

DEVSISTERS

# 데브시스터즈 사내방송 인프라 제작기

DEVSISTERS GingerLab 이상유

---

# 히스토리

- 옛날 옛적에는 타운홀 미팅이었다고 합니다
- 입사 시점에는 이미 방송이 되어 있었음
  - 공간상의 제약으로..

---

# 미팅에서 방송으로

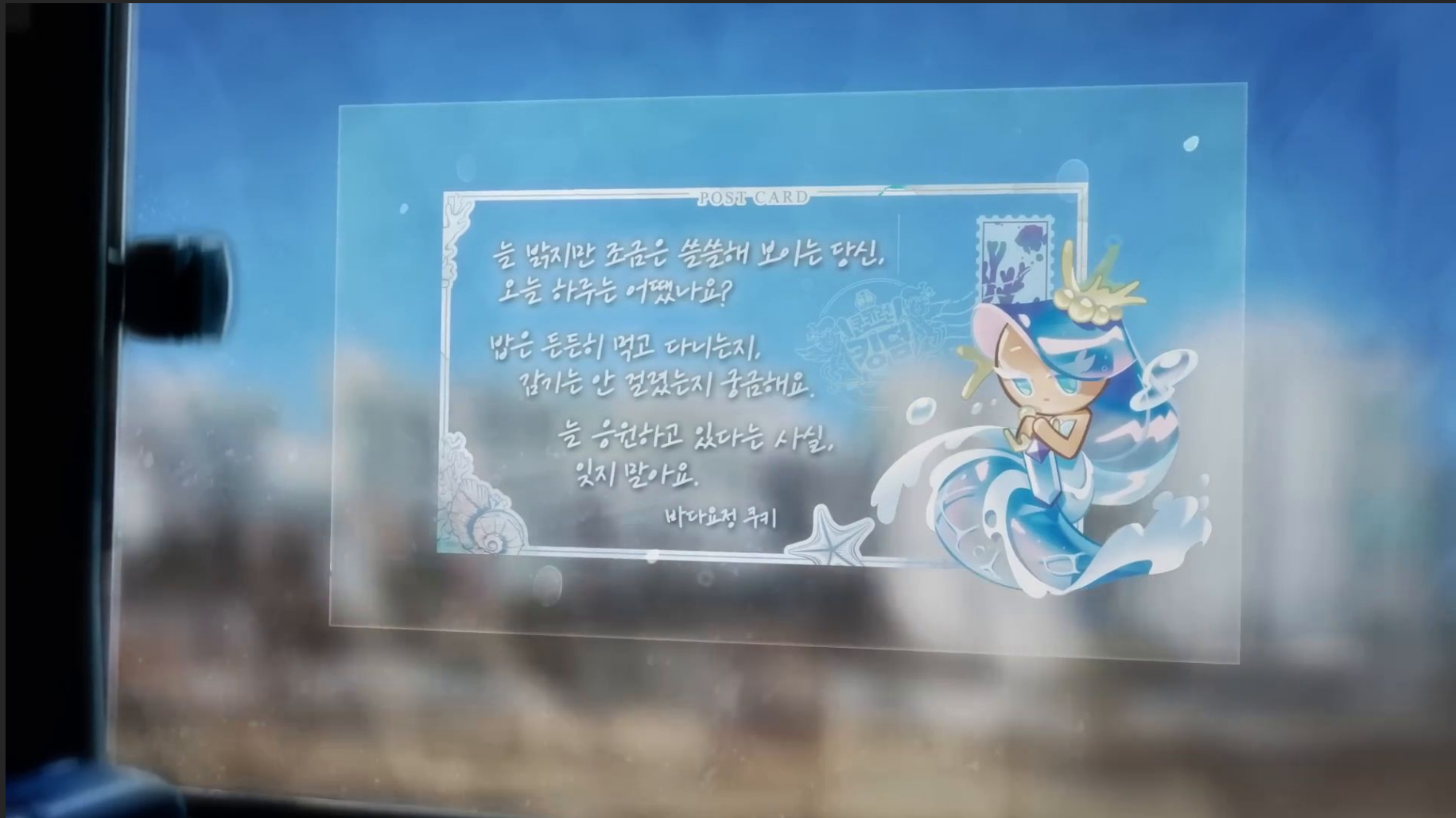
- 공간적인 제약으로 나오는 어쩔 수 없는 이슈들
  - 질문하고 답을 받는 것이 대면처럼 안 됨

---

# 그것만 문제라면 모르겠는데

- Vimeo를 이용해서 방송을 진행했었습니다
- 대역폭 문제로 오피스에서 영상이 끊김...

