DNSSEC: Securing DNS

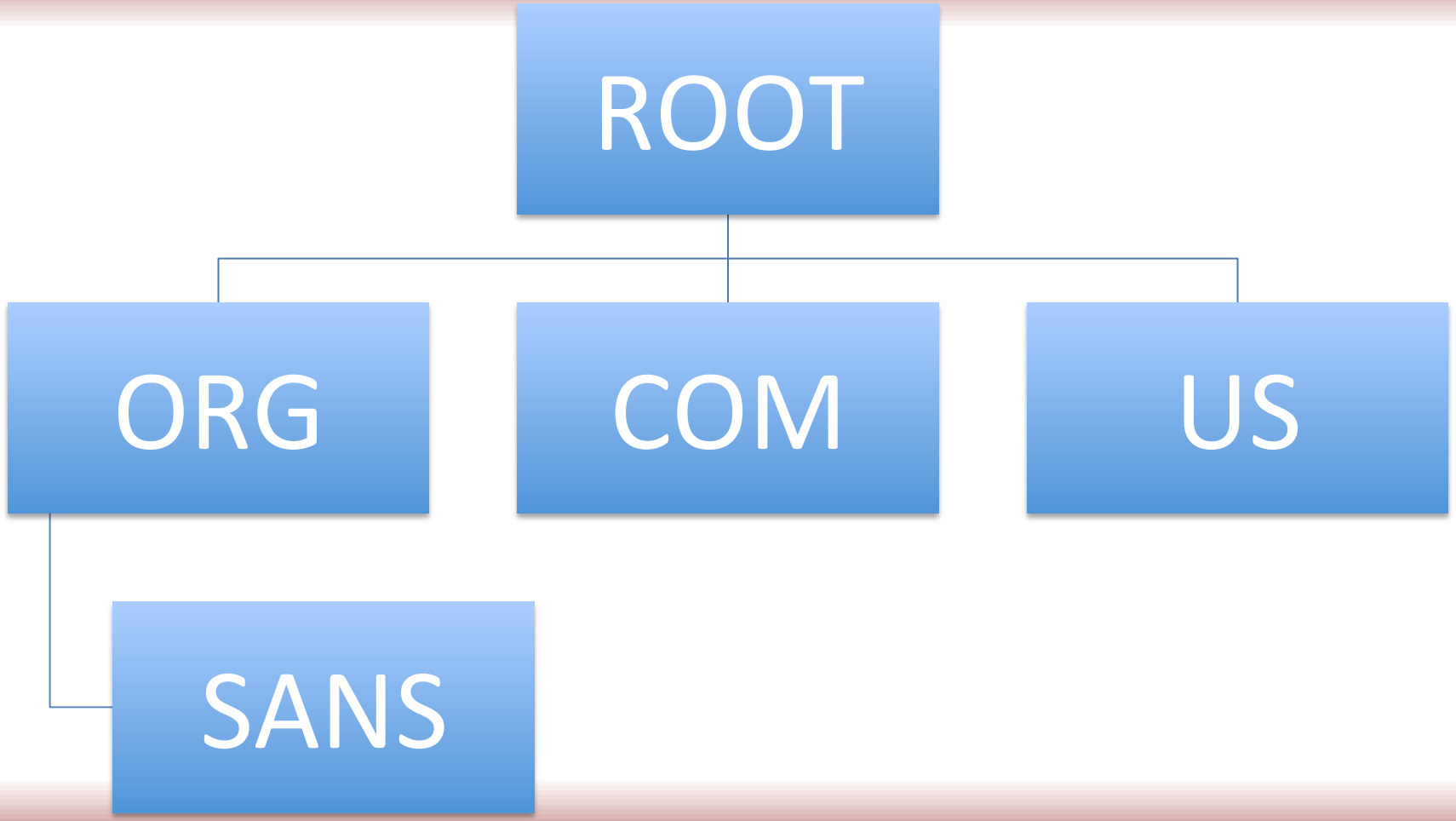
Johannes B. Ullrich, Ph.D.
jullrich@sans.edu

Slides will be at
http://isc.sans.edu/presentations

Outline

- A (very) quick DNS Primer
- Why we need DNSSEC
- Basic DNSSEC Concepts
- How to implement it in BIND
- Testing DNSSEC
- Conclusion

