DNSSEC

An Introduction to Concepts

APNIC-16 august 2003

Why DNSSEC?

- DNS is not secure
  - Applications depend on DNS
  - Known vulnerabilities
- DNSSEC protects against data spoofing and corruptior

Jump to first page ⇦ ⇨

Outline

- Introduction
- DNSSEC mechanisms
  - to authenticate servers (TSIG / SIG0)
  - to establish authenticity and integrity of data
    - Quick overview
    - New RRs
    - Using public key cryptography to sign a single zone
    - Delegating signing authority ; building chains of trust
    - Key exchange and rollovers
- Conclusions

Jump to first page

DNS: Known Concepts

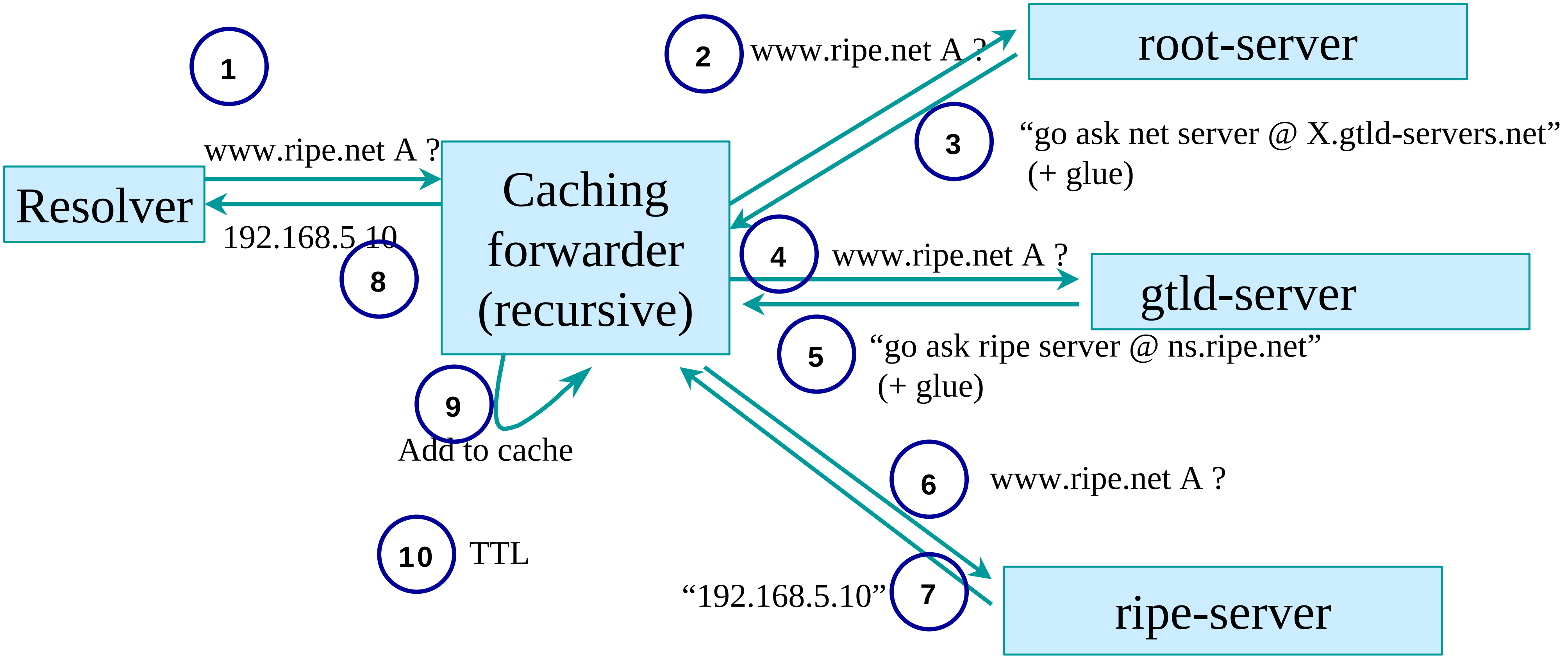
- Known DNS concepts:
  - Delegation, Referral, Zone, RRs, label, RDATA, authoritative server, caching forwarder, stub and full resolver, SOA parameters, etc
  - Don’t know? Do ask!

Jump to first page

Reminder: DNS Resolving

Question:

www.ripe.net A



1

www.ripe.net A ?

Resolver

192.168.5.10

8

Caching forwarder (recursive)

2 www.ripe.net A ?

root-server

3 “go ask net server @ X.gtld-servers.net” (+ glue)

4 www.ripe.net A ?

gtld-server

5 “go ask ripe server @ ns.ripe.net” (+ glue)

6 www.ripe.net A ?

