INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 7, Issue 4 - V7I4-1460)
Available online at: https://www.ijariit.com

Detection of macro-based attacks in office documents using Machine Learning

Aishwarya
vikasaishwaryasinha@gmail.com
Siddaganga Institute of Technology, Tumkur, Karnataka

Bhumica B.
bhumicab6@gmail.com
Siddaganga Institute of Technology, Tumkur, Karnataka

Sumit Suman
sumitsuman014@gmail.com
Siddaganga Institute of Technology, Tumkur, Karnataka

Ravi V.
rsheelavanth@gmail.com
Siddaganga Institute of Technology, Tumkur, Karnataka

ABSTRACT

With the rapid developments in internet, users share information through document files generated through online or offline office software. Due to implicit trust on the web and wide acceptance of these document files (such as PDF, DOC, Office Open XML), users share documents on the web by trusting third-party services which can be easily exploited by cybercriminals to inject malicious code (Malware) into the document files by various means of exploitations. These exploitations are undetectable and easily evaded on antivirus software which makes the problem of malware detection and classification even more complex. In recent years, the attacks that leverage office documents have gradually increased and thus harder to detect since malware authors use various ways to inject malicious code on to the office documents. They offer flexibility in document structure with numerous features for attackers to exploit. In this paper, a broad classification of macro based malicious document attack is provided along with a detailed description of the attack opportunities available using office documents. A hybrid malware analysis technique is proposed which thoroughly analyze the file for any macro attacks along with decision paradigms such as machine learning is used to detect and classify the malicious document present in Microsoft Office applications such as Word, Excel, Power point.

Keywords: *Macros, Malwares, Random forest etc.*

1. INTRODUCTION

Ever since coming to market, Microsoft Office applications such as Word, Excel, PowerPoint have been revered. These applications are commonplace with both enterprise users and well as personal computing users, thus making them a prime target for attacks just like the PDFs. Ever since the deployment of the software suite in 1990, there has been a rich history of being exploited by malicious authors. Despite all the efforts both from the security community, antivirus developers, as well as software developers, attacks using malicious documents are still rising. In Cisco annual report (2018), authors state the most commonly detected malicious file extensions worldwide during the period of January through September 2017 were Office suite extensions which contributed 38% of the file share, the highest among the group. In the report McAfee (2018), total macro malware has seen a steady increase since Q2 of 2016. Aside from the more traditional attacks, malicious documents are being used to drop ransomware payload. Sophos mentions the use of MSWord files to spread ransom ware of the Locky family.

Security solutions provided by various firms such as Sophos , Kaspersky and McAfee [1]–[6] have shown the rise in the cyber-attacks due to malicious code injected in to office and other document formats. Malware authors use various ways to inject malicious code on to the office documents, the malware can be targeted to the victim through three methods: Features: Feature updates are common in every software cycle. When the software starts becoming close to obsolete new features are introduced to provide a new and better experience for the user. In 2017 the security community noticed a trend for using Dynamic Data Exchange (DDE) attacks using Office Documents. These attacks use Windows PowerShell [1][6], a feature provided to documents by Office Suite, for execution of the obfuscated scripts and were listed among the top 10 malwares.

Macro: In general terms, macros are rule or pattern that specifies how a particular input sequence should be mapped to an output sequence on the basis of some procedure. Macros are widely used in the industry, but due to their common use in spreading viruses, they are blocked by default in Microsoft applications. Visual Basic Applications is the language used to program macros, and as such, they have most access to most Microsoft Windows calls vulnerabilities: The point has been made time and again that software vulnerabilities are the key reason why software fails under attacks. Microsoft updates their software regularly but the due to the far reach of the software and lack of updates installed the vulnerabilities are quite easily targeted by the attackers.

Most often though there is a combination of one or more methods to form an attack. In addition to exploiting software vulnerabilities, attackers liberally use features like Macros, Object Linking and Embedding (OLE) and Dynamic Data Exchange to form complex attacks.

