韓國컴퓨터情報學會 論文誌
第17卷 第3號, 2012. 3.

2012-17-3-3-2

SVM을 활용한 악성 웹 페이지 분류

황 영 섭*, 문 재 찬**, 조 성 제**

Classification of Malicious Web Pages by Using SVM

Hwang Young-Sup*, Moon Jae-Chan**, Cho Seong-Je**

요 약

웹 페이지에서 다양한 서비스를 제공하면서 악성코드가 웹 페이지를 통해 배포되는 것도 늘어났다. 악성코드는 개인정보 유출, 시스템의 성능저하, 시스템의 좀비 피씨화 등의 피해를 입힌다. 이런 피해를 막으려면 악성코드가 있는 웹 페이지의 접근을 막아야 한다. 그런데 웹 페이지에 있는 악성코드는 난독화나 변형기법으로 위장하고 있어 기존 안티바이러스 소프트웨어가 사용하는 시그니처 방식의 접근법으로 찾아내기 어렵다. 이를 해결하기 위하여, 웹 페이지를 분석하여 악성 웹 페이지와 양성 웹 페이지를 구별하기 위한 특징을 추출하고, 기계 학습법으로 널리 사용되는 SVM을 통하여 악성 웹 페이지를 분류하는 방법을 제안한다. 제안하는 방법이 우수함을 실험을 통하여 보인다. 제안한 방법으로 악성 웹 페이지를 정확히 분류하면 웹 페이지를 통한 악성코드의 배포를 막는데 이바지할 것이다.

▸ Keyword : 악성 웹 페이지, SVM, 악성 코드, 난독화, 기계학습

Abstract

As web pages provide various services, the distribution of malware via the web pages is being also increased. Malware can make personal information leak, system mal-function and system be zombie. To protect this damages, we should block the malicious web pages. Because the malicious codes embedded in web pages are obfuscated or transformed, it is difficult to detect them using signature-based approaches which are used by current anti-virus software. To overcome this problem, we extracted features to classify malicious web pages and benign ones by analyzing web pages. And we propose a classification method using SVM which is widely used in machine learning. Experimental results show that the proposed method is better than other methods. The proposed method could classify malicious web pages correctly and be helpful to block the

• 제1저자 : 황영섭
• 투고일 : 2011. 11. 04, 심사일 : 2011. 12. 07, 게재확정일 : 2012. 01. 06.
* 선문대학교 컴퓨터공학과(Dept. of Computer Science and Engineering, Sun Moon University)
** 단국대학교 소프트웨어학과(Dept. of Computer Science and Software Science, Dankook University)

distribution of malicious codes.

▸ Keyword : Malicious web page, SVM, malware, obfuscation, machine learning

I. 서 론

인터넷이 널리 활용되면서 인터넷에서 웹 브라우저를 통해 다양한 서비스를 제공하게 되었다. 웹 브라우저가 사용자와 상호 작용하면서 서비스를 제공하려면 웹 페이지에 실행 코드가 들어 있어야 한다. 웹 페이지의 실행 코드는 보통 스크립트로 작성하는데 여기에 취약점이 있어서 악성코드를 배포하는 도구로 웹 페이지가 많이 활용되고 있다[1, 2]. 2010년에 조사된 전체 취약점 중에 49%가 웹 응용 프로그램의 취약점이다[3]. 악성코드는 개인정보의 유출을 불러오고, 시스템 자원을 소모하며 심한 경우, 시스템이 오작동하거나 시스템을 파괴하기도 한다. 이러한 악성코드의 피해를 줄이려면 악성코드가 들어있는 웹 페이지를 미리 탐지하여 방문하지 않도록 하여야 한다.

악성코드가 들어있는 악성 웹 페이지를 찾아내기 위해 안티 바이러스 소프트웨어를 사용할 수 있다. 대부분의 안티 바이러스 소프트웨어는 이진 실행코드를 시그니처와 비교하여 탐지하는 방법을 사용한다[4-6]. 그런데 웹페이지를 변경하는 것은 실행코드를 바꾸는 것보다 쉬우므로 빈번한 변경이 일어나서 시그니처의 변경이 상대적으로 느린 안티 바이러스 소프트웨어의 탐지 능력이 떨어지게 된다[2].

