The Root of the Matter: Hints or Slaves

David Malone
Communications Network Research Institute
Dublin Institute of Technology
Ireland
<dwmalone@cnri.dit.ie>

ABSTRACT

We consider the possibility of having a (recursive) name server act as a slave to the root zone, rather than caching information after it is requested. Tests, described here, indicate that this technique seems to be comparable to the traditional hints mechanism for moderately busy name servers and may offer other benefits such as reducing the number of bogus requests going to the root servers. With further refinement the technique may be operationally useful, but the impact on root servers would have to be fully assessed.

Categories and Subject Descriptors

C.2.6 [**Computer-Communication Networks**]: Internetworking

General Terms

Measurement, Performance

Keywords

DNS, Root Name Server, Zone transfer

1. INTRODUCTION

The Domain Name System [1, 2, 3] is a hierarchical distributed database that allows the lookup of keys such as `www.example.com`, where the dots separate the levels in the hierarchy. To look up a record we first ask a *root* name server for information about the key `www.example.com`, and it will respond with a message pointing at the name servers for `.com`, then we send a message to one of these name servers and they will respond with a message pointing at the name servers for `.example.com` and finally we ask one of these, who provide information about `www.example.com`. Caching is well provided for within DNS: we are told how long to remember who the servers for `.com` and `.example.com` are in responses. This reduces the load on the name servers closer to the root of the tree. Negative caching, ie. caching of records that are found not to exist, is also provided for. Typically, negative responses are cached for a short time.

DNS requires very little boot-strap information and should be able to resolve any query beginning with only a list of root name servers. Traditionally, DNS servers have been provided with a list of addresses that are root name servers at the time the DNS server software was written. The DNS server then queries these addresses requesting the current list of root name servers, and once it gets a response it is ready for operation.

To allow for redundancy within DNS, multiple name servers are permitted (and encouraged) for each point in the hierarchy. For ease of management, the DNS protocol defines a mechanism to transfer all of the data at a point in the hierarchy from one name server to another. This mechanism is called a *zone transfer*. This means that the name servers for a zone may be either masters, who have first hand access to the data in the zone, or slaves, who copy it from a master using a zone transfer. A serial number and various lifetimes are included in the zone so a slave knows how frequently the master must be contacted to keep the zone up to date. While most DNS requests are completed using UDP, zone transfers use TCP.

Some of the root name servers allow zone transfers of the root zone, meaning it is possible to to configure any name server as a slave for the root zone. Why would one want to do this? A server that has a copy of a zone can respond to any request with a positive or negative answer immediately. In particular, we do not need to issue a query for each and every distinct name that would result in a negative response. This makes it possible to 'cache' negative results until the zone expires and a new copy of the zone must be transfered. However, it also means that the complete zone must be transfered even if only a small portion of it will be used.

How common are queries to the root name server for negative results that are not repeated? Results in [5] show that a significant number of requests (14–27%) to the root servers result in a negative response, but a smaller number (4–6%) to the GTLD servers result in a negative response. This suggests that a reasonable number of queries might be avoided by having a copy of the root zone at hand.

Our aim is to compare the traditional hints method with the alternative of acting as a slave for the root zone. There are two groups who might benefit from such a change. The first group is the users of the name server, who may see a reduction in outgoing traffic and faster response times. The second group is the root name servers, who might receive

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
IMC'04, October 25–27, 2004, Taormina, Sicily, Italy.
Copyright 2004 ACM 1-58113-821-0/04/0010 ...$5.00.

fewer queries. Evidence already suggests that much of the traffic to the root name servers is in some way bogus [7] and that if bogus queries can be stopped at the local name server, then that may be useful.

2. SETUP

Our method is to monitor the traffic from two name servers to the root name servers over a period of two weeks. At the end of the first week, we change the name servers from the traditional hints method to be slaves for the root zone.

The two DNS servers were both running FreeBSD 4 and the name server software was BIND 9.2.1.

The first DNS server is used by a small research group of 10–20 people. It serves a quiet mail and web server and the workstations of the research group. We refer to this server as the *quiet server*.

The other DNS server is busy, acting as a name server to an academic department providing Unix services to 1000 undergraduates, postgraduates and staff, including a mail server, a web cache, a web server and an anonymous ftp server. These services will generate quite a high DNS load because the web server and TCP wrappers have been configured to do forward and reverse lookups for most connections. This name server also acts as a primary or secondary server for around 70 zones in each of two 'views' (see [6]). This network also has IPv6 connectivity: the mail and web server are IPv6 capable, though the web cache is not. We refer to this server as the *busy server*.

