IJCTA, 9(22), 2016, pp. 241-247

An Enhanced Approach towards Detection of Malicious PDF Files

Gurjeet Singh* Sanjay Madan Rakesh Kumar Sehgal*** Neeraj Sharma******

Abstract : Malicious PDF files are of grave concern as they pose a serious threat to system security in terms of confidentiality, integrity and availability. Social engineering techniques are generally preferred by the attackers to prompt users for accessing malicious PDF files. The available antivirus mostly lags a step behind the attackers and fails to detect such files. In this paper, a Machine-learning model based framework has been proposed through segregating the features for classification to detect malicious PDF files. The features, which are used to create malicious PDF files, are identified and a model is built based on these features to classify the PDF files as Benign or Malicious.

Keywords : Malicious PDF files, Cyber Security, Machine Learning.

1. INTRODUCTION

The cyber-attacks have been increased rapidly from the last decade and the objectives of these attacks are to steal confidential information, monitoring and spying of the information from the targeted systems. Among the large attack surface in the system, the PDF files are one of the source of infection in the system. The PDF files are vulnerable due to their structure and several attacks are observed in which PDF vulnerabilities are exploited. The malicious PDF files are hosted on the web for luring users to open them using social engineering techniques [1]. E-mails are commonly used for exchange of Digital Data therefore, Emails containing malicious files as attachments become the attack vector for the attackers. The prevention measures like Firewalls, antivirus, intrusion detection system (IDS) [2], Intrusion prevention system (IPS) [3] etc. are unable to detect attacks in which browser vulnerability is exploited when user opens the malicious pdf files in the browser [4]. Attachment of executable files are prevented in most of the email servers but non-executable (PDF or MS office) files are as dangerous as executable files and they are being used in cyber attacks. Most of the users consider non-executable files safer than the executable files and they do not think twice before they open these files. Attackers take advantage of this and use non- executable files as attack vector for cyber attacks.

1.1. PDF File Structure

Portable Document File (PDF) [5] file is the collection of objects interconnected to each other. A PDF file can contain simple text, images, multimedia elements, JavaScript etc. There are four basic parts in the PDF file [6]: objects, file structure, content stream and document structure.

- **Objects :** Objects can be direct objects that do not have any reference number and indirect objects, which are referenced by number. Table 1 shows the 8 types of PDF objects.

* Department of Computer Science & Engineering Chandigarh University, Mohali-140413, India
** Cyber Security Technology Division Center for Development of Advanced Computing, Mohali-160071, India
*** Cyber Security Technology Division Center for Development of Advanced Computing, Mohali-160071, India
**** Department of Computer Science & Engineering Chandigarh University, Mohali-140413, India

Table 1. PDF Objects

Object	*Description*
Boolean	true or false values
Numeric	real or integer number
String	an arrangement of literal characters enclosed by ( ) or hexadecimal information enclosed by < >
Name	A sequence of characters starting with /
Array	sequence of objects enclosed by [ ]
Dictionary	A sequence of pairs of a keyword (name object) and a value (Boolean , numeric, an array or another keyword) enclosed by << >>
Stream	a special dictionary objects enclosed by keywords stream and end stream used to store stream data
Null	Empty object

- **File Structure :** The file structure tells how the objects are updated and accessed inside the PDF file. Table 2 shows the PDF file composition with its components.

Table 2. PDF File Structure Components.

Structure	*Description*
Header	gives information about the version of the file
Body	main portion of file which contains all the PDF objects
Cross-reference table	gives information about every indirect object in the file
Trailer	gives information about root object and number of revisions of the file

- **Content Streams :** These are stream objects, which give the information about the physical appearance of the page.

