VirusTotal API

좋은징조 - 이태희

목차

1. VirusTotal API란?

- 최대 70가지 이상의 각기 다른 바이러스 검사 소프트웨어 제품을 사용하여 무료 파일 검사를 제공하는 웹사이트인 VirusTotal에서 제공하는 오픈소스 API이다.

2. 필요한 Package

- requests - HTTP 요청을 보내는 Python 패키지이며, VirusTotal API를 원활히 사용하기 위해 필요하다.

아래와 같이 가상환경 안에서 설치한다.

```
(env)$ pip install requests
```

3. 오픈소스 API

- VirusTotal이 제공하는 여러 오픈소스 API 중 우리가 사용하고자 하는 오픈소스 API는 **URL값을 토대로 scan**하는 API로써 다음과 같다.

- /url/report API는 Virustotal에서 발급받은 Key값(apikey)과 scan하고자 하는 URL(resource) 두 인자를 필요로 하고, 결과를 JSON 형태로 반환한다.

4. Virustotal API 제공 데이터

```
Example response
{
 'response_code': 1,
 'verbose_msg': 'Scan finished, scan information embedded in this object',
 'scan_id': '1db0ad7dbcec0676710ea0eaacd35d5e471d3e11944d53bcbd31f0cbd11bce31-
1390467782',
 'permalink': 'https://www.virustotal.com/url/__urlsha256__/analysis/1390467782/',
 'url': 'http://www.virustotal.com/',
 'scan_date': '2014-01-23 09:03:02',
 'filescan_id': null,
 'positives': 0,
 'total': 51,
 'scans': {
    'CLEAN MX': {
      'detected': false,
      'result': 'clean site'
    },
    'MalwarePatrol': {
      'detected': false,
      'result': 'clean site'
```

```
    }
  }
}
```

Virustotal 오픈소스 API에서 제공하는 데이터는 위와 같으며, 그 중 우리가 **필요한 데이터들은 다음과 같다.**

데이터	설명
scan_date	최종 스캔 날짜 및 시간
positives	해당 URL을 악성으로 탐지한 엔진의 수
total	총 바이러스 검사 소프트웨어 엔진의 수
scans	해당 URL을 악성으로 탐지한 엔진 명과 탐지 결과 등의 데이터

---

5. Virustotal Module API 설명 (views.py)

- QR Code Previewer를 구현하기 위해 적용된 Virustotal Module API는 다음과 같다.

- vt_api.py - API를 이용해 데이터를 받아와 원하는 데이터만 파싱을 하는 역할

```
import requests
import os
import datetime

# Virustotal API를 이용해 dictionary 형태로 데이터 return
# resource 변수에는 scan하고자 하는 url정보가 들어있다.
def requestVirAPI(resource):
    url = 'https://www.virustotal.com/vtapi/v2/url/report'
    #1. open api key
    try:
        print(os.getcwd())
        # 파일입출력을 이용해 개인으로 발급받은 apikey를 불러온다.
        with open('apikey.txt','r') as f:
            key = f.read()
            key = key.replace("\n","")
            print(key)
            params = {'apikey':key, 'resource':resource}
            response = requests.get(url, params=params)
            return response.json() # JSON형태를 파싱을 위해 dictionary형태로 return
    except Exception as e:
        print('[DEBUG] VT_API error:', e)

# 악성으로 탐지한 엔진명만 골라 파싱한다. API 제공 데이터에서 'scans'데이터에 해당한다.
def extractScanInfo(scanInfo):
```

```
    # parsing ScanInfo
    result = dict()
    for machineName in scanInfo.keys():
        if scanInfo[machineName]['detected']:
            result.update({machineName:scanInfo[machineName]})
    return result # dict형

# VirusTotal API를 이용해 데이터를 받아와 필요한 정보만 파싱해 dictionary 데이터로 변환한다.
def getVirInfo(resource):
    result = dict()
    jsonData=requestVirAPI(resource) # API로부터 데이터를 받아온다.
    # VirusTotal API로부터 성공적으로 데이터를 받아왔으면, 필요한 데이터만 파싱해 저장한다.
    if jsonData['response_code'] == 1:
        result.update({'scanDate':jsonData['scan_date']})
        result.update({'positives':jsonData['positives']})
        result.update({'total':jsonData['total']})
        scanInfo = extractScanInfo(jsonData['scans'])
        result.update({'scanInfo':scanInfo})
        return result # dict형
    # 성공적으로 데이터를 받아오지 못했으면, 기본값을 세팅해 저장한다.
    else:
        result.update({'scanDate':str(datetime.datetime.now())})
        result.update({'positives':0})
        result.update({'total':0})
        result.update({'scanInfo':{}})
        return result # dict형
```

- views.py - 위의 vt_api.py코드를 Django의 View와 연동하는 코드

```
from django.shortcuts import render
from rest_framework import status
from rest_framework.response import Response
from rest_framework.decorators import api_view,renderer_classes
from rest_framework.renderers import JSONRenderer
from .serializers import *
from .screenshoter.screenshoter import Screenshoter
from .vt_validate.vt_api import * #vt_api.py import

# VirusTotalInfo 모델의 객체를 DB에 저장하는 함수
def saveVirusTotalInfo(qrinfo):
    try:
        vt_api_info = getVirInfo(qrinfo.url) # dict형
        # 요청받은 URL의 Virustotal API 결과 값을 저장하는 변수

        virInfo = VirusTotalInfo(qrInfo=qrinfo,scanDate=vt_api_info['scanDate'],
                  positives=vt_api_info['positives'],total=vt_api_info['total'])
        # Virustotal 결과값 중 ScanDate, positives, total 데이터만 파싱해 VirusTotalInfo 모델
의 객체인 virInfo 생성

        virInfo.save()
        # virInfo모델 객체를 DB에 저장
```

```
        #machineName 및 scanInfo 모델 DB에 저장
        for mname in vt_api_info['scanInfo'].keys():
            mnameModel = None
            if not machineName.objects.filter(name=mname).exists():
            # 악성으로 탐지한 엔진명을 machineName 모델의 객체에 저장한다. scanInfo의 탐지 엔진명이
DB의 machineName 모델에 없다면, 객체를 만들어 저장해준다.
                print(mname,"not saved")
                mnameModel = machineName(name=mname)
                mnameModel.save()
                # machineName Model의 객체 DB에 저장
            else:
            # scanInfo의 탐지 엔진명이 이미 machineName 모델에 있다면 이를 불러온다.
                mnameModel = machineName.objects.get(name=mname)

            sinfo = ScanInfo(vInfo=virInfo,machineName=mnameModel,
                             detected=vt_api_info['scanInfo'][mname]['detected'],
                             result=vt_api_info['scanInfo'][mname]['result'])
            # 이미 생성된 virInfo의 scan_date, positives, total 데이터에 탐지한 엔진의 세부정보
인 'scans'데이터를 추가해 scanInfo 모델의 객체를 생성한다.
            sinfo.save()
            # ScanInfo 객체 DB에 저장
    except Exception as e:
        print("DEBUG virustotal api error",e)
    return virInfo
```