DNS Security

bdNOG5 & ION Conference

7-11 April, 2016, Dhaka, Bangladesh

Issue Date:

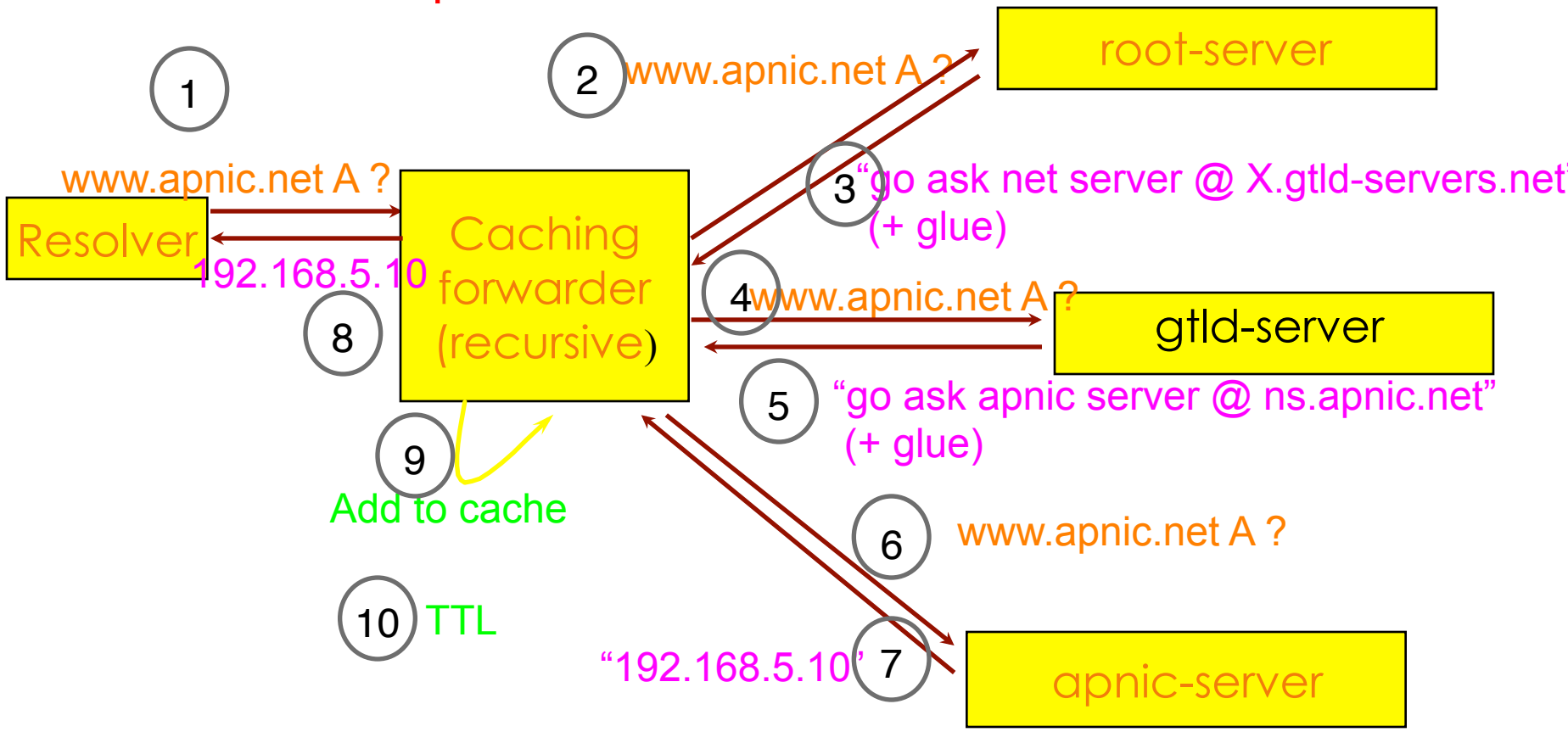
Revision:

APNIC

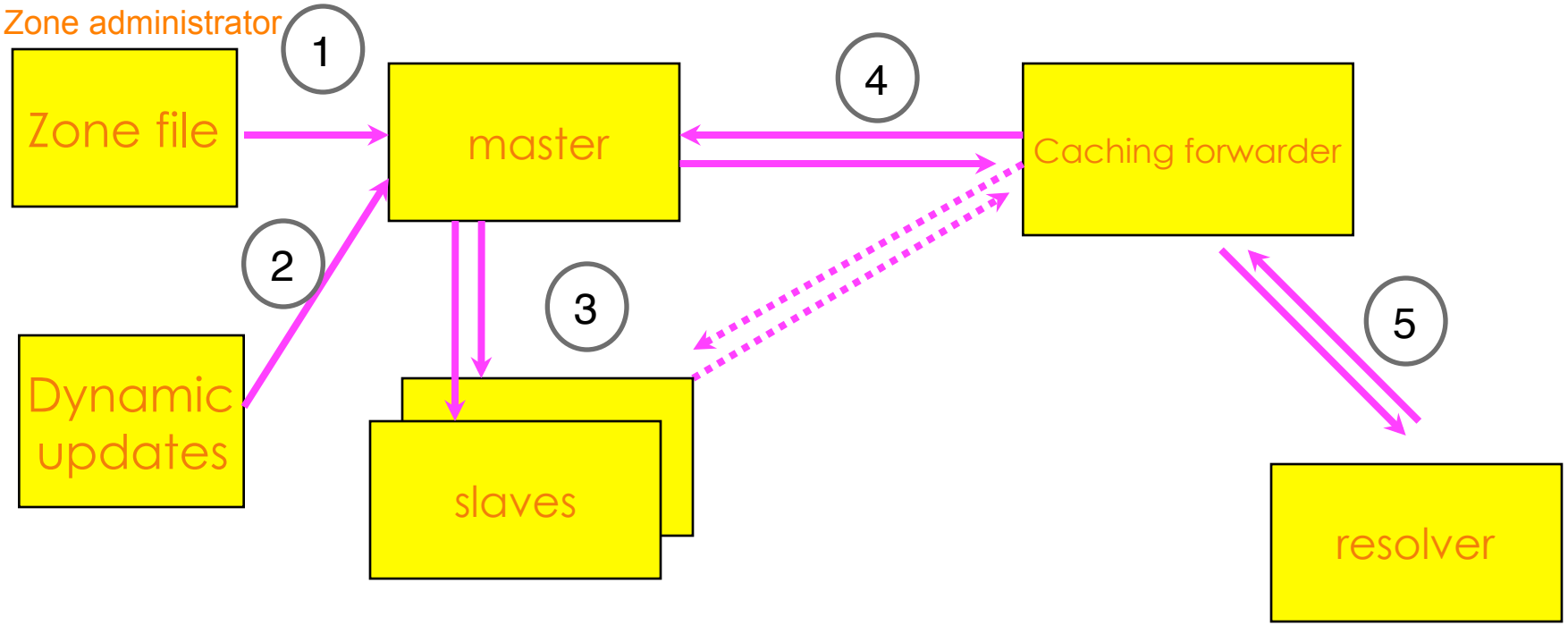
Overview

- How DNS Works
- DNS Vulnerabilities
- Securing the Nameservers
- Transaction Signature (TSIG)
- DNS Security Extensions (DNSSEC)