The activity in the systems should be relatively independent — they are in different Universities, but are connected to the same ISP. The quiet server acts as a tertiary mail exchanger for the busy system. This may result in some correlation in the DNS load due to the delivery of spam/viruses.

DNS traffic from both systems was collected using tcpdump starting on Tuesday 28 January 2003 continuing for a week using the traditional hints configuration, and then beginning on Tuesday 4 February 2003 for the slave configuration. Traffic from the quiet server was collected by running tcpdump on that system. Traffic from the busy server was collected by running tcpdump on a router that sees all external traffic. The tcpdump command line is shown in Figure 1. To ensure that the nameservers' caches were empty, both nameservers were restarted as measurements began.

For the first week, both servers used the traditional hints mechanism, configured as shown in Figure 2. For the second they both acted as slaves for the root zone and used the configuration shown in Figure 3, suggested by Doug Barton on the FreeBSD-stable mailing list [9].

Note that the busy name server operates two BIND views and so the root zone must be configured in both views. For the purposes of this experiment it was configured to obtain two independent zone transfers from the root name servers for the two views and place them in two different files. In normal operation it would only be necessary to get one zone transfer from the root name servers and then the zone could be transfered to other views locally.

At the time of these experiments, the root zone was about 55KB on-disk.

3. RESULTS AND ANALYSIS

Figure 4 and Figure 5 show a summary of the traffic volumes involving the root name servers and our quiet and

```
#!/bin/sh
tcpdump -p -i fxp1 -n \
    -w SERVER-‘date +%Y%m%d%H%M%S‘ \
    ’host SERVER_IP and \
    port 53 and \
    ( host 198.41.0.4 \
    or host 128.9.0.107 \
    or host 192.33.4.12 \
    or host 128.8.10.90 \
    or host 192.203.230.10 \
    or host 192.5.5.241 \
    or host 192.112.36.4 \
    or host 128.63.2.53 \
    or host 192.36.148.17 \
    or host 192.58.128.30 \
    or host 193.0.14.129 \
    or host 198.32.64.12 \
    or host 202.12.27.33 )’
```

Figure 1: Tcpdump parameters used for recording DNS traffic to/from the root servers

```
zone "." {
        type hint;
        file "named.root";
};
```

Figure 2: BIND 9 config for hinting root zone

```
zone "." {
        type slave;
        file "s/root";
        masters {
                128.9.0.107;
                192.33.4.12;
                192.5.5.241;
        };
        notify no;
};
```

