Building a better NetFlow
(to appear in SIGCOMM 2004)

Cristian Estan, Ken Keys, David Moore, George Varghese

University of California, San Diego

IETF60 – Aug 4, 2004 – IPFIX WG
Disclaimers

• "NetFlow" used generically, no particular vendor or implementation implied

• Proposed changes are metering related, but can affect ipfix protocol design

• Not meant to be the definitive solution, but to help encourage discussion and improvements
Sampling pros and cons

- Reduces processor load
- Reduces memory usage
- Reduces bandwidth for reporting
- Results less accurate
- Cannot estimate non-TCP flow counts

Finding the sampling rate that balances the pros and cons is hard
The best choice depends on traffic mix 😞
# Fixing NetFlow

<table>
<thead>
<tr>
<th>NetFlow problem</th>
<th>How we solve it</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory and bandwidth usage strongly depend on traffic mix</td>
<td>Adapting sampling rate (part 2)</td>
</tr>
<tr>
<td>Network operator must set sampling rate</td>
<td></td>
</tr>
<tr>
<td>Mismatch of flow termination heuristics and analysis</td>
<td>Measurement bins (part 1)</td>
</tr>
<tr>
<td>Cannot estimate number of non-TCP flows</td>
<td>Sampling flows (part 3)</td>
</tr>
</tbody>
</table>
Operating with time bins

- Both operators and researchers usually prefer working with fixed time bins
- Use fixed size time bins (say 1 minute)
- Terminate all flow records at the end of the bin (but don’t report immediately)
- Could use different sampling rates for each bin, including decreasing sampling within a bin as needed
- Simplifies analysis and reduces error
- Time bins allow reconstruction of flow timeouts
Analysis uses time bins anyway

<table>
<thead>
<tr>
<th>Category</th>
<th>Flows (%)</th>
<th>Packets (%)</th>
<th>Bytes (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web</td>
<td>47.33</td>
<td>54.35</td>
<td>61.48</td>
</tr>
<tr>
<td>File Sharing</td>
<td>3.74</td>
<td>3.35</td>
<td>2.43</td>
</tr>
<tr>
<td>FTP</td>
<td>0.07</td>
<td>0.52</td>
<td>0.54</td>
</tr>
<tr>
<td>Email</td>
<td>3.24</td>
<td>4.67</td>
<td>4.06</td>
</tr>
<tr>
<td>Streaming</td>
<td>1.60</td>
<td>7.26</td>
<td>13.07</td>
</tr>
<tr>
<td>DNS</td>
<td>27.26</td>
<td>6.13</td>
<td>1.16</td>
</tr>
<tr>
<td>Games</td>
<td>0.03</td>
<td>0.06</td>
<td>0.01</td>
</tr>
<tr>
<td>Other TCP</td>
<td>6.05</td>
<td>21.03</td>
<td>15.86</td>
</tr>
<tr>
<td>Other UDP</td>
<td>0.84</td>
<td>0.78</td>
<td>0.48</td>
</tr>
<tr>
<td>Not TCP/UDP</td>
<td>9.84</td>
<td>1.86</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Site: San Jose (sj-20)
Date: February 5th, 2004
Relationship to IPFIX

- draft-ipfix-protocol-3, section 4:
  - 4.1: seems to require timeout based flows, allows for expiry based on resource constraints, but it is unclear on permissibility of using time bins
  - 4.2: allows for export of long-lasting flows on schedule determined by exporting process, but is unclear about what that entails

- draft-ipfix-protocol-3, section 8:
  - would it require putting the same start/end time (or bin #) in all of the Flow Records, or is there a way to specify the bin efficiently for an entire group of records
## Fixing NetFlow

<table>
<thead>
<tr>
<th>NetFlow problem</th>
<th>How we solve it</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory and bandwidth usage strongly depend on traffic mix</td>
<td>Adapting sampling rate (part 2)</td>
</tr>
<tr>
<td>Network operator must set sampling rate</td>
<td>Measurement bins (part 1)</td>
</tr>
<tr>
<td>Mismatch of flow termination heuristics and analysis</td>
<td></td>
</tr>
<tr>
<td>Cannot estimate number of non-TCP flows</td>
<td>Sampling flows (part 3)</td>
</tr>
</tbody>
</table>
Adaptive NetFlow

• Choose the sampling rate based on traffic
  – Use a high sampling rate when traffic allows
  – Keeping counters meaningful as sampling rate varies
  – Ensuring we never overload CPU
  – Ensuring we never run out of memory
Adapting sampling rate

- If **multiple sampling rates** in effect while flow active, byte and packet counters meaningless
- Decreasing sampling rate – pretend to throw away sampled packets
- Increasing rate – not possible, since information discarded.
- Start each time bin with aggressive sampling
Limiting CPU usage

- Renormalization in parallel with operation
- Efficient renormalization – for most records only simple integer arithmetic, no random number generation
  - Updating 1 entry 3.4 μs
  - Renormalizing 1 entry 1.5 μs
- Vendor configures initial sampling rate high enough for CPU to keep up with minimum sized packets
Memory Usage:
What happens under DoS?
Rate adaptation and memory usage

- Trigger renormalization whenever the number of entries reaches a fixed threshold
- Must choose new sampling rate so that enough records discarded by renormalization
  - Use partial histogram of packet counters
- Actual memory at router must exceed the desired number of records per bin $M$ to allow renormalization and buffering of old records
Main tuning knob: # of records M

