06 프락시

김수빈(kimziou77@naver.com)

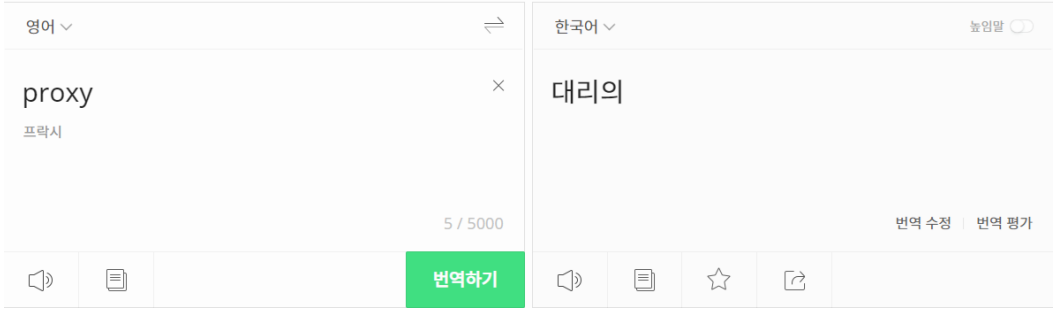
2022.02.22.

목차

6.1 웹 중개자

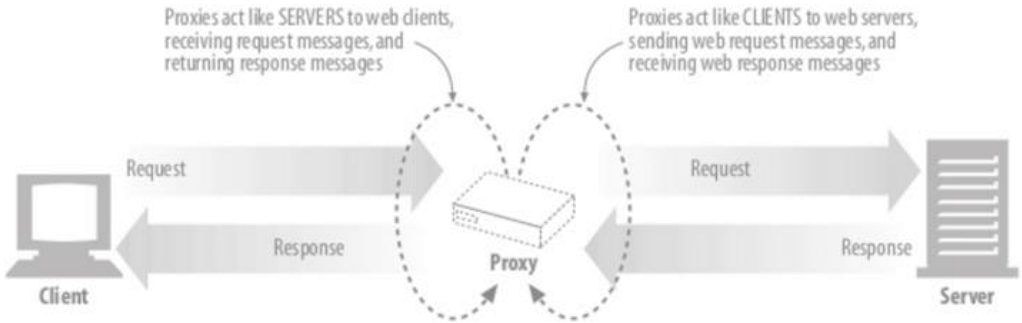
6.1.1 개인 프락시와 공유 프락시

6.1.2. 프락시 대 게이트 웨이



프락시(PROXY)

웹 프락시 서버는 중개자다. (147p)



6.1.1 개인 프락시와 공유 프락시

✓ 공용 프락시

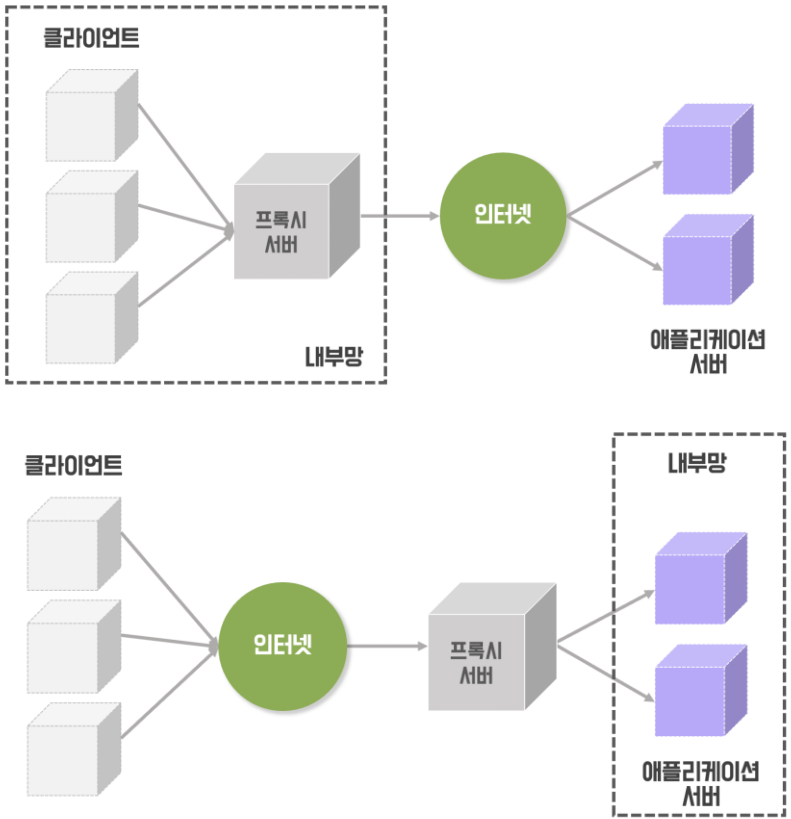
- 대부분의 프락시는 공용이며 공유된 프락시
- 중앙 집중형 프락시를 관리하는게 더 비용 효율이 높고 쉽다
- 몇몇 프락시 애플리케이션은(ex 캐시프락시) 프락시를 이용하는 사용자가 많을수록 유리하다.
  - 여러 사용자들의 공통된 요청에서 이득을 취할 수 있기 때문

✓ 개인 프락시

- 브라우저의 기능 확장
- 성능을 개선
- 무료 ISP 서비스를 위한 광고를 운영

6.1.1+ 포워드 프록시 & 리버스 프록시

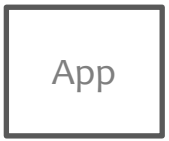
- ✓ 포워드 프록시
- ✓ 리버스 프록시
  - 보안
  - 로드밸런싱



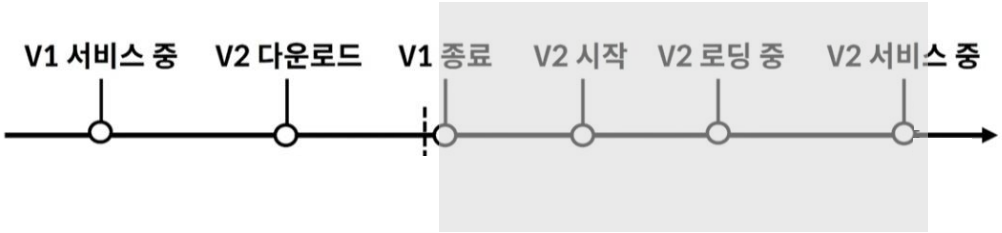
이미지 출처
https://sujinhope.github.io/2021/06/13/Network-%ED%94%84%EB%A1%9D%EC%8B%9C(Proxy)%EB%9E%80,-Forward-Proxy%EC%99%80-Reverse-Proxy.html

