



# Malware Analysis Case Using YARA & yarGen Automation Tool.

## ▼ What is YARA?

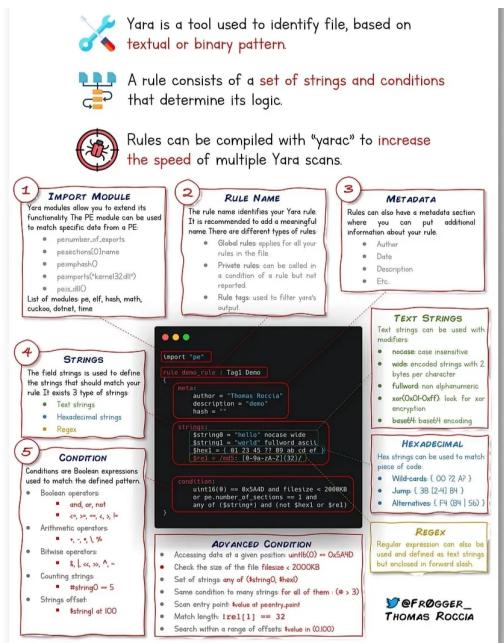
- **YARA** is Yet Another Ridiculous Acronym.
- **YARA** is a tool used for identifying and classifying malware by creating and applying pattern-based rules.
- **YARA** enables security researchers to define *rules* that match specific patterns of strings or binary sequences found in malicious files.

## ▼ Key Functions Of YARA.

- **Rule Creation:** **YARA** rules are used to describe the characteristics of malware based on strings, byte sequences, and patterns, allowing efficient matching.
- **Pattern Matching:** It scans files or processes for patterns that match defined rules, identifying suspicious or known malicious files.
- **Wide Applicability:** **YARA** can be used across different file types, operating systems, and processes.

- **Malware Classification:** It assists in classifying families of malware by grouping similar samples under defined rule sets.

## ▼ Anatomy Of YARA.



## ▼ What is yarGen Tool.

is a tool that automates the creation of **YARA** rules for malware analysis. It is designed to simplify the process of generating **YARA** signatures by analyzing files, extracting relevant strings, and eliminating common or irrelevant strings that could lead to false positives.

## ▼ Key Functions of yarGen.

- **Automatic YARA Rule Generation:** `yarGen` analyzes binary files (e.g., malware samples) and automatically generates **YARA** rules based on unique strings found in the files.
  - **Noise Filtering:** It filters out common strings like URLs, function names, and library references, which are often present in many legitimate files to reduce false positives.
  - **Score-based Selection:** `yarGen` assigns scores to strings based on uniqueness and relevance, ensuring that the most significant strings are included in the generated rules.

- **Metadata Extraction:** It extracts metadata from the files, such as file size, PE header details, and other attributes that can be included in the **YARA** rule.
- **Customizable Rule Generation:** Users can tweak parameters to include or exclude specific types of strings, set thresholds for string selection, and define custom strings for **YARA** rule creation.
- **Cross-Platform Compatibility:** It works on various platforms and can be used for malware in different file formats.

## ▼ Study Case (**steam\_api.dll**)

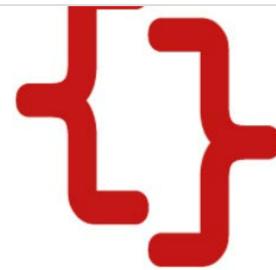
First of all i have introduced previously summary of **YARA** and **yarGen** tool, if you want to deeply understand **YARA** and **yarGen** tool look for these:

**YARA:**

Yara [TryHackMe]

Learn the applications and language that is Yara for everything threat intelligence, forensics, and threat hunting!

 <https://medium.com/@valerie7995/yara-tryhackme-c2d59c33b4a6>



**yarGen:**

<https://github.com/Neo23x0/yarGen>

Now, let's start to solve the case **steam\_api.dll** using **YARA** and **yarGen**.

We know that **steam\_api.dll** is a malicious and we want use **YARA** rule to detect pattern of it.

U can write your **YARA** rules to detect the patterns of **steam\_api.dll**, but we use **yarGen python tool** to automate this step and use the file **.yar** to detect the pattern.

