BIT BY BIT

2025/01/12 (월)

Session 6 – 로그인, http, 네트워크장치

이동인

타임라인

10분 – 오프닝

80분(40분/10분/30분) – 공유세션

30분 – 마무리

마무리에서 할 것

- 다음 세션 담당자 정하기
- 다음 강의 섹션 정하기
- 네트워크 총정리 문제 담당자 정하기

오프닝

와 강의 양 적다 !

설명이 불친절하다 !

외우기가 어렵다 !

공유 세션

안외워지는 건 다같이 외우기도 하고,
이해가 안간 건 이해도 하면서 진행해봅시다.

로그인 방식

세션 기반 인증 방식 vs 토큰 기반 인증 방식

정의만 가볍게 말해줄 사람?

로 방식

세션 기반 인 방식 기반 인증 방식

장 람?

로그인 방식

세션 기반 인증 방식 vs 토큰 기반 인증 방식

	세션 기반 인증	JWT 기반 인증
어디에 저장?	서버 메모리 (RAM)	사용자 브라우저 (쿠키 또는 로컬스토리지)
인증 방식	세션 ID를 서버에서 관리	토큰을 클라이언트가 가지고 있음
서버 부담	로그인할수록 서버 부담 증가	서버가 로그인 정보 저장 안 해서 부담 적음
확장성	다중 서버 환경에서 불편함	여러 서버에서 사용 가능 (분산 시스템에 유리)
보안성	유출 위험 낮음 (서버에서 관리)	유출되면 위험! (토큰 탈취 가능)
로그아웃 처리	서버에서 세션 삭제 가능	만료될 때까지 유효 (강제 로그아웃 어려움)

세션기반 인증방식

용어 및 개념 정리 (선점하십쇼)

http 특징

세션 정의

세션 ID란?

세션기반 인증방식

클라이언트

내 아이디랑 비밀번호야 로그인해줘

서버

흠 로그인 성공적이네,
Session ID 만들어주고 나도 하나 갖고있어야지

자 여기 세션 id야 갖고있으렴

클라이언트

브라우저에 쿠키 저장할게 !! 고마워

세션기반 인증방식

클라이언트

내 아이디랑 비밀번호야 로그인해줘

서버

흠 로그인 성공적이네,
Session ID 만들어주고 나도 하나 갖고있어야지

자 여기 세션 id야 갖고있으렴

클라이언트

브라우저에 쿠키 저장할게 !! 고마워

과연 서버는 session id를 어디에 저장했을까?

세션기반 인증방식

클라이언트

내 아이디랑 비밀번호야 로그인해줘

서버

흠 로그인 성공적이네,
Session ID 만들어주고 나도 하나 갖고있어야지

자 여기 세션 id야 갖고있으렴

클라이언트

브라우저에 쿠키 저장할게 !! 고마워

과연 서버는 session id를 어디에 저장했을까?

1. db(데이터베이스)
2. was(웹 서버)

세션기반 인증방식

클라이언트

내 아이디랑 비밀번호야 로그인해줘

서버

흠 로그인 성공적이네,
Session ID 만들어주고 나도 하나 갖고있어야지

자 여기 세션 id야 갖고있으렴

클라이언트

브라우저에 쿠키 저장할게 !! 고마워

과연 서버는 session id를 어디에 저장했을까?

1. db(데이터베이스)
2. was(웹 서버)

데이터베이스 : ssd처럼 비휘발성 저장장치에 저장
웹서버 : 메모리에 저장

클라이언트

내 아이디랑 비밀번호야 로그인해줘

서버

흠 로그인 성공적이네,
Session ID 만들어주고 나도 하나 갖고있어야지

자 여기 세션 id야 갖고있으렴

클라이언트

브라우저에 쿠키 저장할게 !! 고마워

과연 서버는 session id를 어디에 저장했을까?

1. db(데이터베이스)
2. was(웹 서버)

따라서 데이터베이스에 session id를 저장 조회하는 데에 비용이 듦
=> 직렬화, 역직렬화에 비용이 든다의 뜻

클라이언트

내 아이디랑 비밀번호야 로그인해줘

서버

흠 로그인 성공적이네,
Session ID 만들어주고 나도 하나 갖고있어야지

자 여기 세션 id야 갖고있으렴

클라이언트

브라우저에 쿠키 저장할게 !! 고마워

과연 서버는 session id를 어디에 저장했을까?

