DNS (Domain Name System) Tutorial @ IETF-70

(DNS for protocol designers)

Ólafur Guðmundsson

OGUD consulting

Peter Koch

DENIC eG

Tutorial Overview

- Goal:
  - Give the audience basic understanding of DNS to be able to facilitate new uses of DNS
- Tutorial Focus: Big picture
  - Not software help
    - DNS != BIND
  - No gory protocol details
  - Location of slides: http://tinyurl.com/25nf9v

ogud@ogud.com & pk@denic.de

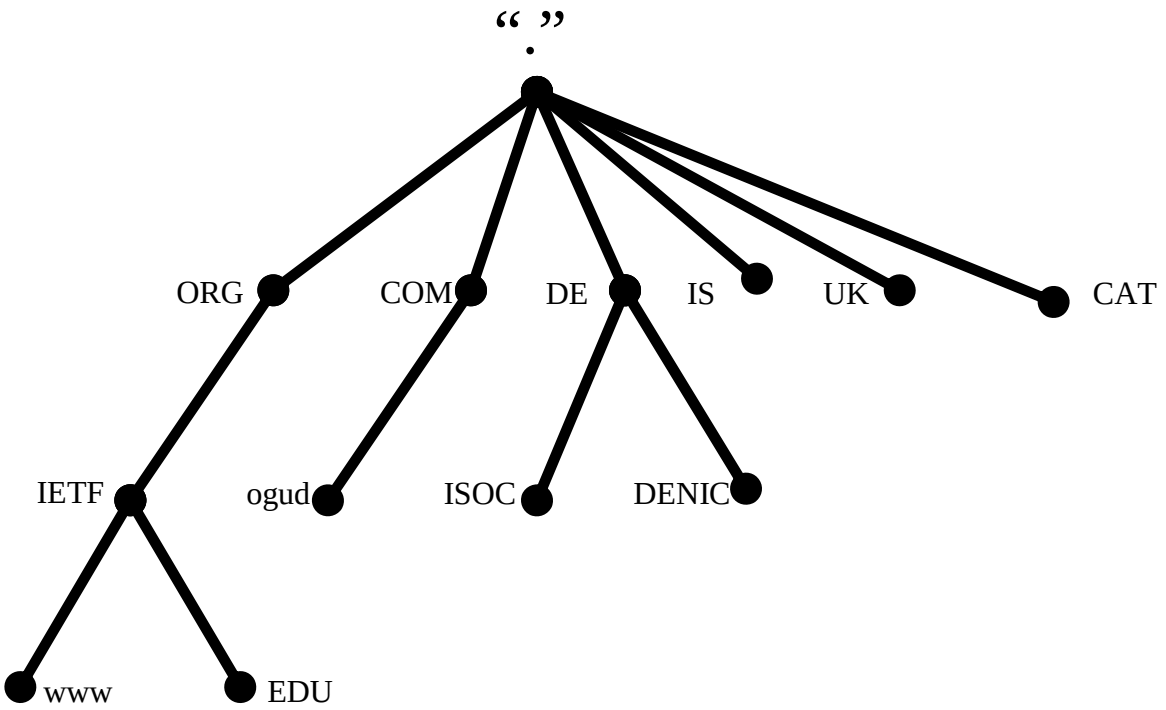
DNS Data Model

DNS is global "loosely consistent" delegated database

- delegated -> contents are under local control
- loosely consistent -> shared information (within constraints)
  - does not need to match or be up-to date.
  - operation is global with owners of "names" responsible for serving up their own data.
- Data on wire is binary
- Domain names are case insensitive for [A-Z][a-z],
  - case sensitive for others (`exämple.com` != `exÄmple.com`)
- Hostname [A..Z0..9-] RFC952
  - Restricts names that can be used
  - IDN provides standard encoding for names in non-US_ASCII

ogud@ogud.com & pk@denic.de

DNS tree



DNS Terms

- Domain name: any name represented in the DNS format
  - `foo.bar.example.`
  - `\0231br.example`.
- DNS label:
  - each string between two "." (unless the dot is prefixed by “\”)
  - i.e. `foo.bar` is 2 labels `foo\.bar` is 1 label
- DNS zone:
  - a set of names that are under the same authority
  - `example.com` and `ftp.example.com`, `www.example.com`
  - Zone can be deeper than one label, example .us, ENUM
- Delegation:
  - Transfer of authority for/to a sub-domain
    - `example.org` is a delegation from `org`
    - the terms parent and child will be used.

