第 XV 部

DNS extension and operation environment

第 15 部
DNS extension and operation environment

第 1 章 Brief description of the DNS working group

The DNS working group discusses DNS operation and DNS protocol extensions. This group is mainly researching operation of top level domain servers, including a root name server, and DNS measurement.

In particular, we are currently studying the following topics.

- DNS operation

  We maintain several major DNS servers (e.g., m.root-servers.net and e.dns.jp), and we are analyzing incoming traffic against these servers for study of DNS operation issues.

- DNS normalization

  DNS is a highly fault tolerant system. Even with misconfiguration, DNS often works as though it does not have any operational or configuration problems. Ironically, this good characteristic has caused many misconfigured servers, which often increase the work load of top level domain servers.

  To improve the situation, we are also working on DNS normalization, cooperating with JPNIC and JPRS. The normalization activity includes categorization of misconfiguration, research on the effect of the misconfiguration, developing and providing tools to detect badly configured servers.

Our major activities in this year were discussions on the above topics in WIDE camps and meetings or at the working group mailing list. We have summarized the discussion results into separate documents, some of which has been officially published as WIDE drafts or Internet Drafts.

In this report, we combine the published results along with our current focus as follows. Chapter 2 describes some misbehavior against DNS queries for IPv6 address, while Chapter 3 explains operational guidelines for ".local" zones in the DNS. Chapter 4 reports address transition of a DNS server of the "JP" zone we are operating.

All of the issues are related to our operational activity in live networks, and some results of the first two issues are incorporated into our normalization work. We believe the results are helpful for other operators as well.

第 2 章 Common Misbehavior against DNS Queries for IPv6 Addresses

(This document was originally submitted to the IETF as an Internet Draft.)

2.1 Introduction

Many DNS clients (resolvers) that support IPv6 first search for AAAA Resource Records (RRs) of a target host name, and then for A RRs of the same name. This fallback mechanism is based on the DNS specifications, which if not obeyed by authoritative servers can produce unpleasant results. In some cases, for example, a web browser fails to connect to a web server it could otherwise. In the following sections, we describe some typical cases of the misbehavior and its (bad) effects.

Note that the misbehavior is not specific to AAAA RRs. In fact, all known examples also apply to the cases of queries for MX, NS, and SOA RRs. The authors even believe this can be generalized for all types of queries other than those for A RRs. In this report, however, we concentrate on the case for AAAA queries, since the problem is particularly severe for resolvers that support IPv6, which thus affects many end users. Resolvers at

end users normally send A and/or AAAA queries only, and so the problem for the other cases is relatively minor.

2.2 Network Model

In this report, we assume a typical network model of name resolution environment using DNS. It consists of three components; stub resolvers, caching servers, and authoritative servers. A stub resolver issues a recursive query to a caching server, which then handles the entire name resolution procedure recursively. The caching server caches the result of the iterative query as well as sends the result to the stub resolver. The authoritative servers respond to queries for names for which they have the authority, normally in a non-recursive manner.

2.3 Expected Behavior

Suppose that an authoritative server has an A RR but not a AAAA RR for a host name. Then the server should return a response to a query for a AAAA RR of the name with the RCODE being 0 (indicating no error) and with an empty answer section[195]. Such a response indicates that there is at least one RR of a different type than AAAA for the queried name, and the stub resolver can then look for A RRs for the same name.

This way, the caching server can cache the fact that the queried name does not have a AAAA RR (but may have other types of RRs), and thus can improve the response time to further queries for a AAAA RR of the name.

2.4 Problematic Behaviors

There are some known cases at authoritative servers that do not conform to the expected behavior. This section describes those problematic cases.

2.4.1 Return NXDOMAIN

This type of server returns a response with the RCODE being 3 (NXDOMAIN) to a query for a AAAA RR, indicating it does not have any RRs of any type for the queried name.

