Behavior of DNS' Top Talkers, a .com/.net View

Eric Osterweil[1], Danny McPherson[1], Steve DiBenedetto[2], Christos Papadopoulos[2], and Dan Massey[2]

[1] VeriSign Labs
[2] Colorado State University

Abstract. This paper provides the first systematic study of DNS data taken from one of the 13 servers for the .com/.net registry. DNS Generic Top Level Domains such .com and .net serve resolvers from throughout the Internet and respond to billions of DNS queries every day. This study uses gTLD data to characterize the DNS resolver population and profile DNS query types. The results show a small and relatively stable set of resolvers (i.e. the *top-talkers*) constitute 90% of the overall traffic. The results provide a basis for understanding for this critical Internet service, insights on typical resolver behaviors and the use of IPv6 in DNS, and provides a foundation for further study of DNS behavior.

1 Introduction

The Domain Name System (DNS) [9] is one of the Internet's core protocols and is essential to looking up Internet resources. The DNS translates names to IP address, identifies the SMTP servers for email addresses, and provides a wide range of other mappings. Virtually every Internet application depends on some form of DNS data, and this makes it critical Internet infrastructure.

In addition, the DNS is also quite flexible and extensible. It has been extended to include security extensions[4] and the IETF has multiple working groups investigating new uses [1, 2]. Researchers are investigating both DNS behaviors and the potential impact of design changes[11, 3, 7, 5]. Characterizing the use of the DNS at the top level can be quite useful for anyone trying to understand the global DNS or add new extensions to this critical service. However, there are still many aspects of the DNS that have yet to be investigated. These missing pieces are not simply corner cases. To the contrary, top level DNS domains such as .com and .net are some of the largest and most widely used DNS zones, but relatively little is known about their characteristics and the characteristics of their client resolvers. In fact, due to the caching behavior of DNS, large TLD zones see more traffic diversity than even the root zone. Thus, observations from the largest TLDs (.com and .net) offer the greatest aggregate view of global DNS traffic. We discuss this further in Section 2. There are no profiles of the resolvers contacting these TLDs and no profiles to provide even basic information such as the types, names, and frequencies of queries.

In this paper we present the first study of all resolver query traffic seen by g.gtld-servers.net (G GTLD), one of the 13 sites serving the two largest

TLDs in the Internet today: `.com` and `.net`. Our study uses data collected from the second quarter of 2011. To address confidentiality restrictions, we do not list specific dates in the graphs. By observing queries at the G GTLD server, one obtains a view of resolvers from throughout the Internet and observes clients ranging from the caching recursive resolvers of large ISPs to smaller *stub* client DNS tools running on end systems. The data volume to this single instance of the 13 `.com`/`.net` name server set was in excess of 900 million queries and roughly 900 thousand unique sources *per day*. These numbers represent typical daily traffic counts and do not include any large attack traffic. To the best of our knowledge, this is the largest study of resolver traffic and query patterns to date. We use this query traffic to create empirical profiles of all resolvers seen.

2 Background

The domain names in DNS form a tree-like hierarchical name space in which each node is called a *domain*. At the top of the tree, the `root` domain delegates authority to *Top Level Domains* (TLDs) like `.com`, `.net`, `.org`, and `.edu`. The `.edu` domain then delegates authority to create the `colostate.edu` domain, `.com` delegates authority to create `verisign.com` domain, and so forth. The repository of information that makes up the domain database is divided up into logical name spaces called *zones*, which each belongs to a single administrative authority and are each served by a set of *authoritative name servers*. The multiple servers for each zone provide redundancy and fault tolerance.

Clients that query for DNS information are called *resolvers*. Typically, an end system (desktop, laptop, smartphone, etc.) is called a *stub resolver* and only implements a very small portion of the DNS resolution process. These stubs are typically configured with the address of one or more local caching resolvers, to which they send all of their queries. The local caching resolver is configured with the IP addresses of the DNS root servers and if no other information is cached, the caching resolver starts by send a query to the root server. For example, to find the IP address for `www.verisign.com`, a caching resolver will first query a root server and the root server will refer the caching resolver to the `.com` servers. The caching resolver will then query one of the `.com` servers (for the entire domain name, `www.verisign.com`) who will refer the caching resolver to the `verisign.com` servers. Finally, the caching resolver will query one of the `verisign.com` servers who return the desired address for `www.verisign.com`.