More DNS terms

- RR: a single Resource Record
- RRSet: all RRs of same type at a name
  - Minimum transmission unit
- Example:

```
- <name>    <TTL>  <Class> <RRtype>   <data>
- ogud.com.  13630   IN      MX      10 mail.ogud.com.
- ogud.com.  13630   IN      MX      90 smtp.elistx.com.
```

- TTL (Time To Live):
  - The maximum time an RRSet can be cached/reused by a non- authoritative server

DNS Elements

- Resolver
  - stub: simple, only asks questions
  - recursive: takes simple query and makes all necessary steps to assemble the full answer,
- Server
  - authoritative: the servers that contain the zone file for a zone, one Primary, one or more Secondaries,
  - caching: A recursive resolver that stores prior results and reuses them
    - Some perform both roles at the same time.

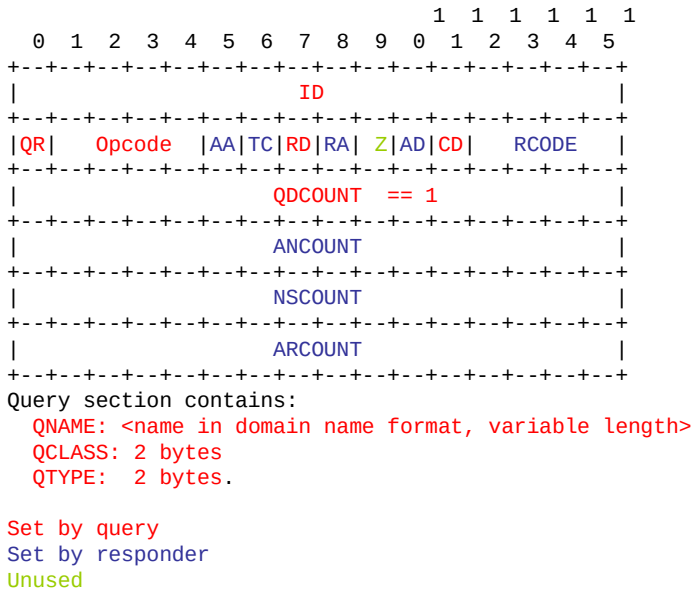
DNS retrieval mode

- DNS is a "lookup service"
  - Simple queries --> simple answers
  - No search
  - no 'best fit' answers
  - Limited data expansion capability

- DNS reasons for success
  - Simple
    - "holy" Q-trinity: QNAME, QCLASS, QTYPE
  - Clean
    - Explicit transfer of authority
      - Parent is authoritative for existence of delegation,
      - Child is authoritative for contents.

DNS Protocol on the wire

- Transport:
  - UDP 512 bytes Payload, with TCP fallback
    - RFC3226 increases to 1220 bytes
  - EDNS0 (OPT RR) (RFC2671) expands UDP payload size by mutual agreement.
  - TSIG (RFC2845) hop by hop authentication and integrity
- Retransmission: built in
  - Resends timed-out-query
    - Possibly to a different server.