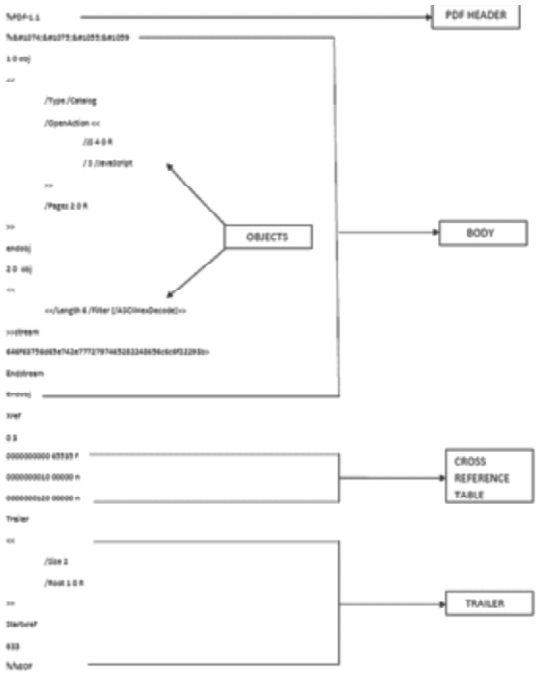

Fig. 1. PDF file structure.

- **Document structure :** It defines the structure of the pdf i.e. how objects are used in the file and hierarchy of the objects. Catalog object is the main object in the hierarchy which is represented by a dictionary and its position can be located in the trailer under the /Root name object. Fig 1 shows the structure of simple PDF file.

1.2. Possible attacks via PDF File

- **JavaScript Code attack :** A PDF file can contain JavaScript code for 3D content, calculations and form validation. The presence of /JS keyword is the indication of JavaScript code in the PDF file. Malicious JavaScript Code can be injected by attacker in the PDF file in order to exploit the vulnerability of PDF reading applications and the browser's pdf executer.
- **Embedded files attack :** A PDF file contain attached files such as SWF, EXE, HTML, XLSX, Microsoft office or even PDF files. These files can be used to hide malicious executables by exploiting their vulnerabilities.
- **Form Submission and URL attack :** There is provision in Adobe reader for submitting PDF form from the client side to the server side [7]. A number of file formats can be used for this purpose. Default file Forms Data Format (FDF) is generated by the Adobe reader, which is used to send data to the specified URL and then reply, comes back to client's web browser. An attack can be made by requesting to a malicious website that will be opened on the client's web browser automatically and which can easily exploit the vulnerabilities of the web browser.

The remaining sections of the paper are organized as follows. In section II, related work for the detection of malicious files is discussed. Section III describes the methodology of proposed work and Section IV presents the experimental results. Section V gives concluding remarks.

2. RELATED WORK

Laskov et al. [8] introduced a pure static analysis tool (PJscan) for detection of Malicious PDF files bearing malicious java scripts. One class support vector machine (OCSVM), a machine learning method was used for classification of the data. The dataset contained 65942 real PDF files captured from VirusTotal in which 25691 files were malicious and 40251 files were benign. True positive rate (TPR) attained for known PDF attacks was 84.80% and for unknown PDF attacks was 71.15% and False positive rate attained was 16.35%.

Vatamanu et al. [9] introduced a static approach based on tokenization of embedded JavaScript in PDF files to detect malicious PDF files. Two clustering techniques were used: hierarchical bottom up and hashtable. The article concentrates on building up a clustering technique for the recognizable proof of comparative scripts that have been obfuscated by different methods.

Mairoca et al. [10] introduced a static analysis tool the PDF malware slayer (PDFMS), for detection of malicious PDF files based on occurrence of embedded keywords. The dataset consisted 11,157 malicious and 9989 benign pdf files out of which 6000 malicious plus 6000 benign files were in the training set and rest formed the test set. Two models were proposed: a data retrieval module and feature extractor module. The proposed tool can detect malicious pdf files irrespective of the presence of JavaScript code. The limitation of the PDFMS was that the attacker can learn keywords which characterize the PDF file and thus bypass the tool.

Smutz et al [11] introduced PDFRate, a malicious PDF detection method based on meta-features extracted from the PDF files. Authors implemented their own program for feature extraction. For classification 202 features were chosen which includes features like number of /JavaScripts, /font, /JS, sum of all pixels in all images in document so on. Data was collected from two sources: Contagio data set and HTTP and SMTP traffic of a large university campus. The data set consisted of 100000 benign and 10000 malicious PDF files. For classification Random Forest classifier was used.

Laskov et al. [12] introduced a static method based on structural properties of PDF files to characterize the PDF files as Benign or Malicious. In this method, PDF file was characterized by converting it into the structural

paths. POPPLER was used to extract structural paths from training the set which consist both benign and malicious PDF files. In classification phase two classifiers were used: Support Vector Machines (SVM) and Decision Tree. TPR attained was 99.98% whereas FPR was .01%. A comparison is made between proposed method and other detectors which includes PJScan, ShellOS and Malware Slayer.

