Lab 07 YARA

In this lab, you will create YARA rules to identify files that contain certain properties. Some of the rules you create must be done manually without the use of existing modules; this it to help you create tools for the malware analysis community to use.

Lab Files

The malware samples are contained within password-protected zip archives with the password: **infected**

Please ensure you are handling the malicious files appropriately. File hashes of the lab samples are:

- MD5 (sample01.bin) = 8a2122e8162dbef04694b9c3e0b6cdee
- MD5 (sample02.bin) = cac3b2053ede00698cca44f54a5b5e44
- MD5 (sample03.bin) = 4d3943edbc9c7e18dc3469a21b30b3ce
- MD5 (sample04.bin) = 2ab31ece6230baf6d919e82a5f6a15c6
- MD5 (sample05.bin) = 889b99c52a60dd49227c5e485a016679
- MD5 (sample06.bin) = cb601b41d4c8074be8a84aed564a94dc
- MD5 (sample07.bin) = 9a29876347a6b7a9fbd942c7181b883d
- MD5 (sample08.bin) = bdcbf1de676cdf9b87a1de1e20607330
- MD5 (sample09.bin) = 05dedf1936a065612e52c37e40143646
- MD5 (sample10.bin) = 8a0ef1eeb6bbaa7461f00ff8858a32a3
- MD5 (sample11.bin) = 2af1b2c042b83437a4be82b19749fa98
- MD5 (sample12.bin) = 3a3b9a5e00ef6a3f83bf300e2b6b67bb
- MD5 (sample13.bin) = 7900f284e2839378ca5fb862d1bf3960
- MD5 (sample14.bin) = 1022e62b18bfa25d88754918528c3d04
- MD5 (sample15.bin) = 961e093be1f666fd38602ad90a5f480f

Lab Instructions

Complete the following tasks in your analysis environment or using the IA lab. All work should be original, discovered, and reported on by you. Do not rely upon a single source of information (i.e., an online sandbox) to answer all questions.

1. Every rule file you write needs to include the following:
    a. Your name in each rule (e.g. **mike_pe_rule** or **mike_x32_rule**)
    b. Comments to describe what the rule is looking for or how it works
    c. A **meta** section at the top with **author** and **description**.

PE File Analysis

Write all of the rules for this section in a single **.yara** file called **fname_pe.yara**. These rules must be written by you; using pre-made modules/libraries is not allowed.

1. Rule 1 - determine: if the file is a valid PE file (.exe or .dll)
    a. MZ header at the beginning of the file (0x4D5A)
    b. Find a valid PE signature at the offset 0x3c (0x50450000)

2. Rule 2 – determine if the PE file is an x32 or x64-bit binary
    a. Use the rule you wrote in the previous step to as a condition (i.e., you only need to find the x32/x64 IF the file was a valid PE.

       Hint: https://yara.readthedocs.io/en/stable/writingrules.html#referencing-other-rules

3. Rule 3 – identify the file as a DLL or EXE file.
    a. Again, reference Rule 1 as a condition. If the file is not a valid PE, there is no need to see if it is a DLL or EXE.

Strings Analysis

YARA is capable of finding strings that are encoded or obfuscated in certain ways (e.g. Base64 or XOR). Write all of the rules for this section in a single **.yara** file called **fname_strings.yara**. These rules must be written by you; using pre-made modules/libraries is not allowed.

1. Rule 1 – determine whether or not the file is a .txt file; those are the only ones we care about for this section.

2. Rule 2 – search for Base64 encoded text that, when decoded will read **I <3 malware analysis, it's my favorite class!!!**
    a. This rule should reference the previous rule; only analyze .txt files.

3. Rule 3 - search for XOR encoded text that, when decoded will read **Where is the any key?**
    a. This rule should reference the previous rule; only analyze .txt files.

Modules

YARA modules can be particularly useful for complex tasks so you do not need to reinvent the wheel. Write all of the rules for this section in a single **.yara** file called **fname_modules.yara**..

1. Import the **PE** module to help write the following rules:

2. Rule 1 – find all of the PE files that are executables (not DLLs) and have a timestamp from within the last year.

   Hint: you will need to use the **pe** and **time** modules.