2025.09.12

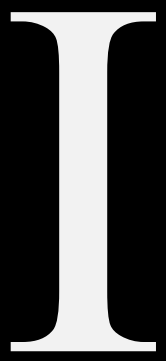
Web Application Tech

가디언 웹 보안 세미나 1

임준서

HTTP/HTTPS

Requests



HTTP

Request

```
GET /auth/488/YourDetails.ashx?uid=129 HTTP/1.1
Accept: application/x-ms-application, image/jpeg, application/xaml+xml,
image/gif, image/pjpeg, application/x-ms-xbap, application/x-shockwave-
flash, */*
Referer: https://mdsec.net/auth/488/Home.ashx
Accept-Language: en-GB
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64;
Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR
3.0.30729; .NET4.0C; InfoPath.3; .NET4.0E; FDM; .NET CLR 1.1.4322)
Accept-Encoding: gzip, deflate
Host: mdsec.net
Connection: Keep-Alive
Cookie: SessionId=5B70C71F3FD4968935CDB6682E545476
```

Hyper Text Transfer Protocol

개쩌는 텍스트 전송 규약

HTTP

Request

```
GET /auth/488/YourDetails.ashx?uid=129 HTTP/1.1
Accept: application/x-ms-application, image/jpeg, application/xaml+xml,
image/gif, image/pjpeg, application/x-ms-xbap, application/x-shockwave-
flash, */*
Referer: https://mdsec.net/auth/488/Home.ashx
Accept-Language: en-GB
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64;
Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR
3.0.30729; .NET4.0C; InfoPath.3; .NET4.0E; FDM; .NET CLR 1.1.4322)
Accept-Encoding: gzip, deflate
Host: mdsec.net
Connection: Keep-Alive
Cookie: SessionId=5B70C71F3FD4968935CDB6682E545476
```

HTTP

Response

```
HTTP/1.1 200 OK
Date: Tue, 19 Apr 2011 09:23:32 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
Set-Cookie: tracking=tI8rk7joMx44S2Uu85nSWc
X-AspNet-Version: 2.0.50727
Cache-Control: no-cache
Pragma: no-cache
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 1067

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://
www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"><html xmlns="http://
www.w3.org/1999/xhtml" ><head><title>Your details</title>
...
```

HTTP

Response

```
HTTP/1.1 200 OK
Date: Tue, 19 Apr 2011 09:23:32 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
Set-Cookie: tracking=tI8rk7joMx44S2Uu85nSWc
X-AspNet-Version: 2.0.50727
Cache-Control: no-cache
Pragma: no-cache
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 1067
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://
www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"><html xmlns="http://
www.w3.org/1999/xhtml" ><head><title>Your details</title>
...
```

HTTP

Methods

GET

데이터 주세요

POST

뭐 해주세요

PUT/DELTE

자료 수정 / 삭제할래요

Note.
GET, POST는 거의 저 용도 고정
PUT/DELETE는 안쓰는 곳도 많음

Note.
이것들 말고도 HEAD, TRACE 등
많지만, 거의 안 쓰임

URL

scheme://⟨user⟩:⟨password⟩@⟨host⟩:⟨port⟩/⟨url-path⟩

https://google.com/

https 프로토콜로 youtube 호스트로 접속

ftp://user@host/foo/bar.txt

Ftp란 프로토콜로 user로 host에 접속하여 foo path 아래 bar.txt 접근

https://google.com@naver.com

IPV6 http://[::1]/index.html

HTTP

Headers - Request

Accept

받을 데이터 양식

Authorization

Auth 관련 데이터

Cookie

쿠키 (곧 설명)

HOST

URL의 HOST 이름

Referer

Request의 원래 출신지

User-Agent

내가 누군지

HTTP

Headers - Response

Location

어디로 가주세요

Server

서버이름

Set-Cookie

쿠키 발급

Expires

캐싱 기간

COOP

이 페이지가 다른 도메인(origin)을 참조 가능?

ACAO

다른 도메인이 서버 자원을 호출 가능?

