IJIRMET ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 8, Issue 3, March 2023

MalJPEG: MACHINE LEARNING BASED SOLUTION FOR THE DETECTION OF MALICIOUS JPEG IMAGES

[1] Dr.Duraipandian M,[2] Mohamed Rashik S,[3] Praveen P,[4] Kamlesh K,[5] Karan
[1] HOD/IT, Department of Information Technology
[2][3][4][5] Department of Information Technology.

Abstract: ***In recent years, cyber-attacks against individuals, businesses, and organizations have increased. Cyber criminals are always looking for effective vectors to deliver malware to victims in order to launch anattack. Images are used on a daily basis by millions of people around the world, and most users consider images to be safe for use; however, some types of images can contain a malicious payload and perform harmful actions. JPEG is the most popular image format, primarily due to its lossy compression. It is used by almost everyone, from individuals to large organizations, and can be found on almost every device (on digital cameras and smartphones, websites, social media, etc.). Because of their harmless reputation, massive use, and high potential for misuse, JPEG images are used by cyber criminals as an attack vector. While machine learning methods have been shown to be effective at detecting known and unknown malware in various domains, to the best of our knowledge, machine learning methods have not been used particularly for the detection of malicious JPEG images. In this paper, we present MalJPEG, the first machine learning- based solution tailored specifically at the efficient detection of unknown malicious JPEG images. MalJPEG statically extracts 10 simple yet discriminative features from the JPEG file structure and leverages them with a machine learning classifier, in order to discriminate between benign and malicious JPEG images. We evaluated MalJPEG extensively on a real-world representative collection of 156,818 images which contains 155,013 (98.85%) benign and 1,805 (1.15%) malicious images. The results show that MalJPEG, when used with the LightGBM classifier, demonstrates the highest detection capabilities, with an area under the receiver operating characteristic curve (AUC) of 0.997, true positive rate (TPR) of 0.951, and a very low false positive rate (FPR) of 0.004.***

INDEX TERMS JPEG, image, malware, detection, machine learning, features.

1. INTRODUCTION

Cyber attacks targeting individuals, businesses, and organi- zations have increased in recent years. Infosecurity maga- zine declared that cyber attacks doubled in 2017.1 Cyber attacks usually include harmful activities such as stealing confidential information, spying, or monitoring, and cause harm (sometimes significant) to the victim. Attackers may be motivated by ideology, criminal intent, a desire for publicity, etc.

Attackers are constantly searching for new and effective ways to launch attacks and deliver a malicious payload to victims. Files sent via the Internet have often served as a means of accomplishing this. Since executable files (∗.exe) are known to be dangerous, attackers are increasingly using non-executable files (e.g., ∗.pdf, ∗.docx, etc.) which are mistakenly considered to be safe for use by most users.

Some non-executable files allow an attacker to run arbitrary malicious code on the targeted victim machine when the file is opened. JPEG (Joint Photographic Experts Group) is the most pop- ular image format,2 mainly because of its lossy compression. JPEG images are used by almost everyone, from individuals to large enterprises, and on various platforms. JPEG images can be found on computers (personal images, documents), devices (smartphones, digital cameras, etc.), and in cyber

space (emails, social media, websites, etc.). Due to their harmless reputation, massive use, and high potential for mis- use, cyber criminals use JPEG images as an attack vector in order to deliver their malicious payload to the victim device. At the 2015 Black Hat conference, Saumil Shah demon- strated3 how to create malicious JPEG images that can be loaded in a browser in order to execute malicious code automatically.4,5 In November 2016, it was reported that attackers used Facebook Messenger to spread the infamous Locky ransomware via JPEG images.6 The malware authors discovered security vulnerabilities in Facebook and LinkedIn that allow them to forcibly download a malicious image on the victim's computer. In August 2017, it was reported that SyncCrypt ransomware

was spread using JPEG images.7 In December 2018, Trend Micro,8 an enterprise cyber security company, reported that cyber criminals used memes on Twit- ter(JPEG images) in order to convey commands to malware.9 Recently, in December 2019, researchers from the Sophos security company published a comprehensive report10 on the MyKings cryptomining botnet that lurks behind a seemingly innocuous JPEG of Taylor Swift.

We evaluate MalJPEG extensively on a real-world representative collec-tion of benign and malicious JPEG images. We also compare MalJPEG features to features extracted using several existing generic feature extraction methods.

