep-alive			
	Cookie:			_T_ANO=X6BWUZRc6z/3ewNkklF0ciHE3BqlafzQPPrdhiZl/s9zrqkdMFdZNO+WerS9lc...			
	Host:			code.yuni.dev.tistory.com			
	Pragma:			no-cache			
	Referer:			https://code.dev.tistory.com/			
	Sec-Ch-Ua:			"Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"			
	Sec-Ch-Ua-Mobile:			?0			

테스트 상황 #4

step 4. 이동하고자 했던 url로 최종 도착

The screenshot displays the Headers tab in a web browser's developer tools. The left sidebar shows a list of requests, with the selected request being a GET request to `https://code.yuni.dev.tistory.com/121?w=321`. The main panel shows the following details:

- General:**
 - Request URL: `https://code.yuni.dev.tistory.com/121?w=321`
 - Request Method: `GET`
 - Status Code: `200`
 - Remote Address: `127.0.0.1:443`
 - Referrer Policy: `unsafe-url`
- Response Headers (16):**
- Request Headers:** (Raw view is selected)
 - Accept: `text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/svg+xml,*/*;q=0.8`
 - Accept-Encoding: `gzip, deflate, br`
 - Accept-Language: `ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7`
 - Cache-Control: `no-cache`
 - Connection: `keep-alive`
 - Cookie: `_T_ANO=X6BWUZRc6z/3ewNkklF0ciHE3BqlafzQUPrdhiZl/s9zrqkdMFdZNO+Wer8s5BAojS0kk2iMhOacHlx2+xUSv7eqWbZvavygjjp1nM9GZxw94+N6ukfKZsv2PM63QAFveTpwV0o2K+EU7v3879M1a5KSTeUfEd9addwNi5HP8qR5Jx6eWLQ4lZzTSAL=1`
 - Host: `code.yuni.dev.tistory.com`
 - Pragma: `no-cache`
 - Referer: `https://code.dev.tistory.com/`** (highlighted with a red box)
 - Sec-Ch-Ua: `"Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"`
 - Sec-Ch-Ua-Mobile: `?0`

왜??? 이번엔 안 됐을까?

그 이유는 바로 Referrer-Policy

Referrer-Policy

The `Referrer-Policy` [HTTP header](#) controls how much [referrer information](#) (sent with the `Referer` header) should be included with requests. Aside from the HTTP header, you can [set this policy in HTML](#).

Header type	Response header
Forbidden header name	no

Syntax

HTTP



```
Referrer-Policy: no-referrer
Referrer-Policy: no-referrer-when-downgrade
Referrer-Policy: origin
Referrer-Policy: origin-when-cross-origin
Referrer-Policy: same-origin
Referrer-Policy: strict-origin
Referrer-Policy: strict-origin-when-cross-origin
Referrer-Policy: unsafe-url
```


private info	#1 https://music.example/superArtist123
	#2 https://news.example/fr/search/?q=science&sort=latest
private + sensitive	#3 https://health-blog.example/path/sports-injuries/knee
private + identifying	#4 https://social-network.example/my-account/johndoe86
	#5 https://social-network.example/email_verified/ ?email=johndoe86@gmail.com
security- critical	#6 https://cloud-storage.example/625x1710s7Gtsr/password_reset

Referrer Policy는 어떻게 적용할까?

