Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: May 18, 2017

L. Song
S. Kerr
D. Liu
Beijing Internet Institute
P. Vixie
TISF
Kato
Keio University/WIDE Project

November 14, 2016

Experiences from Root Testbed in the Yeti DNS Project
draft-song-yeti-testbed-experience-03

Abstract

This document reports and discusses issues in DNS root services, based on experiences from the experiments in the Yeti DNS Project. These issues include IPv6-only operation, the root DNS server naming scheme, DNSSEC KSK rollover, root server renumbering, multiple root zone signer, and so on. This project was founded in May 2015 and has since built a live root DNS server system testbed with volunteer root server and resolver operations.

REMOVE BEFORE PUBLICATION: Although this document is submitted as an independent submission, comments are welcome in the IETF DNSOP (DNS Operations) working group mailing list. The source of the document is currently placed at GitHub [xml-file].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 18, 2017.

Copyright Notice

Table of Contents

1. Introduction

[RFC1034] says the domain name space is a tree structure. The top level of the tree for the unique identifier system is served by the DNS root system. It has been operational for 25+ years. It is pivotal to making the current Internet useful. So it is considered somewhat ossified for stability reasons. It is hard to test and implement new ideas evolving to a more advanced level to counter challenges like IPv6-only operation, DNSSEC key/algorithm rollover[RFC4986], scaling issues, and so on. In order to make the test more practical, it is also necessary to involve users' environments which are highly diversified, in order to study the effects of the changes in question.

To benefit Internet development as a whole, the Yeti DNS Project was proposed to build a parallel, experimental, live IPv6 DNS root system to discover the limits of DNS root name service and deliver useful technical output. Possible research agenda will be explored on this testbed, covering several aspects (but not limited to):

- IPv6-only operation
- DNSSEC key rollover
- Renumbering issues
- Scalability issues
- Multiple zone file signers

Starting from May 2015, three coordinators began to build this live experimental environment and called for participants. At the time of writing, there are 25 Yeti root servers with 16 operators, and experimental traffic from volunteers, universities, DNS vendors, mirrored traffic, non-Yeti traffic, and RIPE Atlas probes.

Note that the Yeti DNS Project has complete fealty to IANA as the DNS name space manager. All IANA top-level domain names will be precisely expressed in the Yeti DNS system, including all TLD data and meta-data[Root-Zone-Database]. Therefore, the Yeti DNS Project is never an "alternative root" in the usual sense of that term. It is expected to inform the IANA community by peer-reviewed science as to future possibilities to consider for the IANA root DNS system.

In order to let people know the technical activities in Yeti DNS Project, this document reports and discusses issues on root DNS services, based on experiences so far from the testbed construction and experiments in the Yeti DNS Project.

2. Problem Statement

Some problems and policy concerns over the DNS Root Server system stem from centralization from the point of view of DNS content consumers. These include external dependencies and surveillance threats.

- External Dependency: Currently, there are 12 DNS Root Server operators for the 13 Root Server letters, with more than 500 instances deployed globally. Yet compared to the number of connected devices, AS networks, and recursive DNS servers, the number of root instances may not be sufficient. Connectivity loss between one autonomous network and all of the IANA root name servers usually results in loss of not only global service but also local service within the local network, even when internal connectivity is perfect.

- Surveillance risk: Even when one or more root name server anycast instances are deployed locally or in a nearby network, the queries sent to the root servers carry DNS lookup information which enables root operators or other parties to analyze the DNS query traffic. This is a kind of information leakage[RFC7626] which is to some extent not acceptable to some policy makers.

People are often told that the current root system with 13 root servers is not able to be extended to alleviate the above concerns, because it is limited to 13 by the current DNS protocol[ROOT-FAQ]. This restriction may be relaxed when EDNS is considered completely deployed and/or when the root system doesn't have to support IPv4 anymore.

There are some technical issues in the areas of IPv6 and DNSSEC, which were introduced to the DNS root server system after it was created. Renumbering DNS root servers also creates some technical issues.

- IPv6-only capability: Currently some DNS servers which support both A and AAAA (IPv4 and IPv6) records still do not respond to IPv6 queries. IPv6 introduces larger minimum MTU (1280 bytes) and a different fragmentation model[Fragmenting-IPv6]. It is not clear whether DNS can survive without IPv4 (in an IPv6-only environment), or what impact in IPv6-only environment introduces to current DNS operations especially in the DNS root server system.

- KSK rollover: Currently, IANA rolls the ZSK every six weeks but the KSK has never been rolled as of the writing. Is RFC5011 [RFC5011] widely supported by resolvers? How about larger key

size or different encryption algorithm? Is the DNS packet size limitation (512 or 1280 bytes) should be respected during KSK rollover nowadays? There are some issues still unknown.

- Renumbering issue: It is likely that root operators may change their IP addresses for root servers as well. Currently resolver can use priming exchange [I-D.ietf-dnsop-resolver-priming] to update its memory in real time. Or it may combine out-band way to periodically get the current list of NS server of Root. However it is observed root renumbering is still a concern which need coordination and interference from human labor which deserves exploring for automation.