The paper's contributions are as follows:

1) MalJPEG – a machine learning based solution for effi- cient detection of known and unknown malicious JPEGimages.
2) MalJPEG features – a compact set of 10 simple yet dis- criminative features for the efficient detection of mali- cious JPEG images using machine learning techniques.
3) The creation of a large and representative labeled collec- tion of benign and malicious JPEG images that can be used for further research by the scientific community.

We provide background information related to the JPEG file format in Section II and discuss related work in Section III. Section IV describes the methods used in this research and the MalJPEG features. We evaluate our method and presentthe results in Section V. We discuss the results and various aspects of security and present our conclusions in Section VI.

II. BACKGROUND

In this section, we provide background material related to our research, as well as technical information regarding the structure of a JPEG image. Since the JPEG file structure is complicated, we only present the basic information needed toenable the reader to comprehend the paper and understand the proposed MalJPEG solution presented in this research. The format of JPEG images is comprehensively described in the JPEG File Interchange Format (JFIF) specification.11

A. JPEG FILE STRUCTURE

JPEG stands for Joint Photographic Experts Group, which has become the most popular image format on the Web. In 1992, JPEG became an international standard for compressing dig- ital still images. JPEG files usually have a filename extension of .∗jpg or .j∗peg.

A JPEG image file is a binary file which consists of a sequence o segments. Segments can be contained in other segments hierarchically. Each segment begins with a two- byte indicator called a ‘‘marker.’’ The markers help divide the file into different segments. A marke's first byte is 0xFF (hexadecimal representation; the second byte may have any value except 0x00 and 0xFF. The marker indicates the type of data stored in the segment. Segment types are assigned names based on their definition or purpose; for example, the name of 0xFFD9 is OI, and the name of 0xFFFE is COM. Segment types 0xFF01 and 0xFF0@hyphe0xFF9 consist entirely of the two-byte marker; all other markers are followed by a two- byte integer indicating the size of the segment, followed by the payload data contained in the segment. Table 1 presents the possible markers, their hexadecimal code, and their defi- nition/purpose.

ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 8, Issue 3, March 2023

JPEG CLASSIFICATION

	Code	
APP_n		app
COM	0xFFFE	Comment
DAC	0xFFCC	Define arithmetic conditioning table(s)
DHP	0xFFDE	Define hierarchical progression
DHT	0xFFC4	Define Huffman table(s)
DNL	0xFFDC	Define number of lines
DQT	0xFFDB	Define quantization table(s)
DRI	0xFFDD	Define restart interval
EXP	0xFFDF	Exp
JPG	0xFFC8	JPEG extensions
JPG_n	0xFFF0-0xFFFD	Reserved for JPEG extensions
RES	0xFF02-0xFFBF	Reserved
RST_m	0xFFD0-0xFFD7	Restart with modulo 8 counter m
SOF_n	0xFFC0-3, 5-7, 9-B, D-F	Start of *Frame*
SOS	0xFFDA	Start of *Scan*
TEM	0xFF01	For temporary use in arithmetic coding
SOI	0xFFD8	Start of image
EOI	0xFFD9	End of image

EMBEDDING MALICIOUS PAYLOAD IN JPEG IMAGES

Vulnerability Exploitation – No software is ever completely protected, and it is almost impossible to prevent the presence of vulnerabilities during the development of a large-scale software project. Such vulnerabilities, when exploited, can allow an adversary to obtain higher privileges or divert the normal execution flow to an arbitrary malicious code.

In addition, in order to view/parse a JPEG image, Steganography (steganos – covered, graphie – writing) – Steganography,16 a technique used for disguising informa- tion (e.g., text or malicious code) inside the image without affecting its appearance (invisible to the human eye) is very difficult to detect. Steganography can be used to exfiltrate sensitive information from the victim's host or network via JPEG images and can even be used for delivering pieces of code into the victim's host or network under the guise of a simple benign JPEG image.

embedded payload; thus, we discriminate between JPEG images that carry hidden information using steganography and JPEG images that carry a malicious payload. In this

It is important to emphasize that malicious JPEG images do not necessarily use steganography methods to conceal the aviewer/parser program is required, and these programs may have some vulnerabilities. Many vulnerabilities related to JPEG images have been discovered since it was first published, and there are currently 303 known vulnerabili- ties13 (CVE – Common Vulnerabilities and Exposures), and 5,520 known related security issues14 associated with JPEG(CVE-2018-6612) may allow a remote attacker to cause a denial-of-service when the victim processes a malicious JPEG file

III. METHODS

In this section, we describe the methods used in this research. We begin by presenting MalJPEG's features as well as the existing generic feature extraction methods. We then compare the features extracted by the existing generic feature extrac- tion methods and the features extracted by the MalJPEG feature extractor. Finally, we describe the machine learning algorithms we used in this research.

