This material is based upon work supported by the Defense
Advanced Research Projects Agency (DARPA) under Contract No.
HR001119C0102. The opinions, findings, and conclusions stated
herein are those of the authors and do not necessarily reflect
those of DARPA.

pcap-ingestion

Introduction

The script in this directory processes CSV files created by `zeek2csv` in a manner analogous to the way that CSV files created the pcap ingestion process are processed.

The central script in this directory is `process.sh`, which takes a parameter file as its single argument. (See `../../params` for information about the parameters, and `process.sh` for information about how those parameters are interpreted.)

The basic CSV ingestion pipeline has the steps described below. Note that some of these steps are optional, and may not be necessary for a given telescope or application

1. *Create files for missing hours*

   If there was a failure during the capture (or any of CSV files are missing for any other reason), then it is possible that some hourly CSV files are missing. This can be awkward, because for some analyses it is convenient to assume that every hour has a corresponding CSV file. For example, if you want to analyze all of the data for the 24 hours following a given hour H, it's much easier to look at "the file for H and the next 23 hours" rather than to do date arithmetic to figure out what the correct month, day, and hour for each of those 23 hours.

   The solution is to search for dates that don't have matching files, and create empty CSV files for those dates. This is done by `fill-missing.sh`.

2. *Find the destination /24 subnets present in the data for each hour*

   Our telescope has changed over time, so that some subnets that were in the telescope are no longer present, and new subnets have been added. Therefore it's not always known *a priori* which subnets are active at a given time.

   Some analyses iterate over all of the destination subnets (usually /24 subnets), so it is useful to know what subnets are present in each hour. The `find-subnets.sh` script scans the CSV file for each new hour and discovers which destination subnets are present.

3. (optional) *Compute some summary statistics*

   The `trend-by-subnet.sh` script computes some summary statistics for the new CSV files, using the `firecracker` utility. This script also computes these summary statistics for packets sourced from any members of the set of acknowledged scanners and its complement (all of the sources that are *not* known to be acknowledged scanners).

   This gives a quick view of any large-scale emerging trends in the traffic (such as a large change in the frequency with which a specific destination port is scanned).