

**The large volume DNS Platform for
TLD Registries wanting presence near ISPs**
Based on registered Patents in UK, EU and USA

**Building a “Community DNS” platform
with Quality of Service**

Table of Contents

Community DNS.....	1
The Purpose.....	3
Special Features	3
The Administration Web Interface	4
Step 1 – Registration	4
Step 2 – Administering the Zones	4
Restrictions.....	5
Updates using Dynamic DNS Interface	5
Adding an entity to the zone	6
Deleting a name from the zone	7
Updates to a zone using the AXFR Interface.....	8
Standard Supported Resource Records Types	9

The Purpose

The purpose of Community DNS is to provide a DNS service that can resolve any TLD registered name in the world. This is achieved with an extremely high speed high volume DNS server designed specifically for the purpose.

Each server is capable of answering just over 100,000 queries per second when holding a database of 500 million entities. Additionally the servers can be clustered to provide higher performance and fail over, with a linear performance increase less approximately 10%. That is, two server would provide about 180,000 queries second, three servers 270,000 queries per second etc.

This document assumes the reader has a good working knowledge of DNS & its terminology, the operation & administration of zones and is familiar with RFC1035, RFC2136 and RFC2845.

Special Features

The Community DNS Server has a number of special features unique to its DNS software.

- 1) If an entry (name) has no Resource Records of type TXT (16) then queries for that name will always return the fixed string "Registered". If TXT Resource Records exist for that name then they will be returned instead.

If no registration exists for that name then the standard error NXDOMAIN will be returned.

- 2) Queries for the Resource Record type ANY (255) will return all Resource Records currently stored against that name.
- 3) Requests for the top level SOA record will always return the server's current Unix Time as the serial number regardless as to when the last update occurred. Any sub-domain SOA records must be passed to the Community DNS Server using the standard DDNS mechanism.
- 4) A (1) and AAAA (28) *glue* records will be returned as "Additional" records where the information is available when queries result in NS (2) or MX (15) records being returned.

Any query that results in no information being available to return, or requesting an unsupported feature, will result in the error NXDOMAIN being returned.

The Administration Web Interface

Administration of the system is done through an SSL web interface with a standard web browser. Currently the Web Interface is extremely simple and it is not envisaged that this would continue to be the interface for production use, but is there to get testing up and running as quickly as possible.

Step 1 – Registration

The first step in using the system is to register. All that is required is an e-mail address and password. These will then be used as a user's login credentials when administering their zones. Although the same e-mail address may not be used multiple times, a user can re-register with different logins if they wish to keep different sets of zones separate.

After registering, the user will be invited to login.

Step 2 – Administering the Zones

Once a user has registered and logged in, they can then add zones to their account. Once the zone information has been transferred over the zones will be served out by the Community DNS Server. When adding a zone the user must specify the names or IP Addresses of the DNS servers that will provide the original zone information and then specify whether that information will be provided by Dynamic DNS (D/DNS) or polled AXFR.

Note, a zone can be updated using both AXFR and D/DNS. For example, if a zone has a refresh interval of 12 hours the AXFR service will transfer the zone, difference it and create journal transactions twice a day. Then between AXFR updates, D/DNS updates can be sent which will also be converted into journal transactions and played into the zone in real time.

This will mean that because the D/DNS updates are not visible in the previously stored AXFR zone file, all of the D/DNS transactions will be repeated when the next AXFR refresh interval occurs. However, this should not effect the contents of the zone once all the AXFR transactions have been played.

If you wish to operate a zone as both D/DNS & AXFR then you should specify it as AXFR.

If you wish to operate a zone as D/DNS we would recommend that you first run it as AXFR in order to rapidly seed all the zone data.

Restrictions

The Community DNS Server will store any Resource Records it is given, including non-standard Resource Records allowing it to support a range of proprietary extensions. However, it stores both the Resource Record Type and Length as single bytes imposing a range of 0 to 255 on the Resource Record type and maximum length of 255 bytes on each Resource Record.

The Community DNS Server supports decoding standard DNS message compression as specified in RFC1035 Section 4.1.4, however, it will only correctly decode compression on host names for Resource Records of type NS (2), MX (15), PTR (12) & CNAME (5). All other Resource Records should be sent **without** the use of DNS message compression.

The Community DNS Server will always respond with NXDOMAIN to IXFR (251) and AXFR (252) requests.

The TTL specified by the upper level server is ignored, but a system wide TTL is used, as specified by the server's system administrator. This will be changed in future versions.

Updates using Dynamic DNS Interface

Updates to the data on the Community DNS Servers can be taken from the upper level (e.g. ccTLD) via a subset of RFC2135 compliant DDNS UDP updates. All examples will be given in the text format support by the program "nsupdate" as supplied with bind v9.

All DDNS UDP packets will be acknowledge once the packet has been decoded and committed to the Community distribution and transport mechanism.

Adding an entity to the zone

When an entity is to be added to the zone **all** resource records for that entity must **always** be presented, in the same DDNS UDP packet, and must be presented in contiguous Resource Record entries. When an update is received for an entity **all** previous resource record information will be discarded and replaced with the information that has just been received. Hence **all** resource records for that entity must be presented in any update to that entity.