3. Yeti Testbed and Experiment Setup

To use the Yeti testbed operationally, the information that is required for correct root name service is a matching set of the following:

- a root "hints file"
- the root zone apex NS record set
- the root zone's signing key
- root zone trust anchor

Although Yeti DNS Project publishes strictly IANA information for TLD data and meta-data, it is necessary to use a special hint file to replace the apex NS RRset with Yeti authority name servers, which will enable the resolves to find and stick to the Yeti root system.

Below is a figure to demonstrate the topology of Yeti and the basic data flow, which consists of the Yeti distribution master, Yeti root server, and Yeti resolver:

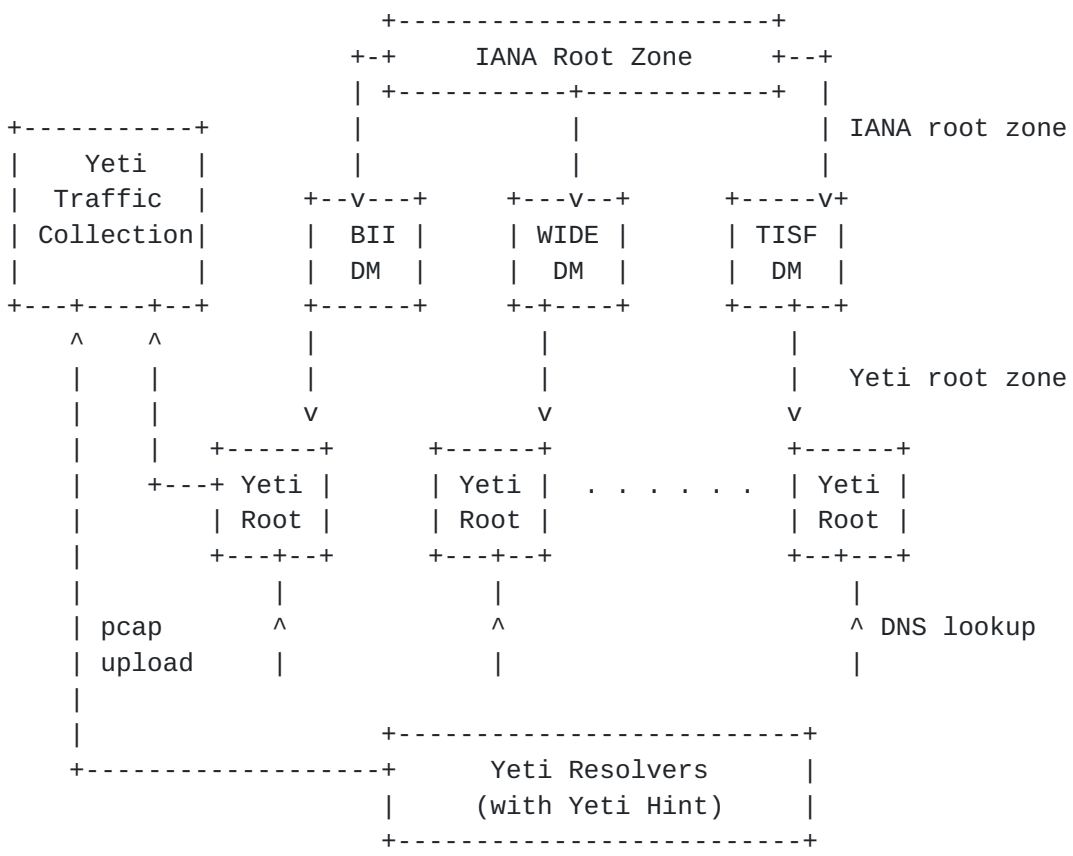

Figure 1. The topology of Yeti testbed

3.1. Distribution Master

As shown in figure 1, the Yeti Root system takes the IANA root zone and performs minimal changes needed to serve the zone from the Yeti root servers instead of the IANA root servers. In Yeti, this modified root zone is generated by the Yeti Distribution Masters (DM), which provide it to the Yeti root servers.

The zone generation process is:

- DM downloads the latest IANA root zone at a certain time
- DM makes modifications to change from the IANA to Yeti root servers
- DM signs the new Yeti root zone with Yeti key
- DM publishes the new Yeti root zone to Yeti root servers

While in principle this could be done by a single DM, Yeti uses a set of three DMs to avoid any sense that the Yeti DNS Project is run by a single organization. Each of three DMs independently fetches the root zone from IANA, signs it and publishes the latest zone data to Yeti root servers.

In the same while, these DMs coordinate their work so that the resulting Yeti root zone is always consistent. There are two aspects of coordination between three DMs: timing and information synchronization.

3.1.1. Yeti root zone SOA SERIAL

Consistency with IANA root zone except the apex record is one of most important point for the project. As part of Yeti DM design, the Yeti SOA SERIAL which reflects the changes of yeti root zone is one factor to be considered.

Currently IANA SOA SERIAL number for root zone is in the form of YYYYMMDDNN, like 2015111801. In Yeti root system, IANA SOA SERIAL is directly copied in to Yeti SOA SERIAL. So once the IANA root zone has changed with a new SOA SERIAL, a new version of the Yeti root zone is generated with the same SOA SERIAL.