- HTTP Header
- HTML
- Javascript

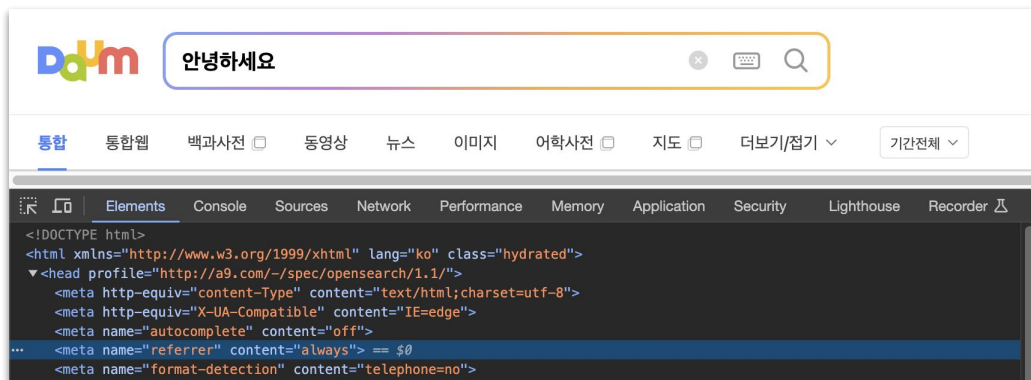
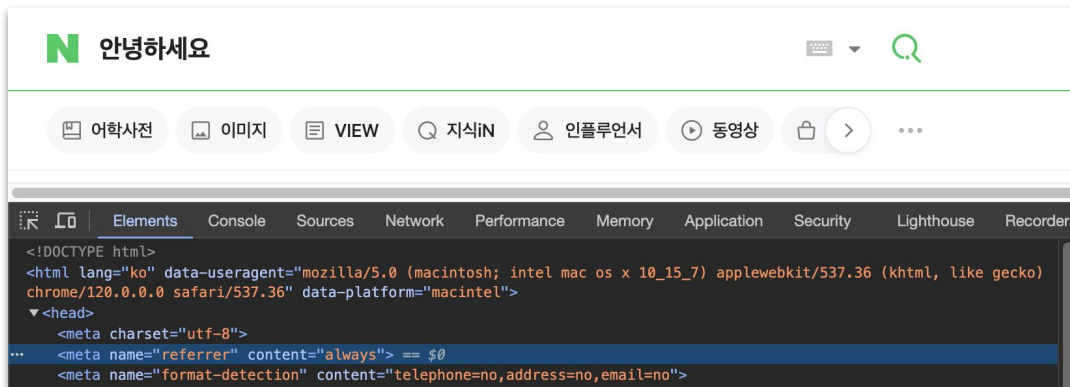
Referrer Policy는 어떻게 적용할까?

- HTTP Header
- HTML
- Javascript

Name	Headers	Preview	Response	Initiator	Timing	Cookies
dsyun96	▼ General					
PostList.naver?blogId=dsyun96...	Request URL:		https://blog.naver.com/dsyun96			
ExternalWidgetRender.naver?bl...	Request Method:		GET			
	Status Code:		200 OK			
	Remote Address:		223.130.192.187:443			
	Referrer Policy:		unsafe-url			
	▼ Response Headers					
	Alt-Svc:		h3=":443"; ma=86400			
	Cache-Control:		no-cache			
	Content-Encoding:		gzip			
	Content-Type:		text/html;charset=UTF-8			
	Date:		Sun, 17 Dec 2023 08:28:55 GMT			
	Expires:		Thu, 01 Jan 1970 00:00:00 GMT			
	Referrer-Policy:		unsafe-url			
	Server:		nfront			
	Set-Cookie:		JSESSIONID=EFDB47010F9EFC2448F142D167A2CF76.jvm1; Path=/; Secure; HttpOnly			
	Vary:		Accept-Encoding			

Referrer Policy는 어떻게 적용할까?

- HTTP Header
- HTML
- Javascript



Referrer Policy는 어떻게 적용할까?

- HTTP Header
- HTML
- Javascript

E.g. `Referer: https://javascript.info/admin/secret/paths`.

If we'd like other websites know only the origin part, not the URL-path, we can set the option:

```
fetch('https://another.com/page', {  
  // ...  
  referrerPolicy: "origin-when-cross-origin" // Referer: https://javascript.info  
});
```

Referrer Policy 적용 우선순위

1. element 레벨 (e.g. ``)
2. page 레벨 (HTTP Header, meta tag)
3. 브라우저 기본값

▼ General

Request URL: https://sandbox-kauth.kakao.com/oauth/authorize?client_id=4000dedcb73719ace32d8bba342395e4&redirect_uri=yuni.dev.tistory.com%2Ftest&response_type=code&prompt=none&state=aHR0cHM6Ly9jb2RlLn11bmkuzGV2LnRpc3F1dJTG%3D%3D&error=login_required

Request Method: GET

Status Code: ● 302 Found

Remote Address: 121.53.217.36:443

Referrer Policy: unsafe-url

▼ Response Headers

Raw

Access-Control-Allow-Headers: Authorization, KA, Origin, X-Requested-With, Content-Type, Accept

Access-Control-Allow-Methods: GET, POST, OPTIONS

Access-Control-Allow-Origin: *

Cache-Control: no-cache, no-store, max-age=0, must-revalidate

Connection: keep-alive

Content-Length: 0

Date: Wed, 26 Jul 2023 06:07:54 GMT

Expires: 0

Kakao: Talk

Location: https://code.yuni.dev.tistory.com/test?error_description=user%20authentication%20required.&state=aHR0cHM6Ly9jb2RlLn11bmkuzGV2LnRpc3F1dJTG%3D%3D&error=login_required

