Packet Traces from a Simulated Signed Root

Duane Wessels
DNS-OARC

DNS-OARC Workshop
Beijing, China
November 2009

Background

- We know from active measurements that some DNS resolvers cannot receive “large” responses.
  - Middleboxes block responses > 512 octets
  - Firewalls block UDP fragments (> 1480 octets)
- We know from existing data that root DNS servers see a small percentage of queries with
  - DO=1
  - EDNS bufsiz=512

Background

- We know that recently signed TLDs (.org) experienced a significant increase in queries over TCP.
- Some resolver products do not support TCP transport.

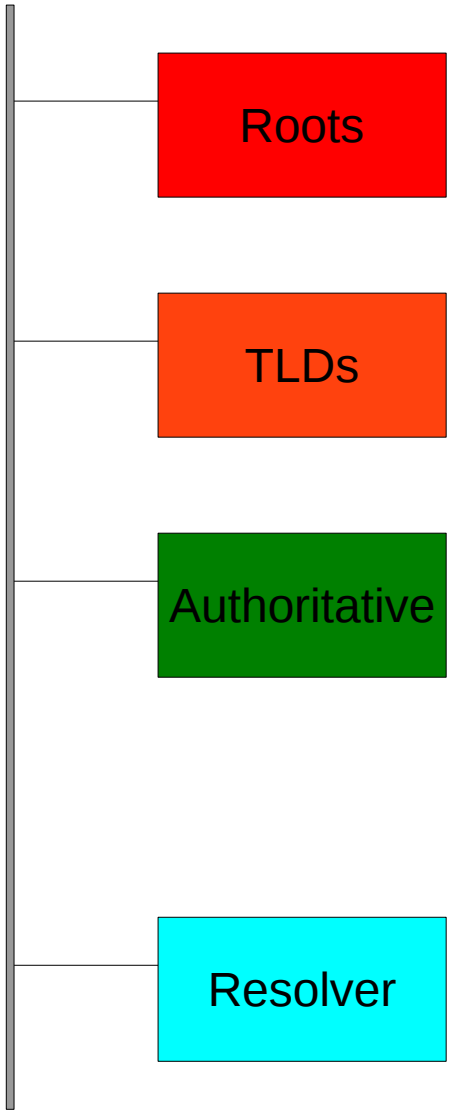
Fear

- When the root becomes signed, some clients will effectively be cut off from the root nameservers.
- Root nameservers will see a sudden increase in query rate (both TCP and UDP) due to panicky resolver clients.

Goal

- To visualize and understand how signing the root affects “EDNS 512 DO=1” clients.

Simulation Setup



- Nameservers run as FreeBSD jails with loopback alias addresses of real servers.
  - (so nameservers for the same zone could be configured differently)
- Zone content taken from actual zones.
  - (so I don't have to make up addresses)
- tcpdump on Resolver
- All caches are empty

Zone Contents

- Root-zone as-is
- TLD zones contain the delegations and necessary glue for example.com and example.net.
- Authoritiative zones include example.com, example.net, iana-servers.net.

DNSSEC Parameters

- KSK(s) 2048-bit RSASHA1
- ZSK(s) 1024-bit RSASHA1
- NSEC

In the following plots...