There is a case of Yeti DM operation that when a new Yeti root server added, DM operators change the Yeti root zone without change the SOA SERIAL which introduces inconsistency of Yeti root system. To avoid inconsistency, the DMs publish changes only when new IANA SOA SERIAL is observed.

An analysis of IANA convention shows IANA SOA SERIAL change twice a day (NN=00, 01). Since October 2007 the maximum of NN was 03 while NN=2 was observed in 13 times.

3.1.2. Timing of Root Zone Fetch

Yeti root system operators do not receive notify messages when IANA root zone is updated. So each Yeti DM checks the root zone serial periodically. At the time of writing, each Yeti DM checks to see if the IANA root zone has changed hourly, on the following schedule:

DM Operator	Time
BII	hour+00
WIDE	hour+20
TISF	hour+40

Note that Yeti DMs can check IANA root zone more frequently (every minute for example). A test done by Yeti participant shows that the delay of IANA root zone update from the first IANA root server to last one is around 20 minute. Once a Yeti DM fetch the new root zone, it will notify all the Yeti root servers with a new SOA serial number. So normally Yeti root server will be notified in less than 20 minute after new IANA root zone generated. Ideally, if an IANA DM notifies the Yeti DMs, Yeti root zone will be updated in more timely manner.

3.1.3. Information Synchronization

Given three DMs operational in Yeti root system, it is necessary to prevent any inconsistency caused by human mistakes in operation. The straight method is to share the same parameters to produce the Yeti root zone. There parameters includes following set of files:

- the list of Yeti root servers, including:
  - public IPv6 address and host name
  - IPv6 addresses originating zone transfer
  - IPv6 addresses to send DNS notify to
- the ZSKs used to sign the root
- the KSK used to sign the root
- the SERIAL when this information is active

The operation is simple that each DM operator synchronize the files with the information needed to produce the Yeti root zone. When a change is desired (such as adding a new server or rolling the ZSK), a DM operator updates the local file and push to other DM. A SOA SERIAL in the future is chosen for when the changes become active.

3.2. Yeti Root Servers

In Yeti root system, authoritative servers donated and operated by Yeti volunteers are configured as a slave to the Yeti DM. As the time of writing, there are 25 Yeti root servers distributed around the world, one of which use IDN as its name (see Yeti hint file in Appendix A). As one of operational research goal, all authoritative servers are required to work in an IPv6-only environment. In addition, different from the IANA root, Yeti root server only serve the Yeti root zone. No root-servers.org zone and .arpa zone are served.

Since Yeti is a scientific research project, it needs to capture DNS traffic sent to one of the Yeti root servers for later analysis. Today some servers use dnscap, which is a DNS-specific tool to produce pcap files. There are several versions of dnscap floating around; some people use the VeriSign one. Since dnscap loses packets in some cases (tested on a Linux kernel), some people use pcapdump. It requires the patch attached to this bug report [pcapdump-bug-report]

System diversity is also a requirement and observed for current 25 Yeti root server. Here are the results of a survey regarding the machine, operation system and DNS software:

- Machine: 20 out of 25 root server operator are using a VPS to provide service.

- OS: 6 operators use Linux (including Ubuntu, Debian, CentOS, ArchLinux). 5 operators use FreeBSD and 1 NetBSD. And other servers are unknown.

- DNS software: 18 our of 25 root server use BIND (varying from 9.9.7 to 9.10.3). 4 of them use NSD (4.10 and 4.15). The other 2 servers use Knot (2.0.1 and 2.1.0). And one use Bundy (1.2.0)

3.3. Yeti Resolvers and Experimental Traffic

In client side of Yeti DNS Project, there are DNS resolvers with IPv6 support, updated with Yeti "hints" file to use the Yeti root servers instead of the IANA root servers, and using Yeti KSK as trust anchor. The Yeti KSK rollover experiment is expected to change key often (typically every three months), it is required that resolver operator to configure the resolver compliant to RFC 5011 for automatic update. For Yeti resolver, it is also interesting to try some mechanism end-system resolvers to signal to a server about their DNSSEC key status, like [I-D.wessels-edns-key-tag] and [I-D.wkumari-dnsop-trust-management] mentioned.

Participants and volunteers are expected from individual researchers, labs of universities, companies and institutes, and vendors (for example, the DNS software implementers), developers of CPE devices & IoT devices, and middle box developers who can test their products and connect their own testbed into Yeti testbed. Resolvers donated by Yeti volunteers are required to be configured with Yeti hint file and Yeti DNSSEC KSK. It is required that Yeti resolver can speak both IPv4 and IPv6, given that not all authoritative servers on the Internet are IPv6 capable.

At the time of writing several universities and labs have joined us and contributed certain amount of traffic to Yeti testbed. To introduce desired volume of experiment traffic, Yeti Project adopts two alternative ways to increase the experimental traffic in the Yeti testbed and check the functionality of Yeti root system.

One approach is to mirror the real DNS query to IANA root system by off-path method and replay it into Yeti testbed; this is implemented by some Yeti root server operators. Another approach is to use traffic generating tool such as RIPE Atlas probes to generate specific queries against Yeti servers.

4. Experiments in Yeti Testbed

The main goal of Yeti DNS Project is to act as an experimental network. Experiments will be conducted on this network. In order to make the findings that result from these experiments more rigorous, an experiment protocol is proposed.

