DNSSEC

Issue Date: 29 June 2016

Revision: 2.0

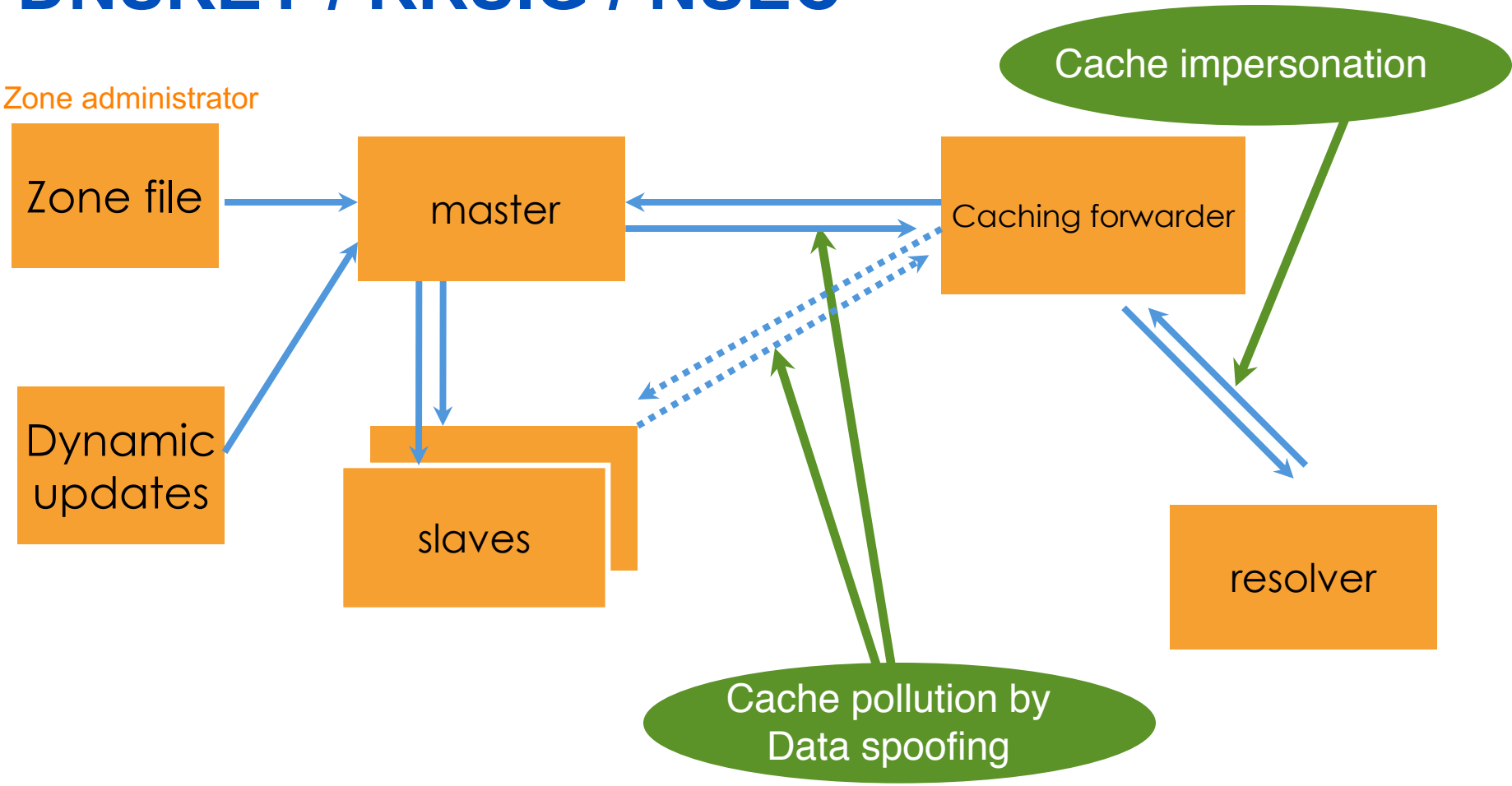
APNIC

Overview

- DNS Overview
- BIND DNS Configuration
- Recursive and Forward DNS
- Reverse DNS
- Troubleshooting
- DNS Security Overview
- DNS Transactions
- **DNS Security Extensions**
- DNSsec Key Management and Automation