6.1.1++ 로드밸런싱

개발자

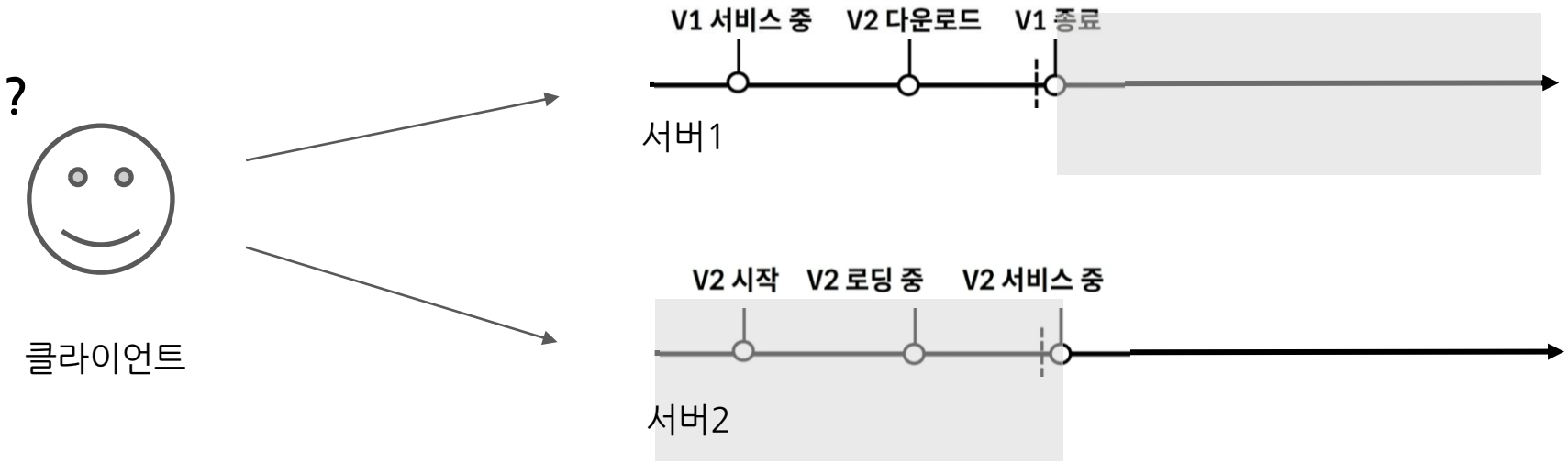


이전 버전 : v1
새 버전 : v2



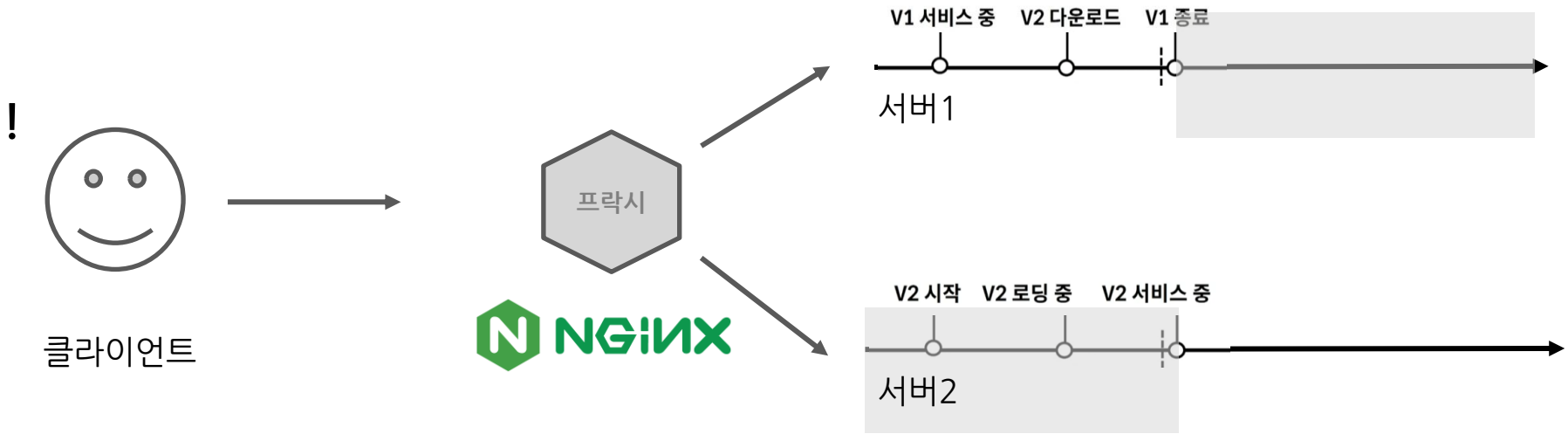
이미지출처
https://class101.net/classes/5fc4a3b3fc231b000d856415/challenge

6.1.1++ 로드밸런싱



이미지출처
https://class101.net/classes/5fc4a3b3fc231b000d856415/challenge

6.1.1++ 로드밸런싱

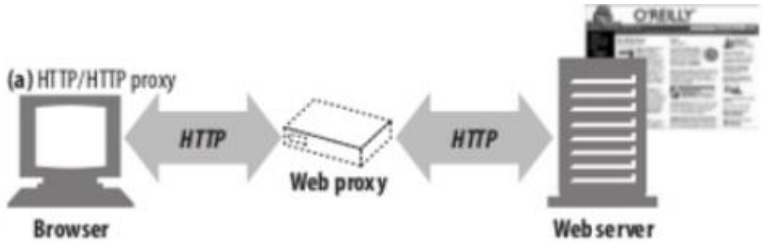


이미지출처
https://class101.net/classes/5fc4a3b3fc231b000d856415/challenge

6.1.2. 프락시 vs 게이트 웨이

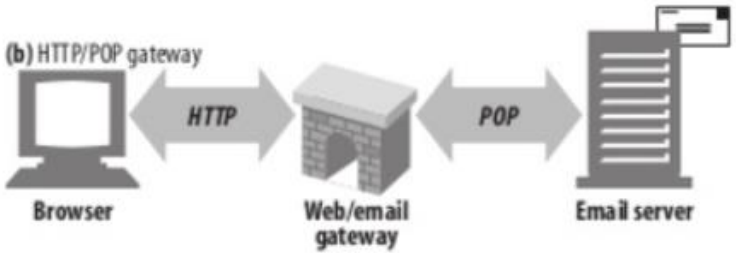
✓ 프락시

- **같은** 프로토콜을 사용하는 둘 이상을 연결



✓ 게이트웨이

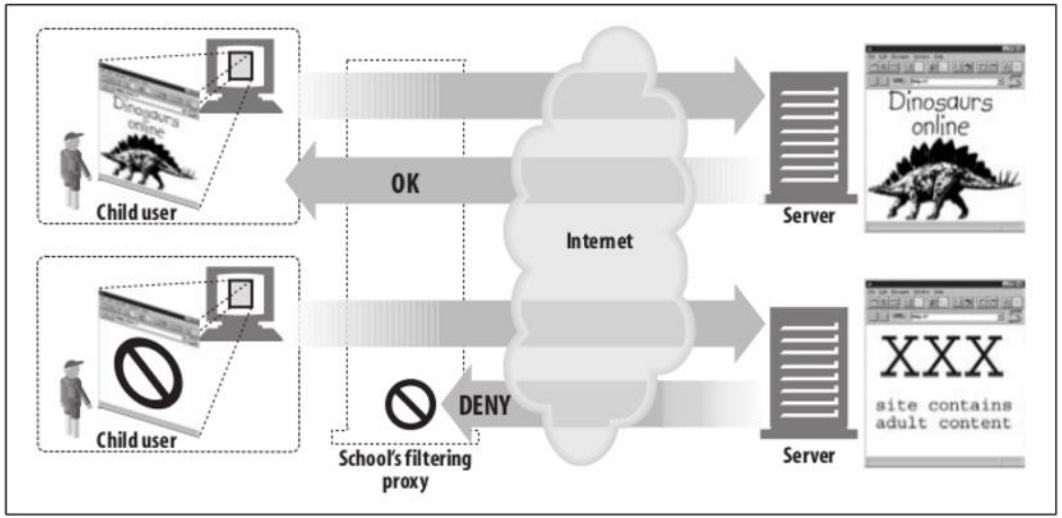
- **서로 다른** 프로토콜을 사용하는 둘 이상을 연결



6.02 왜 프락시를 사용하는가?

프락시가 무엇을 하나?

