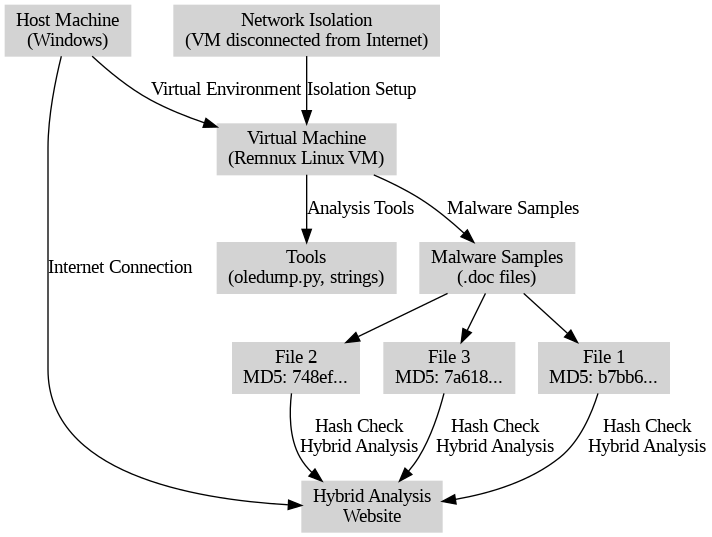
**Executive Summary**

In this lab, I analyzed three potentially malicious document files to determine their behaviors and identify any Indicators of Compromise (IOCs). I conducted the analysis in an isolated virtual machine environment to avoid contaminating the host machine. Using a combination of manual inspection tools and automated analysis on Hybrid Analysis, I assessed each file’s macros, strings, and potential network activity. The findings highlighted varying threat levels, ranging from benign to highly malicious behaviors, requiring recommendations for containment and monitoring.

**Topology**



**Key Syntax**

|  |  |  |
| --- | --- | --- |
| **Task** | **Command/Configuration** | **Description** |
| **Install oledump.py** | wget http://blog.didierstevens.com/programs/oledump-py/ | Downloaded the oledump.py script from Didier Stevens' blog. |
| **Verify oledump.py** | `python3 oledump.py -m | more` |
| **Extract Sample Files** | unzip -P m@lwar3! samples.zip | Extracted malware samples from a password-protected ZIP file. |
| **Inspect File Type** | file <filename> | Used to determine the file type of each sample, confirming they are Microsoft Word documents. |
| **Analyze with oledump.py** | python3 oledump.py <filename> | Analyzed each file to detect the presence of macros. |
| **Inspect Specific Macro** | python3 oledump.py -s <stream\_number> -v <filename> | Used to examine specific macro streams within each file for suspicious functions. |
| **Extract Macro Stream** | python3 oledump.py -s <stream\_number> -d <filename> > output.bin | Extracted the macro streams for further analysis, saving them to output files. |
| **View Strings in File** | strings <filename> | Displayed readable strings from each file to identify any embedded commands or URLs. |
| **Disable VM NIC** | VM Settings > Network Adapter > Disable | Disabled the NIC to isolate the VM from the network, preventing malware from reaching the internet or other hosts. |