A Yeti experiment goes through four phases:

- Proposal. The first step is to make a proposal. It is discussed and if accepted by the Yeti participants then it can proceed to the next phase.

- Lab Test. The next phase is to run a version of the experiment in a controlled environment. The goal is to check for problems such as software crashes or protocol errors that may cause failures on the Yeti network, before putting onto the experimental network.

- Yeti Test. The next phase actually running the experiment on the Yeti network. Details of this will depend on the experiment. It must be coordinated with the Yeti participants.

- Report of Findings. When completed, a report of the findings of the experiment should be made. It need not be an extensive document.

In this section, we are going to introduce some experiments implemented and planned in the Yeti DNS Project.

4.1. Root Naming Scheme

In root server history, the naming scheme for individual root servers was not fixed. Current IANA Root server adopt [a-m].root-servers.net naming scheme to represent 13 servers which are labeled with letter from A to M. The authoritativeness is achieved by hosting "root-servers.net" zone in every root server. One reason behind this

naming scheme is that DNS label compression can be used to produce a smaller DNS response within 512 bytes. But in Yeti testbed there is a chance to design and test alternative naming schemes to solve some issues with current naming scheme.

- Currently root-servers.net is not signed. Kaminsky-like attacks are still possible for the the important information of Root server.

- The dependency to a single name(i.e..net) make the root system fragile in extreme case that all .net servers are down or unreachable but the root server still alive.

Currently, there are two naming schemes proposed in Yeti Project. One is to use separate and normal domains for root servers (Appendix A). One consideration is to get rid of the dependency on the single name. Another consideration is to intentionally produces larger packets for priming responses for less name compression efficiency. Note that the Yeti root has a priming response which is 1031 Bytes as of the writing.

There is also a issue for this naming scheme in which the priming response may not contain all glue record for Yeti Root servers. It is documented as a technical findings [Yeti-glue-issue]. There are two approaches to solve the issue: one is to patch BIND 9 to includes the glue records in the additional section. The other one is to add a zone file for each root server and answer for all of them at each Yeti server. That means each Yeti root server would have a small zone file for "bii.dns-lab.net", "yeti-ns.wide.ad.jp", "yeti-ns.tisf.net", and so on.

Another naming scheme under Yeti lab test is to use a special non-delegated TLD, like .yeti-dns for root server operated by BII. The benefit of non-delegated TLD naming scheme are in two aspects: 1) the response to a priming query is protected by DNSSEC; 2) To meet a political reason that the zone authoritative for root server is not delegated and belong to particular companies or organizations except IANA; 3) reduce the dependency of root server names to other DNS service; 4) to mitigate some kind of cache poisoning activities.

The obvious concern of this naming scheme is the size of the signed response with RRSIG for each root server and optionally DNSKEY RRs. There is a Lab test result regarding the different size of priming response in Octet : 1) with no additional data, with RRISG in additional section , with DNSKEY+RRSIG in additional section (7 keys in MZSK experiment. MZSK is to be described in section 4.2)

No additional data	RRSIG	RRISG +DNSKEY
753	3296	4004

We found that modification of IANA root zone by adding a new TLD is so controversial even for scientific purpose. There are non-trivial discussions on this issue in Yeti discuss mailing list, regarding the proposal .yeti-dns for root name or .local for new AS112 [I-D.bortzmeyer-dname-root]. It is argued that this kind of experiment should based on community consensus from technical bodies like IETF and be operated within a limited duration in some cases.

Note that a document named "Technical Analysis of the Naming Scheme used for Individual Root Servers" is being developed in RSSAC Caucus. And it will be published soon

4.2. Multiple-Signers with Multi-ZSK

According to the Problem statement of Yeti DNS Project, more independent participants and operators of the root system is desirable. As the name implies, multi-ZSK (MZSK) mode introduces different ZSKs sharing a single unique KSK, as opposed to the IANA root system (which uses a single ZSK to sign the root zone). On the condition of good availability and consistency on the root system, the Multi-ZSK proposal is designed to give each DM operator enough room to manage their own ZSK, by choosing different ZSK, length, duration, and so on; even the encryption algorithm may vary (although this may cause some problem with older versions of the Unbound resolver).

4.2.1. MZSK lab experiment

In the lab test phase, we simply setup two root servers (A and B) and a resolver switch between them (BIND only). Root A and Root B use their own ZSK to sign the zone. It is proved that Multi-ZSK works by adding multiple ZSK to the root zone. As a result, the resolver will cache the key sets instead of a single ZSK to validate the data no matter it is signed by Root A or Root B. We also tested Unbound and the test concluded in success with more than 10 DMs and 10 ZSKs.

Although more DMs and ZSKs can be added into the test, adding more ZSKs to the root zone enlarges the DNS response size for DNSKEY queries which may be a concern given the limitation of DNS packet size. Current IANA root server operators are inclined to keep the packets size as small as possible. So the number of DM and ZSK will

be parameter which is decided based on operation experience. In the current Yeti root testbed, there are 3 DMs, each with a separate ZSK.

