Discovery Method for a Validating Stub Resolver

Xavier Torrent Gorjón
xavier.torrentgorjon@os3.nl / sendotux@gmail.com

Final Thesis at SNE MSc

Supervisor: Willem Toorop (NLnet Labs)
willem@nlnetlabs.nl
whoami

- Former student of the SNE Msc at the Universiteit van Amsterdam.
- Did this research project as my final thesis, working with NLnet Labs.
Motivation

When things go wrong, sometimes fingers are pointed in the wrong direction. Seems to happen a lot with DNSSEC.

Objectives

• Measure the current state of DNSSEC deployment, from different points of view.
• Can we improve it without drastic changes?
Tools used

- Python scripts
- Classes provided by NLnet Labs to ease the task of parsing DNS data.
- The RIPE ATLAS probes!
Study case #1 results

- The vast majority of probes queried could successfully perform DNS queries (95%+).
- However, (regular) DNSSEC queries were successful only in 64% of the cases:

<table>
<thead>
<tr>
<th>Received RR</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>No RR</td>
<td>7.94%</td>
</tr>
<tr>
<td>DNSKEY (x2)</td>
<td>28.34%</td>
</tr>
<tr>
<td>DNSKEY (x2)+RRSIG</td>
<td>63.71%</td>
</tr>
</tbody>
</table>
Study case #1 results

- Things got worse when querying non-existing domains (both NSEC and NSEC3):

<table>
<thead>
<tr>
<th>Received RR</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>No RR</td>
<td>22.27%</td>
</tr>
<tr>
<td>Only SOA</td>
<td>21.49%</td>
</tr>
<tr>
<td>SOA + NSEC + RRSIG(x2)</td>
<td>56.23%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Received RR</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>No RR</td>
<td>12.44%</td>
</tr>
<tr>
<td>Only SOA</td>
<td>27.68%</td>
</tr>
<tr>
<td>SOA + RRSIG</td>
<td>3.62%</td>
</tr>
<tr>
<td>SOA + NSEC3(x2) + RSIG(x3)</td>
<td>0.58%</td>
</tr>
<tr>
<td>SOA + NSEC3(x3) + RSIG(x3)</td>
<td>55.67%</td>
</tr>
</tbody>
</table>
Study case #1 results

- With wildcard domain queries, retrieved responses were valid only in 40% of the cases.
• Seems as if, the harder the query, the worse the results. But who is the culprit?

• We attempted to run these queries again, but using the probes' ISP resolver, instead of the resolver predefined on them.
Study case #2 definition

If any of these DNS forwarders is not DNSSEC aware, the end user will not be able to get DNSSEC data.

Diagram:
- User
  - Home Router
  - Forwarding DNS
  - Forwarding DNS
  - ISP Recursive Resolver (DNSSEC aware)

Initial Query
- Forwarded Query
Study case #2 results

- The majority of probes could query their ISP resolvers directly.
- A small percentage didn't manage to do so.
- But, did this change affect the results?
Study case #2 results

• The number of successful DNSSEC queries raised from 64% to almost 80%.

• Valid NXDOMAIN answers increased from 56% to 75%.

• Wildcard queries were properly answered in 60% of the cases, from the previous 40%.

• All around, we observed a 20 points increase on the successful results.
Study case #2 conclusions

- The benefits of directly querying the ISP resolvers were quite noticeable and consistent.
- Individual reasons for this may vary, but we attribute this difference, mostly, to cheap hardware at the end points (home routers).
• Thanks to the people working at RIPE ATLAS, we got a new feature within 2 weeks!

Subject Re: Feature request: set CD bit on atlas DNS measurements

To Willem Toorop, Me <xavier.torrentgorjon@os3.nl>
Cc Robert Kisteleki <robert@ripe.net>, Philip Homburg <philip.homburg@ripe.net>

Dear Willem,

As of today you can include the following parameter in the JSON definition:

"cd": true

This isn't officially documented yet, but once it is working for you I will also document it and we will consider adding it to the web UI.

Kind regards,
Chris
Other remarks

- Querying dnssec-failed.org., with and without the CD bit, we observed that only 26% of the resolvers were validating the data.
- Additionally, we saw no substantial differences on the resolving rate with probes that had more than one resolver defined.
Defining a Discovery Method

• In the best case scenario, the probe will get a proper answer from its default resolvers.

• When that fails, querying the ISP's DNS server directly helps with the issue in a considerable number of cases.

• Users can as well attempt to query public DNS servers (p.e. Google, among others)

• As a last resort possibility, do full recursion from a stub resolver.
Conclusions & wrapping up

- As with many other “new” protocols (hello IPv6), the adoption of DNSSEC is really slow.
- Until things go wrong, users do not really experiment a benefit, so they do not care.
- It is quite difficult to spot where the errors happen in each individual case.
Thanks for your attention!