DNSSEC: Security Extensions for DNS

Introduction

DNSSEC: Security extensions for DNS

Let's start with some DNS and DNSSEC theory

Mauricio Vergara Ereche

LACNIC30 & LACNOG 2018
Sep 2018

Agenda

- Recap: How DNS works
- Understanding DNSSEC
  - What DNSSEC solves and what not
  - Brief reminder on Cryptography
  - How DNSSEC works
- New Concepts
  - Secure Entry Points and Chain of trust
  - New RRsigs and flags
  - Key Rollovers

Recap: How DNS Works

Part 1

DNS in a nutshell

- DNS is a distributed database
- There are 2 types of DNS servers
  - DNS Authoritative
    - Master
    - Slaves
  - DNS Resolver
    - Recursive
    - Cache
    - Stub resolver

DNS Resource Records (RR)

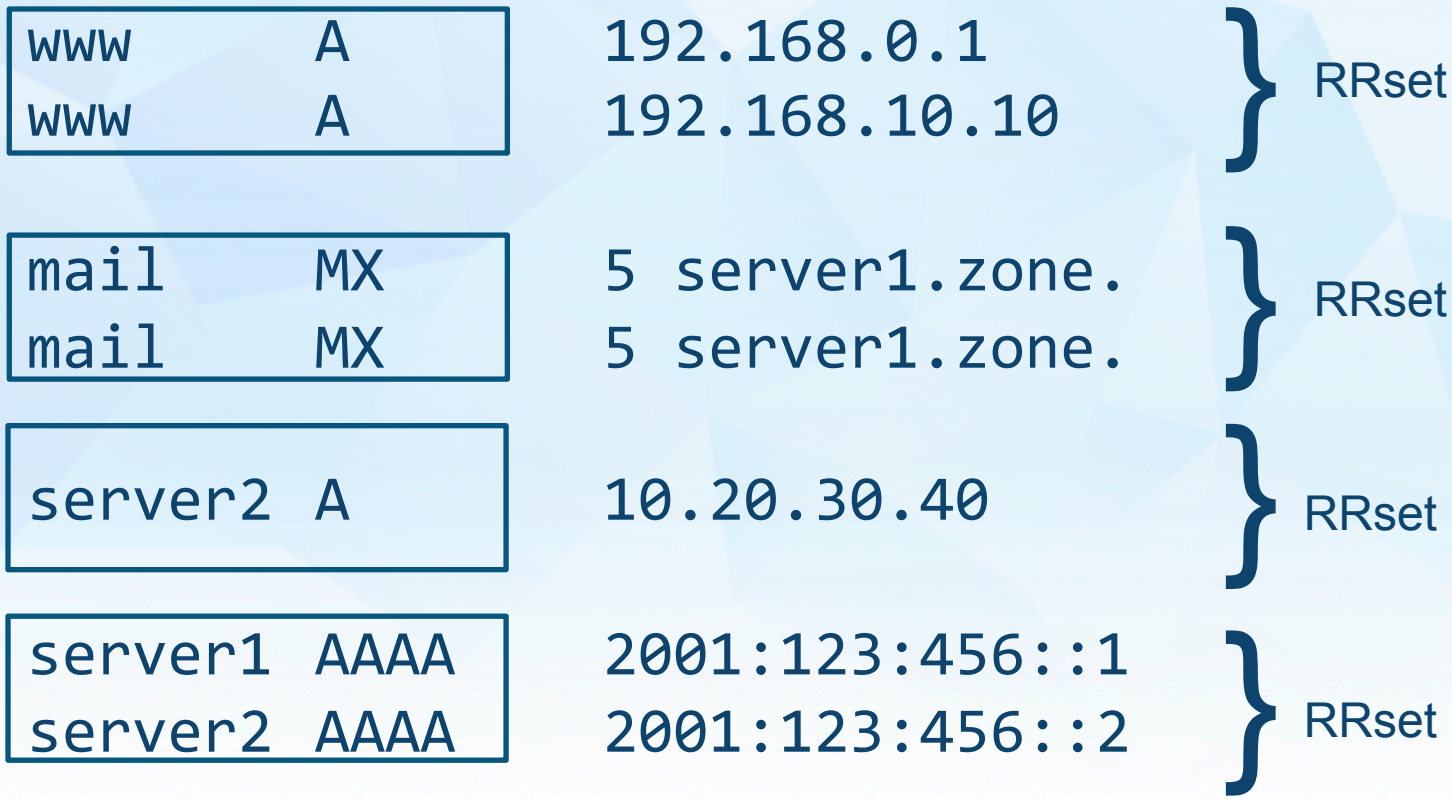
- Unit of data in the Domain Name System
- Define attributes for a domain name