1. db(데이터베이스)
2. was(웹 서버)

토큰 기반 인증 방식

JWT 토큰에 대한 간단한 개념 질문 세 개 – 선점하시오

약어

구성요소

토큰 종류 두 개

토큰 기반 인증 방식

JWT 토큰에 대한 간단한 개념 질문 세 개 – 선점하시오

JSON Web Token

구성요소

토큰 종류 두 개

토큰 기반 인증 방식

JWT 토큰에 대한 간단한 개념 질문 세 개 – 선점하시오

JSON Web Token (JSON 형식으로 주니까 ~)

Header, payload, signature (각각이 뭔지는 뒤에서 ~)

토큰 종류 두 개

토큰 기반 인증 방식

JWT 토큰에 대한 간단한 개념 질문 세 개 – 선점하시오

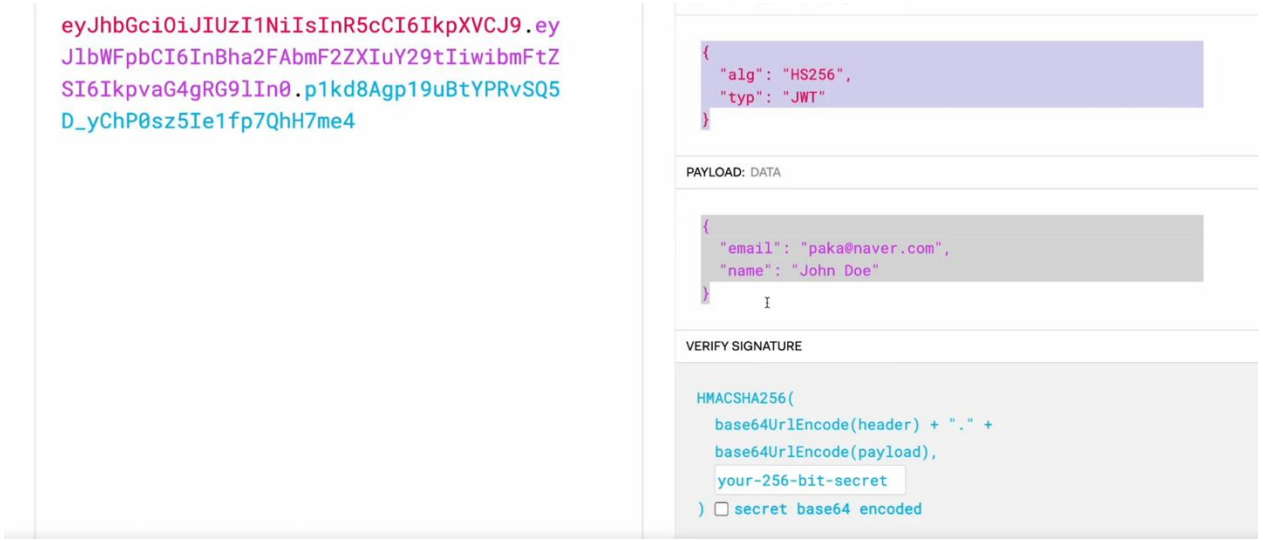
JSON Web Token (JSON 형식으로 주니까 ~)

Header, payload, signature (각각이 뭔지는 뒤에서 ~)

Access 토큰, refresh 토큰 (자세한 인증로직도 뒤에서 ~)