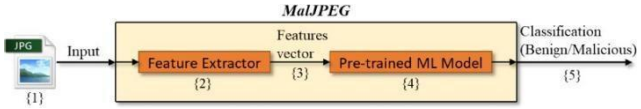
A. MalJPEG SOLUTION

In this section, we present the core contribution of our paper, the MalJPEG machine learning-based solution for the detec- tion of malicious JPEG images. MalJPEG receives a JPEG image as input {1}. The MalJPEG feature extractor {2} extracts the proposed features into a vector of features {3}. The MalJPEG feature extractor inspects the file statically, without actually viewing the image

(which requires executing image viewer software that itself might have a vulnerability), and traverses through the JPEG image file structure in orderto extract the features. The features are then transferred to a pretrained machine learning-based model {4} which outputsa classification (benign/malicious) {5} for the input image. We implemented MalJPEG and its inner modules, the fea- ture extractor {3} and machine learning model, {4} in Java programming language. The next section provides a detailed dehat are extracted using MalJPEG.

1) MalJPEG FEATURES

In this section, we present the compact set of discriminative features extracted by MalJPEG. We engineered these features after manually examining the structure of many benign and malicious JPEG images. We gained an understanding of how attackers use JPEG images in order to launch attacks and howit affects the JPEG file structure. We also found how mali- cious JPEG images differ from regular benign JPEG imagesin terms of file structure.



B. MACHINE LEARNING ALGORITHMS

C.

We applied machine learning classification algorithms on the datasets described in the previous section. In our experiments, we utilized the following commonly used, high performing classic and nonlinear machine learning classifiers: Decision Tree, Random Forest, and Gradient Boosting on Decision Trees (XGBoost and LightGBM). We chose these classifiers as they perform well on highly imbalanced datasets. In is important to mention that in our preliminary experiments we examined classifiers from families other than the decision tree family, such as Logistic Regression and Naïve Bayes, however they did not provide reasonable results; therefore, we did not include them in our evaluation.In addition, we
used the K-Nearest Neighbors classifier (K = 5) on Min- Hash datasets, because it is the only classifierthat can actually compare Min-Hash signatures using the
Hamming distance function. We chose to u. We applied the abovementioned machine learning clas- sifiers with Python using the following packages: scikit- learn,19 XGBoost,20 and LightGBM.21 We used the default configuration for all classifiers.

IV. EVALUATION

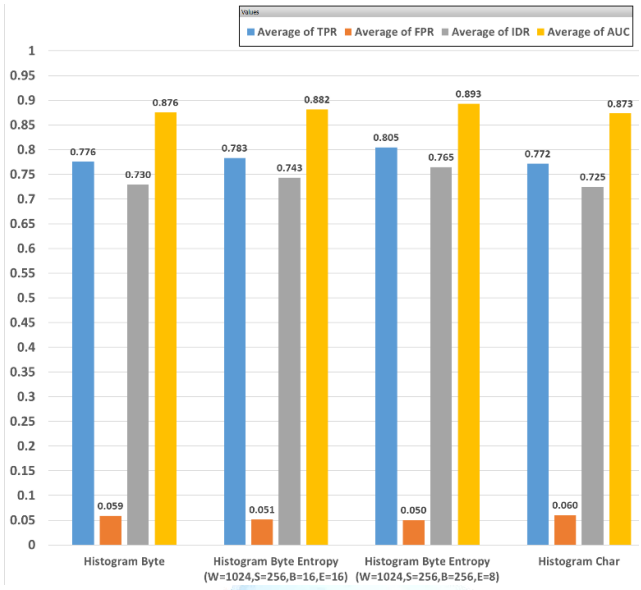
In this section, we evaluate MalJPEG. We begin by present- ing our data collection which we used for evaluation, and then describe the dataset creation process. Then, we present our research questions, evaluation metrics, experimental design, and results.