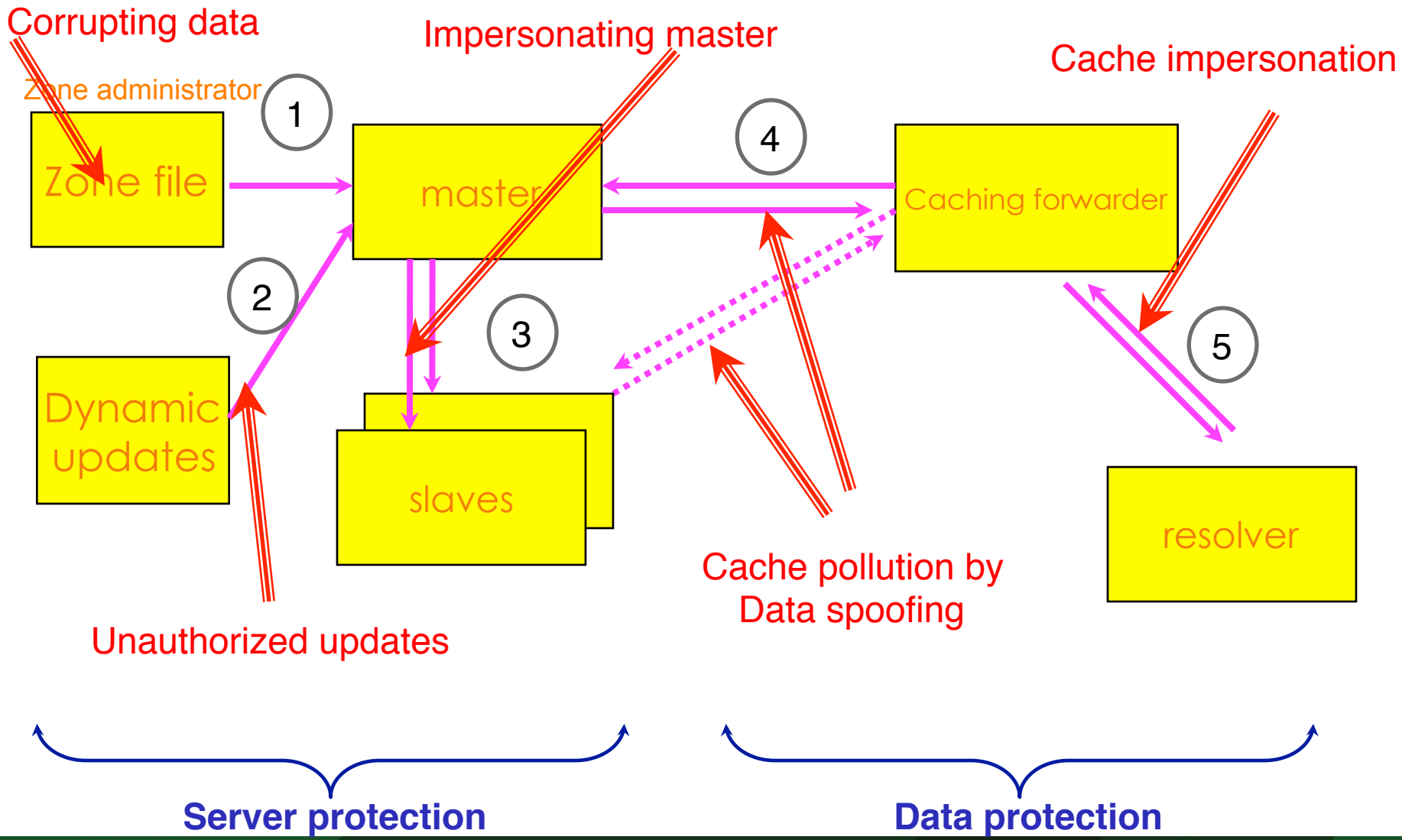
Overview: How DNS Works



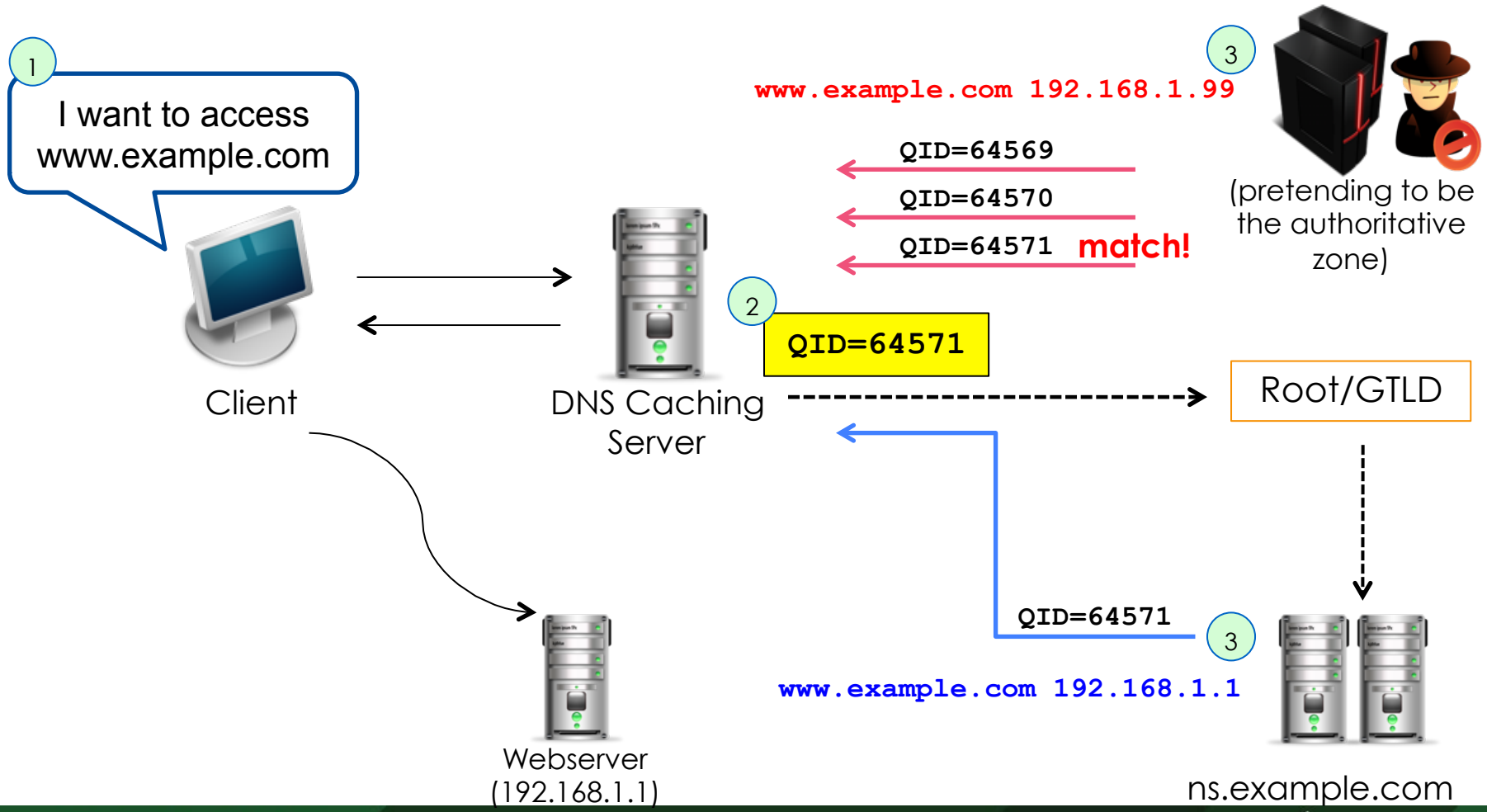
DNS Vulnerabilities



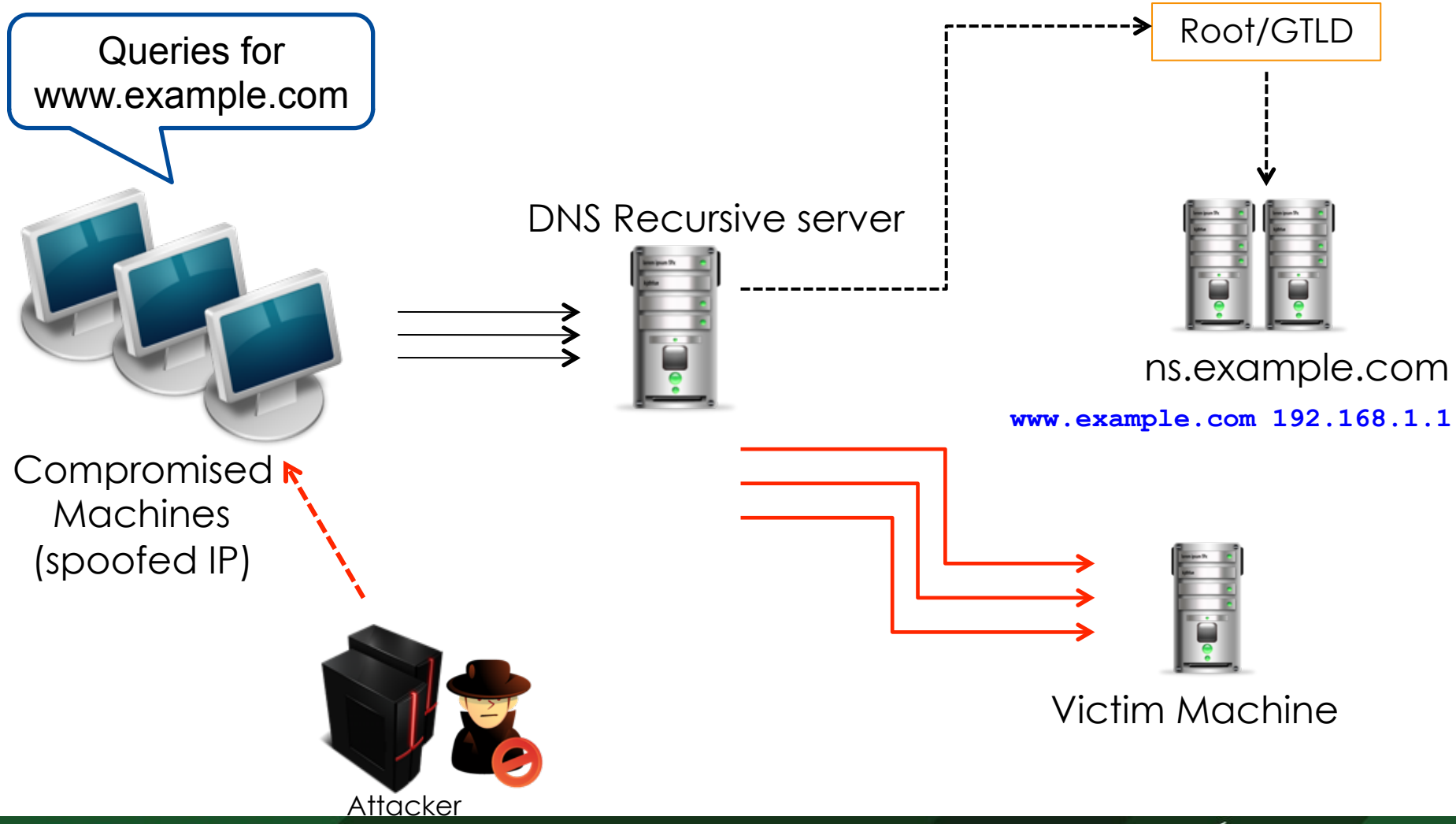
DNS Vulnerabilities



DNS Cache Poisoning