Figure 3: BIND 9 config for slaving root zone

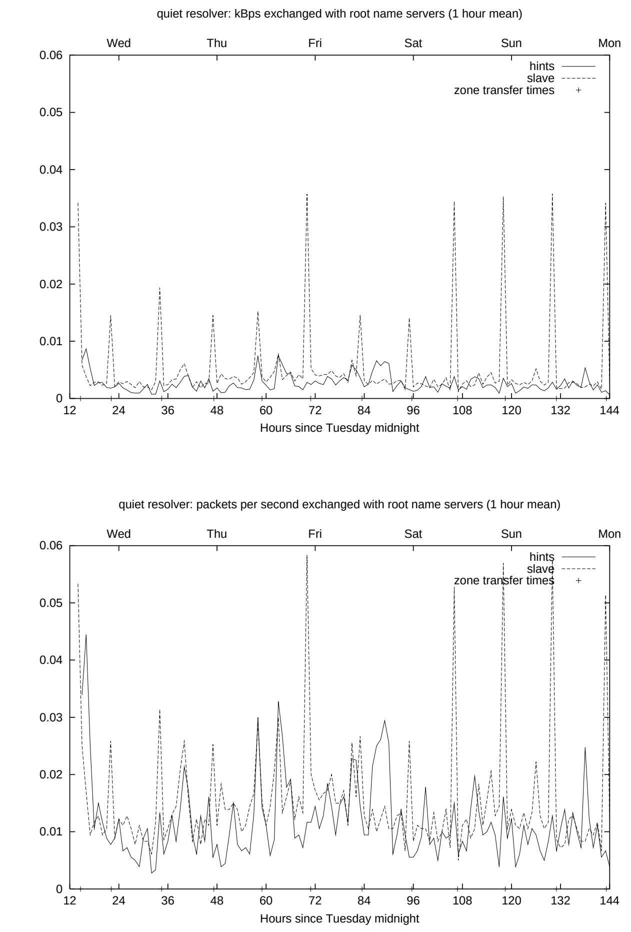

Figure 4: Quiet Server DNS Traffic

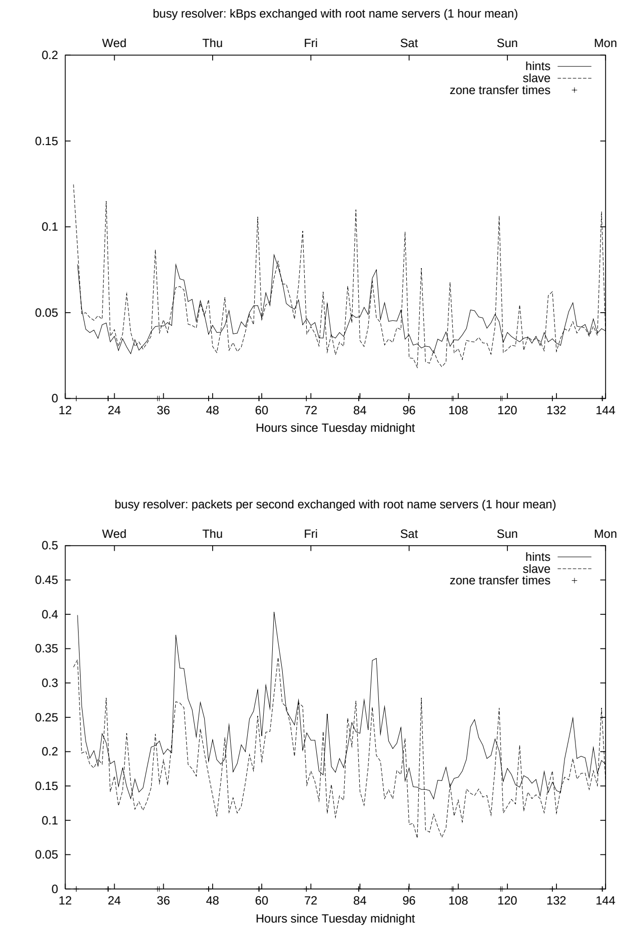

Figure 5: Busy Server DNS Traffic

busy DNS servers respectively. Each graph shows the 1 hour means for the byte or packet rate for both the hint (solid line) and slave (dotted line) schemes over the duration of the experiment. If means are calculated over a shorter time scales then we naturally get noisier, peakier results, but the general pattern is similar.

Note that the graphs only show the first 144 hours (6 days) of the data; around hour 156 of the first week's measurements a network outage disconnected the quiet server from the Internet at large. This resulted in a huge peak in outgoing DNS traffic while BIND repeatedly reissued requests until the network was reconnected.

Several interesting observations come from examining the graphs. First, the traffic for both configurations is quite similar in volume — even after inspecting the graph it is not obvious which method caused a smaller amount of traffic. Second, the slave configuration shows peaks that clearly correspond to zone transfers for the quiet server (the zone transfer times are marked with ticks along the bottom of each graph). For the busy server there are additional peaks that are not attributable to zone transfers. We might expect no traffic to be sent to the root name servers between zone transfers, however we do see some chatter between them, which is discussed below.

Table 1 shows summary statistics for the experiment. We show the packet and byte rates for all traffic, traffic coming into our DNS server and traffic emanating out from our DNS server. For the slave cases we also show the ratio of the slave method to the corresponding statistic in the hint method. The table indicates that acting as a slave decreases the number of packets and bytes transmitted in both the busy and quiet cases. For the busy server it also results in a reduction in the number of packets received.

To avoid the spike in the traffic mentioned above, the summary statistics were calculated again over the first 144 hours of the trace. The results are shown in Table 2. We can see that the statistics do not change significantly.

Further analysis of the (truncated) trace was performed by parsing the output of tcpdump. About 2% of the packets were truncated[1] to an extent that it was not possible to process them beyond the ID field.