APNIC

Vulnerabilities protected by DNSKEY / RRSIG / NSEC



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
- Provides a mechanism to:
  - establish authenticity and integrity of data
  - delegate trust to third parties or parent zones

DNSSEC History

- **1990**: Steven Bellovin discovers a major flaw in the DNS
- **1995**: Bellovin publishes his research; DNSSEC becomes a topic within IETF
- **1998**: Dan Kaminsky discovers some security flaw
- **1999**: RFC 2535, the DNSSEC protocol, is published
- **2005**: Three new RFCs published to update RFC2535
  - RFC 4033 (DNS Security Introduction and Requirements)
  - RFC 4034 (Resource Records for DNS Security Extensions)
  - RFC 4035 (Protocol Modifications)

https://wiki.tools.isoc.org/DNSSEC_History_Project

DNSSEC History

- **2005**: In October, Sweden (.SE) becomes the first ccTLD to deploy DNSSEC
- **2008**: new DNSSEC record created to address privacy concerns (RFC 5155)
- **2010**
  - In July 15, the root zone was signed
  - In July 29, .edu was signed
  - In December 9, .net was signed
- **2011**: In March 31, **`.com`** was signed

https://wiki.tools.isoc.org/DNSSEC_History_Project

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

How DNSSEC Works

Authoritative servers

Sign their zones

Answer queries with the record requested

Also send the digital signature corresponding to the record

Validating Resolvers

Authenticates the responses from the server

Data that is not validated results to a “`SERVFAIL`” error

New Concepts in DNSSEC

- New resource records
- Chain of trust
- Key generation and signing
- Validation

New Resource Records

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

RRs and RRsets

- Resource Record – each entry in the zonefile

```
www.example.net.  7200  IN  A 192.168.1.1
```

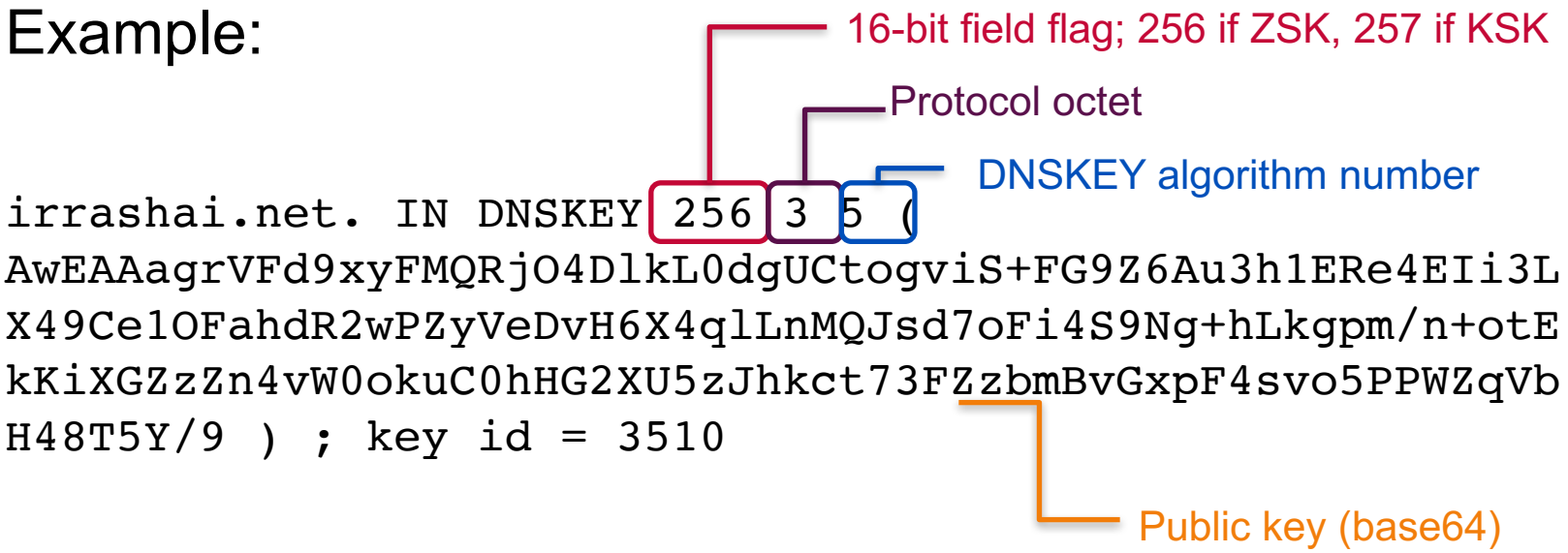
- RRset - RRs with same name, class and type

```
www.example.net. 7200 IN    A     192.168.1.1
web1.example.net. 7200 IN   A     10.0.0.1
web2.example.net. 7200 IN   A     172.16.0.20
```