Label	TTL	Class	Type	Data
www	3600	IN	A	192.168.0.1

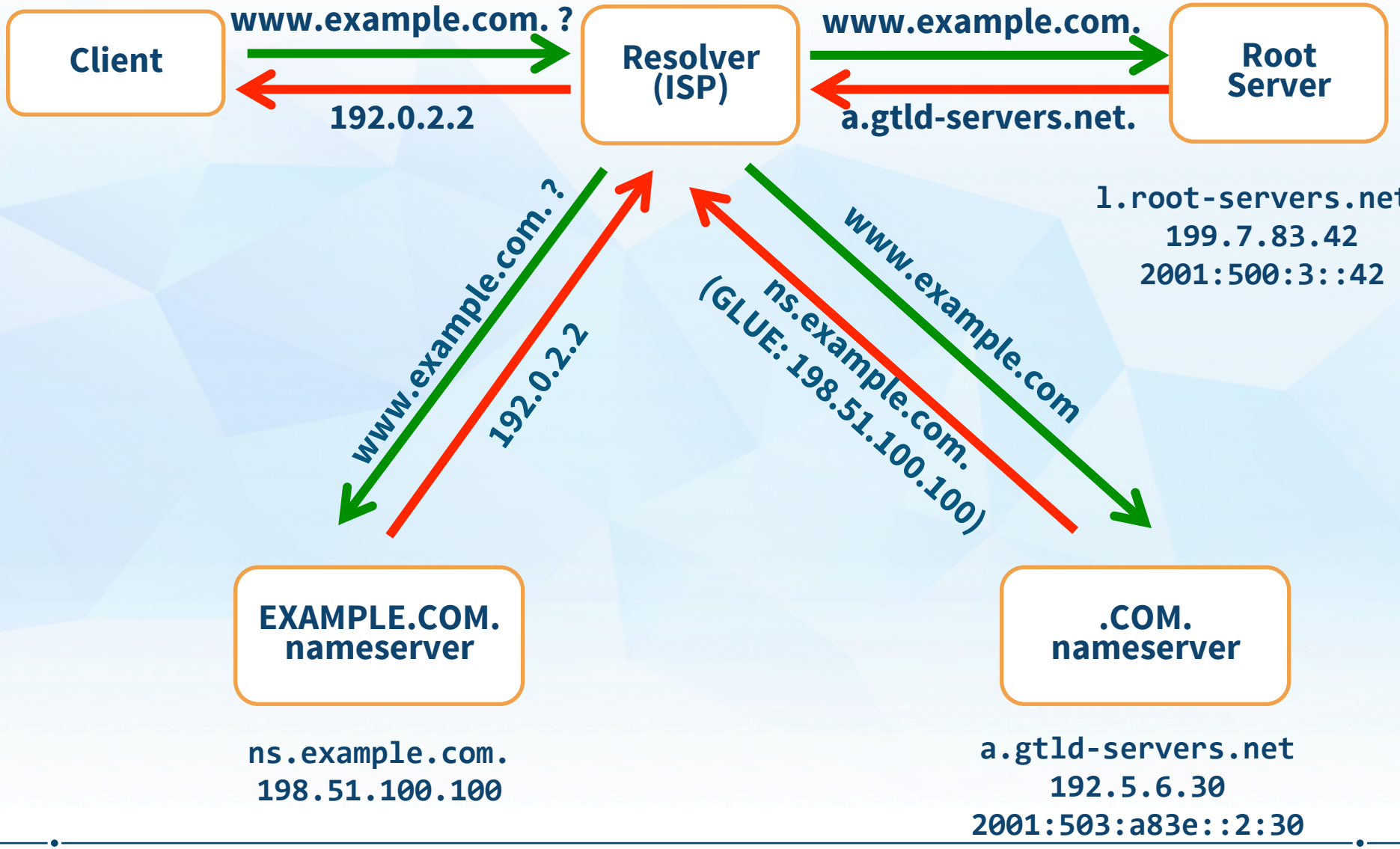
- Most common types of RR
  - A
  - AAAA
  - NS
  - SOA
  - MX
  - CNAME

DNS Resource Records sets (RRsets)

- Multiple RR with same LABEL and TYPE are grouped into Resource Record Sets (RRsets)



