KHALEESI: Breaker of Advertising and Tracking Request Chains

Umar Iqbal, *University of Washington;* Charlie Wolfe, *University of Iowa;* Charles Nguyen, *University of California, Davis;* Steven Englehardt, *DuckDuckGo;* Zubair Shafiq, *University of California, Davis*

https://www.usenix.org/conference/usenixsecurity22/presentation/iqbal

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 31st USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 31st USENIX Security Symposium.

August 10–12, 2022 • Boston, MA, USA

978-1-939133-31-1

A Artifact Appendix

A.1 Abstract

We propose KHALEESI, a machine learning (ML) approach that captures the essential sequential context needed to effectively detect advertising and tracking request chains. We release KHALEESI's classification code, ML model, browser extension, and data sets. Classification code is written in Python 3.6, the ML model is trained using Scikit, the browser extension is written in JavaScript/HTML, and the data is crawled using OpenWPM.

A.2 Artifact check-list (meta-information)

- **Binary:** A browser extension to block advertising and tracking request chains. The extension is designed and tested in Mozilla Firefox.
- **Model:** ML model to detect advertising and tracking request chains. Released ML model was trained on request chains from homepages of Alexa top-10K websites.
- **Data set:** Data sets to train and test ML model. We release crawls of homepages, home and sub pages, home and sub pages with cookies blocked, and home and sub pages with browser spoofed as Safari. All data sets are crawls of Alexa top-10K websites. The data contains requests, responses, and JS execution.
- **Run-time environment:** Scripts can be run using Python 3.6 and above. The code was tested on Ubuntu 16.04.7 LTS.
- **How much disk space required (approximately)?:** We recommend a disk space of ∼100GB to train the classifier. The browser extension does not have any disk space constraints.
- **How much time is needed to complete experiments (approximately)?:** The classifier can be trained in ∼ 10 hours. The browser extension blocks the ads instantaneously.
- **Publicly available (explicitly provide evolving version reference)?:** KHALEESI's code, data, and browser extension is available at https://uiowa-irl.github.io/Khaleesi/.
- **Archived (explicitly provide DOI or stable reference)?:** KHALEESI's code, data, and browser extension is available at https://github.com/uiowa-irl/Khaleesi/tree/bd28513878a363b39b0ee9e7a6a4350f71672912

A.3 Description

A.3.1 How to access

KHALEESI's code, ML model, and browser extension are available on Github at: https://uiowa-irl.github.io/Khaleesi/. Data sets are available on Zenodo at: https://doi.org/10.5281/zenodo.6084582.

A.3.2 Hardware dependencies

KHALEESI ML model was trained on a machine with 16 cores and 96 GB RAM. We recommend a disk space of ∼100 GB to train the classifier. The model can be tested on hardware with less resources.

A.3.3 Software dependencies

KHALEESI browser extension was designed and tested on Mozilla Firefox. We trained and tested KHALEESI ML model on Ubuntu 16.04.7 LTS.

A.3.4 Data set dependencies

KHALEESI is trained on data set crawled through OpenWPM version 0.10.0. The code might require some minor modifications to process data from newer versions of OpenWPM.

A.4 Installation

We provided instructions to run KHALEESI on Github.

A.5 Experiment workflow

In addition to instructions on Github, we provide detailed instructions to run the code below:

A.5.1 Training & Testing ML model

We list the step-by-step process to train and test KHALEESI's ML model below:

1. *Data collection:* Collect network and JavaScript initiated requests using OpenWPM.
2. *Request chain construction:* Organize network and JavaScript initiated requests into chains. Request chains can be constructed with HTTP and JavaScript chain construction scripts.
3. *Request chain labeling:* Once constructed, label request chains using EasyList (EL) and EasyPrivacy (EP) filter lists. Use filter list labeling script and EL/EP filter lists to label the chains.
4. *Feature extraction and transformation:* After labeling, extract features from the request chains using using feature extraction script and encode them using feature encoding script.
5. *Model training:* Since, KHALEESI relies on previous confidence as a feature, extract the previous confidence for each request in a chain before training the final model. The previous confidence can be extracted using compute previous confidence script. The last block of previous confidence script stores the final trained model. An already trained model is available in data directory.
6. *Testing the model:* KHALEESI uses 10-fold cross validation to test the data sets. The encoded features with previous confidence can be tested using test classifier script and the accuracy can be computed using compute accuracy script.

A.5.2 Analysis of Request Chains

We release scripts to analyze cookie syncing and bounce tracking instances in request chains. Use the cookie syncing and bounce tracking scripts to identify cookie syncing and bounce tracking instances, respectively.

A.5.3 Browser Extension

To add KHALEESI to Firefox, enter `about:debugging` in the URL bar, click *This Firefox*, click *Load Temporary Add-on*, navigate to the extension's directory and open `manifest.json`. To view the requests blocked by KHALEESI, open extension's console by clicking *Inspect* in `about:debugging` or see the network tab in the Firefox Developer Tools.

A.6 Evaluation and expected results

Training & Testing ML Model: Upon successful execution, the workflow should produce a trained ML model and output its accuracy.

Analysis of Request Chains: Upon successful execution, the scripts should list the cookie syncing and bounce tracking instances in request chains.

Browser Extension: After installation, the browser extension should block advertising and tracking request chains.

A.7 Version

Based on the LaTeX template for Artifact Evaluation V20220119.