With this response, the stub resolver immediately give up and never fall back. Even if the resolver retries with a query for an A RR, the negative response for the name has been cached in the caching server, and the caching server will simply return the negative response. As a result, the stub resolver considers this as a fatal error in name resolution.

There have been several known examples of this behavior, but all the examples that the authors know have corrected their behavior as of this writing.

2.4.2 Return NOTIMP

Other authoritative servers return a response with the RCODE being 4 (NOTIMP), indicating the servers do not support the requested type of query.

This case is less harmful than the previous one; if the stub resolver falls back to querying for an A RR, the caching server will process the query correctly and return an appropriate response.

In this case, the caching server does not cache the fact that the queried name has no AAAA RR, resulting in redundant queries for AAAA RRs in the future. The behavior will waste network bandwidth and increase the load of the authoritative server.

Using SERVFAIL or FORMERR would cause the same effect, though the authors have not seen such implementations yet.

2.4.3 Return a Broken Response

Another different type of authoritative servers returns broken responses to AAAA queries. A known behavior of this category is to return a response whose RR type is AAAA, but the length of the RDATA is 4 bytes. The 4-byte data looks like the IPv4 address of the queried host name. That is, the RR in the answer section would be described like this:

```
www.bad.example. 600 IN AAAA 192.0.2.1
```

which is, of course, bogus (or at least meaningless).

A widely deployed caching server implementation transparently returns the broken response (as well as caches it) to the stub resolver. Another known server implementation parses the response by themselves, and sends a separate response with the RCODE being 2 (SERVFAIL).

In either case, the broken response does not affect queries for an A RR of the same name. If the stub resolver falls back to A queries, it will get an appropriate response.

The latter case, however, causes the same bad effect as that described in the previous section: redundant queries for AAAA RRs.

2.4.4 Make Lame Delegation

Some authoritative servers respond to AAAA queries in a way causing lame delegation. In this case the parent zone specifies that the authoritative server should have the authority of a zone, but the server does not return an authoritative response for AAAA queries within the zone (i.e., the AA bit in the response is not set). On the other hand, the authoritative server returns an authoritative response for A queries.

When a caching server asks the server for AAAA RRs in the zone, it recognizes the delegation is lame, and returns a response with the RCODE being 2 (SERVFAIL) to the stub resolver.

Furthermore, some caching servers record the authoritative server as lame for the zone and will not use it for a certain period of time. With this type of caching server, even if the stub resolver falls back to querying for an A RR, the caching server will simply return a response with the RCODE being SERVFAIL, since all the servers are known to be "lame."

There is also an implementation that relaxes the behavior a little bit. It basically tries to avoid using the lame server, but still continues to try it as a last resort. With this type of caching server, the stub resolver will get a correct response if it falls back after SERVFAIL. However, this still causes redundant AAAA queries as explained in the previous sections.

2.4.5 Ignore Queries for AAAA

Some authoritative severs seem to ignore queries for a AAAA RR, causing a delay at the stub resolver to fall back to a query for an A RR. This behavior may even cause a fatal timeout at the resolver.

2.5 Security Considerations

The CERT/CC pointed out that the response with NXDOMAIN described in Section 2.4.1 can be used for a denial of service attack[286]. The same argument applies to the case of "lame delegation" described in Section 2.4.4 with a certain type of caching server.

2.6 Acknowledgements

Erik Nordmark encouraged the authors to publish this document as an Internet Draft. Akira Kato and Paul Vixie reviewed a preliminary version of this document. Pekka Savola carefully reviewed a previous version and provided detailed comments.

Appendix Live Examples

In this appendix, we show concrete implementations and domain names that may cause problematic cases so that the behavior can be reproduced in a practical environment. The examples are for informational purposes only, and the authors do not intend to accuse any implementations or zone administrators.

The behavior described in Section 2.4.2 (return NOTIMP) can be found by looking for a AAAA RR of www.css.vtext.com at 66.174.3.4.

