주성분 분석을 사용한 바이러스 탐지 명령어 집합에 대한 연구

김명관, 주현수°
을지대학교
binsum@eulji.ac.kr, hainya1004@naver.com°

A Study on Instruction Set for Virus Detection using PCA

Kim Myung Gwan, Joo Hyun Soo°
Eulji University

요 약

중요한 정보를 저장하고 있는 서버 및 개인용 컴퓨터를 위협하는 바이러스가 현실적인 문제로 대두되고 있다. 범용 바이러스 탐지기법을 위해 주성분 분석(PCA)을 사용하여 휴리스틱 접근으로 바이러스 탐지 능력을 높일 수 있는 명령어 집합을 찾았고, PCA의 결과좌표 분포에 따라 정상파일인 경우 90%의 분류, 바이러스파일에 대하여 85%의 분류 능력을 확인하였다.

1. 서 론

최근 알려진 바이러스를 탐지하는 기술이 발전되어 SpyZero, Norton Anti-Virus, Forefront Client Security 같은 성공적인 소프트웨어들이 활용되고 있다. 이와 같은 방법은 알려진 바이러스 파일의 분석과 단순한 매칭에 기반을 두고 있다. 그러므로 새로운 바이러스를 탐지하지 못하고 시스템이 공격에 노출 후 바이러스 유형과 파일상태를 분석한 후에야 탐지 알고리즘에 반영될 수 있다. 그러나 정상파일과 바이러스 파일의 분류가 가능하면 새로운 바이러스 파일에 대한 대처가 가능하다. 이러한 바이러스 탐지 기법으로는 특징추출, 알고리즘, 학습알고리즘 등이 있다. 특징추출은 n-gram, LibBFD, GNU Strings, Byte Sequences Using Hexdump[1] 등의 방법을 사용한 탐지 기법이 있다. 그리고 알고리즘으로는 Signature, RIPPER[2], Naive Bayes[3], Multi-Naive Bayes[4]등의 방법을 사용한 탐지 기법이 있다.[5] 학습 알고리즘으로는 Support Vector Machine (SVM), Random Forest(RF), k-nearest neighbor (KNN), Fuzzy Diagnosis System (FDS)[6] 등의 다양한 바이러스 진단 방법이 있다.[7] 최근에 특징추출 방법중 n-gram방식으로 이진실행파일을 역어셈블 하여 명령어를 구성하는 연산코드로부터 instruction sequence를 특징패턴으로 추출하는 기법을 제안, 활용하고 있다.[8] 본 논문에서는 정상파일과 바이러스를 분류하기 위해서 일반적인 바이러스 탐지 기법 중에서 특징추출을 이용하여 연산코드의 빈도에 영향에 비중으로 명령어 집합을 찾아내었다. 바이러스 탐지에 필요한 명령어 집합을 찾기 위해서 다변량 분석 중에 하나인 주성분 분석(PCA)을 이용하여 휴리스틱 접근을 하였다. 주성분 분석(PCA)의 결과 좌표 분포를 이용하여 정상파일과 바이러스파일을 분류하였다.

2장에서는 본 연구에 관련되어 사용한 다변량 분석 중에 하나인 주성분 분석(PCA)을 기술한다. 3.1은 바이러스 탐지기법을 실험하기 위한 데이터 준비를 기술한다. 3.2는 바이러스 탐지기법 중에서 특징추출에 주성분 분석(PCA)을 사용하여 휴리스틱 접근을 하는 실험 및 고찰을 기술한다. 4장은 결론으로서 실험결과와 추후 발전방향을 기술하였다.

2. 주성분 분석(PCA)

다변량 분석의 목적은 차원축소(Dimension Reduction)로서 주성분 분석, 인자분석, 정준상관분석 등이 있다. 그중에서 실험을 위해 주성분 분석을 사용하였다. 주성

분 분석(Principal Component Analysis)이란 해석 하고자 하는 다차원의 데이터를 포함된 정보의 손실을 가능한 한 적게 해서 2 혹은 3차원의 데이터로 축약하는 방법이다. 주성분 분석을 사용하면 관측대상이 어떠한 위치에 있는지 시각적으로 파악할 수 있게 된다.[9] 분산이 작은 성분을 제거함으로써 데이터 차원을 줄이는 동시에 데이터에 포함되어 있던 잡음(noise)을 제거할 수 있다. 데이터 행렬 X의 차원을 낮추는 식은 다음 식과 같다.

$$X \cdot V^n$$

여기서 V는 X의 상관행렬의 고유벡터를 해당하는 고유 값의 내림차순으로 정렬한 행렬이고, n은 이 중 n개의 열을 사용하겠다는 의미이다.

