Benchmarking Methodology for Stateful NATxy Gateways using RFC 4814 Pseudorandom Port Numbers

draft-lencse-bmwg-benchmaring-stateful

Gábor LENCSE lencse@sze.hu (Széchenyi István University)
Keiichi SHIMA keiichi@iijlab.net (IIJ Innovation Institute) – presenter

Summary of the Proposal

• Guides to achieve reproducible and meaningful stateful NATxy performance measurements
  – Facilitating to carry out all the measurement procedures of RFC 2544 / RFC 5180 / RFC 8219 like throughput, latency, frame loss rate, etc. to benchmark stateful NATxy (NAT44, NAT64, etc.) gateways
  – Adding new performance metrics specific to stateful testing:
    • Connection setup performance: maximum connection establishment rate
    • Connection tear down performance: connection tear down rate (NEW!)
  – Providing guidelines how to use RFC 4814 pseudorandom port numbers with stateful NATxy gateways
Progress of the draft

• Version 00 (presented at IETF 111)
  – Framework for stateful testing, basics of the methodology

• Version 02 (presented at IETF 112)
  – Refinement of the management of the connection tracking table
    • Benefit: more straightforward measurements method, clear results
    • Also presented measurement results of *iptables* stateful NAT44

• Version 03 (the current one)
  – Introduction of connection tear down rate measurement method
  – Measurement results of Jool Stateful NAT64 are also available
Reminder: Test Setup

- Methodology works with any IP versions
  - To facilitate easy understanding, we use the example of stateful NAT44

```
+--------------------------------------+
| 10.0.0.2 | Initiator                Responder | 198.19.0.2 |
+----------|---------------------------+-----------------+
| Tester   | [state table] | public IPv4    |
| private IPv4 |                     |                 |
+--------------------------------------+
| 10.0.0.1 | DUT: Stateful NATxy gateway | 198.19.0.1   |
+----------|-------------------------------+-----------------+
| private IPv4 | [connection tracking table] | public IPv4     |
+--------------------------------------+
```
Reminder: Measurements in two Phases

• Preliminary test phase
  – It serves two purposes:
    • The connection tracking table of the DUT is filled.
    • The state table of the Responder is filled with valid four tuples.
  – It can be used without the real test phase to measure the maximum connection establishment rate.

• Real test phase
  – It MUST be preceded by a preliminary test phase.
  – The actual measurement procedure (throughput, frame loss rate, latency, PDV, IPDV) is performed as defined in RFC 8219.
Reminder: To support repeatable measurements

• There are two extreme situations that we can simply ensure
  1. When all test frames create a new connection
     • Ideal for measuring maximum connection establishment rate
  2. When test frames never create a new connection
     • Ideal for all other tests: throughput, latency, frame loss rate, PDV, etc.

• Conditions to achieve them:
  – Large enough and empty connection tracking table for each test
  – Pseudorandom enumeration of all possible port number combinations in the preliminary phase
  – Properly high timeout value in the DUT
Connection tear down rate measurements

• Having no better opportunity due to black box testing, we recommend an aggregate measurement:
  – Load $N$ number of connections into the connection tracking table of the DUT
    • Performed as a preliminary phase measurement step (without real test phase)
  – Delete the entire content of the connection tracking table of the DUT
    • Using some out of band method, e.g. removal of Linux kernel module

\[
\text{Connection tear down rate} = \frac{N}{\text{deletion time of the connection tracking table}}
\]

– To be measured using different order of magnitude values for $N$
Connection tear down rate measurements

• Technical refinement (not yet in the draft)
  – Subtract the deletion time of an empty table from that of the full table
    • It counts when low number of connections are used
    • It also eliminates the remote command execution overhead

• Potential problem
  – The deletion of a connection due to timeout MAY require a different amount of work than its deletion due to the deletion of the entire content of the connection tracking table
    • And this may depend on the implementation 😐

• Actual measurements:
  – **iptables** stateful NAT44, Jool stateful NAT64 implementations

Benchmarking Stateful NATxy Gateways
Connection tear down rate measurement of *iptables*

- The \( N \) number of connections was set with the source port number destination port number ranges
  - Usually increased fourfold, except the last case (due to memory limit)
  - The hash table size was usually increased proportionally, except the last two cases (due to memory limit)
  - NUMA issue influenced the last measurement
    - The connection tracking table did not fit into the NUMA local memory
  - The connection tear down time of an empty connection tracking table was measured for all cases (and it was indeed significantly different)
Connection tear down rate of **iptables** stateful NAT44