ogud@ogud.com & pk@denic.de

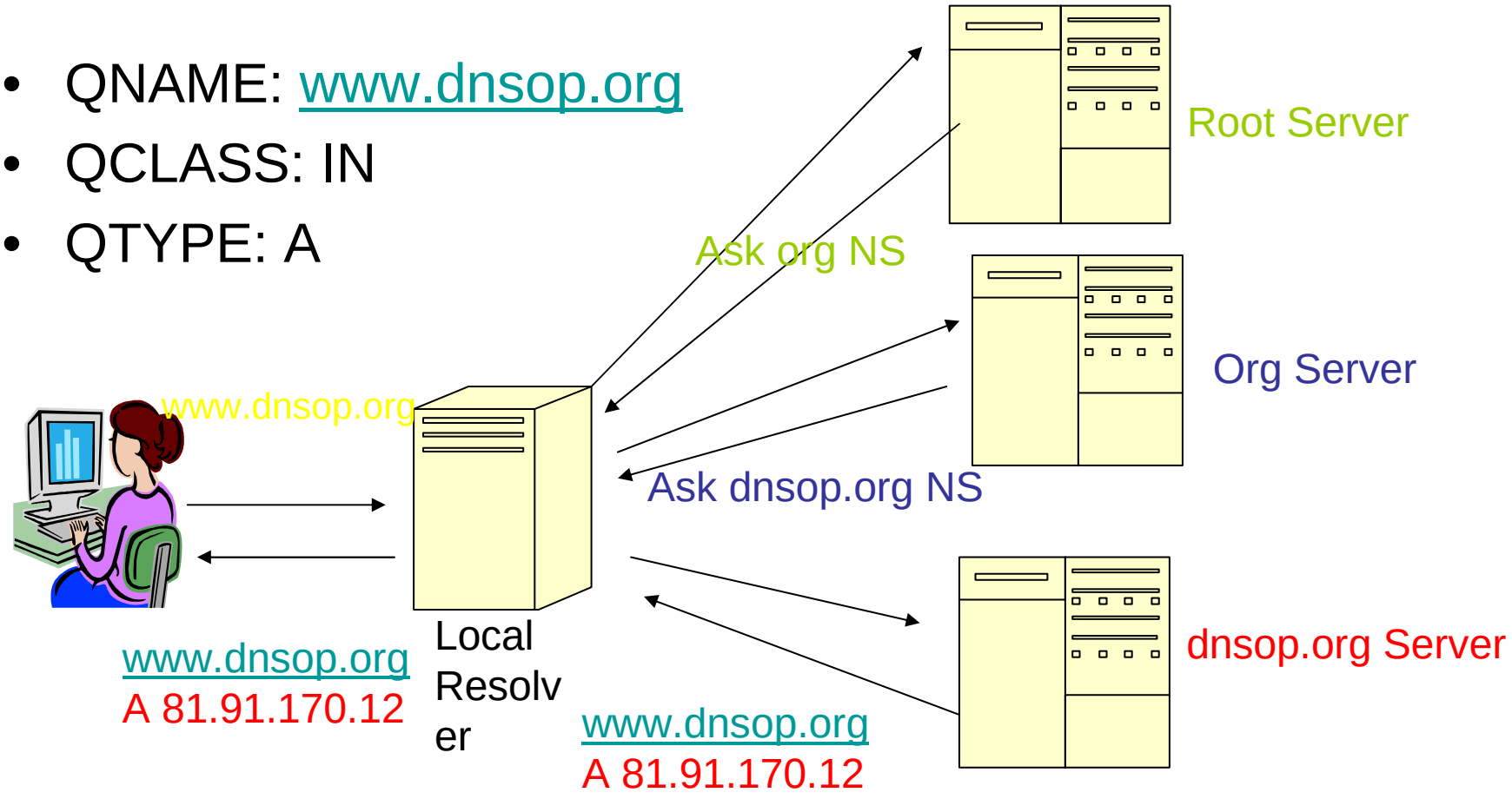
DNS RR wire format

```
+-------------------+-----+------+--------+----+-----------+
+ Domain name       |type | class| TTL    | RL |  RDATA    |
+-------------------+-----+------+--------+----+-----------+
    <variable>       2       2      4       2   <variable>
```

- Owner name (domain name)
  - Encoded as sequence of labels
    - Each label contains
      - Length (1 byte)
      - Name (n bytes [1..63])
      - example.com ➔ 07example03com00
- Type : MX, A, AAAA, NS …
- CLASS: IN (other classes exist, but none global)
- TTL: Time To Live in a cache
- RL: RD LENGTH: size of RDATA
- RDATA: The contents of the RR
  - Binary blob, no TLV (XXX Type Length Value).

