

공인인증서 파밍사건에 대한 분석 및 파싱 코드 구현 보고서

|  |
| --- |
| Digital Forensic Track 2019-01-18 |
| 김태우 |

INDEX

[1. 개요 3](#_Toc535549323)

[1.1 사건 경위 3](#_Toc535549324)

[1.2 파싱 코드 구현을 위한 환경 3](#_Toc535549325)

[2. 코드 구현 5](#_Toc535549326)

[2.1 MySQL 구성 5](#_Toc535549327)

[2.2 전체 코드의 구성 6](#_Toc535549328)

[2.2.1 Main.java 7](#_Toc535549329)

[2.2.2 CrawlerFromSuspectURL.java 7](#_Toc535549330)

[2.2.3 GetFileFromURLWithProxy.java 9](#_Toc535549331)

[2.2.4 ParsingAndInesertMysql.java 14](#_Toc535549332)

[3. 코드 실행 20](#_Toc535549333)

[3.1 용의자의 사이트에서 HTML소스 파싱 및 TXT파일로 출력 20](#_Toc535549334)

[3.2 인증서 다운로드 21](#_Toc535549335)

[3.2 인증서 파싱 및 DB삽입 22](#_Toc535549336)

[4. DB 분석 23](#_Toc535549337)

[5. DB데이터 CSV로 추출 25](#_Toc535549338)

[6. 결론 및 후기 26](#_Toc535549339)

# 1. 개요

## 1.1 사건 경위

피의자는 한국인(해외거주민 포함)을 상대로 공인인증서를 탈취하는 악성코드를 유포하였습니다.

악성코드에 감염된 피해자의 PC에 저장된 공인인증서는 [IP주소].zip 파일의 형태로 해외 서버(http://fl0ckfl0ck.work/cert)에 업로드되어 있습니다.

다행히 디렉터리 리스팅 취약점으로 인하여 공인인증서 들은 노출되어 있습니다.

빠른 시일내에 폐기처분을 해야합니다.

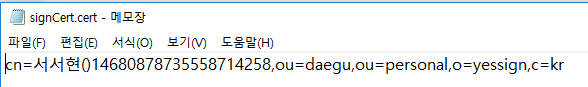


그림 1. 공인인증서의 내용

- 팀장 지시사항

1. 피해자의 일련번호 - 피해시각(서버에 피해자의 공인인증서가 업로드된 시간) - 피해자의 이름 - 은행명 - 계좌번호 - IP주소 - 피해자의 현재 소재지(국가만)를 데이터베이스(mysql)에 저장하시오.

2. 은행별 유출된 공인인증서의 갯수를 계산하시오.

## 1.2 파싱 코드 구현을 위한 환경

표 1. 코드 구현 PC 환경표

|  |  |
| --- | --- |
| OS | Windows 10 Pro x64 17134 |
| Ram | 8GB |
| SSD | 256GB |
| HDD | 2TB |

표 2. 개발 언어 및 사용 도구

|  |  |
| --- | --- |
| 사용언어 | Java |
| IDE | Oxygen.3a Release (4.7.3a) |
| DB | MySQL 8.0.13 |

# 2. 코드 구현

## 2.1 MySQL 구성

DB명 certdb

다음 DDL을 통해 테이블을 구성하였다.

CREATE TABLE cert (

id INT(15) NOT NULL PRIMARY KEY,

time DATETIME,

name VARCHAR(10),

bank VARCHAR(10),

account VARCHAR(30),

ip VARCHAR(20),

country VARCHAR(5)

);

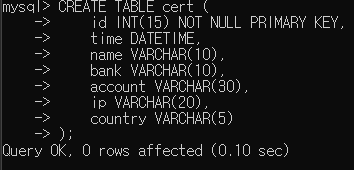


그림 2. MySQL cert table의 생성

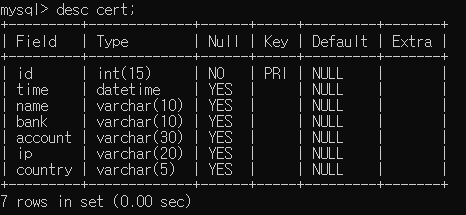


그림 3. MySQL cert 테이블의 구성

## 2.2 전체 코드의 구성

다음 [그림 7]은 전체 코드의 구성을 나타내는 UML이다.