Cross Origin Opener Policy. 나중에 다룸

Access Control Allow Origin. 나중에 다룸

HTTP

Headers

Q. 다 외워야 하나요?

A. 아니지만 굉장히 자주 보게 된다

강조한 것만 알아두면 문제 없음

HTTP

Cookie

Set-Cookie

Set-Cookie: sessionId=a1; HttpOnly; Path=/; Secure; SameSite=Lax

HttpOnly/Secure

Js는 쿠키접근 불가 / https만 허용

Domain / path

Domain/path 아래에서만 유효

HTTP

Status Codes

1xx: hold on

2xx: here you go

3xx: go away

4xx: you fucked up

5xx: I fucked up

-via @abt_programming

HTTP

Status Codes

1xx: hold on	나중에 줄게
2xx: here you go	정상동작
3xx: go away	Redirect
4xx: you fucked up	잘못된 접근
5xx: I fucked up	잘못된 서버

-via @abt_programming

HTTPS

HTTP Secure

HTTP에 SSL이 추가된 구조

Secure Socket Layer

DNS 스푸핑이 있더라도 비교적 안전

HTTPS

간단한 동작 방식

1. SSL key 교환

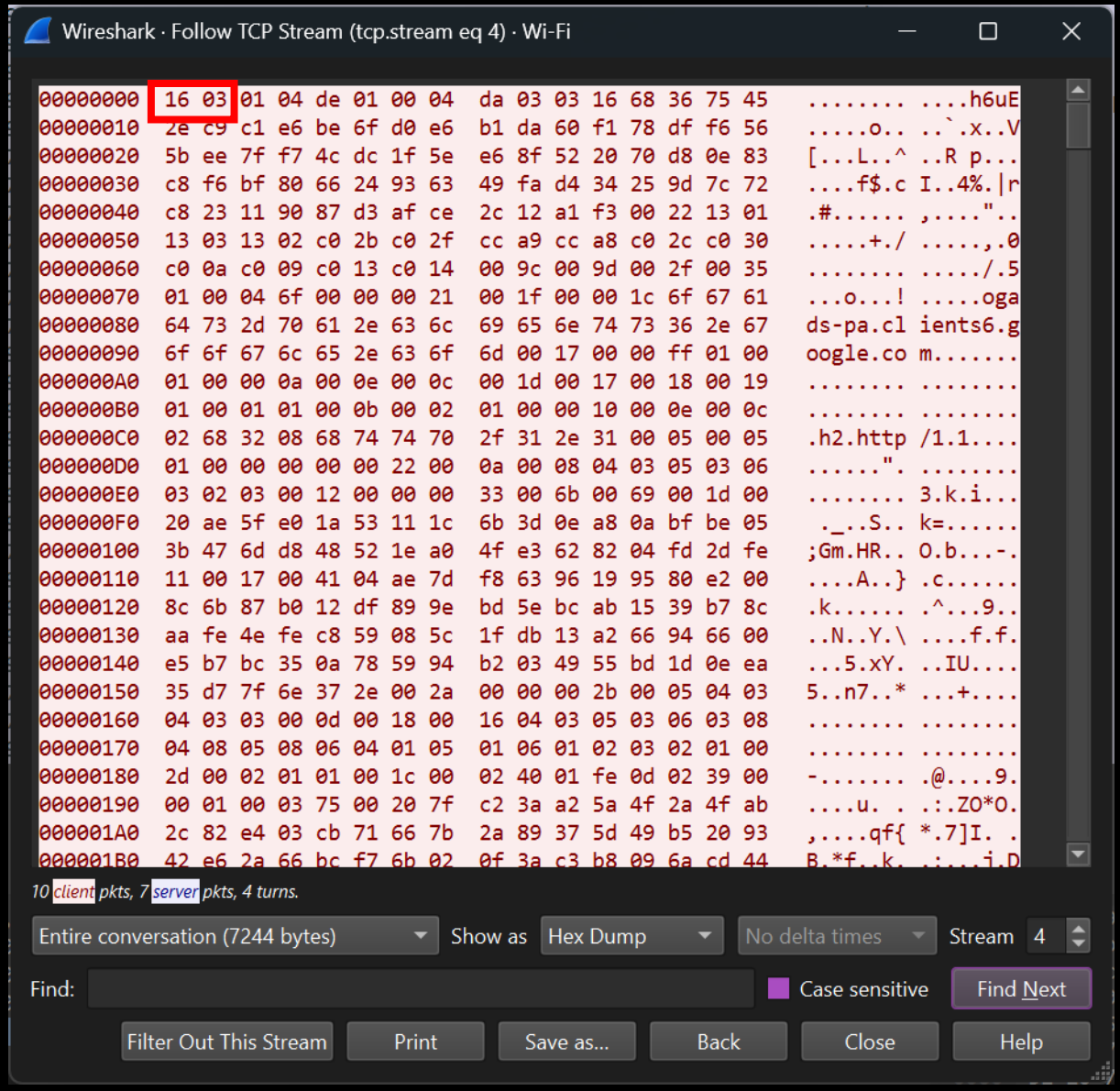
컴퓨터 어딘가 sslkeylogfile 이 있음

2. 암호화 후 통신

모든 HTTP 를 암호화

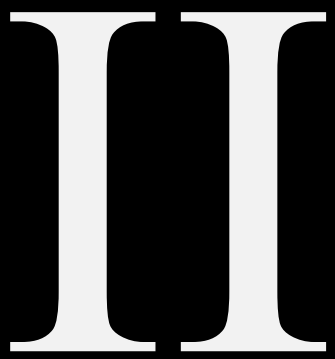
HTTPS

간단한 동작 방식



BuildingBlocks

Gateway / WA / Containers



Nginx

Web Server + reverse proxy + …

Web Server

정적인 파일들 (이미지, HTML, CSS, JS)을 빠르게 뿌려주는 역할

Reverse Proxy

들어오는 요청을 받아서 뒤쪽의 여러 서버에 나눠주는 Load Balancer 역할

Gateway

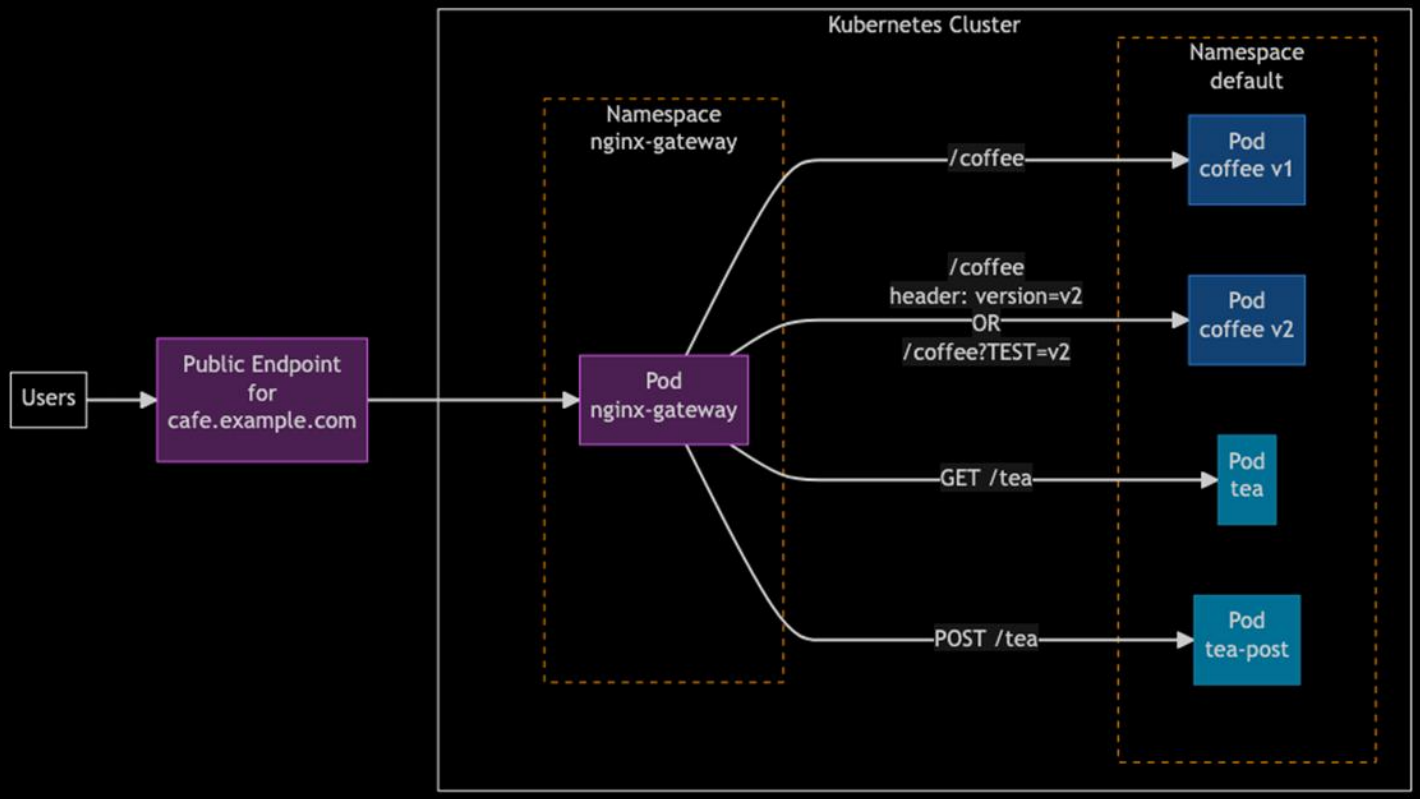
SSL(https 암호화), 캐싱, 압축, 보안 필터링 등 미리 처리

[Engine X]