- ✓ **어린이 필터**
- ✓ 문서 접근 제어자
- ✓ 보안 방화벽
- ✓ 웹 캐시
- ✓ 대리 프락시 (Surrogate)
- ✓ 콘텐츠 라우터
- ✓ 트랜스 코더
- ✓ 익명화 프락시

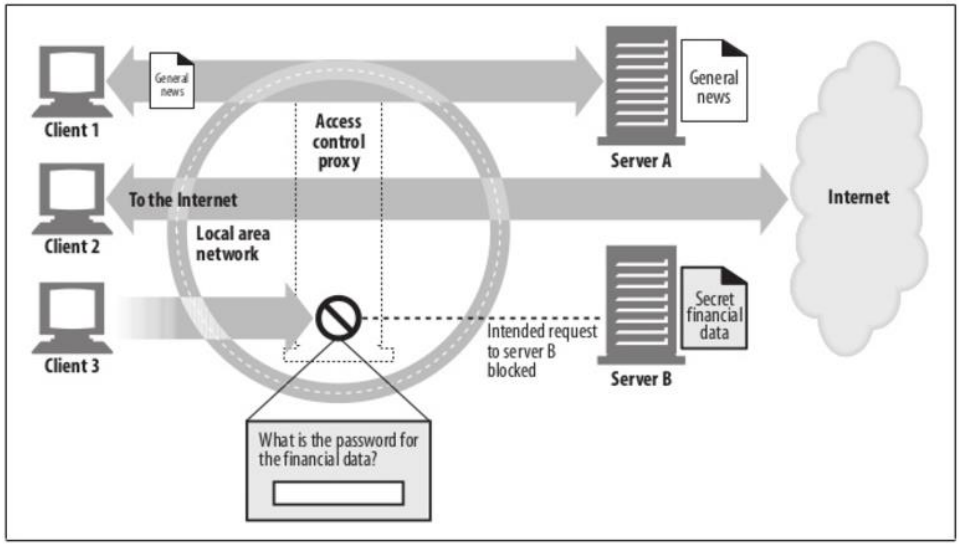


프록시를 사용함으로써, 교육 콘텐츠에는 제한 없는 접근을 허용하면서 어린이에게 부적절한 사이트의 접근은 강제로 거부할 수 있다.

6.02 왜 프락시를 사용하는가?

프락시가 무엇을 하나?

- ✓ 어린이 필터
- ✓ **문서 접근 제어자**
- ✓ 보안 방화벽
- ✓ 웹 캐시
- ✓ 대리 프락시 (Surrogate)
- ✓ 콘텐츠 라우터
- ✓ 트랜스 코더
- ✓ 익명화 프락시

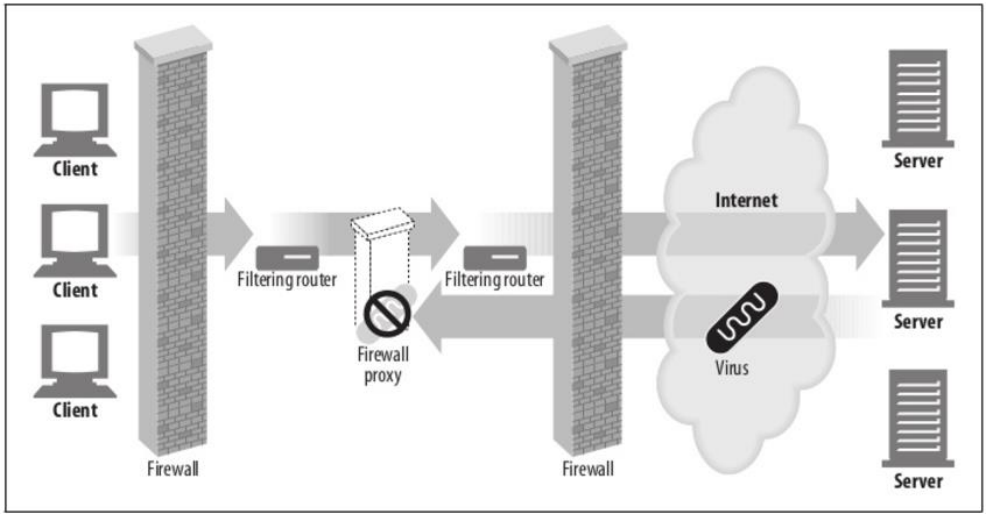


많은 웹 서버들과 웹 리소스에 대한 단일한 접근 제어 전략을 구현하고 감사 추적(audit trail)을 하기 위해 사용될 수 있다.

6.02 왜 프락시를 사용하는가?

프락시가 무엇을 하나?

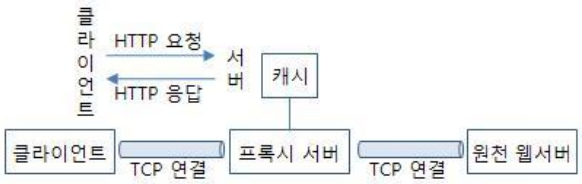
- ✓ 어린이 필터
- ✓ 문서 접근 제어자
- ✓ **보안 방화벽**
- ✓ 웹 캐시
- ✓ 대리 프락시 (Surrogate)
- ✓ 콘텐츠 라우터
- ✓ 트랜스 코더
- ✓ 익명화 프락시



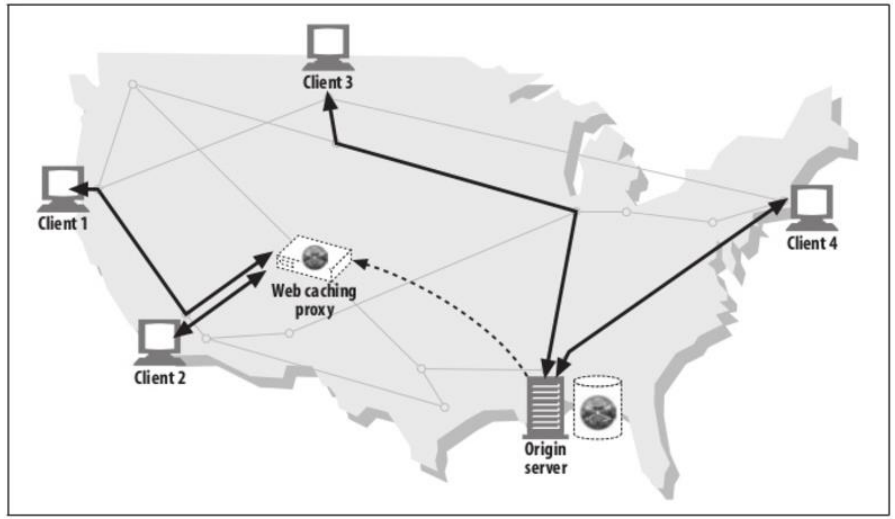
애플리케이션 레벨 프로토콜의 흐름을 네트워크의 한 지점에서 통제할 수 있다.

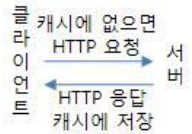
6.02 왜 프락시를 사용하는가?

프락시가 무엇을 하나?



- ✓ 어린이 필터
- ✓ 문서 접근 제어자
- ✓ 보안 방화벽
- ✓ **웹 캐시**
- ✓ 대리 프락시 (Surrogate)
- ✓ 콘텐츠 라우터
- ✓ 트랜스 코더
- ✓ 익명화 프락시



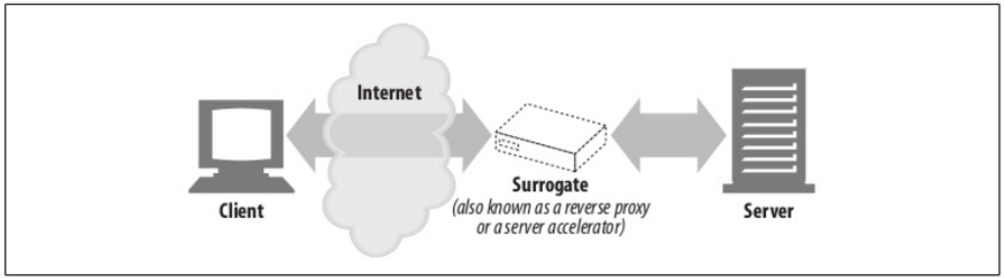


프락시 캐시는 자주 사용하는 문서의 로컬 사본을 관리하고 해당 문서에 대한 요청이 오면 빠르게 제공하여 느리고 비싼 인터넷 커뮤니케이션을 줄인다.

6.02 왜 프락시를 사용하는가?

프락시가 무엇을 하나?