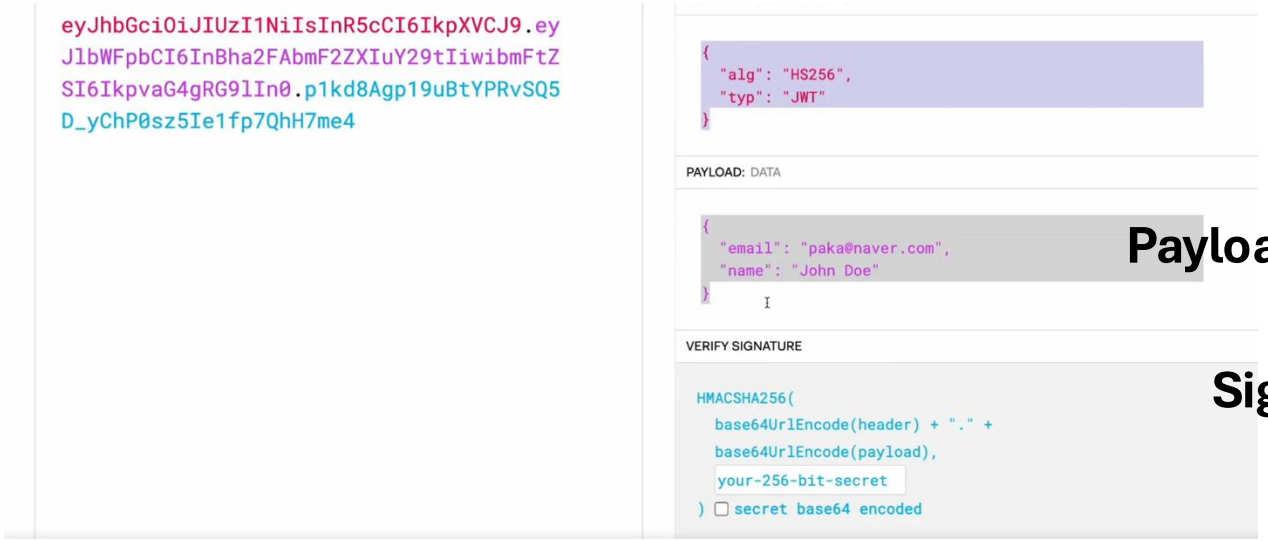
토큰 기반 인증 방식

Header, Payload, Signature



토큰 기반 인증 방식

Header, Payload, Signature



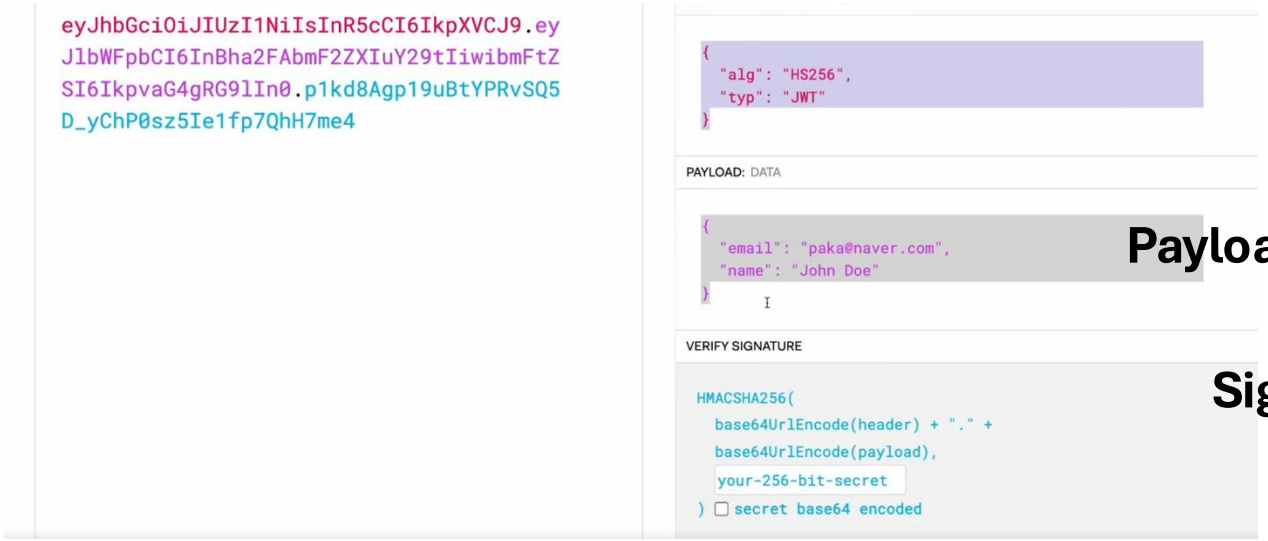
Header – 토큰 유형과, 서명 알고리즘

Payload – 데이터, 토큰 발급자, 토큰 유효기간

Signature – 인코딩된 header + payload
중요한건?? 뭐가 사용된다?

