Evolving the Root Zone technical checks

Kim Davies
Director, Technical Services
Overview

Why do we do test? When do we test? How do we test?

The current tests

Problems and issues we see

What to do about it?
The back story
Why do we test configurations?

Ensure broken configurations are caught

Putting the wrong delegation data in the parent affects fundamental TLD function

Check for known bad behaviour that could have negative stability issues

Top-level domains are considered critical Internet infrastructure

Configuration problems could have impact on load on the root servers

Checks for certain minimum acceptable standards

Not intended to be “best practice”, but certain reasonable minimums agreed by the community
When do we test

1. After submission of a change request, prior to other processing

2. After processing, prior to implementation

3. Prior to insertion into the root zone (by Verisign)
How do we test

All tests are fully automated

Any issues identified are reported to customer, and they are asked to remedy them

Tests are automatically repeated every few hours, or customers can force a re-test

Customers can ask to proceed despite a specific failed check by providing rationale to IANA staff

Subject matter expert internally reviews such requests to see if they make sense
How we got here

Current set of technical checks are the result of public consultation in 2006


CENTR Technical WG contributed feedback, along with several CENTR members at the time

Current set of requirements:
http://iana.org/help/nameserver-requirements

Codified into Root Zone Management System (RZMS) and support tools
Top problems we have to quickly react to

**Natural disaster** — earthquakes, floods, hurricanes etc. significantly affecting local infrastructure

**Loss of service from vendor** — business failure, last minute changes due to end of contract

**Bad key rollover** — rolled the key and forgot to update DS, automation gone wild
The current test suite
Current tests (1)

Minimum number of nameservers

Must be at least two, that don’t share IP addresses

Valid hostnames

Must comport to RFC 1123 hostnames
Current tests (2)

Nameserver Reachability

Must get a port 53 response over both TCP and UDP

Answer authoritatively

Must respond with the AA-bit set to the apex of the child zone

Network diversity

Must be in two topologically separate networks, defined as not sharing the same origin AS
Assessed through inspection of routing tables (RIPE RIS, Cymru, etc.)
Current tests (3)

**Consistency between glue and authoritative data**

IP addresses for glue in parent must match the A and AAAA records for the authoritative zone for those hosts

**Consistency between delegation and zone**

NS records set for parent must match the set at apex of child zone

**Consistency between authoritative nameservers**

NS and SOA RR sets returned by each authoritative name server must match
Current tests (4)

No truncation of referrals

Parent referrals must fit on a 512-byte packet (i.e. non-EDNS0 UDP packet limit)

Payload must fit the maximum QNAME, plus the complete NS set, plus at least 1 glue record for each supported transport

No prohibited networks

No tunnels, private address space, etc.
No open recursive name service

Must not respond with answers for domains they are not authoritative for

Same source address

Response packet must come from same host as where the query was sent
Current tests (6)

**DS record format**

Must be a valid hash for the supported types

**Matching DNSKEY**

Must have a matching DNSKEY for each DS record provided

**Validation of RRSIG**

Validate the RRSIG for the apex of the zone using the DS record
Things we’ve seen
Increasingly seeing a TLDs name server infrastructure operated by a single party

Working assumption 10 years ago is it is good practice to have at least two distinct vendors for resiliency

Pushback is often “it’s anycast, so it’s OK”

Not just seeking to protect against failure in the physical topology, but things like broken announcements and business failure

Some vendors obtain a second AS operated by same party as the first, nominally meeting diversity test

Remember, we need to consider unskilful operators that put everything in one basket

…. or having a bricked router
DS record issues

TLDs wishing to list inactive “standby” DS records

Purports to be an off-line key that would be switched in in an emergency

Can not be verified against a matching DNSKEY

Base assumption has been all root zone data can be correlated/confirmed with other data in the DNS

IANA has had invalid standby keys submitted, explicitly confirmed by TLDs as being valid, to be proven invalid afterward

DS records pointing to keys without the SEP-bit set

Validates fine, meets our rules, but is it what they really wanted to do?

Upon querying the customer, answer was “yes”

In the cases where this has been submitted, customer has been notified and decided
Serial coherency

Zones that change too quickly, and propagate too slowly, to ever see it in a fully coherent state

```
<table>
<thead>
<tr>
<th></th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>blue</td>
<td>green</td>
<td></td>
</tr>
</tbody>
</table>
```
Serial coherency

Zones that change too quickly, and propagate too slowly, to ever see it in a fully coherent state.