**Verification**

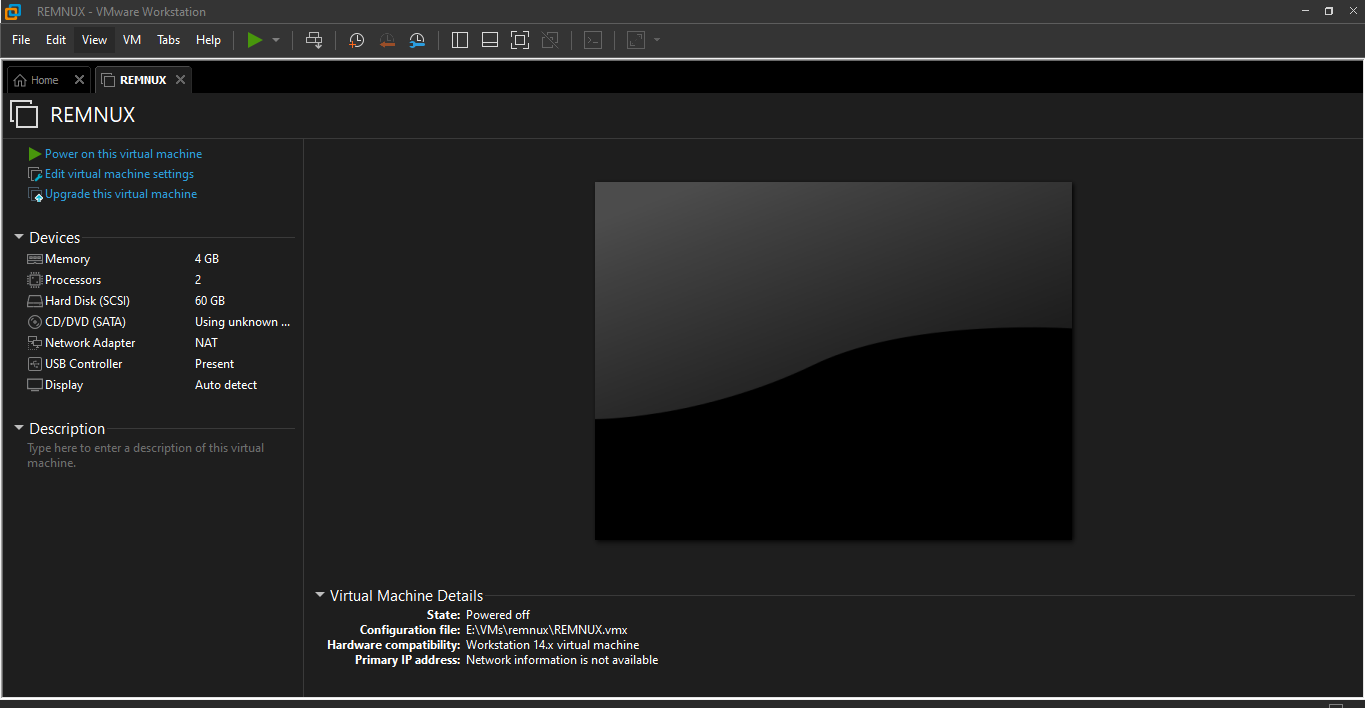
**Task One**

To complete this task, I started by setting up a virtual machine (VM) environment to perform malware analysis within a secure and isolated system, ensuring my host machine remained uncontaminated. The following steps outline the entire process:

1. Downloaded and Installed REMnux  
   I began by downloading REMnux, a Linux distribution designed for malware analysis, from the official REMnux website. Using VMware Workstation Pro 17 as my hypervisor, I imported the REMnux virtual appliance to set up the VM for analysis.

A screenshot of a computer

Description automatically generated



1. Installed Oledump  
   After setting up REMnux, I verified the presence of oledump.py, a Python script used to analyze OLE files, which are common in Microsoft Office documents and may contain malicious macros. To ensure oledump.py was updated, I visited Didier Stevens' blog and downloaded the latest version of the script.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Verified Oledump Installation  
   I ran the command $ python oledump.py -m | more in the terminal to verify that oledump.py was functioning correctly and to review its documentation. This output provided a comprehensive list of options available for analyzing OLE files.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