1.1 Dynamic Analysis features from documents

There are a number of common feature types to extract from dynamic analysis [7-9]. For example, an early type of dynamic analysis was to modify the linker in the operating system to wrap each function call to the OS or other libraries with a special prologue and epilogue. In doing so the functions called could be tracked in the order of their occurrence and one could obtain a sequence of API or function calls. Such trackings of API calls can be used in many ways, and is often interpreted as a sequential ordering or as a directed graph. Special tracking can be added for common tasks, such as registry edits, files created or deleted, mutex operations, and TCP/IP calls. These are all common tasks or operations that malware might perform, so recording extra information (such as method arguments) can be beneficial to analysis. Ultimately, there are many ways to combine the API functions called and the operations performed, with many works using one of or both options, and tracking different subsets of actions. These approaches are often called "behaviour based", and make up a large portion of the dynamic features used.

1.2 Static Analysis features from documents

By its very nature, static analysis greatly reduces the scope of features options to consider for classification. One common choice is to use the raw-bytes themselves as features. A subset of the raw-byte approach is simply to search for and extract what appear to be ASCII strings. This approach assumes the least amount of knowledge and is widely applied to other file types because of its ease of application. Another approach is to instead compute windowed entropy over the raw bytes, mapping each file to a entropy sequence. Regardless of how processed, these approaches have an attractive simplicity at the cost of ignoring relevant properties. For example, while the raw bytes may be processed as one long linear sequence, the locality within a binary is non-linear. Different portions will relate to others through pointers in the storage format as well as various local and long jumps in the assembly. It is also common to build histograms from this information to reduce it to a fixed length format.

1.3 Contextual Features from documents

A third type of features is what we will call contextual features. These are features that are not properties of the malicious binary itself, but come from the context of how the malware may exist or be distributed. The use of contextual features is less common in research, but has been reported to be highly successful in practice. Such systems are generally graph-based in their approach. For example, "reputation" of the machines at which an executable file was found to make a determination about maliciousness, without looking at the content of the file itself. Others have followed this same strategy, and attempt to more precisely define the relations between files to improve results, and to merge both relations and file dependent features.

2. LITERATURE SURVEY

Office documents can be exploited with variety of ways such as macros, vulnerability and exploits, which can be easily achieved through tools and exploit kits available online [10-11]. In recent works only two papers were encountered on classifying the macro-based attacks in documents: Santos and Torres (2017) [14] used python OleFile and OleVba tools for extraction of features from documents but the accuracy of classifying is above 90 percent and can be used as a good filter for basic office documents with macros.

Bearden and Lo (2017) [15] work is an n-gram feature selection approach for classification of documents based on p-code present in VBA macros with higher accuracy of 96.3% with 1-gram opcodes compared to 93.4% of 2-gram opcodes. The above works uses static analysis and does not consider the contextual information of the macro affected files. In the proposed system, a hybrid analysis of documents is carried out in an isolated environment using Virtual box installed with necessary software. An obfuscation technique in malware makes the detection and classification even harder and reduces the accuracy of the prediction. Proposed model considers only macro-based attacks with an assumption of files with no obfuscation.

2.2 Objectives of the proposed system

Following research objectives (RO) have been set for the proposed system.
RO1: Build dataset for analysing the document files supporting various file formats.
RO2: Analyse various files and office documents and structures like Compound File Binary Format (doc/ppt/xls) and Office Open XML (OOXML) format for malware content.
RO3: Build Machine learning models to classify the malware affected documents.
RO4: Build tools for analysing the documents before exploits occur.

3. PROPOSED SYSTEM

The proposed model is designed to automate the process of malware detection and classification problem. Since office documents supports OOXML format which can be a cross platform support for users to use it for document generation. Proposed model

supports detection of documents containing various exploits generated by the methods such as Metasploit, GenerateMacro, ViperMonkey tools.

