The IBR tools

This directory contains subdirectories for several IBR-related tasks. This document describes the general workflow for using the tools, and each individual tool.

Workflow

Before starting to process any pcap files or doing any other analyses, you must create and/or edit the parameter files that define how the your telescope and working data is arranged on your local system. See the example files in '../params for instructions and examples.

For new pcap files

The general workflow for adding new pcap files has the following steps:

1. **Move the new gzip'd pcap files into your `$PCAPRAWDIR` directory**

   If necessary, begin with capturing and gzip'ing the pcap files. The tools and instructions in the `./capture` directory describe how to capture pcap files directly from an Ethernet device, but these instructions *do not* include the step of gzip'ing the pcap files, because this should be done as a separate step in order to minimize the CPU overhead on the capture host.

   If the pcap files are acquired in some other way (using a different tool, or from the third party, then, if necessary, rename them to match the naming convention assumed by the rest of the tools. (See `./capture/README.md` for a specification of the naming convention.) If they are not gzip'd, or are already compressed using some other algorithm, then gzip them.

2. **Use `./pcap-ingestion/process.sh` to ingest the new pcap files**

   The `./pcap-ingestion/process.sh` script is designed to take its parameters from a parameter file (although it can also take parameters from its environment, if appropriate). See `../params` for example parameter files and documentation about each parameter.

   `process.sh` can, optionally, pre-filter the pcap files to remove any packets that you want to omit from the analysis. See the example parameters for more info, and `pcap-ingestion/prefilter-pcap.sh` for an example filter.

   `process.sh` creates CSV files where each line describes a packet from the corresponding pcap file. Optionally, these CSV files are processed to make sure that each CSV file begins and ends on an hourly boundary (because the pcap files, depending on how they are captured, might not align perfectly on hourly boundaries).

   Next, all of the new CSV files are scanned to find the complete set of destination subnets observed for each hour. If your telescope is static, then this step is not necessary, because you should see the same subnets every hour – but if you have a dynamic telescope, then this is useful for dealing with changes. It can also alert you to errors in the network, if it detects that some destinations are not being correctly routed to your telescope.

   Finally, a quick analysis is done to find the most common protocol/destination port pairs, sources, and destinations in the new CSV files. This is done for all the sources, and also for the subsets of acknowledged scanners (known scanners, which account for a large fraction of periodic scanning activity) and for unknown scanners or other sources, split by destination /24 subnet. This gives you a quick view summary of what is being scanned within your telescope.

For new CSV files created by `zeek2csv`

The general workflow for adding new CSV files created by `zeek2csv` is much simpler than adding new pcap files, because we assume that Zeek is configured to capture the data in such a way that the CSV files created by zeek2csv are correctly filtered and time-aligned on hourly boundaries. (See `./zeek-scripts/` for more information.) If this is not true, then you can use the same mechanisms as `./pcap-ingestion/process.sh` to prepare them. Otherwise, just run `./zeek-ingestion/process.sh`, with your local parameter file, to find the subnets, fill any holes, and compute the summary analysis, just like `./pcap-ingestion/process.sh`.

1. **Move the output files from zeek2csv to your `$FCSVDIR` directory**
2. **Use `./zeek-ingestion/process.sh` to ingest the new CSV files**

   Like `./pcap-ingestion.sh`, this script will find all of the destination subnets, and compute the same summary analysis over each destination /24 subnet.

After the new files have been ingested

Use the tools described in the rest of this file on the newly-created CSV and other output files.

Note that some of the tools also require the pcap files be present, so they will not work as-is on CSV files created by `zeek2csv`.

capture

The `./capture` directory contains scripts and instructions for gathering IBR (or network packets in general), using a Linux host.

See `./capture/README.md` for more information.

zeek-scripts

If you don't have a way to capture your own network packets, but your organization is using Zeek to monitor its network, the `./zeek-scripts` directory contains scripts that might help. These scripts extract information about IBR from Zeek connection logs, and stores this information in a form that many of the other tools can use.

The information that Zeek provides is not as detailed as that provided by reading packets directly from the wire, but it can still be very useful.

See `./zeek-scripts/README.md` for more information.

pcap-ingestion

Once you have pcap files (collection with the scripts in `./capture`, or obtained in some other way, the tools in `./pcap-ingestion` take the next steps – filtering out unwanted packets, extracting information from the remaining packets and converting them into a convenient CSV format, and doing some preliminary analyses.

See `./pcap-ingestion/README.md` for more information.

horiz-scans

Once you have CSV files describing each IBR packet, a next-step is to search for groups of packets that are part of a scan for instances of a particular service within a subnet. The tools in `./horiz-scans` find these scans and extract information about them for subsequent analysis.

See `./horiz-scans/README.md` for more information.

ttl-scanner

One interesting phenomena observed in horizontal scans is that sometimes different packets in the same scan arrive with different TTLs. This can happen naturally when different packets take different paths, and those paths have different lengths, but in some cases the TTLs vary in unexpected ways.

In some cases, the TTLs show a large amount of variation – within a single scan, we might see a dozen different TTLs. In other cases, when the TTL for each packet is plotted on a grid corresponding to the offset of the destination address within the destination subnet, it looks like a simple drawing of a video-game sprite. The tools in `./ttl-scanner` create images and other data based on the variations of TTLs in horizontal scans, to help investigate this phenomenon.

See `./ttl-scanner` for more information.

meanies

The "Greenwich Meanie" (aka "The Meanie") is a pervasive, unusual, and yet-to-be-explained IBR phenomenon. This directory contains a description of the Meanie phenomenon, and scripts for extracting information from Meanie packets. This information is used in later analyses (such as the `meanie-geo-location` notebooks).

See `./meanies` for more information.

meanie-geo-location

One of the interesting aspects to the Meanie phenomenon is that its sources do not appear to localized in the same way that many other IBR phenomena are.

This directory contains analyses of the Meanie using geolocation data and device probing.

Consult the Jupyter notebooks in `./meanie-geo-location` for more information.

anomaly-detection

Categorizing scanner behavior is a key problem in IBR analysis. The Jupyter notebook in this directory, and the corresponding documentation in the `Documentation` subdirectory, show how to do a cluster analysis to identify scanners that behave similarly as well as those that behave in an anomalous (or unfamiliar) manner.

fragments

Many IBR analyses ignore fragmented packets, because they are relatively rare, and are usually believed to be caused by misconfiguration. Our tools in `./pcap-ingestion` ignore them, because they are difficult or impossible to categorize in the same manner as "ordinary" packets.

We've recently observed an increase in the occurrence of fragments in our data, however, so we've added tools for extracting fragmented packets and examining them.

See `./fragments/README.md` for more information.

compare-subnets

The `./compare-subnets` directory contains example scripts for comparing the observed IBR activity for two or more subnets.

See `./compare-subnets/README.md` for more information.

bubble-movie

Visualizing IBR data can be challenging. The `./bubble-movie` directory contains some utilities for helping to visualize network behaviors that change over time.

See `./bubble-movie/README.md` for more information.