DNS Architecture



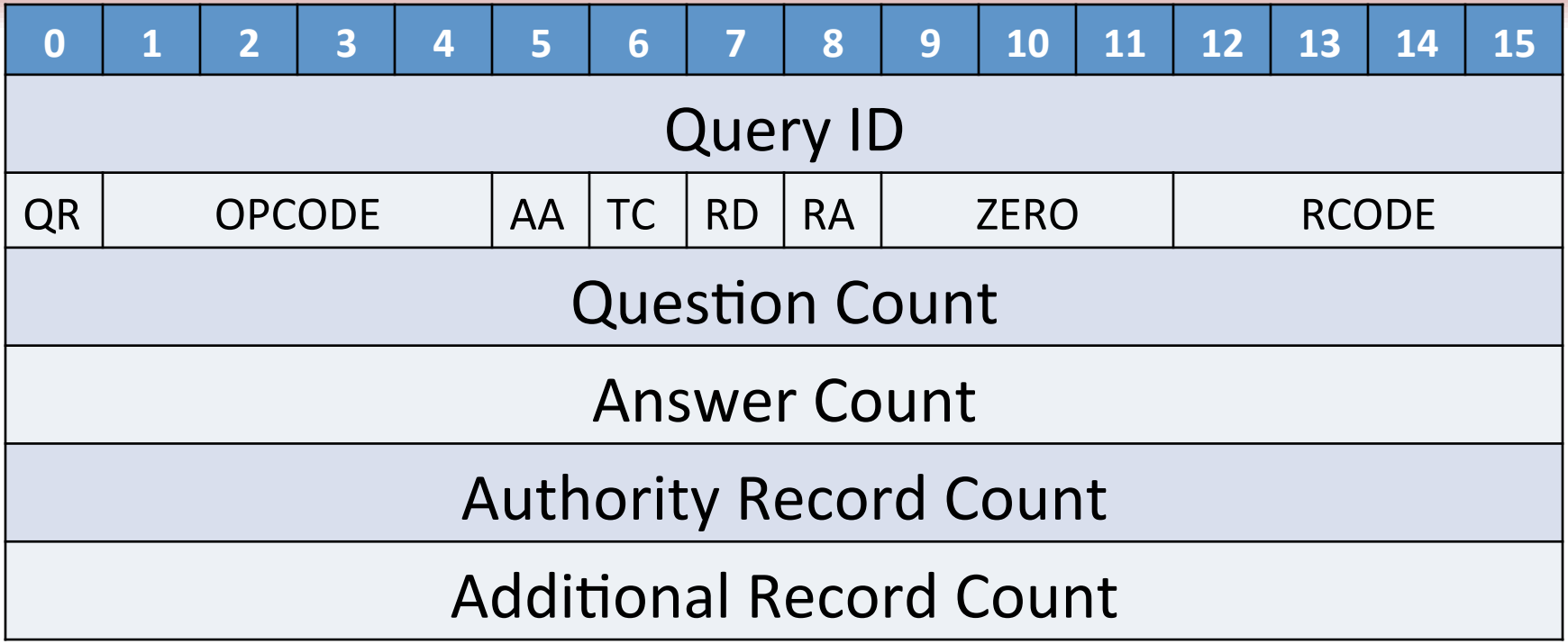
What is DNS?

"Occasionally it is assumed that the Domain Name System serves only the purpose of mapping Internet host names to data, and mapping Internet addresses to host names. This is not correct, the DNS is a general (if somewhat limited) hierarchical database, and can store almost any kind of data, for almost any purpose." (RFC 2181)

DNS Concepts

- Resolvers: Ask Questions (usually hosts)
- Recursive Name Servers: Find answers by asking other recursive name servers or authoritative name servers
- Authoritative Name Servers: Know answers (for specific zones)
- Forwarder: Forwards queries to a recursive server

DNS Header



RFC 1035

Query

- Query Name: host name
- Query Type: A, TXT, AAAA …
- Query Class: IN = Internet

Special case: Type and Class can be “*” for any record.

Resource Record (RR)

- Name
- Type
- Class
- TTL
- Data Length (in bytes)
- Data (depends on type and class)

Label

- 1-63 characters
- Multiple labels make up a domain name
- Maximum size of a domain name: 255 characters
- Any character is allowed for a label

Queries / Responses

- Same format is used for queries and responses
- A DNS server will return a response to each query, but not always an answer
- Response will include the query

Query ID

- 16 Bits (64k)
- Seen as main authentication mechanism in DNS
- Considered too weak (see Kaminski attack)
- According to RFC1035, a response may arrive from a different IP then the request was sent to… (fixed in RFC 2181)

How to Spoof DNS Responses?

