# DNSSEC Deployment a case study

Olaf M. Kolkman

Olaf@NLnetLabs.nl

RIPE NCCs Project Team:

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






NLnet Labs

# Presentation roadmap

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

# Why DNSSEC

- Good security is multi-leveled
  - Multiple defense rings in physical secured systems

Bourtange, source Wikipedia

# Why DNSSEC

- Good security is multi-leveled
  - Multiple defense rings in physical secured systems
  - Multiple ‘layers’ in the networking world
- DNS infrastructure
  - Providing DNSSEC to raise the barrier for DNS based attacks
  - Provides a security ‘ring’ around many systems and applications

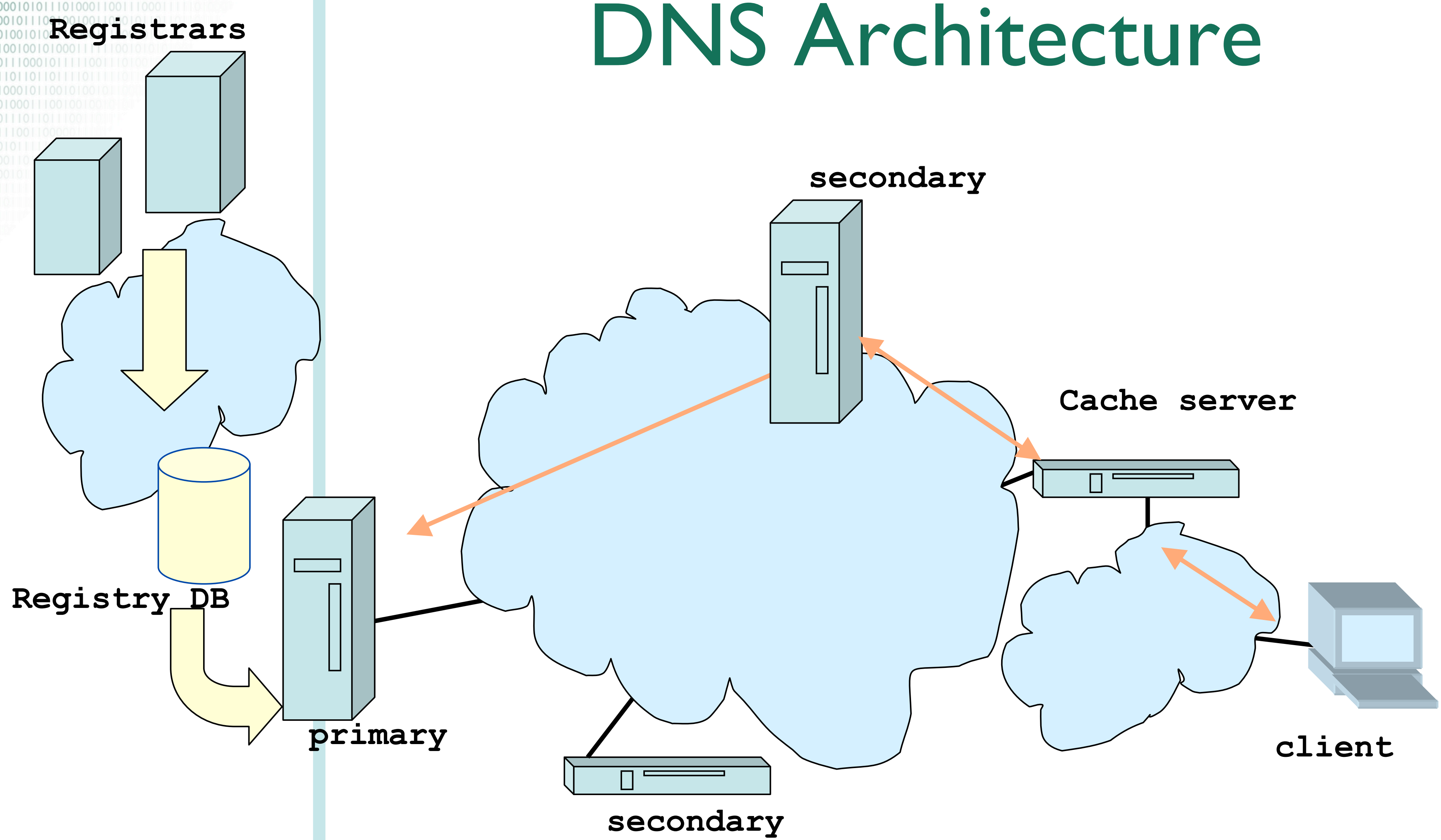
# DNSSEC evangineer of the day

- NLnet Labs (www.NLnetLabs.nl)
  - Not for profit Open Source Software lab
    - NSD, open source nameserver
  - DNS and DNSSEC research
    - Protocol and software development
    - Deployment engineering
- Co-Chair of the IETF DNSEXT working group