In DNSSEC, RRsets are signed and not the individual RRs

DNSKEY

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



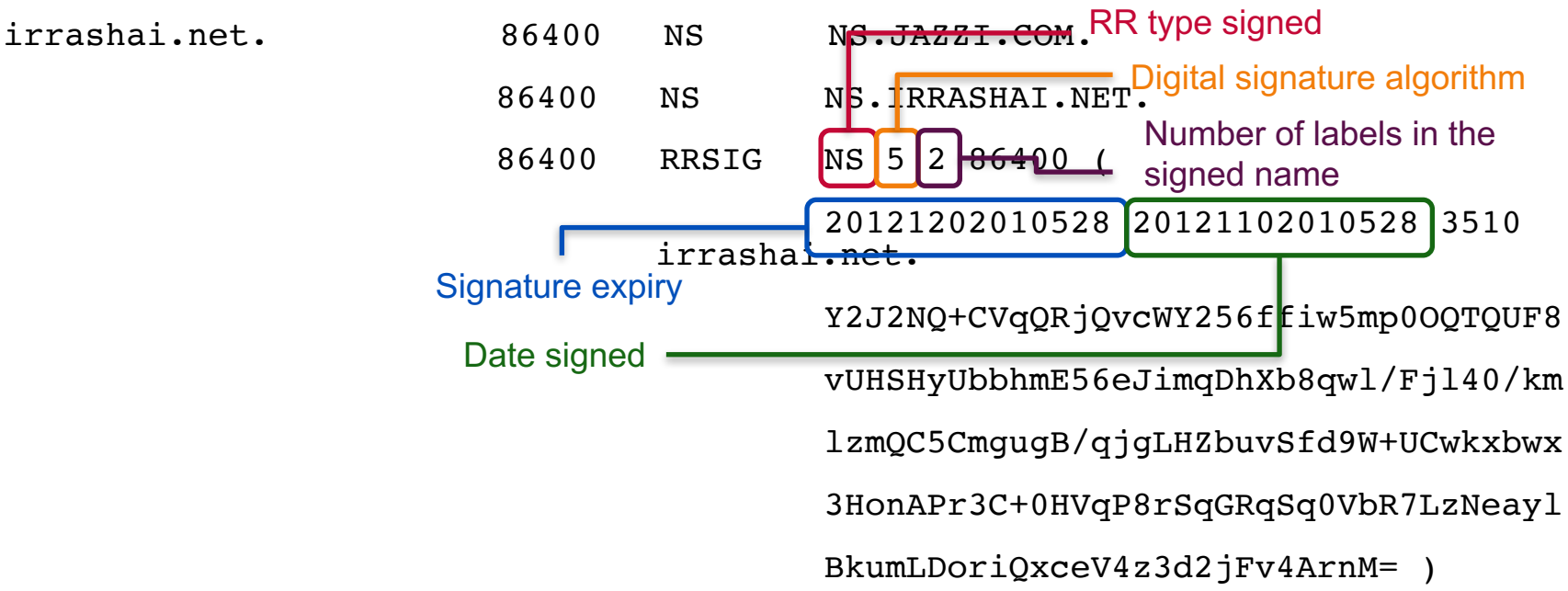
DNSKEY

- 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

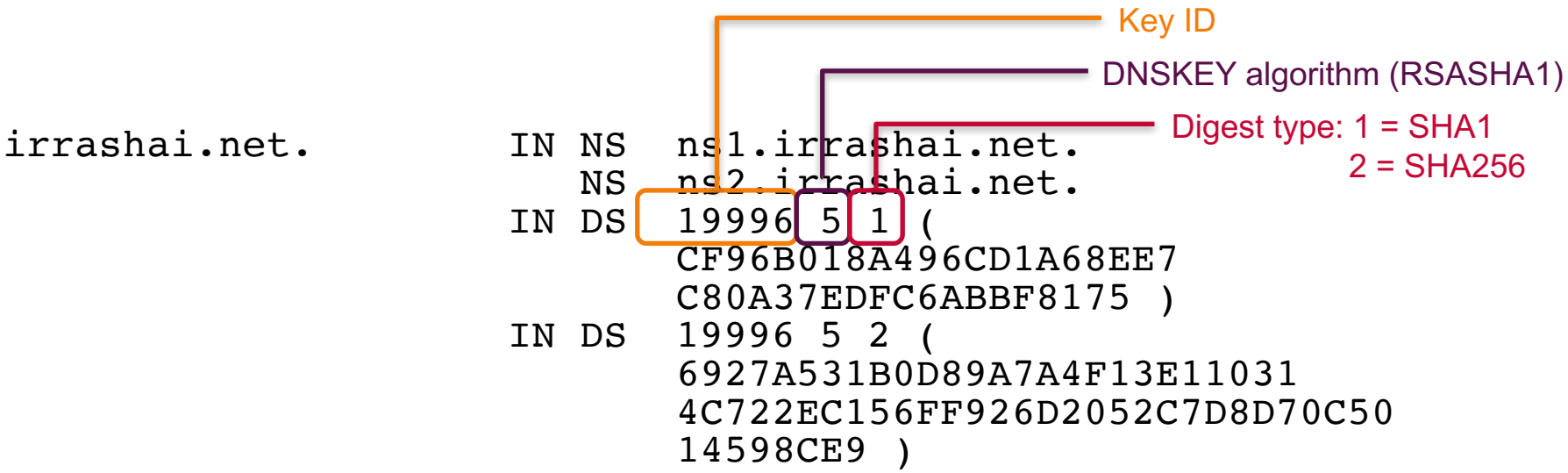
```
$ORIGIN example.net.
@ SOA        …
       NS     NS.example.net.
       DNSKEY …
       NSEC    mailbox.example.net. SOA NS NSEC DNSKEY        RRSIG

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
```

NSEC3

- NSEC allows an attacker to walk through the linked list to find all the records in the zone file. This is called zone walking.
- NSEC3 uses a hashing algorithm to list the next available domain in “hashed” format
- It is still possible for an attacker to do zone walking, although at a higher computation cost.

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



DS Record

- indicates that delegated zone is digitally signed
- Verifies that indicated key is used for the delegated zone
- Parent is authoritative for the DS of the child zone
  - Not for the NS record delegating the child zone
  - DS **should not** be added in the child zone

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)

- Sign 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 security entry point

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 for Network Service Providers

- Enable DNSSEC on your recursive servers and validate responses
  - Deploy DNSSEC-validating resolvers
- Before you fully implement:
  - Domains that can't be validated will be inaccessible
  - Be prepared to answer helpdesk queries related to this


Questions

Implementing DNSSEC

APNIC

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)
  - `option {`
    `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
- Sign the zone
- Update the config file
  - Modify the zone statement, replace with the signed zone file
- Test with dig

Updating the DNS Configuration

- Enable DNSSEC in the configuration file (named.conf)

```
options {
        directory "…."
        dnssec-enable yes; #respond to DNS requests from DNSSEC aware clients
        dnssec-validation yes; #validate responses up the chain
    };
```

- Other options that can be added too

```
        dnssec-lookaside auto ;  #ISC's DLV trusted key repo
```

Generating Key Pairs

- To create ZSK (default is rsasha1 and 1024)

```
dnssec-keygen -a <algo> -b <key-size> -n ZONE <zone>
```