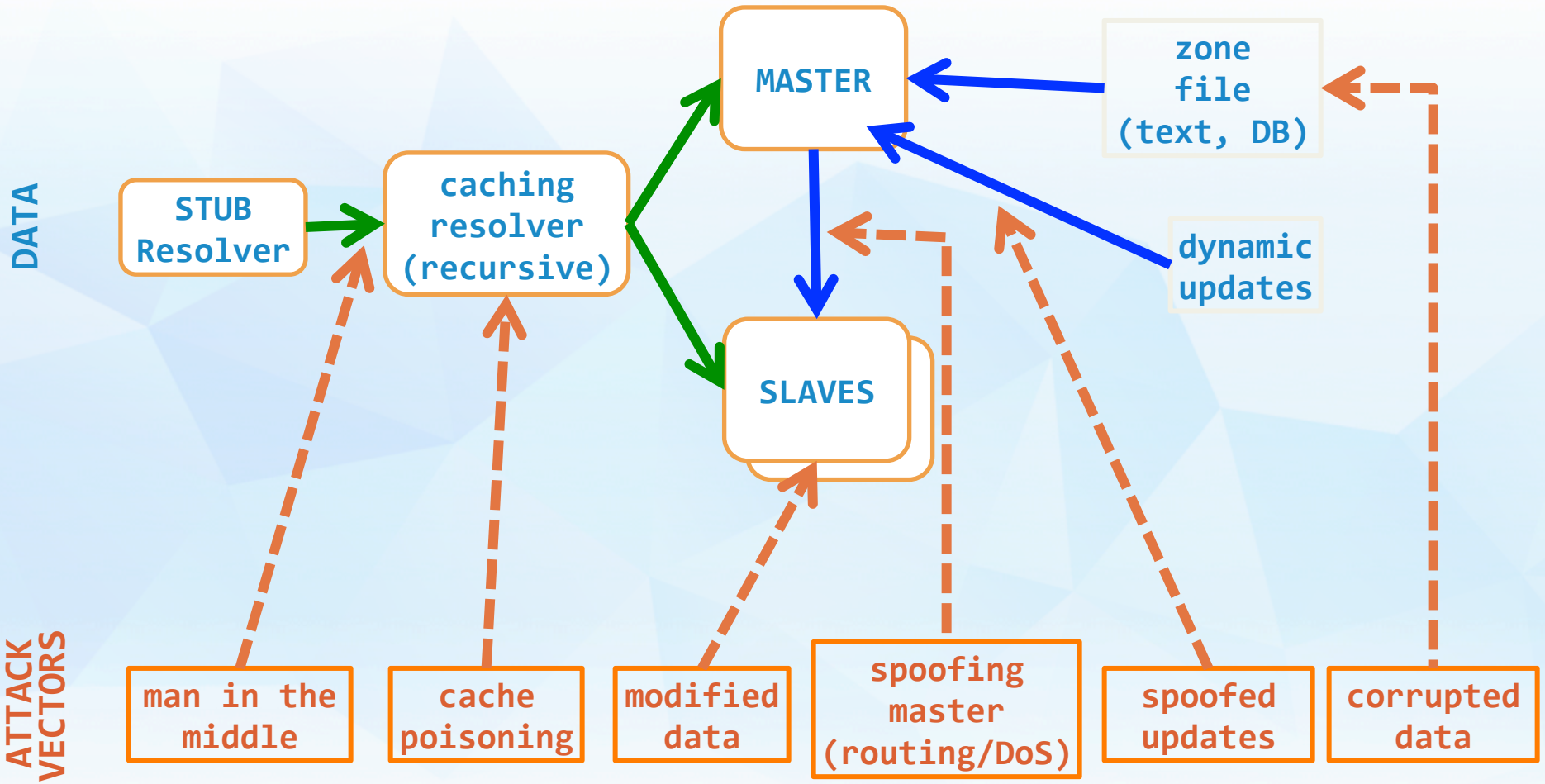
Normal DNS Flow



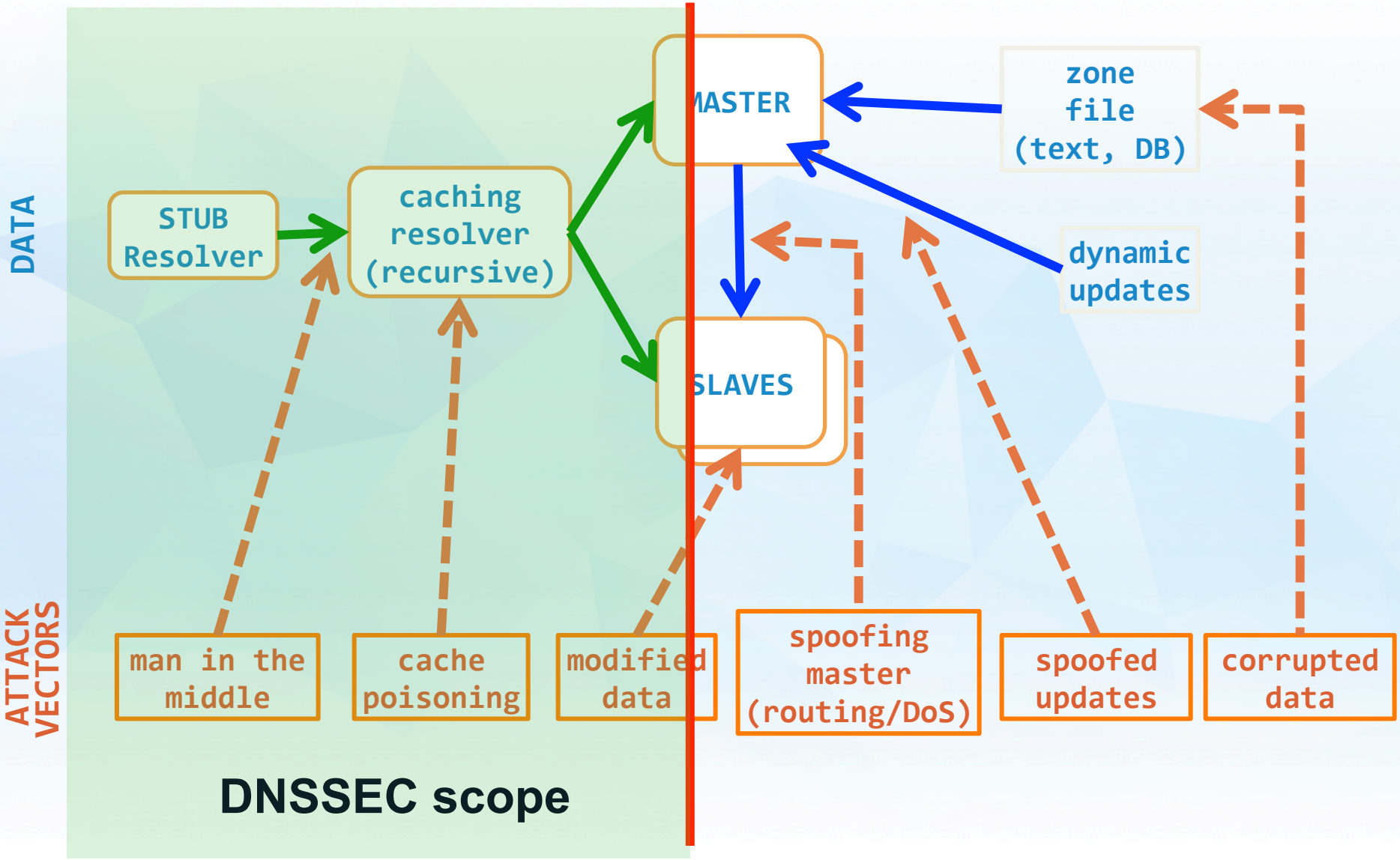
Understanding DNSSEC

Part 2

DNS Data flow



What DNSSEC solves and what not

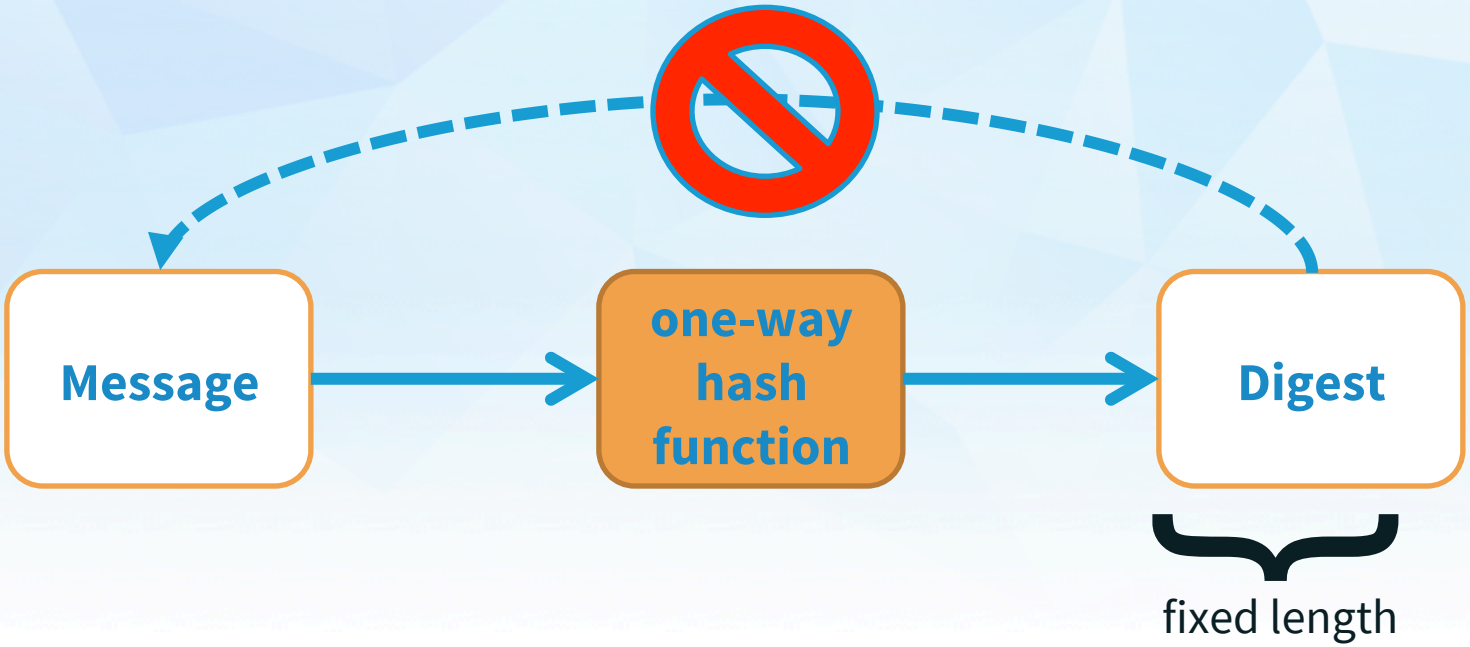


Brief reminder on cryptography

- Nowadays most of our Security Services are based in one (or a combination) of the following areas:
  - One-way hash functions
  - Symmetric key crypto
  - Public-key crypto (or asymmetric)

One-way hash functions

- Takes a **message** of any length as input
- Delivers a short and fixed length output (usually called **hash** or **digest**)
- It can be used as a *digital signature*.