The behavior described in Section 2.4.3 (broken responses) can be seen by querying for a AAAA RR of "www.gslb.mainichi.co.jp," which is an alias of "www.mainichi.co.jp," at 210.173.172.2. The same behavior can be found with the name "vip.alt.ihp.sony.co.jp," an alias of "www.sony.co.jp," at 210.139.255.204.

The behavior described in Section 2.4.4 (lame delegation) can be found by querying for a AAAA RR of "www.ual.com" at 209.87.113.4.

The behavior described in Section 2.4.5 (ignore queries) can be seen by trying to ask for a AAAA RR of "ad.3jp.doubleclick.net," which is an alias of "ad.jp.doubleclick.net," at 210.153.90.9.

Many authoritative server implementations show the expected behavior described in Section 2.3. Some DNS load balancers reportedly have a problematic behavior shown in Section 2.4, but the authors do not have a concrete example. The CERT/CC provides a list of implementations that behave as described in Section 2.4.1[286].

The BIND9 caching server implementation is an example of the latter cases described in Section 2.4.3 and Section 2.4.4, respectively. The BIND8 caching server implementation is an example of the former case described in Section 2.4.3. As for the issue shown in Section 2.4.4, BIND8 caching servers prior to 8.3.5 show the behavior described as the former case in that section. The versions 8.3.5 and later of BIND8 caching server behave like the BIND9 caching server implementation with this matter.

Regarding resolver implementations, the authors are only familiar with the ones derived from the BIND implementation. These implementations always fall back regardless of the RCODE; NXDOMAIN, NOTIMP, or SERVFAIL. It even falls back when getting a broken response. However, the behavior does not help the situation in the NXDOMAIN case (see Section 2.4.1). Lame delegation (Section 2.4.4) also causes a fatal error at the resolver side if the resolver is using some older versions of BIND8 caching server.

The authors hear that a stub resolver routine implemented in some web browsers interprets the broken response described in Section 2.4.3 as a fatal error and does not fall back to A queries. However, we have not confirmed this information.

第 3 章　Operational Guidelines for "local" zones in the DNS

(This document was originally submitted to the IETF as an Internet Draft.)

Abstract

A large number of DNS queries regarding to the "local" zones are sent over the Internet in every second. This memo describes operational guidelines to reduce the unnecessary DNS traffic as well as the load of the Root DNS Servers.

3.1 Introduction

While it has yet been described in a RFC, .local is used to provide a local subspace of the DNS tree. Formal delegation process has not been completed for this TLD. In spite of this informal status, .local has been used in many installations regardless of the awareness of the users. Usually, the local DNS servers are not authoritative to the .local domain, they end up to send queries to the Root DNS Servers.

There are several other DNS zones which describe the "local" information. .localhost has been used to describe the localhost for more than a couple of decades and virtually all of the DNS servers are configured authoritative for .localhost and its reverse zone .127.in-addr.arpa. However, there are other "local" zones currently used in the Internet or Intranets connected to the Internet through NATs or similar devices.

At a DNS server of an university in Japan, half of the DNS queries sent to one of the 13 Root DNS Servers were regarding to the .local. At another DNS Server running in one of the Major ISPs in Japan, the 1/4 were .local. If those "local" queries are able to direct other DNS servers than Root, or they can be resolved locally, it contributes the reduction of the Root DNS Servers.

3.2 Rationale

Any DNS queries regarding to "local" names should not be sent to the DNS servers on the Internet.

3.3 Operational Guidelines

Those queries should be processed at the DNS servers internal to each site so that the servers respond with NXDOMAIN rather than sending queries to the DNS servers outside.

The "local" names have common DNS suffixes which are listed below:

3.3.1 Local host related zones:

Following two zones are described in [10] and .localhost is also defined in [65].

- .localhost
- .127.in-addr.arpa

Following two zones are for the loopback address in IPv6[100]. While the TLD for IPv6 reverse lookup is .arpa as defined in [19], the old TLD .int has been used for this purpose for years[296] and many implementations still use .int. So it is suggested that both zones should be provided for each IPv6 reverse lookup zone for a while.