# The Problem

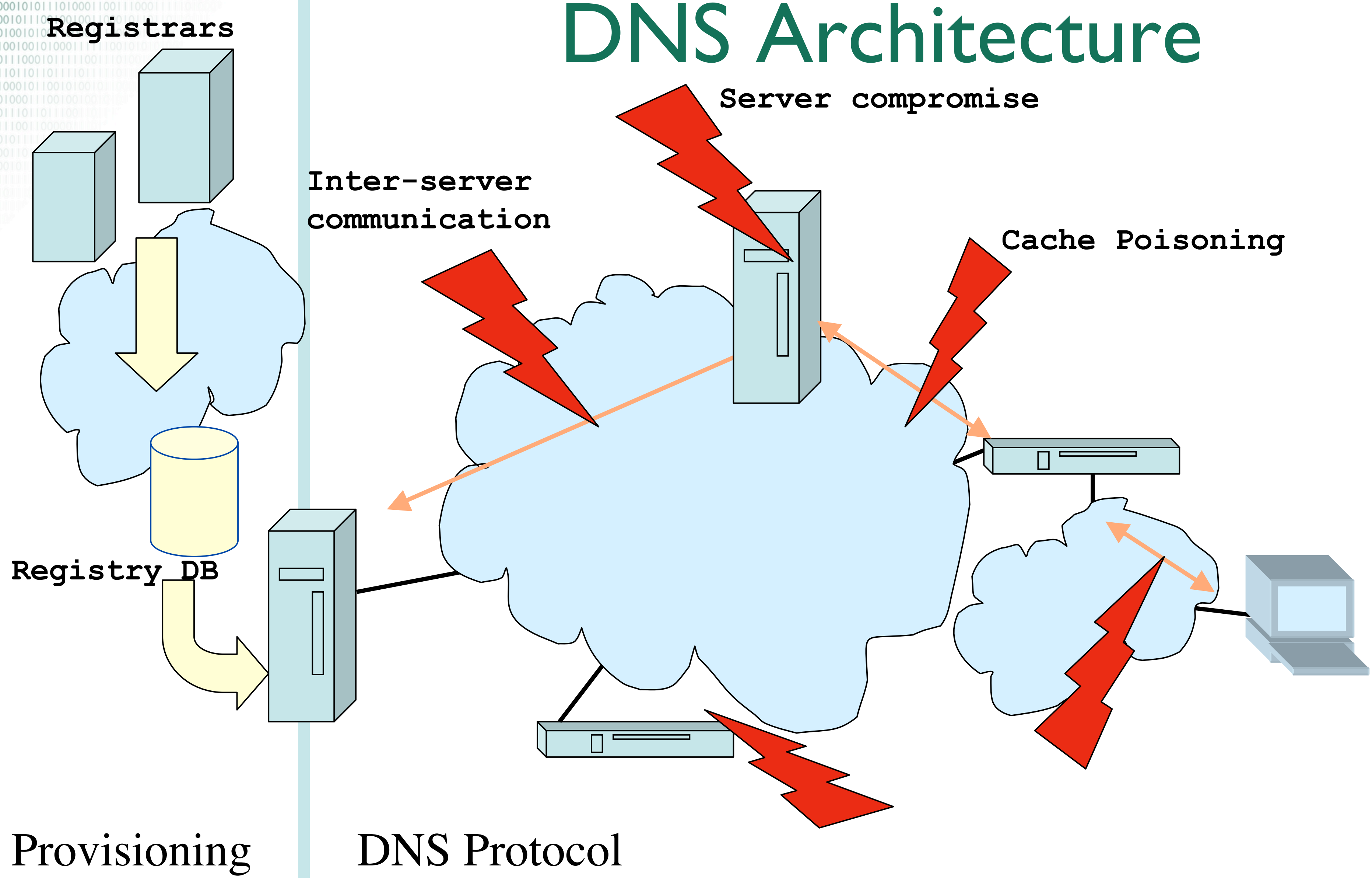
- DNS data published by the registry is being replaced on its path between the "server" and the "client".
- This can happen in multiple places in the DNS architecture
  - Some places are more vulnerable to attacks then others
  - Vulnerabilities in DNS software make attacks easier (and there will always be software vulnerabilities)