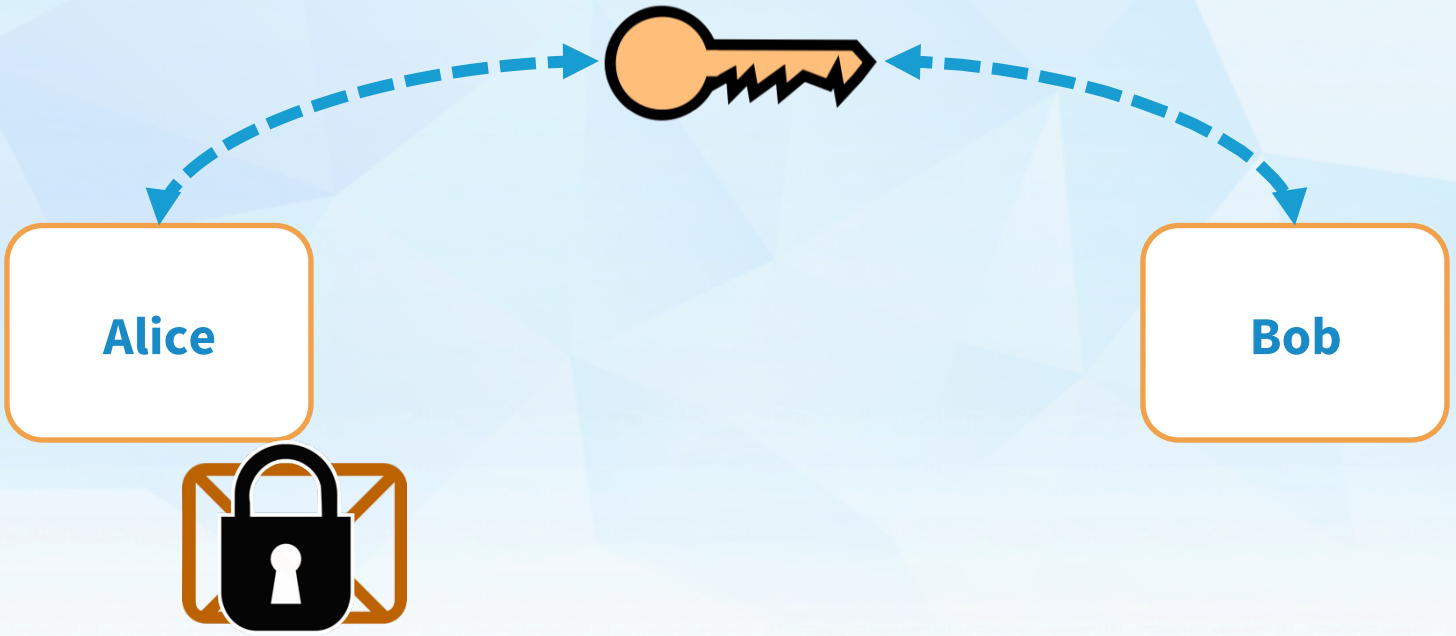


One-way hash functions

- Most common
  - MD5
  - SHA-1
  - SHA-2
  - GOST
  - HAVAL
  - DES
  - checksum
  - CRC{16,32,64}

Symmetric key crypto

- Both sender and receiver share a key.
- Key is used to encrypt a message, which can be decrypted by the same key