All DNS data is stored in *Resource Records* (RRs), and each RR is essentially a tuple ⟨`name, TTL, class, type`⟩. For example, an IPv4 address for `www.verisign.com` is stored in an RR with name `www.verisign.com`, class IN (Internet), type A (IPv4 address), and the record indicates how long it should be cached in its TTL field.

Finally, it is important to note a resolver may have the authoritative servers for popular TLDs in cache. The caching resolver learns of the `.com` servers upon its first query to ⟨`anything`⟩`.com` and should cache this information for two days (the TTL value specified in the RR). Thus in our example above, the caching

resolver almost certainly does not start the query for `www.verisign.com` at the root servers. Instead, the caching resolver has cached the authoritative servers for `.com` and begins by querying one of these servers. This means that client resolvers will forward the first query for a Second Level Domain (SLD) to the Top Level Domain. Thus, traffic to `.com` will show a view of all active sub domains of `.com`. By contrast, none of these queries are sent to the root, because recursive resolvers will have already cached `.com` (a more specific match). This is why the largest TLDs (`.com` and `.net`) see much more traffic that even the root zone.

3 Profiling Resolvers

For the second quarter of 2011, we examined all DNS query traffic sent to g.gtld-servers.net (G GTLD); one of the authoritative name servers for `.com` and `.net`. During this study we recorded `pcap` files from a span port and digested their contents into the following meta-data from each query as a 5-tuple: ⟨`time, source IP, destination IP, query type, query name`⟩. Query types include `A`, `AAAA`, `MX` and all other DNS RR types that were actually seen. Query names were fully qualified DNS domain names such as `www.somezone.com`.

As one might expect, the G TLD server receives queries from millions of sources including institutional DNS resolvers, mail servers, polling systems, botnets, laptop users typing commands like `dig`, and so forth. In order better understand both DNS operations and the nature of query sources, we examined a set of DNS-specific features to help us quantifiably profile the resolvers observed in our study. Based on the scope of the `.com` and `.net` TLDs, we believe that these results provide the largest, most diverse, and perhaps the most detailed profile of global resolver behavior to date.

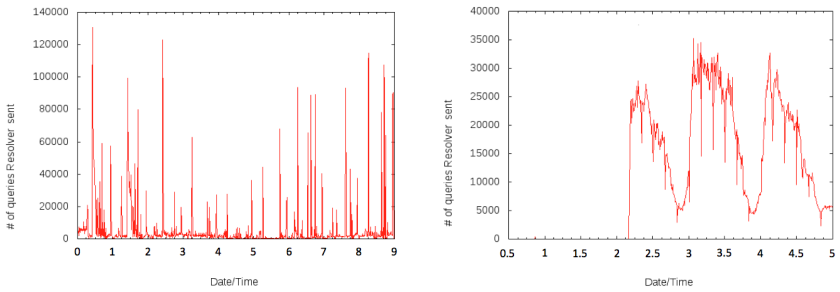
3.1 What We Can Observe

Due to the fact that our data is taken from a very large TLD registry, we generally do not expect to see end systems such as web browsers or smart phones in our study. This is because these stub resolvers typically send their queries to a local caching resolver. This local caching resolver will either service the request from its cache (if it can), or forward the request to the TLD (and then serve future requests for that name from its cache). Mapping between end system addresses and caching resolver addresses is of interest to services such as content distribution networks, and remains an open and active area of research[8]. However, our study does not make effort to map caching resolvers back to stubs.

While we will not see *every* caching resolver in the Internet, we do expect to see a large portion of the full list of caching resolvers that send traffic to `.com` or `.net`. In Section 2 we explained that resolvers query TLDs while looking for referrals. However, even though our data is only taken from one of the 13 sites that comprise these TLDs, we still claim that over time we will observe queries

from almost all caching resolvers. This is because of a behavior we call *polling and pinning*.