4.2.2. MZSK Yeti experiment

After the lab test, the MZSK experiment is being conducted on the Yeti platform. There are two phases:

- Phase 1. In the first phase, we confirmed that using multiple ZSKs works in the wild. We insured that using the maximum number of ZSKs continues to work in the resolver side. Here one of the DM (BII) created and added 5 ZSKs using the existing synchronization mechanism. (If all 3 ZSKs are rolling then we have 6 total. To get this number we add 5.)

- Phase 2. In the second phase, we delegated the management of the ZSKs so that each DM creates and publishes a separate ZSK. For this phase, modified zone generation protocol and software was used [Yeti-DM-Sync-MZSK], which allows the DM to sign without access to the private parts of ZSKs generated by other DMs. In this phase we roll all three ZSKs separately.

The MZSK experiment was finished by the end of 2016-04. Almost everything appears to be working. But there have been some findings [Experiment-MZSK-notes], including discovering that IPv6 fragmented packets are not forwarded on an Ethernet bridge with netfilter ip6_tables loaded on one authority server, and issue with IXFR falling back to AXFR due to multiple signers which is described in [I-D.song-dnsop-ixfr-fallback] as a problem statement.

4.3. Root Renumbering Issue and Hint File Update

With the recent renumbering of H root Server's IP address, there is a discussion of ways that resolvers can update their hint file. Traditional ways include using FTP protocol by doing a wget and using dig to double-check the servers' addresses manually. Each way would depend on manual operation. As a result, there are many old machines that have not updated their hint files. As a proof, after completion of renumbering in thirteen years ago, there is an observation that the "Old J-Root" can still receive DNS query traffic [Renumbering-J-Root].

This experiment proposal aims to find an automatic way for hint-file updating. The already-completed work is a shell script tool which provides the function that updates a hint-file in file system automatically with DNSSEC and trust anchor validation. [Hintfile-Auto-Update]

The methodology is straightforward. The tool first queries the NS list for "." domain and queries A and AAAA records for every name on the NS list. It requires DNSSEC validation for both the NS list and the A and AAAA answers. After getting all the answers, the tool compares the new hint file with the old one. If there is a difference, it renames the old one with a time-stamp and replaces the old one with the new one. Otherwise the tool deletes the new hint file and nothing will be changed.

Note that in current IANA root system the servers named in the root NS record are not signed. So the tool can not fully work in the production network. In Yeti root system some of the names listed in the NS record are signed, which provides a test environment for such a proposal.

4.4. DNS Fragments

In consideration of new DNS protocol and operation, there is always a hard limit on the DNS packet size. Take Yeti for example: adding more root servers, using the Yeti naming scheme, rolling the KSK, and Multi-ZSK all increase the packet size. The fear of large DNS packets mainly stem from two aspects: one is IP-fragments and the other is frequently falling back to TCP.

Fragmentation may cause serious issues; if one of the fragment is lost at random, it results in the loss of entire packet and involve timeout. If the fragment is dropped by a middle-box, the query always results in failure, and result in name resolution failure unless the resolver falls back to TCP. It is known at this moment that limited number of security middle-box implementations support IPv6 fragments.

A possible solution is to split a single DNS message across multiple UDP datagrams. This DNS fragments mechanism is documented in [I-D.muks-dns-message-fragments] as an experimental IETF draft.

4.5. The KSK Rollover Experiment in Yeti

The Yeti DNS Project provides a good basis to conduct a real-world experiment of a KSK rollover in the root zone. It is not a perfect analogy to the IANA root because all of the resolvers to the Yeti experiment are "opt-in", and are presumably run by administrators who are interested in the DNS and knowledgeable about it. Still, it can inform the IANA root KSK roll.

The IANA root KSK has not been rolled as of the writing. ICANN put together a design team to analyze the problem and make recommendations. The design team put together a

plan[ICANN-ROOT-ROLL]. The Yeti DNS Project may evaluate this scenario for an experimental KSK roll. The experiment may not be identical, since the time-lines laid out in the current IANA plan are very long, and the Yeti DNS Project would like to conduct the experiment in a shorter time, which may considered much difficult.

The Yeti KSK is rolled twice in Yeti testbed as of the writing. In the first trial, it made old KSK inactive and new key active in one week after new key created, and deleted the old key in another week, which was totally unaware the timer specified in RFC5011. Because the hold-down timer was not correctly set in the server side, some clients (like Unbound) receive SERVFAILs (like dig without +cd) because the new key was still in AddPend state when old key was inactive. The lesson from the first KSK trial is that both server and client should compliant to RFC5011.

For the second KSK rollover, it waited 30 days after a new KSK is published in the root zone. Different from ICANN rollover plan, it revokes the old key once the new key become active. We don't want to wait too long, so we shorten the time for key publish and delete in server side. As of the writing, only one bug [KROLL-ISSUE]spotted on one Yeti resolver (using BIND 9.10.4-p2) during the second Yeti KSK rollover. The resolver is configured with multiple views before the KSK rollover. DNSSEC failures are reported once we added new view for new users after rolling the key. By checking the manual of BIND9.10.4-P2, it is said that unlike trusted-keys, managed-keys may only be set at the top level of named.conf, not within a view. It gives an assumption that for each view, managed-key can not be set per view in BIND. But right after setting the managed-keys of new views, the DNSSEC validation works for this view. As a conclusion for this issue, we suggest currently BIND multiple-view operation needs extra guidance for RFC5011. The manage-keys should be set carefully during the KSK rollover for each view when the it is created.