- 1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.int
- 1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa

3.3.2 Locally created name space

While the use of .local has been proposed in several Internet-Drafts, it has not been described in any Internet documents with formal status. However, the amount of the queries for .local is much larger than others, it is suggested to resolve the following zone locally:

- .local

3.3.3 Private or site-local addresses

The following IPv4 "private" addresses[148] and IPv6 site-local addresses[100] should be resolved locally:

- 10.in-addr.arpa
- 16.172.in-addr.arpa
- 17.172.in-addr.arpa
- 18.172.in-addr.arpa
- 19.172.in-addr.arpa
- 20.172.in-addr.arpa
- 21.172.in-addr.arpa
- 22.172.in-addr.arpa
- 23.172.in-addr.arpa
- 24.172.in-addr.arpa
- 25.172.in-addr.arpa
- 26.172.in-addr.arpa
- 27.172.in-addr.arpa
- 28.172.in-addr.arpa
- 29.172.in-addr.arpa
- 30.172.in-addr.arpa
- 31.172.in-addr.arpa
- 168.192.in-addr.arpa
- c.e.f.ip6.int
- d.e.f.ip6.int
- e.e.f.ip6.int
- f.e.f.ip6.int
- c.e.f.ip6.arpa
- d.e.f.ip6.arpa
- e.e.f.ip6.arpa
- f.e.f.ip6.arpa

3.3.4 Link-local addresses

The link-local address blocks for IPv4[106] and IPv6[100] should be resolved locally:

- 254.169.in-addr.arpa
- 8.e.f.ip6.int
- 9.e.f.ip6.int
- a.e.f.ip6.int
- b.e.f.ip6.int
- 8.e.f.ip6.arpa
- 9.e.f.ip6.arpa
- a.e.f.ip6.arpa
- b.e.f.ip6.arpa

3.4 Suggestions to developers

3.4.1 Suggestions to DNS software implementors

In order to avoid unnecessary traffic, it is suggested that DNS software implementors provide configuration templates or default configurations so that the names described in the previous section are resolved locally rather than sent to other DNS servers in the Internet.

3.4.2 Suggestions to developers of NATs or similar devices

There are many NAT or similar devices available in the market. Regardless of the availability of DNS Servers in those devices, it is suggested that those devices are able to filter the DNS traffic or respond to the DNS traffic related to "local" zones by configuration regardless of its ability of DNS service. It is suggested that this functionality is activated by default.

3.5 IANA Consideration

While .local TLD has yet defined officially, there are substantial queries to the Root DNS Servers as of writing. About 1/4 to 1/2 of the traffic sent to the Root DNS Servers are related to the .local zone. Therefore, while it is not formally defined, it is suggested that IANA delegates .local TLD to an organization.

The AS112 Project[305] serves authoritative DNS service for RFC1918 address and the link-local address. It has several DNS server instances around the world by using BGP Anycast[97]. So the AS112 Project is one of the candidates to host the .local TLD.

第 4 章　Address Transition of JP DNS server

Since the epoch of Japanese Internet in 1989, WIDE Project has been operating one of the JP DNS servers. Initially, the DNS server named JP-GATE.WIDE.AD.JP had been operated in Yokohama. In 1990, it moved to Fujisawa, Kanagawa. It continued to work as a secondary DNS server after JPNIC took over the primary DNS server in 1993. In November 1996, we renamed it NS.WIDE.AD.JP. These servers had served JP zones in conjunction with other zones for many years. In August 2001, IPv6 transport for DNS query/response was started by a separated server.

In March 2002, we performed a number of improvements on the server:

- Moved to Osaka,
- Became a dedicated DNS server which served JP and related zones,
- authoritative only,
- Configured as a primary server and a backup server to improve availability,
- Started the unified service where both IPv4 queries and IPv6 queries were supported in a single server.