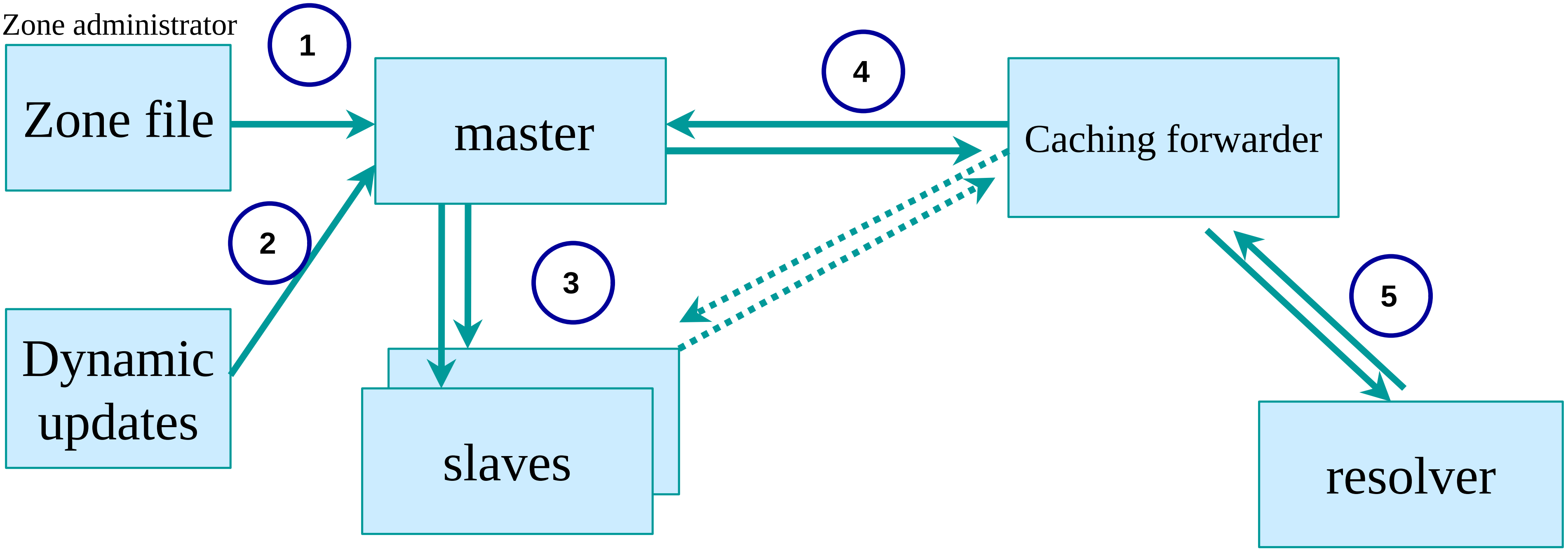
ripe-server

“192.168.5.10” 7

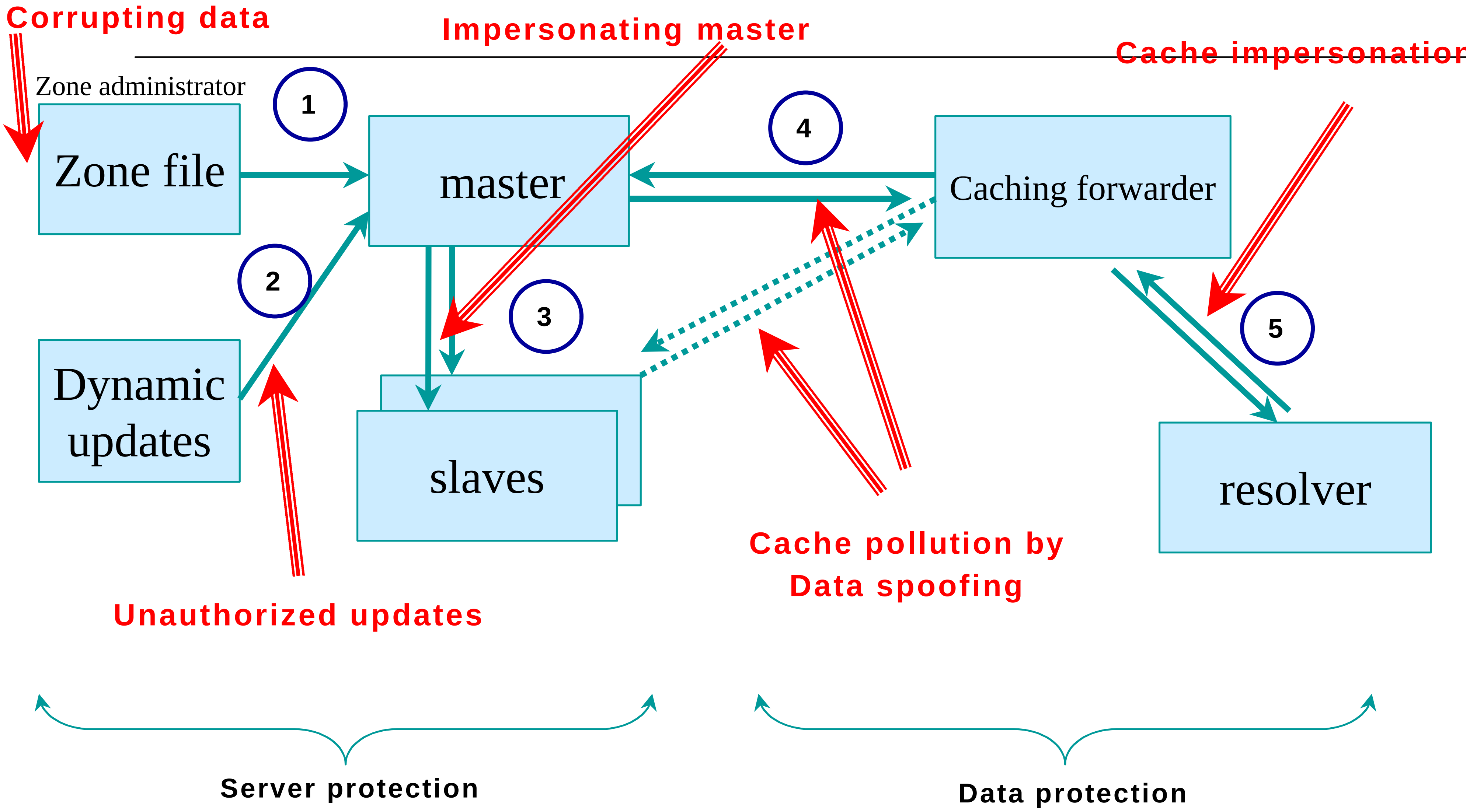
9 Add to cache

10 TTL

DNS: Data Flow



DNS Vulnerabilities



Corrupting data

Impersonating master

Cache impersonatior

Zone administrator

1

Zone file

master

4

Caching forwarder

2

Dynamic updates

3

slaves

5

resolver

Cache pollution by Data spoofing

Unauthorized updates

Server protection

Data protection

Jump to first page

DNS Protocol Vulnerability

- DNS data can be spoofed and corrupted on its way between server and resolver or forwarder
- The DNS protocol does not allow you to check the validity of DNS data
  - ☞ Exploited by bugs in resolver implementation (predictable transaction ID)
  - ☞ Polluted caching forwarders can cause harm for quite some time (TTL)
  - ☞ Corrupted DNS data might end up in caches and stay there for a long time
- How does a slave (secondary) knows it is talking to the proper master (primary)?

Jump to first page ⇦ ⇨

Motivation for DNSSEC

DNSSEC protects against data spoofing and corruption

- DNSSEC (TSIG) provides mechanisms to authenticate servers
- DNSSEC (KEY/SIG/NXT) provides mechanisms to establish authenticity and integrity of data

- A secure DNS will be used as a public key infrastructure (PKI)
  - However it is **NOT** a PKI

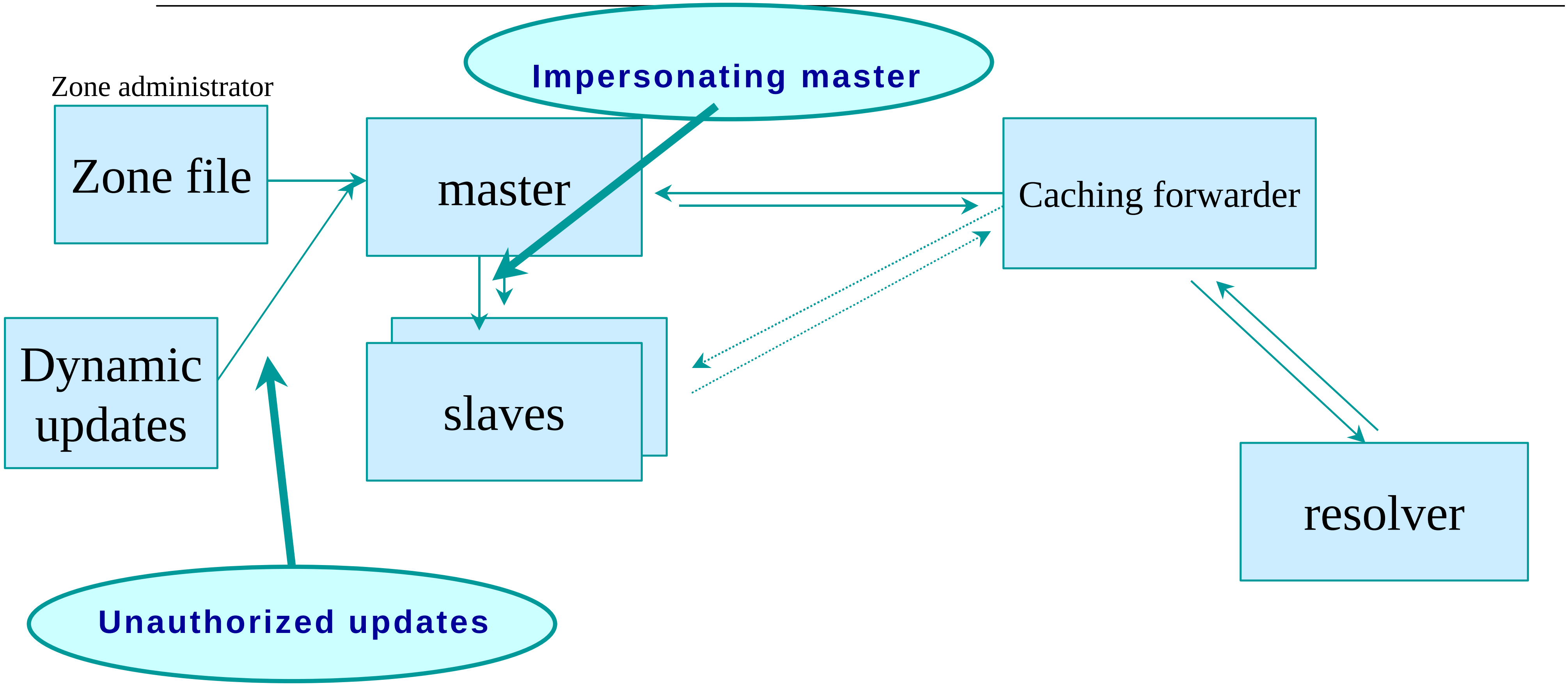
DNSSEC Current State

- This tutorial is based on the ‘current’ RFC2535 with modifications
- Changes to the specs that are now going through the IETF:
  - Rewrite of the specs; mainly an editing job;
  - Incorporation of operational experiences;
  - Changes not backward compatible with current specs!
    - E.g. introduction of DS, NXT, NXT opt-in, AD bit, etc