DNS query

- QNAME: www.dnsop.org
- QCLASS: IN
- QTYPE: A



DNS data operation

- DNS zone is loaded on authoritative servers,
  - servers keep in sync using information in SOA RR via AXFR, IXFR or other means.
- DNS caches only store data for a “short” time
  - defined by TTL on RRSet.
- DNS Resolvers start at longest match on query name they have in cache when looking for data, and follow delegations until an answer or negative answer is received.
  - Longest match := if resolver has some of the right hand side delegations it will use them rather than start all queries at the root servers.
  - DNS transactions are fast if servers are reachable.

DNS Data/API issues

- Whole or none of RRSet will arrive,
  - in non determined order.
- DNS Resolver API should
  - Return known weighed DNS RRSet in weighed order
  - other RRSets in in random order.
- DNS data should reside in one place and one place only
  - at name, or at <prefix>.name
  - zone wide defaults do not exist
    - the "zone" is an artificial boundary for management purpose

ogud@ogud.com & pk@denic.de

DNS Wildcards:
The area of most confusion:
FACTS

- Is not a default but a provisioning aid
- match **ONLY** non existing names
- expansion is terminated by existing names

  →do not expand past zone boundaries

ogud@ogud.com & pk@denic.de

DNS wildcards: The area of most confusion: MYTHS

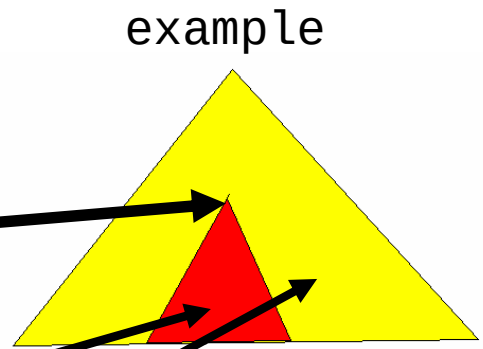
- Record:
  `*.example MX 10 mail.example`
  - matches any name, below the name `example` !!
  - supplies RR type to names present, missing MX RRs.
    - Is added to MX RRSet at a name
  - expands only one level
- `www.*.example` will expand

Wildcard Match

- Contents of a zone:

```
*.example. TXT "this is a wildcard"
www.example. A 127.0.0.1
jon.doe.example. A 127.0.0.2
```

- Name “doe.example” exists w/o any RRtypes ➔empty non-terminal
- Name “tina.doe.example.” will not be expanded from wildcard
- Name: “tina.eod.example.” Matched.



DNS rough corners

- Packet size:
  - 512 for standard DNS, 4K+ for EDNS0
  - Keeping RRSets small is good practice.
- Lame delegations:
  - Parent and children must stay in sync about name servers.
    - When a zone changes name servers it should send new set to parents.
  - Secondary servers must keep up-to date with Primary.
    - problems areas: permissions, transfer protocol not getting through, clock synchronization, old/renumbered primary/secondary, serial numbers not updated.
- Data integrity: Cache Poisoning
  - DNS answer can be forged, in particular if query stream is visible
  - use protected channel to recursive resolvers.
- Broken/old DNS Software:
  - Small percentage, but slowly decreasing base
- DNS name tricks
  - Not at DNS protocol issue but user interface or spoofing