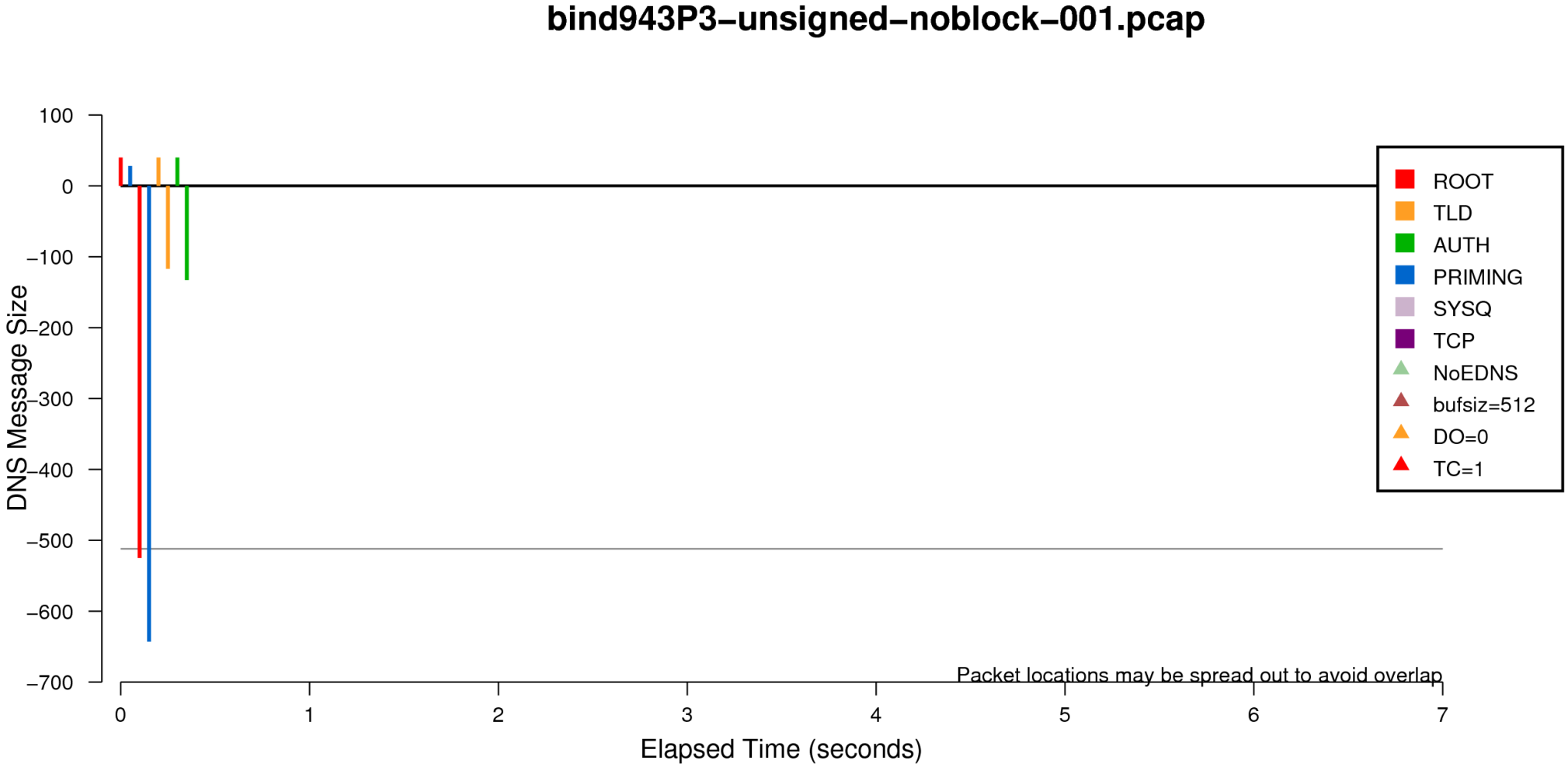
- Packets are represented as vertical bars
- Size of the bar represents the packet size
- Queries above x=0 axis, responses below
- Overlapping packets are spread out in time so we can see them all.