DNS Amplification



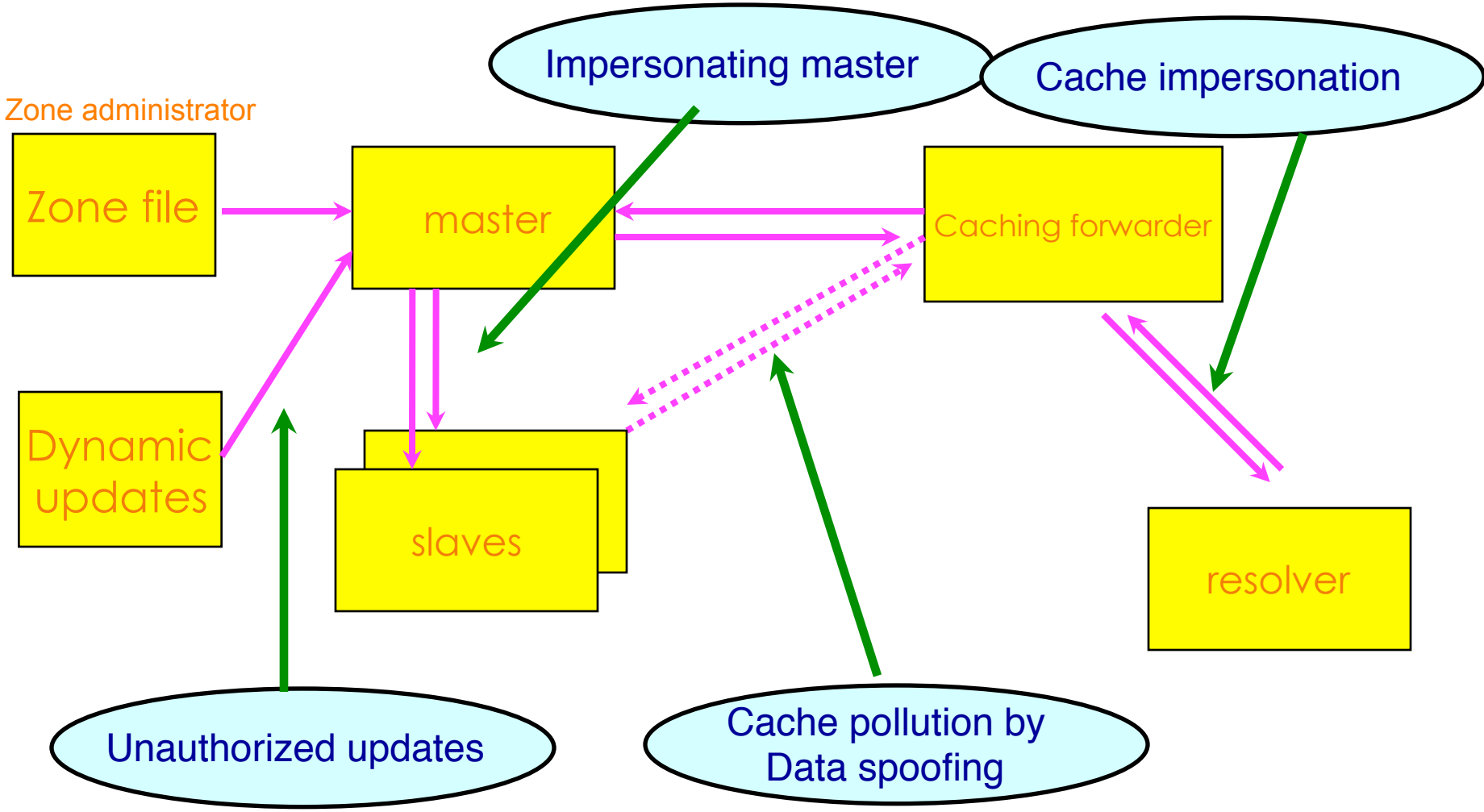
Securing the Nameserver

- Run the most recent version of the DNS software or apply the latest patch
- Restrict queries
- Prevent unauthorized zone transfers
