RIPE NCC's DNSSEC Deployment

Olaf M. Kolkman

Olaf@NLnetLabs.nl

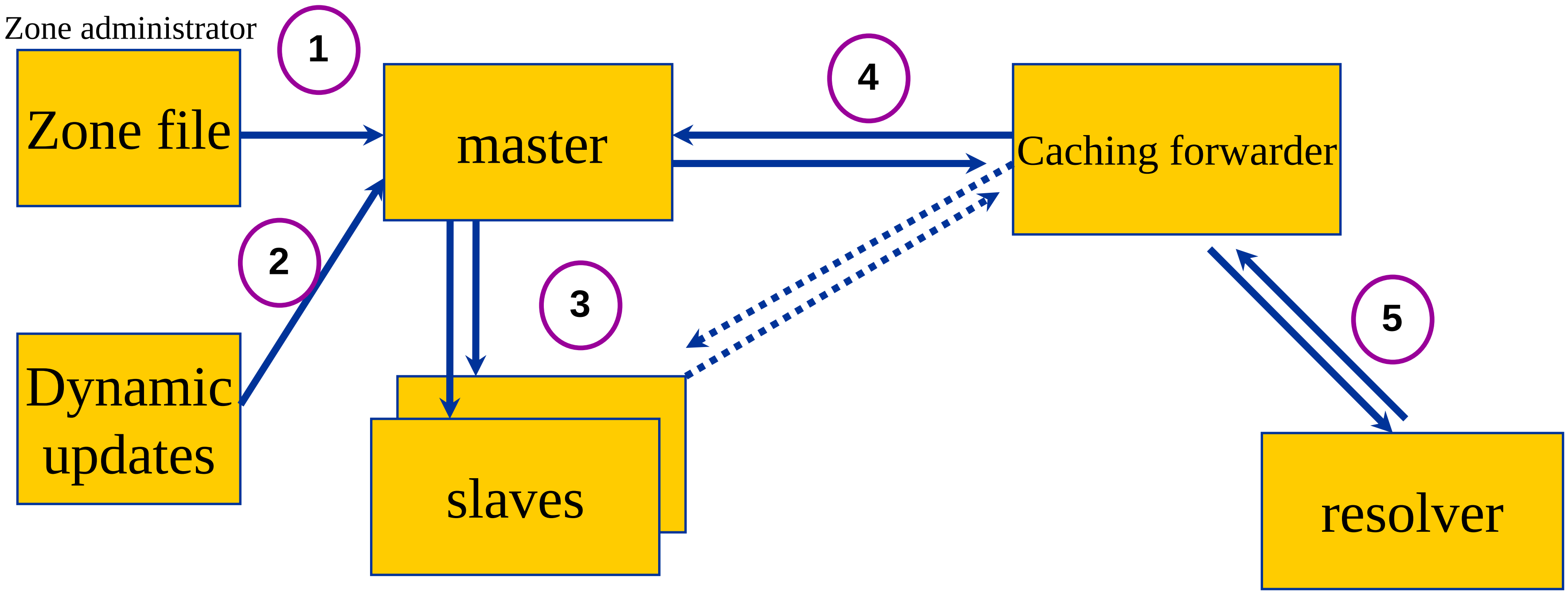
Katie Petrusha, Brett Carr, Cagri Coltekin, Adrian Bedford, Arno Meulenkamp, and Henk Uijterwaal

To avoid confusion: I am employed by NLnet Labs. Presenting on behalf of RIPE NCC

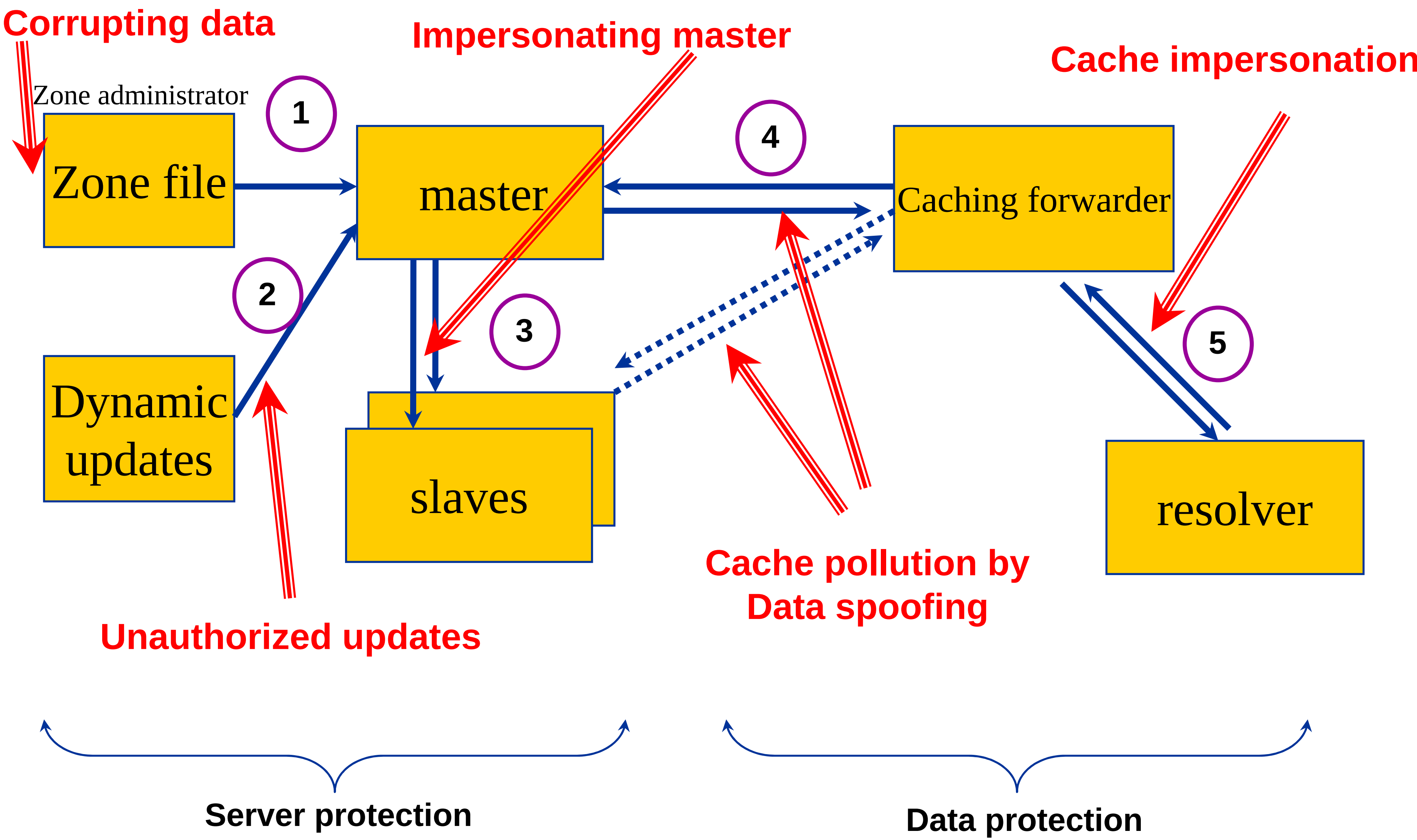
Presentation roadmap

- Overview of problem space
  - DNSSEC in 3 slides
  - Architectural changes to allow for DNSSEC deployment
- Deployment tasks
  - Key maintenance
  - DNS infrastructure
  - Providing secure delegations