악성코드가 웹 페이지에서 유포되는 패턴으로 iframe 태그를 이용하면서 width 값을 0으로 하여 브라우저가 알 수 없게 하거나 EMBED, OBJECT, SCRIPT, LINK 등의 태그를 활용하는 방법, 자바스크립트 함수를 삽입하는데 문자열을 바로 알아보지 못하게 난독화(obfuscation)하는 방법 등이 사용된다. 스크립트 함수의 경우 다양한 문자열 조작 함수를 통해 최종 악성코드를 만들어내므로 단순한 시그니처 방식으로 탐지하기 어렵다[7-10].

본 연구는 악성 웹 페이지를 탐지하기 위하여 악성 웹 페이지에서 공통으로 발견되는 특징을 조사하고, 이 특징을 기계학습 방법으로 학습하여 악성 웹 페이지를 분류하는 방법을 연구한다. 악성코드가 일반적으로 가지는 특성들을 추출하고, 이들을 상대적으로 비교하기 쉬우며 악성코드의 특성을 더 드러내도록 서로 조합하였다. 조합한 특징으로 악성/양성 웹 페이지에 대해 분석하여 제안한 특징의 우수함을 조사하였다. 특징만으로 악성 웹 페이지를 분류하는데 한계가 있으므로 더 나은 분류율을 얻기 위하여 기계학습 방법을 연구하였다. 기계학습법 중에 일반화 능력이 우수하여 최근 널리 사용하는 SVM(support vector machine)을 채택하여 학습시켰으며 실험을 통하여 제안한 방법의 우수함을 보였다.

II. 관련 연구

악성코드를 탐지하기 위한 정적분석이나 동적분석에 대한 연구가 많이 이루어졌으나 악성코드가 숨어있는 웹 페이지의 분석에 대한 연구는 미진한 편이다. 이 절에서 악성 웹 페이지의 분석 연구 중에 대표적인 연구를 제안한 방법과 비교하여 설명한다.

최영한팀은 악성 자바 스크립트를 분석하여 N-gram, 엔트로피, 단어크기의 3가지 척도를 개발했다[11]. 이 방법으로 난독화된 스크립트를 효과적으로 찾아내었다. 자바 스크립트의 eval()과 document.write()함수를 위험한 함수로 가정한다. 그런데 Feinstein과 Peck의 연구에 따르면 악성 웹 페이지보다 양성 웹 페이지에서 document. write()가 더 많이 사용되고, eval()함수도 흔히 사용된다[12]. 본 연구에서는 단순한 척도만이 아니라 척도를 조합한 후, 기계 학습을 통하여 분류율의 향상을 꾀하였다.

Seifert팀은 악성 웹 페이지를 분류하기 위한 휴리스틱 방법을 제시했다[13]. 기계 학습법의 하나인 결정 트리를 이용하였는데 악성 웹 페이지를 찾지 못하는 오탐율이 46.15%로 높게 나왔다. 오탐율이 높은 이유는 학습에 사용한 특징이 충분하지 않거나 단순한 결정 트리에서 오는 것으로 보인다. 본 연구는 더 우수한 특징과 기계학습을 통한 분류율 향상이 목표이다.

Yung-Tsung Hou팀은 결정트리, Naive Bayes, SVM, Boosted 결정트리의 4가지 기계학습 방법으로 악성 웹 페이지를 분류하는 연구를 수행했다 [2]. 154개 자바함수의 사용 횟수, 9가지 HTML 문서 특징, 8가지 ActivX 객체의 사용 횟수를 특징으로 사용하였다. 실험결과 Boosted 결정트리가 가장 좋은 결과를 보였다. 이런 실험결과가 나온 이유는 학습에 사용한 데이터가 176개로 충분하지 않고, 학습에 사용한 데이터로 그대로 테스트에 사용했으므로 학습 데이터에 최적화된 Boosted 결정 트리가 좋은 결과를 낸 것으로 보인다. 본 연구는 기본특징을 조합하여 상대적인 특징을 만들고, 충분히 많은 데이터로 학습을 하여 분류율을 향상시켰다.

III. 특징 추출

3.1 웹 페이지 분류를 위한 척도

한국인터넷진흥원(KISA)에서 수집한 2,604개의 양성 웹 페이지와 444개의 악성 웹 페이지[14]를 분석하여 악성 웹 페이지를 찾는데 도움이 되는 특징을 14개 추출하였다.