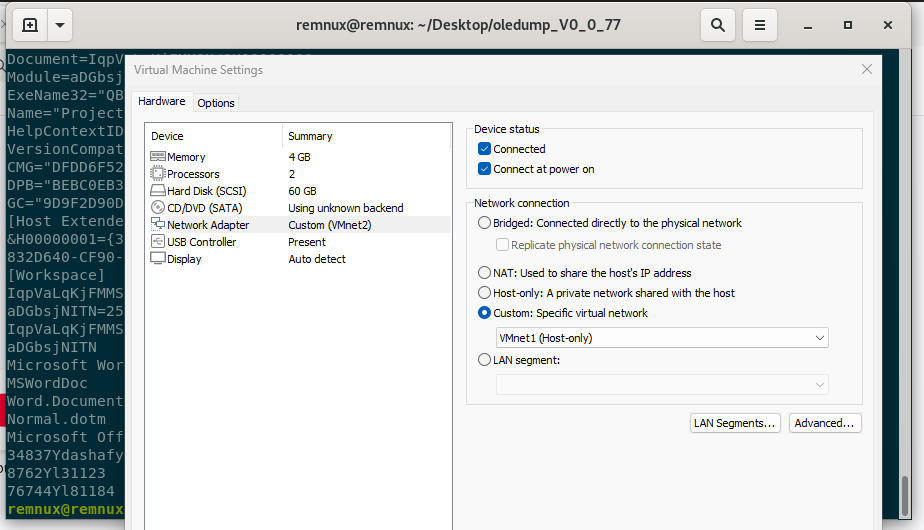
Description automatically generated

1. Downloaded Lab Sample Files  
   Once the setup was complete, I downloaded the lab files containing malware samples as a password-protected ZIP file. Following the instructions, I used the password m@lwar3! to extract these files onto the REMnux VM.

A screenshot of a computer

Description automatically generated

1. Isolated the VM  
   To prevent any accidental network contamination, I disabled the Network Interface Card (NIC) on the REMnux VM. This step ensured the VM was completely isolated from any network, minimizing the risk of malware spreading beyond the analysis environment.



By following these steps, I successfully configured a secure virtual machine environment with REMnux and installed the necessary tools for analyzing malicious documents. This careful setup not only protected my host machine but also allowed for a controlled and safe examination of potentially harmful files.

**File 1: MD5 b7bb6d16c9caaf36e14638a647c67715**

I used the file command in the terminal to inspect b7bb6d16c9caaf36e14638a647c67715, which revealed it is a Microsoft Word document.

A screenshot of a computer

Description automatically generated

Next, I used the oledump.py utility to inspect the file with the command python3 oledump.py b7bb6d16c9caaf36e14638a647c67715. The analysis showed that the file contains a macro in Stream 7.

A screenshot of a computer

Description automatically generated

i. **Does it contain macros?**

* Yes, it contains a macro in Stream 7.

ii. **How can you tell if it contains macros?**

* The presence of the letter "M" next to Stream 7 in the oledump output indicates that this stream contains a macro.

iii. **If the file contains macros, identify the streams that contain macros:**

* Stream 7: Macros/VBA/ThisDocument

iv. **If the file contains macros, inspect them:**

* I inspected the macro in Stream 7 using the command python3 oledump.py -s 7 -v b7bb6d16c9caaf36e14638a647c67715.

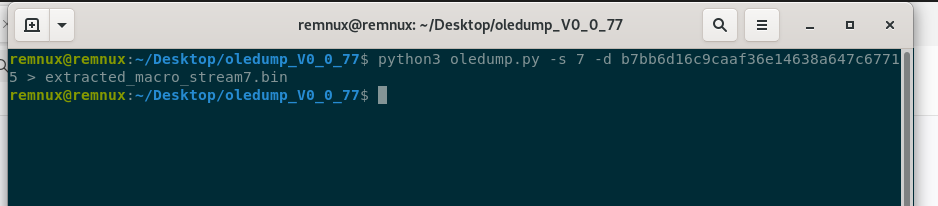
A screenshot of a computer

Description automatically generated

* The macro contains a subroutine named AutoOpen that displays "Hello World" upon document opening. This behavior appears benign.

v. **Attempted to extract those streams:**

* I used python3 oledump.py -s 7 -d b7bb6d16c9caaf36e14638a647c67715 > extracted\_macro\_stream7.bin to extract the macro stream for further analysis, which was successful.



vi. **Can you identify any significant Indicators of Compromise (IOCs)?**

* No significant IOCs were identified, as the macro only displays a simple text message.

vii. **Do you think the file is malicious?**

* No.

viii. **What makes you think the file is malicious (or not)?**

* The macro contains benign code that merely outputs "Hello World" and lacks any suspicious functionality.

