**SECPY**

**Automated Yara Rule**

Contents

[1. Summary 3](#_Toc121682923)

2. Background ……………………………………………………………………………………………………………………………………… 3

3. Prerequisite ……………………………………………………………………………………………………………………………………….4

4. Features …………………………………………………………………………………………………………………………………………… 5

5. Language and Libraries …………………………………………………………………………………………………………………… 6

6. Functions …………………………………………………………………………………………………………………………………….. 6

7. Generated files ………………………………………………………………………………………………………………………………. 7

8. Usage ………………….………………………………………………………………………………………………………………………….8

9. Example …………………………………………………………………………………………………………………………………………8

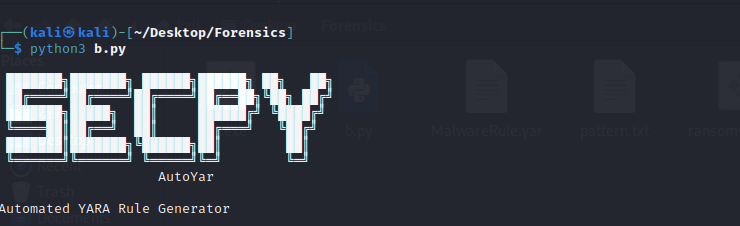
Output ………………………………………………………………………………………………………………………………………………8

Reference ……………………………………………………………………………………………………………………………………….. 9

**YARA Rule Generator Tool**

**Summary:**

The YARA Rule Generator tool is a Python script designed for creating custom YARA rules to detect patterns and characteristics in suspicious files. YARA (Yet Another Recursive Acronym) is a widely used tool and pattern-matching language for identifying and classifying malware. The YARA Rule Generator simplifies the process of generating YARA rules by automating the extraction of various file attributes, such as MD5 and SHA256 hashes, fuzzy hashes (using ssdeep), file strings, and entropy values.



**Background:**

YARA rules are essential in cybersecurity for identifying and categorizing malware based on specific patterns or characteristics. These rules define the conditions that, when met, indicate the presence of malicious code or behavior within a file. YARA allows for the creation of complex rulesets that enhance the detection capabilities of security professionals and tools.

**Prerequisites:**

Create a .txt file and write the black listed string in them.

<https://bazaar.abuse.ch>

Install the malware (ransomware) from the above link.

* **Update the system:**

sudo apt-get update

* **Install Python3**

sudo apt install python3

* **Creating a Virtual Environment:**

python3 -m venv myenv

* **Activating the Virtual Environment:**

source myenv/bin/activate  
  
Note: Make sure that you are creating virtual environment in the same directory where you python script is. Install ssdep in that virtual environment.

* **Install ssdeep**

sudo apt-get install libfuzzy-dev

sudo apt-get install python3-dev

pip install ssdeep

* **Install yara:**

sudo apt-get install yara

pip install -r requirements.txt

**Features:**

1. **File Attribute Extraction:**
   * The tool extracts various attributes from a given file, including MD5 and SHA256 hashes, ssdeep fuzzy hash, and file strings.
2. **String Pattern Search:**
   * Utilizes the **strings** utility to extract printable character sequences from the file.
   * Searches for predefined patterns within the extracted strings.
3. **File Signature Check:**
   * Determines the file type based on its signature.
   * Supports the identification of known file types, such as DOS executables, ELF executables, ransomware, and trojan files.
4. **Entropy Calculation:**
   * Calculates the Shannon entropy of the file, providing a measure of its randomness or disorder.
5. **Dynamic YARA Rule Generation:**
   * Dynamically generates a YARA rule incorporating the extracted attributes and patterns.
   * Allows customization of rule name, pattern file, and threshold values.
6. **Rule Saving and Application:**
   * Saves the generated YARA rule to a specified output file.
   * Facilitates scanning a file against the generated rule using the YARA tool.

**Language and Libraries:**

* **Language:** Python
* **Libraries:**
  + **hashlib**: For MD5 and SHA256 hash calculation.
  + **ssdeep**: For generating fuzzy hashes.
  + **subprocess**: For running external commands like the **strings** utility and YARA tool.
  + **magic**: For determining the file type based on its signature.
  + **math**: For entropy calculation.

**Functions:**

1. **StringDump:**
   * Dumps printable character sequences from a file using the **strings** utility.
2. **SearchPattern:**
   * Searches for predefined patterns within a file.
3. **FileSignature:**
   * Checks the file signature to determine its type.
4. **calculate\_md5, calculate\_sha256:**
   * Calculate MD5 and SHA256 hashes of a file.
5. **calculate\_ssdeep:**
   * Generates the ssdeep fuzzy hash of a file.
6. **calculate\_entropy:**
   * Calculates the Shannon entropy of a file.
7. **generate\_yara\_rule:**
   * Dynamically generates a YARA rule based on file attributes and patterns.
8. **save\_yara\_rule:**
   * Saves a YARA rule to a specified output file.
9. **scan\_file\_with\_yara:**
   * Scans a file against a YARA rule using the YARA tool.
10. **main:**
    * Orchestrates the overall workflow, generating a YARA rule and scanning a file.

**Generated Files:**

* **stringDump.txt:**
  + Contains the printable character sequences extracted from the file using the **strings** utility.
* **<rule\_name>.yar:**
  + The generated YARA rule file based on the provided attributes, patterns, and thresholds.

**Usage:**

1. Provide the path to the target file (**malware\_file**).
2. Specify the desired **rule\_name** for the YARA rule.
3. Customize the **pattern\_file** containing predefined patterns.
4. Run the script to generate the YARA rule and save it to the specified output directory.

**Example:**

python yara\_generator.py

The YARA Rule Generator tool streamlines the process of creating custom rules for malware detection, providing a flexible and automated solution for cybersecurity professional

**Output:**

A screenshot of a computer

Description automatically generated

**Reference:**

https://www.researchgate.net/figure/Fuzzy-hashing-aided-enhanced-YARA-rules\_fig5\_347108793