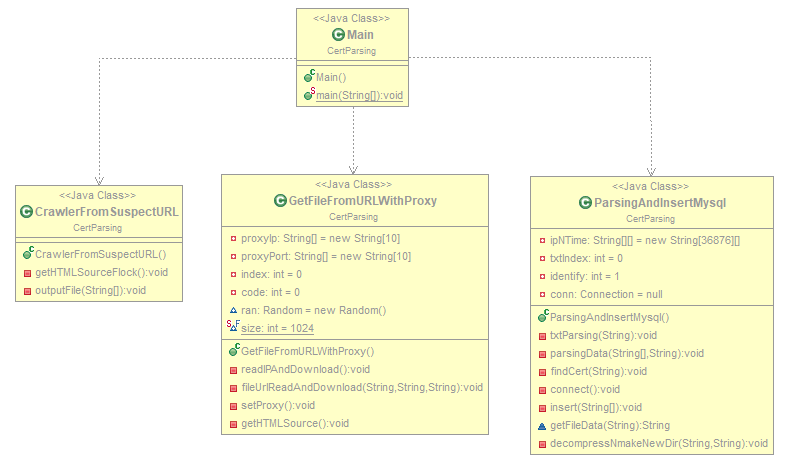


그림 4. 전체 코드의 UML

### 2.2.1 Main.java

메인 함수

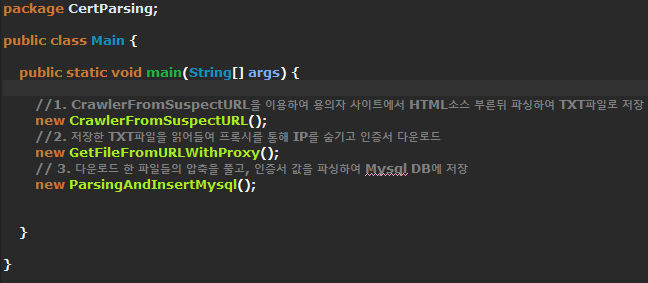


그림 5. 메인 함수

### 2.2.2 CrawlerFromSuspectURL.java

용의자의 사이트에서 HTML소스를 부른 뒤 파싱하여 TXT파일로 저장하는 클래스

#### - 생성자

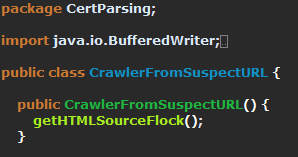


그림 6. CrawlerFromSuspectURL클래스의 생성자

#### - getHTMLSourceFlock 메서드

용의자의 사이트로부터 HTML소스를 불러서 파일 정보와 시간 데이터만 파싱하는 메서드

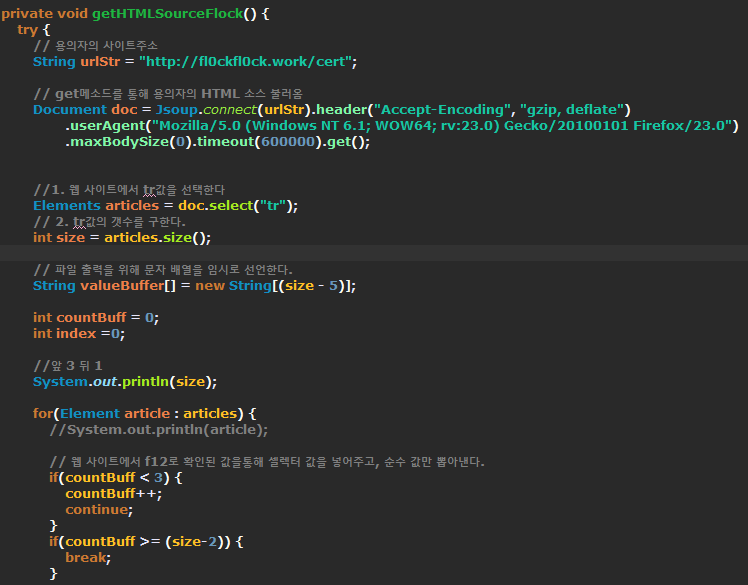


그림 7. getHTMLSourceFlock 메서드 1

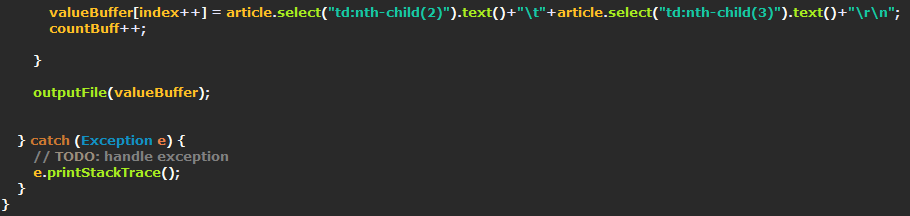


그림 8. getHTMLSourceFlock 메서드 2

#### - outputFile 메서드

파싱한 HTML소스를 문자열로 받아 파일로 저장하는 메서드

