NUS CS-3235: Computer Security November 8, 2020

Assignment 3 Report

Lecturer: Reza Shokri *Student: Wang Xinman A0180257E*

1 Motivation

As no information was provided on what metrics to use for profiling, I first tried to combine all the metadata provided in the trace files: timestamp, number of incoming connections, number of outgoing connections, size of incoming files, and size of outgoing files. However, this did not work out, and when tested against the training data (profiles under `traces/`) produced results of low-accuracy.

As such, I decided to break it down and just focus on file size, since it is likely to be the most prominent trait of a site, as file size is indicative of the type of content transmitted, and that is likely to vary widely from site to site.

2 Methodology

I first collated all the distinct file sizes for every site, across 8 profiles. This is stored in a Python dictionary, in the format

```
dictionary: { url_id: "incoming": { set of all file sizes }, "outgoing": { set of all file sizes } }
```

General Approach The approach is to match observations with profiles by number of unique file size matches. For example, given: the set of file size for URL 1 is {1, 2, 3}, observation 1 has 3 occurrences of file size 1, and observation 2 has 1 occurrence of file size 1, 2, 3. Observation 1 only matches with 1 file size, so it has a score of 1 with respect to URL 1. Observation 2 has matches with 3 file sizes, so it has a score

of 3 with respect to URL 1. In this case, URL 1 is a better match for observation 2 than observation 1.

Assumption This approach is rooted in the assumption that each site profile is likely to differ greatly from one another in terms of file sizes, so that for a observed traffic to match 2 site profiles is unlikely.

3 Implementation

Profiling Each site traffic is parsed and file sizes extracted. This is collated across all 8 profiles, and the final dictionary is dumped into the file `dictionary.txt`, which is loaded into the matching program later.

Matching For each observation, the traffic files are parsed and file sizes extracted. Similar to profiling, a set of distinct file sizes for each anonymous site is collected and put into another dictionary. Each anonymous site data is then compared against the profiled sites, and scored as aforementioned. The URL with the highest number of file size matches with the observed traffic is noted as the best match, and output is collected across all 35 anonymous sites. The result is then sent to the file `result.txt`.

test.sh This file simply calls the matching program with the 2 arguments supplied.