ICIC Express Letters ICIC International ©2017 ISSN 1881-803X
Volume **11**, Number **11**, November **2017** pp. 1611–1618

A MALWARE DETECTION METHOD BASED ON OC-SVM FOCUSING ON FEATURES OF PDF FILES

Mai Iwamoto[1], Shunsuke Oshima[2] and Takuo Nakashima[3]

[1]Center for Technical and Educational Support
[2]ICT Center for Learning Support
National Institute of Technology, Kumamoto College
2627, Hirayama-shinmachi, Yatsushiro-shi, Kumamoto 866-8501, Japan
{ m-iwamoto; oshima }@kumamoto-nct.ac.jp

[3]Information and Communication Technology Education Center
Tokai University
4-1-1 Kitakaname, Hiratsuka-shi, Kanagawa 259-1292, Japan
taku@ktmail.tokai-u.jp

Received May 2017; accepted August 2017

Abstract. *Risks of PDF files are not generally recognized, and PDF files are treated as safe static documents. PDF files, however, are originally defined as executable file format using different executable codes. In previous researches, various machine learning systems learning features of different types of PDF files are proposed. In this research, we propose a malware detection method based on one-class SVM learning by benign PDF files' features. After classification of all PDF files, files with outliers are detected as the malicious PDF file in this method. In addition, we verify the usefulness of this method based on the comparison experiments between previous methods and our method varying parameter values and selection policies of parameters on one-class SVM.*

Keywords: Malware, PDF files, One-class SVM

1. **Introduction.** PDF (Portable Document Format) [1] is popularly used as the file format for information exchange means with the reproductive file under different environments. PDF files are generally static documents, and are recognized as less risky documents compared to the executable files (exe files), files with macro-command (MS Office files). PDF files, but are actually constructed executable file format with executable codes, such as JavaScript. However, general users do not pay attention to opening the PDF file attached on the e-mail, unlike to open the attached exe files. In addition, the browser can open PDF files without user permission when the PDF files are set on Web contents. PDF files are effective files for attacker in the sense that PDF files are easily opened for users.

In previous researches, machine-learning methods are used to detect malwares. Machine-learning based methods are classified two types, supervised and unsupervised learning. In the malware detection system, the new supervised learning data are difficult to get due to the frequent appearances of new malwares. The supervised machine-learning methods are difficult to adapt the malware detection method for the limit of classification to learn the difference between benign and malicious PDF files.

We utilize one-class support vector machine (OC-SVM) [2], and propose the unsupervised malware detection method. In this method, the detection system learns based on benign PDF files and distinguishes malicious PDF files by detecting outliers. Generally machine-learning needs a lot of learning sample dataset. However, preparing malicious files is difficult for the following reasons. It is difficult to collect a number of malwares. Malicious files' lifetime is short. Huge malwares are generated in short period.

In our method, OC-SVM learns only benign files. It is easy to collect only benign files on the Internet. Easiness of collecting dataset is the advantage of our method.

This paper is constructed as follows. Section 2 introduces the specification of PDF format. Section 3 introduces how OC-SVM works. Section 4 introduces other researches about malicious PDF detection. Section 5 presents our proposed method to detect malicious PDF based on OC-SVM. Section 6 explains about experiments and evaluates our propose method. Section 7 presents the conclusions of this paper.

2. PDF File Format.

The specification of PDF format is opened as ISO 32000-1:2008. We will show the structure of PDF as follows.