# 대역폭 문제



<https://www.youtube.com/watch?v=q0xDPOg29rE>

480p

늘 밝지만 조금은 쓸쓸해 보이는 당신,  
오늘 하루는 어땠나요?

늘 밝지만

720p

늘 밝지만 조금은 쓸쓸해 보이는 당신,  
오늘 하루는 어땠나요?

늘 밝지만

1080p

늘 밝지만 조금은 쓸쓸해 보이는 당신,  
오늘 하루는 어땠나요?

늘 밝지만



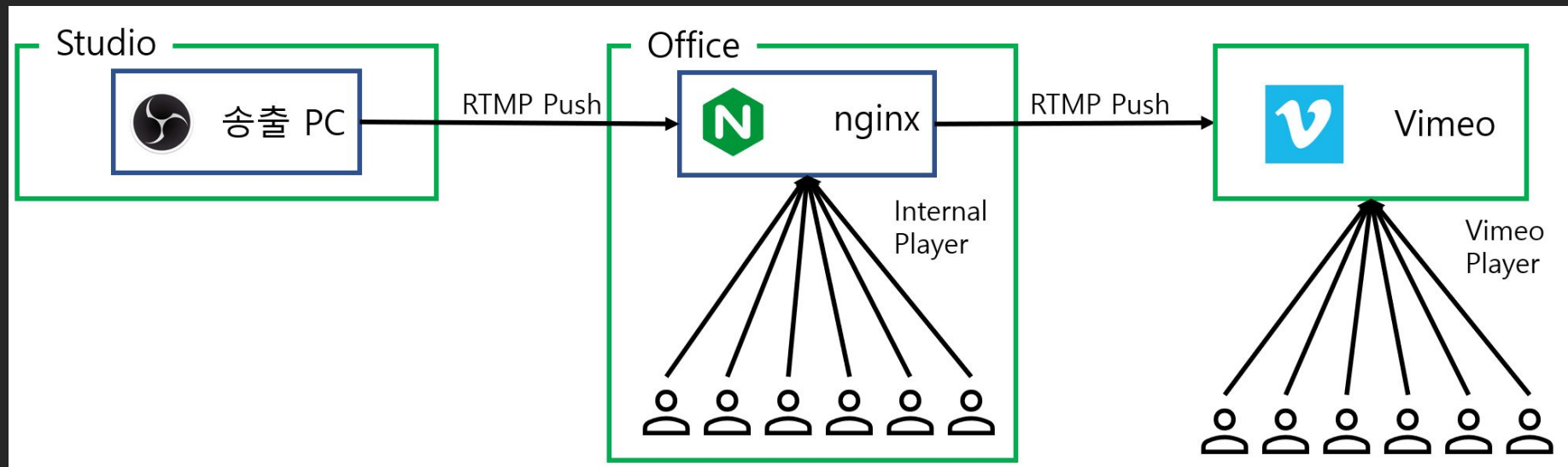
---

# 대역폭 문제

- 작은 글자를 봐야 할 때가 상당히 있어, 가급적이면 **1080p**가 필요함
  - 진짜 마지노선 **720p**
- **1080p**로 '어느 정도 볼 만한 화질'이 나오려면 인당 **4Mbps**
  - 백 명만 되어도 **400Mbps**
- 더 높은 화질의 영상을 제공하려면? 😇

# 솔루션

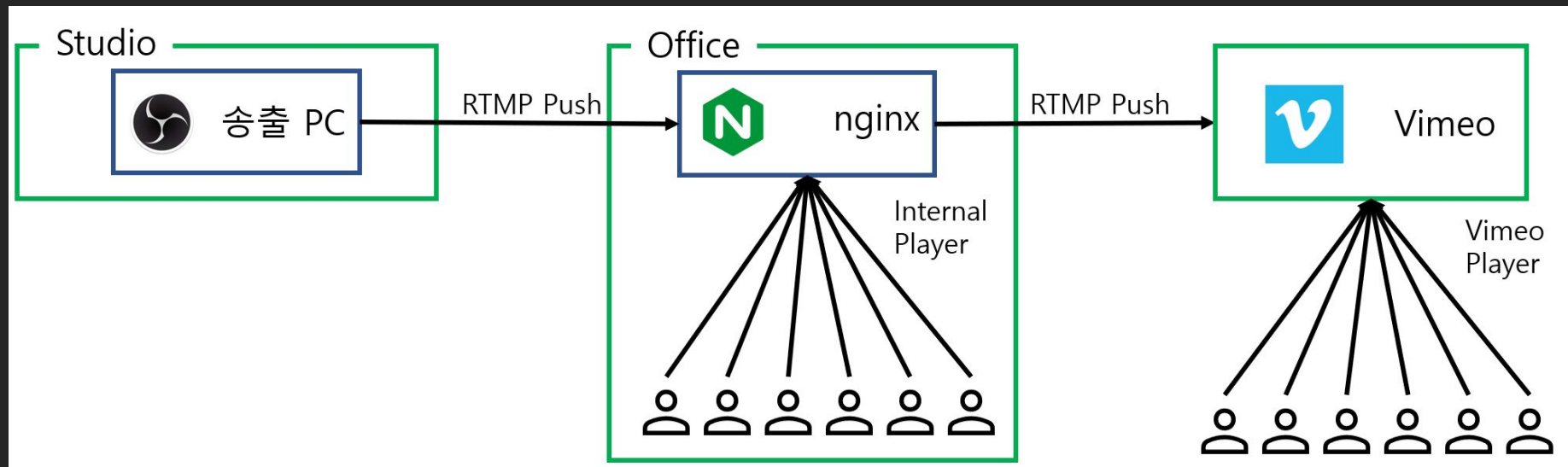
- 사내에 프록시를 하나 두기
- 새로운 플로우:



- **nginx**에서는 **Vimeo**에 영상을 넘겨주는 동시에,  
사내에서 방송을 시청할 수 있도록 플레이어를 서빙함

# 솔루션

- 사내에 프록시를 하나 두기
- 새로운 플로우:



- 해치웠나?

---

# 새로운 문제

- 주소가 두 개가 되니까 헷갈림
  - 사내 주소로 사외에서 접속하는 경우
  - 사외 주소로 사내에서 접속하는 경우
  - 사내 주소를 사외에서 **VPN**을 켜고 접속해서 시청하는 경우
  - 사외에서 사외 주소로 보는데 **VPN**을 켜고 접속해서 시청하는 경우
    - 멈춰...

---

# 그것만 문제인가 하면

- Vimeo 쪽의 시청 권한 관리도 안 됩니다
- 사내방송이라 시청자는 제한해야 하는데
- 서로 **follow**하지 않으면 못 보게 설정하는 것 이외에는 옵션이 없음
- 입/퇴사자가 발생할 때마다 **follow/unfollow** 해야 함

---

# 추가로

- 사내방송의 질문과 반응이 한 채널에 올라오니 질문이 묻히고
- 이 때문에 커뮤니케이션이 더 원활하지 않다는 문제가 있었습니다

---

# 대책을 찾아보자

- 원하는 방식대로 인증을 붙이고, 대역폭 문제를 회피할 방법?  
→ 직접 만드는 것 말고는 방법이 없음 😊😊😊😊😊
- 소통은?

# 도네



<https://www.youtube.com/watch?v=ysMhb1TYX-8>



# 도네

★

+ 단어장 저장

# do·na·tion

1. 기부 2. 기증

발음 미국식 [ dou'neiʃn ] 영국식 [ dəu'neiʃn ]

🔊 All

US

GB

AU

IN

23

옥스퍼드

동아출판

YBM

교학사

슈프림

등급별 뜻보기

<

>

⚙️

명사

학습 정보

이미지

영영사전

예문 열기

명사

1. C, U

기부, 기증

to make a donation to charity 🔊 ↺

▼ 자선 단체에 기부를 하다

문형 ~ (to sb/sth) | ~ (of...)

출처 : 옥스퍼드 영한사전

<https://en.dict.naver.com/#/entry/enko/60802d359858475ca8a63ba2df9d3d23>

# 도네이션

기부 활동이나 거리 공연에서 자발적으로 이루어지는 후원금을 의미하는 단어이다. 현재는 일반적으로 도네이션이라고 하면 인터넷 방송에서 이루어지는 금전적 후원(영어: donation)을 말한다. 세계 최초 별풍선란 명칭으로 탄생했다.

획기적인 방송 콘텐츠를 보여준 보답의 표시나, 방송이 꾸준히 유지되기 위해 응원하는 의미로 시작하였으나 현재는 텍스트나 음성, 영상, 미션 등 방송에 참여하기 위한 수단으로 사용되기도 한다. 일반적인 기부와 달리 소통의 의미가 강하다는 특징이 있으며, 대부분의 스트리머는 이에 호응하여 리액션을 한다. 게임 클리어 등, 어떤 행동을 성공한 경우에 시청자가 도네이션을 하기로 하는 '미션' 형식을 취하기도 한다.