Pareek et al. [13] introduced two discrete static methods Entropy and n-gram for the detection of malicious PDF files. The dataset consisted of 1584 files from which 792 were benign and 792 were malicious. Benign files were collected reports, financial documents and research papers where as malicious files were collected from malware.lu, contagiodumo.com, computing.net and Brandon Dixon. To build a model J48 algorithm was used. The results obtained for FPR and FNR was 0.01 and 0.0044 respectively.

Mairoca et al. [14] introduced an evasion technique named reverse mimicry, which can easily bypass the malicious PDF detectors based on logical structure of the PDF. The reverse mimicry attack was implemented by 3 different methods: EXE Embedding, PDF Embedding, JavaScript Injection. A new framework was also proposed by the authors to deal with the evasion attacks based on analysis of embedded exe, PDF structural analysis and embedded JavaScript analysis.

3. SYSTEM DESIGN AND IMPLEMENTATION

3.1. Approach

In the proposed work, the data is collected from Contagio blogspot [18] and from malware.lu [19] which provides a repository for benign and malware samples. After the data is gathered, it is preprocessed and relevant features are extracted using modified PDFiD.py script [16] by adding new features namely size, Names, Font and entropy. The final feature set consists of 30 features. After feature extraction, the dataset is divided into training set and testing set both consisting of equal number of malicious and benign files. At last we performed test on our dataset using Naive Bayes, Random Forest, ADtree, Random Committee and Decision Stump classifiers. Thus different models were trained to classify PDF files as Benign or Malicious. Figure 2 depicts the architecture of our proposed work.

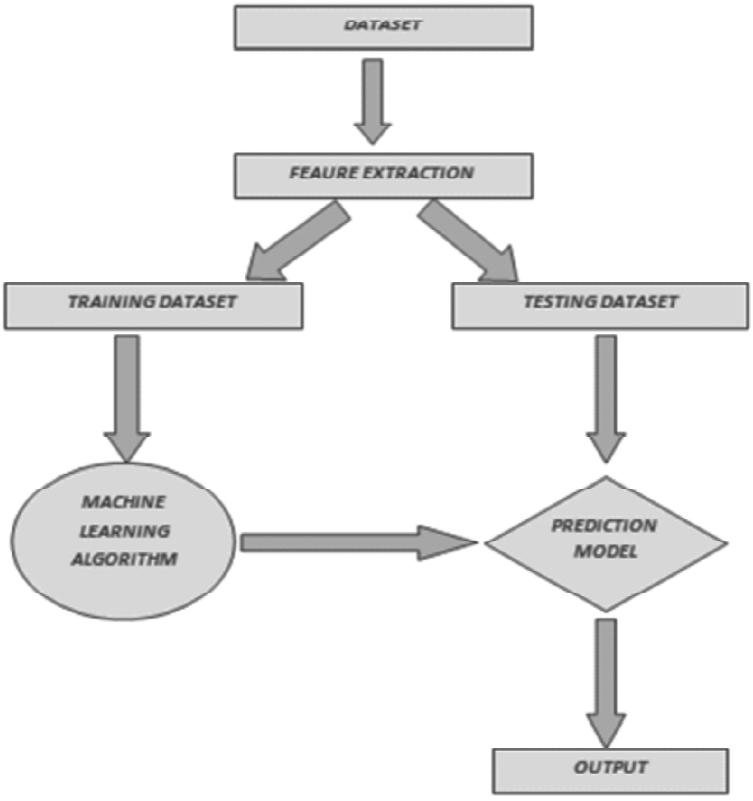

Fig. 2. Proposed approach for the Classification of PDF Files.

3.2. Dataset

For the training phase, both legitimate and malicious files are used. The data has been collected from the Contagio blogspot and from malware.lu. The data was pre-processed and relevant features were extracted. After feature extraction, we divided dataset into two parts, training set and testing set. The division was made on the criteria that the count of malicious and benign files in the training set must be equal for achieving an unbiased training set. Our training set consisted of 5127 files in which 2600 files were benign and 2527 were malicious. Testing set consisted of 1000 PDF files in which 500 were benign and 500 were malicious.