**File 2: MD5 748ef5288c8388d43a89515ef43457a0**

I used the file command in the terminal to inspect 748ef5288c8388d43a89515ef43457a0, which revealed it is a Microsoft Word document.

A screenshot of a computer

Description automatically generated

Next, I used the oledump.py utility to inspect the file with the command python3 oledump.py 748ef5288c8388d43a89515ef43457a0. The analysis showed that the file contains macros in Streams 8, 9, and 10.

A screenshot of a computer

Description automatically generated

i. **Does it contain macros?**

* Yes, it contains macros in Streams 8, 9, and 10.

ii. **How can you tell if it contains macros?**

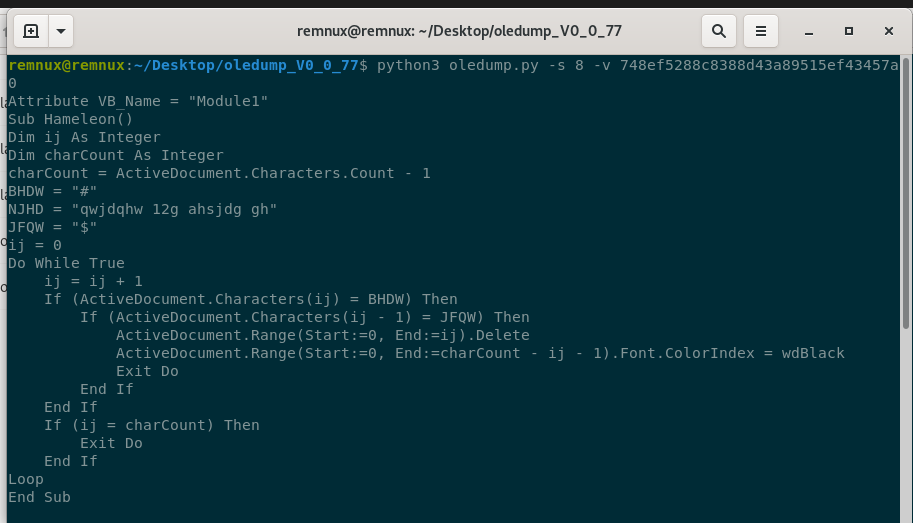
* The presence of the letter "M" next to Streams 8, 9, and 10 in the oledump output indicates that these streams contain macros.

iii. **If the file contains macros, identify the streams that contain macros:**

* Stream 8: Macros/VBA/Module1
* Stream 9: Macros/VBA/Module2
* Stream 10: Macros/VBA/ThisDocument

iv. **If the file contains macros, inspect them:**

* I inspected the macros using the following commands:
  + python3 oledump.py -s 8 -v 748ef5288c8388d43a89515ef43457a0



* + python3 oledump.py -s 9 -v 748ef5288c8388d43a89515ef43457a0

A screenshot of a computer program

Description automatically generated

* + python3 oledump.py -s 10 -v 748ef5288c8388d43a89515ef43457a0

A screenshot of a computer

Description automatically generated

* The macros contain obfuscated functions and code that manipulates document text and properties, with functions Hameleon and Kakarumba potentially designed to obscure intent. There is also a function Fuflmdjoo that uses the Shell function, suggesting potential malicious activity.