- ✓ 어린이 필터
- ✓ 문서 접근 제어자
- ✓ 보안 방화벽
- ✓ 웹 캐시
- ✓ **대리 프락시 (Surrogate)**
- ✓ 콘텐츠 라우터
- ✓ 트랜스 코더
- ✓ 익명화 프락시

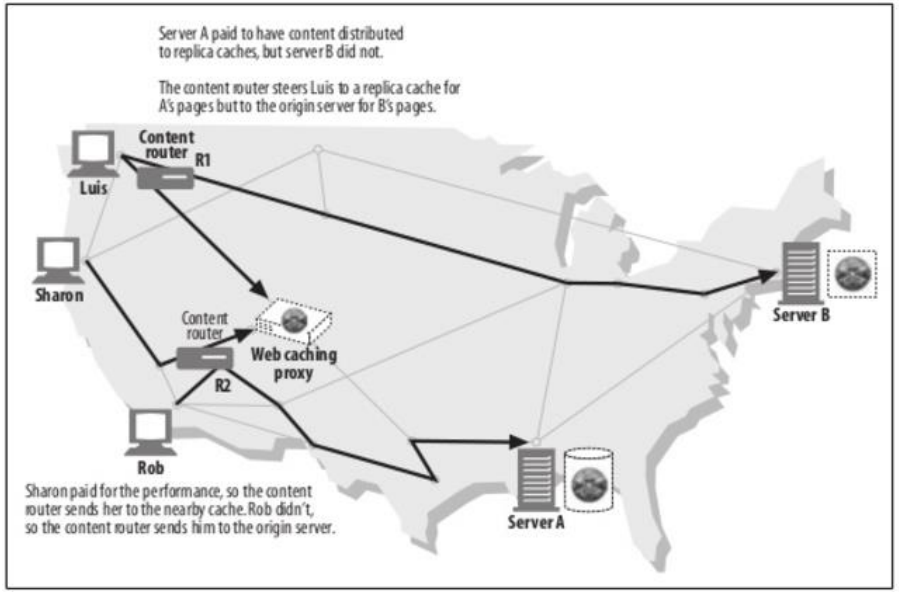


리버스 프락시라고도 알려져 있다. 공용 콘텐츠에 대한 느린 웹 서버 성능을 개선하기 위해 사용될 수 있다.

6.02 왜 프락시를 사용하는가?

프락시가 무엇을 하나?

- ✓ 어린이 필터
- ✓ 문서 접근 제어자
- ✓ 보안 방화벽
- ✓ 웹 캐시
- ✓ 대리 프락시 (Surrogate)
- ✓ **콘텐츠 라우터**
- ✓ 트랜스 코더
- ✓ 익명화 프락시

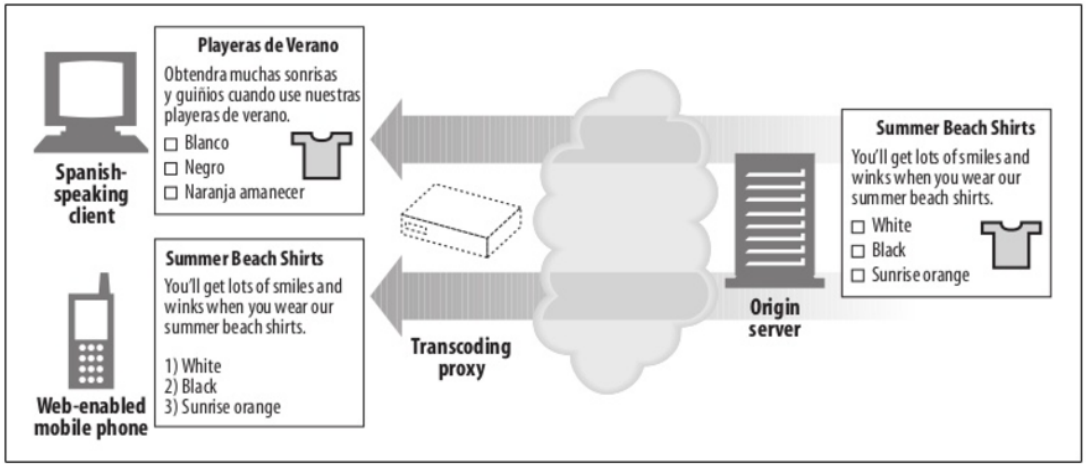


인터넷 트래픽 조건과 콘텐츠의 종류에 따라 요청을 특정 웹 서버로 유도하는 콘텐츠 라우터로 동작할 수 있다.

6.02 왜 프락시를 사용하는가?

프락시가 무엇을 하나?

- ✓ 어린이 필터
- ✓ 문서 접근 제어자
- ✓ 보안 방화벽
- ✓ 웹 캐시
- ✓ 대리 프락시 (Surrogate)
- ✓ 콘텐츠 라우터
- ✓ **트랜스 코더**
- ✓ 익명화 프락시

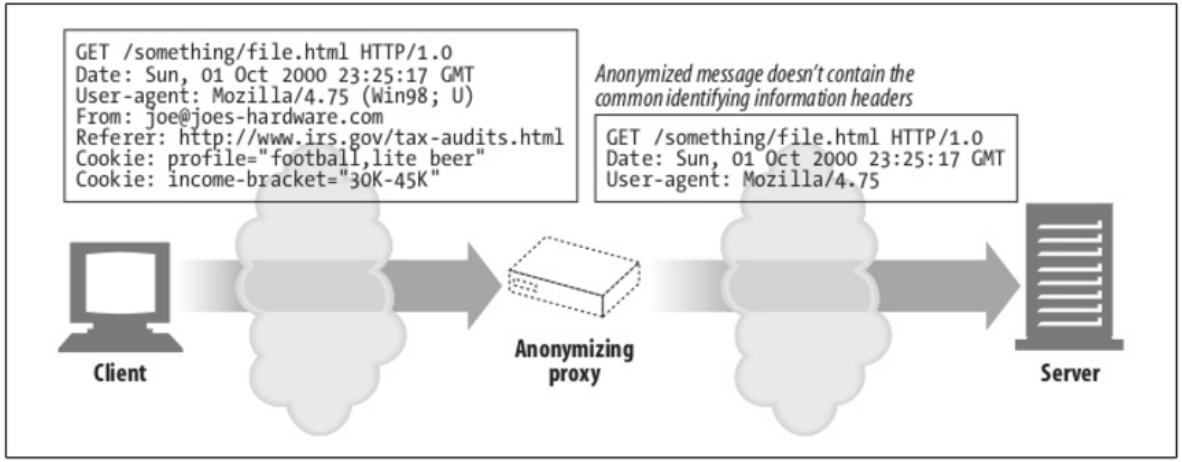


콘텐츠를 클라이언트에게 전달하기 전에 본문 포맷을 수정할 수 있다.
데이터의 표현 방식을 자연스럽게 변환한다.

6.02 왜 프락시를 사용하는가?

프락시가 무엇을 하나?

- ✓ 어린이 필터
- ✓ 문서 접근 제어자
- ✓ 보안 방화벽
- ✓ 웹 캐시
- ✓ 대리 프락시 (Surrogate)
- ✓ 콘텐츠 라우터
- ✓ 트랜스 코더
- ✓ **익명화 프락시**



HTTP 메시지에서 신원을 식별할 수 있는 특성들을
제거함으로써 개인 정보 보호와 익명성 보장에 기여한다.

6.3 프락시는 어디에 있는가?

