Long Term Data Storage Issues for Situational Awareness

[Extended Abstract]

John McHugh
Dalhousie University
and
University of North Carolina
mchugh at cs dot dal dot ca

1. INTRODUCTION

Despite rapid increases in storage capacity and attendant reductions in cost, network traffic continues to grow more rapidly than our ability to archive it. Even a modest network can generate enormous quantities of data in a relatively short time. Despite the fact that disk storage is now in the region of $100US per terabyte, the costs of maintaining a substantial traffic archive are significant. The purpose of this paper is to discuss some of the issues associated with long term archival of network traffic in light of some recent investigations into the distributions of such traffic as observed at the border of a small enterprise network. We will begin by describing the results of some recent investigations into both low frequency and high frequency traffic where frequency, in this case, refers to the intervals between connections involving a given IP address that is outside the enterprise in question. In both cases, we suggest (possibly lossy) data reduction or compression techniques that can be used to greatly reduce the volume of traffic that needs to be stored whole retaining most of its utility for retrospective and historical studies. Following this we will consider data structures that can be used for the efficient storage of the remaining data that is not easily reduced.

2. CONTACT DISTRIBUTION

Some years ago, we noticed that plotting the number of external addresses that contact a specific number of internal addresses across a monitored border approximates a straight line on a log/log scale. Figures 1 and 2 illustrate this on networks comprising a significant single digit percentage of the IPv4 address space and on a /22 that we have monitored for several years. The contact lines are described in more detail in [4]. The distribution departs from the canonical exponential form at both extremes with the portion representing infrequent contacts falling above the straight line fit near the Y axis and the portion representing very frequent contacts falling to the right of the fit near the X axis. Rather than considering this as a single, complex distribution, we prefer to think of it as resulting from the aggregation of several distinct processes. The low frequency component accounts for a substantial portion of the total addresses seen while the high frequency component accounts for a large percentage of the total flows. Each of these two classes has characteristics that lead to efficient storage strategies without the loss of essential or interesting information.

Figure 1 shows the contact line for a large network for part of September 2003 while Figure 2 shows contact data for a month from the /22 that we have been monitoring. The data was first filtered to retain only traffic from an external source address to a destination addresses within the monitored network then sorted by flow start time. The data...
Figure 1: Superimposed hourly contact lines for a large network

Figure 2: Monthly aggregate contact line for a /22

was then filtered using a Bloom filter so that only the first record for each unique source IP / destination IP pair was kept for subsequent analysis. Both lines are similar, but the difference in volume is such that aggregation of a month’s data is required to develop the line on the /22, while a single hour suffices on the larger network. For the /22, several hundred thousand external addresses appear associated with only single internal addresses. At the bottom of the figure, one to three external addresses each contact most of the host counts between 50 or so and 1016.

3. HIGH FREQUENCY DATA

The monitored /22 contains 1016 usable addresses, but is less than 10% occupied with an active host population of 60 - 80 on a typical day. Most of the high frequency traffic is scanning under any reasonable definition, as the vast majority of it consists of attempts to open connections to internal addresses whether or not there is a host present at the address. In Figure 2 peaks at 254, 508, 762, and 1016 indicate scans of exactly 1-4 on the /24s making up the /22. Many of the scans target ports for which no services are offered by any host on the network and elicit no responses. In a few cases, there are significant interactions with a scanner, resulting in compromise of an internal host, but these are rare. Nonetheless, when the traffic is recorded as SiLK NetFlow records[3], scan traffic comprises between 25% and 95% of the total volume being saved. At the time a scan occurs, the precise pattern of the scan may be of interest, but after 6 months to a year, abstracting the scan into a simple record that contains the source address, the range scanned, the number of probes, ports targeted, etc. can result in the elimination of thousands of records from the traffic archive, reducing its volume by 50% or more. Traffic that resulted in interactions with the scanner need not be eliminated, so no forensically relevant material is lost. Even if the archive is being used to inform artificial traffic generation, such records should suffice to induce realistic scan traffic. Note that for archival purposes, it is not necessary to argue about precise definitions of scanning. The majority of the data that can be removed comes from sources that attempt to make a large number of connections to a variety of destinations but do not succeed in most cases. This data is easily identified. Subtle scanners that probe at very slow rates usually do not create an archival problem.