v. **Attempted to extract those streams:**

* I used the following commands to extract the macro streams for further analysis:
  + python3 oledump.py -s 8 -d 748ef5288c8388d43a89515ef43457a0 > extracted\_macro\_stream8.bin
  + python3 oledump.py -s 9 -d 748ef5288c8388d43a89515ef43457a0 > extracted\_macro\_stream9.bin
  + python3 oledump.py -s 10 -d 748ef5288c8388d43a89515ef43457a0 > extracted\_macro\_stream10.bin

A screenshot of a computer program

Description automatically generated

* The extraction of all macro streams was successful.

vi. **Can you identify any significant Indicators of Compromise (IOCs)?**

* Yes, the use of obfuscated strings, functions like Shell, and document manipulation functions indicate potentially harmful behavior.

vii. **Do you think the file is malicious?**

* Yes.

viii. **What makes you think the file is malicious (or not)?**

* The presence of obfuscation, calls to system-level commands through Shell, and manipulative functions suggest that the file may perform unwanted actions if opened.

**File 3: MD5 7a618482be272bb1fcb4af69a3f649a3**

I used the file command in the terminal to inspect 7a618482be272bb1fcb4af69a3f649a3, which revealed it is a Microsoft Word document.

A screenshot of a computer

Description automatically generated

Next, I used the oledump.py utility to inspect the file with the command python3 oledump.py 7a618482be272bb1fcb4af69a3f649a3. The analysis showed that the file contains macros in Streams 8 and 14.

A screenshot of a computer

Description automatically generated

i. **Does it contain macros?**

* Yes, it contains macros in Streams 8 and 14.

ii. **How can you tell if it contains macros?**

* The presence of the letter "M" next to Streams 8 and 14 in the oledump output indicates that these streams contain macros.

iii. **If the file contains macros, identify the streams that contain macros:**

* Stream 8: Macros/VBA/IqpVaLqKjFMMSN
* Stream 14: Macros/VBA/aDGbsjNITN

iv. **If the file contains macros, inspect them:**

* I inspected the macros using the following commands:
  + python3 oledump.py -s 8 -v 7a618482be272bb1fcb4af69a3f649a3

A screenshot of a computer

Description automatically generated

* + python3 oledump.py -s 14 -v 7a618482be272bb1fcb4af69a3f649a3

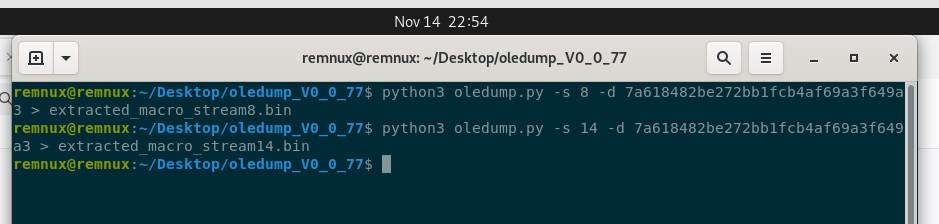
A screenshot of a computer program

Description automatically generated

* Stream 8 includes a function wNjqSj() that calls Shell, indicating potential command execution. Stream 14 has obfuscated strings and concatenations, further pointing toward malicious intent.

v. **Attempted to extract those streams:**

* I used the following commands to extract the macro streams for further analysis:
  + python3 oledump.py -s 8 -d 7a618482be272bb1fcb4af69a3f649a3 > extracted\_macro\_stream8.bin
  + python3 oledump.py -s 14 -d 7a618482be272bb1fcb4af69a3f649a3 > extracted\_macro\_stream14.bin



* The extraction of both macro streams was successful.

vi. **Can you identify any significant Indicators of Compromise (IOCs)?**

* Yes, the use of obfuscated strings, complex concatenations, and the Shell function in macros indicate potential execution of harmful commands.

vii. **Do you think the file is malicious?**

* Yes.