- SoF(Size of File) : 파일의 크기, 악성 웹페이지와 정상 웹페이지의 크기가 통계적으로 크게 다르지 않다. 확장 특징에서 비율을 구하기 위한 요소로 사용된다.
- NANC(Number of Non-Alphanumeric Characters) : 특수문자의 수, 악성 웹 페이지에서 정상 웹 페이지보다 많이 발견된다. SoF와 마찬가지로 확장 특징을 구하기 위한 요소로 사용된다.
- HT(Number of HTML Tags) : HTML 태그의 수, 확장 특징에서 HTML Tag와 JavaScript의 keyword 수의 비율이 악성과 정상사이에 큰 차이가 나타난다.
- CoS(Number of Operators to Concatenate Strings) : 문자열 접합 연산자 수, 악성 웹페이지를 분석하면 문자열이 조합이 되는 것을 많이 발견할 수 있다.
- UDF(Number of User Defined Functions) : 사용자 정의 함수의 수, JavaScript가 제공하는 인코딩 함수를 사용하지 않고, 직접 사용자가 정의한 함수를 통해 인코딩을 하는 경우를 탐지한다.
- JSKEY(Number of JavaScript Keywords) : 자바 스크립트 키워드 수, 자바스크립트의 길이가 아닌 자바스크립트 내에 워드가 얼마나 쓰였나를 통해 자바스크립트의 길이를 탐지한다.
- LoS(Length of Script) : 스크립트의 길이, LoS/SoF 로 확장 특징에 사용된다.
- Func(Number of Functions) : 함수의 수, 악성 페이지들이 악성행위를 하기 위해 많은 함수를 사용한다.
- Var(Number of Variables): 변수의 수, Func과 유사한 이유로 선정하였고, 문자열 조합을 변수를 통해 행하는 경우가 있으므로 이를 탐지한다.
- Oper(Number of Operators): 연산자의 수, 악성 웹페이지가 정상 웹페이지보다 CoS 및 기타 연산이 많다.
- Const(Number of Constants) : 상수의 수, 악성 웹페이지 내에 반복문 등이 많아 상수형이 많이 사용된다.
- NoL(Number of Literals) : 문자열의 수, CoS 탐지와 연관성이 있고, 악성 웹페이지가 일반적으로 스크립트를 난독화한 문자열을 이용하는 것을 탐지한다.
- TLoL(Total Length of Literals) : 문자열 전체의 길이, NoL, CoS 등 문자열 탐지 와 관련된 것이고, NoL과 마찬가지로 난독화된 문자열을 탐지한다.
- DeFunc(Number of Decoding Functions) : 해독 함수의 수, 악성 웹페이지 중 난독화를 위해 JavaScript가 제공하고 있는 인코딩/디코딩 함수를 사용하는 경우를 탐지한다.

CoS, NoL, TLoL은 다음 예처럼 계산한다.

```
Var String = “aaa” + “bbbb”+“cccc”;
// CoS = 2
// NoL = 3
// TLoL = 11
```

표 1에 14개 기본 특징을 요약하였다.

표 1. 웹 페이지 분류를 위한 기본 척도
Table 1. Basic metrics to classify webpage

척도	설명
SoF	파일의 크기(Size of File)
NANC	영문자, 숫자가 아닌 문자의 수(Number of Non-Alphanumeric Characters)
HT	HTML 태그의 수(Number of HTML Tags)
CoS	문자열 접합 연산자 수(Number of Operators to Concatenate Strings)
UDF	사용자 정의 함수의 수(Number of User Defined Functions)
JSKey	자바 스크립트 키워드 수(Number of JavaScript Keywords)
LoS	스크립트의 길이(Length of Script)
Func	함수의 수(Number of Functions)
Var	변수의 수(Number of Variables)
Oper	연산자의 수(Number of Operators)
Const	상수의 수(Number of Constants)
NoL	문자열의 수(Number of Literals)
TLoL	문자열 전체의 길이(Total Length of Literals)
DeFunc	해독 함수의 수(Number of Decoding Functions)

웹 페이지의 크기는 모두 다르므로 척도에 따라 계산한 숫자의 절대 크기는 중요하지 않다. 예를 들어, 스크립트의 길이는 5,000이상일 수 있지만 함수의 수는 수백 개 이내이다. 웹 페이지의 크기에 상관없는 비교를 위하여 기본 척도를 활용하여 표 2와 같은 16가지 확장 척도를 개발하였다.