Figure 1. shows the proposed model which provides solutions to objectives mentioned earlier. **RO1** is met by generating the valid dataset (comprising of benign documents and documents with exploits) for training the model. Dataset generation is automated using Apache POI [12] which helps to generate various office documents. Office documents of size 1GB have been generated through the Apache POI and later macro exploits are embedded to the randomly selected files through metasploit tool and various tools available online.

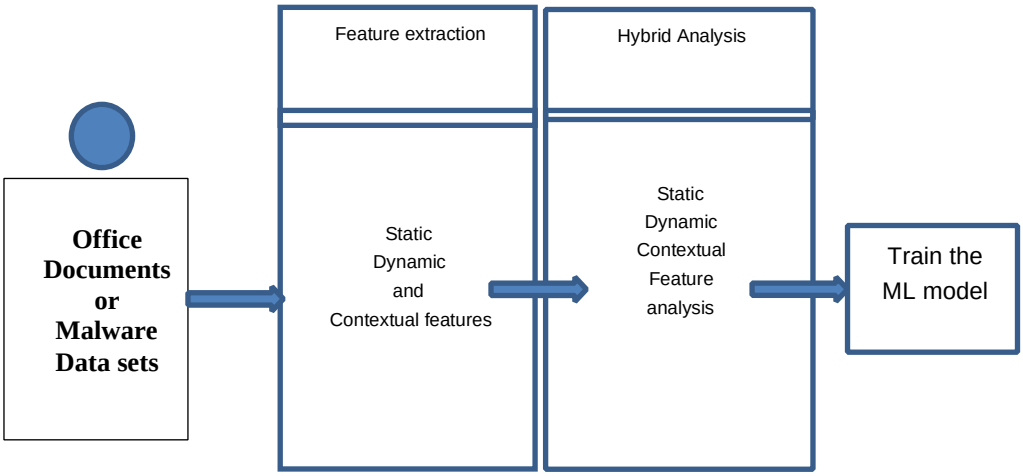

Chart -1: Proposed system

3.1 Feature extraction

Data set generated through apache POI will malware analysts to check for embedded macros in documents. Automated tool for static and dynamic analysis of documents is developed using python which extracts the features set to MalwareDB.csv file which is later used to build training set for the model. Table 1 shows the features extracted from the documents through the Dieder Oletool [13]. Feature set consists of more than 20 fields but in this work we consider only fields related to docx files as listed in Table-1.

Table 1: Features set from documents using Ole Tools

Flags	Macro	AutoOpen	Suspicious	IOCs	HexStrings	Base64	Legitmate	Dieder Strings	Richtext	DDElink	Filename
OLE	1	1	1	0	0	0	0	0	0	0	Demo1.docx
OpX	1	1	1	0	0	0	0	0	0	0	Demo2.docx

3.2 Hybrid analysis

The research objective **RO2** is met by building tools using python to integrate with OleVba for analyzing the CFB format files such as docx format. The figure in chart-2 shows the OleVba tool which is used to extract feature and later build the MalwareDB.csv file. OleVba tool helps to identify the file whether it opens an external executable file or executes the malicious macros when office documents are opened. OleVba provides various fields for hybrid analysis of documents out of which only feature which are necessary for classification are listed in table-1.

```
ravi@ravi-Lenovo-G50-80: ~/Downloads/Malwarefiles/01_emotet
ravi@ravi-Lenovo-G50-80:~/Downloads/Malwarefiles/01_emotet$ olevba bad.bin
olevba 0.56.1 on Python 3.8.10 - http://decalage.info/python/oletools
===============================================================================
FILE: bad.bin
Type: OLE
WARNING  invalid value for PROJECTDOCSTRING_Id expected 0005 got 0032
-------------------------------------------------------------------------------
VBA MACRO Teh9tkv0p83u4g.cls
in file: bad.bin - OLE stream: 'Macros/VBA/Teh9tkv0p83u4g'
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Private Sub Document_open()
Rvpv59xrvp7m2wb
End Sub
-------------------------------------------------------------------------------
VBA MACRO Larj61e5m5vzwh77.bas
in file: bad.bin - OLE stream: 'Macros/VBA/Larj61e5m5vzwh77'
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(empty macro)
-------------------------------------------------------------------------------
VBA MACRO A81c_pcot0t3c8.bas
in file: bad.bin - OLE stream: 'Macros/VBA/A81c_pcot0t3c8'
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Function Rvpv59xrvp7m2wb()
On Error Resume Next
Cju6d_0v951 = "Uvktlkqthymnm68w" + "Ti33bl29fw_53yu"
sf4 = Xj1p0_yor4q8g + Teh9tkv0p83u4g.StoryRanges.Item(2 / 2) + Gdmbhv991jtvgzq
```