4. VERY LOW FREQUENCY DATA

At the other end of the spectrum is data from addresses that are seen only a very few times. In the network that we have been monitoring, over 90% of the observed external addresses appeared 10 or fewer times in 14 months. Figure 4 shows the contact distribution for the network that we have been monitoring.

<table>
<thead>
<tr>
<th>Flows</th>
<th>Sources</th>
<th>%</th>
<th>Cum. %</th>
<th>Flows</th>
<th>%</th>
<th>Cum. %</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6096637</td>
<td>48.74%</td>
<td>48.74%</td>
<td>6096637</td>
<td>5.03%</td>
<td>5.03%</td>
</tr>
<tr>
<td>2</td>
<td>2291307</td>
<td>17.79%</td>
<td>66.53%</td>
<td>4522614</td>
<td>3.67%</td>
<td>8.70%</td>
</tr>
<tr>
<td>3</td>
<td>1034589</td>
<td>8.14%</td>
<td>74.67%</td>
<td>3103767</td>
<td>2.52%</td>
<td>11.22%</td>
</tr>
<tr>
<td>4</td>
<td>648227</td>
<td>5.10%</td>
<td>79.77%</td>
<td>2592908</td>
<td>2.11%</td>
<td>13.33%</td>
</tr>
<tr>
<td>5</td>
<td>419928</td>
<td>3.30%</td>
<td>83.07%</td>
<td>2099640</td>
<td>1.70%</td>
<td>15.03%</td>
</tr>
<tr>
<td>6</td>
<td>310678</td>
<td>2.44%</td>
<td>85.52%</td>
<td>1864068</td>
<td>1.51%</td>
<td>16.55%</td>
</tr>
<tr>
<td>7</td>
<td>224894</td>
<td>1.77%</td>
<td>87.28%</td>
<td>1574898</td>
<td>1.28%</td>
<td>17.83%</td>
</tr>
<tr>
<td>8</td>
<td>148976</td>
<td>1.16%</td>
<td>88.44%</td>
<td>1479897</td>
<td>1.20%</td>
<td>19.03%</td>
</tr>
<tr>
<td>9</td>
<td>145821</td>
<td>1.15%</td>
<td>89.59%</td>
<td>1312389</td>
<td>1.07%</td>
<td>20.06%</td>
</tr>
<tr>
<td>10</td>
<td>126003</td>
<td>0.99%</td>
<td>90.58%</td>
<td>1260030</td>
<td>1.02%</td>
<td>21.12%</td>
</tr>
<tr>
<td>&gt;100</td>
<td>96127</td>
<td>0.76%</td>
<td>91.34%</td>
<td>9612704</td>
<td>0.76%</td>
<td>56.39%</td>
</tr>
<tr>
<td>Total</td>
<td>12712460</td>
<td>100.00%</td>
<td>100.00%</td>
<td>12712460</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
</tbody>
</table>

Figure 3: Address and flow distributions for 14 months on the monitored/22

Addresses that appear in only 1 NetFlow record account for nearly 50% of the total addresses observed while only comprising about 5% of the flow records. Addresses that appear between 1 and 10 times comprise over 90% of the total addresses and about 20% of the flows. Addresses that generate over 100 flows are less than 1% of the addresses but account for more than half of the flows. These will include the scanners discussed in the previous section. While

\[1\] While the /22 contains 1024 addresses, the maximum count that we see is 1016. Addresses 0 and 255 in each /24 do not appear, having been “absorbed” within the instrumented router.
these infrequent addresses do not present an opportunity for a significant reduction in archive volume, they may offer an opportunity for index reduction. We are in the process of developing methods for indexing network traffic archives and believe that delaying the index binding for addresses until they have been seen repeatedly may offer opportunities to reduce the volume of the index. We can use the SiLK sets to record addresses that have been seen only once, allowing a fast efficient lookup to be made. Combining a cumulative set for one time addresses with sets covering specific time periods can help pinpoint the sighting. Alternatively, we can adapt the SiLK multiset or bag to record first and last sightings along with the number of sightings. When the sighting count exceeds some threshold, a proper index entry can be created and all sightings tracked. We are currently experimenting with a prototype implementation that will track the approximately $13 \times 10^6$ IP addresses seen in the 14 months of enterprise data discussed here. This implementation uses a Cuckoo Hash (discussed in the next section) and records the times of first and last observations of each IP address along with the number of flows, packets, and bytes that it sourced.