DNSSEC Mechanisms to Authenticate Servers

- TSIG
- SIG0

TSIG Protected Vulnerabilities

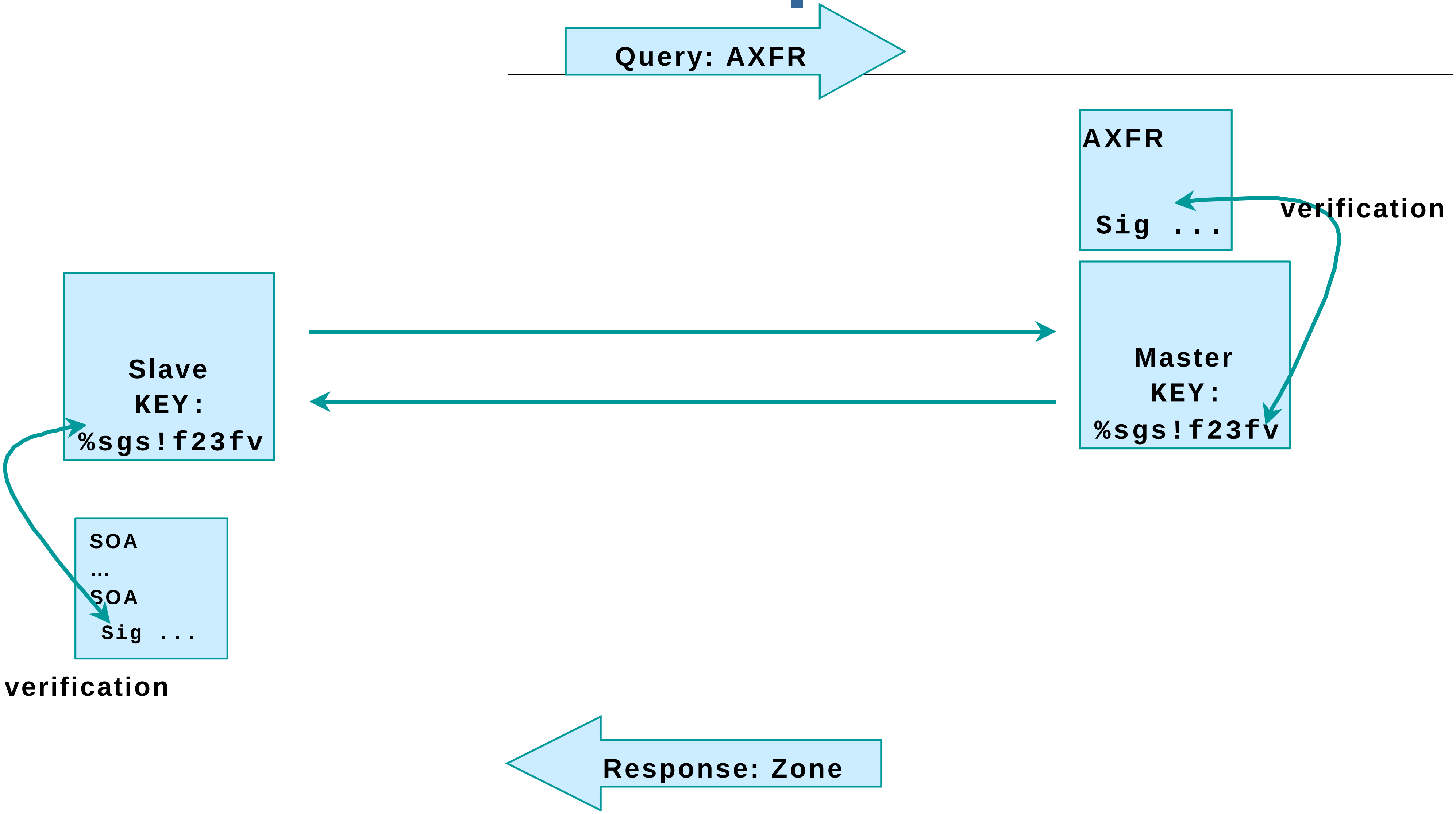


Transaction Signature: TSIG

- TSIG (RFC 2845)
  - authorizing dynamic updates & zone transfers
  - authentication of caching forwarders
  - can be used without deploying other features of DNSSEC
- One-way hash function over:
  - DNS question or answer
  - & the timestamp
- Signed with “shared secret” key
- Used in server configuration, not in zone file

Jump to first page

TSIG example



Query: AXFR

AXFR

Sig ...

verification

Slave
KEY:
%sgs!f23fv

Master
KEY:
%sgs!f23fv

SOA
…
SOA
Sig ...

verification

Response: Zone

Authenticating Servers Using SIG0

- Alternatively its possible to use SIG0
  - Not widely used yet
  - Works well in dynamic update environment

- Public key algorithm
  - Authentication against a public key published in the DNS

Summary: Steps to TSIG Configuration

- Configuring secure transfers between servers with TSIG
  1. Generate a key using “DNSSEC-keygen”
  2. Communicate key with your partner (off-band, PGP…)
  3. Configure your server to require the key for zone transfers
     - ☞ “**`key`**” statement to configure the key
     - ☞ “**`allow-transfer`**” statement in the “**`zone`**” statement
     - ☞ tip: use “**`include <file_name>`**”
  4. Have your partners configure their servers to use the key when talking to you
     - ☞ Using the “**`server`**“ statement

Importance of the Time Stamp

- TSIG/SIG0 signs a complete DNS request / response with time stamp
  - to prevent replay attacks
  - ‘seconds since epoch’

- Operational problems when comparing times
  - Make sure your local time zone is properly defined
  - **`date -u`** will give UTC time, easy to compare between the two systems

- Use NTP synchronization!!!

TSIG: Questions?

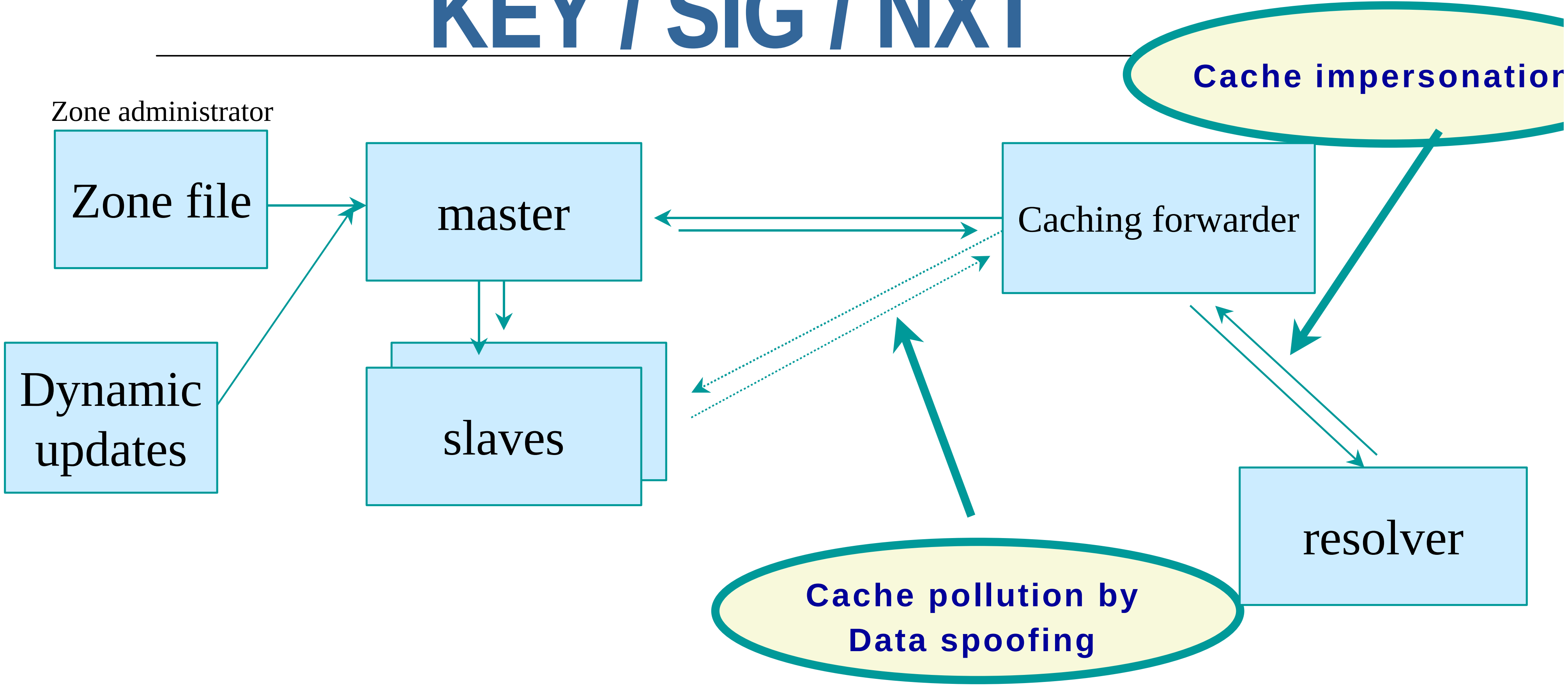
Jump to first page ⇦ ⇨

DNSSEC Mechanisms to Establish Authenticity and Integrity of Data

=> ***Quick overview***

- New RRs
- Using public key cryptography to sign a single zone
- Delegating signing authority ; building chains of trust
- Key exchange and rollovers

Vulnerabilities protected by KEY / SIG / NXT



DNSSEC Summary on 1 page

- Data authenticity and integrity by SIGning the resource records
- Public KEYs used to verify the SIGs
- Children sign their zones with their private key; The authenticity of their KEY is established by a SIGnature over that key by the parent (DS)
- In the ideal case, only one public KEY needs to be distributed off-band

Jump to first page ⇦ ⇨

Authenticity and Integrity of Data

- Authenticity: Is the data published by the entity we think is authoritative?