- Run BIND with the least privilege (use `chroot`)
- Randomize source ports
- Secure the box
- Implement TSIG and DNSSEC

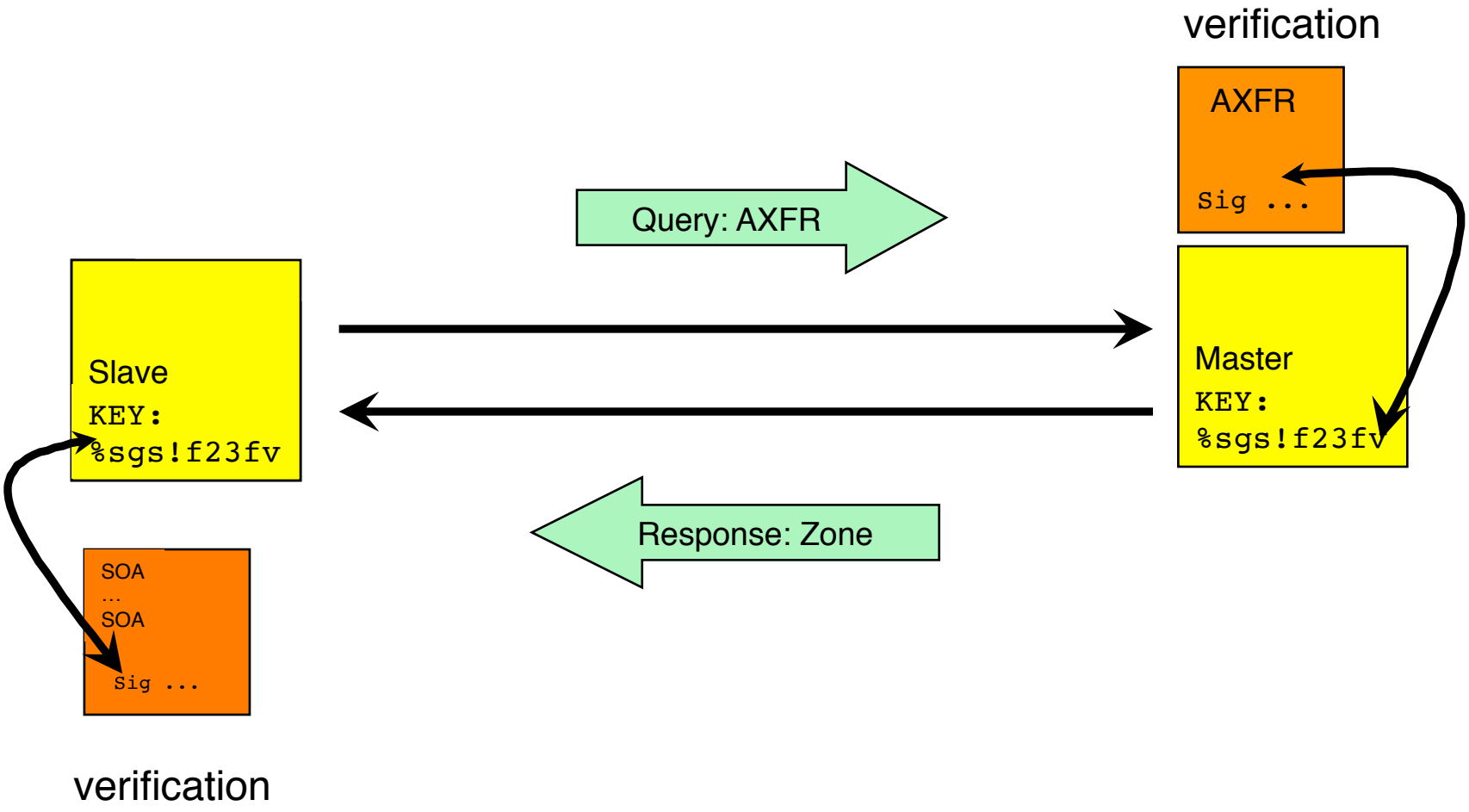
What is Transaction Signature?