## ▼ Installing YARA

In your kali Linux machine terminal:

1. Gain access root.
2. write the following command → `apt install yara`
3. To Check the tool is installed successfully → `yara -v` (check version of yara).
4. To how use YARA use → `yara -h`

## ▼ Installing yarGen.py

In your Kali Linux terminal machine:

1. git clone <https://github.com/Neo23x0/yarGen.git>
2. use → `ls` (to show listed directories and files).
3. U will find directory named (**yarGen**)
4. To enter **yarGen** directory → `cd yarGen`
5. use → `ls`
6. Install all dependencies with `pip install -r requirements.txt` (or `pip3 install -r requirements.txt`)
7. Run `python yarGen.py --update` to automatically download the built-in databases. The are saved into the '**./dbs**' sub folder.
8. See help with `python yarGen.py --help` for more information on the command line parameters.

```
usage: yarGen.py [options] rulefile [rulefile ...]

optional arguments:
  -h, --help            show this help message and exit
  -v, --version         show program's version number and exit
  -o, --output_dir=DIR Path to save for malware
  -r, --min_size=SIZE Minimum string length to consider (default=8)
  -c, --max_size=SIZE Maximum string length to consider (default=1024)
  -s, --high_scoring=SCORE Score required to set string as "highly specific"
  -l, --low_scoring=SCORE Score required to set string as "low specific"
  -w, --superrole-overlap Maximum number of strings that overlap to create a
                           superrole
  -t, --max-tokens=TOKENS Maximum length to consider (Default=1024)
  -u, --no-intelligent   No intelligent filtering will be applied
  -n, --no-case          Force the exclude all lowercase settings
  -e, --exclude=EXCLUDE
                        >>> exclude all given strings
  -o, --output_rule_file Output rule file
                        <<< rulefile1.rule rulefile2.rule ... >>>
```

## ▼ Detection Patterns Steps

1

First, i moved the malicious file in the same directory that the **yarGen** tool installed to facilitate writing commands.

```
File Actions Edit View Help
slash@localhost:~/ProactiveSecurityPractical/yarGen x slash@localhost:~/ProactiveSecurityPractical/yarGen x root@localhost:/home/slash/Downloads
slash@localhost:~/Downloads$ slash@localhost:~/Downloads$ cd /home/slash/Downloads
slash@localhost:~/Downloads$ slash@localhost:~/Downloads$ slash@localhost:~/Downloads$ ls
'Hash.Slash(1).ovpn' PhoneInfoga bin dev home latest lib64 mnt proc run srv sys tar4 tar5 usr
Hash.Slash.ovpn Seclists boot etc kali-red-sticker-16x9.jpg lib media opt root sbin steam_api.dll var
slash@localhost:~/Downloads$ slash@localhost:~/Downloads$ mv steam_api.dll /home/slash/ProactiveSecurityPractical
slash@localhost:~/Downloads$ slash@localhost:~/Downloads$ slash@localhost:~/Downloads$ ls
'Hash.Slash(1).ovpn' PhoneInfoga bin dev home latest lib64 mnt proc run srv sys tar4 tar5 usr
Hash.Slash.ovpn Seclists boot etc kali-red-sticker-16x9.jpg lib media opt root sbin steam_api.dll var
slash@localhost:~/Downloads$ slash@localhost:~/Downloads$ cd /home/slash/ProactiveSecurityPractical
slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ ls
steam_api.dll yarGen
slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ mkdir malwares
slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ ls
malwares yarGen
slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ cd malwares
slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ ls
steam_api.dll
slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ cd ..
slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ cd yarGen
slash@localhost:~/ProactiveSecurityPractical/yarGen$ slash@localhost:~/ProactiveSecurityPractical/yarGen$ slash@localhost:~/ProactiveSecurityPractical/yarGen$ ls
3rdparty LICENSE README.md dbs prepare-release.sh requirements.txt screens tools yarGen.py
```

```
File Actions Edit View Help
slash@localhost:~/ProactiveSecurityPractical/yarGen x slash@localhost:~/ProactiveSecurityPractical/yarGen x root@localhost:/home/slash/Downloads
slash@localhost:~/Downloads$ slash@localhost:~/Downloads$ cd /home/slash/ProactiveSecurityPractical
slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ ls
malwares steam_api.dll yarGen
slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ mv steam_api.dll /home/slash/ProactiveSecurityPractical/malwares
slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ ls
malwares yarGen
slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ cd malwares
slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ ls
steam_api.dll
slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ slash@localhost:~/ProactiveSecurityPractical/malwares$ cd ..
slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ slash@localhost:~/ProactiveSecurityPractical$ cd yarGen
slash@localhost:~/ProactiveSecurityPractical/yarGen$ slash@localhost:~/ProactiveSecurityPractical/yarGen$ slash@localhost:~/ProactiveSecurityPractical/yarGen$ ls
3rdparty LICENSE README.md dbs prepare-release.sh requirements.txt screens tools yarGen.py
```

2

Secondly, i will run a command using **yarGen** tool to automate writing **YARA** Rules.