A. RESULTS

1) EXPERIMENT 1

Figure 7 presents a comparison between the detection results of the Random Forest classifier, evaluated on datasets created using the histogram methods presented in Table 3; we only provide the detection results of the Random Forest classifier, because it outperforms all of the other classifiers used in our experiments on all of the datasets created using the histogram methods. We set the Random Forest threshold to 0.05 (instead of the default of 0.5) to achieve the best results. The results are sorted from the highest to the lowest according to the AUC metric. As can be seen, the best results were achieved

ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 8, Issue 3, March 2023



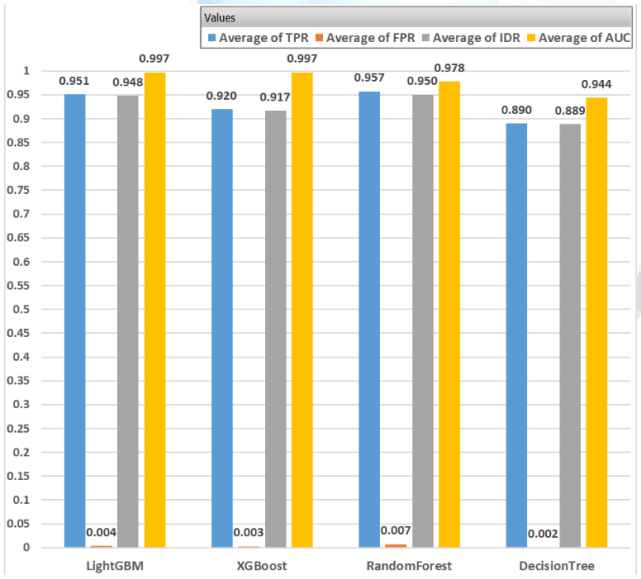


FIGURE 7. Detection results for the *Random Forest* classifier on datasetscreated using different histogram feature extraction methods.

ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 8, Issue 3, March 2023

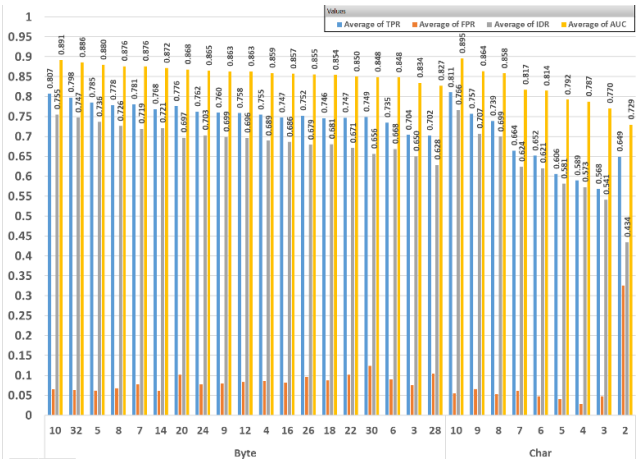


FIGURE 8. Detection results for the *K-Nearest Neighbors* classifier on datasets created using *Min-Hash* feature extraction methods with different configurations.

using the byte entropy histogram: TPR= 0.805, FPR =0.05, IDR= 0.765, and AUC =0.893.

Figure 8 presents a comparison between the detection results of the K-Nearest Neighbors classifier evaluated on datasets created using the Min-Hash methods. It is important to mention that the K-Nearest Neighbors classifier is the only classifier to use with Min-Hash datasets, since it is the only classifier that can actually compare Min-Hash signatures using a distance function (see Section IV-B.2; there is no actual order between the Min-Hash signature's numbers, thus regular machine learning algorithms are not effective on it.

We used the K-Nearest Neighbors classifier with K = 5 and distance function =Hamming. We set the K-Nearest Neighbors classifier threshold to 0.05 (instead of the default of 0.5) to achieve the highest results. The results are sorted from the highest to the lowest according to the AUC metric.

FIGURE 9. Detection results of the machine learning classifiers on adataset containing *MalJPEG* features.

TABLE 4. Summary of the configurations that provide the best results forboth histogram and Min-Hash methods.