In June 2003, in order to provide extra redundancy to survive DNS service in the case of serious disasters in Tokyo, e.g. power outage, we made the following changes:

- The server was detached from the WIDE Internet and reconnected to NSPIXP-3 and JPNAP-Osaka through a dedicated router,
- AS23634 was used to represent the system,
- The server was renamed E.DNS.JP,
- IPv4 address 203.178.136.63 was migrated to 192.50.43.53 while IPv6 address 2001:200:0:1::4 remained unchanged at that moment.

Here we show the IP address migration details and the transition of query to the old and the new addresses.

We planned to migrate forward lookup zone first, i.e. JP and the zones under JP which JPNIC holds. After completion of the work for forward zones, we started working on reverse lookup zones, which are delegated into JPNIC. Each migration consists of 3 phases to avoid lame delegation:

1. Add the new NS record into authoritative zones.

2. Replace the glue record of NS on the root zone (or the upper zone for reverse lookup).
3. Delete the old NS record from authoritative zones.

Figure 4.1 shows the number of queries per second measured during transition period.

2003/6/28 05:00 New A record for E.DNS.JP which pointed to 192.50.43.53 was registered into DNS.JP zone, but there was no NS records which referred it.

2003/7/1 13:47 E.DNS.JP was added as a NS record to the JP zones (JP and the number of domains just under JP). NS.WIDE.AD.JP was still included as NS for JP to avoid lame delegation, since the glue records in the root zone was not changed at the moment. Thus, there were 7 NS records for authoritative JP, and the glue records in root zone still had old 6 NS records for JP.

2003/7/2 Sent an application to replace a secondary for JP: NS.WIDE.AD.JP to E.DNS.JP.

2003/7/10 07:30 The glue NS records for JP in root servers were updated; NS.WIDE.AD.JP was replaced by E.DNS.JP. Again, NS.WIDE.AD.JP still existed in the authoritative JP.

2003/7/17 05:00 NS.WIDE.AD.JP was removed from the authoritative JP and most of the sub-zones of JP. The database at JPRS was designed to trigger DNS update only by modifying actual domain information. Some of the sub-zones of JP still held NS.WIDE.AD.JP.

2003/7/22 05:00 Finally, NS.WIDE.AD.JP was deleted from all of the sub-zones under JP. At the point, NS.WIDE.AD.JP was still included in numbers of the delegated reverse lookup zones.

2003/7/23 05:00 E.DNS.JP was added as well as A.DNS.JP and B.DNS.JP into the reverse lookup zones, which had 9 NS records as a result. The applications for adding these 3 NS records into reverse zones were sent to ARIN and APNIC. But APNIC didn't accept A.DNS.JP and B.DNS.JP because their IP addresses were duplicated with those for some of existing NS records.

2003/7/25 05:00 NS.WIDE.AD.JP and other 2 old NS records were removed from all of the authoritative reverse zones. Thus, all of the zones