Glue records may be presented in a separate DDNS UDP packet, but again, resource records for the same entity must be presented contiguously.

Here is an example of adding a zone with IPv4 & IPv6 glue records for two of its DNS servers :-

```
server 192.168.250.2
update add domain.co.uk 86400 ns NS1.NSTLD.COM
update add domain.co.uk 86400 ns NS2.NSTLD.COM
update add domain.co.uk 86400 ns NS1.domain.co.uk
update add domain.co.uk 86400 ns NS2.domain.co.uk
update add NS1.domain.co.uk 86400 A 1.2.3.4
update add NS1.domain.co.uk 86400 AAAA 1080::5:800:200c:417a
update add NS2.domain.co.uk 86400 A 5.2.3.4
update add NS2.domain.co.uk 86400 AAAA 1055::5:800:200c:417a
send
```

As you can see, in this example, it is permitted to mix updates to different entities within the same DDNS packet so long as the information for each entity is contiguous.

Any number of entities can be added in a single DDNS UDP packet as per the restrictions of the underlying transport.

The entity's name specified in the *Zone Section* (ZNAME) must reflect the name of the first resource record, even if subsequent resource records refer to a different entity.

Deleting a name from the zone

To remove a name from the zone the upper level information provider should issue a DDNS delete specifying Resource record type NS (type=2). This will only remove data associated with the record specified. Any associated glue records that need to be removed will have to be removed individually.

“nsupdate” example :-

```
server 192.168.100.2
update delete domain.co.uk ns
send
```

This example of “nsupdate” script will remove the domain **domain.co.uk** from the zone along with all associated Resource Records. Any subsequent query for the name **domain.co.uk** will result in the error NXDOMAIN. Note, however, any sub domain information will not be deleted unless there is a separate command issued specifically to do so.

So, for example, if the domain **domain.co.uk** has the name servers **ns1.domain.co.uk** and **ns2.domain.co.uk** these will have to be removed with separate command. For example :-

```
server 192.168.100.2
update delete domain.co.uk ns
update delete ns1.domain.co.uk ns
update delete ns2.domain.co.uk ns
send
```

Note, that although the records for **ns1.domain.co.uk** and **ns2.domain.co.uk** do not have any NS Resource Records, the Resource Records type of NS (2) is taken as a command to remove all Resource Records. All other remove command will be ignored.

Updates to a zone using the AXFR Interface

Updates to the Community DNS Server can also be performed using a standard Zone Transfer, AXFR. The Zone is transferred to a central point where the Resource Records are compared with a previously stored copy of the zone and journal transactions are created if any difference is found (add, modify or delete).

Periodically (as specified in the SOA Refresh Interval) the AXFR service will request an SOA record over UDP. If the serial number in the SOA reply is higher than the serial number of the currently stored zone, then the AXFR service will open a TCP connection and request a full Zone Transfer (AXFR).

As the zone comes in, the records will be checked for data integrity and if any checks fail the Zone Transfer will be aborted, and the AXFR service will then wait until the next refresh interval before starting the process again by issue SOA requests.

The AXFR service supports talking to a number of upper level servers. When the refresh interval expires it will issue a UDP SOA request to every upper level server that it knows about. It will then start an AXFR zone transfer to the first upper level server that responds with a higher serial number than the it is currently holding. Thus the service will be unaffected if any of the upper level servers go down, so long as at least one of them continues to get updated copies of the zone. If a Zone Transfer is in progress when subsequent (later) SOA packets are received from other servers, the later SOA packets will be ignored.

Once the Zone has been successfully transferred the service loads the previous copy of the zone, performs some basic sanity checks, then compares the newly transferred copy of the zone with the one previously stored, creating journal transactions for any changes. Once the journal transactions have all been successfully sent to the transport subsystem, and stored in non-volatile storage, the new zone will be saved as the current zone, the current serial number will be stepped on and the refresh timer will be incremented by the refresh interval.

If the AXFR service receives a DNS **NOTIFY** packet from any of the upper level servers specified for that Zone, it will acknowledge the **NOTIFY** and reset the refresh timer for that zone to a time less than 2 minutes in the future.

The System Administrator can optionally set an upper limit on the refresh interval, effectively allowing the system administrator to force the AXFR service to re-poll the SOA Serial number more frequently.

Standard Supported Resource Records Types

A (1);
NS (2);
MD (3);
MF (4),
CNAME (5),
SOA (6),
MB (7),
MG (8),
MR (9),
NULL (10),
WKS (11),
PTR (12),
HINFO (13),
MINFO (14),
MX (15),
TXT (16),
RP (17),
AFSDB (18),
X25 (19),
ISDN (20),
RT (21),
NSAP (22),
NSAP_PTR (23),
SIG (24),
KEY (25),
PX (26),
GPOS (27),
AAAA (28),
LOC (29),
NXT (30),
EID (31),
NIMLOC (32),
SRV (33),
ATMA (34),
NAPTR (35),
KX (36),
CERT (37),
A6 (38),
DNAME (39),
SINK (40),
OPT (41)

In addition to these standard Resource Records types, the Community DNS Server will also support the use of any non-standard Resource Record types up to 255.