Newsflash: DNS data can change ☺

- DNS Update (RFC2136):
  - adds the ability to change DNS contents of the fly ➔ used a lot.
    - SHOULD only be used for "leaf" data
- Difficult to add/modify data due to operator
  - DNS Secure Update (RFC3007) specifies how to securely delegate capability to update DNS names or name/type(s)
- One RR changes whole zone is sent to secondaries
  - Incremental Zone transfer (IXFR) (RFC1995) enables transfers of only changed data
    - DNS anycast clouds with over 100 servers use this to maintain large zones that are updated frequently
      - think seconds between updates
  - Notify (RFC1996) informs secondaries that update is available.

ogud@ogud.com & pk@denic.de

DNS: Unknown RR types

- Some early DNS implementation **hard** coded RR types.
  - Unknown RR were/are dropped by some resolvers
  - Unknown RR were not served by authoritative servers
    - Implication: introduction of **new** RR types took long time
- Solution:
  - RFC3597 defines that all DNS servers and resolvers MUST
    - support unknown RR types and rules for defining them.
    - suggests a common encoding in presentation format for them.
  - Deployment: (partial list)
    - BIND-9, BIND-8.2.2, ANS, CNS, MS DNS-2003, DNSCache, NSD, PowerDNS, Net::DNS, DNSJava, DNSpython, etc.

DNS and security

- Common scheme in *Security Considerations:* … for DNS the considerations in RFC 3833 apply …
- *DNSSEC* is the solution in protocol space
- Fact is, DNSSEC not yet deployed in *forward space,* slightly better in reverse or dedicated name spaces (e.g., ENUM)

DNSSEC: Data integrity and authentication for DNS

- Role: Protect DNS
  - How done: view from 10 km.
    - DNS RRSet is signed by the zone it belongs to.
    - zone DS RRSet is vouched for by parent zone.
- What DNSSEC does not do:
  - Make data in DNS any more correct
  - Particularly important for storing keying material in the DNS

DNSSEC: More details

- Data integrity protection
  - Each DNS RRSet has a special RRSIG containing a signature by the zone private key, for a certain time period
- Existence proof:
  - Chain of NSEC or NSEC3 records lists all names in a zone and their RR types. (authentic proof/denial of existence)
- Parent signs a fingerprint of child's Key Signing DNSKEY (DS RR)
  - allows transition from a secure parent zone to a secure child zone.

What does DNSSEC provide to applications?

1. DNS answer with verifiably signed RR set(s) is known to be identical to what zone maintainer initially entered
2. Widely deployed DNSSEC allows application to place more important data in DNS
   - unsigned keying info
     - IPSECKEY, SSHFP
   - spoof proof service location
   - No need for protocol specific keying infrastructure
   - other...

To do DNS or not to do DNS

- If your data is small (<2K)
- If the naming of the application objects map into DNS names easily.
- If the providers of the information are close to the names
- If you need “global” access
- If the information is PUBLIC

ogud@ogud.com & pk@denic.de

To do DNS or not to do DNS

- Private/confidential data
- Access control needed
- Large data
- Unstructured
- Naming is difficult
- You need search or match capability

To do DNS or not to do DNS

- Some misconceptions to keep in mind
  - Name does not imply location
    - Dnsop.de name and web servers can reside anywhere in the world
  - Replication of DNS data and providing service close to consumers is easy
    - Use anycast
      - There are over 100 locations providing root server function right now.
    - Set up secondary, turn on TSIG, Notify and IXFR
      - Either official one or use query forwarding

Other choices than DNS

- DHCP: if data is consumed locally
  - much better choice
- Service location (see above) and also depends on if accessed via local resource or more “global” one.
  - Enterprise vs site location
  - No search
- Distributed databases

Readily Available Building Blocks

- Address Records (A/AAAA)
- SRV Records
- NAPTR based schemes
  - DDDS
  - S-NAPTR
  - U-NAPTR
- Custom Designed RR type

ogud@ogud.com & pk@denic.de

SRV Record

- mostly used in MS Active Directory and other OS specific applications
  - Also used by some IM application like Jabber.