표 2. 확장 척도
Table 2. Extended metrics

	척도
1	LoS/SoF
2	NANC/LoS
3	CoS/Oper
4	Func/Var
5	TLoL/CoS
6	TLoL/NoL
7	TLoL/LoS
8	Const/Var
9	Const/Oper
10	JSKey/HT
11	DeFunc/Func
12	Var/UDF
13	Func/LoS
14	Const/LoS
15	Var/LoS
16	NANC/TLoL

확장 척도는 표1의 기본 척도들 사이의 비율이며 분류에 도움이 되도록 선택하였다. 분모가 0인 경우 척도값을 0으로 처리한다. 선택하는 기준은 3.2절의 웹 페이지 분석을 참고하여 휴리스틱으로 선정하였다.

3.2 웹 페이지 분석

표 6. 웹 페이지 분석 결과
Table 3. Analysis results of webpages

	LoS/SoF	NANC/LoS	CoS/Oper	Func/Var
양성	0.130	0.133	0.094	0.398
악성	0.832	0.174	0.202	45.400
	TLoL/CoS	TLoL/NoL	TLoL/LoS	Const/Var
양성	71.372	29.036	0.138	0.830
악성	771.745	254.718	0.520	2.405
	Const/Oper	JSKey/HT	DeFunc/ Func	Var/UDF
양성	0.051	0.519	0.044	2.552
악성	0.154	8.037	0.386	10.408
	Func/LoS	Const/LoS	Var/LoS	NANC/TLoL
양성	0.004	0.002	0.006	0.137
악성	0.009	0.009	0.007	0.139

KISA에서 제공한 2,604개 양성 웹 페이지와 444개 악성 웹 페이지를 확장 척도에 따라 분석한 결과를 그림 1과 표 3에 보인다.

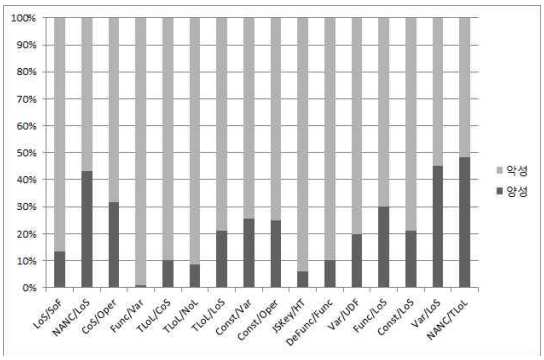

그림 1. 웹 페이지 분석 결과
Fig. 1. Analysis results of webpages

그림 1은 악성(Black) 웹 페이지와 양성(White) 웹 페이지에서 계산된 확장 척도의 평균값의 상대적 비율을 나타낸다. 예를 들어 표 3의 NANC/LoS는 양성 평균이 0.133과 악성 평균이 0.174로 서로 비슷하고, 그림 1의 막대그래프의 비율이 비슷하다. 한편 LoS/SoF는 상대적으로 차이가 크다. 그림 1을 검토하여 보면 제안한 특징만으로 높은 분류율을 얻기 어려우며, 특징의 상대적 비교가 필요함을 알 수 있다. 이러한 과정을 자동으로 하는 방법으로 기계학습법이 있으며, 기계학습법에서 가장 대표적인 SVM을 분류기로 채택한다.

IV. 분류기의 설계와 실험

4.1 SVM

SVM에 대하여 간단히 소개하면 다음과 같다. SVM은 부류를 분류하는 여백을 크게 하는 분류 초평면을 찾아낸다. 특징의 크기가 커도 차원의 저주가 없고, 일반화 능력이 뛰어나 널리 사용되고 있다. 학습데이터 $(\mathbf{x_i}, y_i), i = 1, \cdots, l$, $\mathbf{x}_i$는 입력 특징 벡터, y_i는 원하는 출력(클래스)이 주어졌을 때, SVM은 다음 최적화 문제의 해답을 구한다.