There are several main variants of DNS resolvers that are commonly deployed today: ISC's BIND, NLnet Labs' unbound, PowerDNS, and Microsoft's DNS. Each of these servers attempts to provide its users with the fastest possible resolution. One way they do this is to measure (or *poll*) the Round Trip Time (RTT) to each authoritative name server for each DNS zone they query. Generally, they each have an algorithm to choose (or *pin* themselves to) a specific name server for each zone that appears to be responding the fastest. Furthermore, the polling process is generally ongoing so that the resolvers can adapt to changing network conditions and in some types of servers, the polling volume and frequency is amortized on existing query traffic. As a result, we expect that over time, every resolver that uses this approach will send at least polling queries to our monitored G GTLD site, and sometimes polling will result in a resolver changing its selection and re-pinning itself to a new server.

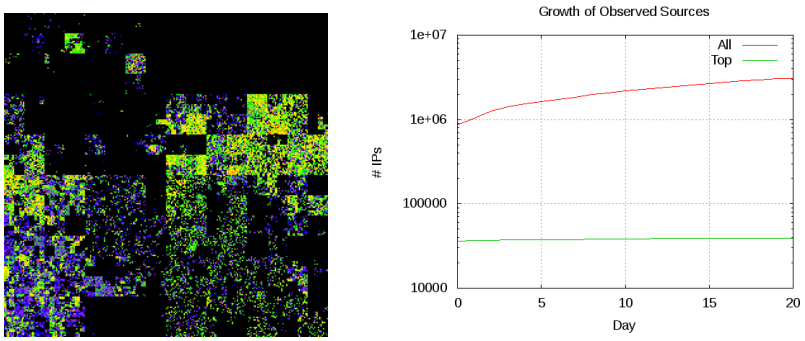


(a) This resolver periodically sends bursts as it pins/unpins itself to the G server.

(b) This resolver polls briefly and then 2 days later pins itself to G with a diurnal pattern.

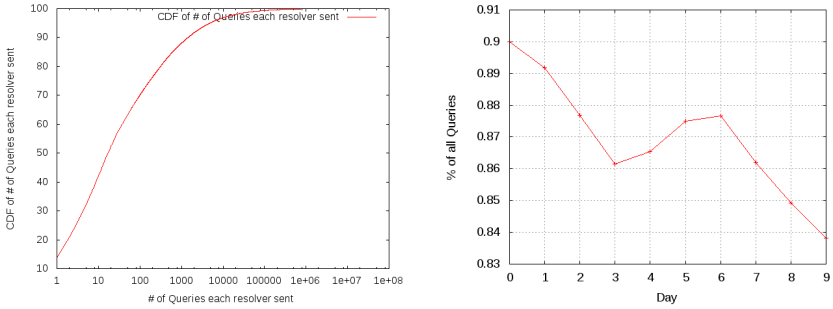
Fig. 1. Pinning and polling behavior of two resolvers.

For example, Figure 1(a) shows the polling behavior of one resolver over the course of five days. Figure 1(b) shows a resolver that had been seen polling at a very low rate and volume for two days before re-pinning itself from another server instance to G GTLD. We can see that server selection does occur, but the approach for selecting an authoritative server is implementation dependent and even varies between versions of the same implementation[6]. Some implementations may simply select a preferred server and pin themselves to the server without polling the zone's other authoritative servers. A complete discussion of server selection is beyond the scope of this paper, but more information can be found in [12, 10].



(a) Location of Sources Querying G (b) Cumulative Number of Sources Sending Queries

Fig. 2. Who is querying `g.gtld-servers.net`.



(a) Resolver Query Rates: A relatively small number of resolvers account for 90% of all Queries

(b) An unchanging top-talkers list (rather than keeping it rolling) causes it to account for less of the overall traffic after 10 days.

Fig. 3. Query rates and top-talkers.

3.2 Who Talks To The G GTLD Server?

Figures 2(a) and 2(b) provide a high level overview of the sources sending queries to the G GTLD server. Figure 2(a) is a Hilbert graph showing the location of all sources that contacted the G GTLD server during one example ten minute period. This Hilbert graph divides the Internet IPv4 address space into /16 address blocks. The color indicates the volume of coming from that address block. The large empty spaces correspond to unallocated or inactive address. For example, the upper right corner of the graph corresponds to the multicast address space; which should not be used as the source in any DNS queries. The main point of the figure is that, even in a relatively short ten minute span, the G GTLD server does indeed serve the global Internet. This figure is very similar during other periods.