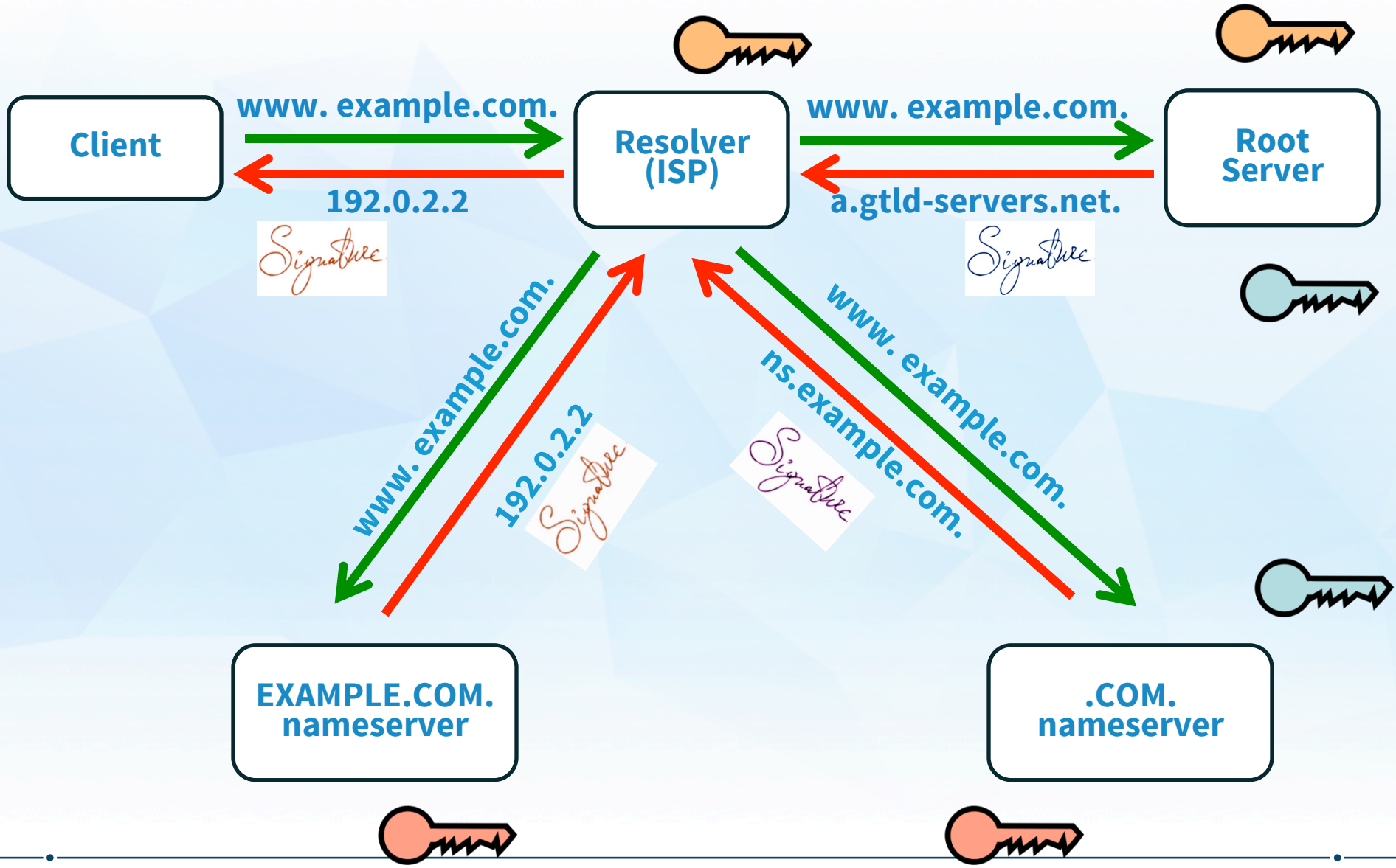
Asymmetric or Public Key crypto

- Sender has a pair of keys. One is private and the other one is public.
- Each key is able to encrypt a message, while the other key can do the opposite.

Public Key crypto functions

- RSA (Riverst Shamir Adleman)
- DSA (Digital Signature Algorithm)
- ElGamal
- ECDSA (Elliptic Curve Digital Signature Algorithm)

How DNSSEC Works



How DNSSEC Works (part 2)

- Data authenticity and integrity by signing the Resource Records Sets with a private key
- Public DNSKEYs published, used to verify the RRSIGs
- Children sign their zones with their private key
  - Authenticity of that key established by parent signing hash (DS) of the child zone's key
- Repeat for parent…
- Not that difficult on paper
  - Operationally, it is a bit more complicated
  - DS_{KEY} → KEY –signs→ zone data

DNSSEC: New concepts

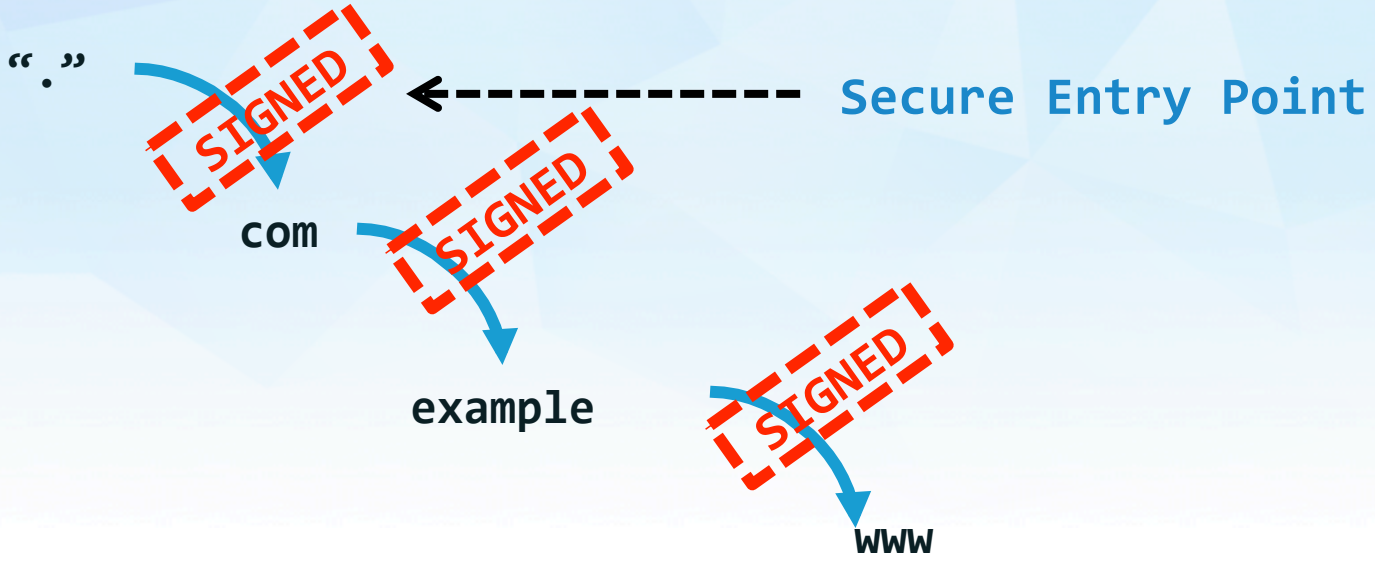
Part 3

New Concepts

- Secure Entry Point and Chain of Trust
  - Delegating Signing Authority
- New packet options (flags)
  - CD, AD, DO
- New RRs
  - DNSKEY, RRSIG, NSEC/NSEC3 and DS
- Signature expiration
- Key Rollovers

Chain of trust and Secure Entry Point

- Using the existing delegation based model of distribution/
- Don’t sign the entire zone, sign a RRset
- Parent **DOES NOT** sign the child zone. The parent signs a pointer (hash) to the key used to sign the data of the child zone (DS record)
- Example with **`www.example.com.`**



New Fields and Flags

- DNSSEC Updates DNS protocol at the packet level
- Non-compliant DNS recursive servers ***should*** ignore these:
  - **CD**: **C**hecking **D**isabled (ask recursing server to not perform validation, even if DNSSEC signatures are available and verifiable, i.e.: a SEP can be found)
  - **AD**: **A**uthenticated **D**ata, set on the answer by the validating server if the answer could be validated, and the client requested validation
  - **DO**: **D**NSSEC **O**K. A new `EDNS0` option to indicate that client supports DNSSEC options

DNSSEC: New Resource Records

Part 4

New RRs

- Adds 5 (five) new DNS Resource Records:
  1. **DNSKEY**: Public key used in zone signing operations.
  2. **RRSIG**: RRset signature
  3. **NSEC** &
  4. **NSEC3**: Returned as verifiable evidence that the name and/or RR type does not exist.
  5. **DS**: Delegation Signer. Contains the hash of the public key used to sign the key which itself will be used to sign the zone data. Follow DS RR's until a ”trusted” zone is reached (ideally the root).

