

## Re: BIND 9.4.1 performance on FreeBSD 6.2 vs. 7.0

---

*Source:* <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/performance/2007-06/msg00051.html>

---

- *From:* Chuck Swiger <[cswiger@xxxxxxx](mailto:cswiger@xxxxxxx)>
  - *Date:* Thu, 14 Jun 2007 16:53:01 -0700
- 

Hi, Kris--

This was interesting, thanks for putting together the testing and graphs.

On Jun 14, 2007, at 1:48 AM, Kris Kennaway wrote:

I have been benchmarking BIND 9.4.1 recursive query performance on an 8-core opteron, using the resperf utility (dns/dnsperf in ports). The query data set was taken from [www.freebsd.org](http://www.freebsd.org)'s httpd-access.log with some of the highly aggressive robot IP addresses pruned out (to avoid huge numbers of repeated queries against a small subset of addresses, which would skew the results).

It's at least arguable that doing queries against a data set including a bunch of repeats is "skewed" in a more realistic fashion. :-) A quick look at some of the data sources I have handy such as http access logs or Squid proxy logs suggests that (for example) out of a database of 17+ million requests, there were only 46000 unique IPs involved.

You might find it interesting to compare doing queries against your raw and filtered datasets, just to see what kind of difference you get, if any.

Testing was done over a broadcom gigabit ethernet cable connected back-to-back between two identical machines. named was restarted in between tests to flush the cache.

What was the external network connectivity in terms of speed? The docs suggest you need something like a 16MBs up/8 Mbs down connectivity in order to get up to 50K requests/sec....

[ ... ]

It would be interesting to test BIND performance when acting as an authoritative server, which probably has very different performance characteristics; the difficulty there is getting access to a suitably interesting and representative zone file and query data.

## Re: BIND 9.4.1 performance on FreeBSD 6.2 vs. 7.0

I suppose you could also set up a test nameserver which claims to be authoritative for all of in-addr.arpa, and set up a bunch (65K?) /16 reverse zone files, and then test against real unmodified IPs, but it would be easier to do something like this:

Set up a nameserver which is authoritative for 1.10.in-addr.arpa (ie, the reverse zone for 10.1/16), and use a zonefile with the \$GENERATE directive to populate your PTR records:

```
$TTL 86400
$origin 1.10.in-addr.arpa.

@ IN SOA localhost. hostmaster.localhost. (
1 ; serial (YYYYMMDD##)
3h ; Refresh 3 hours
1h ; Retry 1 hour
30d ; Expire 30 days
1d ) ; Minimum 24 hours

@ NS localhost.

$GENERATE 0-255 $.0 PTR ip-10-1-0-$.example.com.
$GENERATE 0-255 $.1 PTR ip-10-1-1-$.example.org.
$GENERATE 0-255 $.2 PTR ip-10-1-2-$.example.net.
; ...etc...
```

...and then feed it a query database consisting of PTR lookups. If you wanted to, you could take your existing IP database, and glue the last two octets of the real IPs onto 10.1 to produce a reasonable assortment of IPs to perform a reverse lookup upon.

--  
-Chuck

---

freebsd-performance@xxxxxxxxxxx mailing list  
<http://lists.freebsd.org/mailman/listinfo/freebsd-performance>  
To unsubscribe, send any mail to "freebsd-performance-unsubscribe@xxxxxxxxxxx"