1 - Trigger a DNS Request

send e-mail to the network. Anti spam tools will trigger DNS lookings

send an HTML e-mail to a user (embed images from “evil” domain as well as “victim” domain)

Spoof Responses

2 – Send spoofed responses

- query has to match (I triggered it, I know it)
- has to get IP addresses right

  small number of possible servers
- ports

  fixed (old version of server)
- query id

Spoofing Odds

- Server IP Addresses: 3
- Ports: 1
- Query ID: 65535 (64k)
- I can send 3-5 responses before real response arrives
- 1/64k

How Fast?

- Problem: once the real response arrives, it is cached
- I have to wait for the cache to expire before I try again (hours…)
- Fix: trigger a request for a host name that doesn't exist!
- Real server responds with NXDOMAIN. Not cached!

Result (1)

- I can trigger a request 10/sec.
- Need 30,000 tries
- 3,000 seconds < 1 hr

- Problem: I just managed to poison a useless hostname (e.g. “abcdef.google.com”)

Special Trick

- With each response, add additional information to the response
- If the response is valid, and the additional information is “in bailiwick”, then the additional informtion MAY be cached too

Additional information is the real “poison”

We Need Security!

- Randomize source port: Helps, but not a game changer
- TSIG: Symmetric encryption (for zone transfer)
- DNSSEC: Public Key Crypto
- Other…

TSIG

- Requires a shared secret
  - Dynamic DNS updates (DHCP Server -> DNS Server)
  - Zone Transfer
- Uses HMAC-MD5 hashes to authenticate requests and prevent tampering
- Doesn't scale

DNSSEC

- Public/Private Keys
- Mimics DNS hierarchy
- No “Certificate Authorities” in the SSL sense. Instead, root zone acts as CA
- Each zone responsible to sign delegated zones keys.

DNSSEC Hierarchy

- Creating a key for dshield.org
- Calculate hash for key ("DS Record")
- Deposit DS record with .org zone
- .org zone is validated using .org key
- .org key's DS record is deposited in the root zone

DNS header modifications

- Uses two of the three “zero” flags:
- AD Flag: set if server verified the data cryptographically
- CD Flag: checking disabled. Allows security aware server to send data before all signatures are received/verified.

DNSSEC Records

- DNSKEY: public key
- DS: hash of public key
- RRSIG: signature for specific RR
- NSEC: “next record”
- NSEC3: “hash of next record”
- NSEC3PARAM: “parameters used to calculate NSEC3 record”

Retrieving Dshield.org Key

dig +short DNSKEY dshield.org

256 3 7 AwEAAcABo/CSM0SdWUaG3r6tr…..

256 -> Flags (2 bytes)

3 -> Protocol (3 = DNSSEC)

7 -> Algorithm RSA/SHA1

DS Record for DShield

"Delegation Signer"

```
dig +short DS dshield.org
@d0.org.afilias-nst.org
37695 7 2 49C6…8F56B385
62013 7 1 812B654F…8ABAEE71
```

Algorithm: 7 – RSA/SHA1

Digest Type: 1 – SHA1, 2 – SHA256

How to add DS record?

- Calculate DS records for your Key Signing Keys only
- Add DS records to parent zone using your registrars mechanism (e.g. web interface)
- Wait for DS record to appear in parent zone before using it

RRSIG

```
www.dshield.org. 60  IN  RRSIG A 7 3
60 20110929125604 20110830125604
32252 dshield.org. BnqLVjp/+…Ma8=
```

TTL: 60

Algorithm: 7 (RSA/SHA-1 NSEC3)

Protocol: 3 (DNSSEC)

Valid: August 30th-Sept. 29th 2011

NSEC

- Problem: Proof of non existence
- For everything that exists, we do have a signature. But what about records that don't exist?
- NSEC points to the "Next Secure Record"
- List of all NSEC records provides directory of existing records

NSEC Zone Traversal

- NSEC records allow us to do a “zone transfer”
- Just follow chain of NSEC records
- One of the inhibitors to implementing DNSSEC (“makes us less secure”)
- Don’t put information in a public zone that is confidential

NSEC3

- To solve the information disclosure problem: Lets add hashes, not the actual name
- Only supported recenty
- Various hashing algorithms are possible
- NSEC3PARAM record defines details

Example

```
dig +dnssec www.dshield.org NSEC
QUERY: 1, ANS: 0, AUTH: 4, ADD: 1

SU3…9FSDF.dshield.org. 86400 IN
RRSIG NSEC3 7 3 86400
20110929125604 20110830125604
32252 dshield.org…q/btyNIrQo=
```