New RR: DNSKEY

- FLAGS determines the usage of the key (ZSK or KSK)
- PROTOCOL is always 3 (DNSSEC)
- ALGORITHM can be (3: DSA/SHA-1, 5: RSA/SHA1, 8: RSA/SHA-256, 12: ECC-GOST)
  - http://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xml

DNSKEY: Two keys, not one…

- There are in practice at least **two** DNSKEY *pairs* for every zone
- Originally, **one** key-pair (public, private) defined for the zone
  - **private**: key used to sign the zone data (RRsets)
  - **public**: key published (DNSKEY) in the zone
- DNSSEC works fine with a single key pair
- Problem with using a single key:
  - Every time the key is updated, the DS record must be updated on the parent zone as well
  - Introduction of **K**ey **S**igning **K**ey (flags=257)

KSK and ZSK

- Key Signing Key (KSK)
  - Pointed to by parent zone in the form of DS (Delegation Signer). Also called Secure Entry Point.
  - Used to sign the Zone Signing Key
  - Flags: 256
- Zone Signing Key (ZSK)
  - Signed by the KSK
  - Used to sign the zone data RRsets
  - Flags: 257
- This decoupling allows for independent updating of the ZSK without having to update the KSK, and involve the parents (i.e. less administrative interaction)

New RR: RRSIG (Resource Record Signature)

```
example.com.   600   A   192.168.10.10
example.com.   600   A   192.168.23.45
```

```
                                        TYPE COVERED   #LABELS
 OWNER                    TYPE                ALG              TTL
example.com.   600   RRSIG      A      7      2      600   (
```

```
 SIG. EXPIRATION    SIG. INCEPTION     KEY ID   SIGNER NAME
20180115154303   20171017154303   23807   example.com.
```

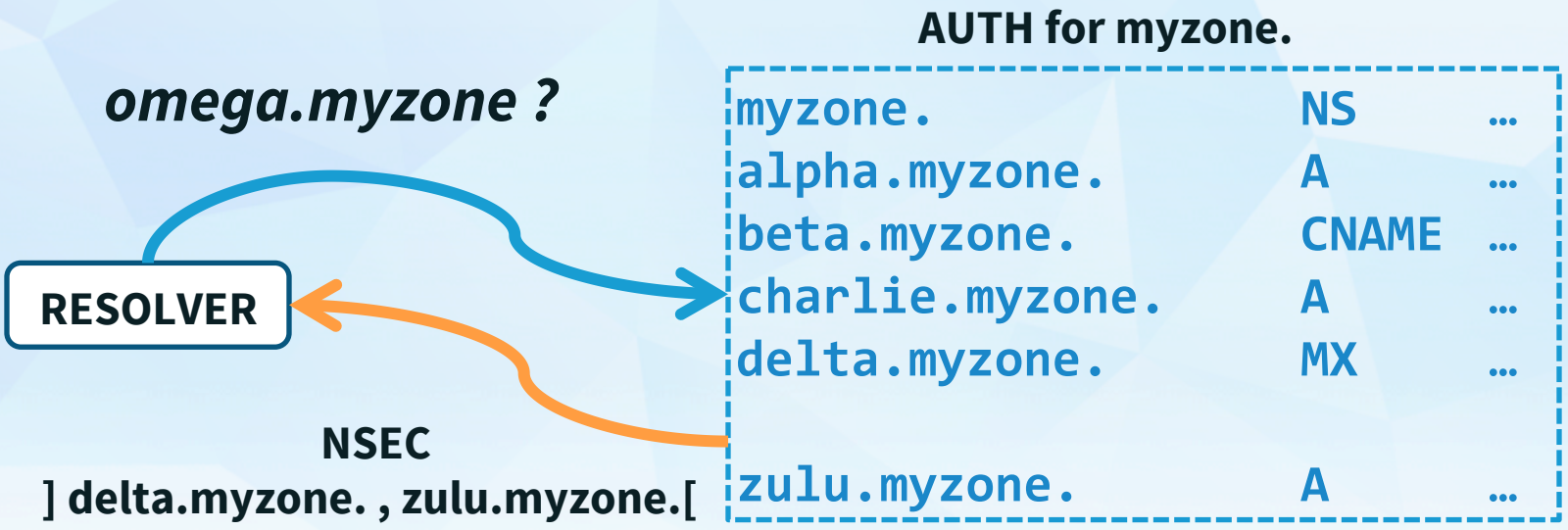
```
 SIGNATURE
CoYkYPqE8Jv6UaVJgRrh7u16m/cEFGtFM8TArbJdaiPu
W77wZhrvonoBEyqYbhQ1yDaS74u9whECEe08gfoe1FGg
. . .
)
```

RRSIG

- Typical default values
  - Signature inception time is 1 hour before.
  - Signature expiration is 30 from now
  - Proper timekeeping (NTP) is required
- What happens when signatures run out?
  - SERVFAIL
  - Domain effectively disappears from the Internet for validating resolvers
- Note that *keys* do **not** expire
- No all RRSets need to be resigned at the same time