Another of the questions of KSK rollover is how can an authority server know the resolver is ready for RFC5011. Two Internet-Drafts [I-D.wessels-edns-key-tag] and [I-D.wkumari-dnsop-trust-management] try to address the problem. In addition a compliant resolver implementation may fail without any complain if it is not correctly configured. In the case of Unbound 1.5.8, the key is only readable for DNS users [auto-trust-anchor-file].

4.6. Bigger ZSK for Yeti

Currently IANA root system uses 1024-bits ZSK which is no longer recommended cryptography. VeriSign announced at DNS-OARC 24th workshop that the IANA root zone ZSK will be increased from 1024 bits

to 2048 bits in 2016. However, it is not fully tested by the real environment.

Bigger key tend to produce a larger response which requires IP fragmentation and is commonly considered harm for DNS system. In Yeti DNS Project, it is desirable to test bigger responses in many aspects. The Big ZSK experiment is designed to test operating the Yeti root with a 2048-bit ZSK. The traffic is monitored before and after we lengthen the ZSK to see if there are any changes, such as a drop off of packets or a increase in retries. The current status of this experiment is under monitoring data analysis.

5. Other Technical findings and bugs

Besides the experiments with specific goals and procedures, some unexpected bugs have been reported. It is worthwhile to record them as technical findings from Yeti DNS Project.

5.1. IPv6 fragments issue

There are two cases in Yeti testbed reported that some Yeti root servers failed to pull the zone from a Distribution Master via AXFR/IXFR. Two facts have been revealed in both client side and server side after trouble shooting.

One fact in client side is that some operation system can not handle IPv6 fragments correctly and AXRF/IXFR in TCP fails. The bug covers several OSs and one VM platform (listed below).

OS	VM
NetBSD 6.1 and 7.0RC1	VMware ESXI 5.5
FreeBSD10.0	
Debian 3.2	

Another fact is from server side in which one TCP segment of AXRF/IXFR is fragmented in IP layer resulting in two fragmented packets. This weird behavior has been documented IETF draft[I-D.andrews-tcp-and-ipv6-use-minmtu]. It reports a situation that some implementations of TCP running over IPv6 neglect to check the IPV6_USE_MIN_MTU value when performing MSS negotiation and when constructing a TCP segment. It will cause TCP MSS option set to 1440 bytes, but IP layer will limit the packet less than 1280 bytes and fragment the packet to two fragmented packets.

While the latter is not a technical error, but it will cause the error in the former fact which deserves much attention in IPv6 operation .

5.2. Root name compression issue

[RFC1035]specifies DNS massage compression scheme which allows a domain name in a message to be represented as either: 1) a sequence of labels ending in a zero octet, 2) a pointer, 3) or a sequence of labels ending with a pointer. It is designed to save more room of DNS packet.

However in Yeti testbed, it is found that Knot 2.0 server compresses even the root. It means in a DNS message the name of root (a zero octet) is replaced by a pointer of 2 octets. As well, it is legal but breaks some tools (Go DNS lib in this bug report) which does not expect such name compression for root. Both Knot and Go DNS lib have fixed that bug by now.

5.3. SOA update delay issue

It is observed one server on Yeti testbed have some bugs on SOA update with more than 10 hours delay. It is running on Bundy 1.2.0 on FreeBSD 10.2-RELEASE. A workaround is to check DM's SOA status in regular base. But it still need some work to find the bug in code path to improve the software.

6. IANA Considerations

This document requires no action from the IANA.

7. Acknowledgments

The editors fully acknowledge that this memo is based on joint work and discussions of many people in the mailing list of the Yeti DNS Project [Yeti-DNS-Project]. Some of them actually are co-authors of this memo but limited by the number of co-authors listed in the headline. The people deserve the credit who help to construct the Yeti testbed and contribute to this document, so their effort is acknowledged here with a name list:

Tomohiro Ishihara, Antonio Prado, Stephane Bortzmeyer, Mickael Jouanne, Pierre Beyssac, Joao Damas, Pavel Khramtsov, Ma Yan, Otmar Lendl, Praveen Misra, Carsten Strotmann, Edwin Gomez, Remi Gacogne, Guillaume de Lafond, Yves Bovard, Hugo Salgado-Hernandez, Andreas Schulze,Li Zhen, Daobiao Gong, Runxia Wan.

Acknowledgment to all anonymous Yeti participants and volunteers who contribute Yeti resolvers to make the experimental testbed functional and workable.

8. References

[auto-trust-anchor-file]
"Unbound should test that auto-* files are writable", 2016, <https://www.nlnetlabs.nl/bugs-script/show_bug.cgi?id=758>.

[Experiment-MZSK-notes]
"MZSK Experiment Notes", 2016, <https://github.com/shane-kerr/Yeti-Project/blob/experiment-mzsk/doc/Experiment-MZSK-notes.md>.

[Fragmenting-IPv6]
Huston, G., "Fragmenting-IPv6", May 2016, <http://blog.apnic.net/2016/05/19/fragmenting-ipv6/>.

