유전자 알고리즘을 적용한 스크립트형 웜 바이러스 탐지 시스템 설계

최준호*, 김판구*
*조선대학교 전자계산학과
e-mail : spica@hitel.net

A Design of Script worm virus detection system using the Genetic Algorithm

Jun-Ho Choi*, Pan-Koo Kim*
*Dept. of Computer Science, Chosun University

요 약

최근 인터넷의 전자 메일 서비스 사용의 증가로 인해 스크립트형 웜 바이러스에 대한 피해가 확산되고 있다. 전자 메일을 통하여 유포되는 스크립트형 웜 바이러스는 지속적으로 새로운 형태로 변이되어 나타나고 있지만, 이에 대한 예방 방법은 새로운 패턴이 분석된 후 이를 토대로 탐지하기 때문에 적극적인 대응을 하지 못하는 실정이다.

이에 본 논문에서는 스크립트형 웜 바이러스의 행위를 추출하여 일정한 패턴을 정의한 후 이를 기반으로 스크립트형 웜 바이러스 탐지 시스템을 설계하고, 기존의 패턴에 유전자 알고리즘을 적용하여 알려지지 않은 새로운 패턴을 생성한 후 바이러스 탐지에 활용할 수 있는 방안을 연구한다.

1. 서론

최근 인터넷 사용이 급증함에 따라 인터넷 서비스를 이용한 바이러스가 확산되고 있다. 지금까지는 기생형 바이러스의 형태를 띄는 단순한 형태가 대부분이지만 현재 나타나고 있는 바이러스는 자동으로 시스템의 결함을 찾아내어 공격하는 방식이 웜 바이러스 형태와 결합되어 더욱 복잡하고 지능적으로 진화하고 있다. 웜 바이러스는 대부분 C++이나 BASIC과 같은 고급 언어로 작성되었지만, 최근에는 *VBS/Love_Letter* 등과 같이 VBScript 등과 같이 스크립트 언어를 이용한 웜 바이러스 제작이 일반화되고 있다.

윈도우 운영체제를 기반으로 하는 웜 바이러스는 다음과 같은 방법으로 전파된다.

- WSOCK32.DLL 변형(I-Worm/Happy99)
- 특정 애플리케이션 이용(Outlook Express)
- SMTP 서버 내장(I-Worm/Cholera)
- 윈도우 공유 폴더 감염(I-Worm/Qaz)

윈도우 기반의 웜 바이러스인 *I-Worm/Happy99*는 *WINSOCK32.DLL*을 변형해서 메일을 보낼 때 웜 바이러스가 포함된 파일을 첨부하는 형태를 취한다. 이는 *WINSOCK32.DLL*이 윈도우에서 인터넷과 관련된 일을 수행할 때 사용된다는 특성을 이용한 것이다.

본 논문에서는 이러한 웜 바이러스의 감염에 대비하기 위하여 신종 웜 바이러스를 탐지할 수 있는 방안을 유전자 알고리즘을 적용하여 제시하고자 한다.

2. 스크립트형 웜 바이러스 패턴 정의 및 추출

스크립트 내부의 악성 행위 모듈을 파악하기 위해서는 이를 비교·판단하기 위한 일정한 패턴이 필요하다. 이에 전처리 과정의 하나로 VBScript로 작성된 웜 바이러스 구문을 파악하여 함수 및 그 인자를 분류하여 패턴 규칙을 정의한다. 이를 위해서 웜 바이러스의 악성 행위의 분류가 필요하고, 이들 행위와 함수간의 관계를 정의할 필요성이 있다.

스크립트에서 각종 객체와 메소드, 그리고 각종 기호를 추출한 후 스크립트에 기술된 행위를 분석하여 패턴으로 분류하고 그 패턴은 다시 정규식을 거쳐 정형화된 형태로 바뀌게 된다. 본 절에서는 스크립트형 웜 바이러스의 구문을 정규화하여 유전자 알고리즘에 적용할 수 있는 부호화 방법(encoding scheme)을 제시하고자 한다.