# DNS Architecture

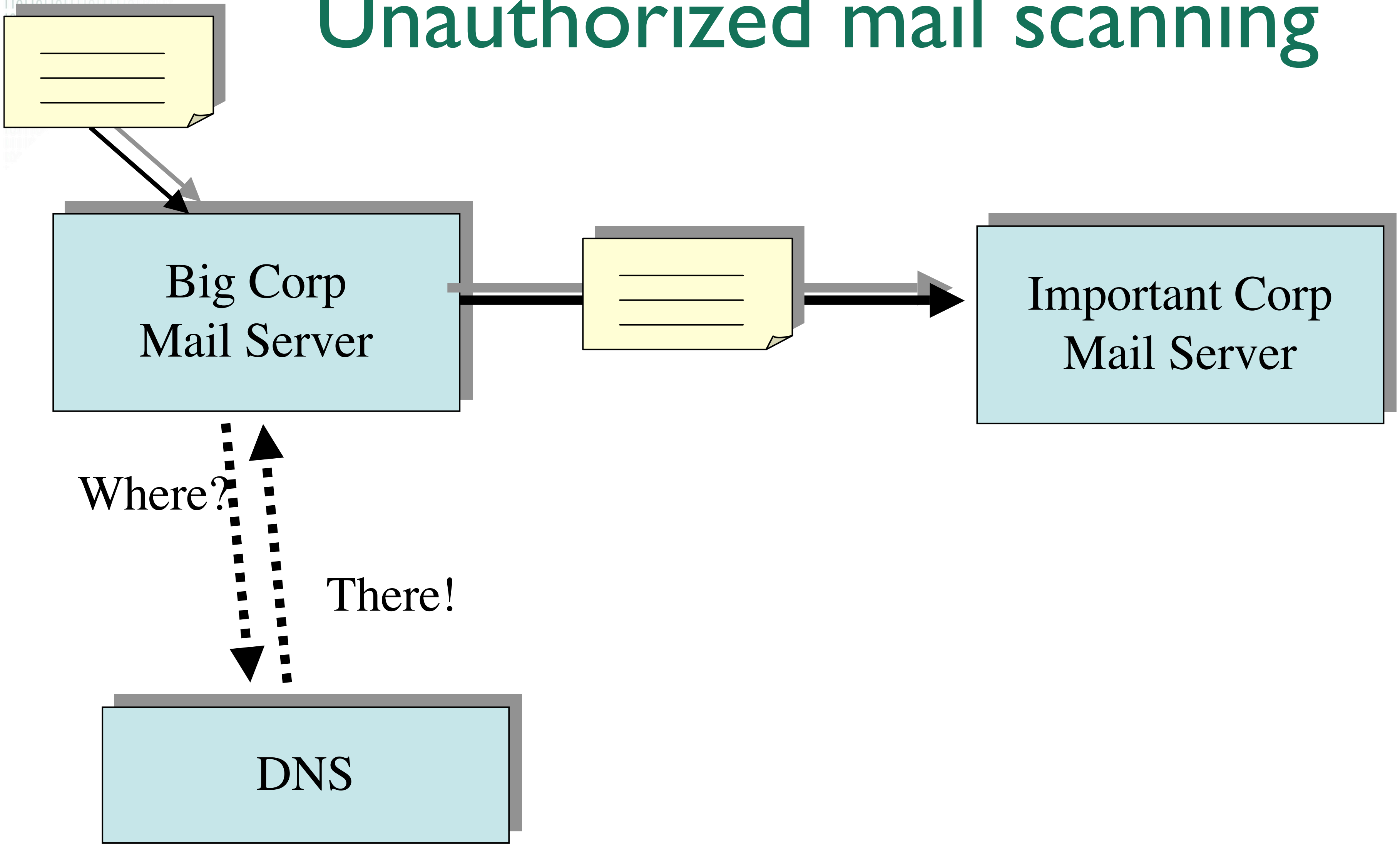



Provisioning DNS Protocol

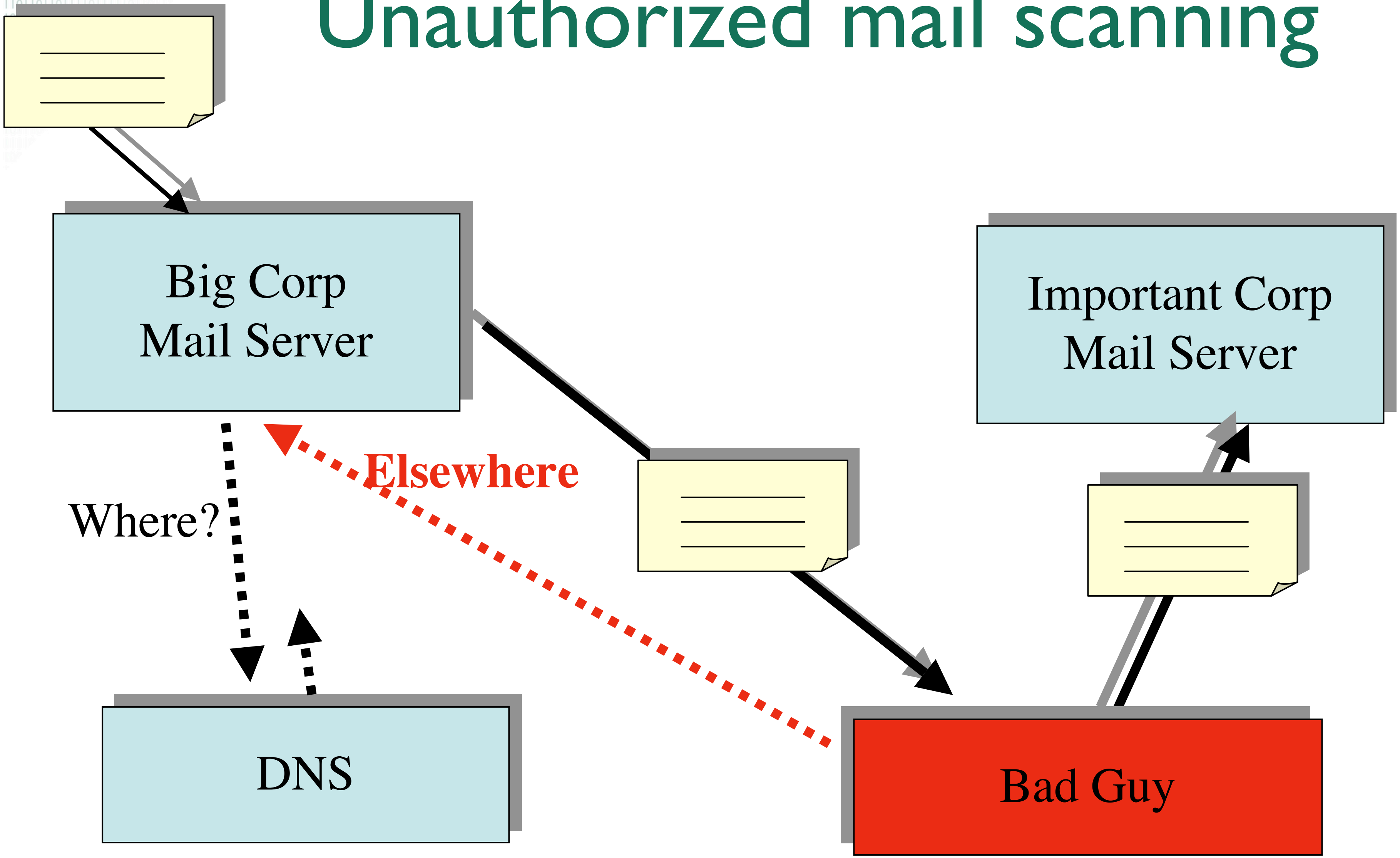
# DNS Architecture



# Example: Unauthorized mail scanning



# Example: Unauthorized mail scanning



# Targets…
## Where do DNS and economics meet?

- SPF, DomainKey and family
  - Technologies that use the DNS to mitigate spam and phishing: $$$ value for the black hats
- Stock tickers, RSS feeds
  - Usually no source authentication but supplying false stock information via a stock ticker and via a news feed can have $$$ value
- ENUM
  - Mapping telephone numbers to services in the DNS

# DNSSEC properties

- DNSSEC provides message authentication and integrity verification through cryptographic signatures
  - Authentic DNS source
  - No modifications between signing and validation
- It does not provide authorization
- It does not provide confidentiality

# Metaphor

- Compare DNSSEC to a sealed transparent envelope.
- The seal is applied by whoever closes the envelope
- Anybody can read the message
- The seal is applied to the envelope, not to the message