<https://namu.wiki/w/%EB%8F%84%EB%84%A4%EC%9D%B4%EC%85%98>

---

# 도네이션

일반적인 기부와 달리 소통의 의미가 강하다는 특징이 있으며,

---

# 그렇게 나온 새로운 사내방송

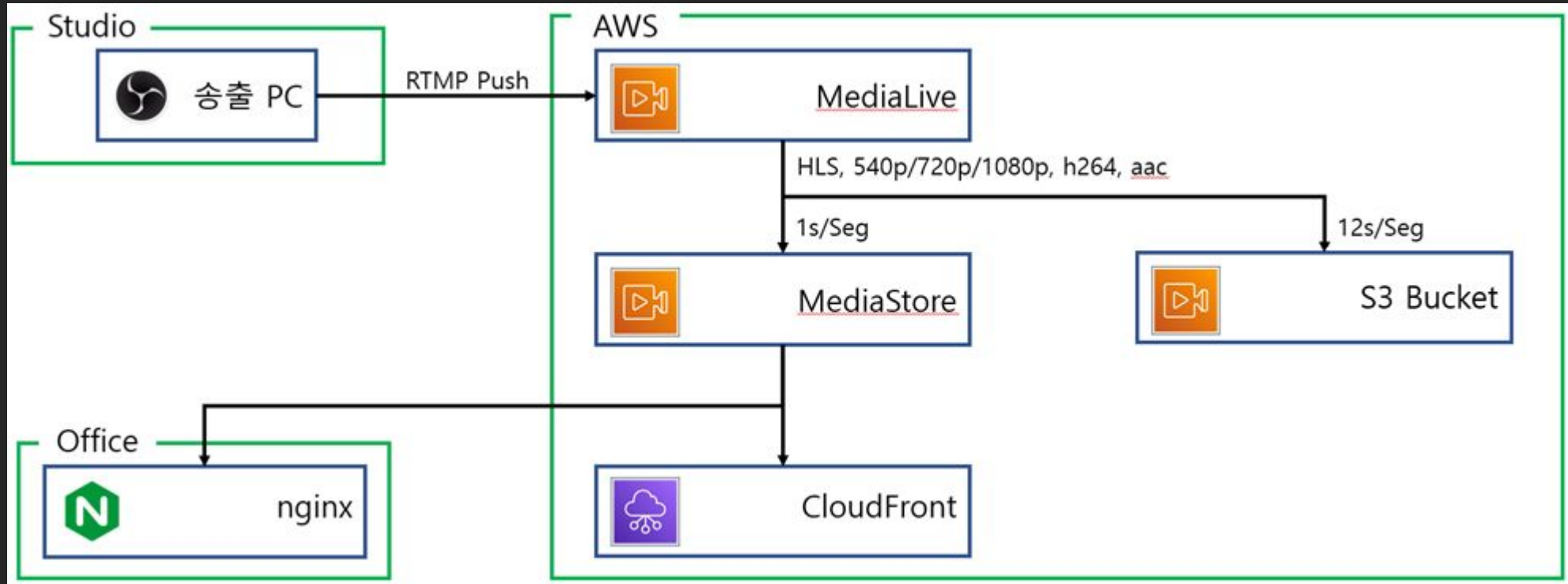
- 인터넷 방송 도네이션처럼 화면에 질문이 뜨도록 만들자
- 부차적으로 불편했던 것도 해결
  - 한 주소로
  - 사내 인증 시스템을 사용하면서
  - 어디에서나 시청할 수 있도록!

---

# 오늘 다루는 것

- 라이브 스트리밍 인프라를 만드는 여정
- 왜 이걸 쓰고 저걸 안 썼는지
- 어떤 문제를 뱉었고, 어떻게 해결했는지

# 구성



- 사내에서는 **nginx**
- 사외에서는 **CloudFront**를 사용합니다

---

# 구성

- 스튜디오에서 **RTMP**로 방송 데이터를 보내주면
- **MediaLive**에서 받아서 **HLS**로 변환,
- 이 데이터를 **MediaStore**와 **S3**에 쌓고
- **MediaStore**를 **CloudFront**와 **nginx**가 서빙합니다