- A mechanism for protecting a message from primary to secondary (and vice versa)
- Provides secure communication of queries and responses
  - Also protects zone transfers and dynamic updates
- How?
  - A keyed-hash is applied so recipient can verify the message source
- Based on a shared secret - both sender and receiver are configured with it

TSIG Protected Vulnerabilities



TSIG Example



TSIG Steps

- Generate secret
  - `dnssec-keygen -a <algorithm> -b <bits> -n host <name of the key>`
- Communicate secret
  - Transfer the key securely (ex. SSH/SCP)
- Configure the servers
  - Edit configuration file for primary and secondary
- Test
  - dig @<server> <zone> AXFR -k <TSIG keyfile>

Configuration Example – named.conf

Primary server 10.33.40.46

```
key ns1-ns2.pcx. net {
        algorithm hmac-md5;
        secret "APlaceToBe";
};
server 10.33.50.35 {
        keys {ns1-ns2.pcx.net;};
};
zone "my.zone.test." {
        type master;
        file “db.myzone”;
        allow-transfer {
        key ns1-ns2.pcx.net ;};
};
```

Secondary server 10.33.50.35

```
key ns1-ns2.pcx.net {
        algorithm hmac-md5;
        secret "APlaceToBe";
};
server 10.33.40.46 {
  keys {ns1-ns2.pcx.net;};
};
zone "my.zone.test." {
        type slave;
        file “myzone.backup”;
        masters {10.33.40.46;};
};
```

You can save this in a file and refer to it in the named.conf using ‘include’ statement:

```
include “/var/named/master/tsig-key-ns1-ns2”;
```

TSIG Testing - dig

- You can use dig to check TSIG configuration

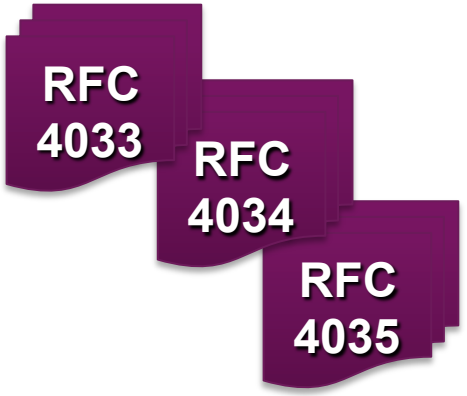
```
dig  @<server> <zone> AXFR -k <TSIG keyfile>
```

```
$ dig @127.0.0.1 example.net AXFR \
-k Kns1-ns2.pcx.net.+157+15921.key
```

- A wrong key will give “Transfer failed” and on the server the security-category will log this.
- Note: TSIG is time-sensitive

What is DNSSEC?

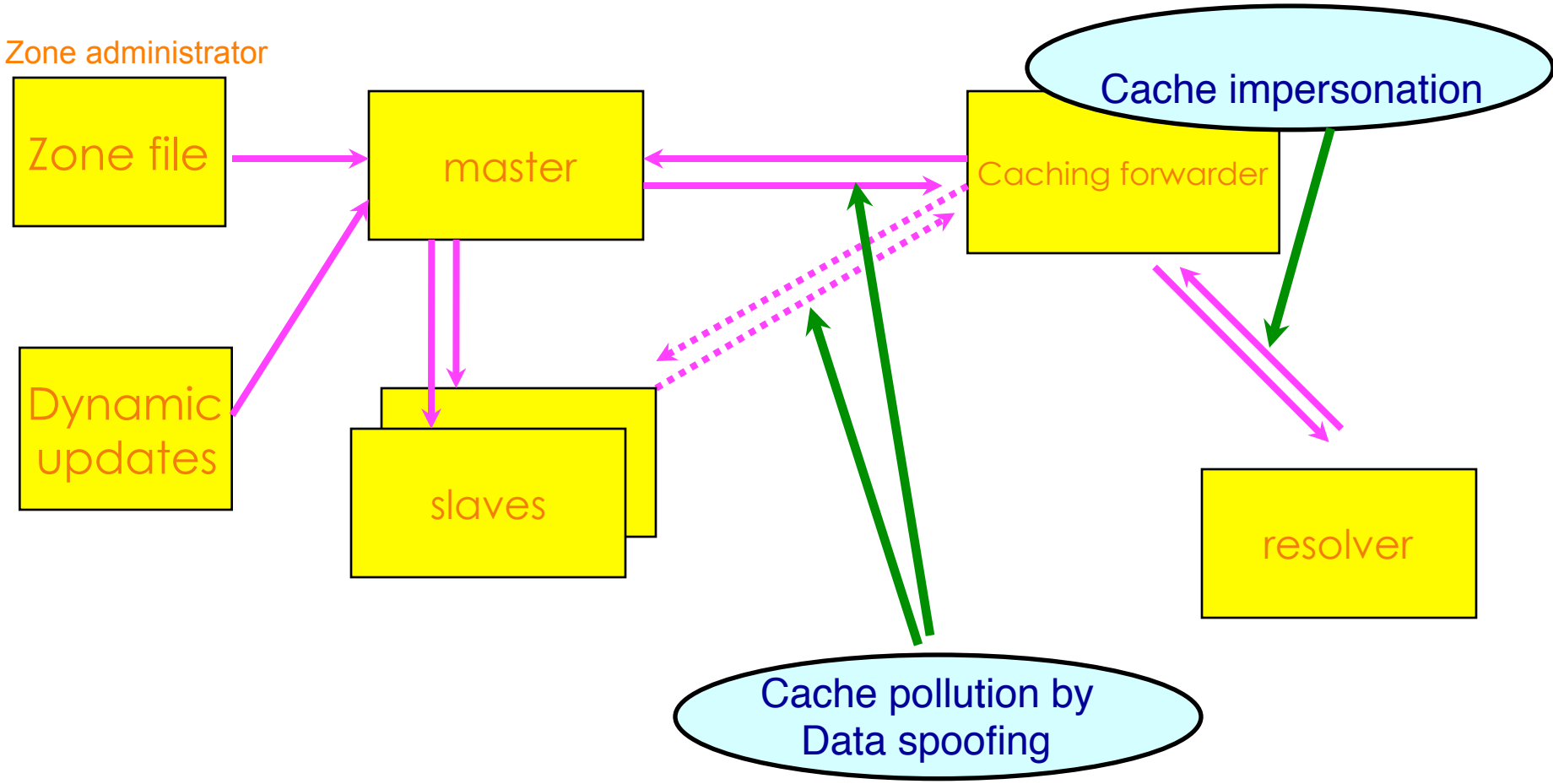
- **DNS Sec**urity Extensions
- Protects the integrity of data in DNS by establishing a chain of trust
- A form of digitally signing the data to attest its validity
- Uses public key cryptography – each link in the chain has a public/private key pair
- Guarantees
  - Authenticity
  - Integrity
  - Non-existence of a domain