3.3. Feature Extraction

Feature extraction is the most significant part of the proposed work, as a wrong choice of the features will lead to inaccurate results. Compressed data streams usually consist of malicious code. There is large variety of PDF objects which makes it difficult to analyse data stream as a whole. Moreover only a fraction of PDF attacks can be detected if focus is on any particular object like ActionScript or JavaScript. In order to resolve this problem, PDF files are characterized as per set of embedded keywords. For a PDF reader to execute actions it is required to recognise certain specific PDF keywords. These keywords occurrences are helpful to understand the complex behaviours of PDF readers when a PDF file is accessed and difference between legitimate and malicious PDFs. In the proposed work, we began with PDFiD.py script to extract various features from the PDF files and then added additional features (size, /Names, /Font and entropy) in the script. The added features were based on the previous research work. For example, size is a crucial feature; according to [8] most of the malicious PDF files are small in size as compare to benign PDF files. Similarly according to [15] [11] there are less number of fonts used in most of malicious files as compared to benign files. According to [13] most malicious PDF files have less entropy as compared to the Benign PDF files. The importance of each feature is given in PDF specification [5]. Table 3 shows some of the important features extracted from the PDF files. The final feature set consists of 30 features.

Table 3. Feature set

Features	***Description***
JavaScript	How many blocks of JavaScript are in the PDF?
OpenAction	Does the PDF perform an action upon opening?
JS	Number of single line JavaScript statements in the PDF.
RichMedia	How many blocks of Flash content are embedded?
JBIG2Decode	How many times is the JBIG2Decode filter used?
Names	How many catalog names listings are there?
Size	Size of the PDF file
Font	How many font objects are used in the PDF file?
AA	Does the PDF respond to user actions?
Xref	Does the xref table exist?
Pages	Number of pages in the PDF file
Launch	How many times Launch object is used in PDF to launch a program / script?

3.4. Classification

For classification of dataset Weka [17] software was used. Weka is a workbench consisting of algorithms for predictive modeling and data analysis with visualization tools and GUI for access to these functions. Weka has many built in machine learning algorithms which are used for building prediction models. In the proposed work, different classifiers such as Naive Bayes, Random Forest, ADtree, Random Committee and Decision Stump have

been considered. These classifiers are analyzed for their accuracy and reliability on the training model. The classifier with the best results is selected and the test dataset is used to validate the model. The accuracy of our system is compared with other commercial available systems.

4. EXPERIMENT AND RESULT

We have executed our tests on Naivebayes, ADtree, DecesionStump, RandoForest and RandomCommittee, using a 10-folds Cross Validation test. Table 4 shows the performance of different classifiers in terms of TPR and FPR.

Table 4. A comparison of different classifiers in terms of TPR and FPR

Classifiers	*TPR*	*FPR*
NaiveBayes	83.1	16.6
ADtree	99.2	0.8
Decisionstump	89.9	9.9
Randomforest	99.6	0.4
Randomcommittee	99.8	0.2

As compared other classifiers, Random committee gave best results with highest TPR and lowest FPR. After the data is trained using 10-folds Cross Validation, accuracy of the classifiers is evaluated on test set. The model built using the proposed work in this paper is compared with 2 commercial available tools PDFRate and PeePDF. These tools were used to analyse the test set. The results of comparison in terms of detection rate are shown in table 5.

Table 5. A comparison in terms accuracy between our system and other commercial tools

Similar Applications	*Accuracy*
Proposed system	95%
PDFRate	85%
PeePDF	62%

5. CONCLUSION

In this paper, the framework for the detection of malicious PDF file is proposed which is based on feature extraction and classification model built upon these features leads to the detection of malicious pdf files. PDF files are generally propagated through emails or the Drive-by-download attacks through malicious link by the attackers to steal the user's data or to gain unauthorized access. Therefore, detection of malicious PDF files is necessary to avoid these types of attacks. The proposed system is based on the detection of malicious PDF file using internal configuration of the file format which is effective against numerous attack types. Based on feature set, Random Committee performed better than Naive Bayes, Decision Stump, Random Forest and ADtree classifiers. Wisely chosen feature set plays a major role in malicious PDF detection as dropping some features like /JS and /OpenAction lead to varying detection results. Our system accuracy depicts that it outperforms two commercial available tools, PDFRate and PeePDF and is effective against detecting novel attacks. The structure of PDF files requires deep investigation of its syntax as it indicates the existence of malicious content, in order to correlate specific PDF features with known attack semantics. Thus it is essential to consider PDF semantics for understanding the PDF content modifications.