- To create KSK (default is rsaha1 and 2048)

```
dnssec-keygen -a <algo> -b <key-size> -f KSK -n ZONE
<zone>
```

Generating Key Pairs - Reverse

- To create ZSK

```
dnssec-keygen -a rsasha1 -b 1024 -n zone
100.168.192.in-addr.arpa
```

- To create KSK

```
dnssec-keygen -a rsasha1 -b 2048 -f KSK -n
zone 100.168.192.in-addr.arpa
```

Publishing the Public Key

- Using `$INCLUDE` you can call the ***public*** key (DNSKEY RR) inside the zone file (eg: `db.myzone.net`)

```
$INCLUDE /path/Kmyzone.net.+005+<ZSK_id>.key ; ZSK
$INCLUDE /path/Kmyzone.net.+005+<KSK_id>.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> -f <output-file> -k
<KSKfile> <zonefile> <ZSKfile>
```

Ex:

```
dnssec-signzone –o myzone.net 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`

Signing the Zone

- Note that only authoritative records are signed
  - NS records for the zone itself are signed
  - NS records used for delegations are not signed
  - DS records are signed
  - Glue records are not signed
- Notice the difference in file size
  - db.myzone.net vs. 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”;
};
```

Publishing the Zone – Reverse

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

```
zone “<myzone>” {
     type master;
     # file “db.192.168.100”;
     file “db.192.168.100.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.irrashai.net +dnssec (+multiline)
```

```
; <<>> DiG 9.9.5-P1 <<>> @localhost www.irrashai.net +dnssec
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10871
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;www.irrashai.net.              IN      A

;; ANSWER SECTION:
www.irrashai.net.       864000  IN      A       192.168.100.100
www.irrashai.net.       864000  IN      RRSIG   A 5 3 864000 20150604031347 20150505031347 44727 irrashai.net. HBFfuooWXCIOyOuySO11/rSru5smi/E2mXAHR2tEP093IT8gMIPSQIL4 78XN3ecg3xQ1o
oeYTFjX6dgknE6Y4o179Ufba+zreHRP6sbBf852Btf4 wSExAZdOS9BmTEtDlhKXRDMnc0/9enqcfnku7IQqDYxudGBGfNmF5mnr gGY=

;; AUTHORITY SECTION:
irrashai.net.           864000  IN      NS      NS.IRRASHAI.NET.
irrashai.net.           864000  IN      RRSIG   NS 5 2 864000 20150604031347 20150505031347 44727 irrashai.net. 0BdYHJMLtvhhfbdwtcA4Z0Ja83L6iB51msJpurYzzffmiB5amq1V30YR vaFHqYM64Lmi
iXAePvq/mpdvutx6FiggNTyVbOHQ7+1ecHdNXO+AkGuF 2h4Go/rpjBpBN9a4Fexvuw7laO8CSykpfTNZ4hNaFagO/WmzbE9Pzm1K Vmg=

;; ADDITIONAL SECTION:
ns.irrashai.net.        864000  IN      A       192.168.100.8
ns.irrashai.net.        864000  IN      RRSIG   A 5 3 864000 20150604031347 20150505031347 44727 irrashai.net. MQQsnqWjMDJXI1VHNzXWywbRqDhYrEqxd3tMtx2Ua8ep+HYMfsJ/8/Im F9IfdPKm3TN+6
okecCionMixtzuvNLAs9FXY5q5V0lpSuC+oRe6Fulip i75uvARtYoLttB3zBHVzAIIlULzsDyrgagZZNrSS+EF12oeKNwOSYEir 64k=

;; Query time: 0 msec
;; SERVER: ::1#53(::1)
;; WHEN: Wed May 06 17:10:44 EST 2015
;; MSG SIZE  rcvd: 625
```