Chart-2: Olevba tool output for document infected with emotet malware

Type	Keyword	Description
AutoExec	Document_open	Runs when the Word or Publisher document is opened
Suspicious	CreateTextFile	May create a text file
Suspicious	Create	May execute file or a system command through WMI
Suspicious	CreateObject	May create an OLE object
Suspicious	Base64 Strings	Base64-encoded strings were detected, may be used to obfuscate strings (option --decode to see all)

Chart-3: Suspicious API call from emotet malware

3.3 Train Machine learning model

The research objective **RO3** is met by generating the training set from valid data set in the MalwareDB.csv file which helps the model to classify the macro based malware attacks in documents. The dataset can be used for classifying the future macro based malware attacks which are unseen and the dataset helps as a basis for classifying new malware families. Random forest algorithm has been implemented in python and trained for classifying the macro based malwares from large collection of malicious and benign files.

3.4 Build tools for analyzing the document

RO4 is met by building python tool with integration of Olevba tool to extract useful features depending on the document to be analyzed for malware attacks, vba attacks, DDE link attacks etc. Tool is designed for automating the complete process of detecting and classifying the documents in large size.

4. RESULTS AND DISCUSSION

The proposed model considers the three popular machine learning models such as Random forest, Gradiant boost and Ada boost algorithm with malware csv file presented in Chart-4 showing the features extracted from OleVba tool given to various algorithms (Chart-5) and corresponding confusion matrix to interpret the accuracy of random forest algorithm with 96% accuracy.

```
set = pd.read_csv('C:/Users/Aishwarya/AppData/Local/Programs/Python/Python37/Lib/site-packag
```

```
In [124]:
MalwareDataset
```

Out[124]:

	Flags	Macro	AutoOpen	Suspicious	IOCs	HexStrings	Base64	DiederStrings	Legit	
0	OLE:	0	0	0	0	0	0	0	1	
1	OLE:	0	0	0	0	0	0	0	1	
2	OLE:	1	0	0	0	0	0	0	1	
3	OLE:	0	0	0	0	0	0	0	1	
4	OpX:	0	1	0	0	0	0	0	1	E:\
...	...	...	...	...	...	...	...	...	...	
194	OpX:	1	1	1	0	1	0	0	0	E
195	OpX:	1	1	1	0	1	0	0	0	I
196	OpX:	1	1	1	1	1	0	0	0	E
197	OpX:	1	1	1	1	0	0	0	0	E
198	OpX:	1	1	1	0	0	0	0	0	

199 rows × 10 columns

Chart-4: MalwareDB.csv file content

```
algorithms = {
        "RandomForest": sklearn.ensemble.RandomForestClassifier(n_estimators=50),
        "GradientBoosting": sklearn.ensemble.GradientBoostingClassifier(n_estimators=50),
        "AdaBoost": sklearn.ensemble.AdaBoostClassifier(n_estimators=100),
    }

results = {}
print("\nNow testing algorithms")
for algo in algorithms:
    clf = algorithms[algo]
    clf.fit(Legit_Train, Malware_Train)
    score = clf.score(Legit_Test, Malware_Test)
    print("%s : %f %%" % (algo, score*100))
    results[algo] = score

winner = max(results, key=results.get)
print('\nWinner algorithm is %s with a %f %% success' % (winner, results[winner]*100))
```

```
Now testing algorithms
RandomForest : 96.000000 %
GradientBoosting : 96.000000 %
AdaBoost : 96.000000 %

Winner algorithm is RandomForest with a 96.000000 % success
```