DNS: Data Flow



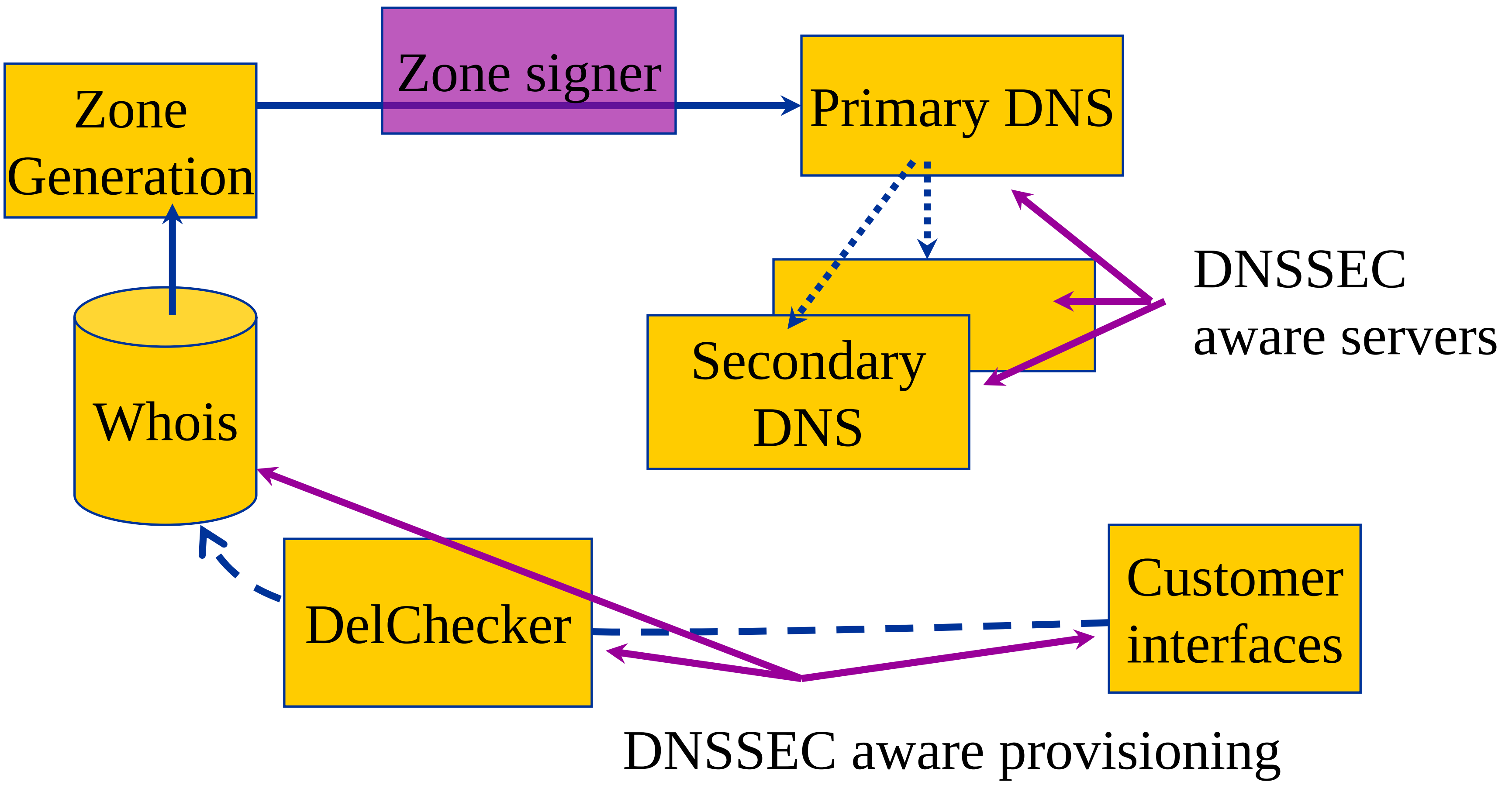
DNS Vulnerabilities



DNSSEC

- Provides data authentication based on public key cryptography
  - Resolver can verify that what went in came out
  - Digitial signatures are validated using public keys
    - RRSIG and DNSKEY Resource Records
  - Chains of trust are build using the DNS
    - DS Resource Record
    - A pointer from parent to child

DNSSEC
Architecture modifications

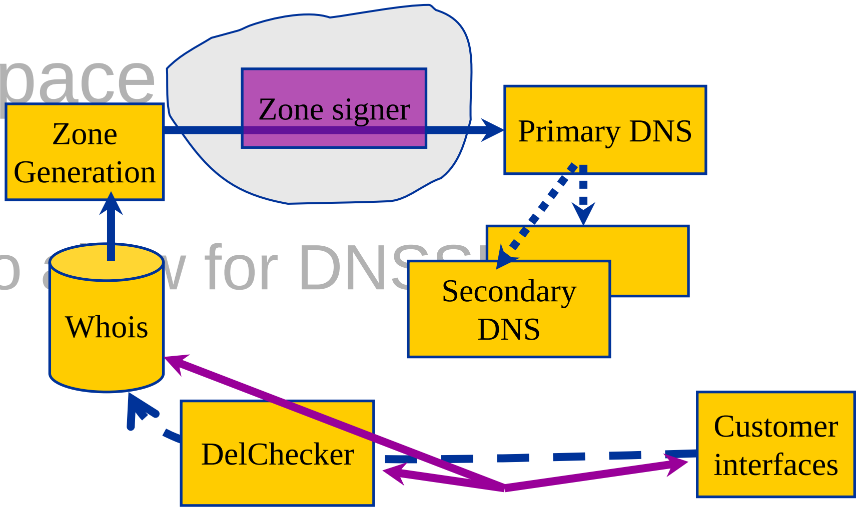


DNSSEC deployment tasks

- Key maintenance policies and tools
  - Private Key use and protection
  - Public key distribution
- Zone signing and integration into the provisioning chain
- DNS server infrastructure
- Secure delegation registry changes
  - Interfacing with customers

Presentation roadmap

- Overview of problem space
  - DNSSEC in 3 slides
  - Architectural changes to allow for DNSSEC deployment
- Deployment tasks
  - Key maintenance
  - DNS server infrastructure
  - Providing secure delegations



DNSSEC aware provisioning

Key Maintenance

- DNSSEC is based on public key cryptography
  - Data is signed using a private key
  - It is validated using a public key

Operational problems:

- Dissemination of the public key
- Private key has a ‘*best before*’ date
  - Keys change, and the change has to disseminate

Public Key Dissemination

- In theory only one trust-anchor needed that of the root
  - How does the root key get to the end user?
  - How is it rolled?
- In absence of hierarchy there will be many trust-anchors
  - How do these get to the end-users?
  - How are these rolled?

- These are open questions, making early deployment difficult.

Public Key Dissemination at RIPE NCC

In absence of a signed parent zone and automatic rollover:

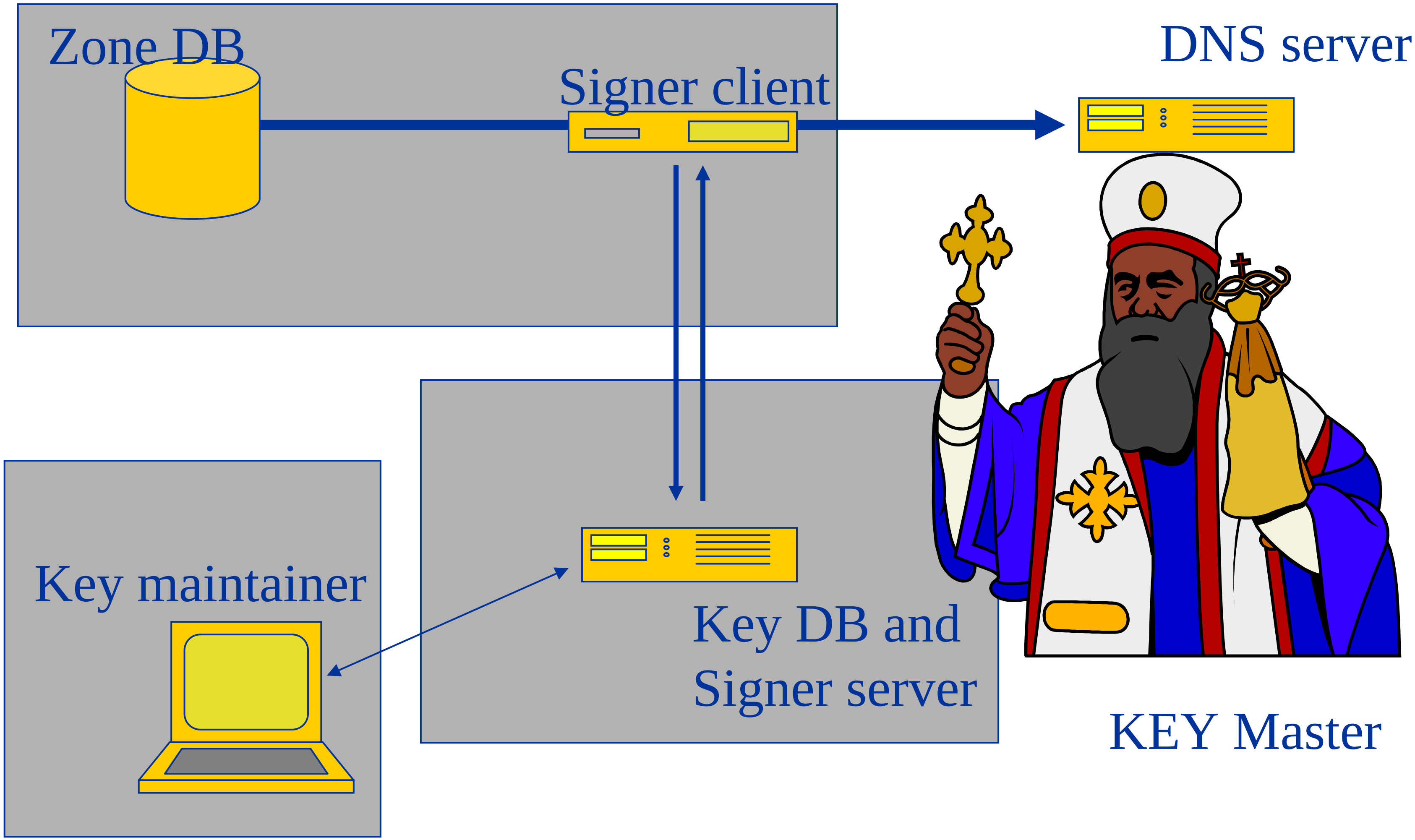
- Trust anchors are published on an “HTTPS” secured website
- Trust anchors are signed with the RIPE NCC public keys
- Trust anchor will be rolled twice a year (during early deployment)
- Announcements and publications are always signed by x.509 or PGP

Key Management

- There are many keys to maintain
  - Keys are used on a per zone basis
    - Key Signing Keys and Zone Signing Keys
  - During key rollovers there are multiple keys
    - In order to maintain consistency with cached DNS data [draft-ietf-dnsop-dnssec-operational-practices]

- Private keys need shielding

Private Key Maintenance Basic Architecture



Maintaining Keys and Signing Zones

- The KeyDB maintains the private keys
  - It ‘knows’ rollover scenarios
  - UI that can create, delete, roll keys without access to the key material
  - Physically secured
- The signer ties the Key DB to a zone
  - Inserts the appropriate DNSKEYs
  - Signs the the zone with appropriate keys
- Strong authentication

QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

Private Key Maintenance
The software

- Perl front-end to the BIND dnssec-signzone and dnssec-keygen tools
- Key pairs are kept on disc in the “BIND format”
- Attribute files containing human readable information
  - One can always bail out and sign by hand.

- Works in the RIPE NCC environment, is a little rough edged but available via the www.ripe.net/disi

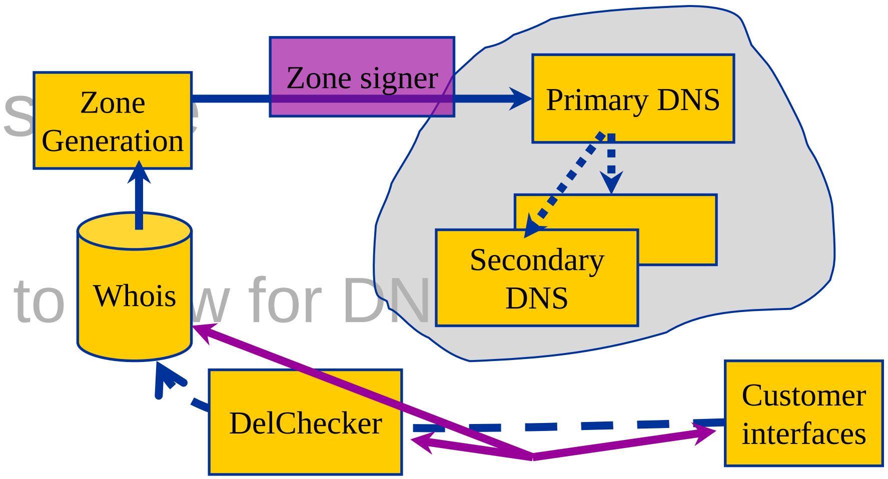
Example session

```
$ maintkeydb create KSK RSASHA1 2048 example.net
        Created 1 key for example.net
$ maintkeydb create ZSK RSASHA1 1024 example.net
        Created 2 keys for example.net
$ dnssigner example.net
        Output written to :example.net.signed
```

```
$ maintkeydb rollover zsk-stage1 RSASHA1 example.net
```

Presentation roadmap

- Overview of problem s[illegible]e
  - DNSSEC in 3 slides
  - Architectural changes to [illegible]w for DN[illegible] deployment
- **Deployment tasks**
  - Key maintenance
  - **DNS server infrastructure**
  - Providing secure delegations



DNSSEC aware provisioning

Infrastructure

- One needs primary and secondary servers to be DNSSEC protocol aware

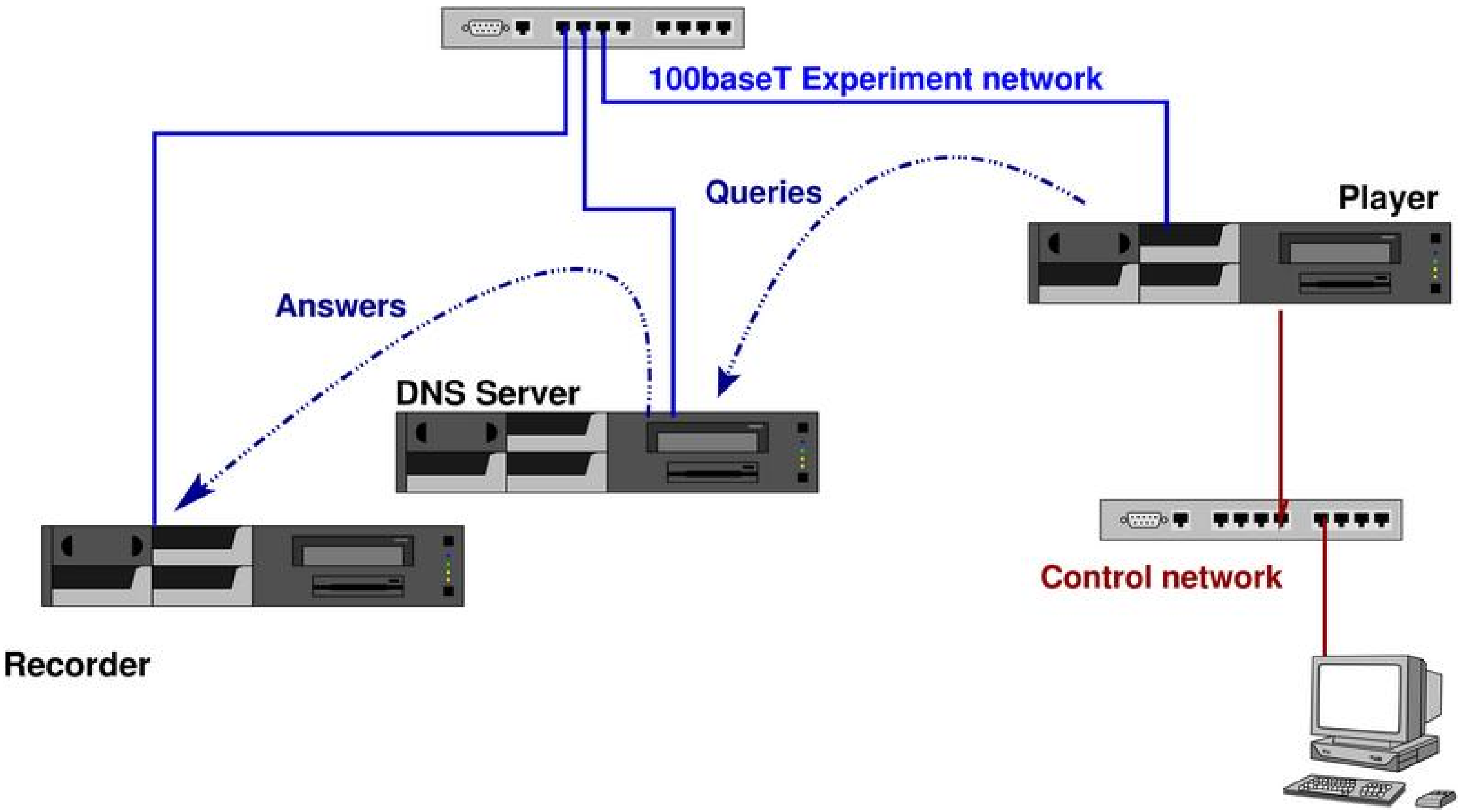
- We had a number of concerns about memory CPU and network load
  - Research done and published as RIPE 352
  - What follows are the highlights of that paper