6. ACKNOWLEDGEMENT

This work has been carried out at Cyber Security Technology Division, C-DAC, Mohali and we are thankful to the organization and director, Sh. D.K. Jain for his support and team members for their time-to-time guidance.

7. REFERENCES

1. About social engineering, http://www.symantec.com/connect/articles/social-engineering-fundamentals-part-i-hacker-tactics/ Date accessed: 02/02/2016.
2. About Intrusion detection system, http://netsecurity.about.com/cs/hackertools/a/aa030504.htm/ Date accessed: 15/01/2016.
3. About Intrusion Prevention system, https://www.paloaltonetworks.com/documentation/glossary/what-is-an-intrusion-prevention-system-ips/ Date accessed: 15/01/2016.
4. H Kaur, S Madan, RK Sehgal, "UAC: A Lightweight and Scalable Approach to Detect Malicious Web Pages", Modern Trends and Techniques in Computer Science: 3rd Computer Science On-line Conference 2014 (CSOC 2014), Springer International Publishing, pp. 241-261.
5. About Pdfs, http://www.adobe.com/devnet/pdf/pdf_reference.html/ Date accessed: 25/01/2016.
6. About Pdf references 6th editions, http://www.adobe.com/content/dam/Adobe/en/devnet/acrobat/pdfs/pdf_reference_1-7.pdf/ Date accessed: 30/01/2016.
7. Hamon, Valentin. "Malicious URI resolving in PDF documents." Journal of Computer Virology and Hacking Techniques 9, no. 2. 2013; 65-76.
8. Laskov, Pavel, and Nedim Šrndiæ. "Static detection of malicious JavaScript-bearing PDF documents." In Proceedings of the 27th Annual Computer Security Applications Conference ACM. 2011; 373-382.
9. Vatamanu, Cristina, Dragoº Gavriluþ, and Rãzvan Benchea. "A practical approach on clustering malicious PDF documents." Journal in Computer Virology 8. 2012; 151-163.
10. Maiorca, Davide, Giorgio Giacinto, and Igino Corona. "A pattern recognition system for malicious pdf files detection." In Machine Learning and Data Mining in Pattern Recognition, Springer Berlin Heidelberg. 2012; 510-524
11. Smutz, Charles, and Angelos Stavrou. "Malicious PDF detection using metadata and structural features." In Proceedings of the 28th Annual Computer Security Applications Conference ACM. 2012; 239-248.
12. Šrndic, Ned, and Pavel Laskov. "Detection of malicious pdf files based on hierarchical document structure." In Proceedings of the 20th Annual Network & Distributed System Security Symposium.2013;
13. Pareek, Himanshu, P. Eswari, N. Sarat Chandra Babu, and C. Bangalore. "Entropy and n-gram Analysis of Malicious PDF Documents." In International Journal of Engineering Research and Technology, vol. 2, ESRSA Publications, 2013.
14. Maiorca, Davide, Igino Corona, and Giorgio Giacinto. "Looking at the bag is not enough to find the bomb: an evasion of structural methods for malicious pdf files detection." In Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security ACM. 2013; 119-130.
15. Maiorca, Davide, Davide Ariu, Igino Corona, and Giorgio Giacinto. "An Evasion Resilient Approach to the Detection of Malicious PDF Files." In Information Systems Security and Privacy Springer International Publishing. 2015; 68-85.
16. About Pdf tools, https://blog.didierstevens.com/programs/pdf-tools/ Date accessed: 22/02/2016.
17. Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. "The WEKA data mining software: an update." ACM SIGKDD explorations newsletter. 2009; 10-18.
18. About malicious PDFs, http://contagiodump.blogspot.in/2013/03/16800-clean-and-11960-malicious-files.html/ Date accessed : 05/03/2016
19. About malware, https://malware.lu/ Date accessed: 12/03/2016.