Plot Legend

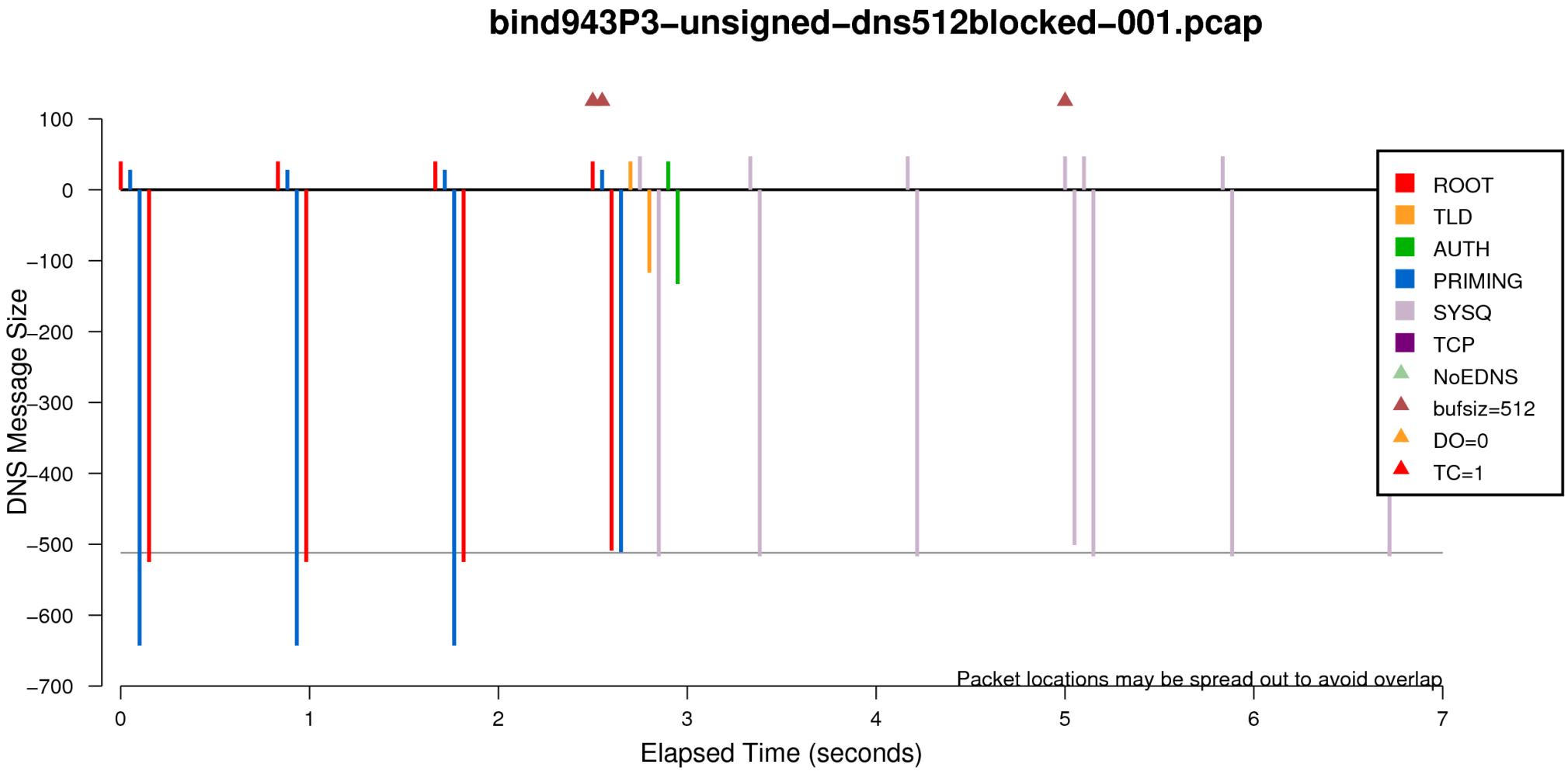
- Red for messages to/from root nameservers
- Orange for messages to/from TLD nameservers
- Green for messages to/from authoritative.
- Blue for priming queries and responses
- Lavender for “SYSQUERY”
  - Address lookup for NS names
- Purple for TCP
- Triangles above packets represent DO=0, TC=1, bufsiz=512, EDNS absence