Question

What would be the immediate and initial effect on memory, CPU and bandwidth resources if we were to deploy DNSSEC on RIPE NCC's 'primary' name server?

- Measure through simulation.

The "DISTEL" Test Lab



DISTEL LAB

- Player plays libpcap traces in real time
  - libpcap traces are modified to have the servers destination address
- Server has a default route to the recorder
- Recorder captures answers

- 2 Ghz Athlon based hardware with 1 Gb memory and 100baseT Ethernet

This Experiment

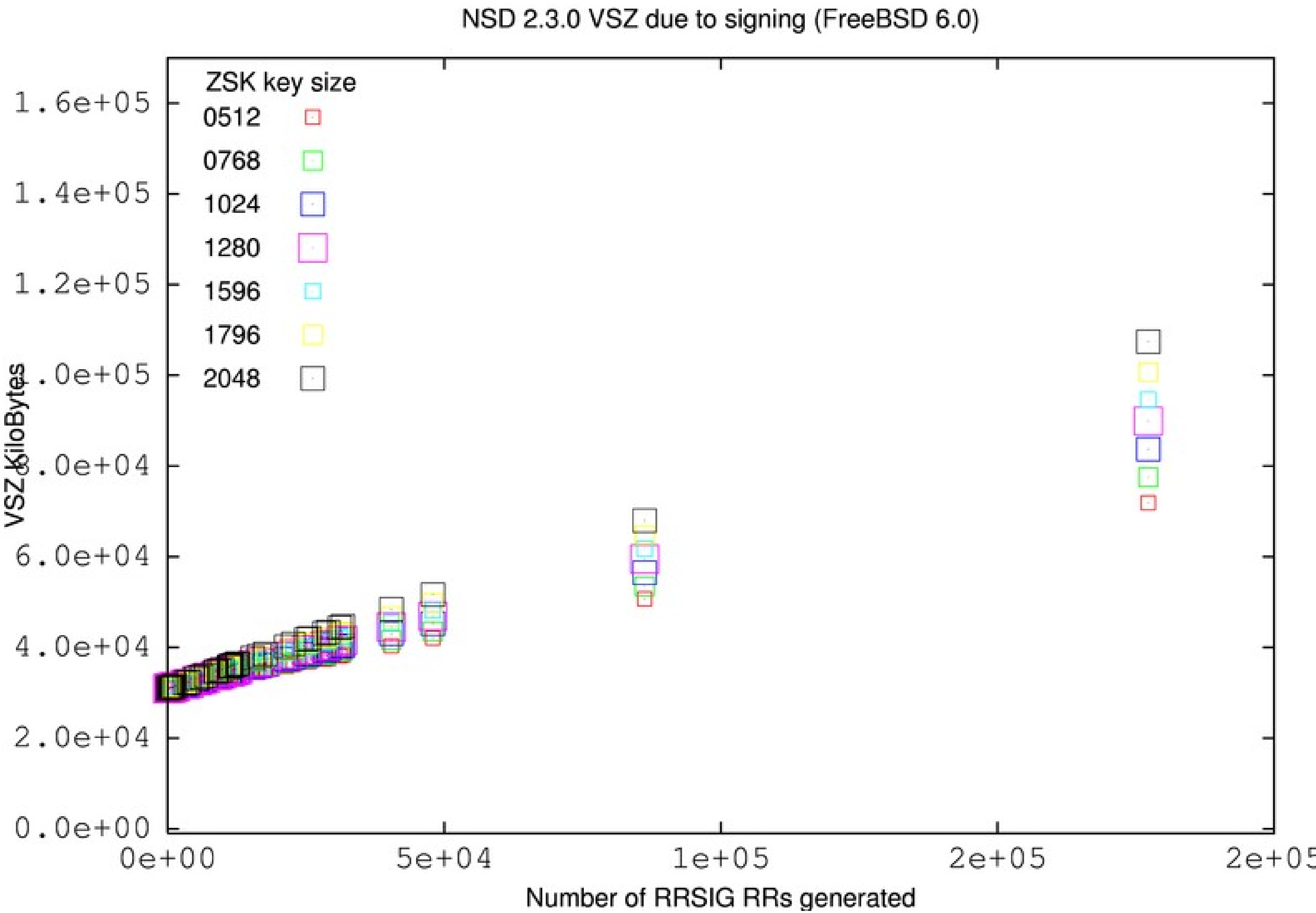
- Traces from production servers:
  - k.root-servers.net
  - ns-pri.ripe.net
- Server configured to simulate the production machines.
  - ns-pri.ripe.net
    - Loaded with all 133 zones.
  - k.root-servers.net
    - Only loaded with the root zone.