How DNSSEC Works

- Records are signed with private key to prove its authenticity and integrity
- The signatures are published in DNS
- Public key is also published so record signatures can be verified
- Child zones also sign their records with their private key
- Parent signs the hash of child zone's public key to prove authenticity

Vulnerabilities protected by DNSSEC



New Resource Records

RFC 4034

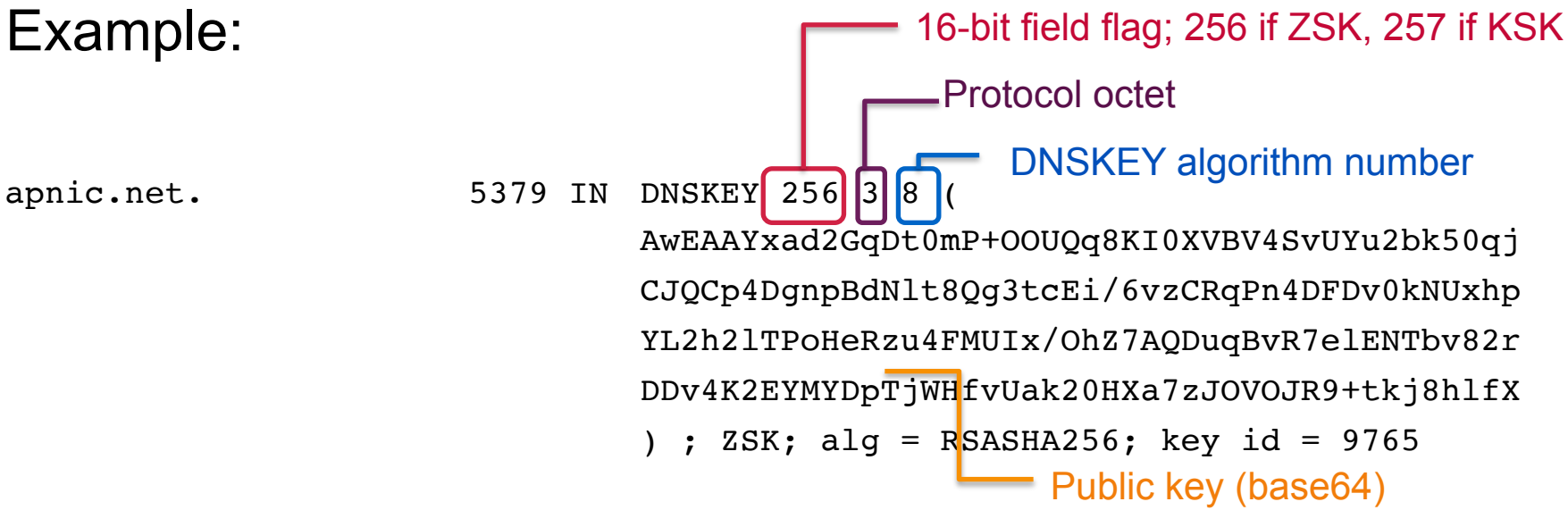
Resource Record		Function
RRSIG	Resource Record Signature	Signature over RRset made using private key
DNSKEY	DNS Key	Public key needed for verifying a RRSIG
DS	Delegation Signer	Pointer for building chains of authentication
NSEC / NSEC3	Next Secure	indicates which name is the next one in the zone and which type codes are available for the current name

New Resource Records

- **RRsets** are signed with private key to prove its authenticity and integrity
- The signatures are published in DNS as **RRSIG**
- Public **DNSKEY** is also published so RRSIG can be verified
- Child zones also sign their records with their private key
- Parent signs the child zone's **DS record** to prove authenticity

DNSKEY

- Contains the zone's public key
- Uses public key cryptography to sign and authenticate DNS resource record sets (RRsets).
- Example:



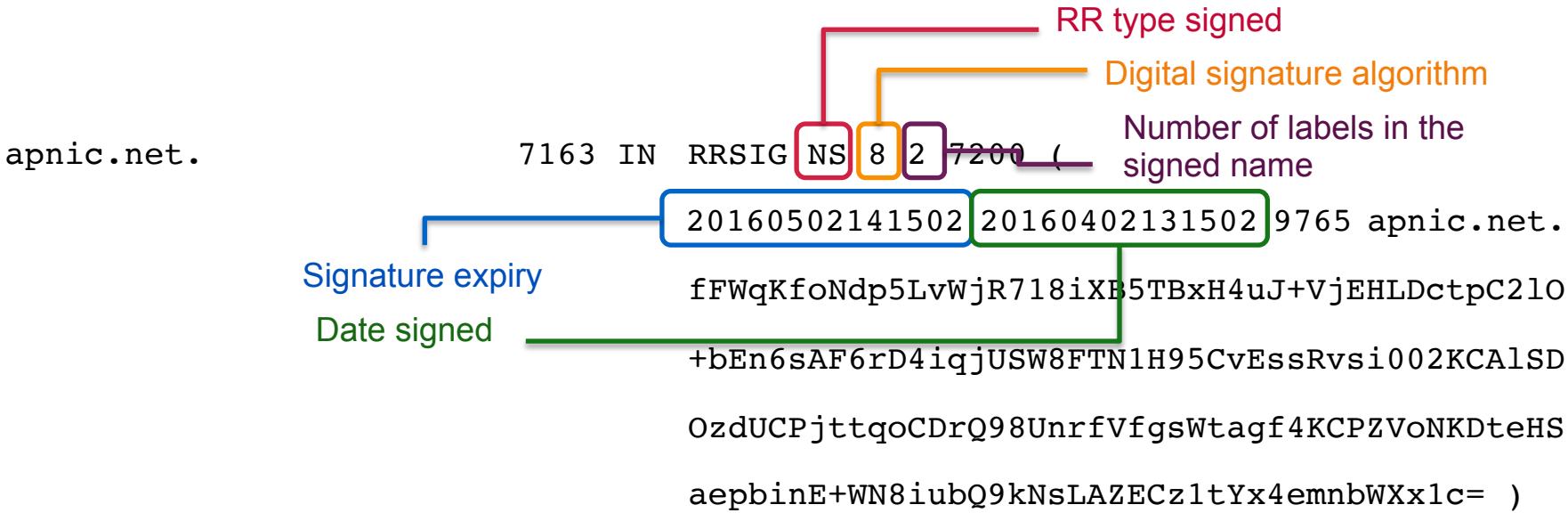
DNSKEY

- Also contains some timing metadata – as a comment in the key file

```
; This is a key-signing key, keyid 19996, for myzone.net.
; Created: 20121102020008 (Fri Nov  2 12:00:08 2012)
; Publish: 20121102020008 (Fri Nov  2 12:00:08 2012)
; Activate: 20121102020008 (Fri Nov  2 12:00:08 2012)
```