BIND 9.4.3-P3

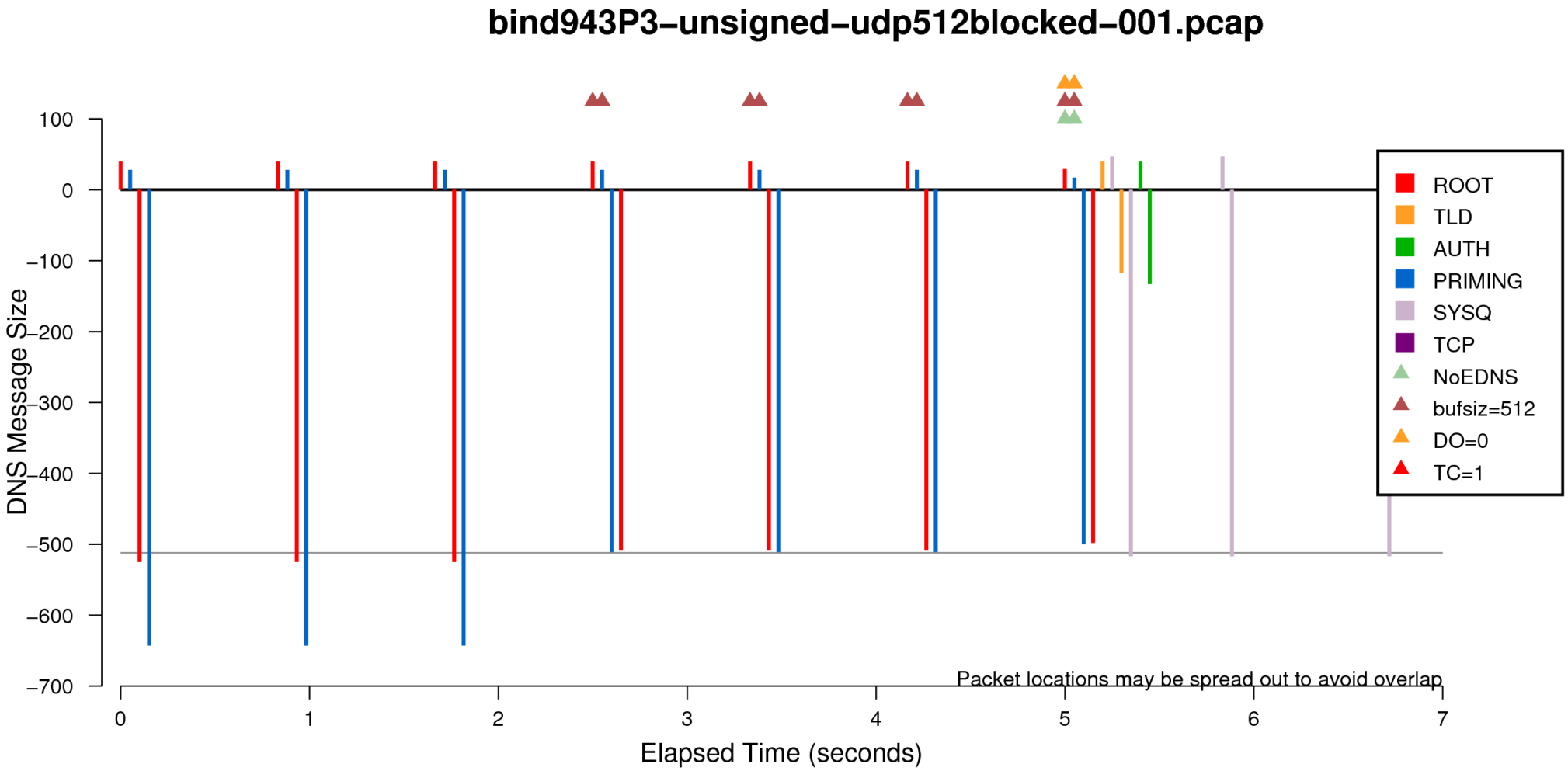
(because this is what comes with FreeBSD, and I'm lazy)