DNSSEC Algorithms

Number	Name	RFC
3	DSA/SHA1	RFC3755
5	RSA/SHA-1	RFC3755
7	RSA/SHA-1 (NSEC3)	RFC5155
8	RSA/SHA-256	RFC5702

DNSSEC Issues

- Performance: DNS is a high performance protocol.
- Larger responses may not fit into 512 bytes
- DNSSEC requires resolvers to verify crypto signatures on each RR they receive
- Tradeoff: Strong Keys vs. Latency

EDNS0

- Extension to DNS
- Defines an “OPT” section in addtion to queries, answers, authority and additional records
- Used to indicate maximum acceptable UDP packet size
- Typically set to 4096 Bytes

DNSSEC Solution to Key Strength

- Short keys require less CPU power to validate, but can also be broken easier
- If a key is short, its lifetime should be limited
- However, we want to limit “churn” on the TLD zones. The DS records shouldn’t change all the time

Zone Signing vs. Key Signing Keys

- We create two keys:
  - key signing key (KSK)
    - Strong Key
    - Hash deposited with parent zone
    - Only used to validate other keys
  - Zone signing key (ZSK)
    - Weak key
    - Rotated frequently
    - Used to sign RRs
    - Signed by KSK

Key Signing Key

- Replace it once a year (make it valid for 13 months)
- Always have two valid keys in your zone, but only use one (for smooth roll over)
- 1024 bits for low value domains
- 2048 bits for high value domains
- Recommended algorithm: RSA/SHA-1
- SEP Flag set

Zone Signing Key

- Can be smaller then KSK
- Validity could be minutes (not very practical)
- 1 month works fine
- Have two valid keys and use one
- 1024 bits RSA/SHA-1
- SEP Flag cleared

Operational Issues

- Keys need to be regenerated regulary
- Need scripts/procedures to create keys
- Forgetting to re-sign zone: Zone is now invalid
- Private key can be kept offline on a signing system
- Use a “signing system” to sign and then copy new zone to DNS server

DNSSEC and Name Servers

- Not all name servers support all algorithms
- Make sure you support at least NSEC3
- BIND 9.6 adds full NSEC3 support
- Win 2008 R2 default support for DNSSEC
- Djbdns does not support DNSSEC
- PowerDNS Version 3 and above

DNSSEC And Resolvers

- Need to add trusted key for at least root zone
- Keep root key current
- As of BIND 9.6.2 “manage keys” (RFC 5011)
- Allows to have key updated automatically

Look Aside Validation

- Using a trust anchor other then root zone
- Used to be more important before root zone was signed
- No less used, but maybe usefull in some cases for “trust islands”
- Supported by BIND

Debugging DNSSEC

- dig
- Drill
- DNSSEC Validator Firefox Plugin
- Dnsviz
- DNSSEC Debugger

dig

- +dnssec – turns on DNSSEC records
- +cdflag – validates DNSSEC records
- +sigchase – will try to validate keys in parent zone (not implemented in all versions of dig)

drill

- Similar to dig, but with better DNSSEC support by default
- drill -D –T www.dshield.org
- Trusted keys can be provided