본 연구에서는 명령어 집합의 분별력을 판단하기위해 주성분 분석(PCA)을 사용하였다. 주성분 분석(PCA)을 사용 하는데 에 있어서 The Unscrambler 프로그램을 이용하여 하였다.[10]

3. 실험 및 고찰

3.1 데이터 준비

탐색기법 실험 준비를 위하여 VX heaven[11-12]에서 200개의 바이러스 파일과 Windows 시스템 실행 파일에서 200개의 정상 파일을 수집하였다. 수집한 자료를 분석하기 위해서 W32sdm 소프트웨어 프로그램을 이용하였다. 역 어셈블 과정을 통하여 중간 파일을 400개 얻을 수 있었다. 이 중간 파일을 이용하여 명령어를 추출하였다. 그중에 바이러스와 정상 파일에 누적 출현 빈도수의 합이 최소 2,612개 이상 최대 567,544개가 존재하는 명령어 50개[표 1]을 선발하였다.

[표 1] Mdata 명령어군

aaa	aas	adc	add	and
arpl	bound	byte	call	cmp
daa	das	dec	imul	inc
insb	insd	int	ja	jb
jbe	jge	jl	jle	jmp
jnb	jne	jns	jo	jpe
js	lea	leave	mov	or
outsb	outsd	outsw	pop	popa
popad	push	repz	ret	sbb
shl	shr	sub	test	xor

이렇게 만들어낸 (400개의 파일 * 50개의 명령어)의 데이터를 이 논문에서는 Mdata라고 부르도록 하고 Mdata를 모든 실험과정에 기본데이터로 하였다.

Mdata에서 정상(200개의 파일 * 50개의 명령어)파일은 Jdata라고 하고 바이러스(200개의 파일 * 50개의 명령어)파일은 Vdata라고 정한다. Jdata와 Vdata의 두 가지 자료[그림 1]을 엑셀 소프트웨어를 이용하여 구성하였다.

	A	B	C	D	E	F	G
1		정상파일	add	and	call	int	jl
2	1	지뢰찾기	357	56	403	8	23
3	2	actmovie	44	2	22	43	2
4	3	ahui	2097	721	1036	1194	108
5	4	alcrmv	760	396	3424	922	63
6	5	append	134	425	255	45	2
7	6	arp	171	41	170	4	3
8	7	asr_fmt	656	87	603	1139	32
9	8	asr_ldm	528	52	260	8	1
10	9	asr_pfu	706	257	456	541	54
11	10	at	319	71	350	405	41
12	11	atmadm	110	34	99	147	9
13	12	attrib	237	41	221	4	0
14	13	auditusr	311	20	228	225	9
15	14	autolfn	150	28	130	192	42

[그림 1] 정상파일의 명령어 빈도(Jdata)

정규화(canonicalization)는 데이터의 규정 일치와 검증된 형식을 확인하고, 비정규 데이터를 정규 데이터로 만드는 것이다.[13] Jdata와 Vdata의 자료의 정규화를 위하여 LOG함수를 이용한 정규화 과정을 작업하였다. 설정범위는 0~1안에 범위를 벗어나지 않도록 설정하였다. 정규 화된 Jdata, Vdata를 자료를[그림 2] 바탕으로 실험을 진행하였다.