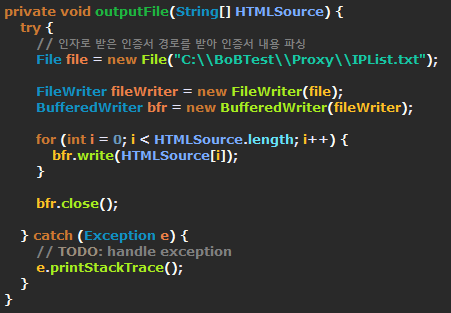


그림 9. outputFile 메서드

### 2.2.3 GetFileFromURLWithProxy.java

저장한 TXT파일을 읽어, 다운받을 파일을 파악하고, 프록시를 설정하여 다운받는 수사관 PC의 IP정보를 숨겨 인증서를 다운로드 받는 클래스

#### - 멤버 변수 및 생성자

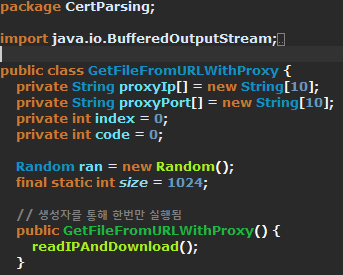


그림 10. GetFileFromURLWithProxy 클래스의 멤버변수와 생성자

#### - readIPAndDownload 메서드

저장된 TXT파일을 파싱하여 파일을 다운받는 메서드

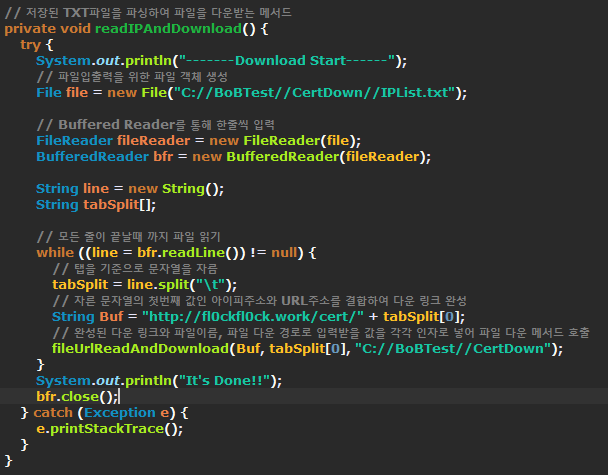


그림 11. readIPAndDownload 메서드

#### - fileUrlReadAndDownload 메서드

파일의 URL주소와 파일이름, 다운로드 경로를 받아 파일을 다운받는 메서드

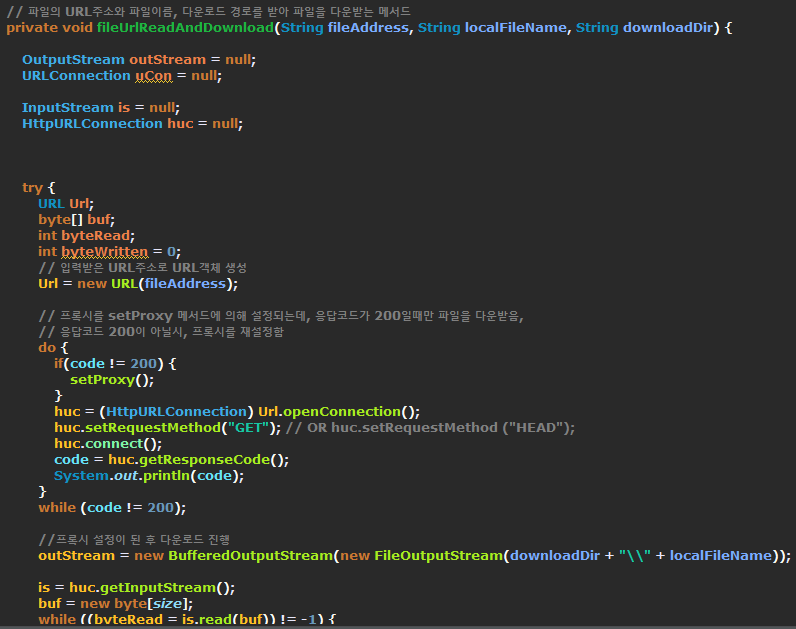


그림 12. fileUrlReadAndDownload 메서드 1

