Document File Analysis 1 (Macros)

By: Ryan Stewart

In the context of **static analysis** in software development, the grep command or similar text-searching techniques can be used to search for patterns, keywords, or **regular expressions** within the source code or files of a program. Static analysis involves examining the code **without** executing it, aiming to find potential issues, security vulnerabilities, or adherence to coding standards.

Here's a general idea of how the grep command might be used in static analysis:

Searching for Vulnerabilities:

- Developers might use grep to search for known vulnerabilities or security issues in their codebase by looking for specific patterns or function calls associated with security problems.

grep pattern file

This command searches for the specified pattern in the given file and prints the lines containing the pattern.

For instance, if you want to search for the word "example" in a file named "sample.txt," you would use:
grep "example" sample.txt

Strings sample.txt | grep (http,https, or smtp)

What is olemeta?

Olemeta is a tool integrated into the python-oletools package, a collection of Python utilities designed for the analysis of Microsoft OLE2 files. These files commonly originate from Microsoft Office applications like Word and Excel.

The primary purpose of **olemeta** is to parse OLE files and retrieve standard properties embedded within them. These properties encompass essential information about the file, such as metadata, creation date, author details, and other pertinent attributes.

olemeta my_document.docx

This command initiates the analysis of the specified Microsoft Word document (my_document.docx), extracting relevant metadata and providing insights into the file's characteristics.

What is oleid?

`Oleid` is a tool designed to identify and classify OLE files based on their characteristics and features. It can be used to determine the type of OLE file and provides information about potential embedded objects, streams, and other properties.

Here's a basic example of using `oleid`:

oleid my_document.docx

Replace `my_document.docx` with the path to your OLE file. The `oleid` command will then analyze the file and provide information about its OLE structure and features.

IOCs:

- **File Format:** This IOC refers to the specific format or extension of the file. Unusual or unexpected file formats may indicate potential security risks. For example, executable files with unusual extensions or disguised as common document formats can be a red flag.
- **Application Name:** Identifying the application associated with a file can be crucial. Certain applications may be known vectors for malware or may have vulnerabilities that attackers exploit. Monitoring for unexpected or unauthorized applications is essential for security.
- **Encrypted:** The presence of encryption in a file can be an IOC, especially if it's unexpected or not in line with the organization's encryption policies. Encrypted files can be used to conceal malicious content or activities.
- **VBA Macros:** Visual Basic for Applications (VBA) macros within documents can pose a security risk. Attackers often use VBA macros to deliver malicious payloads or execute code on a victim's system. Identifying and scrutinizing VBA macros is important for security assessments.

What is olevba?

`olevba` is a tool used for analyzing Microsoft Office files (such as Word, Excel, and PowerPoint documents) to detect and extract **Visual Basic for Applications** (VBA) macros. VBA macros are scripts embedded within Office documents that can be used for automation, but they can also be exploited by malicious actors to deliver malware or perform other malicious activities.

Here's a basic example of using `olevba`:

```
olevba my_document.docm
```

Replace `my_document.docm` with the path to your Microsoft Office document containing VBA macros. `olevba` will then analyze the document and provide information about the detected VBA macros, potential security risks, and any suspicious or malicious behavior.