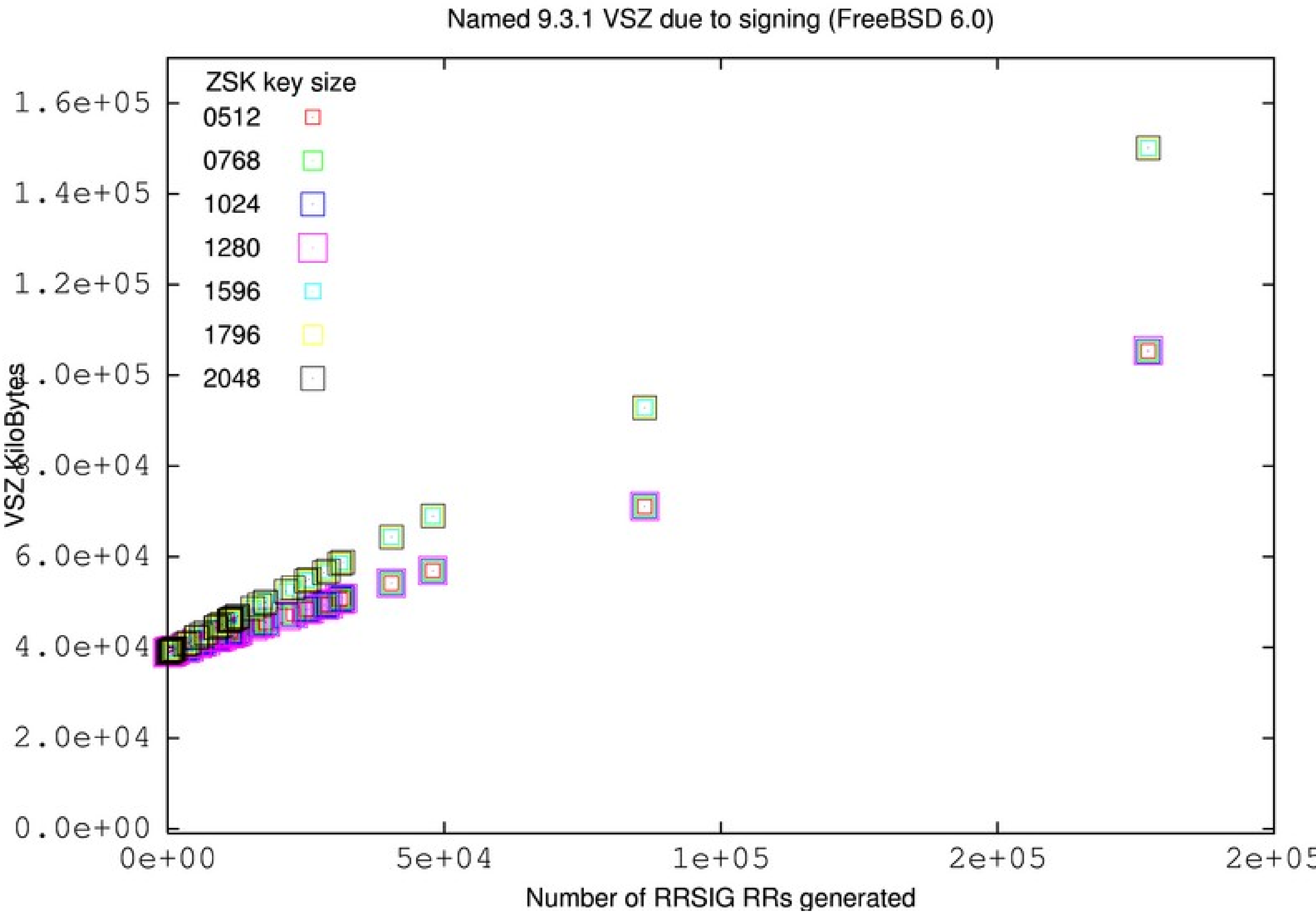
Zone Signing

- 1 Key Signing Key 2048 bit RSASHA1
- 2 Zone Signing Keys of equal length
  - length varied between 512 and 2048
  - Only one ZSK used for signing
    - This is expected to be a common situation (Pre-publish KSK rollover)
- 3 DNSKEY RRs in per zone
  - 1 RRSIG per RR set
  - 2 RRSIGs over the DNSKEY RR set

Loading the Zones: Memory Use

- Various zone configurations were loaded.
  - Mixtures of signed and unsigned zones
  - Memory load for different numbers of RRSIGs and NSECs.
- Memory load is implementation and OS specific


NSD 2.3.0 VSZ due to signing (FreeBSD 6.0)
ZSK key size
0512
0768
1024
1280
1596
1796
2048
1.6e+05
1.4e+05
1.2e+05
1.0e+05
8.0e+04
6.0e+04
4.0e+04
2.0e+04
0.0e+00
VSZ KiloBytes
0e+00
5e+04
1e+05
2e+05
2e+05
Number of RRSIG RRs generated


Named 9.3.1 VSZ due to signing (FreeBSD 6.0)
ZSK key size
0512
0768
1024
1280
1596
1796
2048
1.6e+05
1.4e+05
1.2e+05
1.0e+05
8.0e+04
6.0e+04
4.0e+04
2.0e+04
0.0e+00
VSZ KiloBytes
0e+00
5e+04
1e+05
2e+05
2e+05
Number of RRSIG RRs generated

Memory

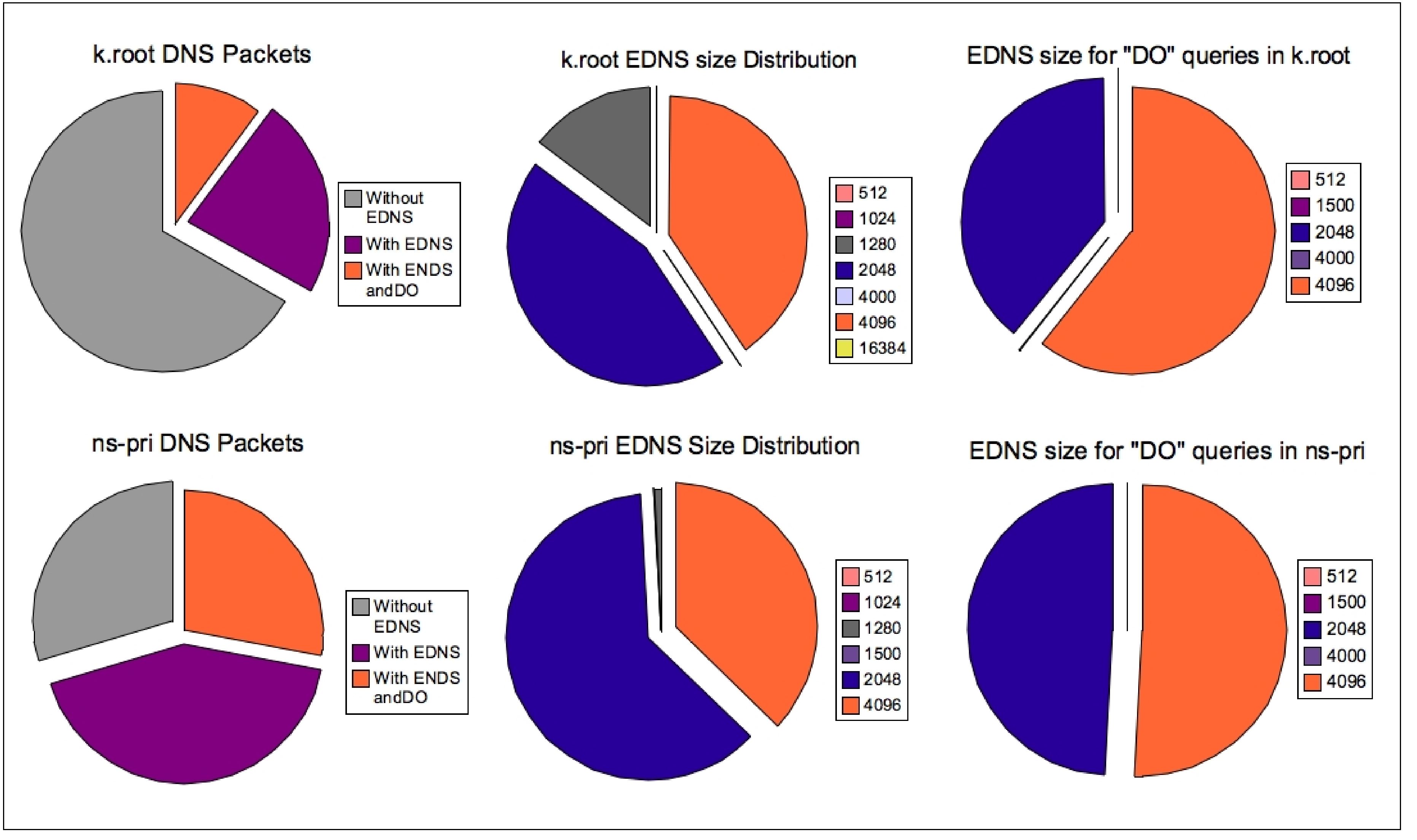
- On ns-pri.ripe.net factor 4 increase.
  - From ca. 30MB to 150MB
  - No problem for a 1GB of memory machine
- On k.root-servers.net
  - Increase by ca 150KB
  - Total footprint 4.4 MB
- Nothing to worry about
- Memory consumption on authoritative servers can be calculated in advance.
  - No surprises necessary

Serving the zones
Query Properties

- DNS clients set the “DO” flag and request for DNSSEC data.
  - Not to do their own validation but to cache the DNSSEC data for.
- EDNS size determines maximum packet size.