<table>
<thead>
<tr>
<th></th>
<th>ns1</th>
<th>ns2</th>
<th>ns3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ns1, ns2, ns3
Zones that change too quickly, and propagate too slowly, to ever see it in a fully coherent state.
Zones that change too quickly, and propagate too slowly, to ever see it in a fully coherent state
Serial coherency

Zones that change too quickly, and propagate too slowly, to ever see it in a fully coherent state.
Fragmentation

Root servers desire to provide referrals in a single UDP response, rather than fragmenting

While EDNS0 provides for greater response sizes, must assume lowest common denominator

512 bytes to fit in referral
;; QUESTION SECTION:
;no. IN NS

;; ANSWER SECTION:
no. IN NS z.nic.no.
no. IN NS njet.norid.no.
no. IN NS y.nic.no.
no. IN NS x.nic.no.
no. IN NS not.norid.no.
no. IN NS i.nic.no.
QUESTION SECTION:

 ANSWER SECTION:

```
no. IN NS
no. IN NS z.nic.no.
no. IN NS njet.norid.no.
no. IN NS y.nic.no.
no. IN NS x.nic.no.
no. IN NS not.norid.no.
no. IN NS i.nic.no.
```
;; QUESTION SECTION:
no. IN NS

;; ANSWER SECTION:
no. IN NS z.nic.no.
no. IN NS njet.norid.no.
no. IN NS y.nic.no.
no. IN NS x.nic.no.
no. IN NS not.norid.no.
no. IN NS i.nic.no.
QUESTION SECTION:
;no. IN NS

ANSWER SECTION:
no. IN NS z.nic.no.
no. IN NS njet.norid.no.
no. IN NS y.nic.no.
no. IN NS x.nic.no.
no. IN NS not.norid.no.
no. IN NS i.nic.no.
;; QUESTION SECTION:
;no. IN NS

;; ANSWER SECTION:
no. IN NS z.nic.no.
no. IN NS njet.norid.no.
no. IN NS y.nic.no.
no. IN NS x.nic.no.
no. IN NS not.norid.no.
no. IN NS i.nic.no.
<table>
<thead>
<tr>
<th>NS record payload (type, class, TTL, etc.) (12 bytes)</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>z.x nic.no.</td>
<td>A</td>
</tr>
<tr>
<td>y.nic.no.</td>
<td></td>
</tr>
<tr>
<td>x.nic.no.</td>
<td></td>
</tr>
<tr>
<td>not.norid.no.</td>
<td></td>
</tr>
<tr>
<td>i.nic.no.</td>
<td></td>
</tr>
</tbody>
</table>
;; QUESTION SECTION:
;no. IN NS

;; ANSWER SECTION:
;no. IN NS z.nic.no.
;no. IN NS njet.norid.no.
;no. IN NS y.nic.no.
;no. IN NS x.nic.no.
;no. IN NS not.norid.no.
;no. IN NS i.nic.no.
QUESTION SECTION:
:no. IN NS

ANSWER SECTION:
:no. IN NS z.nic.no.
:no. IN NS njet.norid.no.
:no. IN NS y.nic.no.
:no. IN NS x.nic.no.
:no. IN NS not.norid.no.
:no. IN NS i.nic.no.
;;; QUESTION SECTION:
:no. IN NS

;;; ANSWER SECTION:
no. IN NS z.nic.no.
no. IN NS njet.norid.no.
no. IN NS y.nic.no.
no. IN NS x.nic.no.
no. IN NS not.norid.no.
no. IN NS i.nic.no.
<table>
<thead>
<tr>
<th>Question Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>no. IN NS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Answer Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>no. IN NS z.nic.no.</td>
</tr>
<tr>
<td>no. IN NS njet.norid.no.</td>
</tr>
<tr>
<td>no. IN NS y.nic.no.</td>
</tr>
<tr>
<td>no. IN NS x.nic.no.</td>
</tr>
<tr>
<td>no. IN NS not.norid.no.</td>
</tr>
<tr>
<td>no. IN NS i.nic.no.</td>
</tr>
</tbody>
</table>
;; QUESTION SECTION:
no. IN NS

;; ANSWER SECTION:
no. IN NS z.nic.no.
no. IN NS njet.norid.no.
no. IN NS y.nic.no.
no. IN NS x.nic.no.
no. IN NS not.norid.no.
no. IN NS i.nic.no.
;; QUESTION SECTION:
;no. IN NS

;; ANSWER SECTION:
no. IN NS z.nic.no.
no. IN NS njet.norid.no.
no. IN NS y.nic.no.
no. IN NS x.nic.no.
no. IN NS not.norid.no.
no. IN NS i.nic.no.
Other feedback

Expand tests to check for protocol compliance

“ICANN should be testing and blocking [TLDs] until these blocks are removed.”