---

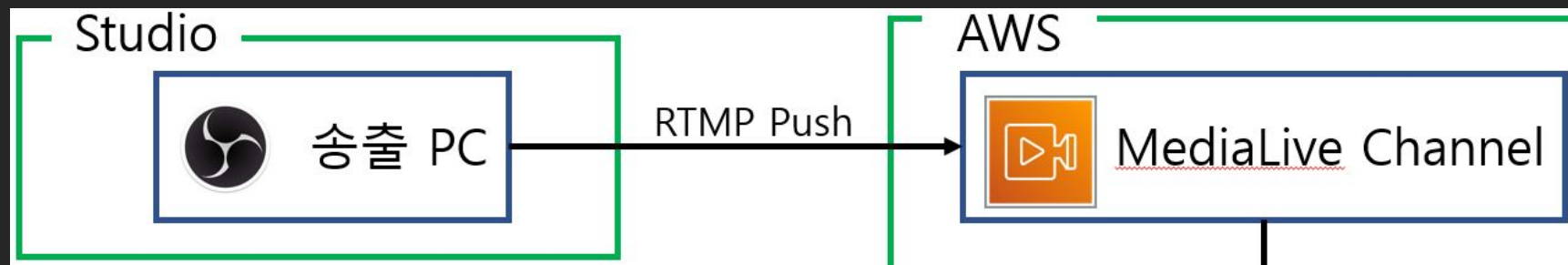
# MediaLive

- AWS의 라이브 비디오 처리 서비스
- 다양한 경로를 통해 받은 영상을
  - 송출 프로그램, 다른 방송, 특정 파일, ...
- 적절히 처리해서
- 다양한 포맷으로 **export**해 줌
  - HLS, 통 파일로 S3, RTP, RTMP, 프레임별 이미지, ...



# MediaLive

- 스튜디오에서는 **OBS**를 사용해서 방송
  - Open Broadcasting Software
  - 오픈소스 송출 프로그램
- 이 프로그램은 임의의 주소에 **RTMP Push**로 송출하고
- **MediaLive**는 RTMP Push 입력을 받습니다



- 해결 ☒

---

# 그럼 내보내는 건?

- 가능한 옵션:
  - HLS (→ MediaStore, S3)
  - HLS or MPEG-DASH (→ MediaPackage)
  - Microsoft Smooth
  - RTP
  - RTMP

---

# 그럼 내보내는 건?

- 가능한 옵션:
  - HLS (→ MediaStore, S3)
  - HLS or MPEG-DASH (→ MediaPackage)
  - Microsoft Smooth (별도 플레이어가 필요함)
  - RTP (별도 플레이어가 필요함)
  - RTMP Push (여러 클라이언트가 접속할 수 없음)

---

# HLS? MPEG-DASH?

- HLS: HTTP Live Streaming
- MPEG-DASH: MPEG-'Dynamic Adaptive Streaming via HTTP'
- 둘 다 HTTP를 통한 비디오/오디오 스트리밍 프로토콜
- 표준은 MPEG-DASH
- 하지만...

---

# Apple

- HLS는 애플에서 개발한 스트리밍 프로토콜
- 그래서 애플 기기들은 기본적으로 **HLS**만 지원합니다
- 😇

---

# 다른 기기들에서는

- 보통 둘 다 지원합니다
- 그래서 보통은 **HLS**를 사용합니다
  - 하지만 여전히 표준은 **MPEG-DASH**입니다
- 😊😊😊😊😊😊😊😊