2.1 바이러스의 악성 행위의 부호화 방안

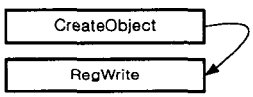
2.1.1 스크립트 행위 분석

일반적으로 스크립트형 웜 바이러스가 지정된 행위를 수행하기 위해서는 먼저 객체를 생성하고 그 객체에 대한 메소드를 호출한다. 이때, 웜 바이러스에 의해 생성되는 윈도우 객체들은 다음과 같다[3].

- *Scripting.FileSystemObject*
- *WScript.Shell*
- *WScript.Network*
- *Outlook.Application*

이러한 객체 가운데 *Outlook.Application*은 Outlook이 설치된 시스템에만 존재하고 나머지 객체들은 WSH(Windows Script Host)가 설치되어 있는 시스템에 항상 존재한다. 대표적인 웜 바이러스인 *'Love Letter'*의 특징은 윈도우 레지스트리 조작, E-mail을 통한 복제, 특정한 파일의 삭제, 그리고 악성 HTML 파일을 생성한다. *'Love Letter'* 바이러스의 행위 패턴을 사용된 함수에 의하여 표현하면 아래와 같다.

(1) 윈도우의 레지스트리를 조작하여 시작 프로그램에 바이러스 경로를 삽입한다.

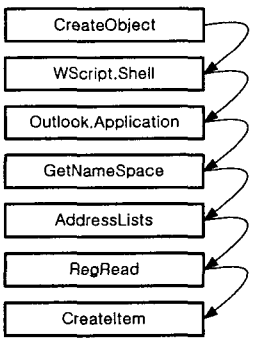


부팅할 때마다 매번 실행되도록 시작프로그램에 해당하는 레지스트리 값을 변경한다.

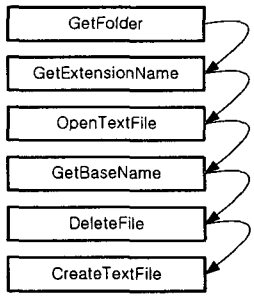
-HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

-HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices

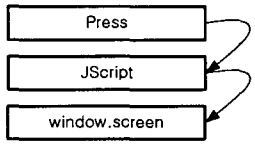
(2) 윈도우의 Outlook Express의 기본 주소록 파일인 WAB(Windows Address Book)에 등록된 주소를 엑세스하여 복제된 메일을 전송한다.



(3) 로컬 시스템의 특정한 파일을 검색하거나 특정 파일을 복사 및 삭제한다.



(4) 악성 HTML 파일을 생성한다.



2.1.2 패턴 규칙 이진화

앞서 제시된 스크립트 내부의 윈도우 객체를 제어하는 함수를 분류하면 하나의 제어규칙을 표시할 수 있다. 예를 들어, 악성 행위 코드에 포함된 함수가 *RegWrite*(레지스트리 입력) 값으로 지정되고, *RegWrite* 함수에 사용된 인수가 $\{P_1, P_2, P_3\}$, *DeleteFile*(파일 삭제) 함수는 $\{Q_1, Q_2, Q_3\}$의 인수가 포함되어 있는 순서화된 이산적인 값을 갖는다면, 이 규칙은 다음과 같은 패턴 규칙으로 표현할 수 있다.

[IF (*RegWrite* = P_1 or P_2) AND (*DeleteFile* = Q_1) THEN *Pattern* = 0]

이 패턴 규칙의 조건부는 AND(Conjunction)로 이루어져 있고, 함수와 인자의 특징을 내부적인 OR(Internal Disjunction)로 구성된다. 이를 유전자 알고리즘에 적용될 염색체로 표현을 하면 다음과 같다.