[Hintfile-Auto-Update]
"Hintfile Auto Update", 2015, <https://github.com/BII-Lab/Hintfile-Auto-Update>.

[I-D.andrews-tcp-and-ipv6-use-minmtu]
Andrews, M., "TCP Fails To Respect IPV6_USE_MIN_MTU", draft-andrews-tcp-and-ipv6-use-minmtu-04 (work in progress), October 2015.

[I-D.bortzmeyer-dname-root]
Bortzmeyer, S., "Using DNAME in the root for the delegation of special-use TLDs", draft-bortzmeyer-dname-root-00 (work in progress), April 2016.

[I-D.ietf-dnsop-resolver-priming]
Koch, P., Larson, M., and P. Hoffman, "Initializing a DNS Resolver with Priming Queries", draft-ietf-dnsop-resolver-priming-07 (work in progress), March 2016.

[I-D.muks-dns-message-fragments]
Sivaraman, M., Kerr, S., and D. Song, "DNS message fragments", draft-muks-dns-message-fragments-00 (work in progress), July 2015.

[I-D.song-dnsop-ixfr-fallback]
Song, L., "An IXFR Fallback to AXFR Case", draft-song-dnsop-ixfr-fallback-01 (work in progress), May 2016.

[I-D.wessels-edns-key-tag]
Wessels, D., "The EDNS Key Tag Option", draft-wessels-edns-key-tag-00 (work in progress), July 2015.

[I-D.wkumari-dnsop-trust-management]
Kumari, W., Huston, G., Hunt, E., and R. Arends, "Signalling of DNS Security (DNSSEC) Trust Anchors", draft-wkumari-dnsop-trust-management-01 (work in progress), October 2015.

[ICANN-ROOT-ROLL]
"Root Zone KSK Rollover Plan", 2016, <https://www.iana.org/reports/2016/root-ksk-rollover-design-20160307.pdf>.

[KROLL-ISSUE]
"A DNSSEC issue during Yeti KSK rollover", 2016, <http://yeti-dns.org/yeti/blog/2016/10/26/A-DNSSEC-issue-during-Yeti-KSK-rollover.html>.

[pcapdump-bug-report]
Bortzmeyer, S., "pcaputils: IWBN to have an option to run a program after file rotation in pcapdump", 2009, <https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=545985>.

[Renumbering-J-Root]
Wessels, D., "Thirteen Years of "Old J-Root"", 2015, <https://indico.dns-oarc.net/event/24/session/10/contribution/10/material/slides/0.pdf>.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <http://www.rfc-editor.org/info/rfc1034>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <http://www.rfc-editor.org/info/rfc1035>.

[RFC4986] Eland, H., Mundy, R., Crocker, S., and S. Krishnaswamy, "Requirements Related to DNS Security (DNSSEC) Trust Anchor Rollover", RFC 4986, DOI 10.17487/RFC4986, August 2007, <http://www.rfc-editor.org/info/rfc4986>.

[RFC5011] StJohns, M., "Automated Updates of DNS Security (DNSSEC) Trust Anchors", STD 74, RFC 5011, DOI 10.17487/RFC5011, September 2007, <http://www.rfc-editor.org/info/rfc5011>.

[RFC7626] Bortzmeyer, S., "DNS Privacy Considerations", RFC 7626, DOI 10.17487/RFC7626, August 2015, <http://www.rfc-editor.org/info/rfc7626>.

[ROOT-FAQ] Karrenberg, D., "DNS Root Name Server FAQ", 2007, <https://www.isoc.org/briefings/020/>.

[Root-Zone-Database] "Root Zone Database", <http://www.iana.org/domains/root/db>.

[xml-file] "XML source file of Yeti experience draft", 2016, <https://github.com/songlinjian/song-iet-draft/tree/master/Yeti-experience>.

[Yeti-DM-Sync-MZSK] "Yeti DM Synchronization for MZSK", 2016, <https://github.com/BII-Lab/Yeti-Project/blob/master/doc/Yeti-DM-Sync-MZSK.md>.

[Yeti-DNS-Project] "Website of Yeti DNS Project", <http://www.yeti-dns.org>.

[Yeti-glue-issue] "Yeti Glue Issue", 2015, <http://yeti-dns.org/resource/Yeti-glue-issue.txt>.