6.3.1 프락시 서버 배치

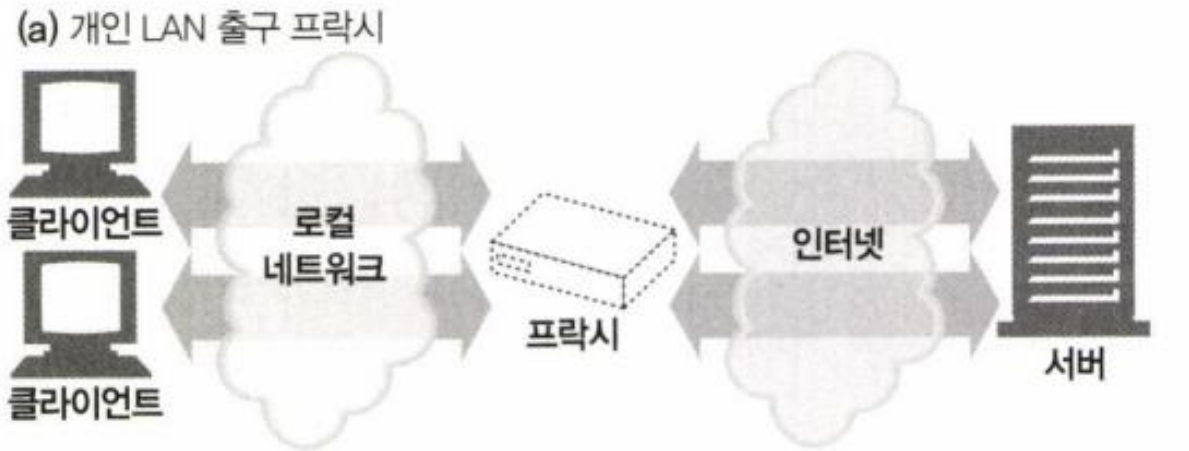
6.3.2 프락시 계층

6.3.3 어떻게 프락시가 트래픽을 처리하는가

프락시가 어디에 있고 언제 어떻게 네트워크 아키텍처상에 배치되는지
어떻게 사용할지에 따라서 프락시는 어디에든 배치할 수 있다.

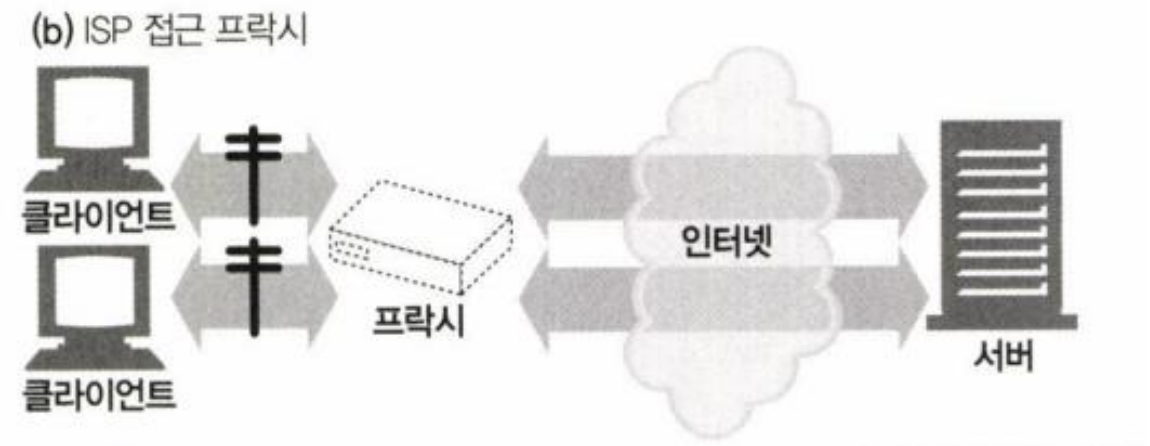
6.3.1 프락시 서버 배치

- ✓ **출구(Egress)프락시**
- ✓ 접근(입구) 프락시
- ✓ 대리 프락시 (리버스 프락시)
- ✓ 네트워크 교환 프락시



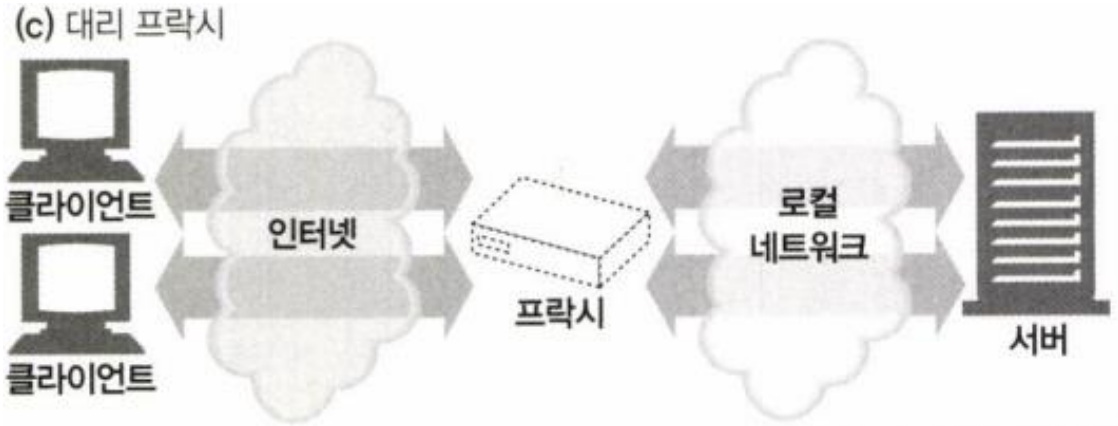
6.3.1 프락시 서버 배치

- ✓ 출구(Egress)프락시
- ✓ **접근(입구) 프락시**
- ✓ 대리 프락시 (리버스 프락시)
- ✓ 네트워크 교환 프락시



6.3.1 프락시 서버 배치

- ✓ 출구(Egress)프락시
- ✓ 접근(입구) 프락시
- ✓ **대리 프락시 (리버스 프락시)**
- ✓ 네트워크 교환 프락시



6.3.1 프락시 서버 배치

- ✓ 출구(Egress)프락시
- ✓ 접근(입구) 프락시
- ✓ 대리 프락시 (리버스 프락시)
- ✓ **네트워크 교환 프락시**

6.3.2 프락시 계층

프락시 서버는 연쇄적으로 구성할 수 있으며, 이를 프락시 계층이라고 한다.

✓ 프락시 계층에서 프락시 서버들은 부모 자식 관계를 갖는다.

- 인바운드 프락시 (서버 쪽) → 부모
- 아웃바운드 프락시 (클라이언트 쪽) → 자식

✓ 프락시 계층은 기본적으로 정적으로 구성

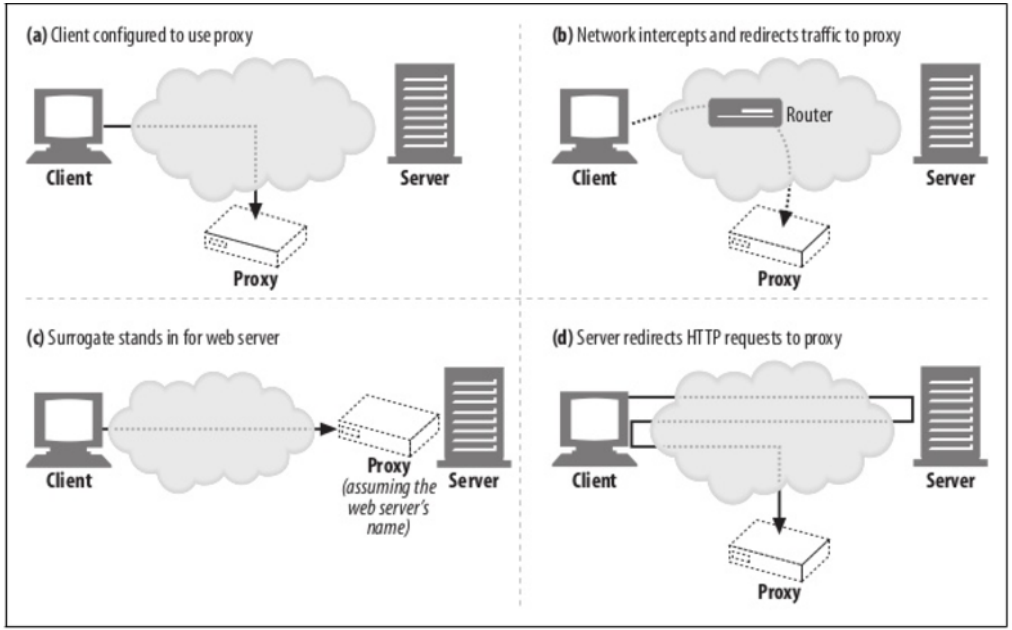
6.3.2 프락시 계층

✓ 동적 부모 선택(라우팅) 로직

- 부하균형
  - 현재 부모들의 작업량 수준에 근거하여 부모 프록시를 고른다.
- 지리적 인접성에 근거한 라우팅
  - 인접한 지역을 담당하는 원 서버에 요청
- 프로토콜/타입 라우팅
  - URI에 근거하여 다른 부모나 원 서버로 라우팅
- 유료 서비스 가입자를 위한 라우팅
  - 대형 캐시나 성능 개선을 위한 압축 엔진으로 라우팅 되어 성능 개선