RegWrite	*DeleteFile*	*Pattern*
0 1 1	1 0	0

이를 토대로 유전자 알고리즘을 적용할 염색체의 표현을 함수와 인자 사이의 상관관계를 이용하여 이진화시키고자 한다.

3. 유전자 알고리즘을 적용한 탐지 시스템 설계

3.1 유전자 알고리즘(Genetic Algorithms)

유전자 알고리즘은 탐색공간에서 여러 개체들에

대해 자연 선택과 자연 유전을 응용, 유전 연산을 적용시켜 개체들을 진화시키는 최적화 알고리즘이다 [2]. 유전자 알고리즘은 주어진 조건에 잘 부합하는 답을 표현하는 문자열로 인코딩하여 개체들을 군집으로 표현한 후 교배, 돌연변이와 같은 유전 연산자의 선택과 재생산의 과정을 반복하여, 좋은 특성들이 다른 좋은 특성들과 섞이거나 교환되면서 개체 집단 전체에 퍼져나가게 한다. 유전자 알고리즘의 동작 개요는 다음과 같다.

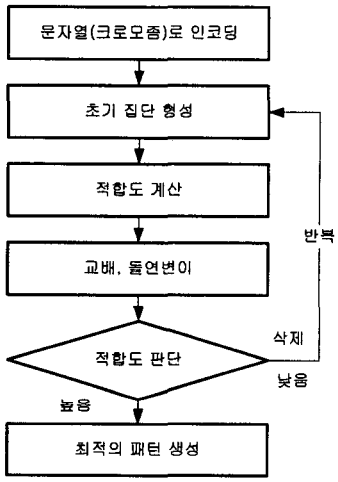


그림 1 유전자 알고리즘 수행과정

3.2 알고리즘 적용을 위한 규칙 집합의 표현

유전자 알고리즘을 적용하기 위한 규칙 집합의 생성은 크게 두 가지로 분류할 수 있는데 첫 번째 방법은 유전자 알고리즘의 연산자를 변형시켜 탐색의 효율을 높이는 방법인데 이는 효율적인 탐색의 장점을 가지고 있지만, 응용 문제의 도메인 지식을 정확히 나타내는 새로운 유전 연산자를 생성하는데 어려움이 따른다. 두 번째 방법은 전통적인 유전자 알고리즘의 형태에 최소한의 변화로 유전자 알고리즘의 효율적이고 적응이 뛰어난 탐색 특성을 유지할 수 있도록 하는 염색체의 표현 방법을 만드는 시도이다. 이는 복잡한 규칙집합을 선형 스트링(염색체)에 매핑(Mapping)하는 방법을 찾기 위해 설계 시 많은 부담을 안게 되지만, 효과적인 매핑 방법을 찾게 되면 시스템에 약간의 변화를 줌으로써 유전자 알고리즘의 효율적이고 적응이 용이한 탐색 특성을 얻을 수 있다.

본 연구에서 악성 패턴을 염색체로 표현하기 위해서는 스크립트 내부의 함수 및 인수 분포를 기반으로 출현 순서 및 값의 유무를 기준으로 패턴을 생성하였다. 이를 고정된 길이의 규칙의 패턴을 하나의 염색체에 표현함으로써 함수 및 인수 집합을 염색체에 사상하는 방법이다. 각 패턴의 표현의미는 다음과 같이 설명될 수 있다.

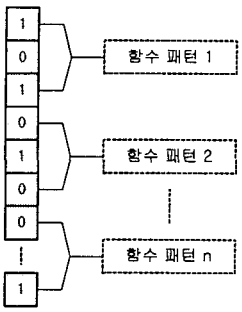


그림 2 함수 패턴 집합

이러한 규칙 집합을 염색체로 표현한다면 그림 2와 같이 표현될 수 있다. 이는 스크립트 구문 내의 악성 행위를 할 가능성이 있는 함수의 유무와 관련이 있기 때문에 스크립트 구문에서 발견된 각각의 악성 행위 규칙을 패턴화 하여 학습할 수 있다. 하나의 바이러스 염색체 집합은 악성 행위를 할 수 있는 스크립트형 바이러스 코드 전체에 대응된다.