Figure 1 shows an example of PDF file. All information of PDF file is written as the combination of multiple objects. Each object has the eight basic types: Boolean, Number, String, Name, Array, Dictionary, Stream and Null. Name object starts / character following arbitrary characters, and is uniquely defined in the file. Array object is surrounded by [ ] characters and constructs arbitrary objects lining up sequentially. Dictionary object is surrounded by << >> characters and consists of pair objects such as name object and arbitrary objects. Stream object consists of dictionary and the byte sequence surrounded by the **stream** string and the **endstream** string. Byte sequences are compressed in arbitrary format, and decoded referring the compression method written in the dictionary when the stream objects are read.

```
%PDF-1.7
1 0 obj
<< /Type /Catalog /Pages 2 0 R
   /OpenAction 5 0 R >>
endobj
2 0 obj
<< /Type /Pages /Kids [3 0 R]
   /Count 1 >>
endobj
3 0 obj
<< /Type /Page /Parent 2 0 R
   /MediaBox [ 0 0 595 842 ]
   /Contents 4 0 R >>
endobj
4 0 obj
<< /Length 74 >>
stream
...(abridged)...
endstream
endobj
5 0 obj
<< /Type /Action /S /JavaScript
/JS 6 0 R >>
endobj
6 0 obj
<< /Length 25 >>
stream
app.alert("Hello World!");
endstream
endobj
...(abridged)...
trailer
<< /Size 6 /Root 1 0 R >>
startxref
597
%%EOF
```

FIGURE 1. An example of PDF file

An arbitrary object assigns the unique object identification to refer from other objects. Indirect objects assign the identifier, and are described between **X**, **Y**, **obj** and **endobj** where **X** is the object number and **Y** is the generation number. PDF files consist of each document page referring to indirect objects. PDF viewer firstly refers the dictionary called trailer described at the tail part. Each object is sequentially referred from the trailer part, and finally a PDF file is totally constituted.

PDF files can restrict the reading, printing and modification using cipher message, but is not totally encrypted but partially encrypted restricting the byte stream of actual pages. This partial encryption leads to extracting file structure from the encrypted PDF files even if the password is not decoded. This structure means that a proposed method is effective not only for unencrypted PDF files but also for encrypted PDF files.

3. One-Class Support Vector Machine.

OC-SVM aims to maximize the distance from the origin pole over the super plane to split the origin pole and the supervised data. When the supervised data $x_1 \cdots x_l \in X$ are given, the function $f(x) = \mathrm{sgn}(w \cdot \Phi(x) - \rho)$ to distinguish in which side the vector exists. $\mathrm{sgn}(x)$ is added $+1$ when x is plus value and is diminished -1 when x is minus value. To minimize the distance from the origin pole on the super plane, $|w|$ should be needed to decease. The supervised data inserting to the origin pole side on the super plane should be minimized at the same time. To resolve two problems in the relation of trade-off at the same time, Equation (1) of the minimizing problem should be resolved and the optimized w and ρ are reduced.

$$\min_{w\in F,\xi\in\mathbb{R}^l,\rho\in\mathbb{R}} \frac{1}{2}|w|^2 + \frac{1}{vl}\sum \xi_i - \rho \tag{1}$$

subject to

$$(w \cdot \Phi(x_i)) \geq \rho - \xi_i,\ \xi_i \geq 0\ (i = 1, \ldots, l) \tag{2}$$

ξ_i is the parameter directing the error distance of supervised data ξ_i, which inserts in the origin pole of the super plane, and is weighed by $1/(vl)$. The parameter $v \in (0, 1)$ is the set value weighted for two problems, minimization of $|w|$ and minimization of $\sum \xi_i$. Smaller value of v makes bigger $\sum \xi_i$ of the error distance. More supervised data could optimize the setting to $f(x_i) = +1$.

In the solving process of the minimizing problem, the calculation of $\Phi(x_i)\cdot\Phi(x_j)$ should be needed. The mapping to $\Phi(x)$ from all supervised data, however, causes the large time complexity. The kernel trick method is used in SVM due to only need for $\Phi(x_i) \cdot \Phi(x_j)$. The kernel function $k(x_i, x_j)$ is exactly used to make $k(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$ instead of calculation of the mapping function $\Phi(x)$. The kernel function has the variety, such as linear kernel, polynomial kernel, and sigmoid kernel. RBF (Radial Basis Function) kernel $k(x_i, x_j) = e^{-|x_i - x_j|^2/c}$ is generally used. In the RBF kernel, the bigger c makes the cover area larger on one support vector.