- Integrity: Is the data received the same as what was published?

- Public Key cryptography helps to answer these questions
  - signatures to check both integrity and authenticity of data
  - verifies the authenticity of signatures

Public Key Crypto Reminder

- Key pair: a secret (or private) key and a public key
- Simplified:
  - If you know the public key, you can decrypt data encrypted with the secret key
    - Usually an encrypted hash value over a published piece of information; the owner is the only person who can construct the secret. Hence this a signature
  - If you know the secret key, you can decrypt data encrypted with the public key
    - data is usually an encrypted key for symmetric cipher
- PGP uses both, DNSSEC only uses signatures

Jump to first page

Public Key Crypto Issues

- Public keys need to be distributed
- Secret keys need to be kept secret
- Public key cryptography is ‘slow’
- Math:
  - The security of the cryptosystem is based on a set of mathematical problems for which guessing a solution requires scanning a huge solution space (*e.g.* factorization)
  - Algorithms *e.g.*: DSA, RSA, elliptic curve
  - RSA/SHA1 is a good choice
    - Better than RSA/MD5

Jump to first page

New Resource Records for DNSSEC

DNSSEC New RRs

- 3 Public key crypto related RRs
  - SIG   Signature over RRset made using private key
  - KEY   Public key, needed for verifying a SIG over a RRset
  - DS   Delegation Signer; ‘Pointer’ for building chains of trust

- One RR for internal consistency
  - ☞ authenticated non-existance of data
  - NXT   Indicates which RRset is the next one in the zone

Other Keys in the DNS

- For non DNSSEC, public keys can appear in the DNS
- CERT
  - For x509 certificates
- Under discussion/development are application keys
  - IP-SEC
  - SSH

Jump to first page

Recap: RRs and RRsets

- Resource Record:
  - name TTL class type rdata

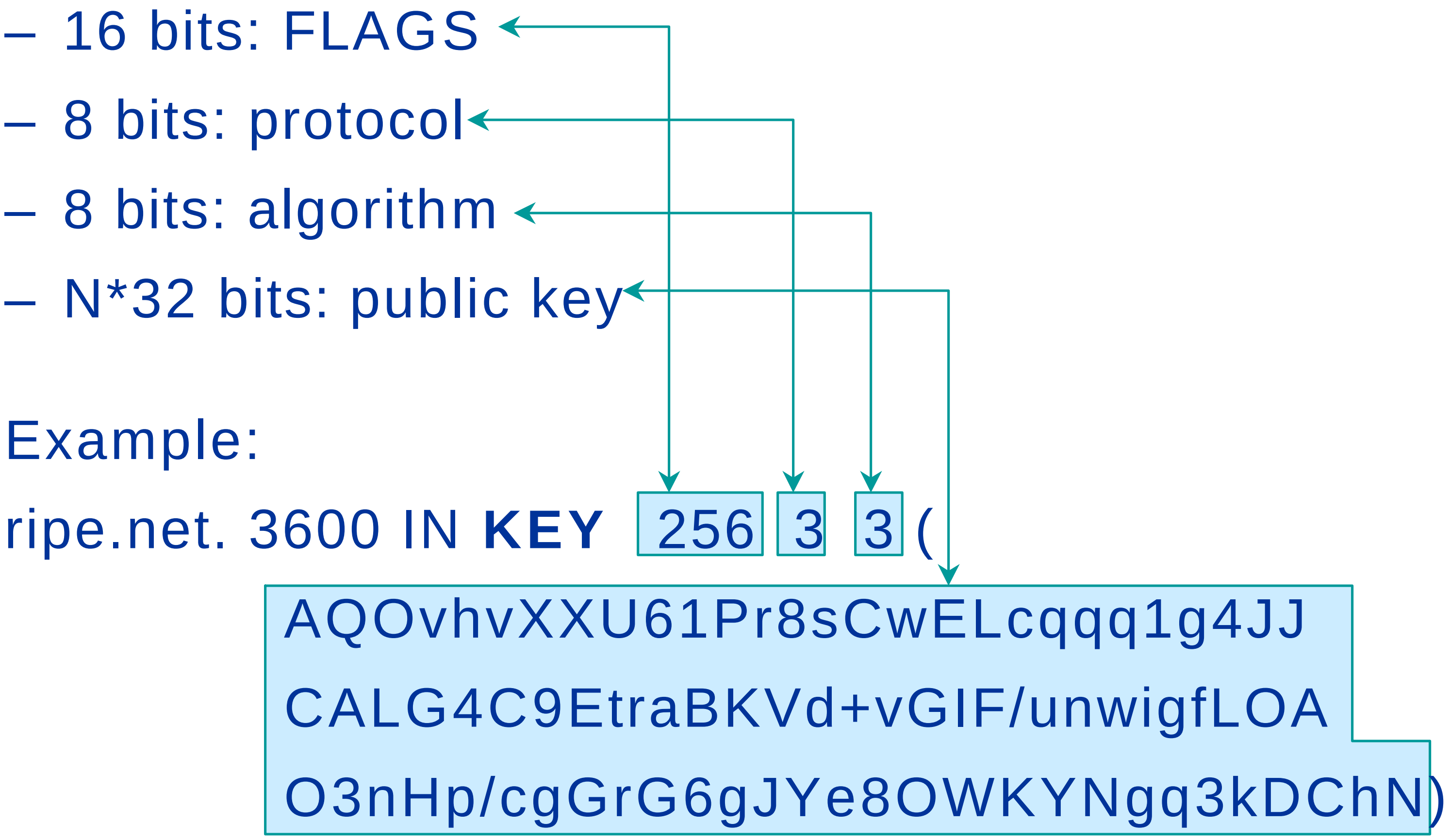
```
www.ripe.net.   7200 IN   A    192.168.10.3
```

- All RRs of a given name, class, type make an RRset:

```
www.ripe.net.   7200 IN   A    192.168.10.3
                          A    10.0.0.3
```