Virus1	→	0 0 1 0 1 0 0 1 0 1 1 1 0 0 1 0 1 0 0 1
Virus2	→	1 1 0 0 1 0 1 1 0 0 0 0 1 1 1 1 0 1 1 0
Virus3	→	1 0 0 0 1 1 0 1 1 0 1 0 0 0 0 1 0 0 0 0
Virus4	→	0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 1 0 0 1 1
Virus5	→	1 0 1 0 1 1 1 1 1 0 1 1 1 0 1 1 0 1 1 0
Virus6	→	0 1 0 0 1 0 0 1 1 0 0 1 1 0 0 1 0 0 0 0
Virus7	→	1 0 1 0 1 1 1 0 0 1 0 0 0 1 1 0 1 0 1 0
Virus8	→	1 1 0 0 1 0 0 1 0 1 0 0 0 0 1 1 0 1 1 0
Virus9	→	0 1 0 0 1 1 1 1 0 0 1 1 1 0 1 1 1 0 0 0
Virus10	→	1 1 1 0 1 1 1 0 0 1 0 1 0 1 0 1 0 0 1 1
Virus11	→	1 0 0 0 1 0 0 1 1 0 0 0 1 0 1 1 0 0 1 0
Virus12	→	1 1 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 0 0
Virus13	→	0 0 0 0 1 0 0 1 0 0 1 1 0 1 1 1 1 0 1 1
Virus14	→	1 1 0 0 1 0 1 1 1 0 0 1 1 0 1 0 0 0 1 0
Virus15	→	0 0 1 0 1 1 0 0 0 1 1 0 0 1 1 1 0 1 1 0

그림 3 스크립트형 바이러스 패턴의 초기 집단

3.3 적합도 함수

염색체의 적합도 함수는 다음 세대의 개체군을 형성하는데 중요한 척도가 된다. 즉, 적합도가 높은 염색체일수록 선택될 확률이 높고 다음 세대의 개체군을 형성하는데 많은 영향을 미친다. 본 논문에서는 염색체의 적합도 함수를 유도하기 위해서 분류된 패턴에 대한 비율을 적합도 상수로 사용하였다. 이는 스크립트 내부의 함수 패턴의 빈도수를 나타내는 것이다.

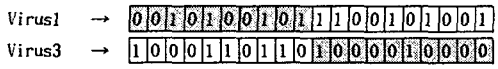
3.4 규칙 데이터 베이스 생성을 위한 유전자 알고리즘 적용

앞장에서 표현된 스크립트 패턴은 하나의 규칙 베이스이다. 본 논문에서는 새로운 패턴 규칙을 생성하기 위해 유전자 알고리즘의 교배 연산(Crossover)

과 돌연변이 연산(Mutation)을 수행한다.

3.4.1 교배 연산(Crossover)

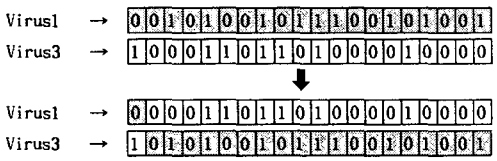
악성 행위에 대한 패턴을 교배점을 기준으로 패턴의 오른쪽 유전자들을 교환하여 새로운 패턴을 생성하고자 한다. 그림 7에서 생성된 초기 집단에서 *Virus1*과 *Virus3*가 교배대상으로 선택되고, 임의의 교배점으로 11번째의 유전자를 사용한다면 교배점의 오른편 부분을 서로 교환하여 새로운 패턴을 생성하게 된다.



여기에서 교배확률을 P_c=0.25로 설정한다. 이는 평균적으로 유전자의 25%가 교배 연산의 대상이 된다고 할 수 있다. 이를 위해 교배 연산에서는 다음과 같은 알고리즘이 수행된다.