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}} \left[\frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \right],$$

$$y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i,$$
$$\xi_i \geq 0.$$

C는 오류율에 영향을 주는 상수이다. SVM은 주어진 문제가 선형 분리 가능하지 않을 경우, 커널 함수 대치를 통하

여 문제를 해결할 수 있다.

$$K(\mathbf{x}_i,\mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$$

커널 함수로 주로 다음과 같은 RBF(Radial Basis Function)가 사용된다.

$$K(\mathbf{x}_i,\mathbf{x}_j) = \exp(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2), \quad \gamma > 0$$

이 경우 결정해야 하는 매개 변수가 오류율을 결정하는 상수 C와 RBF 함수의 형태를 결정하는 상수 g이다. 두 매개변수를 결정하는 일반적인 방법은 없으며 실험을 통하여 결정해야 한다[15].

SVM은 학습과정에서 여백을 크게 하여 일반화 능력을 향상 시키며, 여백을 결정하는 support vector를 찾아내어 저장한다. 학습이 끝나고 학습하지 않은 데이터에 대하여 테스트할 때, 저장된 support vector를 읽어 들여 분류 함수를 빠르게 계산할 수 있다.

4.2 데이터 준비

SVM을 학습시키기 위하여 양성 데이터를 250개, 악성 데이터를 250개씩 선정하였다. 여기에서 데이터는 3.1절의 확장척도에 따라 웹 페이지에서 추출한 데이터이다. 학습 데이터가 두 부류의 모든 경우를 포함하도록 추출되면 이상적이지만 미리 알 수 없으므로 순서대로 250개를 선정하였다. 확보한 데이터가 양성이 훨씬 많지만 악성 웹 페이지를 잘 찾아내는 것이 양성 웹 페이지를 악성으로 오판하는 것보다 나으

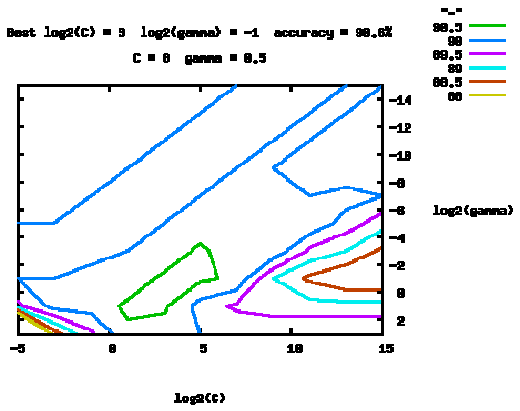

그림 2. SVM의 매개변수 결정
Fig. 2. Finding Parameters of SVM

므로 두 부류의 학습 데이터의 비율을 같게 하였다. 학습에 사용하지 않은 나머지 데이터(양성: 2,354, 악성: 194)를 테스트 데이터로 활용하였다.

4.3 SVM 학습

SVM은 매우 복잡한 수식을 풀어야하므로 구현이 어렵지만 SVM을 잘 구현한 공개 소프트웨어가 있으므로 이를 활용하면 된다. 여기서는 다양한 인터페이스를 제공하는 libsvm을 사용한다[15].

표 3의 데이터를 보면 악성 웹 페이지에 대한 값이 NANC/LoS는 0.174의 값을 가지지만 TLoL/Cos는 771.745를 가져서 상대적으로 값이 크다. 이런 특징을 그대로 사용하면 상대적으로 값이 큰 특징에 많은 영향을 받게 되어 분류율이 떨어진다. 이를 해결하기 위하여 16개의 특징값을 모두 0과 1사이의 값으로 정규화하였다.

4.2절의 데이터를 libsvm이 요구하는 형식에 맞게 바꿔주었다. RBF 커널 함수에 필요한 매개 변수 C와 g를 libsvm이 제공하는 grid 툴을 사용하여 결정하였다. 그리드 방식은 매개 변수를 아주 큰 값을 주어 학습을 시키고, 그 다음 매우 작은 값을 주어 학습을 시켜서 그 결과를 비교하여 결과가 좋아지도록 큰 값은 작게 하고, 작은 값을 크게 하면서 가장 최적의 매개 변수 값을 찾아 나가는 방식이다. 학습결과 C는 8.0, g는 0.5로 결정되었고(그림 2), 이때 support vector의 개수는 116개였다.