RRSIG

- The private part of the key-pair is used to sign the resource record set (Rrset)
- The digital signature per RRset is saved in an RRSIG record



NSEC Record

- **N**ext **Sec**ure
- Forms a chain of authoritative owner names in the zone
- Lists two separate things:
  - Next owner name (canonical ordering)
  - Set of RR types present at the NSEC RR's owner name
- Also proves the non-existence of a domain
- Each NSEC record also has a corresponding RRSIG

```
myzone.net.  NSEC blog.myzone.net. A NS SOA MX RRSIG NSEC DNSKEY
```

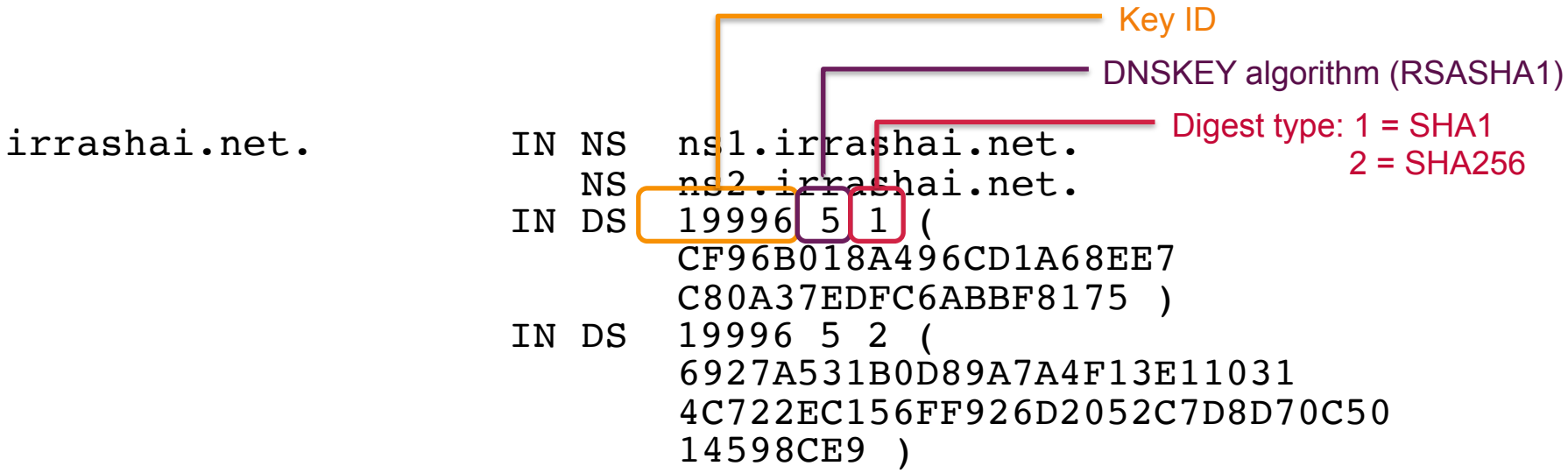
NSEC Record – Example

```
$ORIGIN example.net.
@  SOA        …
         NS       NS.example.net.
         DNSKEY   …
         NSEC     mailbox.example.net. SOA NS NSEC DNSKEY       RRSIG

mailbox  A        192.168.10.2
                         NSEC   www.example.net.   A NSEC RRSIG
 WWW             A        192.168.10.3
                         TXT      Public webserver
                         NSEC   example.net. A NSEC RRSIG TXT
```

DS Record

- **D**elegation **S**igner
- Establishes authentication chains between DNS zones
- Must be added in the parent's zonefile
- In this example, irrashai.net has been delegated from .net. This record is added in the.net zone file



Chain of Trust

- Establishes a chain of trust from parent to child zone
- How?
  - Parent does not sign child zone
  - Parent only signs a pointer to the child zone (key) – DS RECORD
- The root is on top of the chain

Creation of keys

- In practice, we use two keypairs
  - one to sign the zones, another to sign the other key
- Using a single key or both keys is an operational choice (RFC allows both methods)
- If using a single key-pair:
  - Zones are digitally signed using the private key
  - Public key is published using DNSKEY RR
  - When key is updated, DS record must again be sent to parent zone
- To address this administrative load, two keypairs will be used

Types of Keys

- Zone Signing Key (ZSK)
  - Signs the RRsets within the zone
  - Signed by the KSK
  - Uses flag 256

- Key Signing Key (KSK)
  - Signs the ZSK
  - Pointed to by the parent zone
  - Acts as the secure entry point to the

Signature Expiration

- Keys do not expire
  - Still a good practice to generate new ones regularly for added security
- Signatures have validity period
  - By default set to 30 days
  - This info is added in the key metadata
- Expired signatures will not validate
  - Must re-sign the zones

DNSSEC - Setting up a Secure Zone

- Enable DNSSEC in the configuration file (named.conf)
  - `dnssec-enable yes; dnssec-validation yes;`
- Create key pairs (KSK and ZSK)
  - `dnssec-keygen -a rsasha1 -b 1024 -n zone myzone.net`
- Publish your public key
- Signing the zone
- Update the config file
  - Modify the zone statement, replace with the signed zone file
- Test with dig

DNSSEC in the Resolver

- Recursive servers that are dnssec-enabled can validate signed zones
- Enable DNSSEC validation

```
dnssec-validation yes;
```

- The AD bit in the message flag shows if validated

DNSSEC Validation

- Other options if you don't have a validating resolver
  - validator add-on for your web browser
    - ex: https://www.dnssec-validator.cz/
  - Online web tools
    - http://dnsviz.net/
    - http://dnssec-debugger.verisignlabs.com/
- Use an open DNSSEC-validating resolver
  - DNS-OARC's ODVR (link)
    - 149.20.64.20 (BIND9), 149.20.64.21 (Unbound)
  - Google Public DNS
    - 8.8.8.8 or 8.8.4.4