The top curve in Figure 2(b) shows the cumulative number of source addresses over a 20 day period. We can see that initially there is a brief super-linear learning phase. This is a cold-cache artifact of our measurements, and shows that some query sources have longer inter-query periods, and take longer to appear in our measurements. By contrast, active caching resolvers from campus networks, organizations, and ISPs are frequently querying the `.com` and `.net` zones and quickly appear in our data set. Caching resolvers that serve smaller user bases send less frequent queries and likely make up the population of resolvers that take longer to appear. Inactivity due to time of day and caching resolver polling behavior (discussed above) can also delay the time it takes for the G GTLD server to observe the first query from a caching resolver.

Following the initial learning phase, the figure shows there is a constant growth of unique IP sources. Even 20 days into the study, the G GTLD server continues to discover new resolvers at a rapid pace (note the log scale). There are many legitimate explanations for resolvers growth such as a network administrator configuring a new resolver, a user may use the `dig` command to send a query directly to G GTLD server, a DNS monitoring tool may query G GTLD server. There are also illegitimate behaviors in this growth such as bots directed to send attack traffic and attacks using spoofed addresses. As one may expect, a large portion of the "slow to appear" legitimate sources send a very small amount of traffic; many as little as a single query. In contrast, resolvers that send a high volume of queries during some period are classified as *top-talkers* and we examine this group in more detail.

3.3 Query Volume and Top-Talkers

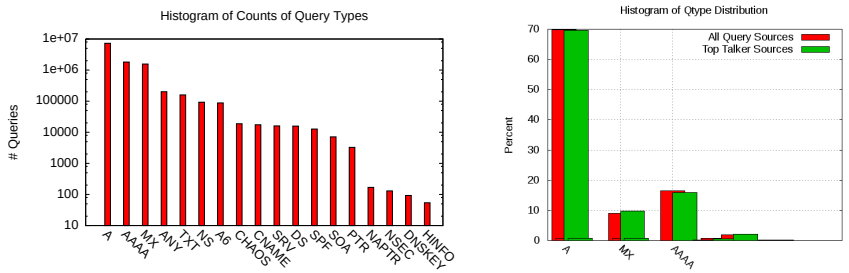
While the set of all resolvers numbers in the millions and continues to grow in our study, Figure 3(a) shows 90% of the overall traffic is generated by just under 40,000 resolvers. This indicates that the large-scale behavior seen at the `.com`/`.net` TLDs is dictated by a relatively small number of query sources. We call these resolvers *top-talkers*. The lower flat line in Figure 2(b) shows the cumulative number of *top-talkers* and indicates that top-talkers are discovered quickly. The set of *top-talkers* is also dynamic, as the behaviors of resolvers change over time. There are long term structural changes where new resolvers are added, old resolvers are retired, users migrate, and new services are provisioned. In addition, there are observable shorter term patterns as load changes due to the time of day, the day of the week, and even due to routing changes. To account for the long term structural changes and shorter term patterns, we developed a rolling list of *top-talkers* where at any given moment the list is based on the previous seven days of data.

In order to see the dynamism in this list, we first compared two top-talkers list from different months. At the beginning of one month, the top-talker list included 39,304 source IP addresses. At the beginning of the following month, the top-talker list included 39,936 sources. 30,071 of the sources were common to both lists. Next, a separate examination (seen in Figure 3(b)) found that the accuracy of a top-talkers list degraded daily if it was *not* continually updated.

By definition, the top-talker list initially accounts for 90% of all queries seen on day 0. As seen in Figure 3(b), if we keep the list fixed, the top-talker list accounts for only 84% of all queries seen on day 10.

3.4 Query Type Profiling

Figure 4(b) shows the distribution of query types over our study period. The figure shows queries for IPv4 addresses (type `A`) clearly dominate. IPv6 addresses (`AAAA`) are the second most popular type requested and requested more frequently than mail server (`MX`) records. Other types including DNSSEC record types[4], service location (SRV) record types, and even obsolete `A6`[3] records are all observed in Figure 4(a), but we can see that they constitute only small portion of the the query traffic. Note the type distribution is roughly the same for top-talkers and the entire set of query sources.



(a) Full qtype distribution of all qtypes seen (note log-scale).

(b) Qtype distribution for all sources and top-talkers.

Fig. 4. Qtype histograms

To help digest the query type behaviors of resolvers, we developed a profiling scheme, seen in Figure 5(a). This figure shows the profile types and also shows how both resolvers and total queries distribute across the profiles during one study day (though the distribution on other days is similar). The top node in the tree shows that we observed 975,391,055 queries from 958,558 distinct addresses.