TCP based queries are not parsed well by tcpdump, as it does not do TCP stream reassembly, but all the TCP queries are zone transfers. Over the 144 hours of the truncated trace the quiet server performed 13 zone transfers and the busy server 26 zone transfers. The difference is explained by the fact that the busy server was performing a zone transfer for each view. As noted above, this is not necessary in a more refined configuration. For the quiet server, the average zone transfer lasted, 2.5s transferring 81kB in 111 packets. The busy server averaged 1.7s, transferring 103KB in 134

[1]Remember to use `-s 0` or `-s 1500`.

	quiet hints		**quiet slave**		**slave/hint**	
Direction	Packets/s	bytes/s	Packets/s	Bytes/s	Packets	Bytes
both	0.0215	3.2420	0.0156	5.1331	73%	158%
in	0.0059	2.1961	0.0079	4.6279	133%	211%
out	0.0157	1.0459	0.0078	0.5051	50%	48%
	busy hints		**busy slave**		**slave/hint**	
Direction	Packets/s	bytes/s	Packets/s	Bytes/s	Packets	Bytes
both	0.2151	46.8377	0.1753	47.4230	81%	101%
in	0.1075	39.2026	0.0878	41.0068	82%	105%
out	0.1076	7.6351	0.0875	6.4162	81%	84%

Table 1: Trace statistics, full trace

	quiet hints		**quiet slave**		**slave/hint**	
Direction	Packets/s	bytes/s	Packets/s	Bytes/s	Packets	Bytes
both	0.0233	3.4348	0.0155	5.2111	67%	152%
in	0.0060	2.2772	0.0078	4.7125	130%	207%
out	0.0173	1.1576	0.0077	0.4986	45%	43%
	busy hints		**busy slave**		**slave/hint**	
Direction	Packets/s	bytes/s	Packets/s	Bytes/s	Packets	Bytes
both	0.2089	45.1614	0.1711	46.7172	82%	103%
in	0.1044	37.8130	0.0857	40.4514	82%	106%
out	0.1045	7.3484	0.0854	6.2658	82%	85%

Table 2: Trace statistics, truncated trace

packets. Note, that the root zone did not actually change 13 times during the experiment, but its serial number is regularly incremented to check that zone transfers to the public root servers are working correctly.

A breakdown of the queries by type is shown in Table 3. In this table we have counted certain classes of bogus queries separately. In particular, requests for records corresponding to dotted quads or addresses enclosed in '[ ]' have been counted separately. This highlights a bug in a locally maintained MTA running near the busy server, as it is making requests for MX records for dotted quads enclosed in '[ ]'.

These tables show that the number of queries made is significantly lower in the slave case, as we expect. Acting as a slave also largely eliminates the obviously bogus requests that we have counted separately — this makes sense as the transfered zone is a complete cache, enabling the server to immediately determine that the top level zones 1–255 and '1]'–'255]' do not exist.

The other startling thing highlighted by the table is the number of A6 queries being made by BIND 9.2.1. BIND's 'v6-synthesis' option is not enabled on either server and, to the best of our knowledge, no resolver served by either server uses A6 records. These queries may be related to BIND internally making A6 queries for glue information that it has found in its zone files.

The SOA queries shown for the slave configuration can be attributed to BIND ensuring that its copy of the root zone is up-to-date. Note that (on average) BIND sent a SOA query to *all three* of the listed masters every 1600s. The frequency of these queries is controlled by one of the parameters within the SOA record and so can be controlled by the root zone maintainers.

In the slave configuration, we might only expect to see requests of type SOA or AXFR. However, in practice we see significant numbers of A, A6, AAAA and PTR queries. Some of these queries can be explained by noting that the domains `arpa`, `in-addr.arpa`, `mil` and `root-servers.net` are served by various subsets of the root servers. This may account for the PTR queries and 423 queries issued for `mil` and `root-servers.net` by the busy slave configuration (the quite slave issued no queries for these domains).

Still, this leaves a significant number of queries unexplained. The names being queried seem to be distributed throughout the DNS namespace without any obvious pattern. It is interesting to note that there are no unexplained MX queries. This may suggest that the unexplained queries are somehow generated as system queries by BIND.

Table 3 also shows the number of requests that resulted in an error response. All the observed errors were of type NXDomain or FormErr. NXDomain is error indicating that the DNS record does not exist and allowing negative caching. FormErr indicates that the server could not parse the request. We can see from the tables that when acting as a slave to the root zone, almost all NXDomain error are eliminated, as expected. In fact, in the busy server case, the reduction in NXDomain responses accounts for 11160 of the 11923 queries saved by acting as a slave to the root zone. The numbers of FormErr errors is small in all cases, but, regardless, acting as a slave to the root zone does reduce the number of queries resulting in this error.