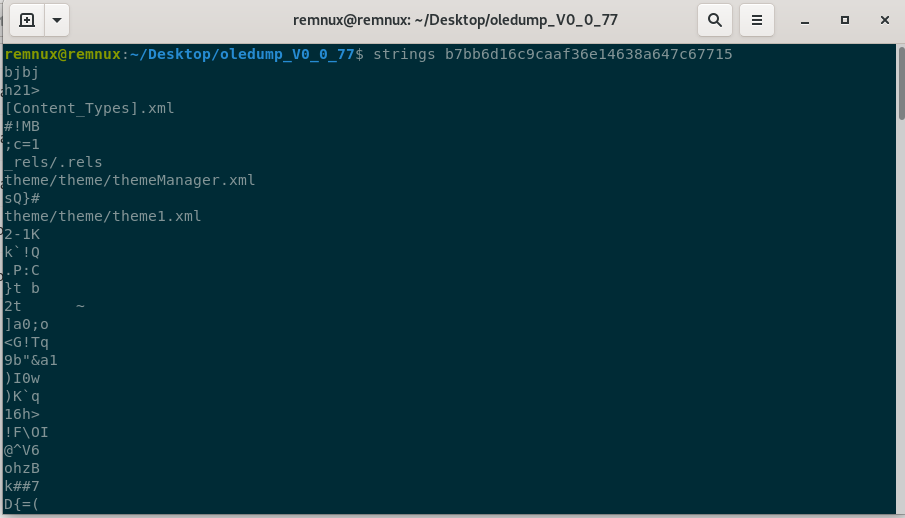
viii. **What makes you think the file is malicious (or not)?**

* The macros contain obfuscated code and utilize the Shell function, suggesting intent to execute potentially harmful actions, aligning with common tactics in malicious documents.

**Strings**

**File 1: MD5 b7bb6d16c9caaf36e14638a647c67715**

I used the strings command to examine the contents of b7bb6d16c9caaf36e14638a647c67715.



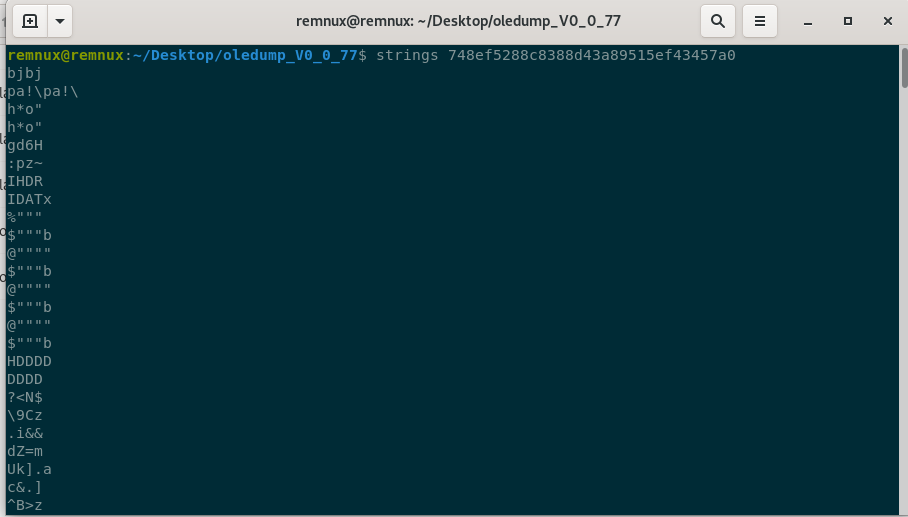
A screenshot of a computer

Description automatically generated

This inspection revealed standard document components, metadata, and a few key strings, including "Hello World," which aligns with the macro's behavior to display this message upon opening. There were no additional suspicious function calls or objects, suggesting the file is benign.

**File 2: MD5 748ef5288c8388d43a89515ef43457a0**

I ran the strings command on 748ef5288c8388d43a89515ef43457a0.