<table>
<thead>
<tr>
<th></th>
<th>1.56M</th>
<th>6.35M</th>
<th>25M x4</th>
<th>100M x4</th>
<th>400M x2</th>
<th>800M x2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>num. conn.</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>src ports</strong></td>
<td>2,500</td>
<td>5,000</td>
<td>10,000</td>
<td>20,000</td>
<td>40,000</td>
<td>40,000</td>
</tr>
<tr>
<td><strong>dst ports</strong></td>
<td>625</td>
<td>1,250</td>
<td>2,500</td>
<td>5,000</td>
<td>10,000</td>
<td>20,000</td>
</tr>
<tr>
<td><strong>conntrack t. s.</strong></td>
<td>2^21</td>
<td>2^23</td>
<td>2^25</td>
<td>2^27</td>
<td>2^29</td>
<td>2^30</td>
</tr>
<tr>
<td><strong>hash table size</strong></td>
<td>2^21</td>
<td>2^23</td>
<td>2^25 x4</td>
<td>2^27 x2</td>
<td>2^28 x1</td>
<td>2^28</td>
</tr>
<tr>
<td><strong>full cont. t. del t.</strong></td>
<td>4.33</td>
<td>18.05</td>
<td>74.47</td>
<td>305.33</td>
<td>1178.3</td>
<td>2263.1</td>
</tr>
<tr>
<td><strong>empty ct. t. del t.</strong></td>
<td>0.55</td>
<td>1.28</td>
<td>4.17 x4</td>
<td>15.74 x2</td>
<td>31.2 x1</td>
<td>31.2</td>
</tr>
<tr>
<td><strong>conn. del time</strong></td>
<td>3.78</td>
<td>16.77</td>
<td>70.30 x4</td>
<td>289.59 x4</td>
<td>1147.2 x2</td>
<td>2232.0</td>
</tr>
<tr>
<td><strong>conn. tear d. rate</strong></td>
<td>413.4</td>
<td>372.7</td>
<td>355.6</td>
<td>345.3</td>
<td>348.7</td>
<td>358.4</td>
</tr>
</tbody>
</table>

Units: *seconds* for time; *1,000 connections/s* for connection tear down rate
Connection tear down rate measurement of Jool

- The $N$ number of connections was set with the source port number destination port number ranges
  - Increased fourfold (usually by doubling the size of both ranges)
  - Unlike previously with `iptables`, no tuning was done with Jool
  - The connection tear down time of an empty connection tracking table was measured only once (without tuning, there was no difference)
Connection tear down rate of Jool stateful NAT64

<table>
<thead>
<tr>
<th>num. conn.</th>
<th>1.56M</th>
<th>6.35M</th>
<th>25M</th>
<th>100M</th>
<th>400M</th>
<th>1600M</th>
</tr>
</thead>
<tbody>
<tr>
<td>src ports</td>
<td>2,500</td>
<td>5,000</td>
<td>10,000</td>
<td>20,000</td>
<td>40,000</td>
<td>40,000</td>
</tr>
<tr>
<td>dst ports</td>
<td>625</td>
<td>1,250</td>
<td>2,500</td>
<td>5,000</td>
<td>10,000</td>
<td>40,000</td>
</tr>
<tr>
<td>full cont. del med</td>
<td>0.87</td>
<td>2.05</td>
<td>7.84</td>
<td>36.38</td>
<td>126.09</td>
<td>474.68</td>
</tr>
<tr>
<td>full cont. del min</td>
<td>0.80</td>
<td>2.02</td>
<td>7.80</td>
<td>36.27</td>
<td>125.84</td>
<td>473.20</td>
</tr>
<tr>
<td>full cont. del max</td>
<td>0.91</td>
<td>2.09</td>
<td>7.94</td>
<td>36.80</td>
<td>127.54</td>
<td>481.38</td>
</tr>
<tr>
<td>emp. ct. del med</td>
<td>0.46</td>
<td>0.46</td>
<td>0.46</td>
<td>0.46</td>
<td>0.46</td>
<td>0.46</td>
</tr>
<tr>
<td>conn. del. time</td>
<td>0.41</td>
<td>1.59</td>
<td>7.38</td>
<td>35.92</td>
<td>125.63</td>
<td>474.22</td>
</tr>
<tr>
<td>conn. t. d. r. (M)</td>
<td><strong>3.811</strong></td>
<td><strong>3.931</strong></td>
<td><strong>3.388</strong></td>
<td><strong>2.784</strong></td>
<td><strong>3.184</strong></td>
<td><strong>3.374</strong></td>
</tr>
</tbody>
</table>

Units: *seconds* for time; *1,000,000 connections/s* for connection tear down rate
Request for feedback

• What do you think of the connection tear down rate measurement method?
  – Does it provide meaningful and reasonable results?
  – Could you recommend a better measurement method?

• Not yet done: measuring the size of the connection tracking table
  – We have ideas that need to be tested how they work in practice

• Is there any other measurement missing?

• Potential WG adoption?