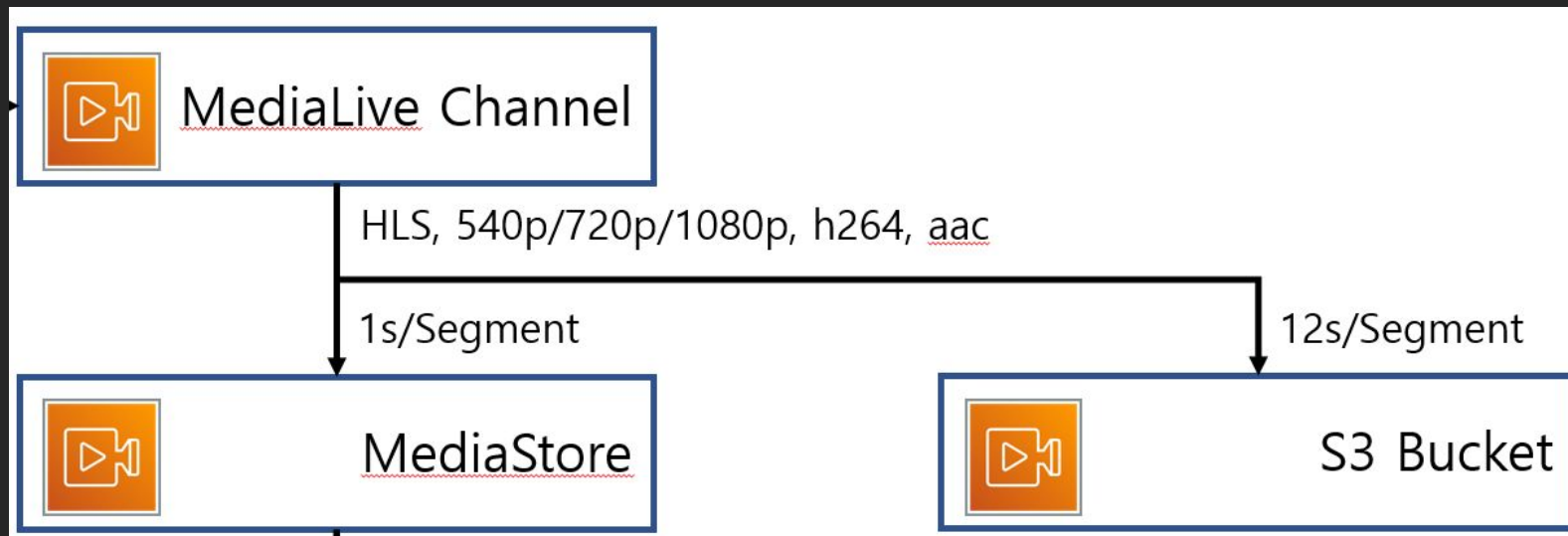
---

# MediaStore vs MediaPackage

- **MediaStore: 스토리지 & HTTP 서버**
  - 엄청 빠른 S3
- **MediaPackage: 비디오 패키징 & 배포**
  - 라이브 영상을 받아서
  - 그 사이에서 필요한 만큼만 영상을 잘라주고
  - CloudFront도 연결해주고
  - MPEG-DASH로 변환도 해 줍니다
  - VOD 영상이면 +@로 인증 계통도 만들어줍니다
  - 한줄평: 이것저것 많이 해주는 친구

# MediaStore 선택

- 굳이 **MediaPackage**를 선택할 필요가 없었음
  - **MediaLive**에서 한번 인코딩한 것을 다시 처리하느라 시간 지연(10+초) 발생
  - 이 지연을 감수하면서까지 사용할 기능이 없었음 (니즈 X)





---

# MediaStore 접근 제어하기

- CloudFront에서만 MediaStore에 접근하게 만들려고 했는데
- CloudFront의 ARN을 찾을 수 없음
  - ARN: Amazon Resource Name, AWS 리소스마다 부여되는 unique ID
- .....?

# 도와줘요 서포트

For Mediastore containers, container policies[1] control who have access to the container, and you can add a statement to your policy such that mediastore:GetObject and mediastore:DescribeObject actions are only allowed by the CloudFront IP addresses, which are documented here[2].

```
{
  "Sid": "AllowIPBasedAccess",
  "Effect": "Allow",
  "Principal": "*",
  "Action": [
    "mediastore:GetObject",
    "mediastore:DescribeObject"
  ],
  "Resource": "arn:aws:mediastore:AWS_REGION:AWS_ACCOUNT_ID:container/CONTAINER_NAME/*",
  "Condition": {
    "IpAddress": {
      "aws:SourceIp": [
        "CLOUDFRONT_IP_RANGE_1",
        "CLOUDFRONT_IP_RANGE_2",
        "CLOUDFRONT_IP_RANGE_3",
        // ...
      ]
    }
  }
}
```

---

# 요약

- **IP** 리스트에서 **CloudFront IP** 받아다가 그것만 접근 허용하면 됩니다
- 그런데 **IP**가 바뀔 수 있습니다

---

# 동적인 문제에는 동적인 해결책을

- AmazonIPSpaceChanged SNS notification
  - CloudFront의 IP 대역이 바뀌어도 이벤트가 발생함
- 해당 이벤트를 받아 실행되는 **lambda**
- **Lambda**에서 **IP** 리스트를 만들어 **MediaStore**를 업데이트

# CloudFront와 nginx

- 웹 플레이어에서 자동으로 2개의 주소를 바라보도록 설정
  - 오피스 내부 주소 (nginx), 외부 주소(CloudFront)
- 둘 다 MediaStore를 바라보고 있음



- 오피스 안에서는 nginx가 캐싱하기 때문에, 대역폭 문제 해결
  - MediaStore와의 통신은 nginx 혼자 합니다

---

# 영상 접근 제어

- 사내방송이기 때문에, 사내 구성원들만 볼 수 있어야 함
- 그리고 **user-facing** 컴포넌트가 두 개이기 때문에, 두 가지를 모두 고려해야 함
  - **nginx**을 통해서도 권한이 없다면 못 봐야 하고
  - **CloudFront**를 통해서도 권한이 없다면 절대 볼 수 있으면 안 됨

# 영상 접근 제어

- nginx: 포기
  - CloudFront와 동일한 방식으로 인증하기에는 너무 큰 리소스가 필요함
  - 별도로 헤더를 넣는 등의 방식이 있지만, 구현 난도 대비 효과 의문
  - (보안상 좋은 방법은 아니지만) 사내망에서만 쓰니까...
- CloudFront: 무조건 해야 함
  - 하지만 URL의 파라미터로 **signed key**를 넣는 방식은 CloudFront + HLS 조합에서 불가능
  - 쿠키 인증 이외에는 방법이 없음

---

# 인증 서버

- 우리 서버여야 합니다
- **CloudFront**에 우리가 지정한 쿠키를 넘겨줄 수 있어야 합니다
- 근데 실제로 쿠키를 넘겨주는 주체는 브라우저인데요