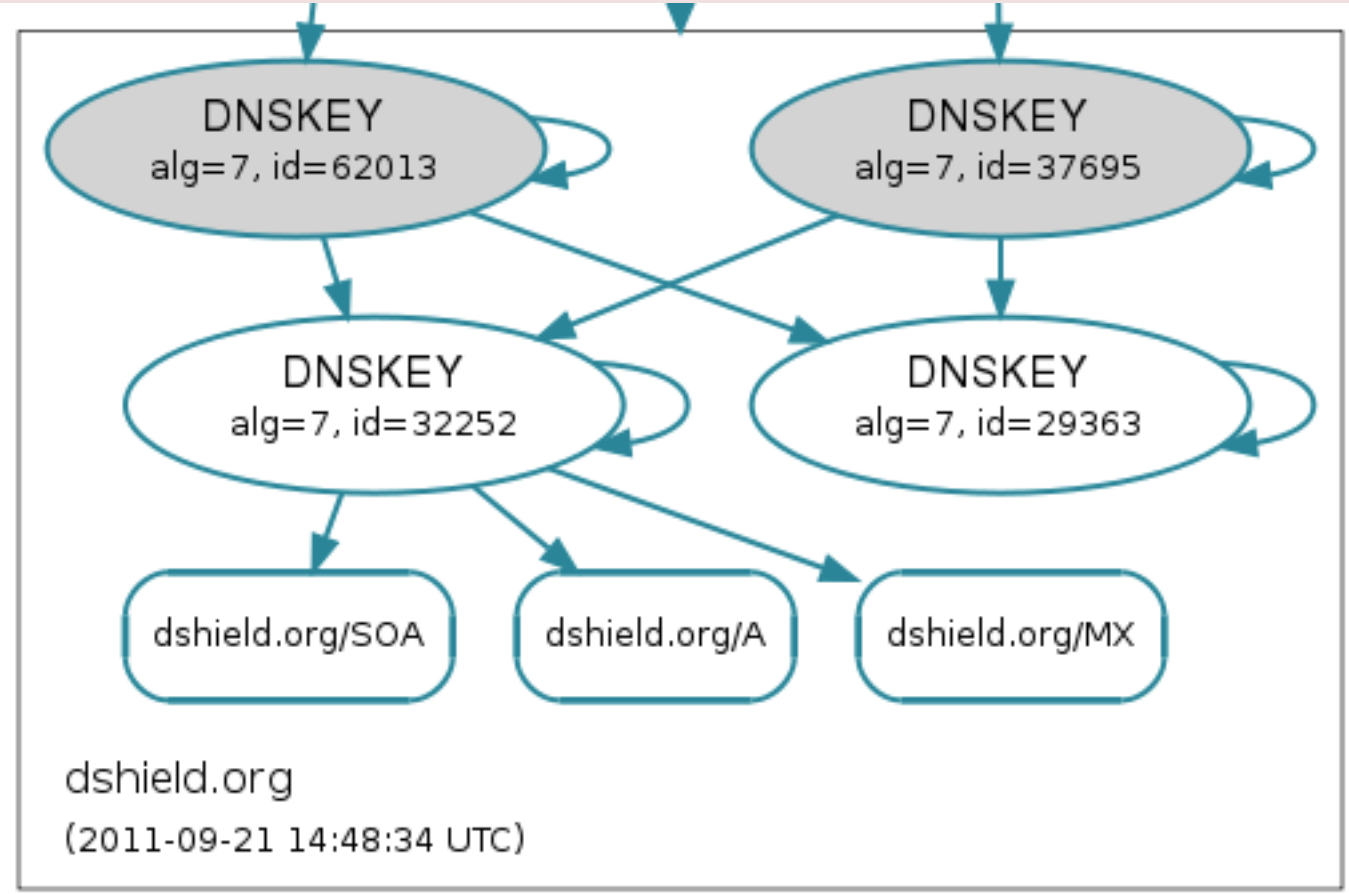
DNSSEC Validator Plugin (FF)

DNSVIZ

- Nice visualization of DNSSEC relationships (and errors)
- Created by Sandia Labs
- FREE!

- http://dnsviz.net

DNSVIZ: DShield.org



DNSSEC Debugger

- Written by Verisign
- Tabular output, less graphical then DNSVIZ
- Easier to spot nature of errors
- FREE!
- http://dnssec-debugger.verisignlabs.com

DNS Debugger Output

.	Found 3 DNSKEY records for . DS=19036/SHA1 verifies DNSKEY=19036/SEP Found 1 RRSIGs over DNSKEY RRset RRSIG=19036 and DNSKEY=19036/SEP verifies the DNSKEY RRset
org	Found 2 DS records for org in the . zone Found 1 RRSIGs over DS RRset RRSIG=39283 and DNSKEY=39283 verifies the DS RRset Found 4 DNSKEY records for org DS=21366/SHA1 verifies DNSKEY=21366/SEP Found 2 RRSIGs over DNSKEY RRset RRSIG=21366 and DNSKEY=21366/SEP verifies the DNSKEY RRset
	Found 4 DS records for dshield.org in the org zone Found 1 RRSIGs over DS RRset RRSIG=56472 and DNSKEY=56472 verifies the DS RRset

Other Methods

- Google proposed to use capitalization to add more entropy to the protocol
- Queries are not case sensitive, but the response will include an exact copy of the query
- Instead of requesting the A record for google.com, we could ask for GoOgLe.com

Does it “work”?

- In my testing: yes. DNS servers will maintain the capitalization of the query
- However: They don’t check it and there isn’t a facility to create random capitalized queries
- Needs patching of existing servers
- It is not a “standard”, it is a “hack”

Will long names be better?

- Sure, we got more “bits” to capitalize or not
- We could also ask more then one question
- Not really strong, not a game changer, just a delay (like switching ports around)

Should I implement DNSSEC

- Use a test zone first
- Script re-signing
- Test processes at least over 3-4 resigning cycles
- Bad configuration = DoS !
- Does your firewall allow > 512 DNS responses?

Reference

- http://www.dnssec.net
- http://www.dnsviz.net
- RFC 4641: Best Practices for DNSSEC