TryHackMe Advent of Cyber 2025

Day 13 Challenge Report

YARA Rules for Malware Detection

1. Executive Summary

This report documents the completion of Day 13 of the TryHackMe Advent of Cyber 2025 event. The challenge focused on YARA rules for malware detection and pattern matching. Successfully created custom YARA rules to detect malicious indicators, scanned directories for suspicious patterns, and decoded hidden messages embedded in image files. Demonstrated comprehensive understanding of YARA syntax, string types, conditions, and practical malware hunting techniques.

2. Challenge Overview

Objective: Learn YARA rule syntax, create custom detection rules, scan directories for malicious patterns, and extract hidden messages from image files.

Context: Defending TBFC against King Malhare's attacks using YARA pattern matching

3. YARA Overview

YARA is a tool built to identify and classify malware by searching for unique patterns - the digital fingerprints left behind by attackers. Think of it as a detective's notebook for cyber defenders: instead of dusting for prints, YARA scans code, files, and memory for subtle traces that reveal a threat's identity.

3.1 Why YARA Matters

YARA detects malware by behavior and patterns, not just by name. It allows defenders to define custom rules for 'malicious' behavior, enabling:

- Faster detection of threats
- Smarter threat hunting
- Fewer threats slipping by unseen
- Sharing and adapting rules across security teams

3.2 Use Cases

- **Post-incident analysis:** Verify if malware exists elsewhere
- **Threat hunting:** Search for known malware families
- **Intelligence-based scans:** Apply shared rules for new IOCs
- **Memory analysis:** Examine processes for malicious fragments

3.3 YARA Values

- **Speed:** Quickly scans large file sets
- **Flexibility:** Detects text strings, binary patterns, complex logic
- **Control:** Analysts define exactly what's malicious

- **Shareability:** Rules reused across teams
- **Visibility:** Connects scattered clues into attack picture

4. YARA Rule Structure

4.1 Core Components

1. **Metadata:** Rule information (author, description, date)
2. **Strings:** Patterns to search (text, hex, regex)
3. **Conditions:** Logic determining when rule triggers

4.2 Example Rule Structure

```
rule TBFC_KingMalhare_Trace {
meta:         author = "Defender of SOC-mas"
description = "Detects traces of King Malhare's malware"
date = "2025-10-10"
strings:
$s1 = "rundll32.exe" fullword ascii
$s2 = "msvcrt.dll" fullword wide
$url1 = /http:\/\/.*malhare.*/ nocase
condition:         any of them }
```

5. YARA String Types

5.1 Text Strings

Most common type representing words or text fragments. Default: ASCII, case-sensitive.

Text String Modifiers:

- **nocase:** Ignore letter casing

```
$xmas = "Christmas" nocase
```

- **wide, ascii:** Two-byte Unicode or single-byte

```
$xmas = "Christmas" wide ascii
```

- **xor:** Check all XOR variations

```
$hidden = "Malhare" xor
```

- **base64, base64wide:** Decode Base64 content

```
$b64 = "SOC-mas" base64
```

5.2 Hexadecimal Strings

Search for raw byte patterns in hex notation. Useful for file headers, shellcode, binary signatures.

```
$mz = { 4D 5A 90 00 }  // MZ header $hex_string = { E3 41 ?? C8 }  // ?? = wildcard byte
```

5.3 Regular Expression Strings

Flexible patterns matching multiple variations. Useful for URLs, encoded commands, dynamic filenames.

```
$url = /http:\/\/.*malhare.*/ nocase $cmd = /powershell.*-enc\s+[A-Za-z0-9+/=]+/ nocase
```

6. YARA Conditions

Conditions determine when rules trigger based on string matches and file properties.

6.1 Basic Conditions

- **Single string:** condition: $xmas
- **Any string:** condition: any of them
- **All strings:** condition: all of them

6.2 Logical Operators

```
condition: ($s1 or $s2) and not $benign
```

6.3 File Property Checks

```
condition: any of them and (filesize < 700KB)
```

7. Practical Exercise

7.1 Challenge Requirements

Search for keyword TBFC: followed by ASCII alphanumeric characters across /home/ubuntu/Downloads/easter directory to extract McSkidy's hidden message.

7.2 YARA Rule Creation

Created custom YARA rule to detect the TBFC: pattern:

```
rule TBFC_Simple_MZ_Detect {
meta:         author = "TBFC SOC L2"
description = "TBFC Data"
date = "2025-10-10"
confidence = "High"
strings:         $tbfc_msg = /TBFC:[A-Za-z0-9]+/ ascii
condition:         $tbfc_msg }
```

7.3 Execution

Executed YARA rule with recursive scan and string printing:

```
yara -rs /home/ubuntu/YARA_Simple_MZ_Detect /home/ubuntu
```

Command Flags:

- **-r:** Scan directories recursively
- **-s:** Print matching strings

8. Challenge Results

8.1 Question 1: Image Count

Question: How many images contain the string TBFC?

Answer: 5

8.2 Question 2: Regex Pattern

Question: What regex would you use to match a string that begins with TBFC: followed by one or more alphanumeric ASCII characters?

Answer: /TBFC:[A-Za-z0-9]+/

Regex Breakdown:

- **TBFC:** Literal match
- **[A-Za-z0-9]:** Character class (alphanumeric)
- **+:** One or more occurrences

8.3 Question 3: Hidden Message

Question: What is the message sent by McSkidy?

Answer: find me in hopsec island

Decoding Process:

- YARA scan identified 5 images with TBFC: patterns
- Each image contained numbered message fragments
- Aligned fragments in ascending/descending order
- Reconstructed complete message

Matched Patterns:

```
TBFC:HopSec TBFC:find TBFC:island TBFC:in TBFC:me
```

9. Key Skills Developed

- YARA rule syntax and structure
- Text, hexadecimal, and regex string patterns
- String modifiers (nocase, wide, xor, base64)
- Condition logic and operators
- File property checks (filesize)
- Regular expression pattern matching
- YARA command-line usage
- Malware pattern hunting techniques

10. Conclusion

Day 13 of the TryHackMe Advent of Cyber 2025 provided comprehensive training in YARA rules for malware detection. Successfully created custom rules using text, hexadecimal, and regular expression patterns to identify malicious indicators embedded in files.

The challenge demonstrated YARA's power as a flexible, fast, and precise tool for threat hunting and malware analysis. Through practical exercises, gained hands-on experience writing rules, scanning directories, and extracting hidden intelligence from suspicious files. YARA empowers defenders to move from passive monitoring to active hunting, defining custom detection logic and sharing intelligence across security teams.

Challenge Status: COMPLETED ✓