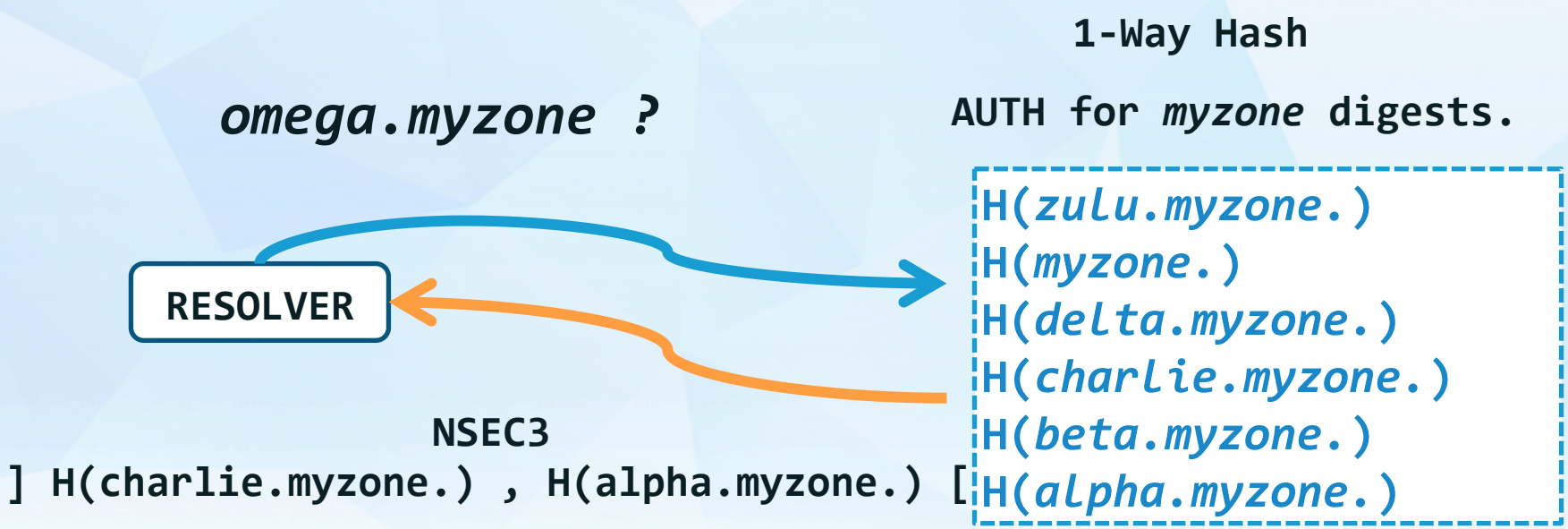
New RR: NSEC

- NXDomains also must be verified
- NSEC provides a pointer to the **N**ext **SEC**ure record in the chain of records.



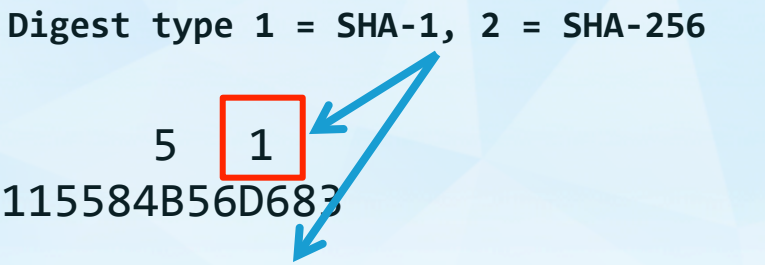
New RR: NSEC3

- To avoid concerns about “zone enumeration”
- To avoid large zone-files: opt-out concept



New RR: DS (Delegation Server)

- Hash of the KSK of the child zone
- Stored in the parent zone, together with the NS RRs indicating a delegation of the child zone.
- The DS record for the child zone is signed together with the rest of the parent zone data
- NS records are NOT signed (they are a hint/pointer)




```
myzone.         DS      61138       5   1
F6CD025B3F5D03040895053 54A0115584B56D683

myzone.         DS      61138       5   2
CCBC0B557510E4256E88C01B0B1336AC4ED6FE08C8268CC1AA5FBF00
5DCE3210
```

Security status of Data

- **Secure**
  - Resolver is able to build a chain of signed DNSKEY and DS RRs from a trusted security anchor to the RRset
- **Insecure**
  - Resolver knows that it has no chain of signed DNSKEY and DS RRs from any trusted starting point to the RRset
- **Bogus**
  - Resolver believes that it ought to be able to establish a chain of trust but for which it is unable to do so
  - May indicate an attack but may also indicate a configuration error or some form of data corruption
- **Indeterminate**
  - No trust anchor to indicate if the zone and children should be secure.
  - Resolver is not able to determine whether the RRset should be signed.

Signatures expiration and Key Rollovers

Part 5

Signature expiration

- Signatures are per default 30 days (BIND)
- Need for regular re-signing:
  - To maintain a constant window of validity for the signatures of the existing RRset
  - To sign new and updated Rrsets
  - Use of jitter to avoid having to resign all expiring RRsets at the same time
- The keys themselves do NOT expire…
- But they may need to be rolled over...

Key Rollovers

- Try to minimize impact
  - Short validity of signatures
  - Regular key rollover
- Remember: DNSKEYs do not have timestamps
  - the RRSIG over the DNSKEY has the timestamp
- Key rollover involves second party or parties:
  - State to be maintained during rollover
  - Operationally expensive

Key rollovers

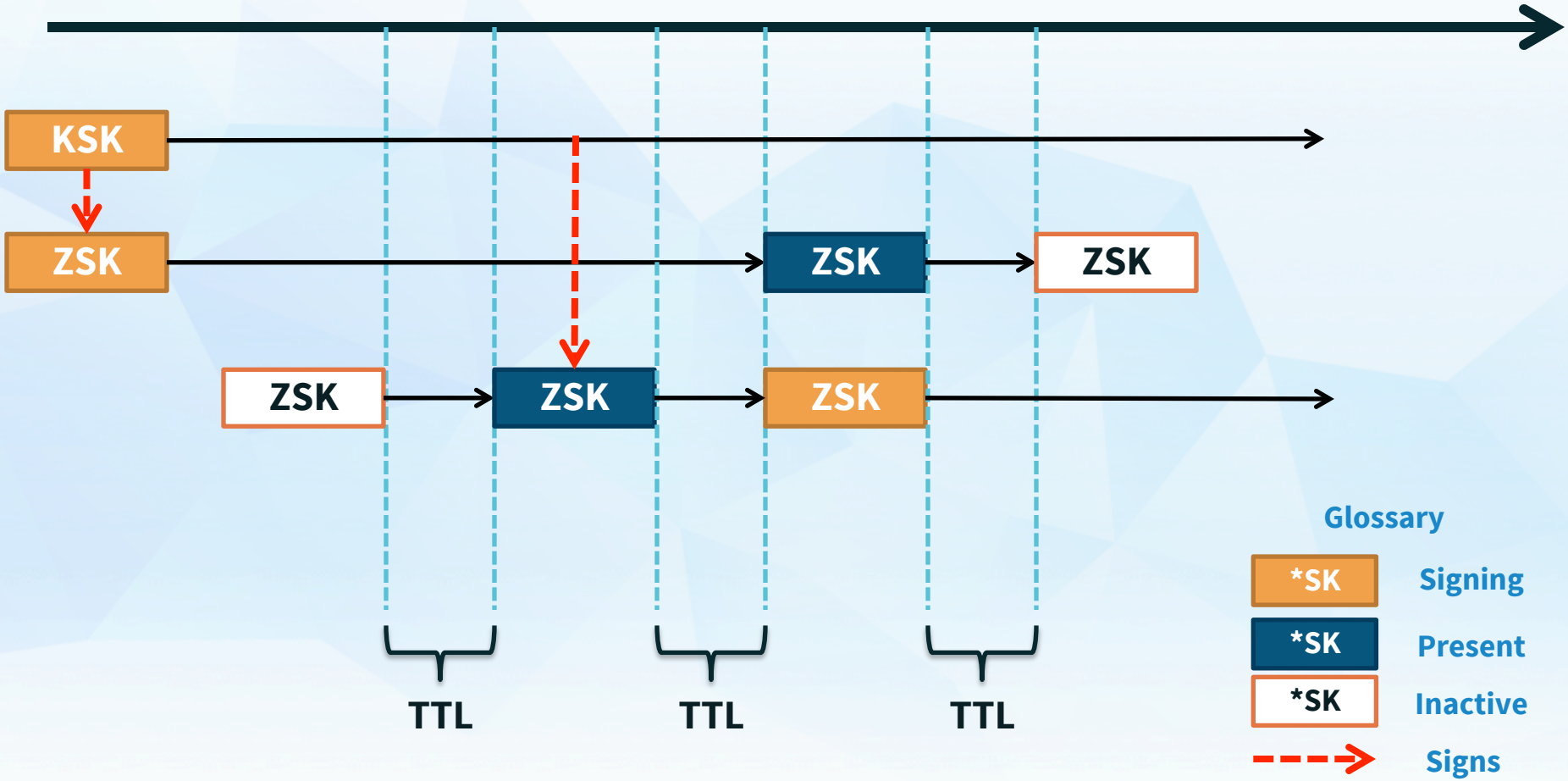
- Two methods for doing key rollover
  - Pre-Publish
  - Double Signature
- KSK and ZSK rollover use different methods.
  - Remember that KSK needs to interact with parent zone to update DS record.