Pragma: no-cache

Referrer-Policy: strict-origin-when-cross-origin

X-Content-Type-Options: nosniff

X-Frame-Options: ALLOW-FROM https://code.yuni.dev.tistory.com

X-Xss-Protection: 1; mode=block

	No Data	Origin Only	Full URL
<code>`no-referrer`</code>	✓		
<code>`origin`</code>		✓	
<code>`unsafe-url`</code>			✓
<code>`strict-origin`</code>	HTTPS → HTTP	HTTPS → HTTPS, HTTP → HTTP	
<code>`no-referrer-when-downgrade`</code>	HTTPS → HTTP		HTTPS → HTTPS, HTTP → HTTP
<code>`origin-when-cross-origin`</code>		<code>`cross-origin`</code>	<code>`same-origin`</code>
<code>`same-origin`</code>	<code>`cross-origin`</code>		<code>`same-origin`</code>
<code>`strict-origin-when-cross-origin`</code>	HTTPS → HTTP	<code>`cross-origin`</code> , HTTPS → HTTPS, HTTP → HTTP	

cross-origin이란?

origin은 scheme(protocol), hostname, port의 조합을 의미

- scheme : https://
- hostname : www.tistory.com
- port : 443

<https://yceffort.kr:443> 를 기준으로 비교했을 때,

Origin	비교결과	이유
https://fake.kr:443	<code>`cross-origin`</code>	도메인이 다르다.
https://www.yceffort.kr:443	<code>`cross-origin`</code>	서브도메인이 다르다.
https://blog.yceffort.kr:443	<code>`cross-origin`</code>	서브도메인이 다르다.
http://yceffort.kr:443	<code>`cross-origin`</code>	scheme이 다르다.
http://yceffort.kr:80	<code>`cross-origin`</code>	port가 다르다.
https://yceffort.kr:443	<code>`same-origin`</code>	완전히 같다.
https://yceffort.kr	<code>`same-origin`</code>	포트가 없지만, https의 기본포트 443이 있다고 간주한다.

필요한 정보는 모두 모였으니
파헤쳐 봅시다!

브런치에서 referer의 query string이 잘 남았던 이유

brunch → kauth → brunch

브런치에서 referer의 query string이 잘 남았던 이유

brunch → kauth → brunch

kauth에 요청할 때 brunch.kr 이라는 referer 정보와,
kauth가 다시 redirect로 요청을 보내야 하는 brunch.kr 이 `same-origin` 이라서,
full url을 제공받을 수 있다.

티스토리에서 referer의 query string이 유실됐던 이유

userblog → kauth → TOP → userblog

티스토리에서 referer의 query string이 유실됐던 이유

userblog → kauth → TOP → userblog

kauth에 요청할 때 yun.tistory.com 이라는 referer 정보와,
kauth가 다시 redirect로 요청을 보내야 하는 www.tistory.com 이 **cross-origin** 이라서,
origin 정보만 제공받을 수 있다.

티스토리에서 referer의 query string이 유실됐던 이유

userblog → kauth → TOP → userblog

kauth에 요청할 때 yun.tistory.com 이라는 referer 정보와,
kauth가 다시 redirect로 요청을 보내야 하는 www.tistory.com 이 **cross-origin** 이라서,
origin 정보만 제공받을 수 있다.

userblog → kauth → userblog

이 경우는 **same-origin** 이므로, full url을 제공받을 수 있다.

하지만 실제 플로우는

브런치의 경우 :

daum/naver → brunch → kauth → brunch

티스토리의 경우 :

daum/naver → userblog → kauth → TOP → userblog

하지만 실제 플로우는

브런치의 경우 :

daum/naver → brunch → kauth → brunch

티스토리의 경우 :