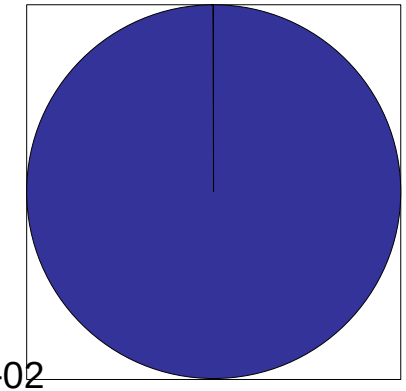
- recurring task: given (new) service named COOL, need to offer it
  - old solution: aliases "`ftp`", "`www`", ...
  - problem: needs well known port, no exceptions;
    - single target (server) or approximately evenly distributed across multiple addresses

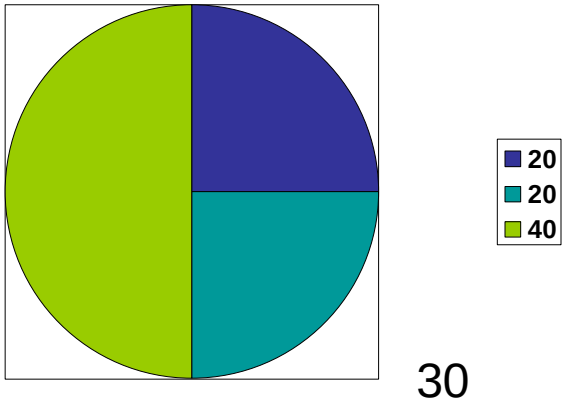
Generalize MX: that COOL SRV

- COOL service in example.org

```
_cool._tcp.example.org  SRV  0  0 5133 srv55.mega.example
_cool._tcp.example.org  SRV 10 20 9876 srv33.mega.example.
_cool._tcp.example.org  SRV 10 20 3456 srv44.mega.example.
_cool._tcp.example.org  SRV 10 40 6738 srv66.mega.example.
```

  "_" avoids conflicts with hostnames
- Services need to be registered
  - currently under discussion: separate registry
  - this is not too good for local service location (-> tree climbing)





ogud@ogud.com & pk@denic.de

When to use SRV

- SRV works best if you have a TCP or UDP service and want to be able to delegate and distribute
- SRV is widely deployed and supported
- See RFC 2782

ogud@ogud.com & pk@denic.de

NAPTR

- Role: map name to set of services represented by URI
- SRV doesn‘t help
  - No local part
  - No variable scheme
- Naming Authority Pointer: NAPTR
  - order 16 bit value
  - preference 16 bit value
  - flags character-string
  - service character-string
  - regexp character-string
  - replacement domain-name

ogud@ogud.com & pk@denic.de

NAPTR frameworks

- NAPTR record does not stand on its own
- DDDS == Dynamic Delegation Discovery System
  - Used in ENUM and ONS (the RFID name space)
    - These create their own name spaces
    - RFC 3401-3405
- S-NAPTR == SRV and NAPTR combined
  - Avoids application specific DDDS overhead
    - RFC 3958
- U-NAPTR == NAPTR maps to single URI
  - Avoids the rewrites
    - RFC 4848

Design Choices for placing new information in DNS.

- New class
  - You need to supply the root servers for it ☺
- New Suffix (TLD)
  - Talk to ICANN
- Reuse TXT (or some other type)
- <prefix>.name
- New Type

ogud@ogud.com & pk@denic.de

Existing DNS Record Types:

