BTU Cottbus, Chair of IT Security
Prof. Dr.-Ing. A. Panchenko
T. Ziemann, PhD
E. Strehle, B.Sc.

Winter term 2019/2020

Part I: Laboratory setup / Browser automation Deadline: 23.10.2018

Study Project: Adversarial Machine Learning

Website Fingerprinting: Proxy

Introduction

Website fingerprinting (*WFP*) is a special type of traffic analysis attack where an adversary attempts to identify which page a user is visiting by analyzing patterns of encrypted communication. The attacker is located between the user and the privacy enhancing technique (e.g., proxy, VPN, Tor, JAP). The adversary is not able to decrypt the observed traffic. Thus, the adversary can only see the meta-data (e.g., packet size, direction) of the communication. WFP relies on the classification of TCP/IP traces to a given set of websites. The key idea is to train a machine learning model with traces from a known website. After the training phase the model should be able to decide if a known website has been visited by analyzing a given trace. To train these models, there is a need for many training examples. To efficiently execute this task, it is required to automate the whole process. In this part of the project, you have to inform yourself about WFP and automation techniques for web browsers. Additionally you have to setup your own laboratory environment.

Preliminaries

At first you have to setup your own laboratory environment. You shall capture encrypted data which is sent to a proxy. Therefore you have to create at least one virtual machine (*VM*) which is configured as proxy. Your host machine or an additional VM have to be connected to this proxy.

a) Ensure, that the traffic to the proxy is encrypted.

b) Consider possible side effects (e.g., updates, background traffic) which may could influence the captured traffic. Name them and think about potential solutions to limit them.

c) How could web browsers cache influence the measurements of captured traffic? Can this be neglected?

d) Think about how the interaction with the browser can be automated. Which kind of automation is needed? Are there any existing extensions or other tools that can be useful for this task?

e) What kind of meta-data should be stored during the automated crawling (error analysis, meta-data needed in the context of WFP and machine learning)?

Main Task: Browser automation

Later, you have to collect data of at least 100 web pages. Manually typing and calling these web pages would be an annoying task. To be more scientific you have to automate a web browser of your choice. Prepare a piece of code which satisfies the following requirements:

a) Read URLs one by one and visit them automatically via the browser.

b) Fetch successively each of the URLs without need for human interactions.

c) Limit possible side effects which may influence the crawling.

d) Log your automated actions (timestamps, remote IP, server or client side errors etc.).

In addition, for each page load you need to collect the following information:

a) Begin and end of the page loading.

b) Endpoints of the communication by means of local and remote IP addresses.

c) Possible errors occurring if a page was not loaded correctly (error log).

Test your source code with the following websites:

- `https://www.google.de/`
- `https://www.b-tu.de/en/`
- `https://www.heise.de/`

Preparation of consultation

Prepare yourself for a short consultation of about 15 minutes. You shall give a short overview of libraries, scripts or already existing browser extensions you have used. Furthermore present and explain your laboratory setup and your own piece of code. Give a proof of work through a demonstration of your program/scripts.

Good luck!