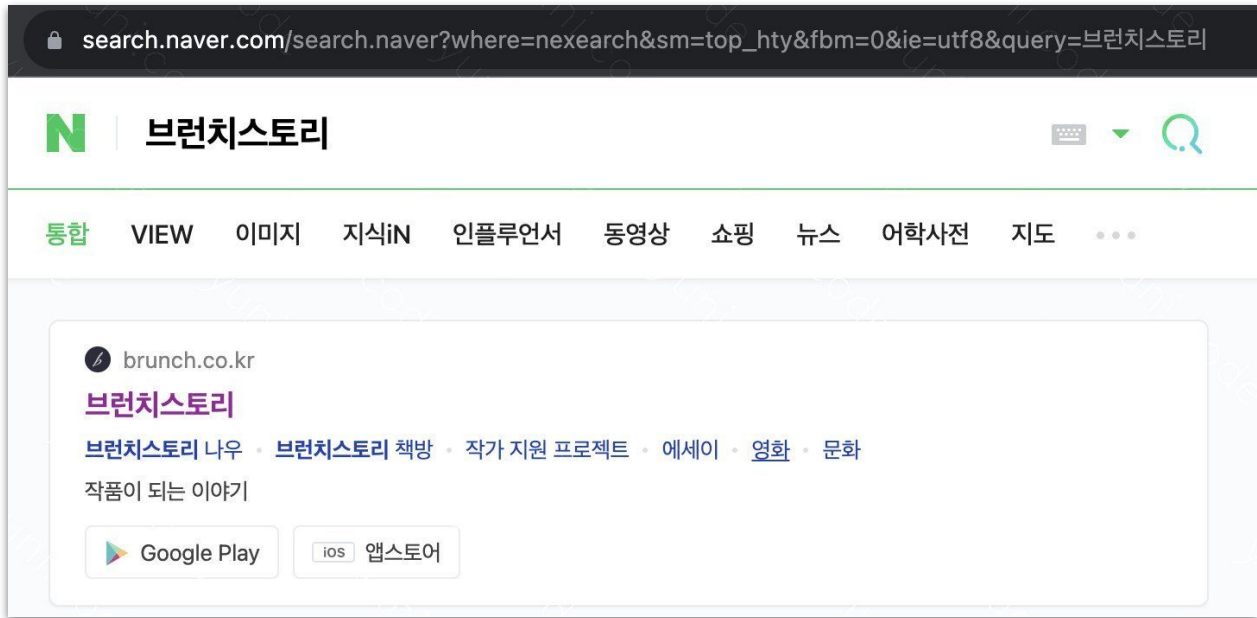
daum/naver → userblog → kauth → TOP → userblog

kauth에 들어오는 요청의 referer 정보는 daum/naver 이므로,
무조건 **cross-origin** 이 된다.

테스트... 해 봐야겠지?

테스트 상황 #5

step 0. 최초 상황



테스트 상황 #5

step 1. 브런치스토리로 접근

Name	Headers	Preview	Response	Initiator	Timing	Cookies
rd?m=1&px=209&py=187...	General					
brunch.co.kr	Request URL:	https://brunch.co.kr/				
kakao?url=https%3A%2F...	Request Method:	GET				
authorize?client_id=e0201...	Status Code:	302				
kakao?error_description=...	Remote Address:	211.249.249.13:443				
brunch.co.kr	Referrer Policy:	unsafe-url				
	Response Headers (6)					
	Request Headers					
	:authority:	brunch.co.kr				
	:method:	GET				
	:path:	/				
	:scheme:	https				
	Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7				
	Accept-Encoding:	gzip, deflate, br				
	Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7				
	Cache-Control:	no-cache				
	Pragma:	no-cache				
	Referer:	https://search.naver.com/search.naver?where=nexearch&sm=top_h ty&fbm=0&ie=utf8&query=%EB%B8%8C%EB%9F%B0%EC%B9%98%EC%8A%A4%ED%86%A0%EB%A6%AC				
	Sec-Ch-Ua:	"Not(A)Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"				
	Sec-Ch-Ua-Mobile:	?0				

테스트 상황 #5

step 3. kauth로 로그인 요청

The screenshot displays the 'Headers' tab in a web browser's developer tools. The left sidebar shows a list of requests, with the selected request being 'authorize?client_id=e0201...'. The main panel shows the following details:

- General:**
 - Request URL: `https://kauth.kakao.com/oauth/authorize?client_id=e0201caea90cafb237e250f63a519b5&response_type=code&auto_login=true&redirect_uri=https%3A%h.co.kr%2Fcallback%2Fauth%2Fkakao&scope=&state=aHR0cHM6Ly9icnVuY2guY28ua3lv&grant_type=authorization_code`
 - Request Method: GET
 - Status Code: 302 Found
 - Remote Address: 203.133.166.32:443
 - Referrer Policy: unsafe-url
- Response Headers (15):** (Collapsed)
- Request Headers:** (Expanded)
 - Accept: `text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7`
 - Accept-Encoding: `gzip, deflate, br`
 - Accept-Language: `ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7`
 - Cache-Control: `no-cache`
 - Connection: `keep-alive`
 - Host: `kauth.kakao.com`
 - Pragma: `no-cache`
 - Referer:** `https://search.naver.com/search.naver?where=nexearch&sm=top_h ty&fbm=0&ie=utf8&query=%EB%B8%8C%EB%9F%B0%EC%B9%98%EC%8A%A4%ED%86%A0%EB%A6%AC` (Highlighted with a red box)
 - Sec-Ch-Ua: `"NotA)Brand";v="99";"Google Chrome";v="115";"Chromium";v="115"`
 - Sec-Ch-Ua-Mobile: `?0`

테스트 상황 #5

step 4. 브런치스토리 callback

Name	Headers	Payload	Preview	Response	Initiator	Timing	Cookies
rd?m=1&px=209&py=187...	▼ General						
brunch.co.kr	Request URL:	https://brunch.co.kr/callback/auth/kakao?error_description=NOT_CONNECTED_USER&state=aHR0cH					
kakao?url=https%3A%2F...	Request Method:	GET					
authorize?client_id=e0201...	Status Code:	302					
kakao?error_description=...	Remote Address:	211.249.249.13:443					
brunch.co.kr	Referrer Policy:	unsafe-url					
	▶ Response Headers (6)						
	▼ Request Headers						
	:authority:	brunch.co.kr					
	:method:	GET					
	:path:	/callback/auth/kakao?error_description=NOT_CONNECTED_USER&state=aHR0cHM6Ly9icnVuY2guY2					
	:scheme:	https					
	Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,ap					
	Accept-Encoding:	gzip, deflate, br					
	Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7					
	Cache-Control:	no-cache					
	Cookie:	b_s_a_l=1					
	Pragma:	no-cache					
	Referer:	https://search.naver.com/					
	Sec-Ch-Ua:	"Not/A Brand";v="99", "Google Chrome";v="115", "Chromium";v="115"					
	Sec-Ch-Ua-Mobile:	?0					

테스트 상황 #5

step 5. 이동하고자 했던 url로 최종 도착

Name	Value
rd?m=1&px=209&py=187...	
brunch.co.kr	
kakao?url=https%3A%2F...	
authorize?client_id=e0201...	
kakao?error_description=...	
brunch.co.kr	
▼ General	
Request URL:	https://brunch.co.kr/
Request Method:	GET
Status Code:	200
Remote Address:	211.249.249.13:443
Referrer Policy:	unsafe-url
▶ Response Headers (4)	
▼ Request Headers	
:authority:	brunch.co.kr
:method:	GET
:path:	/
:scheme:	https
Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/...
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cache-Control:	no-cache
Cookie:	b_s_a_l=1
Referer:	https://search.naver.com/
Sec-Ch-Ua:	"NotABrand";v="99", "Google Chrome";v="115", "Chromium";v="115"
Sec-Ch-Ua-Mobile:	?0

증명 완료

그럼 이걸 어떻게 해결해야 하는가?

그럼 이걸 어떻게 해결해야 하는가?

1. kauth 담당 부서에 Referrer-Policy 변경 요청하기

그럼 이것을 어떻게 해결해야 하는가?

1. kauth 담당 부서에 Referrer-Policy 변경 요청하기 → 어떤 사이드이펙트가 생길지 모른다

그럼 이것을 어떻게 해결해야 하는가?

1. kauth 담당 부서에 Referrer-Policy 변경 요청하기 → 어떤 사이드이펙트가 생길지 모른다
2. kauth로 자동로그인 요청을 보낼 때 redirect uri에 query string으로 세팅하고,
TOP에서 받아서 이를 다시 referer 정보에 세팅하기

그럼 이걸 어떻게 해결해야 하는가?

1. kauth 담당 부서에 Referrer-Policy 변경 요청하기 → 어떤 사이드이펙트가 생길지 모른다
2. kauth로 자동로그인 요청을 보낼 때 redirect uri에 query string으로 세팅하고,
TOP에서 받아서 이를 다시 referer 정보에 세팅하기 → 가능할 것으로 보임 (PoC 필요, 아직 안 해봄)

E.O.D.

참고 자료

<https://yceffort.kr/2020/09/referer-and-referrer-policy>