In the figure, the line thickness of the node's border is proportional to the percent of resolvers that fall into the node. The top node includes all resolvers and has the thickest line. The leaf node second from the left still accounts for a substantial percentage of all resolvers and has a proportional thickness. Very few resolvers fall into the rightmost leaf node and it is proportionally less thick.

Our profile begins by considering whether the resolver sent any `A` record queries, the most popular type observed in our study. 94.07% of all resolvers

[3] A6 was an alternate IPv6 address format and moved to experimental status in 2002, see `http://www.ietf.org/mail-archive/web/dnsop/current/msg05437.html`

observed on this day sent at least one query for A records. These 901,762 resolvers next split into two nearly equal groups. 55.63% of these resolvers also sent queries for AAAA records and 44.37% never requested an AAAA record.

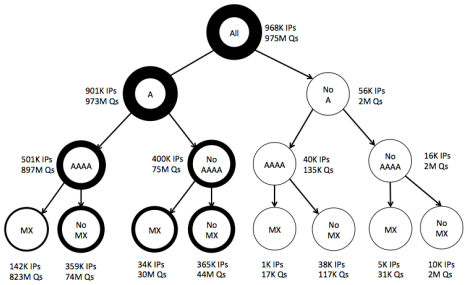
Profiling resolver behaviors can help one detect unexpected behaviors or attacks, better simulate resolvers, and analyze DNS behaviors. For example, this distribution has implications for issues such as tracking the IPv6 deployment progress. It is encouraging that over half of resolvers that request IPv4 addresses also request IPv6. However, this DNS query behavior does not directly translate to IPv6 usage. For example, many operating systems and versions of web browsers, automatically request both IPv4 and IPv6 addresses. Thus, these systems may request both IPv4 and IPv6 addresses in parallel, thus causing the local resolver resend both. Corresponding behavior is observed in the data; a query for the IPv4 address of www.somezone.com is followed in close succession by a query, from the same source, asking for the IPv6 address of the same name. Interestingly, there were 40,211 resolvers whose DNS queries were for IPv6 AAAAs only. These 40,211 resolvers sent a combined 122,998 queries and yet never request a single IPv4 address. Again, note this is the behavior observed from the TLD and does not definitively show these resolvers are IPv6 only.

Completing our query type profile, the bottom row of the tree considers whether a resolver sends queries for MX records. The leftmost leaf node in Figure 5(a) are resolvers that send A, AAAA, and MX queries. For example, one type of resolver that fits in this group is the resolver for a small or medium company. The company caching resolver supports users who request both A and AAAA as well as the company mail server that requests MX records. Note there are also many other examples of resolvers that can fit this profile. Another resolver that fits this group may be a mail server that does its own DNS resolution; the MX records are requested by the mail server and A/AAAA queries are for those servers.

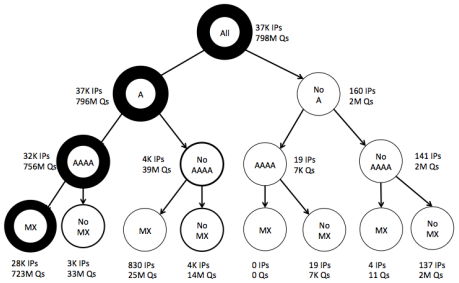
Finally, it has been said that virtually all types of behavior that can occur in the Internet will occur at some point. The rightmost leaf node would support this statement. We observe DNS resolvers that do not request any A, AAAA, or MX records but *do* query for other types of records. On this day, there were 10,658 such resolvers. Examples of the resolvers here include various DNS monitoring tools that may solely check for DNS name server (NS) records or DNSSEC records or other specialized features. The data in Figure 5(a) includes all resolvers, even resolvers that sent only one query during the day. This one query may happen to be polling and may happen to request say TXT record. In fact, the low volume talkers could potentially skew the numbers in any one of the nodes.