Testing with Dig – Reverse

```
dig @localhost -x 192.168.100.100 +dnssec
```

```
; <<>> DiG 9.9.5-P1 <<>> @localhost -x 192.168.100.100 +dnssec
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10393
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;100.100.168.192.in-addr.arpa.  IN      PTR

;; ANSWER SECTION:
100.100.168.192.in-addr.arpa. 864000 IN PTR     www.irrashai.net.
100.100.168.192.in-addr.arpa. 864000 IN RRSIG   PTR 5 6 864000 20150604031101 20150505031101 22107 100.168.192.in-addr.arpa. FyBAUv5Z8Z+8H8ZpbxZjAaFIpC9cJfzwY8Ojuol92wetwdzF0dyUV9v/
 XSwizzqGO9Pe3nchwRJNt7Of27xB52HgYOryyO/UudxF5TxN8Dp10rmj AbbR/9GrWIW9TOunBWfv17Pnxb1AMvTckncdogZeSghRV5QZ6rvmMtx2 yxk=

;; AUTHORITY SECTION:
100.168.192.in-addr.arpa. 864000 IN     NS      NS.IRRASHAI.NET.
100.168.192.in-addr.arpa. 864000 IN     RRSIG   NS 5 5 864000 20150604031101 20150505031101 22107 100.168.192.in-addr.arpa. mXv26lJVvtAZxM7Ni/DZwr7Vw/xZ5da8iflNRTm0zWe3huKiBkCoXnB0
TXmTNQKxfknfA1pLPrC4OZL4UyP00vAOwi5VYFZzwF/KA9xI9o8f59ng KbxWsbGtHLl3/e4Q8+lKSfVb4Al0cAF/m3yauQjYHGXzCHBO76w9nhk+ E7A=

;; ADDITIONAL SECTION:
ns.irrashai.net.        864000  IN      A       192.168.100.8
ns.irrashai.net.        864000  IN      RRSIG   A 5 3 864000 20150604031347 20150505031347 44727 irrashai.net. MQQsnqWjMDJXI1VHNzXWywbRqDhYrEqxd3tMtx2Ua8ep+HYMfsJ/8/Im F9IfdPKm3TN+6
okecCionMixtzuvNLAs9FXY5q5V0lpSuC+oRe6Fulip i75uvARtYoLttB3zBHVzAIIlULzsDyrgagZZNrSS+EF12oeKNwOSYEir 64k=

;; Query time: 0 msec
;; SERVER: ::1#53(::1)
;; WHEN: Wed May 06 17:08:58 EST 2015
;; MSG SIZE  rcvd: 675

[root@testserver master]# _
```

Pushing the DS record

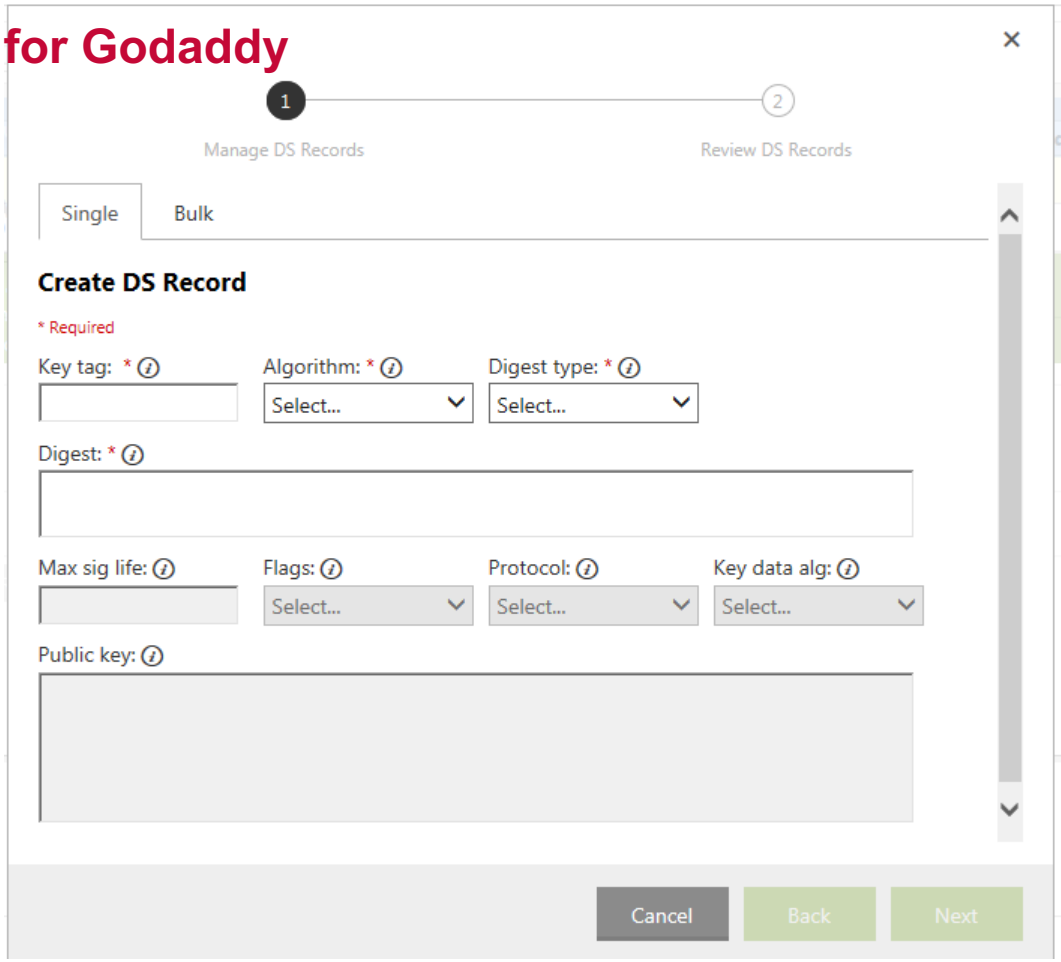
- The DS record must be published by the parent zone