Appendix A. The Yeti root server in hint file

REMOVE BEFORE PUBLICATION: Currently in Yeti testbed, there are cases that multiple servers run by single operator, like VeriSgin runs A and J. It is allowed because we need more server to satisfy Yeti experiment requirement. The name of those servers share common top domain name like yeti.eu.org, dns-lab.net, yeti-dns.net. We intentionally pick two random labels (first 30 characters of SHA256([a-e])) to offset the effect of name compression. According to the Yeti policy those servers will be reclaimed if qualified volunteers apply to host a Yeti server.

```
.                           3600000    IN   NS       bii.dns-lab.net
bii.dns-lab.net             3600000    IN   AAAA     240c:f:1:22::6
.                           3600000    IN   NS       yeti-ns.tisf.net
yeti-ns.tisf.net            3600000    IN   AAAA     2001:559:8000::6
.                           3600000    IN   NS       yeti-ns.wide.ad.jp
yeti-ns.wide.ad.jp          3600000    IN   AAAA     2001:200:1d9::35
```

```
.                             3600000    IN   NS       yeti-ns.as59715.net
yeti-ns.as59715.net           3600000    IN   AAAA
2a02:cdc5:9715:0:185:5:203:53
.                             3600000    IN   NS       dahu1.yeti.eu.org
dahu1.yeti.eu.org             3600000    IN   AAAA
2001:4b98:dc2:45:216:3eff:fe4b:8c5b
.                             3600000    IN   NS       ns-yeti.bondis.org
ns-yeti.bondis.org            3600000    IN   AAAA     2a02:2810:0:405::250
.                             3600000    IN   NS       yeti-ns.ix.ru
yeti-ns.ix.ru                 3600000    IN   AAAA     2001:6d0:6d06::53
.                             3600000    IN   NS       yeti.bofh.priv.at
yeti.bofh.priv.at             3600000    IN   AAAA     2a01:4f8:161:6106:1::10
.                             3600000    IN   NS       yeti.ipv6.ernet.in
yeti.ipv6.ernet.in            3600000    IN   AAAA     2001:e30:1c1e:1::333
.                             3600000    IN   NS       yeti-
dns01.dnsworkshop.org
yeti-dns01.dnsworkshop.org    3600000    IN   AAAA     2001:1608:10:167:32e::
53
.                             3600000    IN   NS       yeti-ns.conit.co
yeti-ns.conit.co              3600000    IN   AAAA
2604:6600:2000:11::4854:a010
.                             3600000    IN   NS       dahu2.yeti.eu.org
dahu2.yeti.eu.org             3600000    IN   AAAA     2001:67c:217c:6::2
.                             3600000    IN   NS       yeti.aquaray.com
yeti.aquaray.com              3600000    IN   AAAA     2a02:ec0:200::1
.                             3600000    IN   NS       yeti-ns.switch.ch
yeti-ns.switch.ch             3600000    IN   AAAA     2001:620:0:ff::29
.                             3600000    IN   NS       yeti-ns.lab.nic.cl
yeti-ns.lab.nic.cl            3600000    IN   AAAA     2001:1398:1:21::8001
.                             3600000    IN   NS       yeti-ns1.dns-lab.net
yeti-ns1.dns-lab.net          3600000    IN   AAAA     2001:da8:a3:a027::6
.                             3600000    IN   NS       yeti-ns2.dns-lab.net
yeti-ns2.dns-lab.net          3600000    IN   AAAA     2001:da8:268:4200::6
.                             3600000    IN   NS       yeti-ns3.dns-lab.net
yeti-ns3.dns-lab.net          3600000    IN   AAAA     2400:a980:30ff::6
.                             3600000    IN   NS
ca978112ca1bbdcafac231b39a23dc.yeti-dns.net
ca978112ca1bbdcafac231b39a23dc.yeti-dns.net 3600000    IN   AAAA
2c0f:f530::6
.                             3600000    IN   NS
3f79bb7b435b05321651daefd374cd.yeti-dns.net
3f79bb7b435b05321651daefd374cd.yeti-dns.net 3600000    IN   AAAA
2401:c900:1401:3b:c::6
.                             3600000    IN   NS       xn--r2bi1c.xn--
h2bv6c0a.xn--h2brj9c
xn--r2bi1c.xn--h2bv6c0a.xn--h2brj9c 3600000    IN   AAAA     2001:e30:1c1e:
10::333
.                             3600000    IN   NS       yeti1.ipv6.ernet.in
```

```
yeti1.ipv6.ernet.in             3600000    IN   AAAA     2001:e30:187d::333
.                               3600000    IN   NS       yeti-
dns02.dnsworkshop.org
yeti-dns02.dnsworkshop.org      3600000    IN   AAAA     2001:19f0:0:1133::53
.                               3600000    IN   NS       yeti.mind-dns.nl
yeti.mind-dns.nl                3600000    IN   AAAA     2a02:990:100:b01::53:0
.                               3600000    IN   NS       yeti-ns.datev.net
yeti-ns.datev.net               3600000    IN   AAAA     2a00:e50:f15c:
1000::1:53
```

Authors' Addresses

Linjian Song
Beijing Internet Institute
2508 Room, 25th Floor, Tower A, Time Fortune
Beijing 100028
P. R. China

Email: songlinjian@gmail.com
URI: http://www.biigroup.com/

Shane Kerr
Beijing Internet Institute
2/F, Building 5, No.58 Jinghai Road, BDA
Beijing 100176
CN

Email: shane@biigroup.cn
URI: http://www.biigroup.com/

Dong Liu
Beijing Internet Institute
2508 Room, 25th Floor, Tower A, Time Fortune
Beijing 100028
P. R. China

Email: dliu@biigroup.com
URI: http://www.biigroup.com/

Paul Vixie
TISF
11400 La Honda Road
Woodside, California 94062
US

Email: vixie@tisf.net
URI: http://www.redbarn.org/

Akira Kato
Keio University/WIDE Project
Graduate School of Media Design, 4-1-1 Hiyoshi, Kohoku
Yokohama 223-8526
JAPAN

Email: kato@wide.ad.jp
URI: http://www.kmd.keio.ac.jp/