V. 실험결과

4.3절에서 학습시킨 SVM으로 학습에 참여하지 않은 테스트 데이터에 대해 실험한 결과는 다음과 같다.

표 7. 실험 결과
Table 4. Experimental Results

	양성	악성	양성	악성
양성	2221	133	94.4%	5.6%
악성	18	176	10.3%	89.7%

실험 결과 악성을 양성으로 잘못 분류한 FP(false positive)가 10.3%이며, 전체 분류율은 94.1% (2,397/2,548)이다. 이 결과를 보고된 다른 결과와 비교하면 다음과 같다.

표 8. 실험 결과
Table 4. Experimental Results

	FN	FP
제안한 방법	5.6%	10.3%
이진영[14]	3.9%	14.4%
김병익[8]	3.8%	12.1%
Peter Likarish[9]		10.5%
Yung-Tsung Hou[2]	7.4-14.8%	0.2-7.6%

Peter Likarish는 4가지 분류기를 비교하였는데 그 중 SVM이 학습 데이터에 대하여 성능이 좋았고, 테스트 데이터에 대한 결과는 FP가 13.7%이었다. 테스트 데이터에 대해 결과가 좋은 분류기는 베이시언 분류기로 FP가 10.5%였다. 양성 데이터를 악성으로 오분류하는 FN(false negative) 값은 제시되지 않았다. Yung-Tsung Hou는 FP를 줄이는 대신 FN이 높게 나오도록 학습시켜서 FP가 0.2일 때, FN이 14.8%로 높게 나왔다. 다른 연구결과와 제안한 방법의 실험결과를 비교하여 보면 제안한 방법의 FP가 10.3%으로 매우 양호함을 알 수 있다. 표 5의 각 연구결과가 서로 다른 데이터를 사용하여 객관적인 비교는 어렵지만 대략의 경향을 확인할 수 있다. 실험을 통해 제안한 16개 확장 특징이 악성 웹 페이지를 분류하는데 유용한 특징이며 SVM의 일반화 능력을 활용하면 FP를 낮출 수 있음을 알 수 있다.

Ⅵ. 결론

악성 웹 페이지를 탐지하기 위하여 양성 웹 페이지와 악성 웹 페이지를 분석하여 이 둘을 분류하기 위한 기본 특징 14개를 추출하고, 웹 페이지의 크기에 영향을 받지 않도록 기본 특징 14개를 조합한 확장 특징 16개를 제시하였다. 확장 특징 16개를 입력으로 하여 일반화 능력이 우수한 SVM을 학습시켜서 학습하지 않은 테스트 데이터에 대해 실험하였다. 실험 결과 분류율이 94.1%이며, 악성 웹 페이지만을 대상으로 했을 때, 89.7%를 찾아내었다. 실험결과가 기존 방법보다 우수하여 제안한 특징과 SVM이 악성 웹 페이지 분류에 유용함을 알 수 있다.

제안한 방법에 의하여 악성 웹 페이지를 미리 분류하여 탐지할 수 있으며, 이를 통해 사용자에 의한 악성 웹 페이지로의 방문을 차단하여 악성코드가 전파되는 것을 막을 수 있다.

악성코드가 끊임없이 변이하고 진화의 속도가 빠르므로, 이미 학습한 분류기라도 새로운 악성 웹 페이지를 제대로 분류할 수 없는 경우가 발생할 수 있다. 이에 대한 대처로 새로운 데이터에 대해 적응적으로 학습해 나가는 분류기에 대한 연구가 필요하다.

기본 특징을 조합하는 방법은 C(14,2)= 91가지가 있다. 이 중 몇 개를 어떻게 선택하는 것이 좋은지에 대한 좀 더 정교한 연구가 필요하다. 제안한 확장척도는 휴리스틱에 의해 결정되었지만 특징선택법에 의해 선택하면 더 좋은 결과를 얻을 수 있을 것이다.

참고문헌

[1] L.C. Tae, J.H, Oh and H.C. Jeong, “Study of the Technique Trend and Analysis Method of Recent Malaware,” Communications of the KIISE, Vol.28, No.11, pp.117-125. Nov. 2010.

[2] Y.-T. Hou, Y. Chang, T. Chen, C.-S. Laih and C.-M. Chen, “Malicious web content detection by machine learning,“ Expert Systems with Applications, Vol.378, pp.55-60, 2010.

