NSD
An Authoritative Nameserver
A peek under the hood of version 3

Olaf Kolkman
olaf@nlnetlabs.nl
Outline

• Background on NSD: what, when, who
• Design and Architecture: goals and description
• NSD3
• DISTEL: Regression and Performance
What Is NSD

• NSD is an authoritative only nameserver
  – High performance
  – Lean and mean
  – RFC compliant
• NSD is developed and maintained by NLnet Labs
  – Not for profit “Open Source Lab”
  – In house DNS expertise

http://www.nlnetlabs.nl/
NSD history

• Conceived in 2000
  – Convergence seen on root and TLD level towards one implementation (BIND)
  – inbreed increases the thread of eradication
  – Biological diversity improves the stability of a species

• Independent reference implementation with specific design goals
Typical NSD Use Case

provisioning

(Hidden) master

BIND

BIND

BIND

NSD
NSD users

• Used on root servers
  – k.root-servers.net, h.root-servers.net
• 19 out of the 885 TLD servers use NSD
  – According to fpdns
  – Include TLD servers for .NL, SE, AT, DK, CZ

http://www.nlnetlabs.nl/
Outline

• Background on NSD: what, when, who
• Design and Architecture: goals and discription
• NSD3
• DISTEL: Regression and Performance
Design Goals

- Conformity to the relevant DNS RFCs
  - Document interpretation in case of ambiguity
- Code diversity from other implementations
  - Written from scratch
- Authoritative server only
- Regression tested against bind8/9
  - Understanding differences
- Resilience to high load
  - To cope with DOS
More Design Goals

- Open source
  - From first public release
- Documentation
  - Operation and inside code
- Reviewed code
  - Internal review and tests
- Simplicity
  - Simple == Secure
- Reasonable Portability
  - Modern *NIX Oss (FreeBSD, Linux, Solaris, etc)
Explicit non-goals

• No caching
  – Not even to optimize for fast responses
• No slavish responsiveness
  – Be able to adapt to DOS
• No end-user “friendliness”
  – Read as “no cuddling”
  – Assume knowledge of the OS and of DNS
• No creeping featurism
  – Such as random order RR in RR set
Important features through time

• NSD 1.0.0 as a master server only
• NSD 1.0.1 support for AXFR through external tools
• NSD 2.0.0 Support for DNSSEC bis
  – Internal DB design change
• NSD 3
  – Support for IXFR
  – Improved IPC
NSD Architecture
Main features

• Pre-compile answers as much as possible and perform as little work as possible during serving
  – NSD 1 had fully compiled answers
    • Only some name compression at run-time
  – NSD 2 only compiled RR sets
    • Assembly at run-time
    • Mainly to enable support of DNSSEC
    • Small performance penalty
NSD Data

- Precompiled data stored in memory in a Red Black Tree
- Pointer structure to fetch all pieces
- Additional data structures to perform “accounting”
NSD operation model (version 1 and 2)
NSD 2 Operational Features

- Requires ‘cron’ and manual control for ingress zone transfers
- .NL zone signed with 1024big ZSK

<table>
<thead>
<tr>
<th></th>
<th>unsigned</th>
<th>signed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB file</td>
<td>46</td>
<td>251</td>
</tr>
<tr>
<td>Core</td>
<td>109</td>
<td>388</td>
</tr>
</tbody>
</table>
Outline

• Background on NSD: what, when, who
• Design and Architecture: goals and description
• NSD3
• DISTEL: Regression and Performance
NSD 3
Wish list

- Incremental update support
  - Full zone network transport and recompilation is expensive
- Cronjob triggered AXFR does not really support SOA timings
- DNAME support
  - Recent ICANN announcement w.r.t. testing IDN support in the root
- NSEC3 support
NSD3 Architecture

The processes
NSD3 Architecture
IXFR/AXFR

- NSD parent
- NSD child
- namedb
- nsd.diff
- notify
- xfrd
- ixfr
NSD3 Architecture

Reload

Copy-on-write minimizes memory overhead
NSD3 Architecture
Reload

[Diagram showing the relationship between namedb, NSD child, NSD parent, reloader, xfrd, and nsd.diff]
Reload process kills all, restarts new children that can access the shared memory.
NSD3 Architecture

Reload

Merge diffs into zonefile using a patch utility
Recompile using zonec
All happens outside the nsd processes

http://www.nlnetlabs.nl/
NSD3 Architecture
Possible DOS handling

Rate limited process takes all ‘difficult traffic’
(e.g. NSEC3 calculation)
Remember: its work in progress

• Rate limiting by moving all data over IPC might be more expensive than handling the packet by the clients directly
  – Performance measurements will help us decide

• Other details may also be subject to change
NSD3 timeline

• Expect first public (beta) release by end of Q2
Outline

• Background on NSD: what, when, who
• Design and Architecture: goals and description
• NSD3
• DISTEL: Regression and Performance
Distel Testlab

• Using production zones and real-time query load

• Performance
  – Replaying traces in real time, accelerated and delayed

• Regression
  – Understanding differences with various implementations
The “DISTEL” Test Lab
DISTEL properties

- Player plays libpcap traces in real time
  - libpcap traces are modified to have the servers destination address
  - Needed modified tcpreplay to get to ms timing precision
- Server has a default route to the recorder
- Recorder captures answers
- 2 Ghz Athlon based hardware with 1 Gb memory and 100baseT Ethernet

http://www.nlnetlabs.nl/
DISTEL shortcoming

• DISTEL only reports features that are present in a zone and are triggered by provided queries
  – We perform separate tests, but may miss corner cases
  – It happened before and it will happen again

• You can help provide zone content and query traces
  – High volume traces, zone content you had problems with in other implementations
  – Useful for regression testing
Some Test Results

• Using a query trace captured from k.root-servers.net againsts the test server configured as k.root-server.net
  – NB: not the same hardware specs as the “real” thing
• Comparing unsigned, signed and worse case
  – Number of DO bits set in the query streams
• Read RIPE 352 for more details
NSD 2.3.3 performance (root-zone, k-root traces)
NSD 2.3.3 effect of signed zones (root-zone, k-root traces)
Trace k.root against modified nsd 2.3.0

Bandwidth Increase

ZSK size
unsigned
0512
0768
1024
1280
1536
1792
2048

Upper Bound

http://www.nlnetlabs.nl/
How can you help?

• Provide zone content and query traces
  – High volume traces, zone content you had problems with in other implementations
  – Useful for regression testing

• Use the program
  – Report bugs, omissions in documentation, etc
Support

• Guaranteed for at least two years
  – Community support
  – Two year advance notice before support is stopped

• We will also offer support contracts in the near future