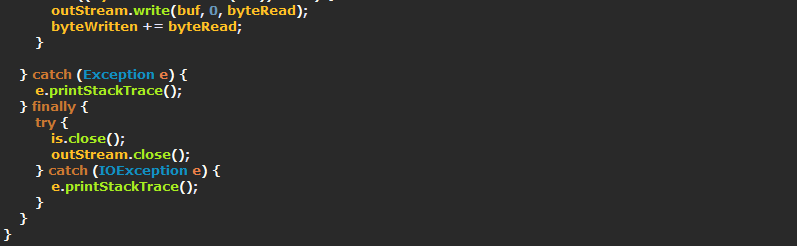


그림 13. fileUrlReadAndDownload 메서드 2

#### - setProxy 메서드

무료 프록시 사이트를 파싱하여 프록시를 설정하는 메서드



그림 14. setProxy 메서드

#### - getHTMLSource 메서드

무료 프록시 사이트인 <https://free-proxy-list.net/> 에서 IP주소와 Port주소를 파싱하여 배열에 저장하는 메서드

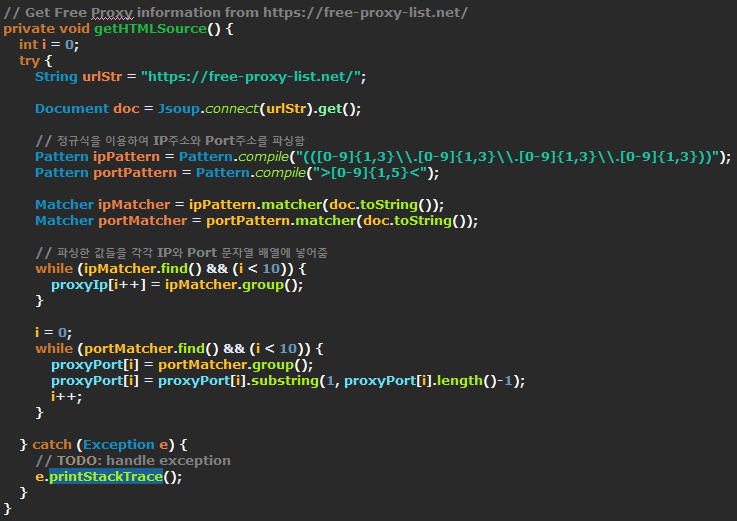


그림 15. getHTMLSource 메서드

### 2.2.4 ParsingAndInesertMysql.java

다운받은 데이터를 파싱하고 DB에 저장하는 클래스

#### - 멤버 변수 및 생성자

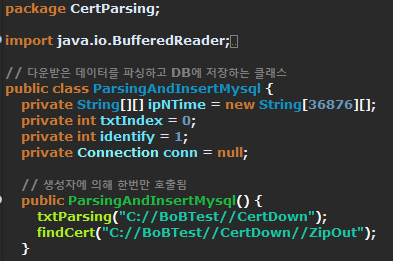


그림 16. ParsingAndInesertMysql 클래스의 멤버 변수 및 생성자

#### - txtParsing 메서드

주어진 txt파일을 읽고 정규식을 이용하여 IP와 시간정보를 뽑아내는 메서드

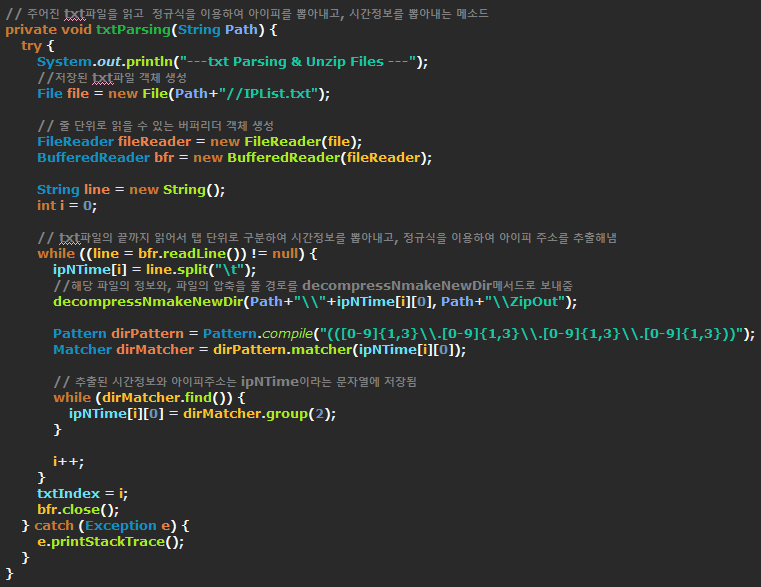