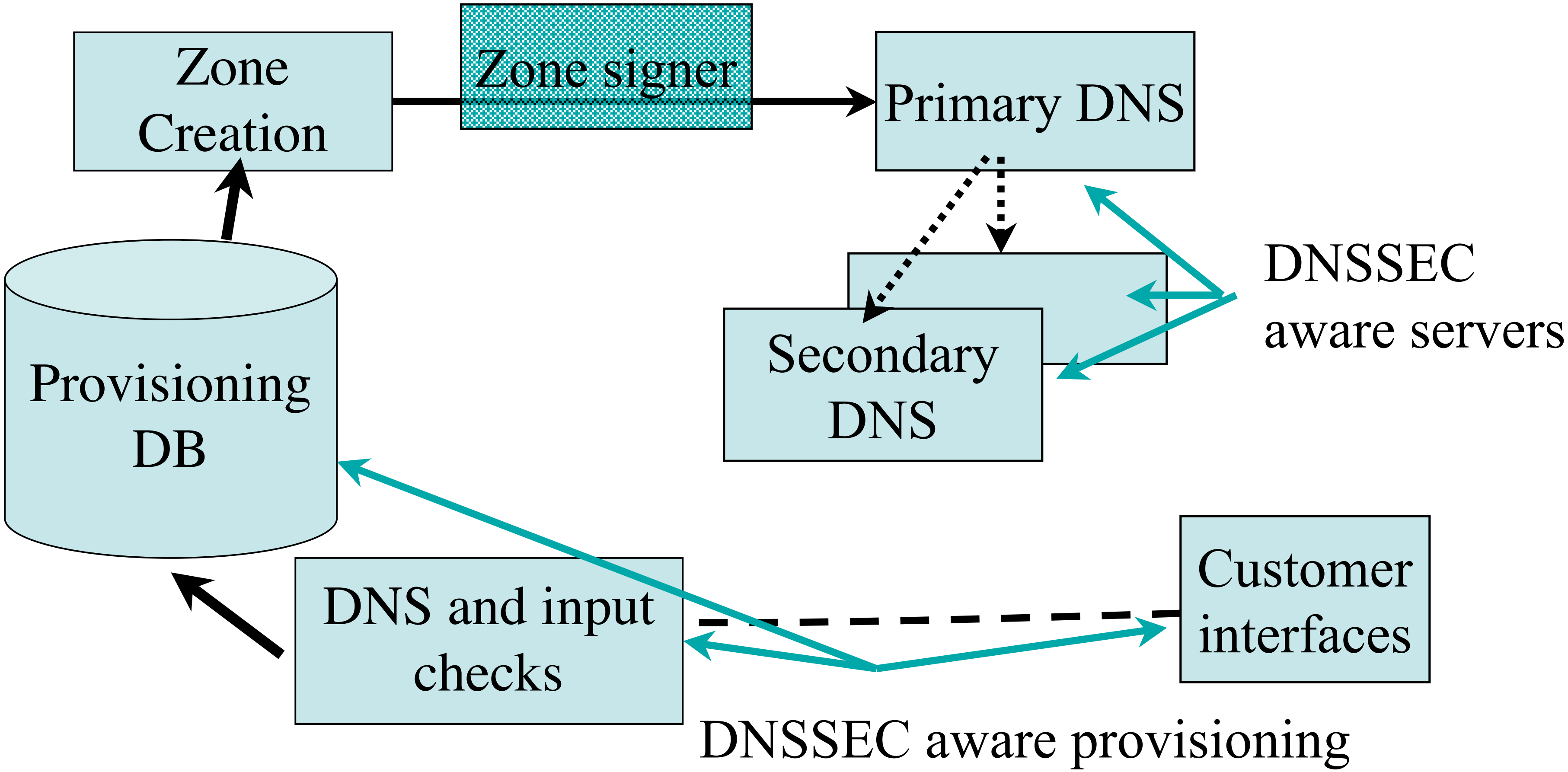
```
nlnetlabs.nl.           86400 IN NS omval.tednet.nl.
nlnetlabs.nl.           86400 IN RRSIG NS 5 2 86400 20060131005003 (
                            20060101005003 43791 nlnetlabs.nl.
                            y9URn7lLgXdPh2MY/I/CRkmG5PZ2nJQxndGHwSB8OP15
                            [...]
                            Qgl+b0Oj54b/65TyYVqPp3CLVt/q6L67SMK9orE= )

nlnetlabs.nl.           86400 IN DNSKEY 257 3 5 (
                            AQPzzTWMz8qSWIQlfRnPckx2BiVmkVN6LPupO3mbz7Fh
                            [...]
                            9QP7y2bVw5zSbFCrefk8qCUBgfHm9bHzMG1UBYtEIQ==
                            ) ; key id = 43791
nlnetlabs.nl.           86400 IN RRSIG DNSKEY 5 2 86400 20060131005003 (
                            20060101005003 43791 nlnetlabs.nl.
                            6mEWVaFKLQrypCJb0wgzW56UC1SVGYW3z+KJVM/uANnA
                            [...]
                            4cCBqAX/mE2N5nUZians0/IT1TzuqVS5h12R2jE= )
open.nlnetlabs.nl.      86400 IN RRSIG A 5 3 86400 20060131005003 (
                            20060101005003 43791 nlnetlabs.nl.
                            ZoZl6om+q5e4jrrDB64mMXpiloCQVe6NDu7KZbv7k9fg
                            [...]
                            t+Xc4iQPI+HTPn5A18pjaEA9KAK7NFLTZHz5uV8= )
open.nlnetlabs.nl.      86400 IN RRSIG AAAA 5 3 86400 20060131005003 (
                            20060101005003 43791 nlnetlabs.nl.
                            [...]
```

# Presentation roadmap

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

# DNSSEC
# Architecture modifications

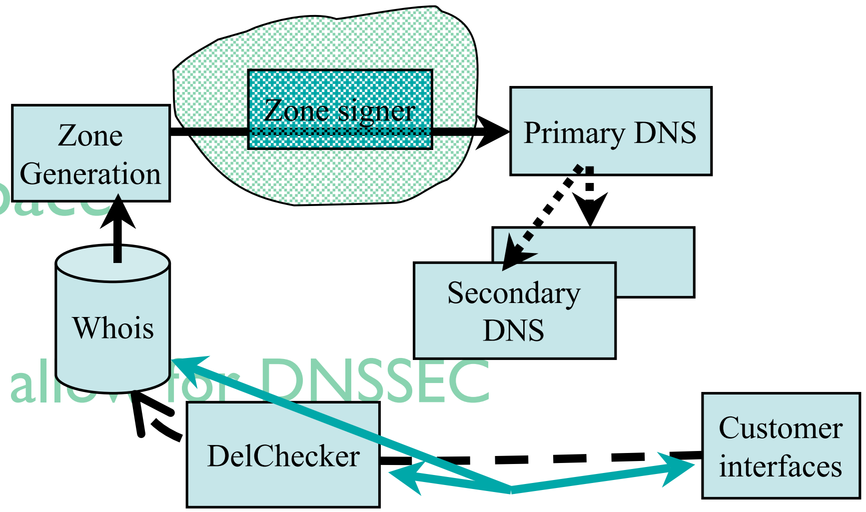


# DNSSEC deployment tasks

- 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 introduction
  - 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:

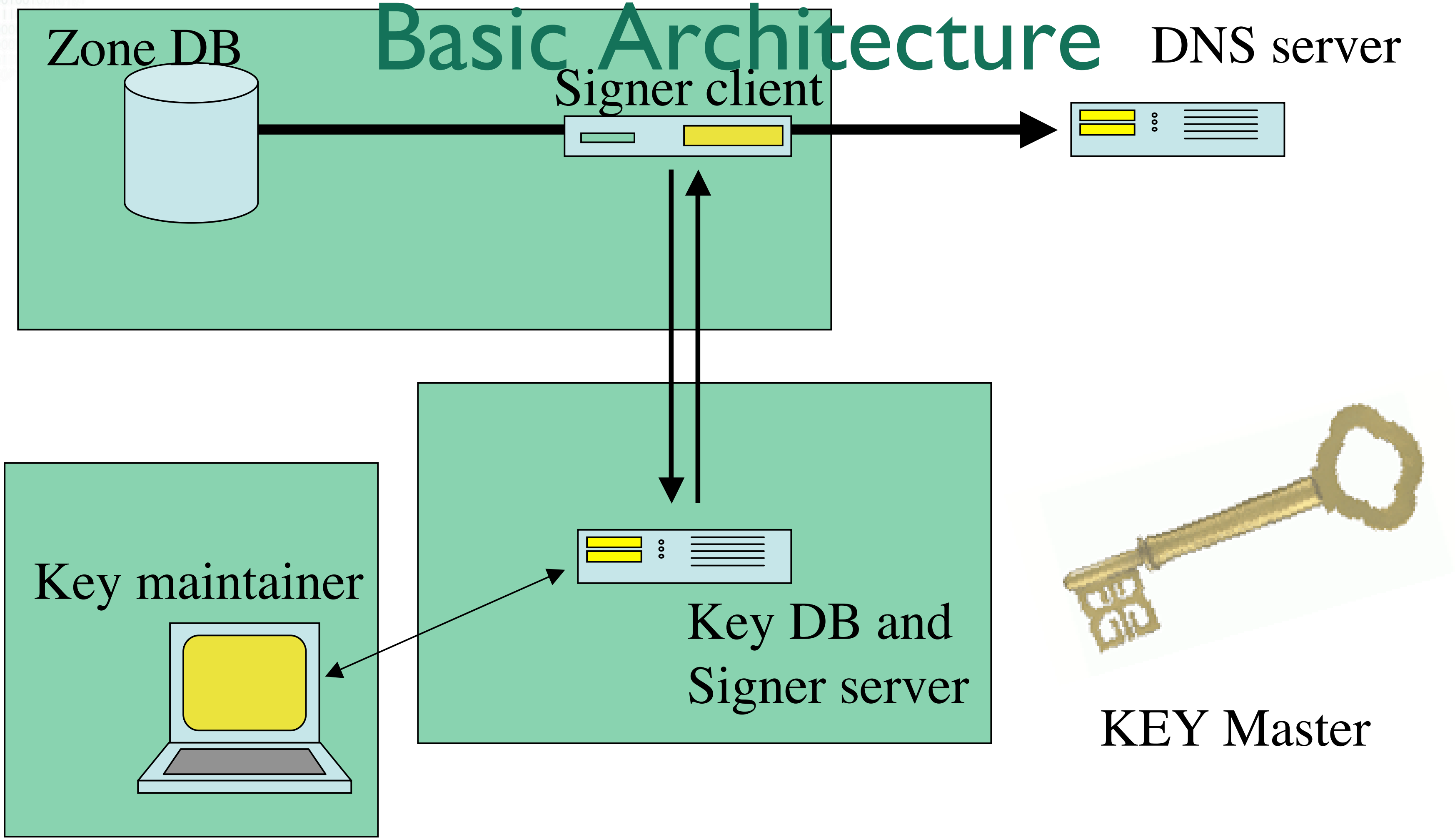
- Trust anchors are published on an "HTTPS" secured website
- Trust anchors are signed with the RIPE NCC public keys
- Trust anchor will be rolled twice a year (during early deployment)
- Announcements and publications are always signed by x.509 or PGP