4. **Related Researches.**

[3] explains widely for the threat of malwares. For example, the MS Office malware, vulnerability of PDF documents, attacker's methods, the trend of malwares, devices hardly to detect malwares are included. [4] shows concrete examples to camouflage from malicious files to benign files, to stop the system by the malicious PDF file to steal and abuse the information. [5] shows the introduction and classification of dangerous functions used by attackers and shows the evaluation of dangers about fishing and second-step attacks.

The actual malware analyzing methods are classified to static methods and dynamic methods. Static methods do not execute codes with analyzing features of codes and extracting codes. Dynamic methods are to extract data executing codes, and analyze the activation of extracted data. Our proposed method is classified to the static method. The static method is classified to methods analyzing JavaScript and to methods distinguishing benign PDF files from malicious PDF files from the feature of meta-data. [6] supervises based on OC-SVM of token vectors extracted from JavaScript in PDF files, and establishes the distinction model of differences between benign PDF files and malicious PDF files. [7] proposed the PDF Malware Slayer [8] (PDFMS) tool. PDFMS is the detection tool based on the machine learning. This method firstly executes the clustering to classify between benign dataset and malicious dataset of keywords appearing in PDF files, and distinguishes the benign PDF files from malicious PDF files based on the machine learning. The learning method uses the random forest classifier. PDFMS is one of static methods similar to our method. The others of PDFMS provide the tool used in our research as the comparative experiments between PDFMS method and our proposed method. [9] distinguishes the benign PDF files from malicious PDF files based on the hierarchal structure of PDF files. SVM and decision tree are used in the learning of the hierarchal structure. [10] learns and classifies 202 features, such as the number of appearances of name objects, for example, **/Font**, **/JavaScript**, and **/JS**, of PDF files using the random forest method.

Our previous research [11] verified whether the specification of PDF files enabling to abuse in the research [4,5] is actually used in malicious PDF files or not. In addition, we verified the features of real malicious files. In this research, we propose to learn the feature vector of benign PDF files using OC-SVM, and propose the detection method of malicious PDF files with outliers on OC-SVM.

5. Proposed Method.

5. Proposed Method. Our method observes eight orders feature vector of benign PDF files based on OC-SVM. Original files are classified, and if the value of OC-SVM is in outlier, then these files should be the malware. We show the eight elements in detail used in the feature vector in our experiments as follows.

(1) **"/JavaScript":** JavaScript Execute. Most of malicious PDF files use JavaScript.
(2) **"Illegal /Length 0":** When **/Length 0** is written in dictionary object on stream objects, there is a possibility of artifices by attackers even if the real byte data exists.
(3) **"Stream Decode Error":** Data in stream objects are compressed by the FlateDecode, and the compression format is designated in /Filter. The compressed data could not be decoded by the designated compression format, and then there is a possibility of artifices by attackers.
(4) **"application/x-javascript":** After the version of PDF 1.5, the dialogue type forms are supported based on Adobe XFA (XML Forma Architecture).
The **application/x-javascript** is designated in the resource to execute JavaScript on XFA based format.
(5) **"No %%EOF":** PDF files are defined of **%%EOF** in the specification. PDF files are generally produced by software to transform to the PDF file, and are not directly modified by the user. Most of benign PDF files obey this specification. Some of malicious PDF files, however, do not include **%%EOF** description.
(6) **"The data after %%EOF":** According to the specification of the PDF files, there is no data after the "end of file" descriptor **%%EOF**. However, some malicious PDF files include embedded data such as JavaScript codes after **%%EOF**.
(7) **"Syntax error no included in the benign PDF files":** Malicious PDF files include many syntax errors to be changed by attackers. In this experiment, we checked using the pdfinfo [12] whether PDF files include syntax errors or not. The pdfinfo, however, generates errors by the limit of functions which pdfinfo cannot recognize the compressed format. Therefore, the syntax errors appearing in benign PDF files are excluded over the syntax errors of analyzing PDF files.
(8) **"File size":** The file size of malwares tends to be small size compared to the benign PDF files. In this research, the value of normalized file size S_N is defined as $S_N = \log_{10}\left(\min\left(10^8, S\right)\right)/8$ which is calculated by the PDF file size S Bytes, and is set to be the feature vector. S_N is the normalized value between 0 and 1 of log scale with the upper limit of 100M Bytes.