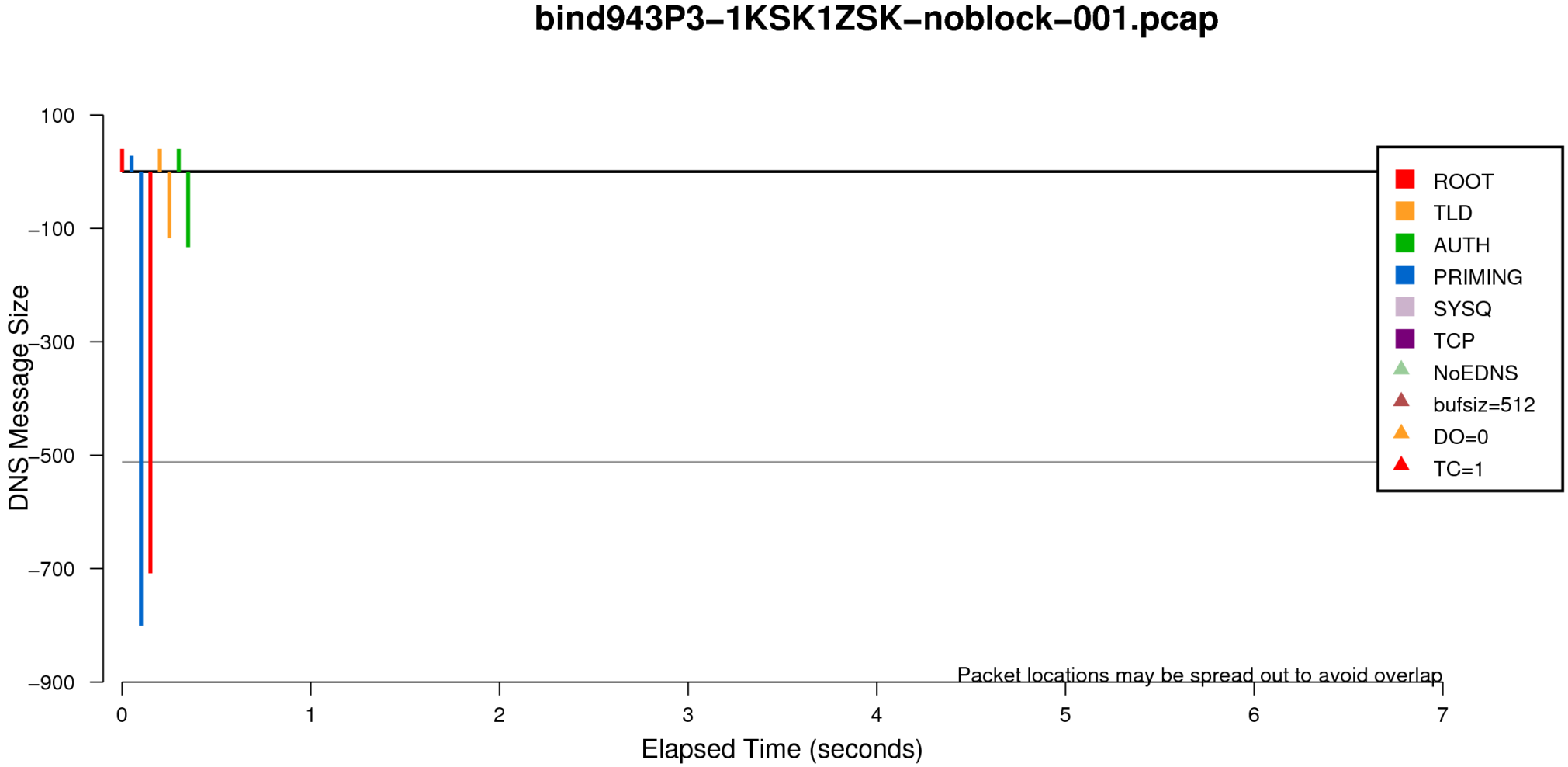
Normal transaction, no responses blocked, no retransmits.