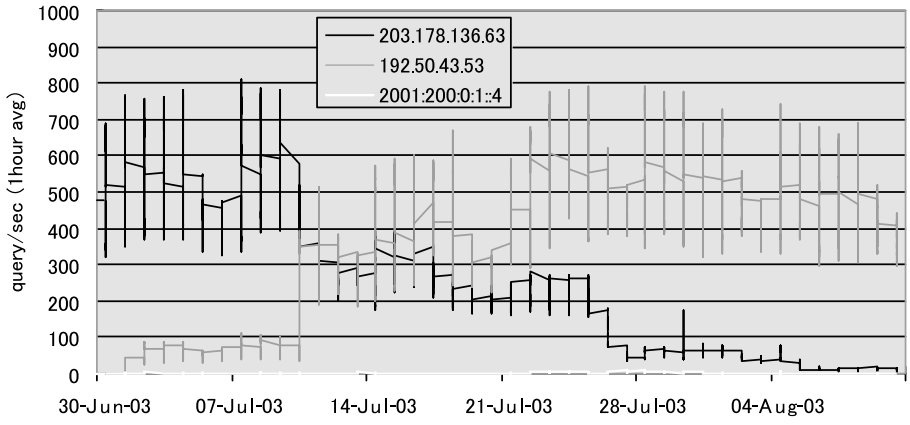

Fig. 4.1. Transition of queries during DNS changes

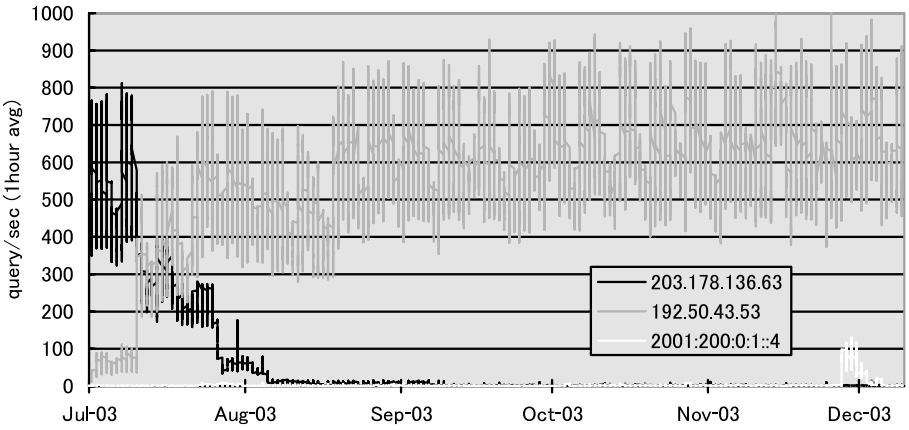

Fig. 4.2. Long term transition of queries

had 6 unique NS records in total; 3 old NS and 3 new NS. We resent the applications for replacing NS records instead of adding/removing. APNIC accepted them without any problems and updated the DNS records soon.

At the moment, we expected all the references to NS.WIDE.AD.JP were already removed and much less queries to be measured to the old address. But more than 50 queries per second were still reached to the old address, NS.WIDE.AD.JP. And we were afraid that we have to keep the old address forever.

After careful lookup of tcpdump results, we noticed some of the bogus reverse lookup zones had NS records which pointed to NS.WIDE.AD.JP. The zones in question were:

1.150.in-addr.arpa　2.150.in-addr.arpa.
4.150.in-addr.arpa.　5.150.in-addr.arpa.
6.150.in-addr.arpa.　8.150.in-addr.arpa.
(omitted)
94.150.in-addr.arpa.　95.150.in-addr.arpa.
98.150.in-addr.arpa.　131.163.in-addr.arpa.
132.163.in-addr.arpa.　133.163.in-addr.arpa.
141.163.in-addr.arpa.　144.163.in-addr.arpa.
146.163.in-addr.arpa.　147.163.in-addr.arpa.

JPNIC said that these zones were listed at ERX projects[1] and had a set of bogus NS records inserted at the transition. JPNIC had requested APNIC to remove these bogus record, and they were cleaned up before 2003/8/4 17:00. After that, we have only seen 10 queries per second to the old address.

Now most of the queries to the old address are bogus, requesting recursive query which will be refused anyway. We asked some of the hostmaster and heard that NS.WIDE.AD.JP was listed in resolve.conf on the host for many years. Three months later, there are only about 1–2 queries per second to the old address, NS.WIDE.AD.JP.

After the experience in transition of the DNS addresses, we realized that an updated document is important, which should describe how to configure DNS, including the resolv.conf file and DHCP configuration. It helps not only for the address transition, but reducing bogus query in normal operation.

Now we have isolated E.DNS.JP from the WIDE Internet and assigned it a new AS and an IP address. We are planning to distribute E.DNS.JP into several places with anycasting technology, which enables stable and less-RTT services for DNS to the Internet.

1　http://www.apnic.net/db/erx/index.html
The records known as "early registration" which is a result of transition from InterNIC to RIRs.