# 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]

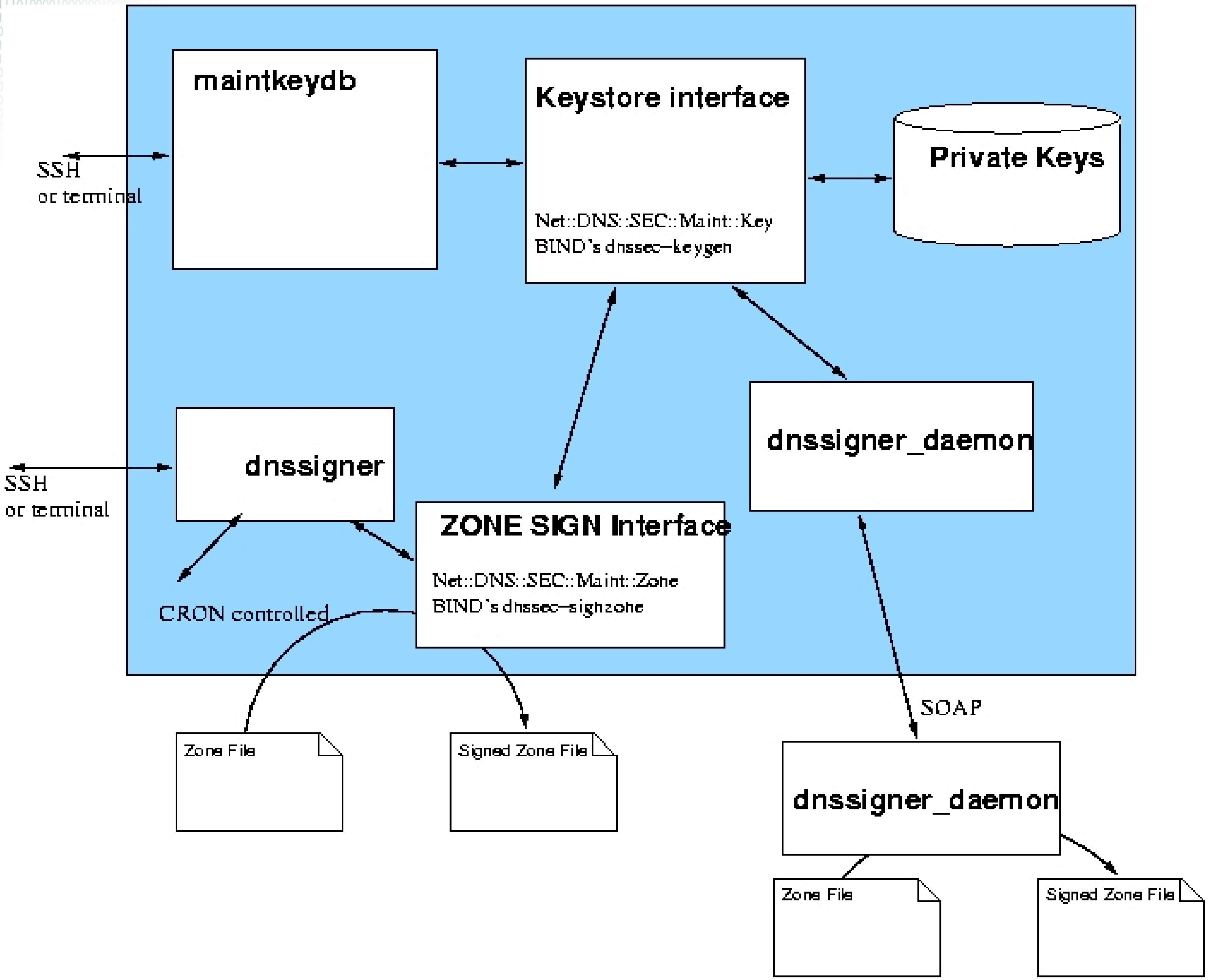
- Private keys need shielding

# 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


maintkeydb
Keystore interface
Private Keys
SSH
or terminal
Net::DNS::SEC::Maint::Key
BIND's dnssec-keygen
dnssigner
dnssigner_daemon
SSH
or terminal
ZONE SIGN Interface
Net::DNS::SEC::Maint::Zone
BIND's dnssec-signzone
CRON controlled
SOAP
Zone File
Signed Zone File
dnssigner_daemon
Zone File
Signed Zone File