- In DNSSEC the RRsets are signed, not the individual RRs

KEY RDATA

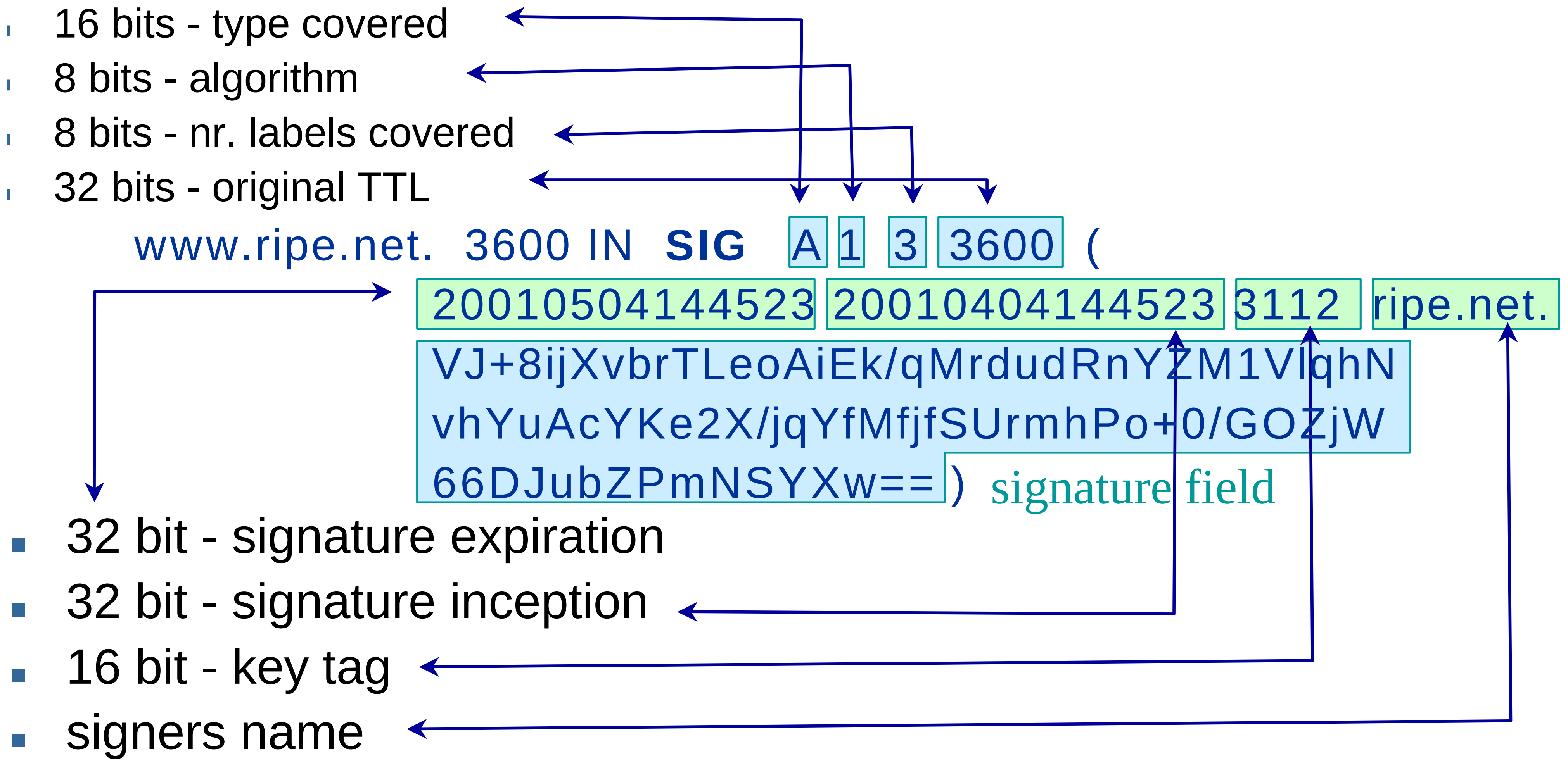


- 16 bits: FLAGS
- 8 bits: protocol
- 8 bits: algorithm
- N*32 bits: public key

Example:

```
ripe.net. 3600 IN KEY 256 3 3 (
        AQOvhvXXU61Pr8sCwELcqqq1g4JJ
        CALG4C9EtraBKVd+vGIF/unwigfLOA
        O3nHp/cgGrG6gJYe8OWKYNgq3kDChN)
```

SIG RDATA



- 16 bits - type covered
- 8 bits - algorithm
- 8 bits - nr. labels covered
- 32 bits - original TTL

```
www.ripe.net.  3600 IN  SIG  A 1 3 3600 (
     20010504144523 20010404144523 3112 ripe.net.
     VJ+8ijXvbrTLeoAiEk/qMrdudRnYZM1VlqhN
     vhYuAcYKe2X/jqYfMfjfSUrmhPo+0/GOZjW
     66DJubZPmNSYXw== )
```

signature field

- 32 bit - signature expiration
- 32 bit - signature inception
- 16 bit - key tag
- signers name

NXT RDATA

- Points to the next domain name in the zone
  - also lists what are all the existing RRsets for “name”
- N*32 bit type bit map
- Used for authenticated denial-of-existence of data
  - authenticated non-existence of TYPEs and labels

- Example:

```
www.ripe.net. 3600 IN   NXT ripe.net. A SIG NXT
```

NXT Record

```
$ORIGIN ripe.net.
@   SOA       …..
           NS      NS.ripe.net.
           KEY       …..
           NXT     mailbox.ripe.net. SOA NS NXT KEY SIG
mailbox    A       192.168.10.2
           NXT     www.ripe.net.  A NXT SIG
WWW A      192.168.10.3
           NXT      ripe.net. A NXT SIG
```

‘popserver’ is missing

- query for popserver.ripe.net would return:

  aa bit set   RCODE=NXDOMAIN

  authority: mailbox.ripe.net. NXT www.ripe.net. A NXT SIG

- query for www.ripe.net MX would return: an empty answer section and the www NXT record in the authority section

Jump to first page

Meaning of NXT

- If you query for data does not exist in a zone, the NXT RR provides proof of non-existence

- If after a query the response is:
  - NXDOMAIN: One, and maybe many more, NXT RRs indicate that the name or a wildcard expansion does not exist
  - NOERROR and empty answer section: The NXT TYPE array proves that the QTYPE did not exist
- NXT records are generated by tools
  - You do not have to generate NXT RRs by hand

FYI: NXT opt-in Variant

- New variety of the NXT resource record
    - Introduced to cope with the problem that in a secure zone each name is accompanied by a NXT RR with a SIG
- Instead of authenticated denial of existence it indicates authenticated denial of security
- The change in semantic is indicated by leaving the NXT from the bitmap
- Only at delegation points

Jump to first page

NXT opt-in Variant

- Still under discussion in the IETF
  - First implementations have been tested

```
a.com    ns     ns.a.com
a.com    NXT    SIG NS w.com
         SIG    NXT ....

b.com    NS     ns.b.com
c.com    NS     ns.c.com

w.com    NS     ns.w.com
         NXT    SIG NS z.com
         SIG    NXT ....
z.com    NS     ns.z.com
         NXT    SIG NS .com
         SIG    NXT
```

Question for non-existent ba.com will return:

NXDOMAIN
Auth: A.COM NXT   SIG NS w.com

One can not be sure ba.com does not exist.

Jump to first page

New DNS RRs: Questions?

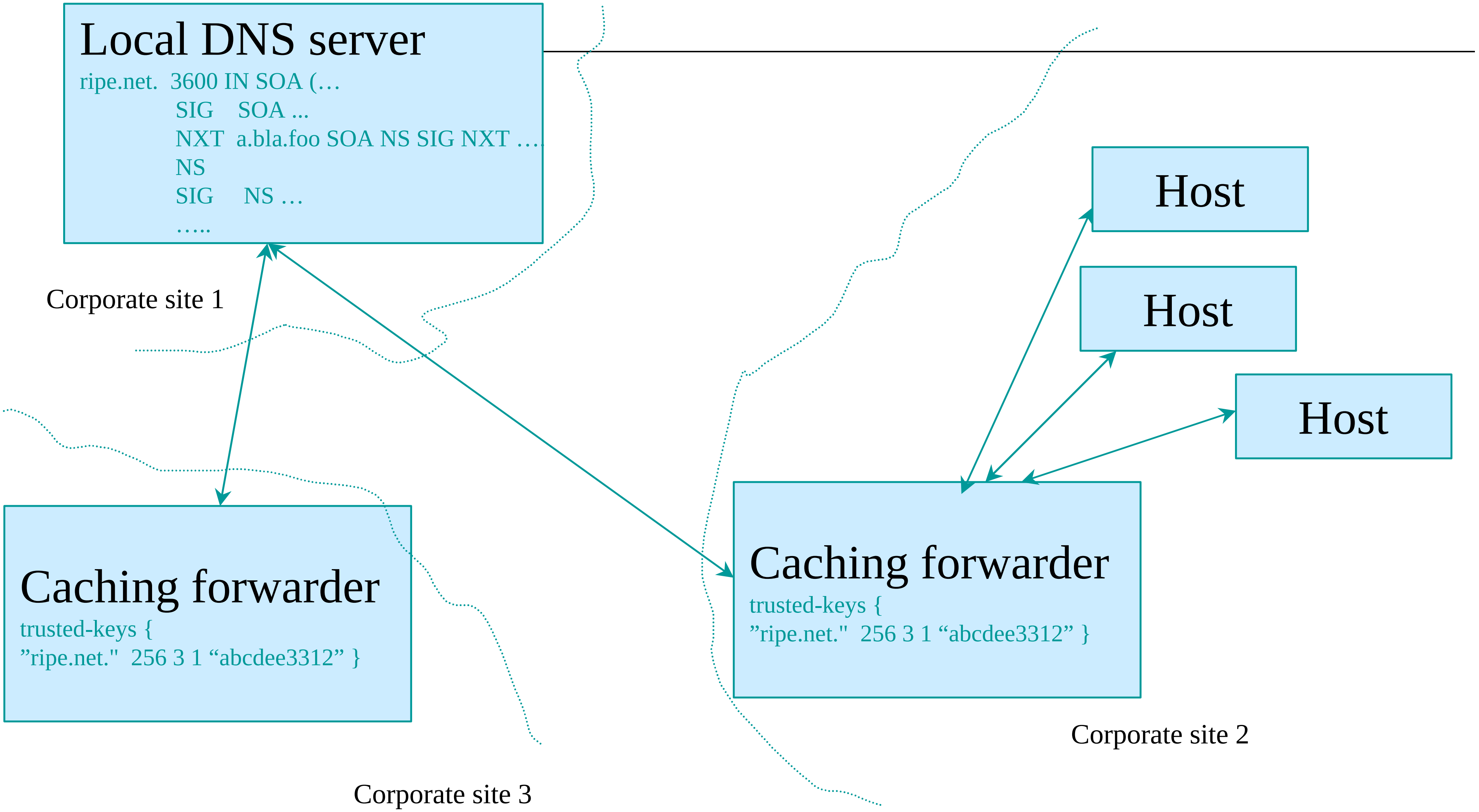
Jump to first page

DNSSEC Signing of a Local Zone

DNSSEC Signing of a Local Zone

1. Generate keys and include them in the zone file
2. Sign your zone; signing will:
   - sort the zone
   - insert the NXT records
   - insert SIG-s containing a signature over each RRset
     - made with your private key
   - generate key-set file (used later)
3. Distribute the Public KEY to those that need to be able to trust your zone
   - they configure your key in their resolver
   - thus configuring “secure entry point” in the tree