6.3.3 어떻게 프락시가 트래픽을 처리하는가

- ✓ 클라이언트를 수정한다
  - 수동 · 자동 프락시 설정
- ✓ 네트워크를 수정한다
  - 인터셉트 프락시
- ✓ DNS 이름공간을 수정한다
  - 대리 프락시(리버스 프락시)
- ✓ 웹 서버를 수정한다
  - HTTP 리다이렉션



(포워드 프락시 설정)

6.4 클라이언트 프락시 설정

6.4.1 클라이언트 프락시 설정: 수동

6.4.2 클라이언트 프락시 설정: PAC 파일

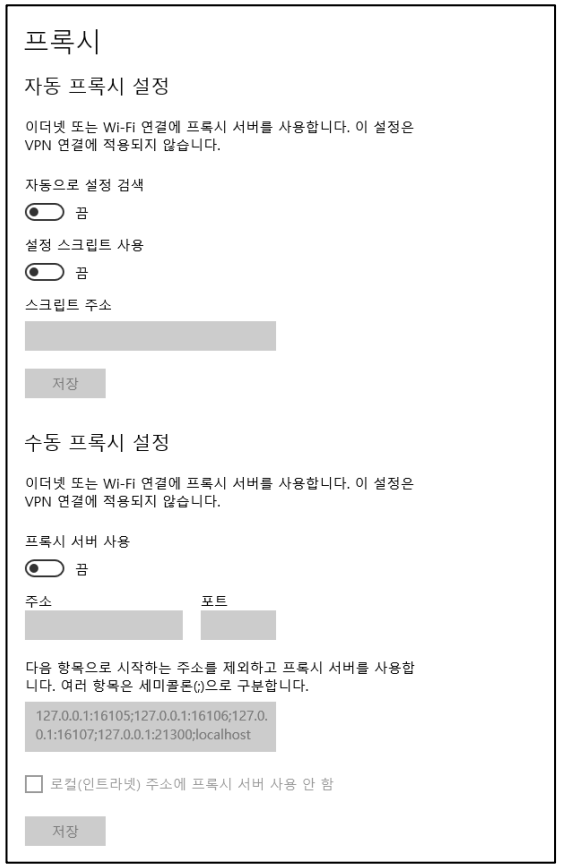
6.4.3 클라이언트 프락시 설정: WPAD

많은 브라우저가 프락시를 설정하는 여러가지 방법을 제공한다 (161p)

6.4.1 클라이언트 프락시 설정: 수동

- ✓ 프락시를 사용하겠다고 명시적으로 설정한다.

- ✓ 단순하지만 유연하지 못함
- ✓ 단 하나의 프락시 서버만 지정할 수 있음
- ✓ 장애시 대체 작동에 대한 지원 x



6.4.2 클라이언트 프락시 설정: PAC 파일

프락시 자동설정 자바스크립트
(Proxy auto-configuration, PAC)

- ✓ 프락시 설정을 상황에 맞게 계산해주는 작은 js 프로그램
- ✓ PAC파일의 URI를 브라우저에 설정
- ✓ 일반적으로 **proxy.pac** 이라는 이름을 갖는다
- ✓ Content-type : application/x-ns-proxy-autoconfig (MIME타입)
- ✓ FindProxyForUrl(url, host) 함수 구현해야함

```
function FindProxyForURL(url, host) {
    if (url.substring(0,5) == "http:") {
        return "PROXY http-proxy.mydomain.com:8080";
    } else if (url.substring(0,4) == "ftp:") {
        return "PROXY ftp-proxy.mydomain.com:8080";
    } else {
        return "DIRECT";
    }
}
```

반환값

- ✓ DIRECT
- ✓ PROXY host:port
- ✓ SOCKS host:port

6.4.3 클라이언트 프락시 설정: WPAD

- ✓ WPAD(Web Porxy Autodiscovery Protocol) 웹 프락시 자동발견 프로토콜
- ✓ PAC파일을 자동으로 찾아주는 알고리즘
- ✓ WPAD는 성공할 때까지 각 기법을 하나씩 시도해본다.
  - 동적 호스트 발견 규약(DHCP)
  - 서비스 위치 규약(SLP)
  - DNS 잘 알려진 호스트 명
  - DNS SRV 레코드
  - DNS TXT 레코드 안의 서비스 URI

6.5 프락시 요청의 미묘한 특징들

6.5.1 프락시 URI는 서버 URI와 다르다

6.5.2 가상 호스팅에서 일어나는 같은 문제

6.5.3 인터셉트 프락시는 부분 URI를 받는다.

6.5.4 프락시는 프락시요청과 서버요청을 모두 다룰 수 있다.

6.5.5 전송중 URI 변경

6.5.6 URI 클라이언트 자동확장과 호스트명 분석

6.5.7 프락시 없는 URI 분석(URI Resolution)

6.5.8 명시적인 프락시를 사용할 때의 URI 분석

6.5.9 인터셉트 프락시를 이용한 URI 분석

6.5.1 프락시 URI는 서버 URI와 다르다

✓ 웹 서버와 웹 프락시에 요청을 보내는 경우, 요청 URI가 서로 다르다.

- 서버 요청 : GET /index.html HTTP/1.0 **(부분 URI)**
- 프락시 요청 : GET http://www.marrys-antiques.com/index.html HTTP/1.0 **(완전한 URI)**

✓ 왜 각각 다른 요청 형식을 갖는가?

현재는 프락시와 서버 요청 모두에 대해
HTTP/1.1은 현재 서버들이 **완전한 URI**를 다룰것을 요구하고 있다.

- 원래 HTTP 설계에서 클라이언트는 단일서버와 통신했음
- 프락시가 부상하면서, 목적지 서버와 커넥션을 맺어야 하므로 해당 서버의 이름을 알 필요가 있음
- 프락시 기반 게이트웨이의 경우 URI의 스킴이 필요

✓ 명시적으로 프락시가 설정된 클라이언트는 해당하는 요청을 보내게 된다.