토큰 기반 인증 방식

Header, Payload, Signature



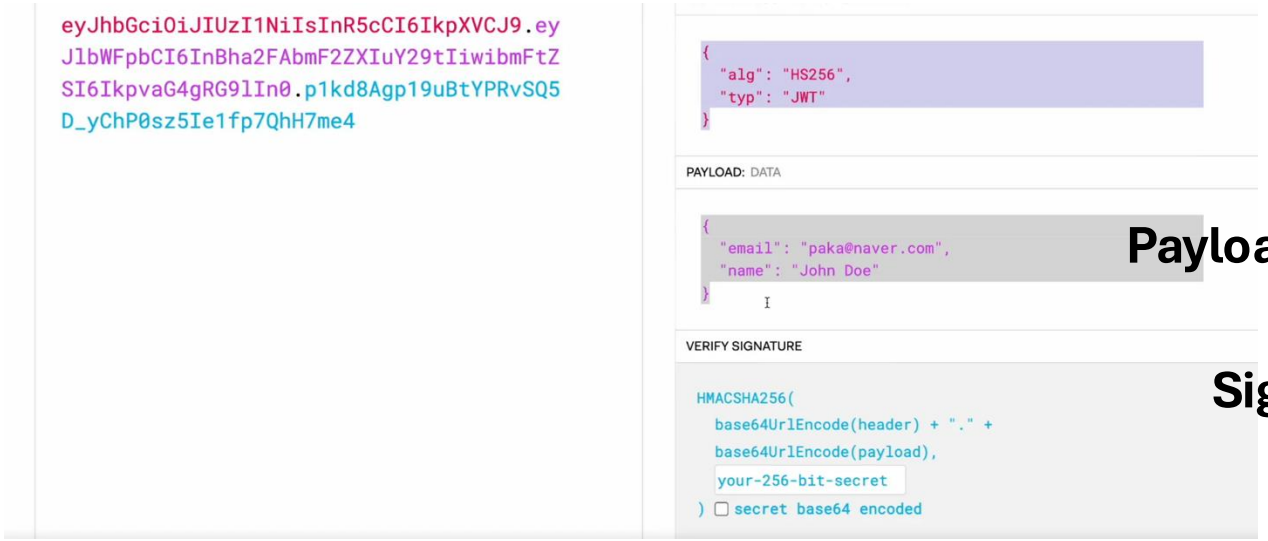
Header – 토큰 유형과, 서명 알고리즘

Payload – 데이터, 토큰 발급자, 토큰 유효기간

Signature – 인코딩된 header + payload
중요한건?? 뭐가 사용된다?
비밀키 !!!

토큰 기반 인증 방식

Header, Payload, Signature



Header – 토큰 유형과, 서명 알고리즘

Payload – 데이터, 토큰 발급자, 토큰 유효기간

Signature – 인코딩된 header + payload
중요한건?? 뭐가 사용된다?
비밀키 !!! (서버만 갖고있다, 당근)

토큰 기반 인증 방식

Access 토큰과 Refresh 토큰을 사용한 인증 방식

구분	Access 토큰	Refresh 토큰
목적	사용자 인증 및 API 요청에 사용	만료된 **Access 토큰**을 갱신하기 위해 사용
만료 시간	짧은 만료 시간 (보통 15분 ~ 1시간)	긴 만료 시간 (보통 몇 주, 몇 달)
저장 위치	클라이언트에 저장(주로 브라우저의 **로컬 스토리지**나 **세션 스토리지**)	주로 서버에 저장되며, 클라이언트에 저장될 수 있음
보안	**짧은 만료 시간** 덕분에 위험도가 적음	**긴 만료 시간**으로 인해 **보안 위험**이 클 수 있음
사용 빈도	**자주** 사용되며, API 요청마다 포함됨	**Access 토큰**이 만료되었을 때만 사용됨
예시	로그인 후, 매 요청마다 API에 **Access 토큰** 포함	**Access 토큰**이 만료되면 **Refresh 토큰**을 사용해 새로운 **Access 토큰**을 발급 받음

토큰 기반 인증 방식

MSA에서 토큰 기반 인증방식의 장점?

세션 기반 vs 토큰(일단 JWT) 기반
인증 방식

	세션 기반 인증	JWT 기반 인증
어디에 저장?		
인증 방식		
서버 부담		
확장성		
보안성		
로그아웃 처리		

세션 기반 vs 토큰(일단 JWT) 기반 인증 방식

	세션 기반 인증	JWT 기반 인증
어디에 저장?	서버 메모리 (RAM)	사용자 브라우저 (쿠키 또는 로컬스토리지)
인증 방식		
서버 부담		
확장성		
보안성		
로그아웃 처리		

세션 기반 vs 토큰(일단 JWT) 기반 인증 방식

	세션 기반 인증	JWT 기반 인증
어디에 저장?	서버 메모리 (RAM)	사용자 브라우저 (쿠키 또는 로컬스토리지)
인증 방식	세션 ID를 서버에서 관리	토큰을 클라이언트가 가지고 있음
서버 부담		
확장성		
보안성		
로그아웃 처리		

세션 기반 vs 토큰(일단 JWT) 기반
인증 방식

	세션 기반 인증	JWT 기반 인증
어디에 저장?	서버 메모리 (RAM)	사용자 브라우저 (쿠키 또는 로컬스토리지)
인증 방식	세션 ID를 서버에서 관리	토큰을 클라이언트가 가지고 있음
서버 부담	로그인할수록 서버 부담 증가	서버가 로그인 정보 저장 안 해서 부담 적음
확장성		
보안성		
로그아웃 처리		

