draft-fujiwara-dnsop-bad-dns-auth-01

IETF61 DNSOP
2004/11/07

Kazunori Fujiwara, JPRS <fujiwara@jprs.co.jp>
Keisuke Ishibashi, NTT <ishibashi.keisuke@lab.ntt.co.jp>
Katsuyasu Toyama, NTT <toyama.katsuyasu@lab.ntt.co.jp>

Topics

- Protecting cache server against misconfiguration of DNS authorititive servers
  - We had a terrible cache server overload.
  - This is caused by misconfigured authoritative servers.
  - Protection method is needed for ISP cache server.
- This topic has more issues about DNS protocol.
  - DNS Response size consideration
  - EDNS0 implementation status
  - EDNS0(with fragmentation) and IPv6
  - DNS anycast vs TCP query (not mentioned in the draft -01)

Cache server overload (1)

- There are some authoritative server misconfiguration
  - large response size RRSet
    - Many(32) PTRs in one IP address
  - no EDNS0
  - TCP filtering at authoritative servers

- and this RRSet is major IP address, many clients query this address frequently.

- What happenes as a result?
  - In every query, truncation occurs.
  - At that time, cache server queries again by TCP.
  - But TCP is blocked by filter.
  - Then cache server has many ’stocled’ TCP SYN_SENT status and makes high load.

Cache server overload (2)

- Authoritative server misconfiguration can create significant overloads on cache servers.
  - This behavior was found through the observation of query traffic to/from ISP cache servers.
  - And we reported it in NANOG32 meeting in October.
- Attacker can make a DoS attack to ISP cache servers using this problem.
  - Attacker prepares an authoritative server and a RRSet with this problem.
  - Attacker sends a lot of queries with spoofed source addresses, as if they are sent from various clients.
- From the ISP users view, failure of DNS cache server is almost equal to the failure of the Internet service itself.
- We should protect DNS cache servers.

How to decrease TCP sessions in Resolver server

- One idea: do not query by TCP when truncation
  - It reduces TCP sessions.
  - But the answer which is supposed to be able to get it properly if it listens with TCP can’t be resolved.
  - it cannot cache any data (RFC 2308).
  - All resolving request, the cache server queries to all the authoritative servers by UDP.
  - More, it may violate RFC2181.
- Needs new cache/resolver server algorithm

Cache/Resolver server algorithm improvement

- We propose
  - As before, the cache server queries by UDP (w/wo EDNS0) and if TC bit is set, the cache server queries again the authoritative server by TCP.
  - (new)When queries for all authoritative servers are unsuccessful, the cache server caches that RRSet(name,class,type) as unresolvable.
  - (new)Next query for the same RRSet from stub resolvers, the cache server does not query to authoritative servers and answers "unresolvable".
  - (optional)Cache server marks misconfigured servers which does not respond TCP. (equivalent to BIND9’s EDNS0 capability database.)

Protocol consideration

- RFC2308 section 7 - Other Negative Responses
  - This section does not mention about TCP filtering.
    - UDP is OK
    - no answer by TCP, no TCP reset
  - RFC 2308 7.1: "... In either case a resolver MAY cache a server failure response. If it does so it MUST NOT cache it for longer than five (5) minutes, and it MUST be cached against the specific query tuple <query name, type, class, server IP address>."
- But
  - 5 minutes is too small to protect from DoS.
  - In our case, authoritative server's misconfiguration lasted in about a half year.
- Our proposal
  - For protecting cache servers, we recommend to cache unresolvable information for several hours.

DNS Response size consideration

- DNS response size lower than 512 octets
  - safe with Original UDP DNS protocol
- 512 < DNS response size <= 1280 - (IP/UDP)header size (1200? octets)
  - safe with EDNS0 without IPv6 fragmentation
  - (on the present Internet, IPv4 is the same as IPv6.)
  - TCP is OK
- 1200? < DNS response size
  - EDNS0 needs IP/IPv6 fragmentation
  - TCP is OK

EDNS0 implementation status

- Question
  - Now, EDNS0 requirement is "SHOULD". Is this OK?
  - When will EDNS0 requirement be "MUST" ?
- This discussion is need for enum-wg.
- RRSet may be large in ENUM.

EDNS0(with fragmentation) and IPv6

- According to RFC3226 section 3
  - "All RFC 2535 and RFC 2874 compliant entities MUST be able to handle fragmented IPv4 and IPv6 UDP packets." (to support EDNS0 with large response size)
- But RFC2460 "IPv6 Specification" section 5
  - "the use of such fragmentation is discouraged in any application that is able to adjust its packets to fit the measured path MTU."
- Question
  - EDNS0 with large response size requires IPv4 and IPv6 fragmentation. Is it OK? (I think OK.)

DNS anycast vs TCP query (not mentioned in the draft -01)

- TCP queries may work on DNS anycast with BGP.
  - Routing information may be stable for a short time.
  - Equal cost multi path doesn’t occur in principle.
    - But ECMP problem occurs in the Internet, more investigation is necessary.
  - TCP communication may work for a short time.
  - DNS queries using TCP is completed in a short time.
- DNS anycast with IGP
  - "Equal cost multi path" problem can be solved with per flow routing.
- Need more consideration.

A minimal requirement may be

- DNS response size exceeds 512 octets
  - the authoritative name servers MUST permit TCP queries
  - or MUST support EDNS0
- DNS response size exceeds 1200 octets
  - the authoritative name servers MUST permit TCP queries
  - AND MUST support EDNS0

Summary

- TCP support for DNS is now mandatory, but there are many authoritative servers which do not support TCP.
- EDNS0 has IP/IPv6 fragment issues.
- Still need for protection mechanism for DNS cache server.
- This I-D should be separated as two I-Ds.
  - Negative cache issue
  - Today's DNS requirements

Acknowledgement

- We would like to thank Ichiro Mizukoshi, Haruhiko Ohshima, Masahiro Ishino, Chika Yoshimura, Tsuyoshi Toyono, Hirotaka Matsuoka, Yasuhiro Morisita, Bill Manning and Rob Austein.

Questions

- need Comments
- Please discuss in dnsop mailing list.