6.5.2 가상 호스팅에서 일어나는 같은 문제

✓ 프락시의 '스킴/호스트/포트번호 누락' 문제

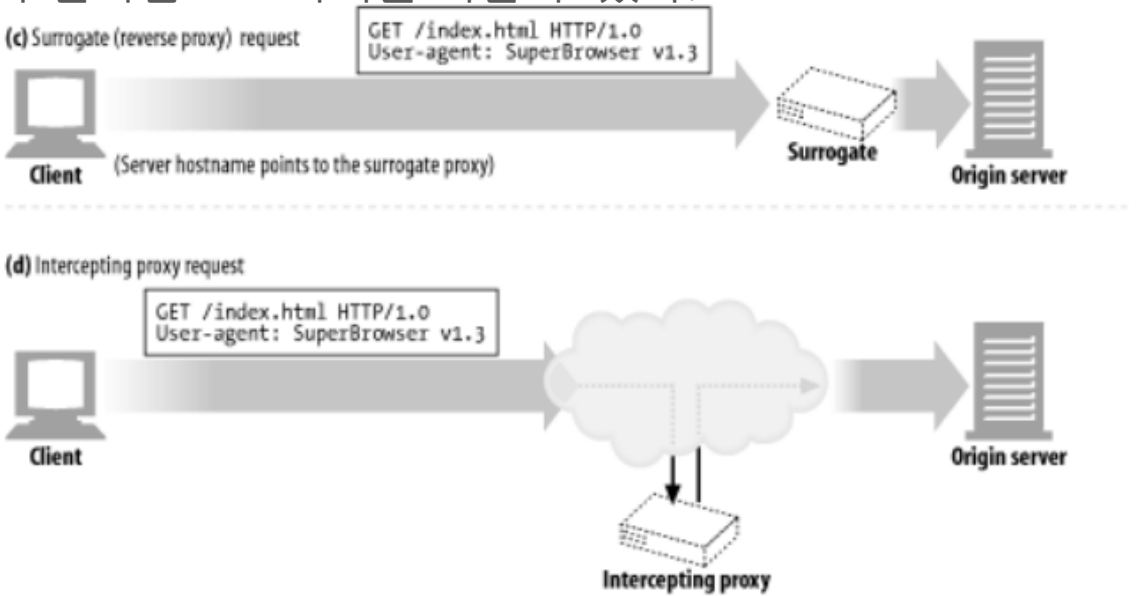
- 가상 호스팅 웹 서버는 여러 웹 사이트가 같은 물리적 웹 서버를 공유

✓ 해결 방법

- 명시적인 프락시는 요청 메시지가 **완전한 URI**를 갖도록 함
- 가상으로 호스팅 되는 웹 서버는 Host 헤더를 요구 (호스트와 포트번호 정보가 담겨 있음)

6.5.3 인터셉트 프락시는 부분 URI를 받는다.

- ✓ 클라이언트는 자신이 프락시와 대화하고 있음을 항상 알고 있지 않다.
- ✓ 클라이언트가 프락시를 사용한다고 설정되어 있지 않더라도,
- ✓ 여전히 대리 프락시나 인터셉트 프락시를 지날 수 있다.



6.5.4 프락시는 프락시요청과 서버요청을 모두 다룰 수 있다.

✓ 다목적 프락시 서버는 완전한URI와 부분URI를 모두 지원해야 한다.

✓ 프락시가 URI를 사용하는 규칙

- **완전한 URI** → just 사용
- **부분 URI + host헤더** → host헤더를 이용해 **`원서버 이름:포트번호`** 알아낸다.
- **부분 URI** → 다음의 방법으로 원 서버를 알아낸다.
  - 대리 프락시 → 프락시에 **`원서버 이름: 포트번호`** 가 설정되어있다.
  - 인터셉트 프락시가 가로챘던 트래픽 + 원 IP주소와 포트번호를 사용할 수 있도록 설정 → 그 IP주소와 포트번호를 사용할 수 있다.
- **모두 실패** → 에러메시지

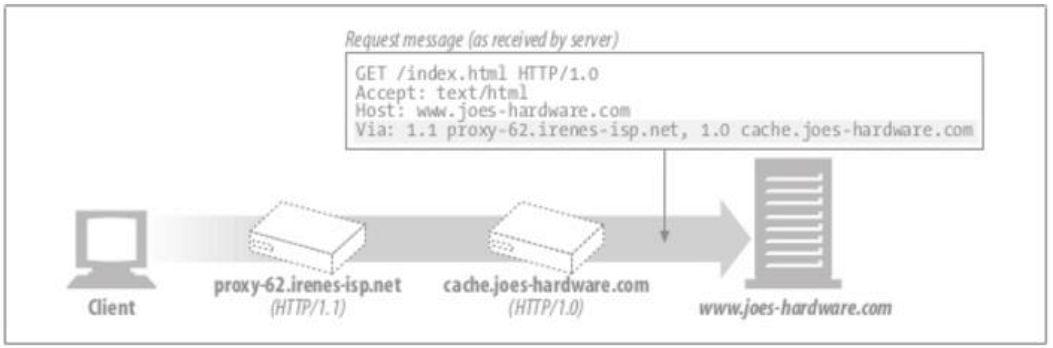
6.5.5 전송중 URI 변경

- ✓ 프락시 서버는 요청 URI의 변경에 매우 신경을 써야한다.
  - 무해해 보이는 사소한 URI 변경이라도 다운스트림 서버와 상호 운용성 문제를 일으킬 수 있다.
- ✓ 일반적인 인터셉트 프락가 URI를 전달할 때 절대 경로를 고쳐 쓰는 것 금지
  - 유일한 예외 : 빈 경로 → '/' 로 변경

6.5.6 URI 클라이언트 자동확장과 호스트명 분석

- ✓ 브라우저는 프락시 존재 여부에 따라 요청 URI를 다르게 분석한다.

- ✓ 프락시가 없다면 사용자가 타이핑한 URI를 가지고 그에 대응하는 IP 주소를 찾는다.
- ✓ 만약 호스트명이 발견되면 그에 대응하는 IP 주소들을 연결 할 때까지 시도한다.
- ✓ 브라우저들은 자동화된 호스트 명 확장을 시도한다.
    - www. {입력URI} .com
    - 서드파티 사이트로 넘김 (오타교정 시도)



6.6 메시지 추적

프락시가 점점 더 흔해지면서
프락시를 넘나드는 메시지의 흐름을 추적하고
문제점을 찾아내는 것도 필요한 일이 되었다

6.6.1 via 헤더

6.6.2 TRACE메서드

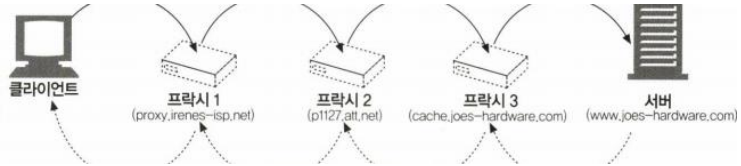
6.06 메시지 추적

```
Via: [ <protocol-name> "/" ] <protocol-version> <host> [ ":" <port> ]
or
Via: [ <protocol-name> "/" ] <protocol-version> <pseudonym>
```

via : FTP/1.1 proxy.irenes-isp.net, 1.1 p1127.att.net

✓ Via 헤더

- 메시지가 지나는 각 중간노드의 정보를 나열
- 모든 메시지 발송자들의 프로토콜을 다루는 능력을 알아보기 위해 사용
- 응답 Via헤더는 요청 Via헤더의 반대
- Server헤더와 via헤더



MDN Web Docs moz://a ▸ Technologies ▸ Referen

개발자를 위한 웹 기술 › HTTP › HTTP 헤더 › Via

이 페이지는 영어로부터 커뮤니티에 의하여 번역되었습니다. MDN Web Docs

Table of contents

문법
지시자

Via

Via 헤더는 요청헤더와 응답헤

Headers Preview Response Initiator Timing Cookies

cloud media.stage.mdn.mozit.cloud interactive-examples.mdn.mozilla.net wikipe
s.cdn.mozilla.net; worker-src 'none';
date: Mon, 28 Feb 2022 02:58:06 GMT
etag: W/"f5008001600f4eae11de8a371f1b50a8"
server: AmazonS3
strict-transport-security: max-age=63072000
vary: Accept-Encoding
via: 1.1 292f247ccfcda16bfbba015355e8351a.cloudfront.net (CloudFront)
x-amz-cf-id: 9PBCGedAGktzuOXNuooMgA-JFOHQmLTdN7W1syFteyuL_smLsJp2zw==

6.06 메시지 추적

- ✓ TRACE 메서드
  - 프락시가 복잡해 질수록 상호운용성 문제가 증가
  - 프락시 흐름을 디버깅 하는데 사용
    - 네트워크를 통해 홉에서 홉으로 전달될 때 메시지의 내용이 어떻게 변하는지 관찰
  - TRACE 응답 content-type: message/http
  - 널리 구현되지 않은 메서드..