The low frequency data appears to come from a number of activities. We find infrequent connection attempts that may be the result of random scanning by infected machines as well as evidence of backscatter from DDoS attacks of various kinds. We also find a significant number of full connections, primarily SMTP. In the course of the 14 months of analysis, we observed several hundred thousand one time mail connections with a complete 3-way handshake and significant data transfers. This seems to be excessive for a network with a handful of real users. In addition, interesting patterns appear in port usage. Figure 4 shows monthly port volumes for the top 20 ports targeted by addresses sourcing fewer than 11 flows in 14 months. Only a few ports show activity in more than a few months and these appear to be associated with known P2P services that waxed and waned in popularity. Of note is the exponential decay of port 27015. This is used by the multi-player game, “half-life” and an infected machine containing a half-life server was discovered and removed from the network shortly after data collection started. Nonetheless, attempts to contact the server persist to the present time. Among the lower ranked ports are several where all connection attempts occurred in a short period and all targeted an address that was unoccupied. We speculate that these may have been based on faulty information, but are at a loss to explain its source or distribution.

Figure 4: Monthly low frequency UDP port usage for frequently accessed ports.

The behavior for TCP data is similar as can be seen in figure 4. Among the top ports are common service ports as well as a few well known P2P ports. For most of the lower ranked ports, no information concerning their possible use appears to be available. If these patterns are found to apply to other situations, groupings by targeted port or service, associating a set of IPs with each group might serve to simplify indexing and storage of this low frequency data.

5. INDEX DATA STRUCTURES

For IPv4 addresses, we have been using nested pointer arrays to realize sets and multisets[5], however, these do not scale to IPv6 or to larger index spaces such as connections involving pairs of addresses. For these, we are currently investigating structures such as Cuckoo hashes[6], a hash table structure based on Bloom filters[1]. Where the Bloom filter relies on multiple hash functions to avoid the need for collision resolution, the cuckoo hash moves entries to alternate locations to resolve collisions. These functions have several desirable properties.

1. They offer fast, bounded, constant time lookup for entries already present in the table.

2. The entry of new items is also bounded in time, but the bound is a function of the degree of occupancy of the table.

Figure 5: Monthly low frequency TCP port usage for frequently accessed ports.
3. The multiple hash functions and lookup strategies lend themselves to parallel processing, so that, at a processor per hash offers the possibility of performing multiple hash and lookup operations in the time required by a single operation on a uniprocessor.

4. The degree of table occupancy that can be guaranteed before a deadlock requiring table reallocation may occur is a function of the number of hash functions used and of the number of table entries per hash cell. It reaches 90% plus with just 4 functions and one entry per cell.

We have an experimental implementation that allows exploration of the tradeoffs between the number of hash functions used and the number of entries per cell\(^2\). Using 4 hash functions and 4 entries per cell, we typically see utilization of 65% to 70% before any relocation of entries is required. Utilization in excess of 99% is typically obtained before reallocation. Reallocation consists of doubling the table size and rehashing the existing entries using an additional bit of the hash. Since the new table is less than half full, this process never requires relocation of an entry, once it is placed. Once a table is created, it can be sorted and stored in a compact binary format that supports set operations such as union, intersection, and differencing of its keys in linear time. If the entries contain information such as volumes of traffic associated with IP addresses or connection parameters (the keys), the usual multiset operations of addition, subtraction, computation of ratios, etc. can also be supported and performed in linear time. We believe that these structures will prove to be an effective substitute for the SiLK sets and Bags that form the basis of much of our current analysis.

For static indices such as those covering a past period of activity, we are also considering minimal perfect hashes[2]. Indices covering billions of entries can be created in a few hours of processing time on modest machines. We can envision indexing a large set of IP addresses, using minimal perfect hashes to record, for example, the months in which we have data for the address. A 128 bit vector will cover a span of over ten years at this granularity, and even using multiple vectors for (source, destination) × (TCP, UDP, ICMP, Other), we can provide a ten year index with a size per entry of about 1000 bits plus the size of the key. Such indices could be constructed monthly or even less frequently with more dynamic structures being used to index more recent data. Indices of this sort lend themselves to distribution, as well. Hits on the ten year index would lead the user into monthly indices which might lead to hourly data repositories from which the data of interest could be extracted.

6. REFERENCES


\(^2\)If a hash value is allowed to point to a cell that can contain multiple entries, linear search is used to find an entry within the cell.