LINGI2144: Secured System Engineering
Tutorial 8: YARA

Academic year : 2020 - 2021

Teacher : Legay Axel
Course : LINGI2144
Collaborators :
Crochet Christophe
Duchene Fabien
Given-Wilson Thomas
Strebelle Sebastien

1 Prerequisite

Working directory: `~/SecurityClass/Tutorial-08`

Connection:

username	password
admin	nimda

NOTE: To run the tutorial today you will need to install YARA, this can be done with

```
sudo apt-get install yara
```

Also note that due to lots of files and sharing between parts of the tutorial, the tutorial files have not been split up according to section.

2 Exercise

2.1 YARA basic

YARA is a system to identify files according to rules. In `browsers.yar` you will find an example rule to detect three different web browsers. An example of how to use this rule with YARA is

```
yara browsers.yara files
```

that will use this rule on all the files in the "files" directory.

We can also see that YARA only reports files that match the YARA rule(s) by default:

```
yara browsers.yara bin
ls bin
```

According to our `browsers.yar` YARA rule none of the files in the `"bin"` directory are browsers. YARA has many advanced options we can use that we can see with[1]

```
yara --help
```

for example we could check that YARA has tried all of the rules and found they fail to match with

```
yara -n browsers.yara bin
```

which will show that each file is checked with each of the browser rules.

2.2 Simple Malware Detection

In the `"bin"` directory you will find five files that are clearly named by what they are meant to be:

```
ls bin
```

four are meant to be malware, and four are cleanware.

You can execute each of them to observe their behaviour and observe what malicious behaviour looks like.

Write a YARA rule that can detect `malware_1` but not `cleanware_1` (detecting other malware is fine but not required, and falsely detecting other cleanware is bad but we don't care for now).

NOTE: A really simple (and broken) example of a YARA rule is left for you to start with: `malware.yar`

[1] `https://yara.readthedocs.io/en/stable/`

2.3 Improving Detection

Now your challenge is to improve your YARA rule to detect all the malware but none of the cleanware. You should do this by exploring how YARA rules work and experimenting with what you can do in the rules.

One helpful command when looking for things to use in YARA rules is the "strings" command that lists all the strings in a file. For example:

```
strings bin/malware_1
```

will show you all the strings in the `malware_1` file . You can then also look at the strings from other files to see whether there are strings that are common in malware, or cleanware.

NOTE: Using the filename (e.g. testing if the filename contains "clean" or "malware") is not an acceptable solution. The files are named this way so you can be sure, not as a way for you to solve the tutorial.

HINTS: YARA allows you a lot of options for classifying files, don't just look for strings.

CHEATING: The source code for all the malware and cleanware is provided in the " directory. This is done so that you can build the files yourself if necessary, and so you can see how some "malware" is written if you become stuck. Refer to these if you are having big problems, but try not to use them immediately.

2.4 Defeat Your Friends

In the real world there is an ongoing attack and defence struggle between malware creators and security experts to detect malware. For the rest of this tutorial you will play this game. Pick a friend (or make two teams). One team will be the malware writers, and the other team will be the YARA rule writers.

Malware writers, your goal is to write a new piece of malware or cleanware that the YARA rules from the other team label incorrectly. This can be malware they don't detect, or cleanware that does not exhibit malicious behaviour.

YARA rule writers, your goal is to improve your rules each time the other team creates a new program.

NOTE: Both teams can play both roles. Both teams can write programs to defeat the other team's rules, and both teams can improve their rules to label more programs accurately. You can even use the knowledge from your own programs to improve your rules (or knowledge from your rules to improve your programs).

Remember when writing new rules to keep all the old programs in your test set - you don't want to let old malware through by not detecting it any more!