  (DNSSEC requires EDNS)
- EDNS/DO properties determine which fraction of the replies contain DNSSEC information

EDNS properties



Serving the zones

- Measured for different keysizes.
  - named for ns-pri.ripe.net
  - nsd and named for ns-pri.ripe.net and k.root-servers.net

- We also wanted to study “worst case”; *What if all queries would have the DO bit set?*
  - Modified the servers to think that queries had EDNS 2048 octets size and DO bit set

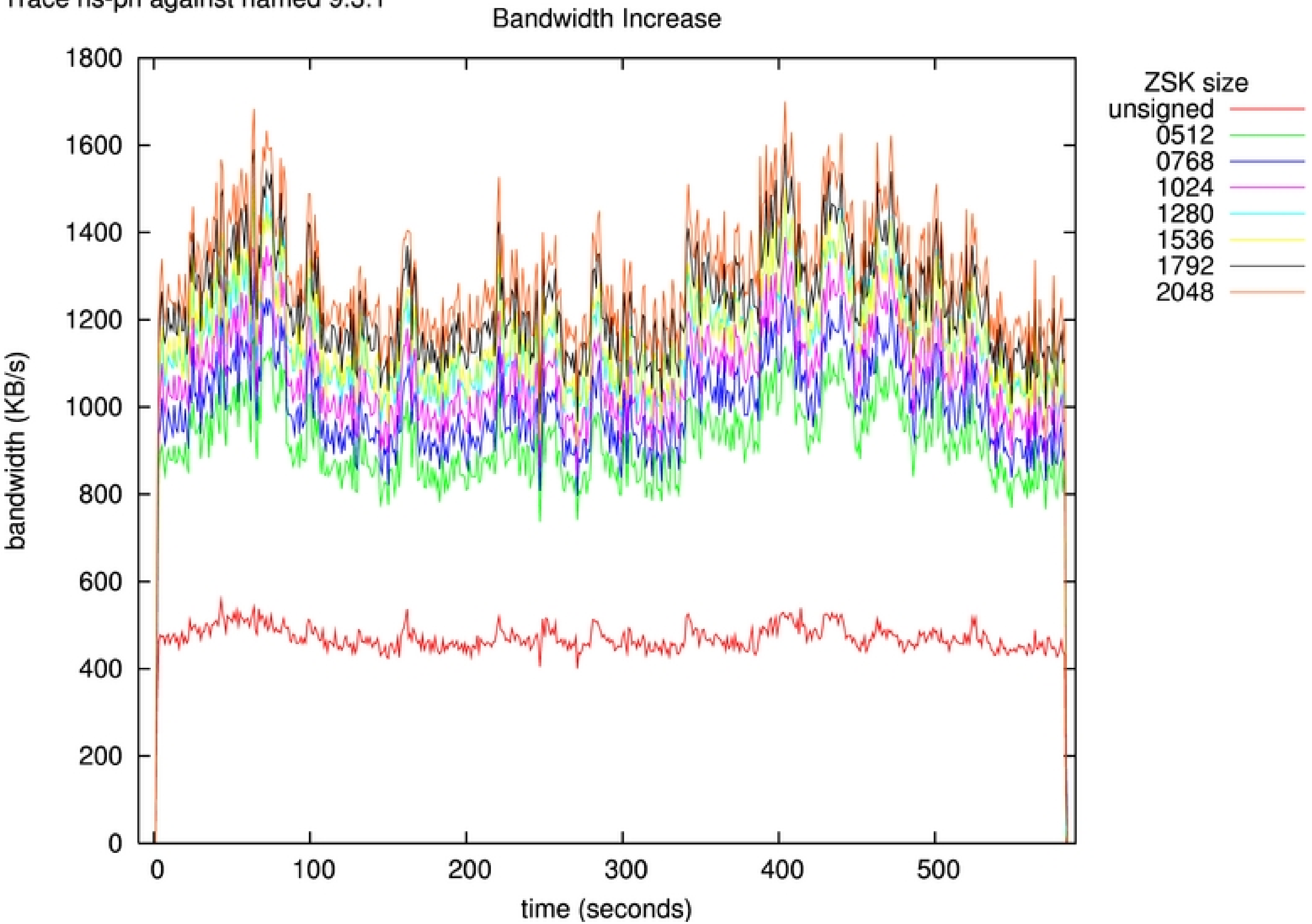
CPU

trace	server		ZSK size	WCPU
ns-pri	BIND 9.3.1		0000	ca 14%
ns-pri	BIND 9.3.1		2048	ca 18%
k.root	BIND 9.3.1		0000	ca 38%
k.root	BIND 9.3.1		2048	ca 42%
k.root	BIND 9.3.1	mod	2048	ca 50%
k.root	NSD 2.3.0		0000	ca 4%
k.root	NSD 2.3.0		2048	ca 4%
k.root	NSD 2.3.0	mod	2048	ca 5%