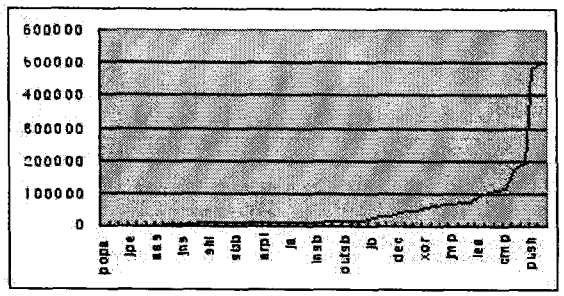
	A	B	C	D	E	F	G
203		정규화	add	and	call	int	jl
204	1	지뢰찾기	0.593502	0.406458	0.60574	0.209971	0.316605
205	2	actmovie	0.382107	0.06999	0.312117	0.379785	0.06999
206	3	ahui	0.77228	0.664477	0.701079	0.715411	0.472776
207	4	alcrmv	0.669796	0.60397	0.821788	0.689307	0.418351
208	5	append	0.494557	0.611107	0.559526	0.384376	0.06999
209	6	arp	0.519177	0.374976	0.518585	0.13998	0.110932
210	7	asr_fmt	0.654937	0.450943	0.646431	0.710649	0.349951
211	8	asr_ldm	0.633019	0.398975	0.561487	0.209971	0
212	9	asr_pfu	0.662354	0.560315	0.618216	0.635475	0.402786
213	10	at	0.582137	0.430422	0.591502	0.60624	0.374976
214	11	atmadm	0.474629	0.356073	0.46399	0.503907	0.221864
215	12	attrib	0.552135	0.374976	0.545077	0.13998	#NUM!
216	13	auditusr	0.579573	0.302493	0.548226	0.546888	0.221864
217	14	autolfn	0.505947	0.336468	0.491497	0.530873	0.377409
218	15	bootcfg	0.68608	0.496793	0.771117	0.316605	0.291854

[그림 2] 정상파일의 명령어 빈도의 정규화(Jdata)

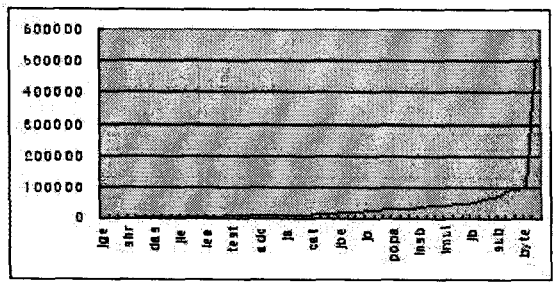
3.2 실험 및 고찰

본 논문에서는 Jdata 명령어 출현빈도수는 최소 6개인 popa 명령어를 시작으로 최대 501,862개인 mov 명령어까지 이다. Vdata 명령어 출현빈도수는 최소 1,314개

인 repz 명령어를 시작으로 최대 512,830개인 and 명령어까지로 하였다. 그 기준으로는 Mdata, Jdata, Vdata에 출현빈도수, 출현빈도수 차이를 이용하였다. Jdata 명령어들에 출현빈도수의 그래프[그림 3]와 Vdata 명령어들에 출현빈도수의 그래프[그림 4]를 밑에 나타낸다.



[그림 3] Jdata 명령어 누적 출현빈도수 그래프



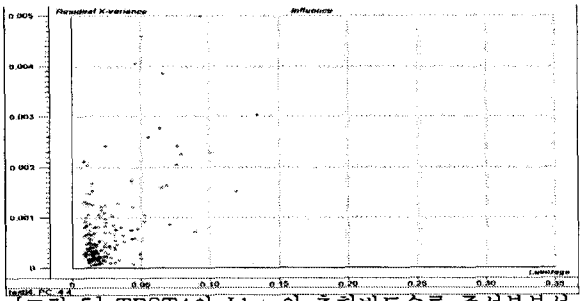
[그림 4] Vdata 명령어 누적 출현빈도수 그래프

실험에 적용하는 집합들의 성격은 아래 [표 2]에서 나타내었다. [표 2]에서 임계값은 파일들에 명령어 누적 출현빈도수의 양이 크게 증가하는 부분으로 정한다.

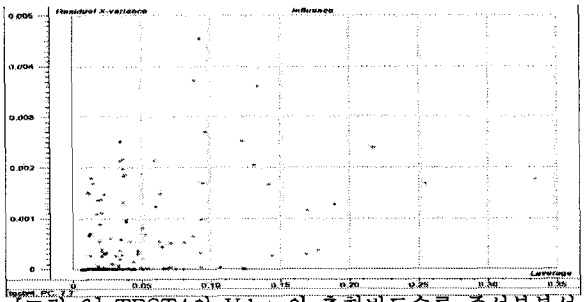
[표 2] 1차 실험에 사용되는 집합

종류	집합의 성격
TEST0	전체파일의 누적 출현빈도수가 최소 6개 이상명령어 (50개)
TEST1	정상파일의 명령어 누적 출현빈도수가 임계값(50,000)보다 큰 명령어 (13개)
TEST2	바이러스파일의 명령어 누적 출현빈도수가 임계값(35,000)보다 큰 명령어 (13개)
TEST3	정상파일, 바이러스파일의 명령어 누적 출현빈도수 차이 수치가 임계값(40,000)보다 큰 명령어 (14개)
TEST4	정상파일, 바이러스파일의 명령어 누적 출현빈도수 차이 수치가 바이러스파일보다 정상파일이 크고 임계값(40,000)보다 큰 명령어 (11개)
TEST5	정상파일, 바이러스파일의 명령어 누적 출현빈도수 차이 수치가 정상파일보다 바이러스파일이 크고 임계값(10,000)보다 큰 명령어 (15개)

