A Artifact Appendix

A.1 Abstract

In this work, we developed a voting-based domain ranking method that operates on passive DNS (PDNS) data to construct a domain top list. We open-source our top list construction implementation at *https://github.com/SecRank/secrank-sourcecode/releases/tag/v1.0.0*, to provide transparency into the design of our ranking method. The code provided (written in Scala) processes proprietary PDNS data to compute a daily top 1M domains list, running in a distributed fashion using Apache Spark on YARN. As we are unable to release the raw proprietary PDNS data used, the code is not directly runnable. Instead, it serves as a reference for understanding the details of our ranking method, and as a template that can be modified for other PDNS datasets and computing environments.

As our top list design achieves favorable stability and manipulation resistance properties, we also provide public access to a regularly updated domain top list constructed using our ranking method at *https://secrank.cn/topdomain*, for others to use in their research. After registering for an account on the website, a user can download a daily top 1M domains list as well as historical top lists.

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

- **Algorithm:** We provide a new voting-based domain ranking algorithm that operates on PDNS data, where the domain preferences of individual IP addresses are first computed, and then a global top list ranking is produced by applying a voting scheme across all IP addresses.
- **Security, privacy, and ethical concerns:** This artifact does not raise any security, privacy, or ethical concerns.
- **Publicly available?:** Our top list construction implementation is publicly available at https://github.com/SecRank/secrank-sourcecode/releases/tag/v1.0.0. A regularly updated domain top list constructed using our ranking method is publicly available at *https://secrank.cn/topdomain*, upon registering for a user account.
- **Code licenses?:** MIT License.
- **Archived?:** Our top list construction implementation is publicly archived at https://github.com/SecRank/secrank-sourcecode/releases/tag/v1.0.0.

A.3 Description

A.3.1 How to access

Source Code Access. We open-source our top list construction implementation at *https://github.com/SecRank/secrank-sourcecode/releases/tag/v1.0.0*. Our implementation relies on proprietary PDNS data, which we are unable to release for privacy and commercial reasons. Thus, the code is not directly runnable, and rather provides transparency into the design of our domain ranking method.

Those interested may adapt our code for their own PDNS data/format and computing environment. Note that our implementation uses Apache Spark on YARN, with input and output data stored in HDFS. For users with a similar computing environment, our code can be most directly applied by providing the proper input and output data file paths, as well as adjusting the data field names extracted from the input data. Further details are provided in the *README.MD* file.

The code repository consists of the following main files:

- *README.MD*: The README file providing guidance on using the code.
- *TopFQDNDailyRelease.scala*: The main algorithm source code file, containing detailed comments for each algorithm component that reference the relevant sections in our paper describing the algorithm's design. We additionally document the input/output file paths and data formats that must be modified if adapting this code for other PDNS datasets.
- *pom.xml*: The XML file that contains information about the software package and configuration details (including software and library dependencies).
- *submit.sh*: The shell script to submit the Spark application to a YARN cluster.

Daily Top 1M Domains List Access. Public access to a regularly updated top 1M domains list is available at https://secrank.cn/topdomain, through registering for a free account. With an activated account, users can download the latest daily list as well as historical lists.

A.3.2 Hardware dependencies

N/A

A.3.3 Software dependencies

Our released implementation is written in Scala and runs on Apache Spark on YARN. While our released code is not directly runnable, those modifying it for use will likely require IntelliJ IDEA, Java 1.8, Maven JDK 1.8, Scala 2.11.8, Apache Spark 2.4.5, and Hadoop 2.7.2. (These dependencies are also shown in the *pom.xml* file in the release package.)

A.3.4 Data sets

The source code relies on proprietary PDNS data, which we are unable to release for privacy and commercial reasons.

A.3.5 Models

N/A

A.3.6 Security, privacy, and ethical concerns

N/A

A.4 Installation

While our released code is not directly runnable (as we cannot release our raw input PDNS data), we provide guidance on how one could modify the code to run on their own input PDNS dataset. As our implementation is executed via Apache Spark on YARN, we assume a similar computing environment (i.e., Java 1.8, Maven JDK 1.8, Scala 2.11.8, Apache Spark 2.4.5, and Hadoop 2.7.2).

1. We suggest using IntelliJ IDEA to create an Apache Maven project, and replacing the default *pom.xml* file with the *pom.xml* file in our Github repository (which contains all dependency configurations and package requirements).
2. Next, place *TopFQDNDailyRelease.scala* in the path *$PROJECT_PATH$/src/main/java/com/secrank/examples/*.
3. In *TopFQDNDailyRelease.scala*, modify the *trends_path* and *access_path* variables to reference the input PDNS data file paths on HDFS, and also modify accordingly the output file path (in the code's final stage).
4. As documented in the comments of *TopFQDNDailyRelease.scala*, the code assumes certain fields are present in the input data format. If those fields are not available, either the source data must be modified to provide these fields, or the field names must be adjusted accordingly in the code to reflect the source data.
5. After modifying the code, package the Maven project into a JAR file, upload this JAR file to your Spark client, and execute *submit.sh* to submit the Spark application to the YARN cluster. After the code fully executes, the output will contain the top 1M domains list.

A.5 Experiment workflow

N/A

A.6 Evaluation and expected results

Users are expected to modify the code for their own input PDNS data and computing environments, with the expected output being a top 1M domains list computed using our domain ranking method.

A.7 Experiment customization

N/A

A.8 Notes

N/A

A.9 Version

Based on the LaTeX template for Artifact Evaluation V20220119.