In this experiment, we use totally eight features vector included four features vector (1)-(4), which we found the differences between benign PDF files and malicious PDF files in previous research [11], and included four new feature vectors (5)-(8). Each feature vector (1)-(7) has the logical two values, 1 means existence of feature, 0 means no existence of feature, and the feature vector (8) is the normalized value from 0 to 1. The analysis of PDF files used the pdf-parser.py [13].

6. Experiments.

6.1. Evaluation method. In these experiments, we used 1035 PDF files collecting on the proxy server sited on National Institute of Technology, Kumamoto College Yatsushiro Campus as the benign PDF files, and used 221 malicious files registered on VirusTotal [14] from 349 PDF files on D3M dataset collecting from 2010 to 2015. These files of experiments are all unique files and no overlapped files. If the same malicious file is observed on multiple years, appearance year is set to the oldest year.

We used the evaluation method called K-fold cross-validation, which is generally applied in valuation of SVM. We firstly conducted four subset data from 1035 benign PDF files and 221 malicious PDF files. Secondly, we applied three subset data as the learning data and other subset data as the evaluation data. In the proposed method, malicious dataset for the learning dataset is not used due to learning the benign PDF files only. Previous methods [7] compared and evaluated to our method use the malicious dataset as the learning data.

In this research, our method learns the feature vector of benign PDF files as $+1$ class. If the feature vector is designated as $+1$, then the file is the benign PDF file. On the other hand, the feature vector is designated as -1, and the file is the malicious PDF file. Accuracy of four dataset is calculated, and the average accuracy is reduced. This accuracy is the rate of classified data existing in the right class over the evaluation data. The accuracy should be evaluated for malware detection in the three viewpoints to detect benign PDF files as the benign PDF file, to detect malicious PDF files as the malicious PDF file, to detect rightly evaluating total evaluation composing two features. We use three metrics, benign accuracy: the rate of detecting benign PDF files as the begin PDF file, malicious accuracy: the rate of detecting malicious PDF files as the malicious PDF file, and benign and malicious accuracy: the rate of detecting begin and malicious PDF files as the right class.

6.2. **Selection of parameters.** In this experiment, we selected OC-SVM implemented in LIB-SVM [15] to classify based on eight order feature vector. The RBF kernel is used as the kernel function used in generally other researches. We should set two parameters, c and v described in Section 3 for using RBF kernel and OC-SVM. The optimized parameter values are extracted using experiments varying c and v values.

Table 1 shows benign accuracy varying c and v, Table 2 and Table 3 show malicious accuracy and benign and malicious accuracy respectively. Colored parts in each table indicate when the average accuracy of four times is larger than 0.954 ($\geq 2\sigma$). In Table 1, benign accuracy covers large area with more than 0.954 over $2^{-2} \leq c \leq 2^6$, $0.001 \leq v \leq 0.02$. In Table 2, malicious accuracy covers all area with more than 0.90 for all parameters.

In the condition of the largest value of Benign Accuracy with $c = 2^2$, $v = 0.001$ in Table 1 four benign PDF files were not rightly detected over 1035 benign PDF files. In four no accurate detections, one case is that **/JavaScript** is included, one case is that **%%EOF** is not described and two cases are that file sizes are small such as 4090 and 4129 Bytes. Every case has out of features of benign learning PDF files leading every case