[3] IBM X-Force Team, “IBM X-Force 2010 Trend and Risk Report”, IBM Published, March, 2011

[4] ByungHa Choi and Kyungsan Cho, “An Improved Detecting Scheme of Malicious Codes using HTTP Outbound Traffic,” Journal of the KSCI, Vol.14, No.9 pp.47-54, Aug. 2009. (in Korean)

[5] Hee-Hwan Park and Dea-Woo Park, “A Study on Treatment Way of a Malicious Code to injected in Windows System File,” Journal of the KSCI, Vol.14, No.2, pp.255-262, De. 2006. (in Korean)

[6] Chong-Woo Woo and Kyoung-Hui Ha, “A Development of Malware Detection Tool based on Signature Patterns,” Journal of the KSCI, Vol.10, No.6, pp.127-136, De. 2005. (in Korean)

[7] N. Proves, D. McNamee, et al., “The Ghost In The Browser Analysis of Web-based Malware”, Proc.Of the first USENIX workshop on hot topic in Botnets, 2007.4

[8] B. Kim, C. Im, H. Jung, “Suspicious Malicious Web Site Detection with Strength Analysis of a JavaScript Obfuscation”, International Journal of Advanced Science and Technology, Vol.26, pp.19-32, Jan, 2011.

[9] Peter Likarish, E. Jung, I. Jo, “Obfuscated Malicious JavaScript Detection using Classification Techniques”, in 4th International Conference on Malicious and Unwanted Software, pp.47-54, 2009.

[10] H. Chang, M. Kim, D. Kim, J. Lee, H. Kim, and S. Cho, “An Implementation of System for Detecting and Filtering Malicious URLs,”

Journal of KIISE:Computing Practices and Letters, Vol.16, No.4, pp.405-414, Apr. 2010. (in Korean)

[11] Y. Choi, T. Kim, and S. Choi, "Automatic Detection for JavaScript Obfuscation Attacks in Web Pages through String Pattern Analysis", International Journal of Security and Its Applications, Vol.4, No.2, pp.13-26, Apr. 2010.

[12] B. Feinstein and D. Peck, "Caffeine Monkey: Automated Collection, Detection and Analysis of Malicious JavaScript", Black Hat USA, 2007.

[13] Christian Seifert, Ian Welch, Peter Komisarczuk, "Identification of Malicious Web Pages with Static Heuristics," Telecommunication Networks and Applications Conference, pp.91-96, Dec. 2008.

[14] J. Lee, J. Moon, S. Cho, Y. Lee, M. Park, and W. Choi, "Malicious Web Page Detection Using Malicious Code Spreading Pattern," The 3rd International Conference on Internet (ICONI 2011), pp.195-200, Dec. 2011.

[15] Chih-Chung Chang and Chih-Jen Lin, LIBSVM: a library for support vector machines, 2001. Software available at *http://www.csie.ntu.edu.tw/~cjlin/libsvm*

저 자 소 개

황영섭

1989 : 서울대학교 컴퓨터공학과 공학사
1991 : POSTECH 컴퓨터공학과 공학석사
1997 : POSTECH 컴퓨터공학과 공학박사
1997-2002 : 한국전자통신연구원 선임연구원
2002-현재 : 선문대학교 컴퓨터공학과 부교수
관심분야 : 패턴인식, 영상처리, 바이오인포매틱스, 문서인식, 신경회로망
Email : young@sunmoon.ac.kr

문재찬

2011 : 단국대학교 컴퓨터과학과 이학사
2011~현재 : 단국대학교 컴퓨터과학 석사과정
관심분야 : 컴퓨터보안, 인터넷보안
E-mail : answocks@dankook.ac.kr

조성제

1989 : 서울대학교 컴퓨터공학과 공학사
1991 : 서울대학교 컴퓨터공학과 공학석사
1996 : 서울대학교 컴퓨터공학과 공학박사
1997~현재 : 단국대학교 소프트웨어학과 교수
2001 : 미국 University of California, Irvine 객원 연구원
2009 : 미국 University of Cincinnati 객원연구원
관심분야 : 컴퓨터보안, 시스템소프트웨어, 실시간스케줄링, 임베디드 소프트웨어 등
Email : sjcho@dku.edu