Locally Signed Zone



Local DNS server

```
ripe.net.  3600 IN SOA (…
           SIG    SOA ...
           NXT  a.bla.foo SOA NS SIG NXT ….
           NS
           SIG     NS …
           …..
```

Corporate site 1

Host

Host

Host

Caching forwarder

```
trusted-keys {
”ripe.net."  256 3 1 “abcdee3312” }
```

Caching forwarder

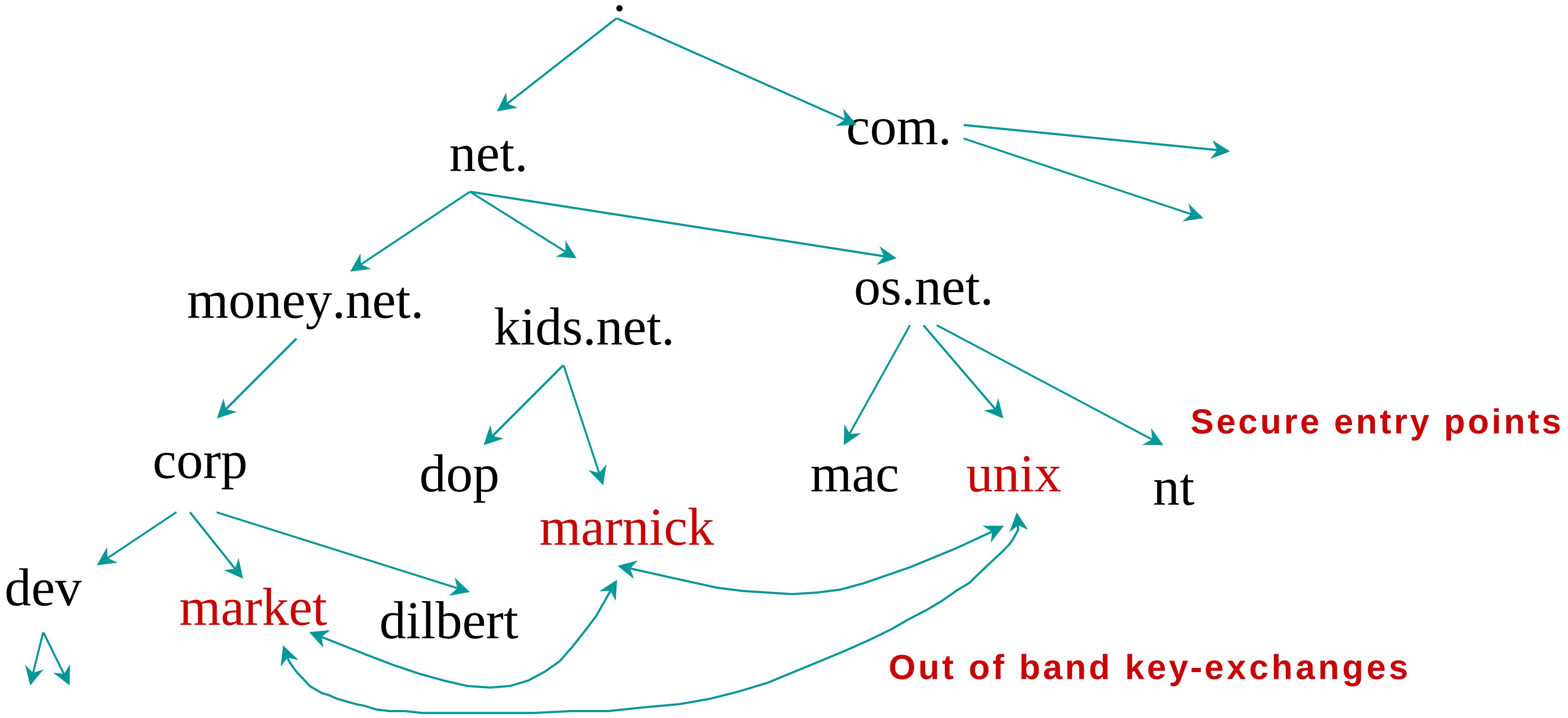
```
trusted-keys {
”ripe.net."  256 3 1 “abcdee3312” }
```

Corporate site 2

Corporate site 3

Locally Secured Zones

- Key distribution problem for distributing keys
  - It would be better if the whole tree would be secured!



Signing Local Zone: Questions?

Jump to first page

Delegating Signing Authority

building chains of trust

Using the DNS to Distribute Keys

- Securing a DNS zone tree
- Building chains of trust from the root down

- Tools: KEY, SIG and DS records

- This material is based on new developments
  - Only in bind9.3.0 November 15 snapshot or later !

Chain of Trust

- The goal is to build a chain of trust from the root down the DNS tree
- You need to verify the public keys with which signatures over other keys are made
- Parents need to sign the keys of their children

- *Outline:*
  - *Which key is used to make a SIG*
  - *How do parents sign children keys*
  - *Walking the chain of trust*

Jump to first page ⇦ ⇨

SIG RDATA

Recap for next slides

```
www.ripe.net.   3600 IN  SIG   A 1 3 3600
20010504144523 (   20010404144523 3112 ripe.net.
                   VJ+8ijXvbrTLeoAiEk/qMrdudRnYZM1VlqhN
                   vhYuAcYKe2X/jqYfMfjfSUrmhPo+0/GOZjW
                   66DJubZPmNSYXw== )
```

This field indicates the signer.

Jump to first page

Delegation Signer (DS)

- The parent delegates authority to sign DNS RRs to the child using this RR

- DS is a pointer to the next key in the chain of trust
  - You may trust data that is signed using a key that the DS points to

- New RR to solve problems with key-rollovers
  - More on that later

Jump to first page ⇦ ⇨

DS RDATA

- 16 bits: key tag
- 8 bits: algorithm
- 8 bits: digest type
- 20 bits: SHA-1 Digest

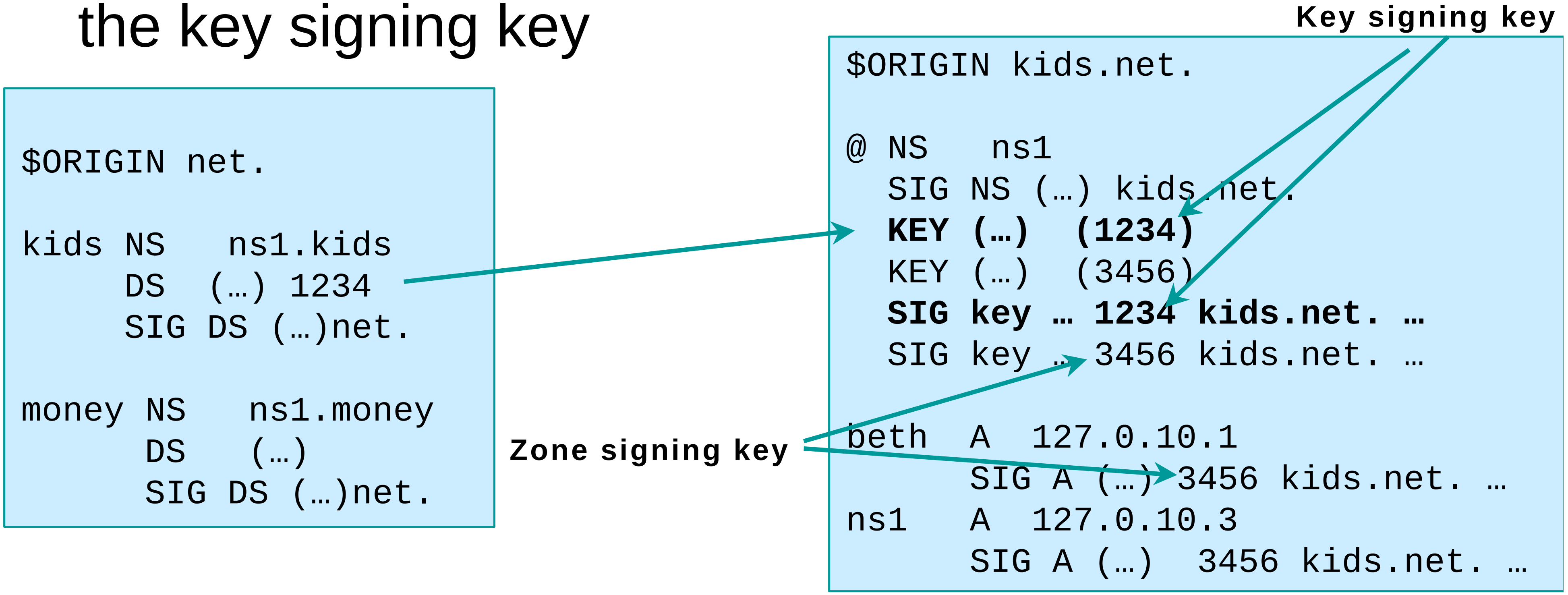
This field indicates which key is the next in the chain of trust

```
$ORIGIN ripe.net.
disi.ripe.net     3600 IN   NS   ns.disi.ripe.net
disi.ripe.net.    3600 IN   DS   3112 1 1 (
                                 239af98b923c023371b52
                                 1g23b92da12f42162b1a9
                                 )
```

Jump to first page

Delegating Signing Authority

- Parent signs the DS record pointing to the key signing key



```
$ORIGIN net.

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

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

```
$ORIGIN kids.net.