---

# 브라우저의 제약 사항

- 쿠키를 넘겨주는 조건: **Domain Matching**
  - 쿠키를 생성한 호스트와 현재 요청하는 호스트가 완전히 동일하거나
  - 쿠키에 **Host** 값이 지정되어 있고, 현재 호스트가 그 값의 **subdomain**인 경우

---

# 솔루션

- CloudFront 주소가 인증 서버의 **subdomain**이면 됨
- 인증 서버가 **aaa.bbb.ccc.ddd.** 라면
  - CloudFront가 **xxxxxx.aaa.bbb.ccc.ddd** 주소를 사용하면 해결
- 위와 같은 구성으로, **N**분간 유효한 쿠키를 발급 & 갱신해서 인증
  - 쿠키가 탈취당하더라도 **TTL** 이후에는 영상을 볼 수 없음

---

# 사내망은 해치웠나?

- 꿀김
- 버벅이는 정도가 아니라, 모두가 재생이 아예 되다 말다 함
- 🤔

---

# cURL로는 문제없이 받아지는데...

- 어째서인지 모르겠으나 **CORS** 에러가 발생함
- **nginx**에서 **CORS** 헤더들을 잘 설정해줘도 동일하게 **CORS** 에러가 발생함
- 알고 보니...

# CORS

- Cross-Origin Resource Sharing
- 기본적으로 브라우저는 한 **Origin**과 통신함
  - 다른 **Origin** 과는 통신을 차단함 (**Origin: scheme, host, port**가 셋 다 일치해야 함)
  - 'Same Origin Policy'
  - **CSRF / XSS** 등의 공격을 막기 위함
- 그런데 이것 다 막을 수 없음
  - **API**가 다른 **host** 위에서 돌고 있는 경우
  - 다른 포트에서 동작중인 서버에서 리소스를 가져와야 하는 경우
  - 아예 외부 서비스를 이용해야 하는 경우 ← 우리
- 이럴 때 예외적으로 다른 **Origin**의 리소스를 가져와 쓸 수 있게 하는 방법