Nginx 말고 Apache[아파치]도 많이 쓴다

Nginx

Diagram



FrontEnd

Client-Side + Server-Side Rendered

HTML, CSS, JS를 전달하는 곳

웹 3요소

UI/UX

브라우저에서 실행

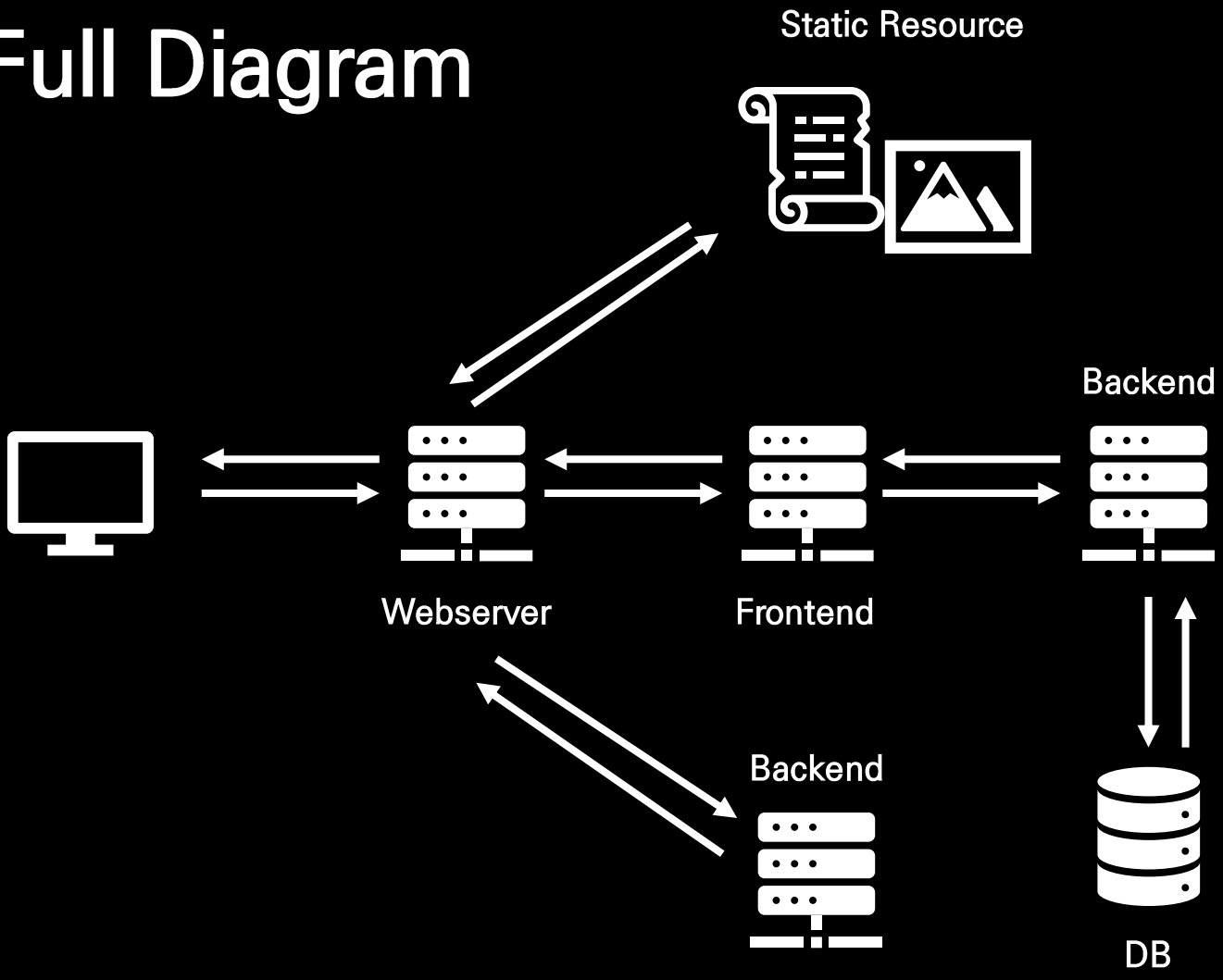
BackEnd

Server Logic

서버 로직 처리

DB 연동

Full Diagram



1cm-1cm 떨어진 주석 12px

1cm-1cm 떨어진 주석 12px

사용 언어? 프레임워크?

js, python, java, php, …

어차피 비슷하기에 그때그때 문법을 알아두자.

