Modifying NSD for DNSSEC: Design, Implementation, Performance

Erik Rozendaal <erik@nlnetlabs.nl>
NLnet Labs
What is NSD?

• NSD is an RFC compliant, authoritative only name server:
  - Simple
  - High Performance

• Adding DNSSEC support was not hard, but required some fundamental changes to NSD:
  - NSD 1.x pre-encodes all possible answers using the zone compiler.
  - NSD 2.x pre-encodes all RRsets and encodes answers at run-time.
Advantages of NSD 1.x approach

• Complexity moved from server to zone compiler:
  ■ Determining RRsets to be included in the answer
  ■ Pre-compute name compression

• High performance, simple server algorithm:
  ■ Analyze query
  ■ Find answer in database
  ■ Update compression pointers
  ■ Send answer to client
Advantages of NSD 2.x approach

- Smaller database and less memory usage:
  - .nl database size: From 126 to 46 Megabytes
  - .nl memory usage: From 155 to 109 Megabytes

- More flexibility in determining contents of answer based on query.
  - Important for DNSSEC.

- Lower overall complexity.
Why modify the pre-encoding of answers for DNSSEC

• DNSSEC increases the zone size, mainly due to the presence of NSEC and RRSIG.
  - ~5 times for .nl signed with a single 1024-bit RSA key.

• DNSSEC requires additional answers to be stored in the database:
  - DO bit set/not set
  - Answers for DS, NSEC, and RRSIG queries
  - Answers for NXDOMAIN and NODATA responses
Estimated answer database size

- Assuming answers grow ~5 times in size and we need to store ~2 - ~4 times more answers.

- Estimate: database size increases ~10 - ~20 times.
  - .nl database from 126 Megabytes to ~1.2 - ~2.5 Gigabytes.

- Runs into 32-bit memory limit.
NSD 2.0.0 DNSSEC Memory Usage

- .nl zone signed with a single 1024-bit RSA key.

- .nl database size:
  - Unsigned: 46 Megabytes
  - Signed: 251 Megabytes

- .nl memory usage:
  - Unsigned: 109 Megabytes
  - Signed: 388 Megabytes
Performance Comparison

• NSD 2.x server algorithm:
  ▪ Analyze query
  ▪ Lookup information about query name
  ▪ Determine RRsets to include
  ▪ Encode RRsets and perform name compression
  ▪ Send answer to client

• What happened to performance?
  ▪ Slightly less, but still very fast.
Performance Comparison (continued)

- DNS servers: bind 8.4.4, bind 9.2.3, nsd 1.2.4, nsd 2.0.0

- "echo" server: modified nsd 2.0.0 that simply echoes the query back to the client
  - Used to measure network and OS overhead

- Server hardware:
  - Off the shelf AMD Athlon XP 2400+ PC
  - 1 Gigabyte main memory
  - 3COM 3C905B-TX Fast Etherlink 10/100 PCI TX NIC
Performance Comparison: k.root-servers.net

- bind 8.4.4
- bind 9.2.3
- nsd 1.2.4
- nsd 2.0.0
- echo

Graph showing the percentage of queries answered versus the average number of queries per second for different software versions.
Performance Comparison: .nl

![Graph showing performance comparison of different DNS servers.]

- **bind 8.4.4**
- **bind 9.2.3**
- **nsd 1.2.4**
- **nsd 2.0.0**
- **echo**
NSD 2.0.0 DNSSEC Performance: k.root-servers.net

Assume DO set

% of Queries Answered

Average Queries to k.root-servers.net per Second

Unsigned root
Signed root
Assume DO set
echo
NSD 2.0.0 DNSSEC Performance: .nl

% of Queries Answered

Average Queries to .nl per Second

UnSigned .nl
Signed .nl
Assume DO set
echo
Conclusion and Next Steps

• Conclusion: NSD 2.0.0 performs very well, with or without DNSSEC.

• Next Steps: Release NSD 2.0.0 next month with DNSSEC disabled by default.

• Next Steps: Release NSD 2.x with DNSSEC enabled as soon as DNSSEC is standardized.

• Wanted: Complicated zones and tcpdump query traces to perform more regression testing against bind.
Questions?

- Questions?