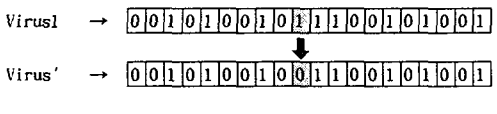
```
k=0
  while( k < 10 ) {
      r_k = [0,1]사이의 난수값;
      if ( r_k < 0.25)
            virus k를 교배 대상으로 선택
      k++;
  }
```

본 연구에서 초기집단의 길이는 20임으로 고려하여 교배점으로 [1, 19] 사이의 정수형 난수값을 구한다. 예를 들어 생성된 난수값이 1이라면 다음과 같이 첫 번째 비트 이후의 부분을 교환하여 새로운 패턴을 생성한다.



3.4.2 돌연변이 연산(Mutation)

돌연변이 연산은 돌연변이 확률과 동일한 확률로 하나 혹은 그 이상의 유전자를 바꾼다. 바이러스 패턴 *Virus1*의 10번째 비트가 돌연변이 대상으로 선택되었다면, 이 비트 1은 돌연변이를 일으켜 0이 된다.



만약 돌연변이 확률 P_m을 0.01로 설정한다면 이는 평균적으로 바이러스 패턴 내에서 모든 유전자가 1%정도의 돌연변이 연산의 대상이 된다고 할 수 있다. 본 논문에서는 바이러스 패턴의 총 유전자 수, 즉 비트의 총 개수는 *Virus m* × *Pattern size* = 15 × 20 = 300이므로 각 패턴마다 약 3회의 돌연변이가 발생한다고 볼 수 있다. 따라서 돌연변이 과정을 수행하기 위해 [0, 1]사이의 값을 갖는 300개의 임의의 난수 r_k(k = 1,2,…,300)를 생성시킨다. 이 중 0.01보다 작은 값을 가지는 난수 위치와 대응되는 비트가 돌연변이 대상이 된다.

대상 위치	유전자 번호	대상 비트	발생 난수
109	5	3	0.006524
180	8	12	0.004659
287	10	18	0.000123

돌연변이 후 최종적으로 발생한 규칙 패턴은 다음과 같다.

그림 4 유전자 알고리즘이 적용된 바이러스 패턴

그림 4에서와 같이 유전자 알고리즘을 적용하여 생성된 각각의 패턴은 새로운 스크립트형 웜 바이러스일 가능성이 높다. 따라서 스크립트형 파일을 정규화한 후 유전자 알고리즘을 적용시켜 파생된 패턴을 다른 패턴과 비교함으로써 패턴의 유사도 및 출현 빈도수를 산출하여 바이러스일 가능성을 제시한다.

5. 결론 및 향후 연구과제

본 논문에서는 알려진 형태의 스크립트형 바이러스뿐만 아니라 알려지지 않는 혹은 변형된 바이러스를 탐지할 수 있는 방안을 제시하였다. 이는 급속히 퍼지고 코드의 변형 주기가 빠른 웜 바이러스에 대해 기존의 출현 후 분석하는 대응에서 벗어나 보다 적극적으로 웜 바이러스를 대처할 수 있는 방안이 될 수 있을 것으로 본다. 향후에는 보다 정확한 패턴의 규칙과 함수뿐만 아니라 다른 요소를 이용한 패턴 적용에 대한 연구가 필요하다.

참고문헌

[1] Jeff Kephart "Automatic Extraction of Computer Virus Signatures" Virus Bulletin International Conference, 1994, Pages 178~184

[2] M.Mitchell, " An Introduction to Genetic Algorithms", MIT Press, 1996

[3] Mark Kennedy "Script-Based Mobile Threats" Symantec AntiVirus Research Center, 2000

[4] "Prevalence Tables" Virus Bulletin, January~May 2001, Page 3