- DNS Internal types
  - NS, SOA, DS, DNSKEY, RRSIG, NSEC
    - Only used by DNS for its operation
- Indirect RR:
  - CNAME, DNAME
    - Indirect DNS RR cause Resolver to change direction of search
      - Server must have special processing code
- Terminal RR:
  - Address records
    - A, AAAA,
  - Informational
    - TXT, HINFO, KEY, SSHFP
      - carry information to applications
- Non Terminal RR:
  - MX, SRV, PTR, KX, A6, NAPTR, AFSDB
    - contain domain names that may lead to further queries.
- META:
  - OPT, TSIG, TKEY, SIG(0)
    - Not stored in DNS zones, only appear on wire

ogud@ogud.com & pk@denic.de

Placing New information in DNS: Reuse existing Type

- Needs careful consideration if type is used by core protocols
  - Record type does not stand on its own, needs resolution context before it is useful
  - RBL use A for policy information
    - **BUT** only in non routable address space (127/8)
- TXT may appear as the obvious choice
  - No semantics
  - RFC 1464 sub-typing
  - prefixing could help, but has its own problems
  - TXT wastes space, this is still important
  - If new RRSet is large you want EDNS0 support
    - Modern software does this and unknown types as well!!!!
      - MORAL: Fight for local upgrades, do not force the whole Internet to work around your local issues.

Placing New information in DNS: Name prefix, magic name

- Selector put in front of (underneath) domain name:
  - `_axfr.example.org APL 1:127.0.0.1`
  - May interfere with zone maintainer's naming policy
  - Prefix may end up in a different zone
  - Wildcards will not work like expected, i.e. `_prefix.*.example.org` does not expand
  - No registry for prefixes
- Magic name, e.g. `www`
  - Overloading of multiple names in single application server
  - Again may conflict with naming policy

New RR Type Benefits

- Full control over contents
- Application centered semantics
- Simpler for applications to parse
  - If your specification is simple: KISS
- No collisions, smaller
- Resolution context provided

ogud@ogud.com & pk@denic.de

How to get a new DNS RR type

- Technical Rules (based on RFC 3597)
  1. No additional section processing
  2. No name compression of embedded domain names
  3. Clean definition, no overly complicated structure
- OLD Process:
  1. Write an ID, get review by people that understand your protocol, update draft.
  2. Ask DNS experts (WG chairs) for quick review, update ID
  3. Ask WG(s) for review
  4. Submit to IESG, you get type code from IANA after IESG processes
  5. Advertise new type code
- New Process (RFC2929-bis in IETF last call)
  - Fill out template from RFC2929-bis and send to IANA
  - IANA will forward template to an expert and conduct a public review
  - DNS expert will render decision based on guidance in 2929bis

ogud@ogud.com & pk@denic.de

New Type design guidance

- Tailor it to your needs,
  - Be specific
  - Restrict flexibility (avoid being overly generic)
- Be compact, binary fields are fine
- Ask the experts for help early
  - DNSEXT and DNSOP chairs will help

How to enable the use of new type?

- Provide tools to
  - convert new RR type from/to textual format to RFC3597 portable format for zone inclusion,
  - Provide dynamic update tool of new types.
    - Good tool kits: NET::DNS, DNSJava, DNSpython
- Assume software is modern !!
  - Modern Servers: (partial list)
    - BIND-9, MS DNSServer2003, NSD, PowerDNS, ANS, CNS

Pointers to more information

- IETF working groups
  - DNS EXTensions: http://www.dnsext.org
  - DNS Operations: http://www.dnsop.org
- Individual sites
  - http://www.dns.net/dnsrd
  - http://www.dnssec.net

ogud@ogud.com & pk@denic.de

DNS More resources

- DNS book list
  - http://www.networkingbooks.org/dns

RFC starting reading list

- DNS related RFC 100+
  - Many obsoleted
- Important ones
  - 1034, 1035 Original specification
  - 4033, 4034, 4035 DNSSEC
  - 1123, 2181 Clarifications
  - 3597, 2136, 1996, 1995, 3007 Major protocol enhancements
  - 3833 Threat Analysis for DNS

End of talk

- Extra information provided in background slides
- Questions & Comments

Current DNS Infrastructure

- Old implementations still around as authoritative/caching servers
  - Declining population: due to security concerns
- Middle boxes have old DNS software or their own implementation that is non-compliant
  - Some Load balancers do stupid things,
  - Applications interfaces refuse to ask for unknown types
  - Assume world is still RFC1034/5 (i.e. 1987),
    - NO: AD bit, TSIG, OPT, NAPTR,
  - Think name compression is mandatory.
  - Increasing population
- Majority of the infrastructure
  - is RFC3597 enabled.
  - has EDNS0 support
    - TCP DNS query are sometimes blocked.

