Recommendations for using Multiple IP Addresses in Benchmarking Tests

draft-lencse-bmwg-multiple-ip-addresses

Gábor LENCSE lencse@sze.hu (Széchenyi István University) – presenter
Keiichi SHIMA keiichi.shima@g.softbank.co.jp (SoftBank)

IETF 119, BMWG, March 21, 2024.
Outline

• Reminder
  – Problem description: why testing with multiple IP addresses is needed?
  – Recommended Solution: Usage of multiple, pseudorandom IP addresses

• Measurements: OpenBSD IPv4 and IPv6 packet forwarding
  – Description of the measurements
  – Results
Reminder: Problem Description: Conditions

- RFC 2544 has defined a test frame format with fixed IP addresses and fixed port numbers.
- RFC 4814 introduced pseudorandom port numbers, but it kept the usage of a single source and destination IP address pair when a single destination network is used.
- Receive Side Scaling (RSS) supports the receiving of multi-million packets per second by distributing the load among the CPU cores
  - Depending on implementation, the hash function includes:
    - 1\textsuperscript{st} type: source IP address, destination IP address, source port, destination port
    - 2\textsuperscript{nd} type: source IP address, destination IP address
Reminder: Problem Description: Unfairness

• RFC 4814 pseudorandom port numbers + 1\textsuperscript{st} RSS implementation
  – Works perfectly (port numbers ensure entropy)
  – All CPU cores are used, load is distributed approximately evenly

• RFC 4814 pseudorandom port numbers + 2\textsuperscript{nd} RSS implementation
  – Gives poor results (no entropy is ensured as IP addresses are fixed)
  – Thus only two CPU cores are used (one core per direction)

• However, network interconnect devices using the 2\textsuperscript{nd} RSS implementation work perfectly, when they forward Internet traffic (IP addresses ensure entropy)

→ Conditions for the laboratory tests should be improved!
Reminder: Recommended Solution

• Basic idea: Let us use pseudorandom IP addresses!
  – This is the spirit of RFC 4814 applied to the IP addresses 😊

• Problems to solve:
  – What ranges can be used?
    • There is scarcity in IPv4 addresses reserved for benchmarking
      – 198.18.0.0/15 was reserved for benchmarking
    • There is abundance in IPv6 addresses reserved for benchmarking
      – 2001:2::/48 was reserved for benchmarking
  – What ranges should be used?
    • A trade-off is pointed out
Measurement Environment

- **Dell PowerEdge R730 servers**
  - 3.2GHz Intel Xeon E5-2667v3 CPUs
  - 128GB 2133MHz DDR4 SDRAM
  - Intel 10G dual-port X540 NIC
- **Direct cable connections**
- **Tester: Debian 9.13**
  - Running *siitperf*
- **DUT: OpenBSD 7.3**
  - 2nd type of RSS implementation
  - IPv4 and IPv6 packet forwarding

---

*Benchmarking with multiple IP addresses*
Parameters for IPv4 packet forwarding

• Tester
  – eno1: 1000 different addresses
    198.18.0.2/16-198.18.3.233/16
  – eno2: 1000 different addresses
    198.19.0.2/16-198.19.3.233/16

• DUT
  – ix0: 198.18.0.1/16
  – ix1: 198.19.0.1/16
Parameters for IPv6 packet forwarding

- **Tester**
  - eno1: 1000 different addresses
    - 2001:2::[0000-03e7]:2/64
  - eno2: 1000 different addresses
    - 2001:2:0:8000::[0000-03e7]:2/64

- **DUT**
  - ix0: 2001:2::1/64
  - ix1: 2001:2:0:8000::1/64
IPv4 Packet Forwarding Performance of the OpenBSD

- Throughput of OpenBSD using bidirectional traffic, expressed as frames per second *per direction*

<table>
<thead>
<tr>
<th>IP addresses</th>
<th>fixed</th>
<th>random</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median (fps)</td>
<td>390,125</td>
<td>1,277,414</td>
</tr>
<tr>
<td>Minimum (fps)</td>
<td>367,116</td>
<td>1,249,999</td>
</tr>
<tr>
<td>Maximum (fps)</td>
<td>437,745</td>
<td>1,296,876</td>
</tr>
<tr>
<td>Dispersion (%)</td>
<td>18.10</td>
<td>3.67</td>
</tr>
</tbody>
</table>

\[ dispersion = \frac{\text{maximum} - \text{minimum}}{\text{median}} \cdot 100\% \]

- The usage of multiple IP addresses resulted in a more the 3-fold increase in the throughput and much more stable results.
IPv6 Packet Forwarding Performance of the OpenBSD

• Throughput of OpenBSD using bidirectional traffic, expressed as frames per second *per direction*

<table>
<thead>
<tr>
<th>IP addresses</th>
<th>fixed</th>
<th>random</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median (fps)</td>
<td>384,970</td>
<td>582,165</td>
</tr>
<tr>
<td>Minimum (fps)</td>
<td>351,553</td>
<td>577,024</td>
</tr>
<tr>
<td>Maximum (fps)</td>
<td>385,749</td>
<td>597,657</td>
</tr>
<tr>
<td>Dispersion (%)</td>
<td>8.88</td>
<td>3.54</td>
</tr>
</tbody>
</table>

\[ dispersion = \frac{\text{maximum} - \text{minimum}}{\text{median}} \cdot 100\% \]

• The usage of multiple IP addresses resulted in about 50% increase in the throughput and more stable results.
We would like to ask for feedback

• Do you think that the proposed solution is appropriate?
• Do you have any idea what to change, add, etc.?
• All your comments and suggestions are welcome!