Chart-5: Training Machine learning models


```
print("False positive rate : %f %%" % ((CM[0][1] / float(sum(CM[0])))*100))
print('False negative rate : %f %%' % ( (CM[1][0] /float(sum(CM[1]))*100)))
```

```
False positive rate : 0.000000 %
False negative rate : 5.405405 %
```

Chart-6: Final confusion matrix

5. CONCLUSION

In the proposed model, three popular algorithms have been implemented and trained for classifying the macro based malwares on a CSV file which is generated by the automated tool. The process is static and can be even improved for online learning where files can be analyzed and also models can be parallel trained to detect and classify the documents in future infected with various attack vectors.

6. REFERENCES

[1] Kaspersky office document malware report 2020 - Google Search.

[2] P. Singh, S. Tapaswi, and S. Gupta, "Malware Detection in PDF and Office Documents: A survey," *Inf. Secur. J. Glob. Perspect.*, vol. 29, no. 3, pp. 134–153, May 2020, doi: 10.1080/19393555.2020.1723747.

[3] "McAfee Labs COVID-19 Threats Report, July 2020," p. 40.

[4] "McAfee Labs Threats Reports – Threat Research | McAfee." https://www.mcafee.com/enterprise/en-in/threatcenter/mcafee-labs/reports.html (accessed Oct. 04, 2020).

[5] "The Rise of Document based Malware - Data Threat Detection and Prevention | Sophos Security Topics - Virus, Malware, Web, Antivirus and Social Media Security Trends." https://www.sophos.com/en-us/security-newstrends/security-trends/the-rise-of-document-based-malware.aspx (accessed Oct. 04, 2020).

[6] "View reports for Advanced Threat Protection - Office 365 | Microsoft Docs." https://docs.microsoft.com/enus/microsoft-365/security/office-365-security/view-reports-for-atp?view=o365-worldwide (accessed Oct. 04, 2020).

[7] Laliberte, M. (2018, 03). *Watchguard's q4 2017 internet security report released; malicious office document usage on the rise*. Retrieved from https://www.secplicity.org/2018/03/27/watchguards-q4-2017-internet-security-reportreleased-malicious-office-document-usage-on-the-rise/

[8] Nissim, N., Cohen, A., & Elovici, Y. (2017). Aldocx: Detection of unknown malicious microsoft office documents using designated active learning methods based on new structural feature extraction methodology. *IEEE Transactions on Information Forensics and Security*, 12(3), 631–646. doi:10.1109/TIFS.2016.2631905

[9] Qbeitah, M. A., & Aldwairi, M. (2018). Dynamic malware analysis of phishing emails. In *Information and communication systems (icics), 2018 9th international conference on* (pp. 18–24), Irbid, Jordan . doi:10.1142/S2424835518500029. Russinovich, M. (2019, 03). *Process monitor v3.52*. Retrieved from https://docs.microsoft.com/en-us/sysinternals/downloads/procmon

[10] Metasploit: https://www.metasploit.com/

[11] Vipermonkey: https://github.com/decalage2/ViperMonkey

[12] Apache POI: https://poi.apache.org/

[13] Oletools: https://github.com/decalage2/oletools

[14] Santos, S. D. L., & Torres, J. (2017). Macro malware detection using machine learning techniques - a new approach. In Proceedings of the 3rd international conference on information systems security and privacy - volume 1: Icissp, (p. 295–302). Proto, Portugal: SciTePress

[15] Bearden, R., & Lo, D. C.-T. (2017). Automated Microsoft office macro malware detection using machine learning.

[16] In Big data (big data), 2017 ieee international conference on (pp. 4448–4452), Boston, MA, USA. https://ieeexplore.ieee.org/abstract/document/8258483