“We have ICANN checking query rates and uptimes but not protocol basics (like answering all non meta query types) prior to letting new TLDs go live. … ICANN and the TLDs should be showing leadership in this area.”

Treat IPv4 and IPv6 the same

IPv6 currently optional in IANA tests, but mandatory for gTLDs per contract
What could we do?
Remember

Checks need to accommodate all top-level domains, regardless of skill level

These checks form the only mechanism to seek a minimum level of compliance across all TLDs

Many TLDs have no SLAs and other agreements with ICANN

Some TLDs today still have their entire infrastructure sitting in a single room
Revise technical checks?

Anticipate a public comment period soliciting structured feedback, similar to 2006

Some specific ideas to consider

1. How to test for “loose coherence” in a fully automated way?
2. An improved network diversity test that copes with single origin AS?
3. What is proper expectation for DS records and standby keys?
4. Add support for more DNSSEC algorithms?
   
   … or skip testing requirement for unimplemented DNSSEC algorithm/hash types?
Introduce technical check waivers?

Identify checks that may be waived

Only a subset of checks are potential candidates for allowing a TLD to skip the particular test

Provide a mechanism for TLDs to put a waiver on file

Noting the risks and opt-out reason

Update RZMS

Skip over tests?
Make them non-blocking or skippable?

Apply for permanent waiver

Certain technical configurations will often fail our technical checks. If you have a configuration that regularly fails the technical checks, you may opt to have us automatically skip those tests. Choosing these permanent waivers should be considered carefully as enabling them can mask legitimate problems that we are trying to identify to ensure the stable operation of your domain.

Permanent waivers

- **Waive serial coherency check**
  
  Waive this requirement if your technical configuration updates the zones so regularly that the entire set is not never fully synchronised. Only registries that update their zones multiple times per minute need to consider this option. **Using this option on a zone that updates less regularly will mask problems with your zone propagation.**

- **Waive DNSKEY must match DS record**
  
  Waive this requirement if you list standby keys in the root zone which are not represented in the apex of your zone. **Using this option gives us no way of verifying your DS record is valid. Use with extreme care.**
 Improved implementation with clearer communication

System output can be obtuse/insufficient

Rewriting the whole architecture of the technical check process to support better reporting of issues identified

Clearer output via email and web

Verbose debug logging of test runs available for TLDs to access via self-service portal

Remove reliance on third-party tools (weird recursor caching bug, etc.)

Review technical issues

We have performed a number of tests on the technical configuration for the domain. The following issues have been identified. In most normal cases these are problems that need to be fixed. On occasion they may represent normal configuration, in which case you can apply for a waiver of the requirement by providing information for us to review.

Parent and child NS record sets do not match

<table>
<thead>
<tr>
<th>Proposed for parent (root zone)</th>
<th>Served by child (.xyz zone)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.ns.xyz</td>
<td>a.ns.xyz</td>
</tr>
<tr>
<td>b.ns.xyz</td>
<td>b.ns.xyz</td>
</tr>
<tr>
<td>c.ns.xyz</td>
<td>c.ns.xyz</td>
</tr>
<tr>
<td>d.ns.xyz</td>
<td>d.ns.xyz</td>
</tr>
<tr>
<td>e.ns.xyz</td>
<td>e.ns.xyz</td>
</tr>
</tbody>
</table>

Explain this issue

Next steps

Do nothing

Typically you will need to take steps to fix these issues. We will continue to re-test your configuration every hour. Once we notice the issues are fixed we will automatically begin processing the request. If these issues are not fixed by 18 August 2014 the request will automatically close.

Retest

If you have fixed these issues, we can re-test the configuration now.

Apply for waiver

If you have reviewed the test results and believe they are reporting errors that do not impact your TLD, you can apply for a waiver from ICANN staff. Our technical experts will review your explanation and made a decision whether to issue a waiver to the technical requirements.

Withdraw

If there was an error in your submission and you wish to alter the changes you have requested, you can withdraw this request and submit a new request with the revised technical parameters.
IANA can regularly perform checks on all TLDs

Notify TLDs of any new issues they identify as a courtesy

Provide link in email to easily trigger a (pre-populated?) change request

Manage notifications via self-service portal
Other ideas

RFC 7344 (CDS/CDNSKEY) support

Poll for keys, triggers invitation to create a matching change request

Skipping supplemental technical check

Has the second test become superfluous?

Can retest only if longer than $x$ days since first test

Self-service testing

Open implementation

Community requests for more testing
What’s next?

Informal feedback welcome at any time
kim.davies@icann.org

Public comment period soon
Structured feedback mechanism to provide evidence of evolution required

Technical work already underway in RZMS
Major revision underway to implement various other changes
Plan to implement new technical checking platform in that release

Thanks!