Key Rollovers: Pre-Publish method

- ZSK Rollover using the pre-publish method

1. Wait for old zone data to expire from caches (TTL)
2. Sign the zone with the KSK and published ZSK
3. Wait for old zone data to expire from caches
4. Adjust Key list and sign the zone with new ZSK

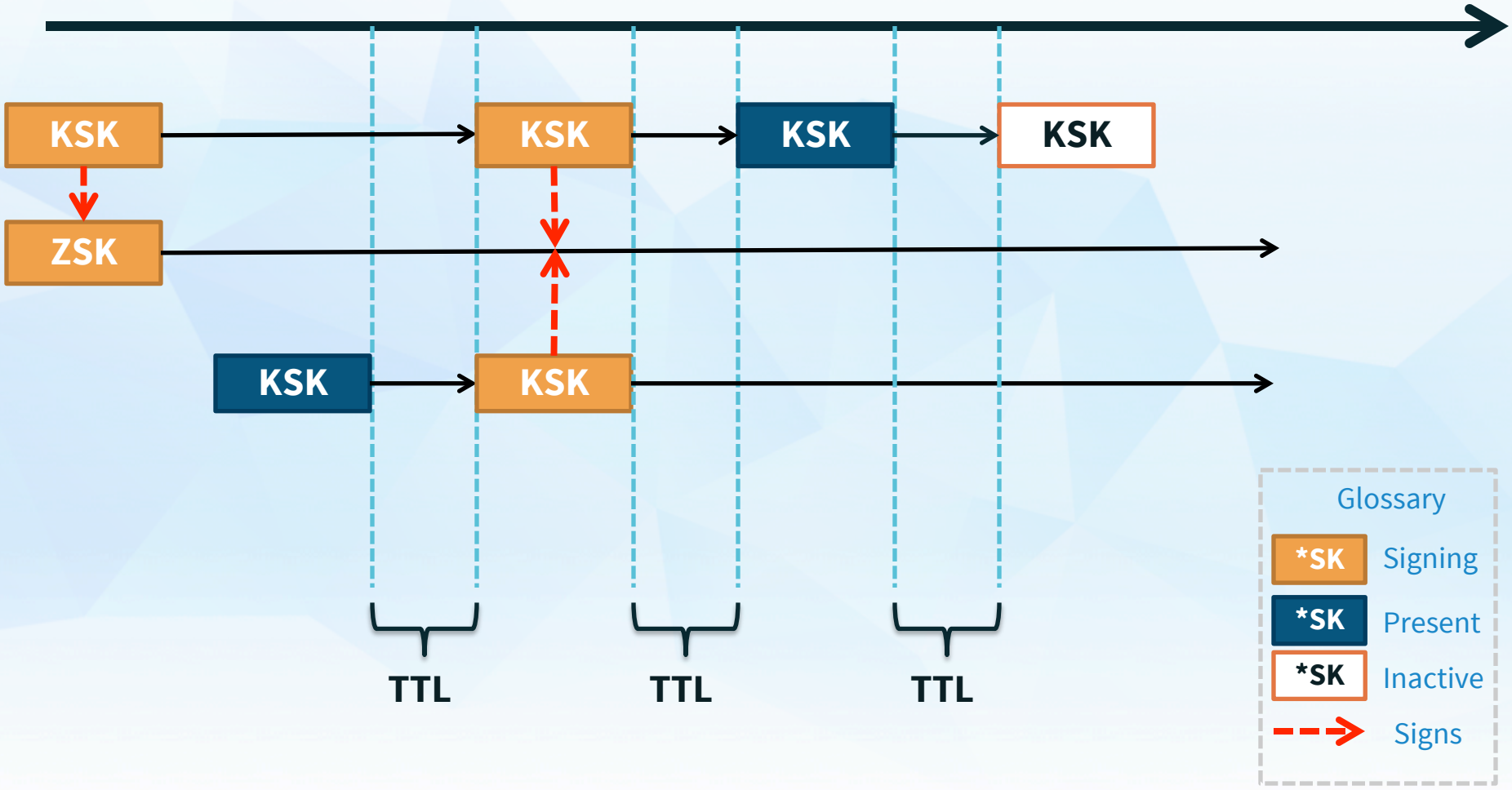
Key Rollovers: Pre-Publish method



Key Rollovers: Double Signature

- KSK Rollover using the Double Signature method
  1. Wait for old zone data to expire from caches
  2. generate a new (published) KSK
  3. Wait for the old DNSKEY RRset to expire from caches
  4. roll the KSKs
  5. Transfer new DS keyset to the parent
  6. Wait for parent to publish the new DS record
  7. Reload the zone
- It is also possible to use dual DS in the parent zone

Key Rollovers: Double Signature



Final thoughts

Useful links

- https://www.dnssec-deployment.org
- http://www.internetsociety.org/deploy360/dnssec
- http://dnssec-debugger.verisignlabs.com
- http://dnsviz.net
- http://www.dnssec-failed.org

ICANN

One World, One Internet

Visit us at **icann.org**

@icann

linkedin/company/icann

facebook.com/icannorg

slideshare/icannpresentations

youtube.com/icannnews

soundcloud/icann

flickr.com/icann

instagram.com/icannorg