DB

SQLs (mySQL, pSQL, SQLight), noSQLs (MongoDB, redis)

Next, express, flask, spring, …

구조가 다 거기서 거기

1cm-1cm 떨어진 주석 12px

1cm-1cm 떨어진 주석 12px

Container

Docker

VM *같은거*

정확히는 가상화 기술 종류에 Containers와 VM이 있는 것. VM이 더 무거움

어디서든 동일한 환경 제공

OS level 의 가상화라서 OS가 달라도 괜찮음

OS 가상화이기에 하드웨어가 다르면 작동 안 함

ex – CISC에서 빌드한 컨테이너를 AMD64에서 사용

Docker

Container

어떤 OS에서 동작할거에요?

이후 비어있는 윈도우/리눅스/이것저것 생성

만든 OS에서 어떤 파일들을 쓸거에요?

“비어있는” OS

만든 OS의 어떤 포트를 쓸거에요?

“비어있는” OS

“.dockerfile” 이라는 이름의 파일로 작성한다

```dockerfile
# Use an LTS base image for stability and security updates
FROM node:18-alpine AS base

# Create and use non-root user for better security
RUN addgroup -S app && adduser -S app -G app
WORKDIR /app

# Install only production dependencies using cached layer
# 1) copy package manifests first to leverage build cache
COPY package*.json ./
RUN npm ci --omit=dev

# 2) copy the rest of the source code
COPY . .

# Document the port the app listens on
EXPOSE 3000

# Drop privileges
USER app

# Run the server
CMD ["node", "server.js"]

```

요?

```
# Build an image from Dockerfile in the current directory
docker build -t myapp:1.0 .

# Run a container (publish host port 3000 -> container 3000)
docker run -d --name myapp -p 3000:3000 myapp:1.0

# See running containers
docker ps

# Tail logs (Ctrl+C to stop tailing)
docker logs -f myapp

# Shell into a running container (useful for debugging)
docker exec -it myapp sh   # or bash if available

# Stop and remove the container
docker stop myapp && docker rm myapp

# Remove dangling images and reclaim space
docker image prune -f
docker system df

```

Docker

Container

어떤 OS에서 동작할거에요?

이후 비어있는 윈도우/리눅스/이것저것 생성

만든 OS에서 어떤 파일들을 쓸거에요?

“비어있는” OS

만든 OS의 어떤 포트를 쓸거에요?

“비어있는” OS

“.dockerfile” 이라는 이름의 파일로 작성한다

Docker-compose

Containers

Dockerfile 여러 개 한 번에 관리

일단 여러 개를 하나로 묶어준다

만든 OS에서 어떤 파일들을 어떻게 쓸거에요?

.dockerfile‘들’ 실행할 때 매번 cd 치기 귀찮음

만든 OS의 어떤 포트를 어떻게 쓸거에요?

.dockerfile‘들’ 실행할 때 –p 8080:3000 치기 귀찮음

“docker-compose.yml” 이라는 이름의 파일로 작성한다

D

Con

```yaml
version: "3.9"
services:
  web:
    build: .
    ports:
      - "8080:3000"     # host 8080 → container 3000
    volumes:
      - ./data:/app/data
    environment:
      - NODE_ENV=production
    depends_on:
      - db              # db first

  db:
    image: postgres:16
    environment:
      - POSTGRES_USER=appuser
      - POSTGRES_PASSWORD=secret
      - POSTGRES_DB=mydb
    volumes:
      - dbdata:/var/lib/postgresql/data

volumes:
  dbdata:
```

에요?

?

“docker-co

Docker-compose

Containers

만약 .docker-compose.yml 이 보인다?

아묻따 `docker compose up -d`

종료하고 싶으면?