- Contact the parent zone to communicate the KSK to them.

```
myzone.net. IN DS 4297 5 1 C5A8C518B2208463F87CB30E35F247DD7EACCDB1
```

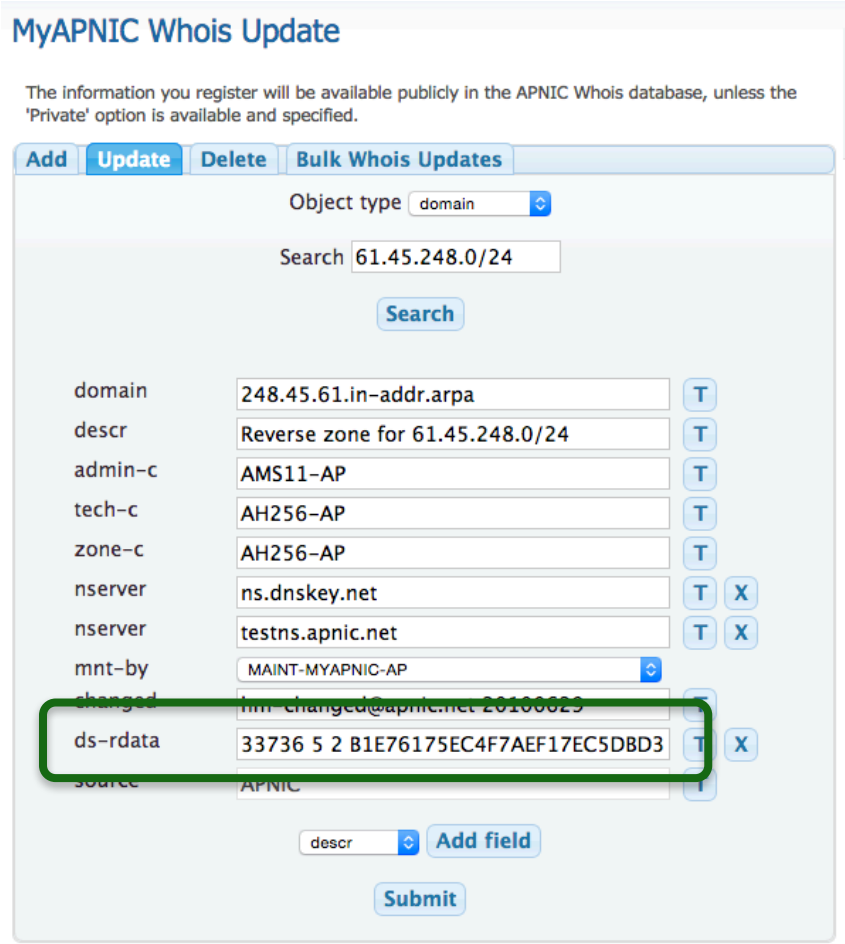
Pushing DS Records for Forward Zone

Example form for Godaddy



Pushing DS Record for Reverse Zone

Using MyAPNIC



DS record added in the domain object


Questions

Overview

- DNS Overview
- BIND DNS Configuration
- Recursive and Forward DNS
- Reverse DNS
- Troubleshooting
- DNS Security Overview
- DNS Transactions
- DNS Security Extensions (DNSSec)
- **DNSSEC Key Management and Automation**

APNIC

Key Rollover

- RFC 4641: DNSSEC Operational Practices
  - Covers general practices, procedures, recommendations
  - Update: http://tools.ietf.org/html/draft-ietf-dnsop-rfc4641bis-11
- Most commonly used:
  - KSK rollover: double signature policy
  - ZSK rollover: pre-publish policy

ZSK Key Rollover

Using Pre-publish

- Generate new ZSK, and publish the DNSKEY in the zone, but do not yet sign zone data with it
- Wait zone propagation time + TTL of the DNSKEY RRset
- Use new ZSK for signing zone records instead of old ZSK, but leave the old ZSK published in the zone
- Wait zone propagation time + largest TTL of all records in the zone
- Remove old key & re-sign DNSKEY RRset

KSK Key Rollover

Double signing

DNSSEC Operational Practices RFC 6781

- Lists down choices and decisions available when deploying DNSSEC
- Keep the chain of trust
  - Broken chains result in data being marked as Bogus
  - Shared responsibility by admins
- Key generation and storage
  - The motivations to differentiate KSK and ZSK are purely operational
  - Timing parameters
  - Key compromise and risk of cryptanalysis
  - Keys should be large enough to avoid all known crypto attacks during the effectivity period of the key
  - zone private keys and the zone file master copy to be signed be kept and used in off-line