# 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

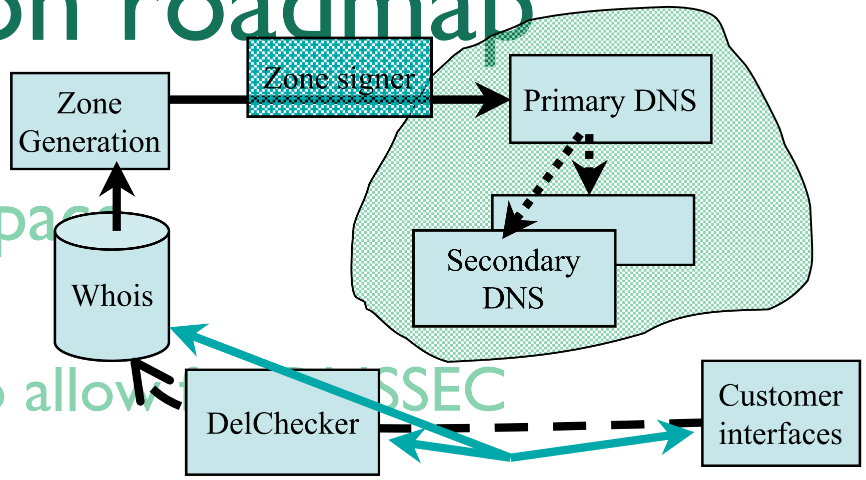
# Example session

```
$ 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 space
  - DNSSEC introduction
  - Architectural changes to allow for DNSSEC deployment
- Deployment tasks
  - Key maintenance
  - DNS server infrastructure
  - Providing secure delegations




DNSSEC aware provisioning

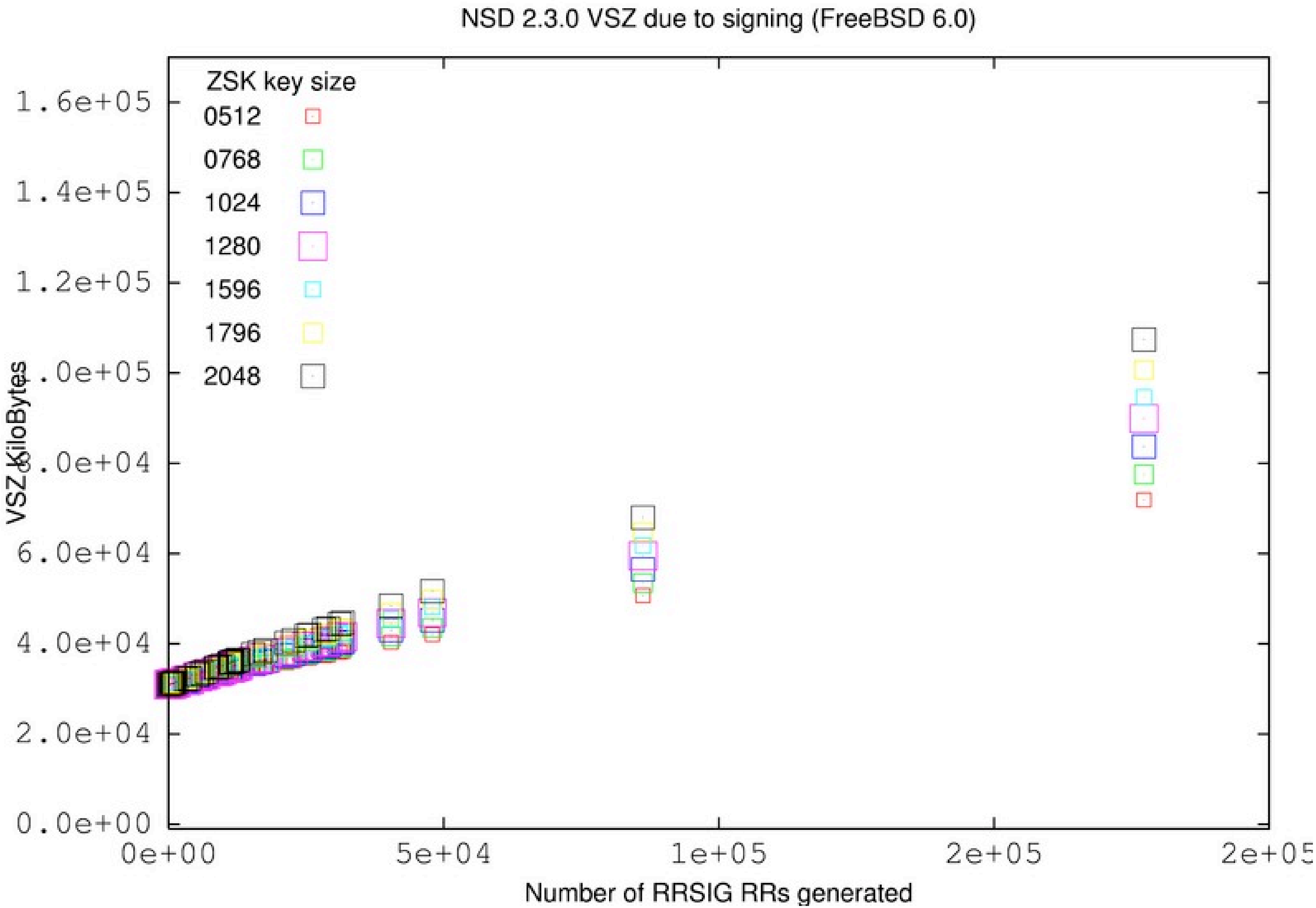
# Infrastructure

- 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.
- Published in RIPE352


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

# Memory

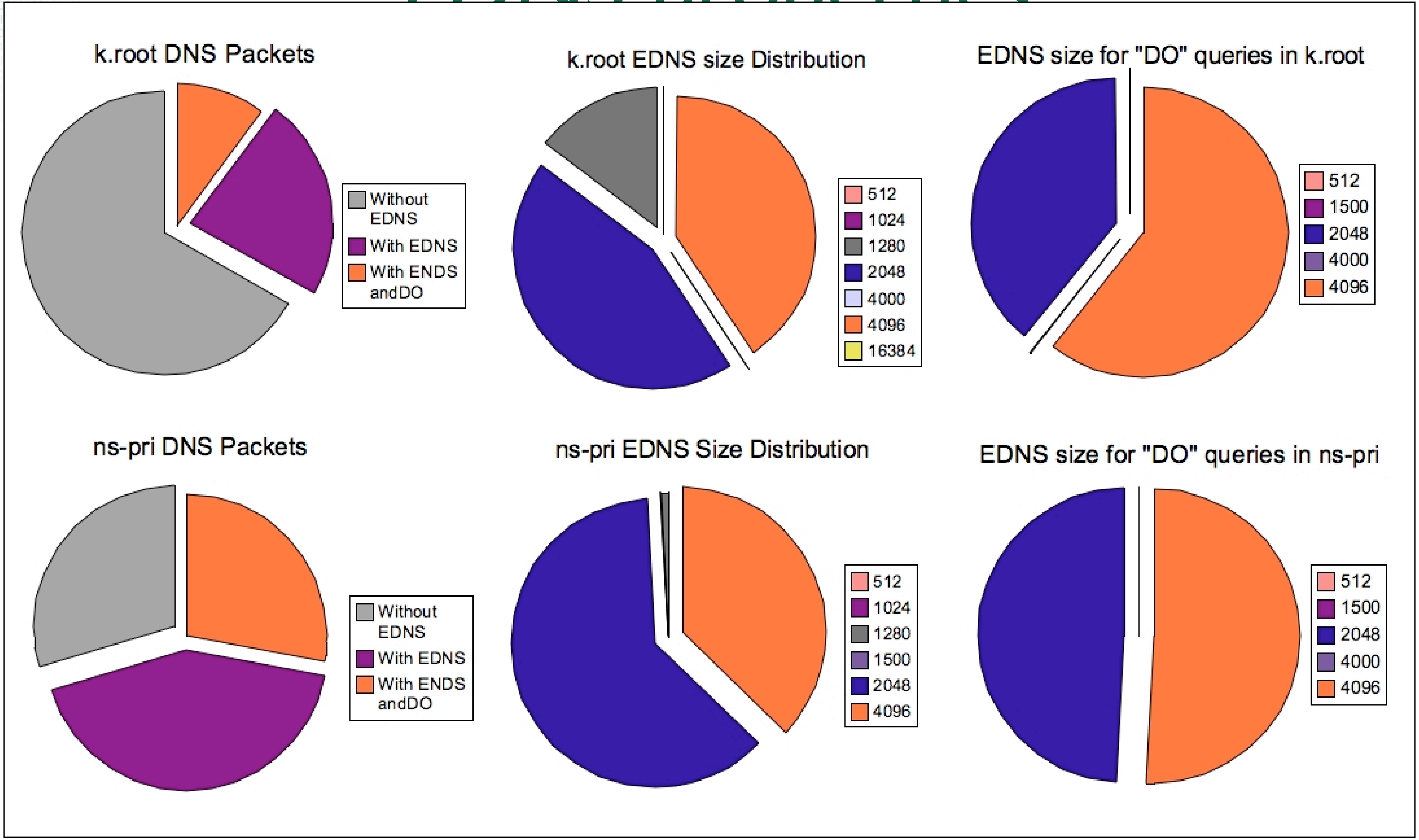
- On ns-pri.ripe.net factor 4 increase.
  – From ca. 30MB to 150MB
  – No problem for a 1GB of memory machine
- On k.root-servers.net
  – Increase by ca 150KB
  – Total footprint 4.4 MB
- Nothing to worry about
- Memory consumption on authoritative servers can be calculated in advance.
  – No surprises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

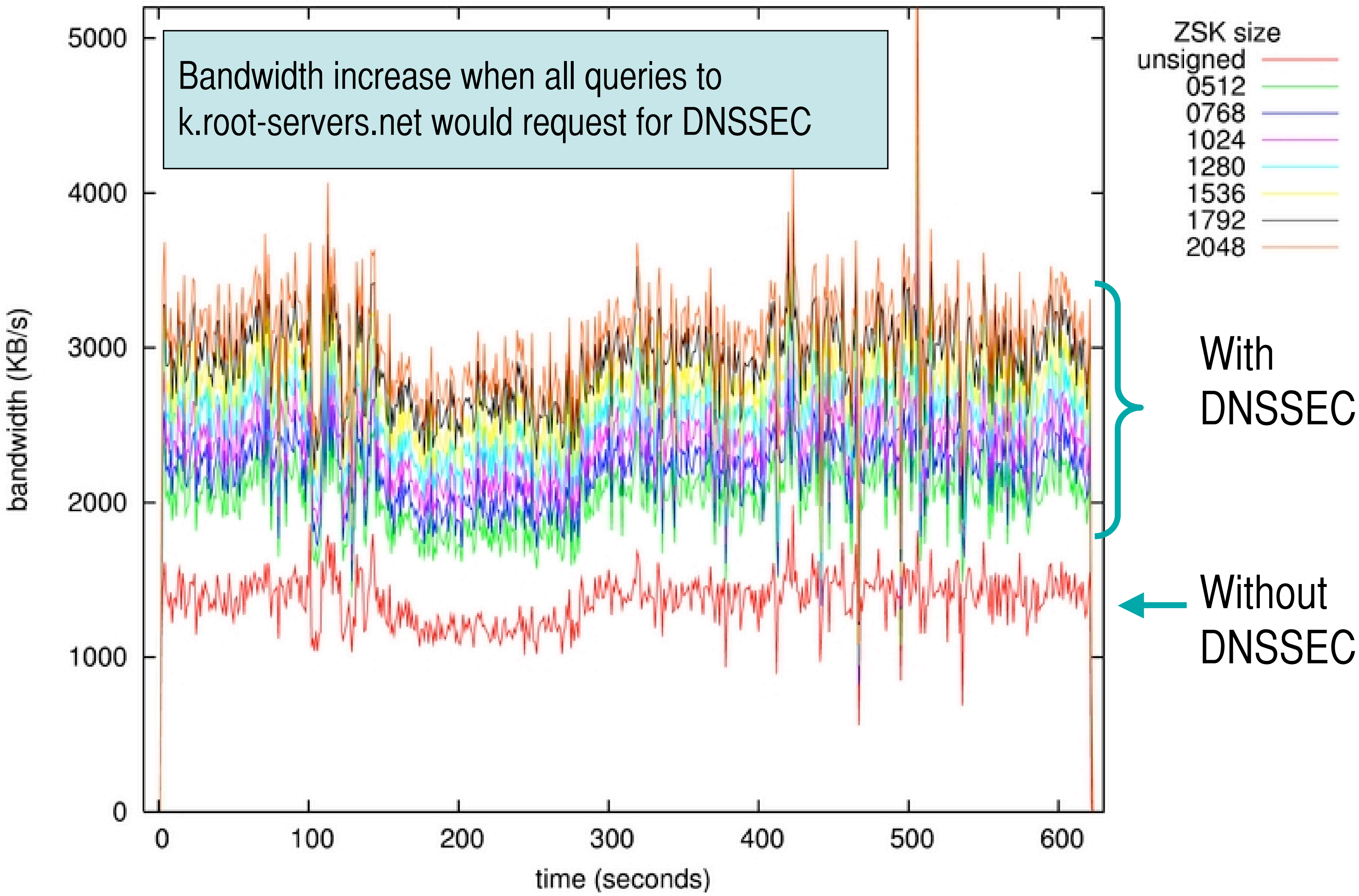


# CPU

| 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