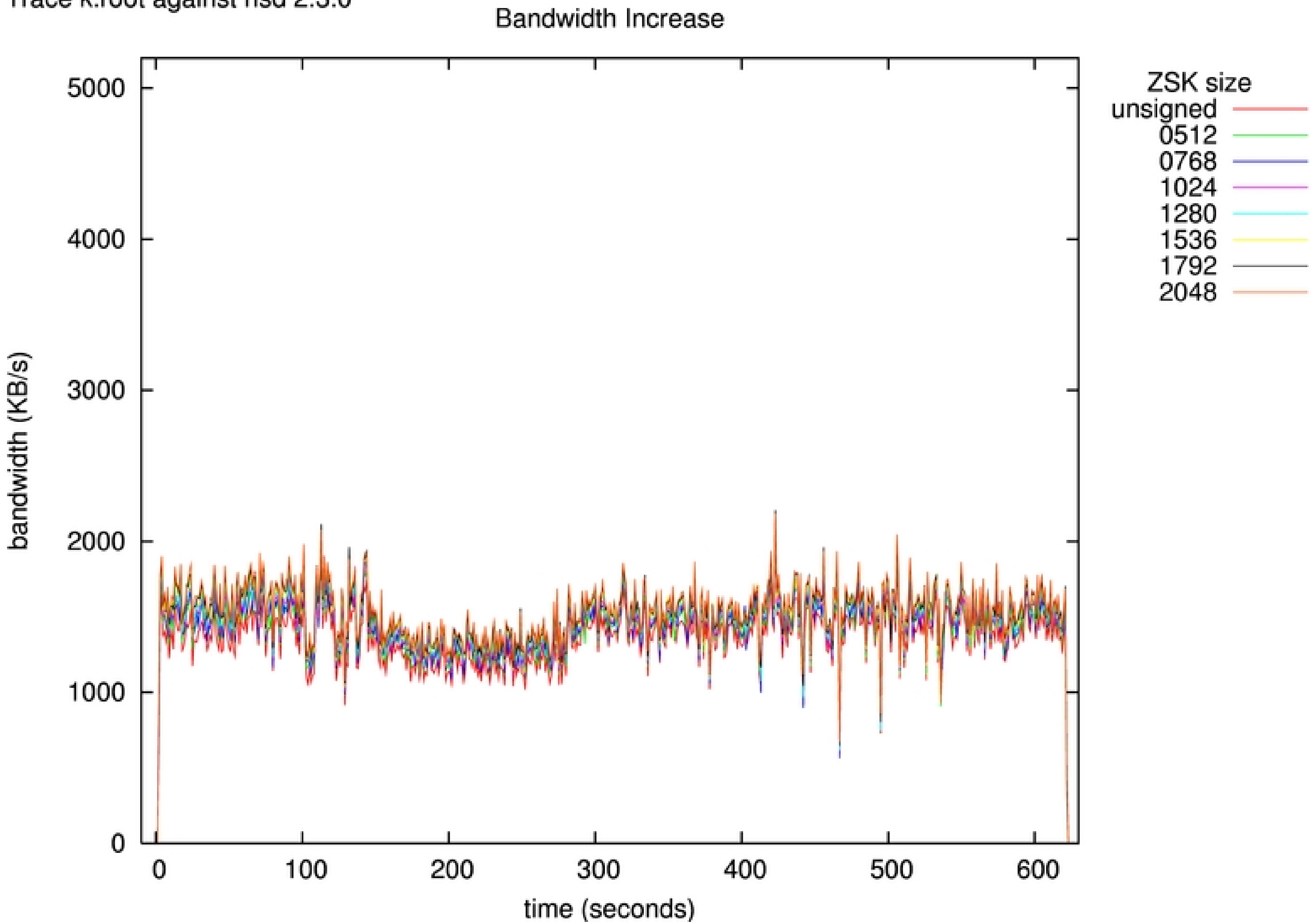
Bandwidth Factors

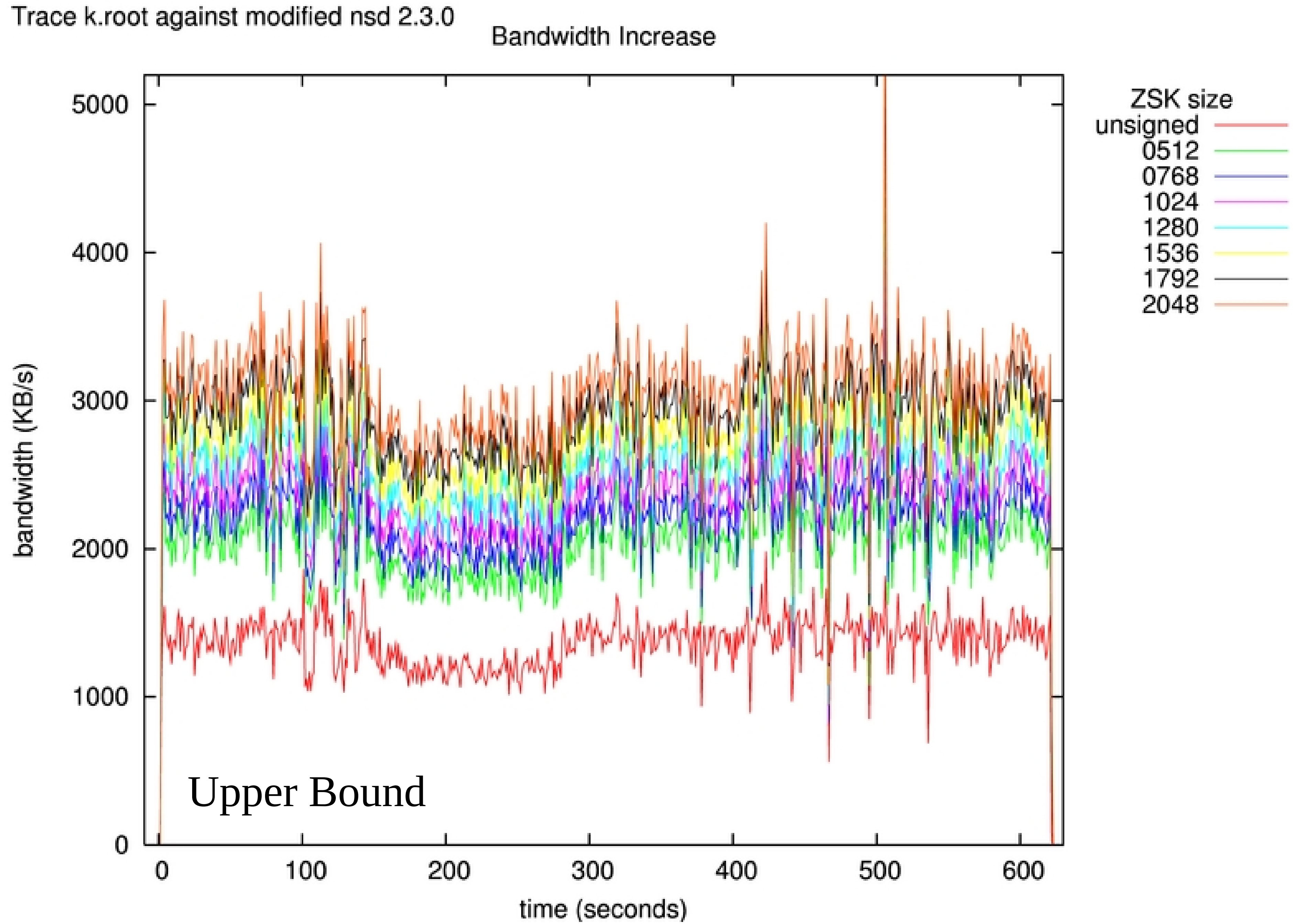
- fraction of queries with DO bit
  - Seen in difference between ns-pri and k.root result
  - Seen in difference between modified and unmodified servers
- Including DNSKEY RR in additional section.
  - Seen in difference between k.root traces from modified nsd and modified named
- Difference in answer patterns
  - Name Errors vs Positive answers
  - Difficult to asses from this data