A screenshot of a computer

Description automatically generated

This file contained various obfuscated strings, metadata, and specific terms like "Shell" and "Hameleon," which indicate possible attempts to hide functionality. The presence of function names like Fuflmdjoo and Kakarumba suggests an intention to manipulate or control certain elements. Reviewing these function calls, it appears the file may execute system commands or other suspicious actions if opened.

**File 3: MD5 7a618482be272bb1fcb4af69a3f649a3**

Using the strings command on 7a618482be272bb1fcb4af69a3f649a3 produced obfuscated strings and concatenated commands, including terms such as "Shell" and AutoOpen.

A screenshot of a computer

Description automatically generated

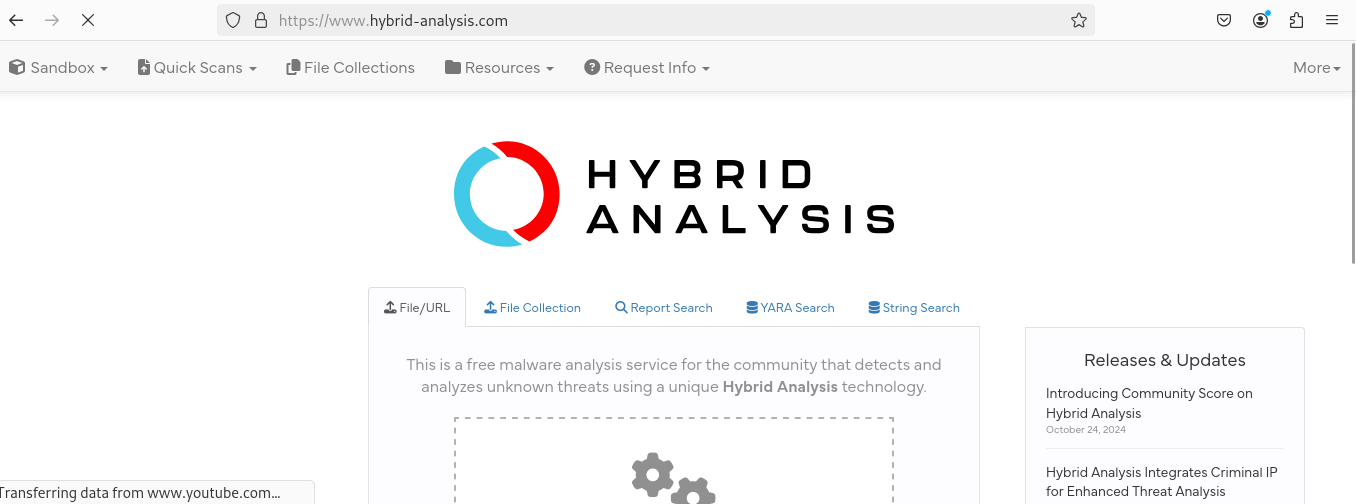
A screenshot of a computer

Description automatically generated

The file contains function names like wNjqSj, which, along with obfuscated text, suggest potential malicious behavior by executing hidden commands upon opening. These observations align with known tactics for disguising harmful functions within documents, pointing to potentially malicious intent.

**Hybrid Analysis**

After completing my manual analysis, I connected my VM to the internet. I opened my web browser and navigated to [Hybrid Analysis](https://www.hybrid-analysis.com/).



I entered the MD5 hash of each file to search for existing analysis results and gain further insights into potential malicious behaviors. This allowed me to view detailed automated analysis data, which complemented my manual findings.

**File 1: MD5 b7bb6d16c9caaf36e14638a647c67715**

* **Malware Analysis Summary**  
  The MS Word document returned a high Threat Score of 100/100 on a Windows 11 environment, though it showed no specific malicious indicators on other Windows versions. Antivirus scans also marked the file as clean. While some indicators suggested potential phishing behavior, such as attempts to access mail credentials in the registry, no evidence of active command-and-control (C2) domains or harmful payloads was detected. These findings suggested a cautious approach, but no conclusive malicious activity was confirmed.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* **Process Activity Section**  
  I reviewed the process activity details for this file. No external connections or significant domain involvement was identified, reducing the likelihood of active network-based threats.