@ NS   ns1
  SIG NS (…) kids.net.
  KEY (…)  (1234)
  KEY (…)  (3456)
  SIG key … 1234 kids.net. …
  SIG key … 3456 kids.net. …

beth   A  127.0.10.1
       SIG A (…) 3456 kids.net. …
ns1    A  127.0.10.3
       SIG A (…)  3456 kids.net. …
```

- The parent is authoritative for the DS RR of its children

Key / Zone Signing Keys

Only an administrative distinction, you cannot tell from the KEY record itself!

- DS points to a key signing key (KSK)
- The zone is signed with a zone signing key (ZSK)
  - (these keys may be the same)

- Key signing key may be long lived, and “bigger”
- Zone signing key may be short lived
  - can be “smaller” = “faster”

Chain of Trust Verification, Summary

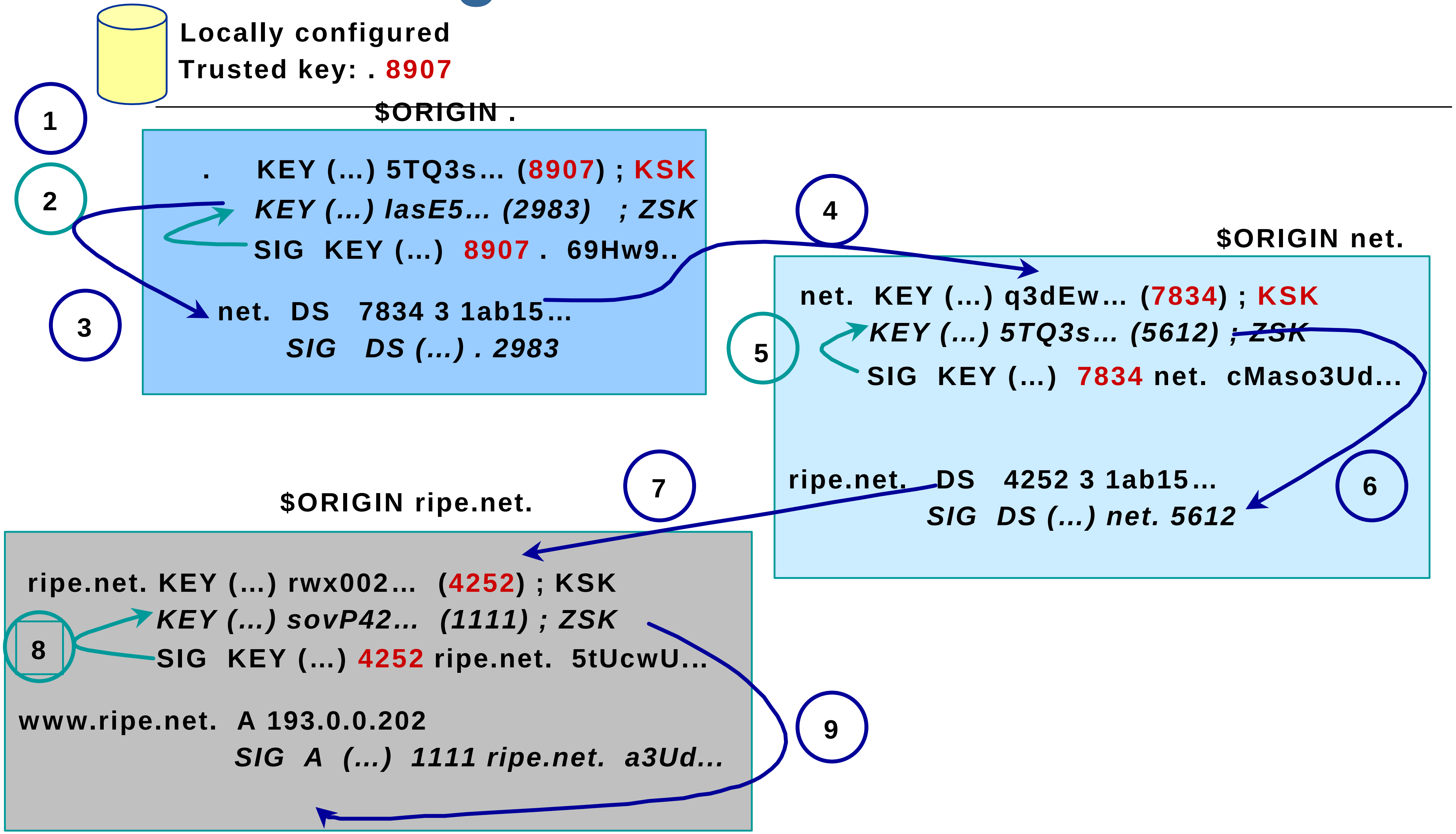
- Data in zone can be trusted if signed by a Zone-Signing-Key
- Zone-Signing-Keys can be trusted if signed by a Key-Signing-Key
- Key-Signing-Key can be trusted if pointed to by trusted DS record
- DS record can be trusted
  - if signed by the parents Zone-Signing-Key

  or

  - DS or Key records can be trusted if exchanged out-of-band and locally stored (Secure entry point)

Jump to first page ⇦ ⇨

Walking the Chain of Trust



Locally configured
Trusted key: . 8907

$ORIGIN .

```
.    KEY (…) 5TQ3s… (8907) ; KSK
     KEY (…) lasE5… (2983)   ; ZSK
     SIG  KEY (…)  8907 .  69Hw9..

net.  DS   7834 3 1ab15…
      SIG   DS (…) . 2983
```

$ORIGIN net.

```
net.  KEY (…) q3dEw… (7834) ; KSK
      KEY (…) 5TQ3s… (5612) ; ZSK
      SIG  KEY (…)  7834 net.  cMaso3Ud…

ripe.net.  DS   4252 3 1ab15…
           SIG  DS (…) net. 5612
```

$ORIGIN ripe.net.

```
ripe.net. KEY (…) rwx002…  (4252) ; KSK
          KEY (…) sovP42…  (1111) ; ZSK
          SIG  KEY (…) 4252 ripe.net.  5tUcwU…