Gpt한테 물어보세요 (`docker compose down` || `docker stop ~~`)

"docker-compose.yml" 이라는 이름의 파일로 작성한다

2025.09.12

질문받습니다

질문이 있다면 하십시오

임준서

2.5cm-3.5cm 떨어진 제목 36px

제목 하단의 부제목 18px

3.5cm 떨어진 내용 1 32px

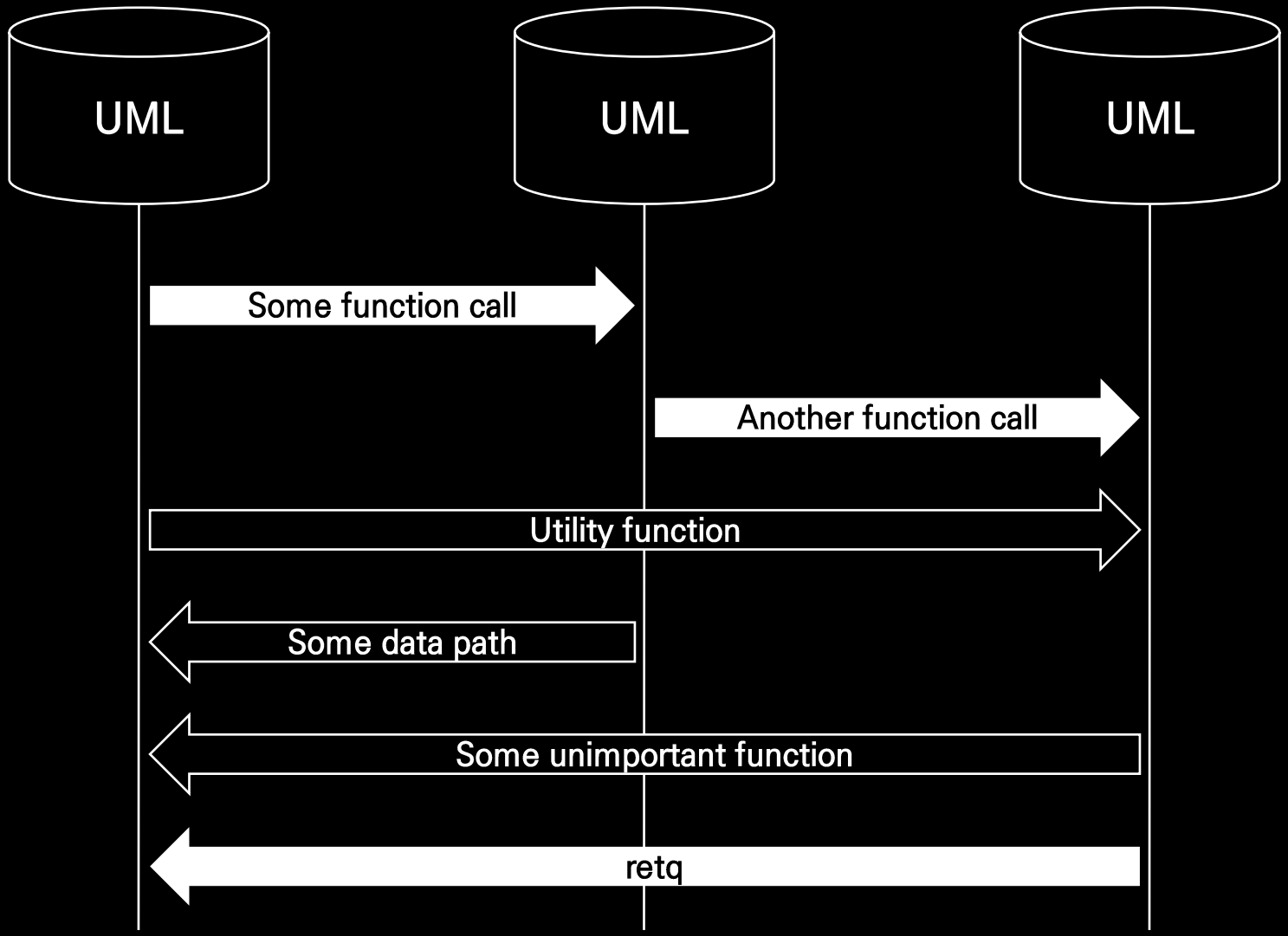
```
Git init
Git status
Git add text.txt
Git add .
Git commit
Ctrl+C
Git commit -m “genesis”
Git log
Git log --oneline
```

```
Git add .
Git reset .
Git commit -m “add README”
Git log --oneline -n 3
Git commit –a –m “hello”
```

1cm-1cm 떨어진 주석 12px

1cm-1cm 떨어진 주석 12px

중심에서 0.3cm 떨어진 소속 18px



중심에서 0.3cm 떨어진 소속 18px