그림 17. txtParsing 메서드

#### - parsingData 메서드

파싱한 txt파일정보(IP, 시간)와, 파싱한 cert데이터를 가지고 DB에 삽입

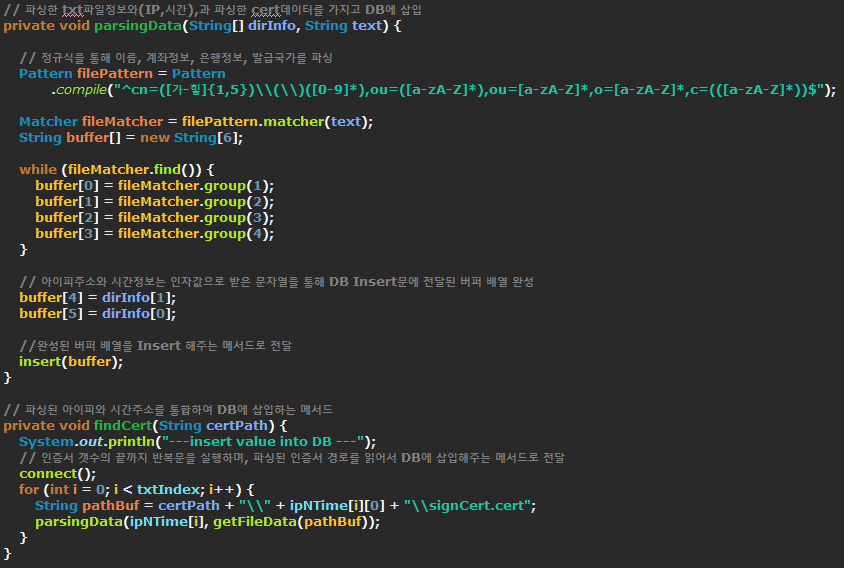


그림 18. parsingData 메서드

#### - findCert 메서드

파싱된 IP와 시간정보를 통합하여 DB에 삽입하는 메서드

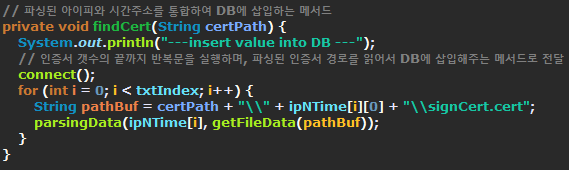


그림 19. findCert 메서드

#### - connect 메서드

DB의 연결을 성립하기 위한 메서드

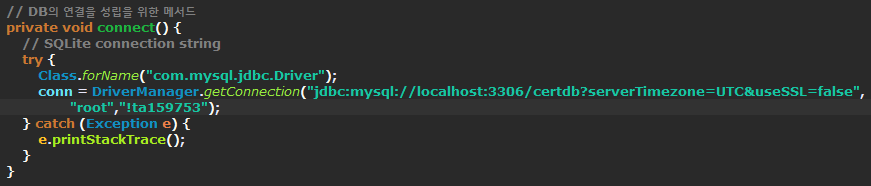


그림 20. connect 메서드

#### - insert 메서드

DB에 값을 삽입하기위한 메서드

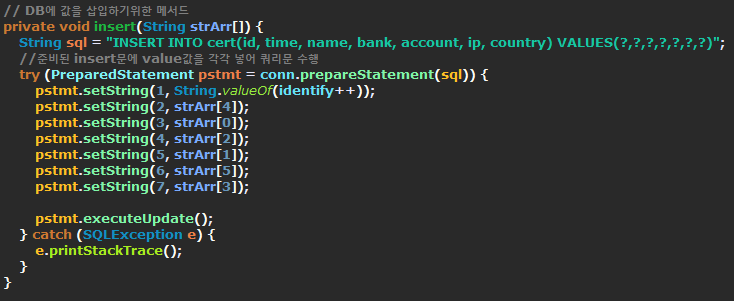


그림 21. insert 메서드

#### - getFileData 메서드

인증서 파일을 읽어들이기 위한 메서드

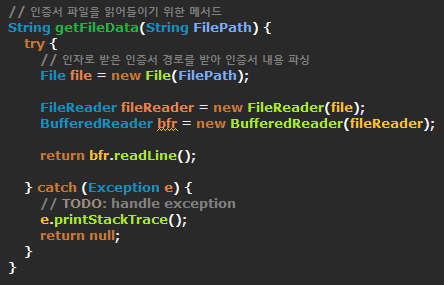


그림 22. getFileData 메서드

