

Evaluation and consideration of multiple responses

Kazunori Fujiwara, JPRS

fujiwara@jprs.co.jp

OARC 28

Past discussion

- Background
 - DNS is query response based protocol
 - Each query contains one QNAME / QTYPE pair
 - Recent applications request multiple DNS queries.
 - For example, dual stack host (IPv4 and IPv6) requires both IPv4 address (type A) and IPv6 address (type AAAA)
- There have been many proposals to optimize the situation
 - (QDCOUNT > 1)
 - New Query Type that queries both A and AAAA (similar to MAILB or ANY)
 - New DNS protocol that carry multiple queries / responses
 - Pre-populate resolver's cache

Recent multiple response proposals

- draft-vavrusa-dnsop-aaaa-for-free
 - Additional AAAA in answer section
- draft-wkumari-dnsop-multiple-responses
 - Pseudo RR controls additional RRs
- draft-fujiwara-dnsop-additional-answers
 - Developers choose additional RRs (+NSEC*)
- draft-bellis-dnsexext-multi-qtypes
 - New EDNS option carries additional qtypes
- draft-yao-dnsop-accompanying-questions
 - New EDNS option carries additional qnames, qtypes, rcodes

Recent multiple response proposals

- These two drafts propose new DNS protocol that carry multiple queries/responses in EDNS0
 - We need to develop new resolvers to evaluate
 - Evaluation is not easy, not tested yet
- draft-bellis-dnsexth-multi-qtypes
 - New EDNS option carries additional qtypes
- draft-yao-dnsop-accompanying-questions
 - New EDNS option carries additional qnames, qtypes, rcodes

Recent multiple response proposals

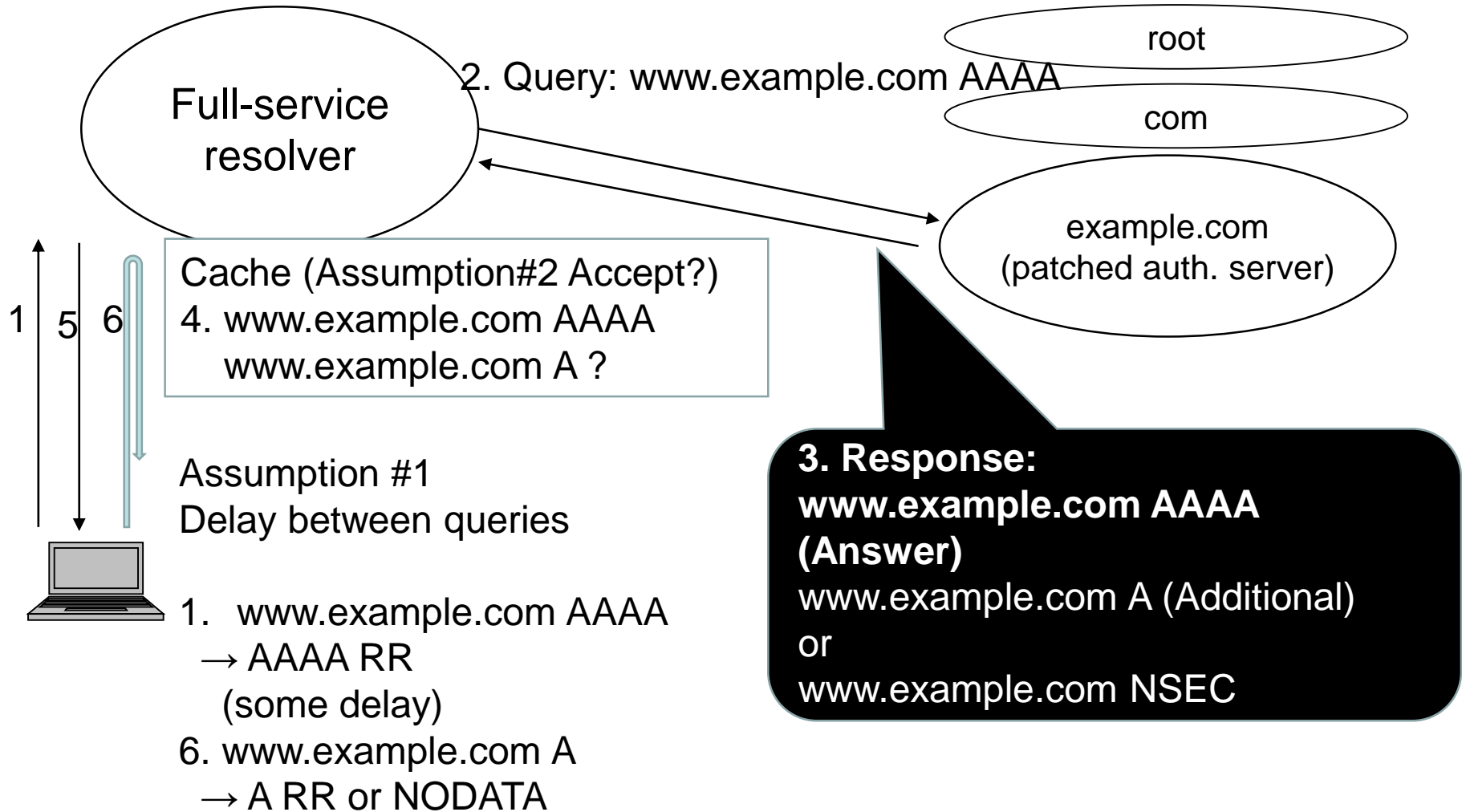
- draft-vavrusa-dnsop-aaaa-for-free
 - Additional AAAA in answer section
- draft-wkumari-dnsop-multiple-responses
 - Pseudo RR controls additional RRs
- draft-fujiwara-dnsop-additional-answers
 - Developers choose additional RRs (+NSEC*)