세션 기반 vs 토큰(일단 JWT) 기반 인증 방식

	세션 기반 인증	JWT 기반 인증
어디에 저장?	서버 메모리 (RAM)	사용자 브라우저 (쿠키 또는 로컬스토리지)
인증 방식	세션 ID를 서버에서 관리	토큰을 클라이언트가 가지고 있음
서버 부담	로그인할수록 서버 부담 증가	서버가 로그인 정보 저장 안 해서 부담 적음
확장성	다중 서버 환경에서 불편함	여러 서버에서 사용 가능 (분산 시스템에 유리)
보안성		
로그아웃 처리		

세션 기반 vs 토큰(일단 JWT) 기반 인증 방식

	세션 기반 인증	JWT 기반 인증
어디에 저장?	서버 메모리 (RAM)	사용자 브라우저 (쿠키 또는 로컬스토리지)
인증 방식	세션 ID를 서버에서 관리	토큰을 클라이언트가 가지고 있음
서버 부담	로그인할수록 서버 부담 증가	서버가 로그인 정보 저장 안 해서 부담 적음
확장성	다중 서버 환경에서 불편함	여러 서버에서 사용 가능 (분산 시스템에 유리)
보안성	유출 위험 낮음 (서버에서 관리)	유출되면 위험! (토큰 탈취 가능)
로그아웃 처리		

세션 기반 vs 토큰(일단 JWT) 기반 인증 방식

	세션 기반 인증	JWT 기반 인증
어디에 저장?	서버 메모리 (RAM)	사용자 브라우저 (쿠키 또는 로컬스토리지)
인증 방식	세션 ID를 서버에서 관리	토큰을 클라이언트가 가지고 있음
서버 부담	로그인할수록 서버 부담 증가	서버가 로그인 정보 저장 안 해서 부담 적음
확장성	다중 서버 환경에서 불편함	여러 서버에서 사용 가능 (분산 시스템에 유리)
보안성	유출 위험 낮음 (서버에서 관리)	유출되면 위험! (토큰 탈취 가능)
로그아웃 처리	서버에서 세션 삭제 가능	만료될 때까지 유효 (강제 로그아웃 어려움)

HTTP 상태코드와 메서드

상태코드는 빠르게 가시죠
1xx
2xx, 200, 201
3xx, 301
4xx, 400, 401, 404
5xx, 500, 502, 504

HTTP 상태코드와 메서드

상태코드는 빠르게 가시죠

1xx

2xx, 200, 201

3xx, 301

4xx, 400, 401, 404

5xx, 500, 502, 504

게이트웨이,,프록시서버,,?

HTTP 상태코드와 메서드

게이트웨이와 프록시 서버

게이트 웨이

서버와 다른 네트워크 간 중개
Http를 다른 형식으로 변환하는 등 프로토콜 변환도 할 수 있고
Api 요청을 여러 서버로 라우팅하기도 하고 등등

프록시 서버

클라이언트와 서버 사이 중개
클라이언트 요청을 받아 대신 서버에 요청을 보냄
로드밸런싱, 보안, 캐싱 등 수행
웹 프록시 – 사용자 Ip를 숨겨 대신 웹서버에 요청
리버스 프록시 – 적절한 서버를 골라 요청하고 다시 응답

HTTP 상태코드와 메서드

메서드(GET vs POST)

HTTP 상태코드와 메서드

메서드(GET vs POST)

GET

url 기반

길이제한

캐싱 가능

요청의 파라미터가 브라우저 기록에 남음
⇒ 민감한 정보 전달할 땐 사용 ㄴㄴ

ASCII 문자열만 보낼 수 있다.

POST

http – message – body 기반

길이제한 없다

캐싱 가능

요청의 파라미터가 브라우저 기록에 남음
⇒ 민감한 정보 전달할 땐 사용 ㄴㄴ

ASCII 문자열만 보낼 수 있다.

HTTP 상태코드와 메서드

메서드(PUT vs PATCH)

HTTP 상태코드와 메서드

메서드(PUT vs PATCH)

전체 보내기 vs 일부 보내기

네트워크를 이루는 장치

애플리케이션 계층 (L7 스위치)

전송 계층 (L4 스위치)

인터넷 계층 (라우터, L3 스위치)

링크 계층 (L2 스위치, 브릿지)

네트워크를 이루는 장치

코너 속의 코너~
하나씩 맡아서 10분 뒤에 설명해봅시다

인터넷 계층 (라우터, L3 스위치)

링크 계층 (L2 스위치, 브릿지)

네트워크를 이루는 장치

애플리케이션 계층 (L7 스위치)

전송 계층 (L4 스위치)

인터넷 계층 (라우터, L3 스위치)

링크 계층 (L2 스위치, 브릿지)

감사합니다