NUS CS-3235: Computer Security October 28, 2021

Assignment 3

Lecturer: Reza Shokri *Student: Name Number*

Instructions

- All submissions should be uploaded in GitHub repo. The report should be in PDF. You are strongly encouraged to use LaTeX (use the provided template).
- We provide the submission template repository. Your submission should be in the same folder structure, except for adding a folder A01xxxx containing your code folder and your `test.sh`, replacing A01xxxx by your student number.
- Your file name should be your student number (e.g., A01xxx.zip)
- Use font size 12.
- We will ignore all the submissions that violate any of the above.
- You have 2 weeks to submit your solutions.
- Report page limit: ≤ 2
- **You are not allowed to share the content of the assignment with others (e.g., release them on the Internet).**
- **No collaboration** is permitted on this assignment. Any cheating (e.g., copying from any source, or permitting your work to be copied) will result in a failing grade. The assignments are provided to help you learning by solving problems. What's the point of obtaining the solutions from elsewhere? Your grade won't help you securing your code.

Traffic Analysis

In part 1 of the course, we introduced how nested encryption and intermediate proxy can be combined to realize anonymous routing. This encryption and mixing of channels, however, are not enough to secure the anonymity of the long term network traffics. More specifically, an attacker who observes the messages sent and received in a network can deduce patterns from the network traffics. The patterns then serve as "fingerprints" for certain messages. The more messages the attacker can observe, intercept, or modify, the stronger are the patterns that the attacker can infer from network traffics. For example, from the encrypted streamed video contents, the attackers can guess what video a user is watching [1].

The objective of this assignment is to show the vulnerability of the HTTPS traffic to traffic analysis attacks. Prior to the attack, in the profiling phase, the attacker connects to a set of URLs and captures the communicated packets between his machine and the server hosting the URL. The attacker performs the profiling attack multiple times, in order to obtain many samples of the HTTPS traffic to target webpages, for generating a fingerprint for each webpage. In the attack phase, where the attacker observes the connection of the victim to an anonymous webpage (through a proxy server, or anonymity network such at Tor), he tries to match his observation with the fingerprints.

Profiling phase

You receive the processed captured packets of 35 URLs, to build the fingerprints. The packet traces for each URL are saved in a file, with the same and persistent file ID across the 8 measurements. The packet trace files contain the time of arrival for each packet, its size in Bytes, and direction (in: towards the client, out: towards the server). You can obtain the profiles in traces.zip. Figure 1 is an example of a trace file.

```
00:00:00.000000 0 in
00:00:00.000056 0 out
00:00:00.000897 201 out
00:00:00.134478 0 in
00:00:00.134523 0 out
00:00:00.135007 201 out
00:00:00.243595 0 in
00:00:00.262097 1448 in
00:00:00.262118 0 out
...
```

Figure 1: *An example of a trace file.*

Attack phase

You will receive the captured packets, corresponding to the attacker's observation of anonymous webpages. The set of anonymous URLs is the same as the profiled URLs, but of course, their identities are randomly permuted. This is known as the closed-world setting. Your algorithm will be tested by the observations of two different attackers `observation1` and `observation2`. The observations from one attacker are in the same folder which contains 35 files, as shown in Figure 3.

The expected solution of your algorithm is 1 file for 2 columns (one for each attack observation), with 35 lines (for each URL in the profiles, where the line number represents the URL id) and 1 column (for this attacker observation), separated with a space character. In each column, please provide the anonymous id of a URL corresponding to its id in the profiles. Please do NOT put any extra character/line, as your solution file will be corrected automatically. Figure 3 shows an example of a solution, where the first line shows the URL with anonymous id 2 in `observation1` and anonymous id 34 in `observation2` are associated with the profiles with id 1:

```
observation1/
             1-anon
             2-anon
             3-anon
             4-anon
             ...
             35-anon
```

Figure 2: *An example of attacker's observation folder.*

```
2 34
7 1
12 8
9 11
30 2
15 28
...
```

Figure 3: *An example of a solution file.*

Code requirement

Please provide a `test.sh` bash script which takes the attacker's observations folders (`observation1, observation2`) as the input, and produces the result file. We will run your `test.sh` by command:

```
$ ./test observation1 observation2
```

Your bash script should automatically produce `result.txt` as figure 3. The assignment is considered as successfully submitted only if the `test.sh` can produce the result file with the correct format.

You can only use Python and the libraries we provided (See details in the virtual

machine on MS Teams: File → Assignment → Assignment-3.ova). If you need other libraries, please contact TAs to add the library and update the virtual machine image accordingly.

NOTE: Your program will be tested on the virtual machine we provide.
All code submissions that fail to run on this virtual machine will not be graded.

Submission

Please submit your code files and bash script along with 2 page summary of your attack algorithm (A01xxxx.pdf).

1. Please copy the folder A0XXXX of your VM into this git repository, replacing A01xxxx by your student number. The folder A0XXXX contains your code folder and your `test.sh`.

2. Please write your attack summary in the latex template `./solution.tex` file, and also submit your compiled pdf file.

Please do not change the folder structure in the repository that we provided.

References

[1] Roei Schuster, Vitaly Shmatikov, and Eran Tromer. Beauty and the burst: Remote identification of encrypted video streams. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*, pages 1357–1374, 2017.