Table 1. Benign accuracy

c/v	0.001	0.002	0.005	0.01	0.02	0.05	0.1
2^{-10}	0.619	0.611	0.645	0.734	0.691	0.700	0.644
2^{-8}	0.636	0.753	0.720	0.710	0.782	0.760	0.743
2^{-6}	0.664	0.767	0.763	0.888	0.949	0.894	0.900
2^{-4}	0.846	0.904	0.920	0.968	0.966	0.949	0.889
2^{-2}	0.992	0.991	0.991	0.990	0.980	0.952	0.901
2^{0}	0.992	0.992	0.991	0.990	0.980	0.952	0.901
2^{2}	0.996	0.995	0.993	0.990	0.980	0.951	0.901
2^{4}	0.996	0.995	0.993	0.990	0.980	0.952	0.901
2^{6}	0.992	0.994	0.993	0.987	0.979	0.950	0.901
2^{8}	0.831	0.833	0.932	0.987	0.979	0.951	0.901
2^{10}	0.000	0.872	0.701	0.642	0.822	0.953	0.901

Table 2. Malicious accuracy

c/v	0.001	0.002	0.005	0.01	0.02	0.05	0.1
2^{-10}	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2^{-8}	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2^{-6}	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2^{-4}	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2^{-2}	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2^{0}	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2^{2}	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2^{4}	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2^{6}	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2^{8}	0.901	1.000	1.000	1.000	1.000	1.000	1.000
2^{10}	1.000	1.000	1.000	1.000	1.000	1.000	1.000

Table 3. Benign and malicious accuracy

c/v	0.001	0.002	0.005	0.01	0.02	0.05	0.1
2^{-10}	0.686	0.679	0.708	0.781	0.745	0.752	0.706
2^{-8}	0.700	0.796	0.769	0.761	0.820	0.803	0.788
2^{-6}	0.723	0.808	0.805	0.908	0.958	0.912	0.918
2^{-4}	0.873	0.921	0.934	0.974	0.972	0.958	0.908
2^{-2}	0.994	0.993	0.993	0.992	0.983	0.960	0.919
2^{0}	0.994	0.994	0.993	0.992	0.983	0.960	0.919
2^{2}	0.997	0.996	0.994	0.992	0.983	0.959	0.919
2^{4}	0.997	0.996	0.994	0.992	0.983	0.960	0.919
2^{6}	0.994	0.995	0.994	0.990	0.982	0.959	0.919
2^{8}	0.843	0.862	0.944	0.990	0.982	0.959	0.919
2^{10}	0.176	0.894	0.753	0.705	0.854	0.961	0.919

to the outlier cases. The case $c = 2^8$, $v = 0.001$ in Table 2 could not detect all malwares. 22 files of 221 malicious PDF files are detected as benign PDF files. All wrong detected files included **/JavaScript** and with smaller then 40KB file size. All case have no six features except **/JavaScript** over the seven feature vector with Boolean values. For the detection as the benign PDF file with the same features, this feature is recognized and optimized as +1 class in learning process. These wrong results show the attention that the performance of SVM varies on the value of parameters. Our method, however, covers wide effective parameter area and adapts the variation of different dataset.

6.3. **Comparison to previous method.** Table 4 shows the result of comparison using the previous method, PDFMS, and proposed method for benign accuracy, malicious accuracy, benign and malicious accuracy. The OC-SVM parameter values of c and v are selected as $c = 2^2$, $v = 0.001$ at the case of the highest accuracy value in Section 6.2. PDFMS [7,8] is compared to the previous method.

Table 4. Accuracy of previous method and the proposed method

Method	Benign	Malicious	Benign and Malicious
PDFMS	0.913	0.946	0.919
Proposed Method	0.996	1.000	0.997

Table 4 shows the results of experiments. In previous method, benign accuracy is 0.913, malicious accuracy is 0.946, and benign and malicious accuracy is 0.913. On the other hand, benign accuracy is 0.996, malicious accuracy is 1.000, and benign and malicious accuracy is 0.997 in the proposed method. These results show the proposed method achieved high accurate rate. In the previous method, unclassified files with analyzing error contain 89 benign PDF files and 12 malicious PDF files.