DNSSEC - Setting up a Secure Zone

- Enable DNSSEC in the configuration file (named.conf)
  - `dnssec-enable yes; dnssec-validation yes;`
- Create key pairs (KSK and ZSK)

  `dnssec-keygen -a rsasha1 -b 1024 -n zone` **`myzone.net`**
- Publish your public key
- Signing the zone
- Update the config file
  - Modify the zone statement, replace with the signed zone file
- Test with dig

Generating Key Pairs

- To create ZSK

```
dnssec-keygen -a rsasha1 -b 1024 -n zone
myzone.net
```

- To create KSK

```
dnssec-keygen -a rsasha1 -b 2048 -f KSK -n
zone myzone.net
```

Publishing the Public Key

- Using `$INCLUDE` you can call the public key (DNSKEY RR) inside the zone file

```
$INCLUDE /path/Kmyzone.net.+005+33633.key ; ZSK
$INCLUDE /path/Kmyzone.net.+005+00478.key ; KSK
```

- You can also manually enter the DNSKEY RR in the zone file

Signing the Zone

- Sign the zone using the secret keys:

```
dnssec-signzone –o <zonename> -N INCREMENT -f
<output-file> -k <KSKfile> <zonefile> <ZSKfile>
```

```
dnssec-signzone –o myzone.net -k Kmyzone.net.
+005+11111 db.myzone.net Kmyzone.net.+005+33633
```

- Once you sign the zone a file with a .signed extension will be created
  - `db.myzone.net.signed`

Publishing the Zone

- Reconfigure to load the signed zone. Edit named.conf and point to the signed zone.

```
zone “<myzone>” {
     type master;
     # file “db.myzone.net”;
     file “db.myzone.net.signed”;
};
```

Testing the Server

- Ask a dnssec-enabled server and see whether the answer is signed

```
dig @localhost www.apnic.net +dnssec
+multiline
```

Testing with Dig

```
dig @localhost www.apnic.net +dnssec +multiline
```

```
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 48910
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 512
;; QUESTION SECTION:
;www.apnic.net.          IN A

;; ANSWER SECTION:
www.apnic.net.          4539 IN A 203.119.102.244
www.apnic.net.          4539 IN RRSIG A 8 3 7200 20160502141502 (
                                20160402131502 9765 apnic.net.
                                T7bXHN/CJvkXMPOeNPWK3qkvBvsIHl0+pBu9JgszoSE7
                                gWn9U0ufmB8iq7sIwIrKavYRAZNKFjARluwmWK+6L+7U
                                xHSqFI7baAcnB6nqfvip7Az+bPoeIoVdb30+1UimaOZ8
                                8W9w9Ouf59sncughSvniAezZfO7T5vdZ2nP50Gw= )

;; Query time: 168 msec
```

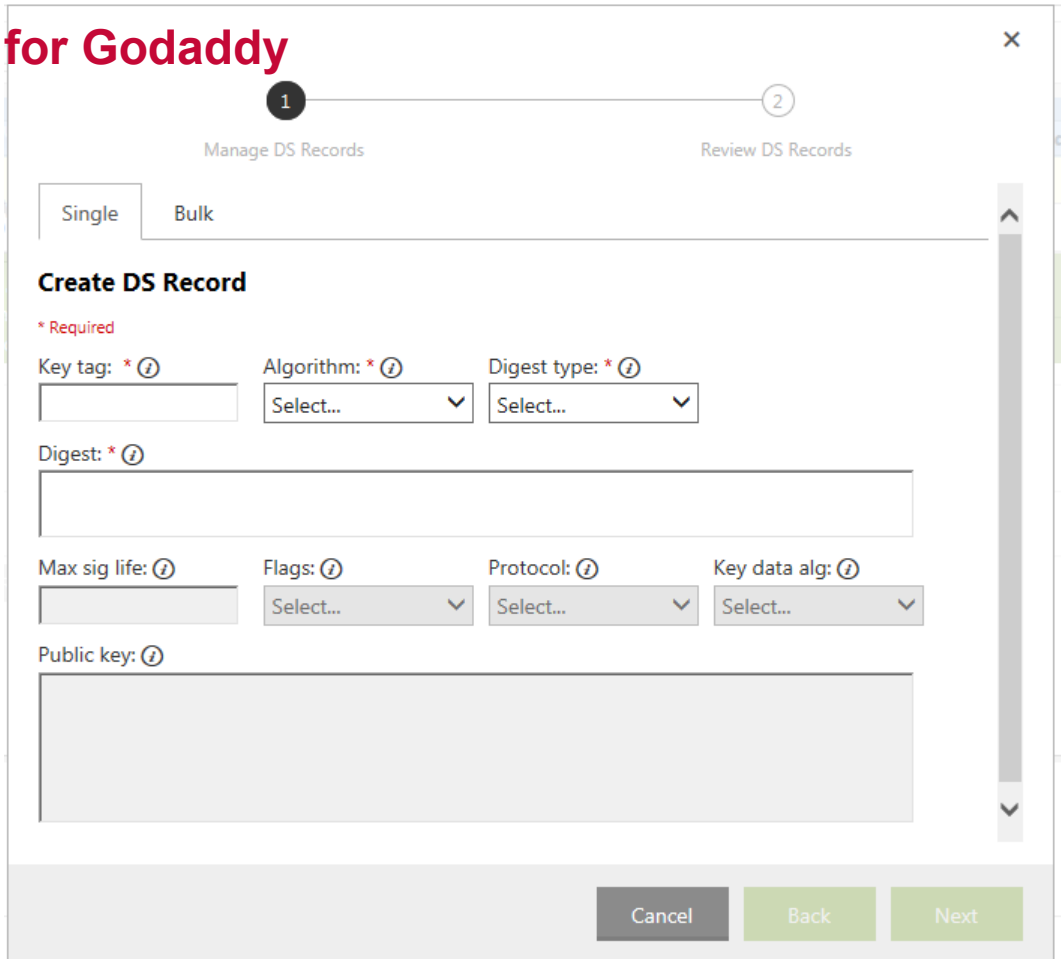
Pushing the DS record

- The DS record must be published by the parent zone.
- Contact the parent zone to communicate the KSK to them.

- There are proposals in the IETF DNSOP WG to address this:
  - Automating DNSSEC Delegation Trust Maintenance (link)
  - Child to Parent Synchronization in DNS (link)

Pushing DS Records for Forward Zone

Example form for Godaddy



Question