- These 3 drafts propose additional RRs in responses
 - Pre-populate resolvers' cache approach
 - Query is not changed
 - Authoritative server implementation is easy
- Can evaluate current full-resolver implementations

draft-fujiwara-dnsop-additional-answers proposes adding NSEC in Responses

- RFC 8198: Aggressive Use of DNSSEC-Validated Cache
- RFC 8198 enabled NODATA response generation from cached matching NSEC
 - v6.example. IN NSEC zz.example. AAAA RRSIG NSEC
 - If the NSEC RR in the cache, resolvers can generate NODATA of v6.example. A immediately

Idea: Pre-populate resolvers' cache



Possible additional answer pairs

Query: additional answer

- name A: name AAAA (answer section)
- name A: name AAAA (additional section)
- name A: name NSEC* (authority section)
- name AAAA: name A (additional section)
- name AAAA: name NSEC* (authority section)

- Existing case of current implementation
 - name MX: mail_exchange A/AAAA (additional)
 - name SRV: target_host A/AAAA (additional)
- Different QNAME scenario
 - name A/AAAA: _443._tcp.name TLSA/NSEC*
 - _443._tcp.name TLSA: name A/AAAA/NSEC*

the evaluation

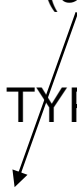
- Full-service resolvers accept additional Resource Records ?
 - (To check assumption 2)
- Purpose of the evaluation
 - To check easy deployment
 - If existing resolvers accept additional RRs in responses, deployment is easy
 - Updating authoritative server is easy
 - To know implementation status of RFC 8198

Authoritative server implementation

- NSD (4.1.19) is easy to read and patch
- Add a code at add_rrset() in nsd/query.c
- Always add A / AAAA / NSEC RRs
- http://member.wide.ad.jp/~fujiwara/files/nsd-always-add-a_aaaa_nsec.diff
- http://member.wide.ad.jp/~fujiwara/files/nsd-always-add-a_aaaa_in_anssec.diff

```
add_rrset(...) {  
    ....  
    switch (rrset_rrtype(rrset)) {  
case TYPE_A:  
    rrset2 = domain_find_rrset(owner, query->zone, TYPE_AAAA);  
    if (rrset2) {  
        answer_add_rrset(answer, ADDITIONAL_A_SECTION, owner, rrset2);  
    } else {  
        answer_nodata(query, answer, owner); // add NSEC* (and SOA)  
    }  
    break;
```

Or ANSWER_SECTION
(aaaa for free)



Experiment authoritative servers

- Patched NSD (203.178.129.11) appends A,AAAA in additional section, NSEC in authority section
 - additional.dnslab.jp and additional-nosec.dnslab.jp zones
 - both.additional{,-nsec}.dnslab.jp has both A, AAAA (+NSEC)
 - v6.additional{,-nsec}.dnslab.jp has AAAA only (+NSEC)
 - v4.additional{,-nsec}.dnslab.jp has A only (+NSEC)
 - Signed with NSEC (additional.dnslab.jp)
 - Try drill -D -o rd @203.178.129.11 both.additional.dnslab.jp A
drill -D -o rd @203.178.129.11 v4.additional.dnslab.jp A
- Patched NSD (203.178.129.10) appends A,AAAA in answer section (aaaa-for-free scenario)
 - answer-aaaa.dnslab.jp and answer-aaaa-nosec.dnslab.jp zones
 - both.answer-aaaa{,-nsec}.dnslab.jp has both A,AAAA
 - Try drill -D -o rd @203.178.129.10 both.answer-aaaa.dnslab.jp A

Tested resolvers

- BIND 9.11.2
- BIND 9.12.1-rc1 (v9_12 branch at March 6, 2018)
 - Support RFC 8198 (NSEC)
- Knot Resolver 1.5.3
- Knot Resolver 2.0.0
 - Supports RFC 8198 (NSEC)
- Unbound 1.6.7
- Unbound 1.7.0rc1
 - Supports RFC 8198 (NSEC): “aggressive-nsec: yes”
- PowerDNS Recursor 4.1.1
- Google Public DNS

Test result

- BIND 9, Google Public DNS
 - don't accept/use all additional RRs (A, AAAA, NSEC)
- Unbound, Knot Resolver
 - don't accept/use additional A, AAAA
 - accept additional NSEC in authority section
 - Knot resolver 2.0.0 and Unbound 1.7.0rc1 generate NODATA using cached NSEC RR
- PowerDNS recursor
 - does not accept/use A,AAAA in additional section
 - accepts additional A,AAAA in answer section (with/without dnssec)

Reason of rejecting additional RRs

- Current full-service resolver implementations reject additional RRs because they avoid cache poisoning
- Recent full-service resolvers accept RRs that match query (QNAME, QTYPE)
 - Even if additional RRs are DNSSEC signed
- Small/minimal response size is preferred

Summary

- Pre-populate resolvers' cache approach requires updating resolver software to accept additional RRs
 - Written in Section 5.4.1 “Ranking Data” of RFC 2181 and Section 8 of draft-wkumari-dnsop-multiple-responses-05
 - Knot resolver 2.0.0 and Unbound 1.7.0rc1 accept additional NSEC RRs and generate NODATA responses (a half of additional-answer scenario)
 - PowerDNS recursor accepts additional A/AAAA in answer section (aaaa-for-free scenario)

Next steps ?

- My intent is to increase the effect of RFC 8198
- Another (minimal) proposal maybe
 - Aggressive append NSEC* by authoritative server
 - Accept additional NSEC* by full-service resolver that support RFC 8198