#### - decompressNmakeNewDir 메서드

파일명을 받아 새로운 디렉터리를 생성하고, 해당 디렉터리에 압축을 푸는 메서드

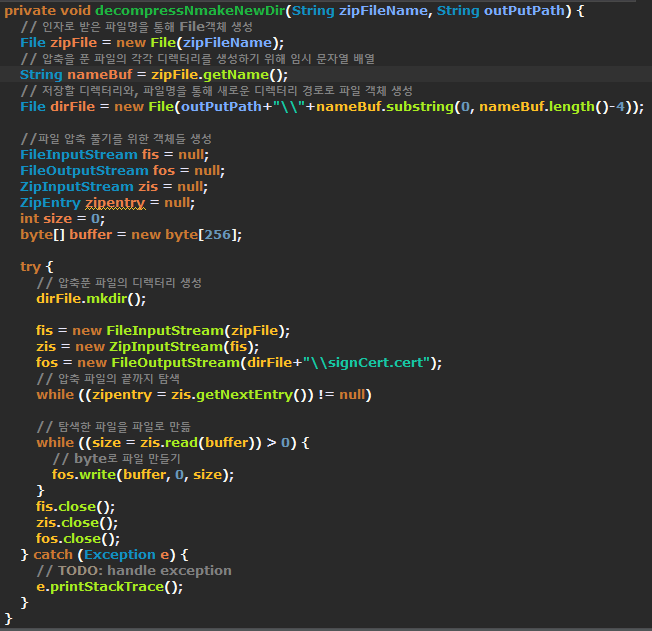


그림 23. decompressNmakeNewDir 메서드

# 3. 코드 실행

## 3.1 용의자의 사이트에서 HTML소스 파싱 및 TXT파일로 출력

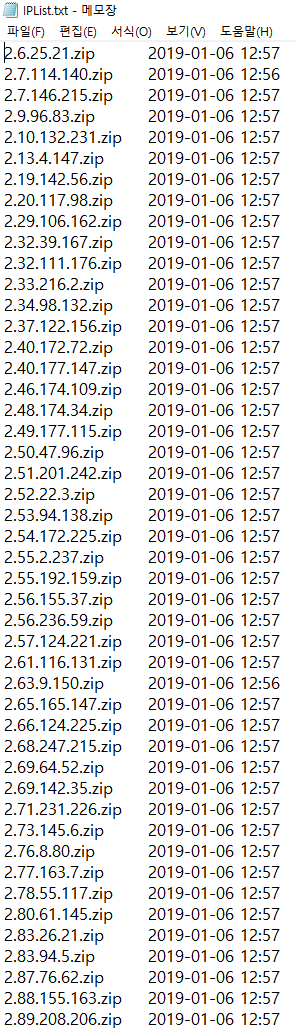


그림 24. 용의자의 사이트에서 파싱한 IP주소와 시간값

## 3.2 인증서 다운로드

GetFileFromURLWithProxy 클래스의 생성자를 수행함으로써 자동으로 인증서가 다운로드 된다.

다운로드가 완료된 인증서는 [그림 28]과 같다.

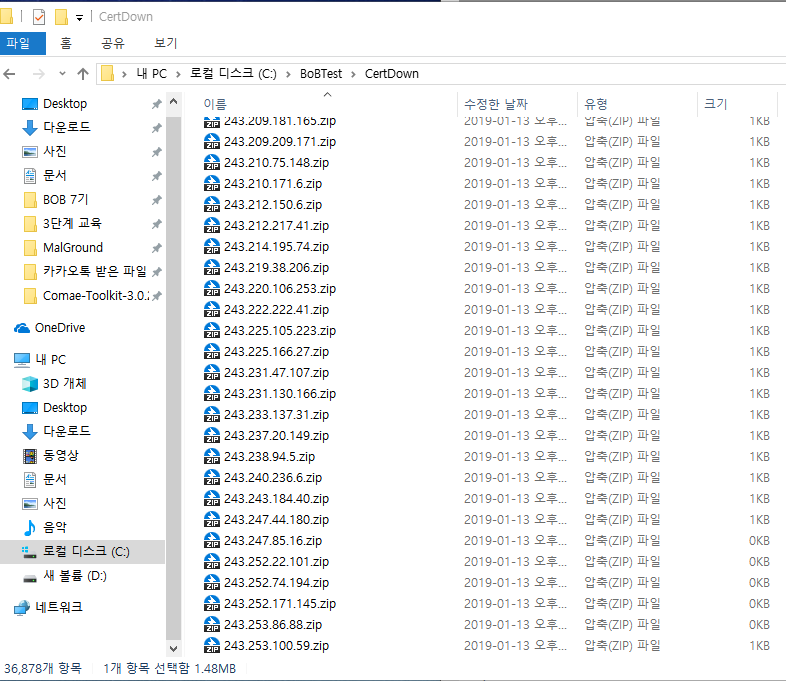


그림 25. 다운로드가 완료된 인증서들

총 36,876개의 인증서가 다운로드 되었다.

## 3.2 인증서 파싱 및 DB삽입

[그림 29]과 같이 MySQL에서 DB에 Insert되며 데이터가 증가하고, 36,876개의 데이터가 삽입된 것을 확인할 수 있다.

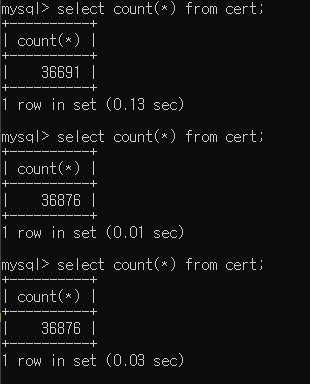


그림 29. MySQL DB 데이터의 증가

또한 [그림 30]과 같이 데이터가 정상적으로 Insert된 것을 확인할 수 있다.



그림 30. MySQL에 삽입된 데이터

# 4. DB 분석

은행별 유출된 공인인증서의 개수 파악은 다음 쿼리문을 통해 파악할 수 있다

SELECT bank, count(\*) as bankCount

FROM cert

GROUP BY bank

ORDER BY bankCount desc;



그림 31. MySQL 각 은행별 유출된 공인인증서 개수

표 3. 각 은행별 유출된 공인인증서 개수

|  |  |
| --- | --- |
| bank | count |
| ibk | 2933 |
| gwangju | 2921 |
| woori | 2909 |
| shinhan | 2870 |
| gyungnam | 2865 |
| jeju | 2861 |
| kookmin | 2822 |
| sc | 2818 |
| daegu | 2810 |
| jeonbuk | 2794 |
| busan | 2778 |
| citi | 2775 |
| nh | 2720 |

가장 많은 공인인증서가 유출된 은행은 ibk은행인 것을 확인 할 수 있다.