https://tools.ietf.org/html/rfc6781

DNSSEC Operational Practices RFC 6781

- Signature generation, key rollover and policies
  - Data published in previous versions still live in caches
  - ZSK can be rolled without taking into account the DS record from parent
  - KSK rollover requires interaction with the parent
  - Emergency key rollover
- Motivation to deploy NSEC3 over NSEC
  - Prevention of zone enumeration

https://tools.ietf.org/html/rfc6781

DNSSEC Practice Statement – RFC 6841

- a means for stakeholders to evaluate the strength and security of the DNSSEC chain of trust
- DNSSEC Policies (DPs) – security requirements and standards to be implemented for a DNSSEC-signed zone
- DNSSEC Practice Statement (DPS) – practice disclosure document; states how the management of a given zone implements procedures and controls at a high level

https://tools.ietf.org/html/rfc6841

DNSSEC Practice Statement

- The DPS for Root Zone Signing Key (ZSK) is published
  - https://www.iana.org/dnssec/icann-dps.txt
- Published DPS of TLD operators

.SE's DNSSEC Practice Statement

www.iis.se/docs/se-dnssec-dps-eng.pdf

.CL's DNSSEC Practice Statement

http://www.nic.cl/dnssec/en/dps.html

.NET DNSSEC Practice Statement

http://www.verisigninc.com/assets/20100925-NET+DPS-FINAL.pdf

DNSSEC Guides

- Good Practice Guide for Deploying DNSSEC
  - ENISA
  - Published 2010
- Secure Domain Name System Deployment Guide
  - NIST
  - Published 2013


Questions

References

- **RFC 4033:** DNS Security Introduction and Requirements
- **RFC 4034:** Resource Records for the DNS Security Extensions
- **RFC 6841:** A Framework for DNSSEC Policies and DNSSEC Practice Statements
- **RFC 6781:** DNSSESC Operational Practices, Version 2
- **RFC 6605:** Elliptic Curve Digital Signature Algorithm (DSA) for DNSSEC