Bandwidth increase when all queries to k.root-servers.net would request for DNSSEC
bandwidth (KB/s)
time (seconds)
ZSK size
unsigned
0512
0768
1024
1280
1536
1792
2048
With DNSSEC
Without DNSSEC
0
1000
2000
3000
4000
5000
0
100
200
300
400
500
600

# 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

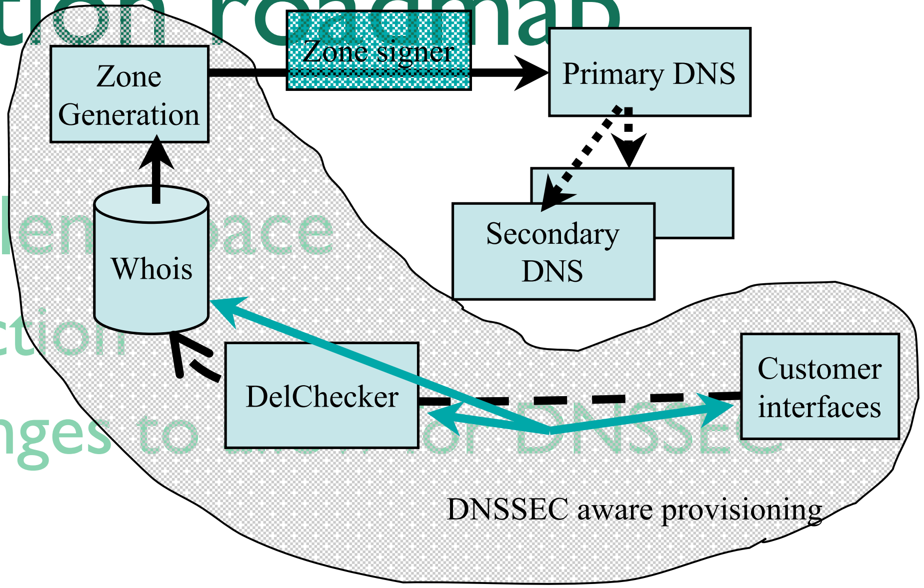
*(Key size influences bandwidth but bandwidth should not influence your key size)*

# 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 introduction
  - Architectural changes to ... 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

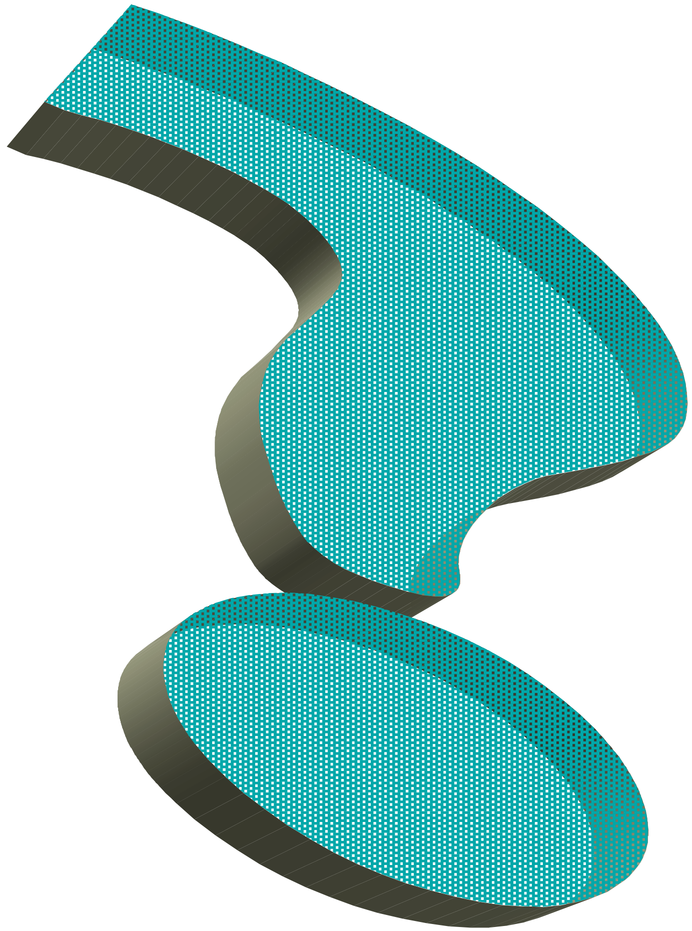
May the Demo Gods be with us.

We'll cheat.

# NCC Roadmap

- RIPE NCC is signing its zones
  - Forward zones (ripe.net &c) are signed
  - Signatures are still being introduced in reverse zones (v4 and v6)
- Secure Delegations available for a number of /8 equivalent zones
- Policy and procedures available
  - www.ripe.net/reverse

# QUESTIONS?



Acknowledgements: A number of these slides are based on earlier work at RIPE NCC.