**File 2: MD5 748ef5288c8388d43a89515ef43457a0**

* **Malware Analysis Summary**  
  Hybrid Analysis flagged this MS Word document as a Trojan, specifically classified as Trojan.Msword, with a high Threat Score of 100/100 and an antivirus detection rate of 85%. The document attempted connections to the domain bigdiscountsonline.info and multiple IP addresses, indicating possible command-and-control (C2) activity or data exfiltration attempts.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* **Process Activity Section**  
  In the process activity section, I identified network interactions with the domain bigdiscountsonline.info, suggesting potential C2 activity. Based on these findings, I determined that immediate containment was advisable, along with monitoring for any network traffic associated with this domain and its IP addresses to mitigate further risks.

**File 3: MD5 7a618482be272bb1fcb4af69a3f649a3**

* **Malware Analysis Summary**  
  Hybrid Analysis classified this MS Word document as a downloader (specifically VB.Heur2.EmoDldr) with a high Threat Score of 100/100, indicating strong malicious activity across different Windows environments. This classification implied the file’s primary function was likely to retrieve and install additional malware, posing a significant risk if left uncontained.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* **Process Activity Section**  
  I observed that this file exhibited downloader behavior, such as writing data to remote processes, installing hooks, and reading machine-specific identifiers. It also generated logs like powershell.exe.log, hinting at command execution. The document attempted to contact finance-advisors-ca.bid, likely to download further malicious payloads or communicate with a C2 server. Based on this, I recommended immediate containment to prevent further downloads and additional malicious actions.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **File** | **MD5 Hash** | **Classification** | **Threat Score** | **Domain Involvement** | **Recommended Action** |
| File 1 | b7bb6d16c9caaf36e14638a647c67715 | Potential Phishing | 100/100 (Windows 11) | None Identified | Caution, though no confirmed malicious activity |
| File 2 | 748ef5288c8388d43a89515ef43457a0 | Trojan.Msword | 100/100 | bigdiscountsonline.info | Immediate containment, monitor network traffic |
| File 3 | 7a618482be272bb1fcb4af69a3f649a3 | VB.Heur2.EmoDldr (Downloader) | 100/100 | bigdiscountsonline.info | Immediate containment, prevent further downloads |

By following these steps and reviewing the data from Hybrid Analysis, I gained a comprehensive view of each malware sample's behavior and confirmed the need for immediate action on Files 2 and 3.

**Conclusion**

In this lab, I successfully analyzed three potentially malicious Microsoft Word documents in an isolated virtual machine environment, utilizing both manual tools and automated analysis through Hybrid Analysis. Each file was examined for macros, strings, and network activities to identify Indicators of Compromise (IOCs). While Files 2 and 3 displayed significant malicious behaviors, including command execution and potential command-and-control communication, File 1 appeared benign. The analysis concluded with clear recommendations for containment and monitoring based on the detected threat levels.

**References**

1. Didier Stevens. *oledump.py - Analyze OLE files (like Word and Excel)*. Retrieved from [Didier Stevens Blog](http://blog.didierstevens.com/programs/oledump-py/).
2. REMnux Documentation. *Get REMnux Virtual Appliance*. Retrieved from [REMnux](https://docs.remnux.org/install-distro/get-virtual-appliance).
3. *Hybrid Analysis: Advanced Malware Analysis Sandbox*. Retrieved from [Hybrid Analysis](https://www.hybrid-analysis.com/).
4. MITRE ATT&CK. *Command and Control Techniques in Malware*. Retrieved from MITRE ATT&CK Framework.
5. Microsoft Documentation. *Malicious Macros in Word Documents*. Retrieved from [Microsoft Security Documentation](https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/detect-malicious-macros).
6. Linux man pages. *file command documentation*. Retrieved from Linux manual.