TEST0부터 TEST5까지 PCA를 이용한 분류를 하였다. 그 결과 분류능력이 있는 집합과 분류능력이 없는 집합을 발견하였다. [표 3]에 보여준 분류정확도는 1차 실험 결과이다. [그림 5]와 [그림 6]은 실험결과 중에서 가장 좋은 분류 성능을 보여준 TEST4의 결과를 보여준다. TEST4의 실험결과가 정상파일 기준에서 94%의 분류능력을 확인하였고 바이러스파일 기준에서 71%의 분류능력이 확인되었다.



[그림 5] TEST4의 Jdata의 출현빈도수를 주성분분석한 결과



[그림 6] TEST4의 Vdata의 출현빈도수를 주성분분석한 결과

[표 3] PCA 실험결과 TEST 집합의 분류정확도

집합 / 종류	TEST0	TEST1	TEST2	TEST3	TEST4	TEST5
정상파일	74.5%	67.5%	분류불가	분류불가	94%	분류불가

바이러스	58%	69.5%	분류불가	분류불가	71%	분류불가

TEST4는 실험결과 중에서 가장 좋은 효과를 얻을 수가 있었다. TEST4는 add, call, cmp, int, jmp, jne, lea, mov, push, ret, test의 명령어로 구성되어 있다. 이 명령어들을 한 개씩 제거하여 10개의 명령어 집합으로 실험을 하였다. 한 개씩 제거한 명령어 집합을 [표 4]에서 나타낸다.

[표 4] 2차 실험에 사용되는 집합

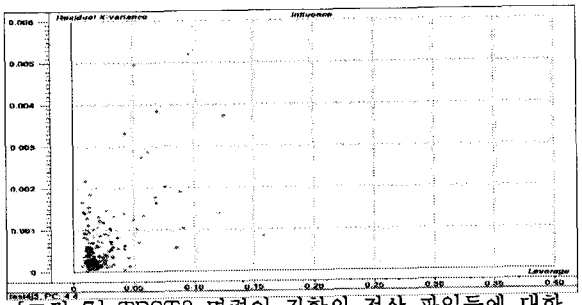
종류	집합의 성격
Test6	Test4에 add를 제외한 명령어 집합
Test7	Test4에 call를 제외한 명령어 집합
Test8	Test4에 cmp를 제외한 명령어 집합
Test9	Test4에 int를 제외한 명령어 집합
Test10	Test4에 jmp를 제외한 명령어 집합
Test11	Test4에 jne를 제외한 명령어 집합
Test12	Test4에 lea를 제외한 명령어 집합
Test13	Test4에 mov를 제외한 명령어 집합
Test14	Test4에 push를 제외한 명령어 집합
Test15	Test4에 ret를 제외한 명령어 집합
Test16	Test4에 test를 제외한 명령어 집합

TEST6부터 TEST16까지 PCA를 이용한 분류를 하였다. 그 결과 분류능력이 있는 집합과 분류능력이 없는 집합을 발견하였다. [표 5]에 보여준 분류정확도가 2차 실험 결과이다.

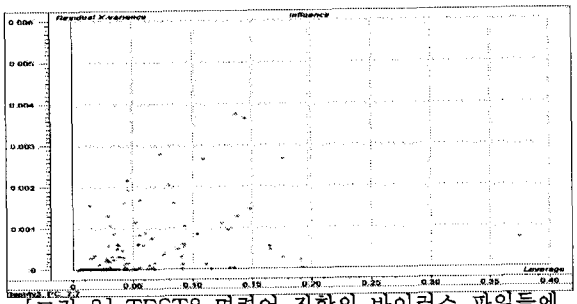
[표 5] PCA 실험결과 TEST 집합의 분류정확도

집합 / 종류	TEST6	TEST7	TEST8	TEST9	TEST10	TEST11
정상파일	96%	99%	90%	91%	98%	99.5%
바이러스	64.5%	67.5%	85%	79%	64%	63%
집합 / 종류	TEST12	TEST13	TEST14	TEST15	TEST16	
정상파일	93%	분류불가	99%	97.5%	99%	
바이러스	72%	분류불가	65.5%	75%	74%	