Feature Extraction Method	Configuration	Number of features	Classifier	TPR	FPR	IDR	AUC
Entropy Histogram	Basic Unit = Byte, Window Size = 1024, Stride = 256, Bytes Axis Size = 256, Entropy Axis Size = 8	2048	*Random Forest*	0.805	0.050	0.765	0.893
Min-Hash	Basic Unit = Byte, Hash Functions = 200, Window Size = 10, Stride = 1	200	*Random Forest*	0.810	0.054	0.766	0.895
MalJPEG	N/A	10	*LightGBM*	**0.951**	**0.004**	**0.948**	**0.997**

V. DISCUSSION AND CONCLUSION

In this paper, we present MalJPEG, a machine learning- based solution for efficient detection of unknown malicious JPEG images. To the best of our knowledge, we are the first to present a machine learning-based solution tailored specifically for the detection of malicious JPEG images. MalJPEG extracts 10 simple but discriminative features from the JPEG file structure and leverages them with a machine learning classifier, in order to discriminate between benign and malicious JPEG images.

MalJPEG features are extracted based on the structure of the JPEG image. MalJPEG features were defined based on an understanding of how attackers use JPEG images in order to launch attacks and how it affects the JPEG file structure in comparison to regular benign JPEG images. We evaluate MalJPEG in four experiments. For our evalua- tion, we used a very large collection of 156,818 JPEG images: 155,013 (98.9%) benign and 1,805 (1.15%) malicious, col- lected between 2016 and 2018 from social media (benign images) andVirusTotal(malicious images). Note that the per- centage of malicious images in our collection is extremely low (1.15%). We intentionally prepared our collection that way so the collection reflects, as much as possible, the low percentage of malicious images (compared to benign images)in the real world. Note also that the extremely low percentage of malicious instances (positive) in the collection makes thedetection of malicious images in our experiments extremely difficult.constantly and quickly updated with their signatures. In con- trast, MalJPEG which is machine learning-based, is able to effectively detect both known and unknown malicious JPEGimages.

In addition, MalJPEGcan be parallelized easily and scaled to cope with the massive amount of images in the large- scale systems of enterprises. Based on the results of our experiments, it would be valuableto implement MalJPEG, in order to protect organizations, cloud services (e.g., Microsoft Office 365, Google Drive, etc.), social media (Facebook, Instagram, etc.), and their users from malicious JPEG images.

REFERENCES

[1] A. Cohen, N. Nissim, L. Rokach, and Y. Elovici, ''SFEM: Structural feature extraction methodology for the detection of malicious office doc- uments using machine learning methods,'' *Expert Syst. Appl.*, vol. 63, pp. 324–343, Nov. 2016.

[2] N. Nissim, A. Cohen, and Y. Elovici, ''ALDOCX: Detection of unknown malicious microsoft office documents using designated active learning methods based on new structural feature extraction methodology,'' *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 3, pp. 631–646, Mar. 2017.

[3] A. Cohen, N. Nissim, and Y. Elovici, ''Novel set of general descriptive features for enhanced detection of malicious emails using machine learning methods,'' *Expert Syst. Appl.*, vol. 110, pp. 143–169, Nov. 2018.

[4] N. Nissim, A. Cohen, C. Glezer, and Y. Elovici, ''Detection of malicious PDF files and directions for enhancements: A state-of–The art survey,'' *Comput. Secur.*, vol. 48, pp. 246–266, Feb. 2015.

[5] N. Nissim, Y. Lapidot, A. Cohen, and Y. Elovici, ''Trusted system-calls analysis methodology aimed at detection of compromised virtual machinesusing sequential mining,'' *Knowl.-Based Syst.*, vol. 153, pp. 147–175, Aug. 2018.

[6] A. Cohen and N. Nissim, ''Trusted detection of ransomware in a private cloud using machine learning methods leveraging meta-features from volatile memory,'' *Expert Syst. Appl.*, vol. 102, pp. 158–178, Jul. 2018.

[7] N. Nissim, A. Cohen, R. Moskovitch, A. Shabtai, M. Edri, O. Bar-Ad, and Y. Elovici, ''Keeping pace with the creation of new malicious PDF files using an active-learning based detection framework,'' *Secur. Inform.*, vol. 5, p. 1, Dec. 2016.

[8] N. Nissim, R. Moskovitch, L. Rokach, and Y. Elovici, ''Novel activelearning methods for enhanced PC malware detection in windows OS,'' *Expert Syst. Appl.*, vol. 41, no. 13, pp. 5843–5857, Oct. 2014.

[9] N. Nissim, R. Moskovitch, L. Rokach, and Y. Elovici, ''Detecting unknowncomputer worm activity via support vector machines and active learning,'' *Pattern Anal. Appl.*, vol. 15, no. 4, pp. 459–475, Nov. 2012.

[10] *proach for Malware Detection*. Accessed: 2019. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3383953