www.ripe.net.  A 193.0.0.202
               SIG  A  (…)  1111 ripe.net.  a3Ud…
```

RFC3090 Terminology

- Verifiable Secure
  - RRset and it’s SIG can be verified with a KEY that can be chased back to a trusted key, the parent has a DS record
- Verifiable Insecure
  - RRset sits in a zone that is not signed and for which the parent has no DS record (more next slide)
- BAD
  - RRset and its SIG can not be verified (somebody messed with the sig, the RRset, or the SIG expired)
  - A zone and it’s subzones are BAD when the parent’s SIG over the Child’s key is BAD

Jump to first page

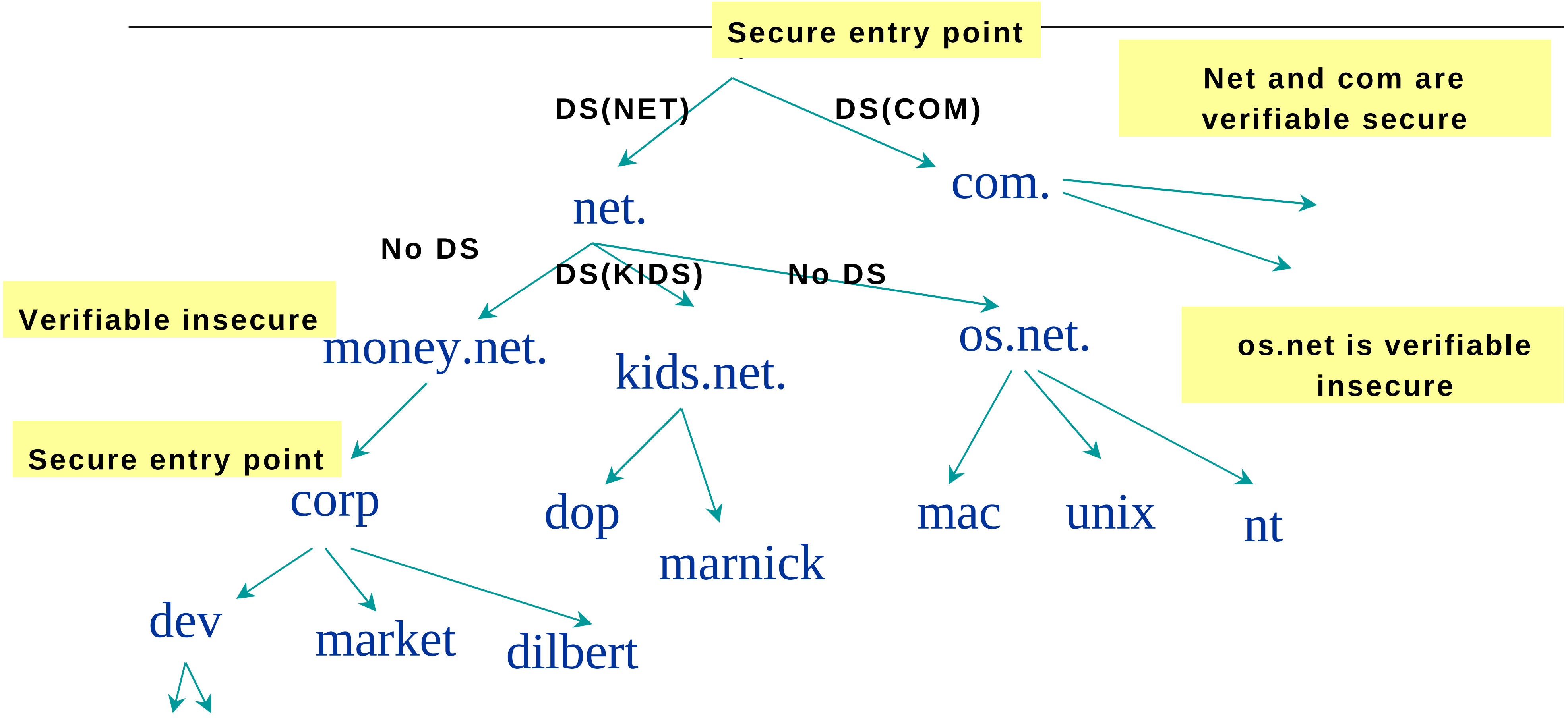
Insecure Children

- Cryptographic evidence for the verifiably insecure zone status is given by parent
- If there is no DS record as proved by a NXT record with valid signature, the child is not secured
- A child may contain signatures but these will not be used when building a chain of trust

- In RFC2535 the parent has a “NULL” key with a signature

Jump to first page

Illustrated Terminology



Resolver has key of root and corp.money.net configured as secure entry points

Building the Chain of Trust

- The child has to:
  - be secure (see “Signing the local zone”)
  - upload (off-band) the KSK to the parent
- The parent has to:
  - generate the DS record from the KSK of the child
  - sign the DS record with his own ZSK (re-sign his zone)

- Then the parent has to repeat the process, going to his own parent, and so on, till the "." (root)

All of this is done automatically - using tools

Jump to first page

Parental signature
adopting orphans carefully…

- Parents needs to check if the child KEY is really their child's… Did you get the KEY from the source authoritative for the child zone?
- This needs an out-of-DNS identification

Open operational issue:

- How do you identify the KEY comes from an authoritative source?
  - Billing information?
  - Phone call?
  - Secret token exchange via surface mail?

Jump to first page

The DNS is not a Public Key Infrastructure (PKI)

- All procedures on the previous slide are based on local policy i.e. policy set by the zone administrator
- A PKI is as strong as it’s weakest link, we do not know the strength of the weakest link
  - Certificate Authorities control this by SLAs
- If the domain is under one administrative control you might be able to enforce policy

Jump to first page

The DNS is not a PKI (cont’d)

- The DNS does not have Certificate Revocation Lists
  - There is no way to explicitly say: Do not trust that KEY

- But it is closest to a globally secured distributed DB
  - IPsec distribution of key material
    - opportunistic keys; if there is a key in the DNS and nothing better we’ll use it
  - discussions on using the DNS for key distribution
  - <keydist@cafax.se>

Jump to first page ⇦ ⇨

DS: Questions?

Jump to first page

Key Exchange and Rollovers

Why Key Exchange

- You have to keep your private key secret
- Private key can be stolen
    - Put the key on stand alone machines or on bastion hosts behind firewalls and strong access control
- Private key reconstruction (crypto analysis)
    - random number not random
    - Leakage of key material (DSA)
    - Brute force attacks

Private Key Compromise

- Try to minimize impact
  - Short validity of signatures
  - Regular key-rollover

- Remember: KEYs do not have timestamps in them -- the SIG over the KEY has the timestamp

- Key exchange involves 2nd party:
  - State to be maintained during rollover
  - operationally more expensive

Jump to first page

Short Signature Life Time

- Short parent signature over DS RR protects child
- Order 1 day possible

```
www.ripe.net.   3600 IN  SIG   A 1 3 3600 20010504144523 (
              20010404144523 3112 ripe.net.
              VJ+8ijXvbrTLeoAiEk/qMrdudRnYZM1VlqhN
              vhYuAcYKe2X/jqYfMfjfSUrmhPo+0/GOZjW
              66DJubZPmNSYXw== )
```

Signature expiration

Jump to first page

Key Rollover (part 1)

- Scheduled rollover of the child’s Key Signing Key
- Child replaces key-1 with key-2 and wants parent to sign it

```
$ORIGIN net.

kids NS   ns1.kids
     DS  (…) 1
     SIG KEY (…)net.
```

old parent zone

```
$ORIGIN kids.net.
@ NS   ns1
 KEY (…)       (1)
 KEY (…)       (5)
 SIG KEY (…) kids.net. 1
 SIG KEY (…) kids.net. 5
ns1     A       127.0.10.3
 SIG A (…) kids.net. 5
```

old child zone

```
    $ORIGIN kids.net.
    @ NS   ns1
       KEY (…)  (1)
a)     KEY (…)  (2)
       KEY (…)  (5)
       SIG KEY (…) kids.net. 1
b)     SIG KEY (…) kids.net. 2
       SIG KEY (…) kids.net. 5
       ns1   A  127.0.10.3
       SIG A (…) kids.net. 5
```

a) Create key 2

b) Sign key-set with key 1 and 2
and send key 2 to parent

Key Rollover (part 2)

c) Parent generates and signs DS record

d) Child signs his zone with **only** key 2, once parent updated his zone

```
$ORIGIN net.

kids NS   ns1.kids
     DS   (…) 2
     SIG KEY (…)net.
```

```
$ORIGIN kids.net.

@ NS   ns1
  KEY (…) 2
  KEY (…) 5
  SIG KEY (…) kids.net. 2
  SIG KEY (…) kids.net. 5
ns1   A  127.0.10.3
      SIG A (…) kids.net. 5
```

Jump to first page ⇦ ⇨

Timing of the Scheduled Key Rollover

- Child should not remove the old key while there are still servers handing out the old DS RR.
- The new DS will need to be distributed to the slave servers
  - max time set by the SOA expiration time
- The old DS will need to have expired from caching servers.
  - Set by the TTL of the original DS RR.
- You (or your tool) can check for the master and slave to have picked up the change.

Jump to first page

Scheduled Key Rollover Issues

- Currently one can not distinguish between a key signing key and a zone signing key.
- Once that distinction can be made, the rollover can be fully automated.

Jump to first page ⇦ ⇨

Unscheduled Rollover Problems

- Needs out of band communication with the parent and to pre-configured resolvers
- The parent needs to establish your identity out of band again
- Your children need protection. How to protect them best? Leaving them unsecured?
- There will be a period that the stolen key can be used to generate data useful on the Internet
- There is no ‘revoke key’ mechanism
- Emergency procedure must be on the shelf

Jump to first page

Key Rollover: Questions?

Jump to first page

DNSSEC - Conclusions

What Did We Learn

- DNSSEC provides a mechanism to protect DNS
- DNSSEC implementation:
  - TSIG for servers
  - SIG, KEY and NXT for data
- DNSSEC main difficulties:
  - keeping private key safe
  - distributing keys

Jump to first page

Open Issues
(the where-shall-I-put-it slide)

DNSSEC is still a moving target…

- RFC 2535 rewrite
- NXT/OPT-IN
- Delegation Signer (DS)
- BIND development
  - Current bind snapshots have bugs in DNSSEC.
- Operational issues
  - Webfarms and keymanagement
  - NXT RR walk and privacy
- API resolver<->cache

Jump to first page

End of Part I...
Questions???

Jump to first page

PART II

DNSSEC Operations

Description of tools