[그림 7]과 [그림 8]은 실험결과 중에서 가장 좋은 분류 성능을 보여준 TEST8의 결과를 보여준다. TEST8의 실험결과가 정상파일 기준에서 90%의 분류능력을 확인하였고 바이러스파일 기준에서 85%의 분류능력이 확인되었다.



[그림 7] TEST8 명령어 집합의 정상 파일들에 대한 PCA결과



[그림 8] TEST8 명령어 집합의 바이러스 파일들에 대한 PCA결과

4. 결 론

1985년 최초의 컴퓨터 바이러스가 등장한 이래 매년 새롭게 발생하는 바이러스는 중요한 정보와 재산을 위협하고 있다. 원시형, 암호형, 은폐형, 갑옷형, 매크로 바이러스 등이 발전하고 새로 개발되고 있다.[13] 이런 바이러스를 탐지하고 중요한 정보와 재산을 지켜내는 것이 반드시 필요하다. 컴퓨터 바이러스를 탐지하는 여러 가지 탐지방법이 있다. 그중에 정상파일과 바이러스를 분류하기 위해서 일반적인 바이러스 탐지 기법 중에서 특징추출을 이용하여 연산코드의 빈도에 영향에 비중으로 명령어 집합을 찾아내었다. 바이러스 탐지에 좋은 효과의 명령어 집합을 찾기 위해서 다변량 분석 중에 하나인 주성분 분석(PCA)을 이용하여 휴리스틱 접근을 하였다. 실험결과 정상파일 기준에서 90%의 분류능력을 확인하였고 바이러스파일 기준에서 85%의 분류능력이 가장 좋게 확인되었다. 향후 많은 실험을 통하여 분류능력이 더 나은 명령어 집합을 얻을 수 있을 것이다. 이와 같은 분류능력은 준비하는 샘플데이터와 명령어 집합 등에 따라 다른 결과를 나타내므로 다양한 실험데이터와 집합을 가지고 수정 보안되어야 한다.

이후, 이 결과를 바탕으로 여러 가지 알고리즘을 이용하여 실시간으로 정상파일, 바이러스파일 구별을 할 수 있는 실험할 예정이다.

참 고 문 헌

[1] http://www.dmares.com/maresware/html/hexdump.htm

[2] M.Sasaki and K.Kita. "Rule-based text categorization using hierarchical categories Systems, Man, and Cybernetics", IEEE International Conference, pp.2827-2830, 11-14 Oct 1998.

[3] Daniel Lowd and Pedro Domingos. "Naive Bayes Models for Probability Estimation", University of Washington, 2005.

[4] Liangxiao Jiang, Zhang, H. "Learning instance greedily cloning naive Bayes for ranking Data Mining", IEEE International Conference, pp.88, 27-30 Nov 2005.

[5] Matthew G.Schultz and Eleazar Eskin, Erez Zadok, Salvatore J.Stolfo, "Data Mining Methods for Detection of new Malicious Executables", IEEE Computer Society, 2001

[6] 이현숙, "컴퓨터 바이러스 분류를 위한 퍼지 클러스터 기반 진단 시스템", 정보처리학회논문지 B 제14-B권, 제1호, pp.1-6, 2007.

[7] I. Witten and E. Frank, "Data mining: Practical machine learning tools and techniques with java implementations", Morgan Kaufmann, San francisco, CA, 2000.

[8] Jianyong Dai, Joohan Lee and Morgan C. Wang, "Detecting Unknown Computer Virus Using Data Mining Techiques", Business Intelligent Symposium, poster presentation, April, 2006.

[9] 노형진, "다변량 분석 이론과 실제", 형설출판사, 2005.1.

[10] http://www.camo.com/

[11] http://vx.netlux.org/lib/

[12] http://vx.netlux.org/src.php

[13] James Lawrence and April Kerby Alma College. Miami University. Alma, MI. Oxford, OH, "A Multivariate Statistical Analysis of Stock Trends", 2003.

[14] http://terms.naver.com/