Figure 5(b) shows the profile (on the same day) for only the top-talkers. The restriction to top-talkers still covers 90% of the traffic, but reduces the total number of resolvers from over 900,000 to under 40,000. Note the top-talkers are more likely to request AAAA records in addition to A. Of all top-talkers, over 80% of the resolvers who request IPv4 records also request IPv6 records. This can be partly explained by polling by resolvers. Consider a resolver who has selected a server other than G as its preferred .com/.net server. Further suppose during the day that this resolver sends one poll query to G GTLD server. The poll

query is chosen from the existing queries sent by the caching resolver and the most popular query type is `A`. If the one poll query is sent during the the day is an `A` query, to G it will appear the caching resolver is only requesting `A` records. Similarly, if this one query happens to be an `AAAA` query, to the G GTLD server it will appear the caching resolver is only requesting `AAAA` records. Top talkers send large numbers of queries and their profiles are more likely to be an accurate reflection of the resolver type distribution.



(a) Resolver Query Type Profiles - Includes All Resolvers



(b) Resolver Query Type Profiles - Includes Only Top-Talkers

Fig. 5. Resolver taxonomy.

This classification scheme proves to be useful in using query type distributions to infer certain high level roles that resolvers may have, and possibly (as in the case with `A6` queries) be able to identify specific types of resolvers.

4 Conclusions

In this paper we present what we believe to be the first analysis of DNS query traffic at one of the `.com/.net` TLD servers that serves data to the entire Internet. Our analysis includes characterization of the IP addresses of the DNS resolver population, the distribution of query volume including the top-talkers,

and query type profiling. We conducted our measurements during periods that did not have DDoS traffic to avoid measurement bias.

The motivation for our analysis is to define and understand important features of normal resolvers. Analysis of query types provides insights on issues ranging from identifying typical resolver behaviors to the use of IPv6 in DNS. What we found is that resolvers display an interesting *pinning and polling* behavior pattern, the query type distribution highlights just three types as the main targets of traffic, and that the prominence of `A6` queries demonstrates the old Internet adage that, "nothing ever really dies in the Internet."

Our results indicate that of all of the hundreds of millions of queries seen at this one instance of the `.com`/`.net` name servers, the bulk of them come from a relatively small and dynamic group. Under 40,000 resolvers account for the 90% of the traffic and this set of *top talkers* evolves daily. Based on these measurements we conclude that when measuring large-scale DNS behaviors, the top-talkers dominate the observed behavior and serve as a useful low-pass filter when trying to characterize typical traffic patterns. For example, many clients were seen to have issued a single query during our measurement period, but including them in analysis of (say) average query rates of caching resolvers, the results become biased. The top-talkers allow our results to focus on typical, well-maintained, and active resolvers when determining the global query behaviors.

References

1. DANE. `https://datatracker.ietf.org/wg/dane/charter/`.
2. ENUM. `http://tools.ietf.org/wg/enum/`.
3. Bernhard Ager, Wolfgang Mühlbauer, Georgios Smaragdakis, and Steve Uhlig. Comparing DNS Resolvers in the Wild. In *Proceedings of ACM IMC 2010*, Melbourne, Australia, November 2010.
4. R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. Protocol Modifications for the DNS Security Extensions. RFC 4035, March 2005.
5. Nevil Brownlee and Evi Nemeth. Dns measurements at a root server. In *In Proceedings of IEEE Global Telecommunications Conference (Globecom '01)*, 2001.
6. Internet Software Consortium. ISC BIND Features. `http://www.isc.org/software/bind/new-features/11.6/`.
7. Jaeyeon Jung, Emil Sit, Hari Balakrishnan, and Robert Morris. DNS Performance and the Effectiveness of Caching. In *Internet Measurement Workshop*, 2001.
8. Zhuoqing Morley Mao, Charles D. Cranor, Fred Douglis, Michael Rabinovich, Oliver Spatscheck, and Jia Wang. A precise and efficient evaluation of the proximity between web clients and their local dns servers. In *USENIX Annual Technical Conference*, 2002.
9. P. Mockapetris and K. J. Dunlap. Development of the domain name system. In *SIGCOMM '88*, 1988.
10. NLnet Labs. unbound DNS resolver. `http://www.unbound.net/documentation/`.
11. Eric Osterweil, Michael Ryan, Dan Massey, and Lixia Zhang. Quantifying the operational status of the dnssec deployment. In *IMC '08*, 2008.
12. Xin Wang Zheng Wang and and Xiaodong Lee. ANALYZING BIND DNS SERVER SELECTION ALGORITHM. *International Journal of Innovative Computing, Information and Control*, 2010.