Browsing the list of domains provoking NXDomain responses in the non-slave case shows domain names that are not fully qualified, domain names with typos[2], Unix tty names and private names such as `localhost.localdomain` or `loopback`. We observed one application that, as a user

[2]Some names obviously contain HTML, suggesting they have come from links in web pages, others seem likely to be URLs mistyped into browsers.

Wait time in seconds		
all types		without SOA
quiet hints	**quiet slave**	**quiet slave**
103.1	205.7	56.8
busy hints	**busy slave**	**busy slave**
3700.3	2099.0	1806.4

Table 4: Wait times

types a name `aa.bb.cc`, looks up the domains `a`, `aa`, `aa.b`, `aa.bb`, ...!

In all cases, all of the FormErr errors and a significant number of the NXDomain errors were associated with queries containing an additional resource record included in the query (denoted by '[1au]' in the tcpdump output). It seems these queries are related to probing for EDNS0 support [4]. These accounted for 220 of the quiet-hint NXDomains, all 12 of quiet-slave NXDomains, 6531 of the busy-hint NXDomains and all 22 of the busy-slave NXDomains.

Finally, an attempt was made to determine how long BIND spent waiting for responses to queries sent to root servers. BIND will usually focus queries on remote servers with a low response time and probe other servers periodically to update estimates of response time. However, it is difficult to tell which queries are probes and which queries are 'real'. Thus a wait time was calculated by summing min(response time, 5s) over all UDP queries (zone transfers are discussed above). Queries outstanding at the end of the trace are assumed to be answered at the time the trace ended. Note that this wait time statistic should not be taken as indicative of how long clients usually wait.

The calculated wait times are shown in Table 4. The quiet server shows a longer wait time when slaving the root zone and the busy server shows a shorter wait time. When a breakdown of the waiting times by type was examined, it transpired that a significant amount of time was spent waiting for a response to SOA requests in the slave case. These requests should not have an impact on the server's ability to respond to a query, so we also show the total waiting time without these requests included. For both the quiet and busy server, the total waiting times without SOA requests is halved by acting as a slave. It is worth observing that the servers listed in Figure 3 are not the closest root servers, and when given a choice BIND will choose issue queries to a closer server.

4. CONCLUSION

The results seem to indicate that acting as a slave to the root zone results in a saving in the number of packets and queries involving the root servers. However, it is not clear that it results in a reduction in the byte volume of traffic; if anything the total volume of traffic seems to have increased. Zone transfers also involves a number of TCP connections to the root servers. As TCP connections are persistent, this will cause extra overhead on the root servers and it is unclear if the reduction in the number of queries would offset this load.

Because of this state, TCP connections are less suitable for anycast DNS deployment[8], which is now commonly being used to distribute multiple instances of root servers throughout the Internet. The short timescales associated with these zone transfers does mean that normal routing changes are unlikely to be a problem, but that per-packet load sharing to different instances of a root server might be. The fact that the zone information does not immediately become invalid combined with the mitigating factors described in [8] mean that zone transfers from root servers should still work acceptably well in practice.

If the use of zone transfers of the root zone became common then the number of TCP connections could be significantly reduced by changing the root zone's serial number only when the zone changed, or by incrementing the serial number less frequently for maintenence/testing purposes. Similarly, the number of SOA queries could be reduced by adjusting the lifetimes listed in the SOA response. Other flexibility might be gained by noting that the root name server addresses listed in the hints file, in the root zone and as masters for the root zone need not all be the same.

We have not considered the possibility of taking a single transfer of the root zone and then making the DNS server a master for the root zone, using that static data. This technique offers many of the benefits of the slaving technique described here. Though changes to the root zone are relatively infrequent (every week or so), the data will gradually become stale. It would be possible to automate this process, but then we're heading back in the direction of acting as a slave and doing zone transfers.

One incidental benefit of using TCP-based DNS is that TCP responses are not limited in size as UDP responses are. In the case of some queries, particularly queries returning the name servers for top level domains, the response data has been tailored to accommodate the maximum number of IPv4 addresses of name servers into the standard UDP response. This practice is complicating the addition of AAAA records to show where IPv6 name servers for domains can be found, as there is no space left in a standard UDP response. Thus, acting as a slave might simplify the addition of AAAA glue to the root zone.

One area where slaving the root zone seems particularly strong is in the reduction of bogus queries (dotted quads, [address], typos, unqualified domains). One question is whether this technique could be usefully generalised to other zones. The root zone was a good candidate: it is a small zone, the average name server will access many of its records and typos are likely to lie within the zone. Other zones, such as the `arpa` and `in-addr.arpa` zones might also benefit, as they are likely to be small and frequently queried. However, these are less likely to be queried for names with typos (though we did see queries corresponding to classic DNS typos such as `ns1.ivm.net.254.247.195.in-addr.arpa`, these typos cannot be detected in the `in-addr.arpa` zone).