- Controlled resource usage
- User configures number of desired records to be exported
- More meaningful than sampling rate
  - Relative error in estimating an aggregate that is a certain fraction of the traffic depends on $M$
- Can produce reports of various sizes and send them with different reliability levels
  - Dropping random records is worse than generating fewer records by using lower sampling rate
Relationship to IPFIX

- SCTP-PR: use different priority levels for different report sizes
- Reliable transport in general: may be able to share memory for flows from previous time bin with memory needed for retransmission
- draft-ipfix-protocol-3, section 8:
  - The sampling rate can vary frequently, should it be in the Flow Record or an Option Record?
  - If exporting multiple reports at different effective sampling rates, the same flow may be exported more than once, how should this be handled?
# Fixing NetFlow

<table>
<thead>
<tr>
<th>NetFlow problem</th>
<th>How we solve it</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory and bandwidth usage strongly depend on traffic mix</td>
<td>Adapting sampling rate (part 2)</td>
</tr>
<tr>
<td>Network operator must set sampling rate</td>
<td>Measurement bins (part 1)</td>
</tr>
<tr>
<td>Mismatch of flow termination heuristics and analysis</td>
<td>Sampling flows (part 3)</td>
</tr>
<tr>
<td>Cannot estimate number of non-TCP flows</td>
<td></td>
</tr>
</tbody>
</table>

COOPERATIVE ASSOCIATION FOR INTERNET DATA ANALYSIS
University California, San Diego – Department of Computer Science
Counting flows

- Goal: Unbiased, accurate flow counts for arbitrary post aggregation of the flows.
Flow Counting Extension

- Use “adaptive sampling” by Wegman and Flajolet
- Keep a table of all flow identifiers with hash(flowID)<\(1/2^{\text{depth}}\)
- At analysis scale flow counts by \(2^{\text{depth}}\)
- Implement with CAM
- To fit memory, increase depth dynamically
Relationship to IPFIX

- SCTP-PR: use different priority levels for different report sizes
- draft-ipfix-protocol-3, section 8:
  - The sampling rate can vary frequently, should it be in the Flow Record or an Option Record?
  - If exporting multiple reports at different effective sampling rates, the same flow may be exported more than once, how should this be handled?
- Would this require a separate template to export?
  - Basically the only thing to be exported here are the Flow Keys themselves.
Measurements

• Limited time, so for more details and results:

ANF results
# FCE results

<table>
<thead>
<tr>
<th>Aggregate</th>
<th>size</th>
<th>FCE</th>
<th>$\hat{M}_1$</th>
<th>$\hat{M}_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>bias</td>
<td>st.dev.</td>
<td>bias</td>
</tr>
<tr>
<td>ALL Traffic (*)</td>
<td>100</td>
<td>0.02</td>
<td>0.96</td>
<td>-35.55</td>
</tr>
<tr>
<td>ALL TCP Traffic</td>
<td>78.</td>
<td>0.10</td>
<td>1.16</td>
<td>-17.39</td>
</tr>
<tr>
<td>HTTP</td>
<td>58.</td>
<td>0.27</td>
<td>1.29</td>
<td>-19.24</td>
</tr>
<tr>
<td>ALL UDP Traffic (*)</td>
<td>20.</td>
<td>-0.13</td>
<td>2.26</td>
<td>-100.00</td>
</tr>
<tr>
<td>DNS (*)</td>
<td>8.0</td>
<td>0.03</td>
<td>3.94</td>
<td>-99.26</td>
</tr>
<tr>
<td>Netbios (*)</td>
<td>7.9</td>
<td>-1.97</td>
<td>3.90</td>
<td>-39.27</td>
</tr>
<tr>
<td>AS 2914 src (*)</td>
<td>7.2</td>
<td>0.92</td>
<td>5.43</td>
<td>-15.66</td>
</tr>
<tr>
<td>Unclassified TCP</td>
<td>5.1</td>
<td>2.19</td>
<td>5.60</td>
<td>-47.07</td>
</tr>
<tr>
<td>SMTP</td>
<td>2.3</td>
<td>-0.54</td>
<td>5.96</td>
<td>0.56</td>
</tr>
<tr>
<td>ALL ICMP Traffic (*)</td>
<td>1.5</td>
<td>-2.12</td>
<td>8.54</td>
<td>-100.00</td>
</tr>
<tr>
<td>POP</td>
<td>0.3</td>
<td>4.23</td>
<td>19.01</td>
<td>17.71</td>
</tr>
<tr>
<td>IRC (*)</td>
<td>0.3</td>
<td>-9.01</td>
<td>18.32</td>
<td>-71.48</td>
</tr>
</tbody>
</table>
Conclusions

- Adaptive NetFlow improves NetFlow
  - Predictable resource usage even under adverse traffic
  - More meaningful tuning knob $M$ or records $M$
  - Binned measurement matches analysis better
  - No hardware changes required

- Flow Counting Extension gives accurate flow counts for non-TCP flows too
Any more questions?
Theoretical results

- If ANF/NetFlow generates $M$ entries, the relative standard deviation for aggregate that is fraction $f$ of the traffic is at most $\sqrt{\frac{1}{Mf}}$ in packets and $\sqrt{\frac{s_{\text{max}}}{s_{\text{avg}}}Mf}$ in bytes.

- If FCE generates $M$ entries, the relative standard deviation for aggregate that is fraction $f$ of the traffic is $\sqrt{\frac{1}{Mf}}$ in flows.
Flow termination versus bins

- Flow termination heuristics require extra work to do the binning that can increase error in results
- Terminating flows at end of bin is backward compatible