CMSC 449
Malware Analysis

Lecture 19
YARA Rules

Malware Families - Review

- **Malware Family** – a group of malicious files all derived from a common base of source code

- Malware authors are continually updating their malware
  - New functionality
  - Evading antivirus detection
  - Using new packer / obfuscation

- Important to track how malware families change

Malware Triage - Review

- Hundreds of thousands of unique, malicious files every day
- Most new malware is an updated version of an existing malware sample
- Need to prioritize which malware samples get human attention
  - Malware belonging to a family of interest
  - Malware which cannot be classified into a known family

Malware Signatures

- **Malware Signature** – pattern which can uniquely detect a specific family of malware
- Often based on a unique byte sequence, string, or other feature of the file
- Malware signatures for network traffic also exist

YARA Rules

- **YARA** is a tool for writing malware signatures. Can be used to scan files and running processes
- Often included within open-source malware reports
- Large collections of public YARA rules exist on Github, as well as other threat intelligence sharing sites

YARA Rule Example

```
rule ExampleRule {
    meta:
        description = “This is an example”
        author = “John Doe”
    strings:
        $s1 = “Malware string” // Comment
        $b1 = { 00 11 22 33 }
    condition:
        $s1 or $b1
}
```

Adapted from https://yara.readthedocs.io/en/stable/writingrules.html

YARA Meta Section

- Meta section defines metadata about the rule
  - Description
  - Author
  - List of file hashes used to make the rule

- Optional and does not affect the rule at all
  - But helpful when sharing the rule publicly

YARA Strings Section

- A bit of a misnomer, not just sequences of characters

- Three types of ‘strings’:
  - Character strings (enclosed in “ “)
  - Byte sequences (enclosed in { })
  - Regular expressions (enclosed in / /)

- Variables can have any names

YARA Character Strings

- Can apply modifiers to match different types of strings
- **`nocase`** modifier allows strings to be case-insensitive
- **`ascii`** and **`wide`** modifiers force certain character encodings
- **`fullword`** modifier requires that the string is delimited by non-alphanumeric characters

YARA Character String Example

```
rule StringExampleRule {
    strings:
        $s1 = "string1" nocase
        $s2 = "string2" ascii
        $s3 = "string3" wide
        $s4 = "string4" fullword
    condition:
        2 of them
}
```

Adapted from https://yara.readthedocs.io/en/stable/writingrules.html

YARA Byte Sequences

- Can search for sequences of raw bytes within a file
- Can have wildcards:
  - `{00 11 ?? 33 44}`
- Can have a varying number of wildcards:
  - `{00 11 [2 - 4] 33 44}`
- Can have alternatives:
  - `{00 11 (22 | 02 02) 33 44}`

YARA Character String Example

```
rule ByteExampleRule {
    strings:
        $b1 = {00 11 22 33 44 55 66}
        $b2 = {00 ?? 22 ??}
        $b3 = {00 [2-4] 11 22}
        $b4 = {00 (11 | 22) 33 44}
    condition:
        $b1 or ($b2 and $b3) or $b4
}
```

Adapted from https://yara.readthedocs.io/en/stable/writingrules.html

YARA Regular Expressions

- **Regular Expressions (Regex)** – Syntax for matching patterns

- Regex for matching alphanumeric strings:
  - `/[A-Za-z0-9]+/`

- Regex for matching an IP address:
  - `/[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}/`

YARA Character String Example

```
rule RegexExampleRule {
    strings:
        $r1 = /google.+\.com/
        $r2 = /192\.168\.[0-9]{1,3}\.[0-9]{1,3}/
    condition:
        all of them
}
```

Adapted from https://yara.readthedocs.io/en/stable/writingrules.html

YARA Condition Section

- Can set conditions about strings and metadata
- Condition determines how strict a YARA rule is
- “Loosen” a rule by requiring fewer string matches
  - But this may increase false positive detections!

YARA Condition Examples

- Other keywords
  - `all of them` / `any of them` / *`n`* `of them`
- Two strings and one byte sequence
  - `2 of ($s*) and 1 of ($s*)`
- Number of occurrences
  - `#s1 == 2 and #s2 > 10`

Location Conditions

- Can specify where a string appears in a file
- Byte sequence beginning 100 bytes into the file
  - `$b1 at 100`
- Byte sequence between 100-200 bytes into the file
  - `$b1 in (100…200)`
- Byte sequence at the entry point
  - `$b1 at entrypoint`

Checking for PE Files

```
rule IsPE {
    strings:
        $b1 = {4D 5A} // MZ
    condition:
        // MZ signature at offset 0
        $b1 at 0
}
```

Adapted from https://yara.readthedocs.io/en/stable/writingrules.html

File Size Keyword

```
rule LargeFile {
    condition:
        filesize > 1000KB
}
```

YARA Modules

- PE and ELF modules for conditions based on Windows / Linux executable file formats
- Math module for common math operations, including entropy
- Hash module for hashing data in files

YARA Rule Demo

Developing YARA Rules Demo