2

Use this command to automate writing **YARA** Rules: `python yarGen.py -m [Path of malicious file]`

```
File Actions Edit View Help slash@localhost: ~/ProactiveSecurityPractical/yarGen x slash@localhost: ~/ProactiveSecurityPractical/yarGen x root@localhost: /home/yarGen x
slash@localhost:~/ProactiveSecurityPractical$ python yarGen.py -m /home/slash/ProactiveSecurityPractical/malwares

Yara Rule Generator
Florian Roth, August 2023, Version 0.24.0

The rule generator produces code to identify similarities between the files.
This is done by generating YARA rules. The output is a single file.
Note: Rules have to be post-processed
See this post for details: https://medium.com/@cyb3rops/121d29322282

[+] Using identifier 'malwares'
[+] Using reference 'https://github.com/Neo23x0/yarGen'
[+] Using prefix 'malwares'
[+] Processing PEStudio strings ...
[+] Reading goodware strings from database 'good-strings.db' ...
  (This could take some time and uses several Gigabytes of RAM depending on your db size)
[+] Loading ./dbs/good-imphashes-part4.db ...
[+] Total: 2539 / Added 2539 entries
[+] Loading ./dbs/good-exports-part6.db ...
[+] Total: 8065 / Added 8065 entries
[+] Loading ./dbs/good-strings-part5.db ...
[+] Total: 4230341 / Added 4230341 entries
[+] Loading ./dbs/good-exports-part1.db ...
[+] Total: 114741 / Added 106676 entries
[+] Loading ./dbs/good-exports-part2.db ...
[+] Total: 180298 / Added 65557 entries
[+] Loading ./dbs/good-exports-part9.db ...
[+] Total: 180298 / Added 0 entries
[+] Loading ./dbs/good-imphashes-part3.db ...
[+] Total: 6426 / Added 3887 entries
[+] Loading ./dbs/good-imphashes-part7.db ...
[+] Total: 9991 / Added 3565 entries
[+] Loading ./dbs/good-imphashes-part6.db ...
[+] Total: 10019 / Added 28 entries
```

3

After few seconds the tool generated file **.yar** contains all rules to detect this malicious file.

4

Use command `cat` (to read the `.yar` file)

5

In my case, I renamed the **.yar** file for personal purposes.

```
[slashed@caltech ~] $ ./gradlew :PracticalSecurity:practical:prepareRelease  
Dartify LICENSE README.md does prepare-release.sh requirements.txt screens tools jarGen.py yargon_rules.yar  
slashed@caltech ~] $ ./gradlew :PracticalSecurity:practical:release  
Dartify LICENSE README.md does prepare-release.sh requirements.txt screens tools jarGen.py yargon_rules.yar  
slashed@caltech ~] $ ./gradlew :PracticalSecurity:practical:release  
Dartify LICENSE README.md does prepare-release.sh projectSetAssignment_1.yar  
slashed@caltech ~] $ ./gradlew :PracticalSecurity:practical:release  
Dartify LICENSE README.md does prepare-release.sh projectSetAssignment_1.yar requirements.txt screens tools yarden.py  
slashed@caltech ~] $ ./gradlew :PracticalSecurity:practical:release  
Dartify LICENSE README.md does prepare-release.sh projectSetAssignment_1.yar requirements.txt screens tools yarden.py
```

6

Now, i will **YARA** tool using file has been generated by [yarGen.py](#)

```
slash@localhost:~/ProactiveSecurityPractical/yarGen$ la
.git .gitignore 3rdparty LICENSE README.md dbs prepare-release.sh proactSecAssignment_1.yar requirements.txt screens tools yarGen.py
slash@localhost:~/ProactiveSecurityPractical/yarGen$ yara proactSecAssignment_1.yar /home/slash/ProactiveSecurityPractical/malwares/steam_api.dll
steam_api_0xMekawy /home/slash/ProactiveSecurityPractical/malwares/steam_api.dll
slash@localhost:~/ProactiveSecurityPractical/yarGen$
```

*Credits : 0xM3k4wy.*