DNS Operational considerations

- Low TTL: if TTL is low RRSet is cached for short time and frequent lookups are required:
  - negative effects: DoS on self and infrastructure, slower lookup,
  - positive effects: Highly dynamic and allows primitive load balancing
- Bad delegations:
  - Timeouts may happen due to no reachable name servers
- Old Software still in use after vendor recommends retirement

ogud@ogud.com & pk@denic.de

DNS Operational Excuses

- Adding data to reverse tree is problematic
  - Due to ownership of namespace.
- Specifying a new RR type is sometimes opposed based on “arguments”:
  - Not supported by our software
    - Provisioning, Authorative servers, resolvers, firewalls, middleboxes,
      - take your pick.
  - Do not feel like it
    - Turf war, politics ….
    - …..

DNS Sub Typing ISSUE

- DNS responses MUST consist of complete RRSets
  - You cannot query for a subset of the RRSet
  - ... nor for partial matches (only QNAME, QTYPE, QCLASS)

- I.e. you cannot ask for, say, at most eight address records (A RRs) for a given name or for only those MX RRs with priority 10 or all TXT RRs containing "money".

- Some RR types are "containers", e.g.
  - KEY (the original)
  - NAPTR
  - TXT (with the RFC1464 convention)

- Subtyping means that the application will have to select their RRs from the response, potentially dumping larger parts of the RRSet, depending on one or more secondary qualifiers buried within RDATA
- ENUM NAPTR overload

SubTyping side effects

- Subtyping results in larger responses
  - (wasted bandwidth) [well, large RRSets are always a DDoS vector]
  - danger of truncation
  - TCP based re-queries
- Subtyping should be avoided when designing new types
- Subtyping can be avoided by
  - dedicated types instead of type/subtype
  - selector prefixes (cf SRV)
- Method of choice depends on number and nature of subtypes expected and the necessity to deal with wildcards

DNS is not for search

- Tree climbing == BAD
- Few applications have said that if RR does not exist at name then look for zone default at apex,
  - Zone cut is hard to find by stub resolvers,
  - hierarchy in naming does not necessarily imply hierarchy in network administration.
  - Although DNS name space is hierarchic, there's no inheritance zone wide defaults are also bad due to "apex overload"

Optimization considered evil

- Problem:
  - Frequently Non-terminal records proposed demand that, terminal records be returned in answer ==> Additional section processing
- Facts:
  1. Additional section processing is done in servers
  2. Before updated servers are deployed RRtype aware resolvers need to do all work.
  3. Not all authoritative servers may have the necessary glue
  4. Glue may not fit
  5. Recursive resolver may have data already
  6. Roundtrips are cheap,
  7. Lacy resolver writer will ASSUME additional section processing is done
- Result:
  - Recursive Resolver has to be able to do work forever,
- Moral: Do not attempt to optimize DNS, it causes more problems than you can imagine.

DNS Query Model: Question → Answer

Stub_resolver -> Recursive_Resolver → Auth Server[1]

←

…….

Recursive_Resolver → Auth Server[n]

←

← Recursive_Resolver

Stub_resolver has an answer and returns that to the application.

DNSSEC: impacts

- Zones
  - become larger
  - need periodic maintenance
  - have to deal with key management
- Resolvers need to know Secure Entry Points to signed sub trees.
  - Changes over time, needs updating.
- implementations supporting DNSSEC:
  - NDS, BIND-9, DNSJava, DNSpython, Net:DNS, NDS, ANS, CNS
    - Microsoft will support in the near future.