6.7 프락시 인증

사용자가 유효한 접근 권한 자격을 프락시에 제출하지 않는 한 콘텐츠에 대한 요청 차단하는 메커니즘(178p)

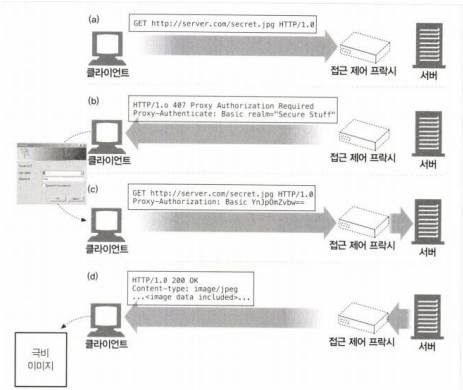
6.07 프락시 인증

✓ 프록시 - 제한된 콘텐츠에 대한 요청이 도착

- 407 Proxy Authorization Required 상태코드 + Proxy-Authenticate 헤더필드와 함께 반환

✓ 클라이언트 - 407 응답 수신

- 요구되는 자격 수집
- 수집시 Proxy-Authenticate 헤더필드에 담아 재 요청



✓ 프락시 연쇄상에 인증 참여 프락시가 여러개 있을때 일반적으로 잘 동작하지 않는다

6.8 프락시 상호운용성

6.8.1 지원하지 않는 헤더와 메서드 다루기

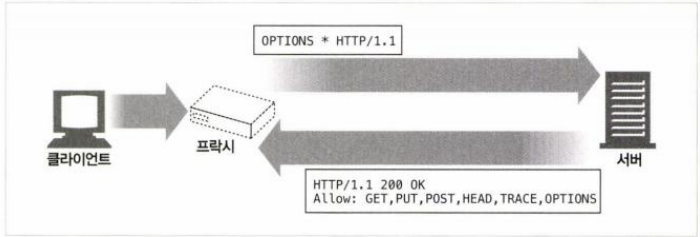
6.8.2 OPTIONS: 어떤 기능을 지원하는지 알아보기

6.8.3 Allow 헤더

클라이언트, 서버, 프락시는 HTTP명세의 **여러버전**에 의해 **여러벤더**에 의해 만들어진다. (180p)
(제각각 다른 기능, 제각각 다른 버그 보유)

6.08 프락시 상호운용성

- ✓ 지원하지 않는 헤더와 메서드 다루기
  - 이해하지 못하는 헤더
  - 친숙하지 않은 메서드
- ✓ OPTIONS 메서드
  - 특정 리소스가 어떤 기능을 지원하는지 알게 해준다
  - OPTIONS * HTTP/1.1
- ✓ Allow 헤더
  - 요청 자원에 대해 지원되는 메서드들/서버가 지원하는 모든 메서드 열거
  - Allow: GET, HEAD, PUT




```java
@Configuration
public class WebMvcConfig implements WebMvcConfigurer {

    private final long MAX_AGE_SECS = 3600;

    @Override
    public void addCorsMappings(CorsRegistry registry) {
        registry.addMapping("/**")
                .allowedOrigins(
                        "https://app.kimyooo.com",
                        "http://localhost:3000"
                )
                .allowedMethods("GET", "POST", "PUT", "PATCH", "DELETE", "OPTIONS")
                .allowedHeaders("*")
                .allowCredentials(true)
                .maxAge(MAX_AGE_SECS);
    }
}
```

QnA

출처

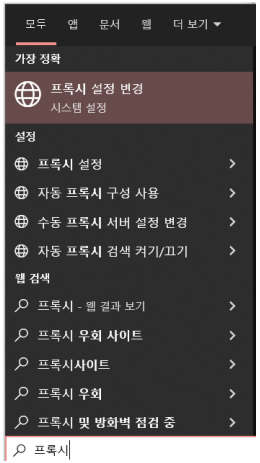
- ✓ https://sujinhope.github.io/2021/06/13/Network-%ED%94%84%EB%A1%9D%EC%8B%9C(Proxy)%EB%9E%80,-Forward-Proxy%EC%99%80-Reverse-Proxy.html

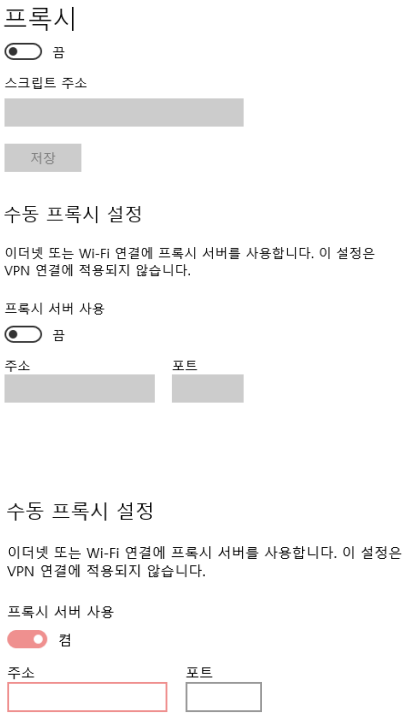
- ✓ https://class101.net/classes/5fc4a3b3fc231b000d856415/challenge
- ✓ HTTP완벽가이드

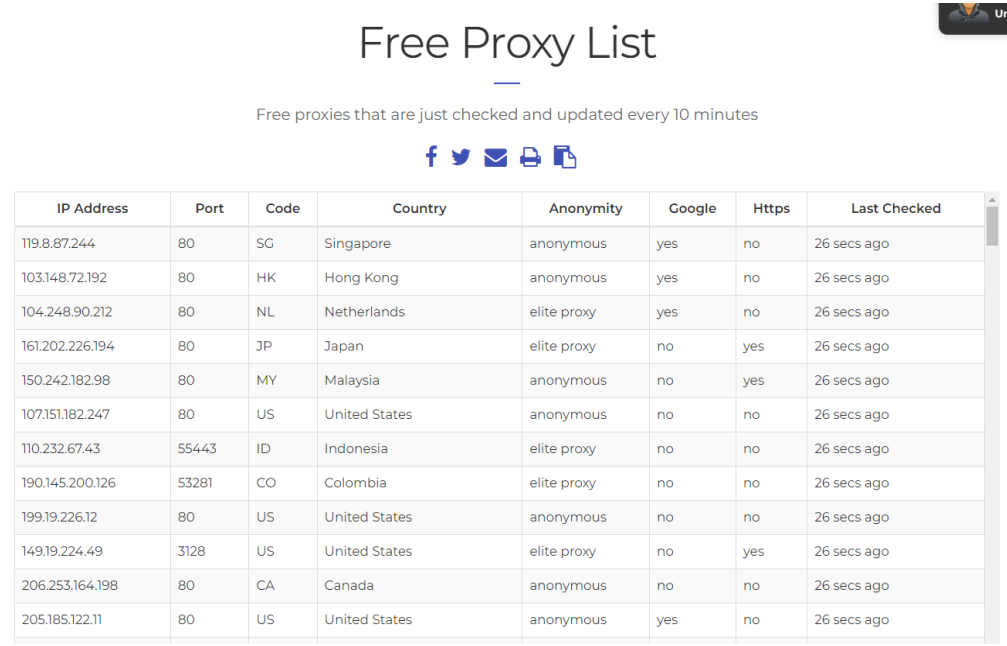
6.05 프락시 요청의 미묘한 특징들

- ✓ 프락시 요청의 URI는 서버 요청과 어떻게 다른가?
- ✓ 인터셉트 프락시와 리버스 프락시는 어떻게 서버 호스트 정보를 알아내기 어렵게 만드는가
- ✓ URI 수정에 대한 규칙
- ✓ 프락시는 브라우저의 똑똑한 URI 자동완성이나 호스트명
- ✓ 확장기능에 어떻게 영향을 주는가

6.02+ 간단하게 프록시 사용해보기







https://free-proxy-list.net/