Interoperability testing

... on live Internet

Petr Špaček • petr.spacek@nic.cz • 2019-05-13
Interoperability: Theory

- Read RFCs
- Implement
- Verify MUST/SHOULD/MAY ...
Interoperability: Reality

- Customers complain
- BIND can resolve _that_ domain, so ...
- You have to resolve it as well
RFC 1925: The Twelve Networking Truths

- (1) It Has To Work.
- (3) With sufficient thrust, pigs fly just fine. However, this is not necessarily a good idea.
- (8) It is more complicated than you think.
- (9) For all resources, whatever it is, you need more.
Finding a balance

- Remember:
  
  (3) With sufficient thrust, pigs fly just fine. However, **this is not necessarily a good idea.**
Interoperability vs. Knot Resolver

- Make it work
- On real Internet
  - RFCs do matter, but ...
  - "Fix" domains which matter
    - Do not add workarounds
      - Unless
        - Absolutely
          - Necessary
- Focus on real queries
Introducing respdiff

• “response differences”
• Pre-generate queries in wire-format
• Send DNS payload to multiple addresses
• Compare received responses
• Compute statistics
# respdiff toolchain

<table>
<thead>
<tr>
<th>Tool</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>qprep</td>
<td>generate wire-format query (PCAP, text ...)</td>
</tr>
<tr>
<td>orchestrator</td>
<td>send queries, gather responses</td>
</tr>
<tr>
<td></td>
<td>- alternative dnsjit</td>
</tr>
<tr>
<td>msgdiff</td>
<td>analyze response differences</td>
</tr>
<tr>
<td>diffsum</td>
<td>summarize differences</td>
</tr>
<tr>
<td>sumcmp</td>
<td>compare test summary against reference</td>
</tr>
</tbody>
</table>

### Additional tooling

<table>
<thead>
<tr>
<th>Tool</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>diffrepro</td>
<td>test diff reproducibility</td>
</tr>
<tr>
<td>sumstat</td>
<td>aggregate results to create reference</td>
</tr>
<tr>
<td>histogram</td>
<td>combine latency histograms</td>
</tr>
</tbody>
</table>
respdiff config

[servers]
  names = bind, kresd

[bind]
  ip = ::1
  port = 5301
  transport = tcp

[kresd]
  ip = ::1
  port = 5302
  transport = tcp

[diff]
  target = kresd
  criteria = opcode, rcode, flags, question, answertypes, answerrrsigs

[report]
  field_weights = timeout, malformed, opcode, question, rcode, flags, answertypes, answerrrsigs, answer, authority, additional, edns, nsid
First attempt: two resolvers

- BIND vs. Unbound
  - Try to use BIND as reference ...
- Compare all fields in responses
- **Way too many differences!**
- Load-balancers => differing rdata
- Authority and additional sections are a mess
  - Except for NXDOMAIN authority
Second attempt: two resolvers

- BIND vs. Unbound
- Ignore authority and additional sections
- Ignore rdata values in answer section
  - Compare sets of present types
- Still ~ 1% differences
- Dynamic auths?
- Broken auths?
- Too noisy
Third attempt: third resolver

- BIND vs. Unbound vs. Knot Resolver
- Comparison as before (sets of present types)
- Compare BIND vs. Unbound first
  - Skip query if BIND vs. Unbound disagree
  - Reference = BIND + Unbound
- Actually works
  - Filters out too “wild” domains
  - First "sieve" to detect major breakage
Three-resolver mode

[servers]
names = bind, unbound, kresd

[diff]
target = kresd

$ diffsum.py

== Differences statistics
manually ignored       0  0.00 % of answers
upstream unstable   1955  0.45 % of answers
not 100% reproducible 0  0.00 % of answers
target disagrees     302  0.07 % of not ignored
Diffsum output

== Field "rcode" mismatch statistics

<table>
<thead>
<tr>
<th>Expected</th>
<th>Got</th>
<th>Count</th>
<th>% of mismatches</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOERROR</td>
<td>SERVFAIL</td>
<td>238</td>
<td>78.81</td>
</tr>
<tr>
<td>NOERROR</td>
<td>NXDOMAIN</td>
<td>6</td>
<td>1.99</td>
</tr>
<tr>
<td>SERVFAIL</td>
<td>NOERROR</td>
<td>2</td>
<td>0.66</td>
</tr>
</tbody>
</table>

== Field "answertype" mismatch statistics

<table>
<thead>
<tr>
<th>Expected</th>
<th>Got</th>
<th>Count</th>
<th>% of mismatches</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td>7</td>
<td>2.32</td>
</tr>
<tr>
<td>CNAME</td>
<td>CNAME AAAA</td>
<td>1</td>
<td>0.33</td>
</tr>
</tbody>
</table>

== Field "rcode", expected 'NOERROR' got 'NXDOMAIN'

<table>
<thead>
<tr>
<th>Count</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>webserve-www.dynamicyield.com. A</td>
</tr>
</tbody>
</table>
Post processing: reproducibility

- Often diff is not reproducible
- Tool "diffrepro"

[bind]
restart_script = /usr/local/bin/restart-bind

[unbound]
restart_script = /usr/local/bin/restart-unbound

[kresd]
restart_script = /usr/local/bin/restart-kresd

$ diffrepro.py
diffrepro usage

$ diffsum.py

== Differences statistics
upstream unstable 1955 0.45 % of answers
not 100% reproducible 0 0.00 % of answers
target disagrees 302 0.07 % of not ignored

$ diffrepro.py
$ diffsum.py

upstream unstable 1961 0.45 % of answers
not 100% reproducible 123 0.03 % of answers
target disagrees 173 0.04 % of not ignored
Magic begins here
Post processing: classification

- Classification by hand
  - 173 diffs to be classified!
- Different approaches
  - focus on "difference"
  - focus on "new"
  - classify domains by "quality" - DNSViz?
  - combination of these
Post processing: looking for new

• Combine results from "reference" runs
• Compare last run with reference
• Tool "sumcmp"
  • summary compare
Post processing: differences vs. ref

1546dace-gl1556799847_vs_master_shortlist.iter.udp6.j384

stat sample size: 265
Key to the violin plot
Post processing: differences vs. ref

1546dace-gl1556799847_vs_master_shor
## Find new diffs

```
$ diffsum.py --without-ref-failing
```

<table>
<thead>
<tr>
<th>Category</th>
<th>Count</th>
<th>Percentage of Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>manually ignored</td>
<td>1129</td>
<td>0.26 %</td>
</tr>
<tr>
<td>upstream unstable</td>
<td>826</td>
<td>0.19 %</td>
</tr>
<tr>
<td>not 100% reproducible</td>
<td>0</td>
<td>0.00 %</td>
</tr>
<tr>
<td>target disagrees</td>
<td>302</td>
<td>0.07 %</td>
</tr>
</tbody>
</table>
Classify domains by "quality"

• Idea: DNSViz domains on list

• Categories
  • Ok
  • Warning
  • Error

• Investigate ok first, then warning ...

• Implementation difficulties
Links

- https://gitlab.labs.nic.cz/knot/respdiff
Open problems

- Automatic classification
- Reproducibility
  - in face of ever changing Internet