7. **Conclusions.** We proposed the method to detect the feature of malware classified to -1 class (outlier) learning the feature vector of benign PDF files as $+1$ class using OC-SVM. The learning process uses the Boolean value of $(0, 1)$ indicating the appearances of seven features included in actual malicious PDF, and uses eight order vector, which element consists of the normalized values from 0 to 1 of PDF file size.

As the results of experiments, the value of malicious accuracy becomes 1.000 over the large parameter value area with $2^{-10} \leq c \leq 2^{10}$, $0.001 \leq v \leq 0.1$. On the other hand, the $2^{-2} \leq c \leq 2^{6}$, $0.001 \leq v \leq 0.05$. The performance of SVM varies depending on the value of parameters. Our method can respond the change of dataset due to the high accuracy value over the wide area.

Our method, which learns only benign PDF files, has better accuracy rate than the previous method, which conducts the machine learning both benign and malicious PDF files. These results lead that our method, firstly producing feature vector of malicious PDF files, secondly learning features of benign PDF file, finally identifying as the malware with outlier, is effective method based on OC-SVM.

The learning method of data from benign PDF files using OC-SVM has the merit of no need of learning malicious data. On the other hand, features as the element of vector are difficult to be selected. The number of dataset in this experiment is small number. In the future, we will conduct experiments using other dataset, and find the new features to improve the accuracy of detection.

REFERENCES

[1] Adobe Systems Incorporated, *Document Management – Portable Document Format – Part 1: PDF 1.7*, http://www.adobe.com/content/dam/Adobe/en/devnet/acrobat/pdfs/PDF32000_2008.pdf.

[2] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor and J. C. Platt, Support vector method for novelty detection, in *Advances in Neural Information Processing Systems 12*, S. A. Solla, T. K. Leen and K. Müller (eds.), MIT Press, 2000.

[3] K. Selvaraj and N. Fred Gutierrez, *The Rise of PDF Malware*, https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/the_rise_of_pdf_malware.pdf.

[4] F. Raynal, G. Delugr and D. Aumaitre, Malicious origami in pdf, *Journal in Computer Virology*, vol.6, no.4, pp.289-315, 2010.

[5] A. Blonce, E. Filiol and L. Frayssignes, Portable document format (PDF) security analysis and malware threats, *Presentations of Europe BlackHat 2008 Conference*, 2008.

[6] P. Laskov and N. Srndic, Static detection of malicious javascript-bearing PDF documents, *Proc. of the 27th Annual Computer Security Applications Conference (ACSAC'11)*, NY, USA, pp.373-382, 2011.

[7] D. Maiorca, G. Giacinto and I. Corona, *A Pattern Recognition System for Malicious PDF Files Detection*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[8] Pattern Recognition and Applications Lab, *Slayer*, https://pralab.diee.unica.it/en/Slayer.

[9] N. Srndic and P. Laskov, Detection of malicious PDF files based on hierarchical document structure, *NDSS*, The Internet Society, 2013.

[10] C. Smutz and A. Stavrou, Malicious PDF detection using metadata and structural features, *Proc. of the 28th Annual Computer Security Applications Conference (ACSAC'12)*, NY, USA, pp.239-248, 2012.

[11] M. Iwamoto, S. Oshima and T. Nakashima, A study of malicious PDF detection technique, *The 10th International Conference on Complex, Intelligent, and Softaware Intensive Systems*, pp.197-203, 2016.

[12] Clyph & Cog, LLC., *Xpdf*, http://www.foolabs.com/xpdf/home.html.

[13] Didier Stevens, *Didier Stevens PDF-parser.py*, http://blog.didierstevens.com/2008/10/30/pdf-parserpy/.
[14] *VirusTotal*, https://www.virustotal.com/ja/.
[15] C.-C. Chang and C.-J. Lin, *Libsvm – A Library for Support Vector Machines*, https://www.csie.ntu.edu.tw/∼cjlin/libsvm/.