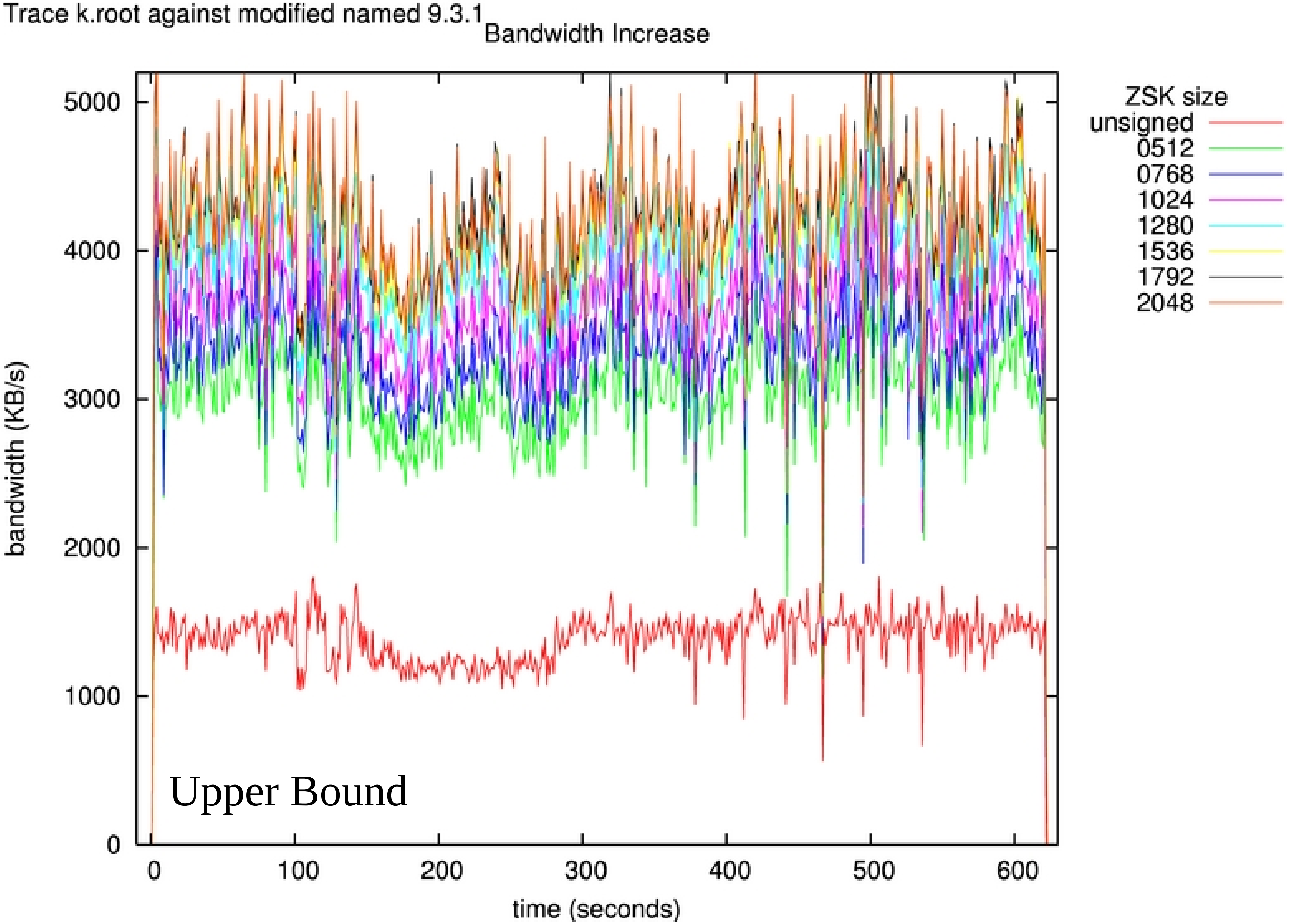
Trace ns-pri against named 9.3.1



Trace k.root against nsd 2.3.0




Trace k.root against modified nsd 2.3.0
Bandwidth Increase
bandwidth (KB/s)
0
1000
2000
3000
4000
5000
time (seconds)
0
100
200
300
400
500
600
ZSK size
unsigned
0512
0768
1024
1280
1536
1792
2048
Upper Bound


Trace k.root against modified named 9.3.1
Bandwidth Increase
bandwidth (KB/s)
5000
4000
3000
2000
1000
0
0
100
200
300
400
500
600
time (seconds)
ZSK size
unsigned
0512
0768
1024
1280
1536
1792
2048
Upper Bound

Bandwidth observation

- DNSKEY RR set with RRSIG in the additional section
  - Fairly big chunk of data
  - None of the clients today validate the data
  - Clients that need the data will query for it
- Servers MAY include the DNSKEY Rrset
- NSD does not include
- Named does include
  - Recommendation to make the inclusion configurable

Bandwidth Increase

- Significant for ns-pri.ripe.net
  - Well within provisioned specs.
- Insignificant for for k.root-servers.net
  - Upper bound well within provisioning specs
    - even when including DNSKEY RR set in additional section

(Key size influences bandwidth but bandwidth should not influence your key size)

Not Measured

- The experiment has been done in a closed environment
- What about the behavior of clients that do expect DNSSEC information but do no not receive it?
  - Firewalls dropping packets with DNSSEC
  - BIND behavior is well understood
- What about implementations that set the DO bit but cannot handle DNSSEC data that is returned?
- Measure these on the Internet

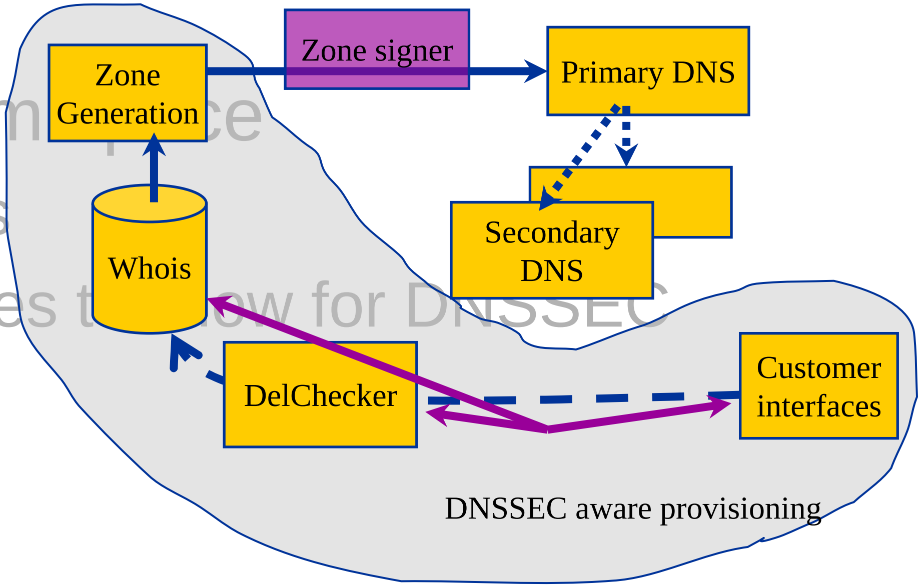
Conclusion of these measurements

- CPU, Memory and Bandwidth usage increase are not prohibitive for deployment of DNSSEC on k.root-servers.net and ns-pri.ripe.net

- Bandwidth increase is caused by many factors
  - Hard to predict but fraction of DO bits in the queries is an important factor
- CPU impact is small, Memory impact can be calculated
- Don’t add DNSKEY RR set in additional

Presentation roadmap

- Overview of problem space
  - DNSSEC in 3 slides
  - Architectural changes to allow for DNSSEC deployment
- Deployment tasks
  - Key maintenance
  - DNS server infrastructure
  - Providing secure delegations



Parent-Child Key Exchange

- In the DNS the parent signs the "Delegations Signer" RR
  - A pointer to the next key in the chain of trust

```
$ORIGIN net.

kids NS    ns1.kids
     DS   (…) 1234
     RRSIG DS (…)net.

money NS    ns1.money
      DS    (…)
      RRSIG DS (…)net.
```

```
$ORIGIN kids.net.

@ NS    ns1
  RRSIG NS (…) kids.net.
  DNSKEY (…)  (1234)
  DNSKEY (…)  (3456)
  RRSIG dnskey … 1234 kids.net.
  RRSIG dnskey … 3456 kids.net.

beth   A   127.0.10.1
       RRSIG A (…) 3456 kids.net.
```

- DNSKEY or DS RR needs to be exchanged between parent and child

Underlying Ideas

- The DS exchange is the same process as the NS exchange
  - Same authentication/authorization model
  - Same vulnerabilities
  - More sensitive to mistakes
- Integrate the key exchange into existing interfaces
  - Customers are used to those
- Include checks on configuration errors
  - DNSSEC is picky
- Provide tools
  - To prevent errors and guide customers

How Did we Proceed

- The `ds-rdata:` attribute was added to the Domain object
- The zone generation tool:extract DS RRs from `ds-rdata:` attributes
- We introduced a filter, to block `ds-rdata:`for zones not yet signed
- Added a number of “DelChecker” checks

Intergration issue

- Thinking about DNSSEC made the NCC look at the provisioning system as a whole
  - Prompted a couple of modifications
  - Zone generation (generation of zone now from the Whois DB)
  - Authentication model (introduction of mnt-domain)
  - Possible replay attacks (countered by using timestamps of the strong auth. mechanisms)
- All these issues are NOT DNSSEC specific
- Addressed over the last 2 years

Introducing the Web Interface

- Eases registration of keys and the rollovers
  - Can also be used for “classic” delegations
- Restricts user somewhat
  - Fewer degrees of freedom mean fewer errors
  - One can always manually create the Domain object
- Version 1 to appear shortly
  - Demo in the hallway

Web Interface Examples

May the Demo Gods be with us.

We’ll cheat.

Roadmap

- RIPE NCC is signing its zones
- Policy last call ends this week
- Signed /8 in-addr zones will be introduced
  – Starting 19 October, Finished by early 2006
  – Details reported in the DNS WG
- Secure Delegations will be possible shortly after such /8 has been signed

Questions and Discussion