# 5. DB데이터 CSV로 추출

해당 과정은 다음과 같은 쿼리문을 통해 진행 하였다.

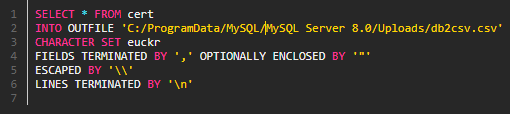


그림 32. DB데이터의 CSV추출을 위한 쿼리문

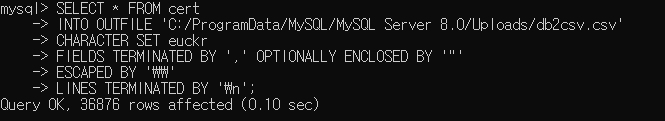


그림 33. MySQL데이터의 CSV추출을 위한 쿼리문 수행

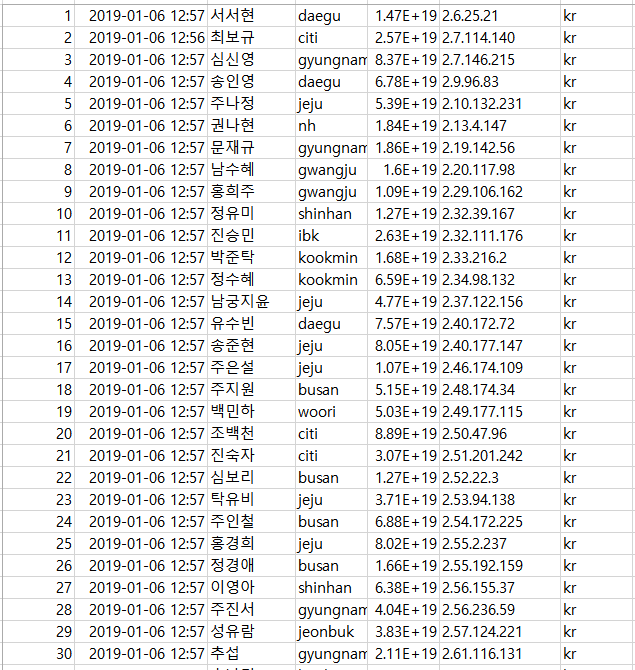


그림 34. MySQL DB에서 추출된 CSV파일의 일부

# 6. 결론 및 후기

수사관의 관점에서 피해상황을 최소화 하고자 코드를 제작하여 자동으로 인증서를 폐기하기위해 DB에 삽입하는 코드를 구현하였다. 이전에 진행하였던 과제였으나, 다양한 이슈가 발견하였으나 다시한번 코드를 구현하며 모두 해결할 수 있었다.

해결한 문제 중 이전에 DB로 사용하였던 SQLite에서 이름 데이터에 한글자에서 네글자 까지만 처리하게 하여 3개의 튜플에 NULL값이 삽입되었던 문제가 있었다. 해당 에러를 이번 과제에서 파악하여 한글자에서 다섯글자까지 처리하게 하여 모든 데이터가 정상적으로 Insert될 수 있도록 조치하였다.

또한 손으로 직접 IP값을 드래그 하여 TXT파일을 만드는 것을 자동화 하였으며, 압축을 푸는 것 또한 압축 프로그램을 통해 압축을 푸는 방식이 아닌, 코드를 구현하여 압축풀기를 자동화 하였다. 무엇보다, Proxy연결을 설정 및 연결이 확실한 Proxy만 불러와 사용하기 때문에 비교적 안정적으로 수사관의 IP주소를 숨겨 파일을 다운받을 수 있었다.

그리하여, 용의자의 사이트에서 다운받을 파일들을 파싱 및 TXT파일로 출력하고, 출력된 TXT파일을 통해 프록시를 설정하여 다운로드 받은 뒤, 압축을 풀고, 풀어진 각 인증서의 데이터를 파싱하여 DB에 삽입하는 전 과정을 자동화 하였다.