A large number of queries made seem to be for A6 records. It is unclear why BIND is making these queries, but it may be attempting to obtain A6 records for glue records. The responses to these queries will often contain additional information records, meaning that the request may not be wasted, but it seems strange for BIND to direct these queries to the root servers at all. If these requests are unnecessary and can be eliminated, then the gains of slaving the root zone could be larger. It is more likely that these requests will be replaced by AAAA queries in BIND 9.3.

Two other aspects of BIND's behaviour also came to light in this study. One is the choice to send a SOA query to all masters for a zone once per refresh period. For our purpose,

Type	**quiet hints** Queries	**quiet slave** Queries	**slave/hint** ratio	**quiet hints** NXDomain	**quiet slave** NXDomain	**quiet hints** FormErr	**quiet slave** FormErr
A	810	318		53		11	4
A....	28			28			
A6	7469	1647		25		21	8
A6....	28			28			
AAAA	147			73		1	
AAAA....	9			9			
AXFR		13					
MX	6						
PTR	380	343			12	2	2
SOA		970					
SRV	6			6			
total	8883	3291	37%	222	12	35	14
except A6	1386	1644	119%	169	12	14	6
Type	**busy hints** Queries	**busy slave** Queries	**slave/hint** ratio	**busy hints** NXDomain	**busy slave** NXDomain	**busy hints** FormErr	**busy slave** FormErr
A	11206	9158		1405	1	31	4
A....	315			314			
A6	31460	29418		33		24	15
A6....	268			265			
AAAA	720	157		555			
AAAA....	53			53			
ANY	2			2			
AXFR		26					
MX	558			492	12		
MX....	115	13		115			
MX[]	7899			7895		4	
NS	5					2	
PTR	660	647		7	9	1	3
SOA		1943					
SRV	24			24			
total	53285	41362	77%	11160	22	62	22
except A6	21557	11944	55%	10862	12	38	7

Types ending with '....' are queries for IPv4 dotted quads. Queries ending with '[]' are queries for addresses enclosed in brackets.

Table 3: Breakdown of number of requests and number of errors by query type

it would be better if BIND round-robined or contacted the closest master. The second is the reissuing of requests when BIND does not receive any responses. Acting as a slave to the root zone could actually help spread the burden in cases where some important name server becomes unresponsive.

This initial study suggests that further work in this area may be warranted. Running two name servers in parallel on the same set of client queries or conducting a longer trial might produce a more definitive comparison. Investigating if BIND can be tweaked to deal more elegantly with the slaving of zones in this way would also be interesting. It may even be possible to automatically determine when it is beneficial to perform a zone transfer for a particularly active zone.

5. ACKNOWLEDGEMENTS

I would like to thank Doug Barton, Niall Murphy and Niall O'Reilly for advice and feedback; João Damas and Jim Reid for answering my questions; the anonymous reviewers for offering insight and raising several interesting questions.

This work has been supported by Science Foundation Ireland under the National Development Plan.

6. REFERENCES

[1] P. Mockapetris, "Domain names — concepts and facilities," RFC 1034, November 1987.

[2] P. Mockapetris, "Domain names — implementation and specification," RFC 1035, November 1987.

[3] R. Elz and R. Bush, "Clarifications to the DNS Specification," RFC 2181, July 1997.

[4] P. Vixie, "Extension Mechanisms for DNS (EDNS0)," RFC 2671, August 1999.

[5] J Jung, E Sit, H Balakrishnan and R Morris "DNS Performance and the Effectiveness of Caching" Proc. ACM SIGCOMM Internet Measurement Workshop, 2001.

[6] Internet Software Consortium, "BIND 9 Administrator Reference Manual" 2000–2004.

[7] N Brownlee, kc Claffy and E Nemeth, "DNS Measurements at a Root Server," Cooperative Association for Internet Data Analysis, 2001.

[8] T. Hardie, "Distributing Authoritative Name Servers via Shared Unicast Addresses," RFC 3258, April 2002.

[9] D Barton et al, "discussion on root hints file," FreeBSD-stable mailing list, `http://groups.google.ie/groups?selm=fa.m8djbmi.161q9ga%40ifi.uio.no`, January 2003.