컴퓨터 보안 HW2

컴퓨터소프트웨어학부

2016024839 박정민

언어는 python을 사용하였고 컴파일 환경은 visual studio code python 3.8.2입니다.

저는 opcode sequence를 분석하는 static analysis방법을 선택하였습니다. 주어진 opcode를 분석하기 위해 우선

1. 제시해주신 sample file opcode를 분석하기 위해 opcode폴더 안에 있는 정상파일과 악성코드 텍스트 파일이름을 list로 묶었습니다.

2. 정상파일인지 악성코드인지 분류할 기준을 세우기 위해 교수님이 수업자료로 사용하신 06. Malware.pdf 를 참고하여 75페이지의 Opcode Frequency도표를 참조하였습니다.

Opcode	Goodware	Kernel RK	User RK	Tools	Bot	Trojan	Virus	Worms
mov	25.3%	37.0%	29.0%	25.4%	34.6%	30.5%	16.1%	22.2%
push	19.5%	15.6%	16.6%	19.0%	14.1%	15.4%	22.7%	20.7%
call	8.7%	5.5%	8.9%	8.2%	11.0%	10.0%	9.1%	8.7%
pop	6.3%	2.7%	5.1%	5.9%	6.8%	7.3%	7.0%	6.2%
cmp	5.1%	6.4%	4.9%	5.3%	3.6%	3.6%	5.9%	5.0%
jz	4.3%	3.3%	3.9%	4.3%	3.3%	3.5%	4.4%	4.0%
lea	3.9%	1.8%	3.3%	3.1%	2.6%	2.7%	5.5%	4.2%
test	3.2%	1.8%	3.2%	3.7%	2.6%	3.4%	3.1%	3.0%
jmp	3.0%	4.1%	3.8%	3.4%	3.0%	3.4%	2.7%	4.5%
add	3.0%	5.8%	3.7%	3.4%	2.5%	3.0%	3.5%	3.0%
jnz	2.6%	3.7%	3.1%	3.4%	2.2%	2.6%	3.2%	3.2%
retn	2.2%	1.7%	2.3%	2.9%	3.0%	3.2%	2.0%	2.3%
xor	1.9%	1.1%	2.3%	2.1%	3.2%	2.7%	2.1%	2.3%
and	1.3%	1.5%	1.0%	1.3%	0.5%	0.6%	1.5%	1.6%

<Daniel Bilar, "Statistical Structures: Fingerprinting Malware for Classification and Analysis", Black Hat, 2006>

3. 위에 나와있는 opcode frequency를 기준으로 삼아 각 opcode별 비율 * 10을 하여 list로 만들었습니다.

```
goodware = [253, 195, 87, 63, 51, 43, 39, 32, 30, 30, 26, 22, 19, 13]
kernel_rk = [370, 156, 55, 27, 64, 33, 18, 18, 41, 58, 37, 17, 11, 15]
user_rk = [290, 166, 89, 51, 49, 39, 33, 32, 38, 37, 31, 23, 23, 10]
tools = [254, 190, 82, 59, 53, 43, 31, 37, 34, 34, 34, 29, 21, 13]
bot = [346, 141, 110, 68, 36, 33, 26, 26, 30, 25, 22, 30, 32, 5]
trojan = [305, 154, 100 ,73, 36, 35, 27, 34, 34, 30, 26, 32, 27, 6]
virus = [161, 227, 91, 70, 59, 44, 55, 31, 27, 35, 32, 20, 21, 15]
worms = [222, 207, 87, 62, 50 ,40, 42, 30 ,45, 30 ,32, 23, 23, 16]
```

4. 유사도를 구하는 방법으로 cosine similarity를 사용하였습니다. Numpy 라이브러리를 사용하여 유사도를 구하는 함수를 만들었습니다.

```
def cos_sim(A, B):
    return dot(A, B) / (norm(A) * norm(B))
```

5. 샘플 파일 100개를 분석하여 텍스타 파일을 구성하는 opcode 중 일치하는 opcode의 빈도수를 체크하기 위해 index list를 만들어 위의 기준표와 같이 빈도수를 체크하였습니다.

```
opcodes = ['mov\n', 'push\n', 'call\n', 'pop\n', 'cmp\n', 'jz\n', 'lea\n',
    'test\n', 'jmp\n', 'add\n', 'jnz\n', 'retn\n', 'xor\n', 'and\n']
```

```
f = open(file_dir + is_malware + mal_file_list[j], mode = 'rt')
idx = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
for line in f:
    for i in range(len(opcodes)):
        if opcodes[i] == line:
            idx[i] += 1
```

6. 그렇게 구한 index list와 위 도표로부터 구한 goodware, malware opcode 빈도수 list의 cosine similairty를 구하였습니다. 아래 스크린샷은 정상파일과 악성코드별 유사도입니다.

```
goodware similarity :  0.8911639497927966
kernel_rk similarity :  0.9581132831980295
user_rk similarity :  0.9367188053993563
tools similarity :  0.9003298458474945
bot similarity :  0.9547357187886442
trojan similarity :  0.9407132775480678
virus similarity :  0.755356623332537
worms similarity :  0.8574813651175621
most similar model : kernel_rk
```

7. 그렇게 정상파일 샘플에서의 샘플 100개, 악성코드 샘플에서의 샘플 100개를 테스트하였습니다.

8. 정상파일 샘플의 경우 가장 높은 유사도가 goodware이면 정상파일이라고 판단하고, 악성코드 샘플의 경우에는 가장 높은 유사도가 goodware가 아니면 악성코드로 판별하였습니다. 결과는 정상파일 샘플 판별 정확도는 17%, 악성코드 샘플 판별 정확도는 96%입니다.

```
accuracy :  17.0 %
```

```
accuracy :  96.0 %
```