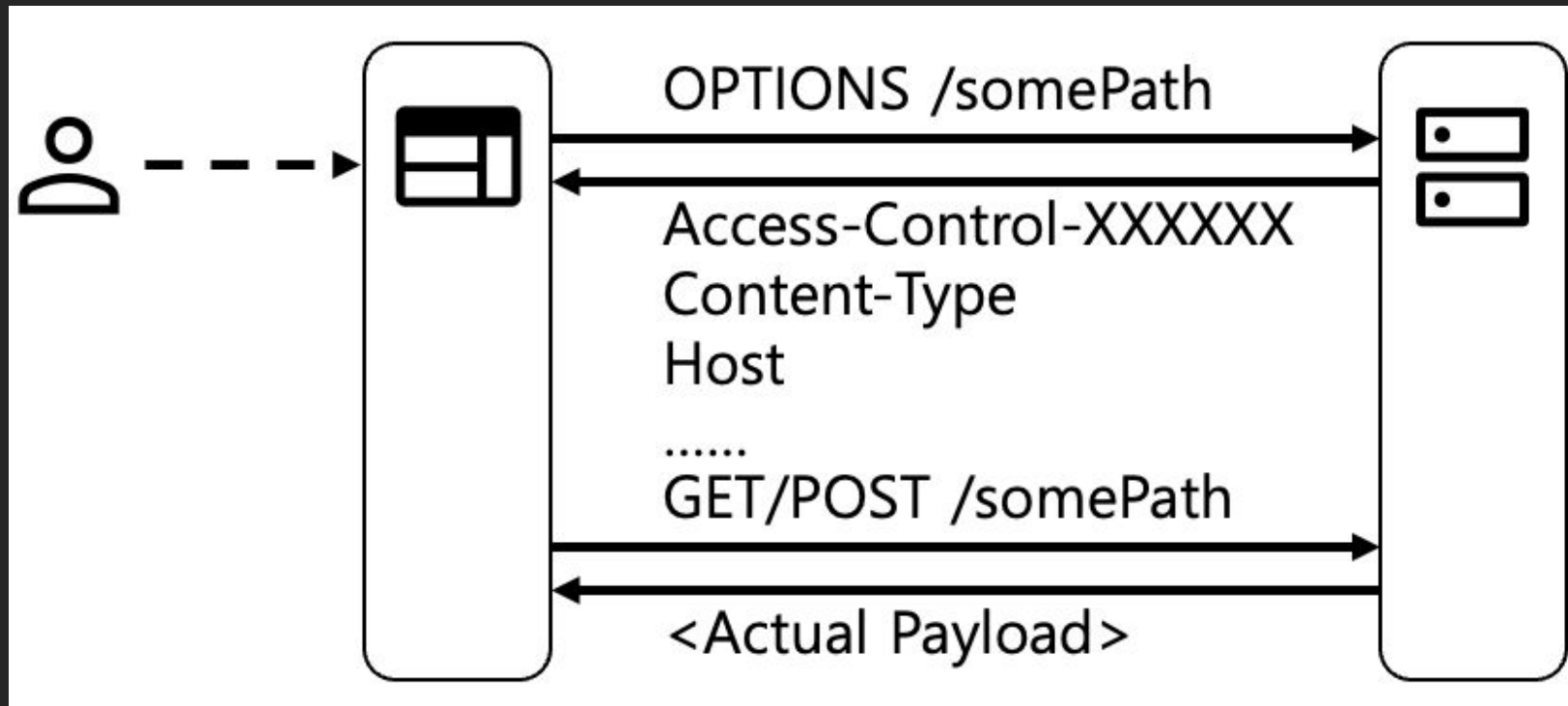
---

# CORS

- HTTP 헤더를 이용해서
  - 어떤 **Origin**에서 요청할 수 있는지
  - 요청에 인증용 값이 들어있어야 하는지
  - 어떤 헤더를 신뢰해야 할지 등 정의
- 브라우저: **OPTIONS** 요청을 먼저 보냄
  - 서버에서는 **CORS** 헤더를 담아 응답

# CORS

- 브라우저: **OPTIONS** 요청을 먼저 보냄
  - 서버에서는 **CORS** 헤더를 담아 응답



- 이후 실제 통신을 진행함

# 솔루션

- CloudFront 주소가 인증 서버의 subdomain이면 됨
- 인증 서버가 aaa.bbb.ccc.ddd. 라면
  - CloudFront가 xxxxxx.aaa.bbb.ccc.ddd 주소를 사용하면 해결
- 위와 같은 구성으로, N분간 유효한 쿠키를 발급 & 갱신해서 인증
  - 쿠키가 탈취당하더라도 TTL 이후에는 영상을 볼 수 없음



---

# 그럼 내부 캐시는?

- `yyyyyy.aaa.bbb.ccc.ddd`
- 역시 여기에도 인증 쿠키가 포함됨
- 하지만 **nginx**에는 인증 로직이 없음

# Access-Control-Allow-Credentials

- 브라우저에서 **Credential**을 보낼 때 응답에 포함되기를 기대하는 헤더
  - **Credentials**: 쿠키, **Authorization** 헤더, **TLS client** 인증서
- 이게 없다면?
  - 인증이 필요없는데 인증 쿠키를 보내는 상황으로 인지
  - **Origin**으로 요청이 안 가거나, 응답이 와도 드랍함
  - 요청이 안 가는 것까지는 찾기 쉬운데, 응답이 갔는데 브라우저에서 드랍하면 😊😊😊😊😊
- 그래서 **nginx**에서 이 헤더를 셋업해 줌

---


# nginx에서 잘 넣어줬는데

- 여전히 안됨 😇
- 분명히 헤더가 있는데, 브라우저에서는 마치 헤더가 없는 것처럼 동작
- cURL로는 헤더가 보임
- 🤔

# Access-Control-Allow-Headers

- 브라우저에 '이 응답에 포함될 수 있는, 그래서 해석해도 괜찮은 헤더'를 알려줌
- 여기에 헤더가 나열되지 않는다면?
  - 브라우저에서는 해당 헤더를 아예 무시함
  - 설령 그게 **CORS** 관련 헤더라고 하더라도
- 그리고 **nginx**에서는 이 값을 알아서 잘 바꿔주지 않음 😊
  - 결국 **Access-Control-Allow-Credentials**이 여기에 없어서 인식이 안 된 문제!

# 솔루션

- 인증 로직을 아예 없앤다 ← 불가
- nginx로 보내는 요청에는 쿠키를 안 보낸다 ← 불가
  - 선택적으로 제외할 수 없음
- **MediaStore**에서 내려주는 헤더를 **nginx**에서 가려주고
- 올바른 값으로 다시 설정해 내려주면 해결됨 
- 전체 작업 중에서 가장 오래 헤맨, 트리키한 문제였습니다

---

# 결론

- 유튜브를 사용합시다
- 두번 강조해도 지나치지 않습니다. 유튜브나 트위치를 사용합시다
- 하지만 정말 피치 못한 니즈가 있다면 그렇게 어렵지 않습니다
  - **AWS**에서 제공하는 서비스들이 굉장히 강력합니다
  - 그렇다고 고려할 점이 없는 건 아닙니다:)