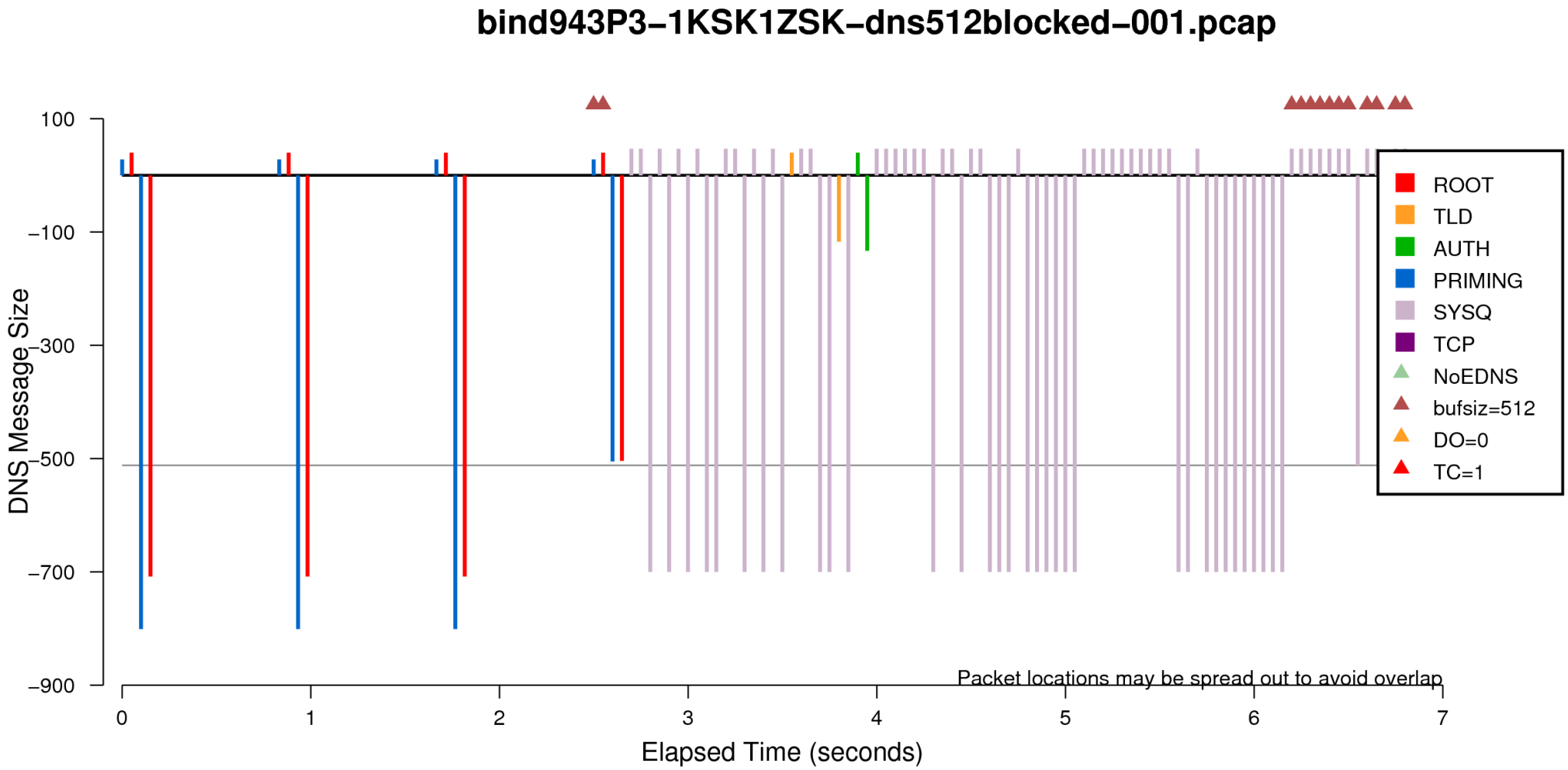
When DNS replies larger than 512 bytes are blocked, we see two retransmits, followed by a third with EDNS bufsiz=512. Then some number of “sysqueries” for glue that was dropped.



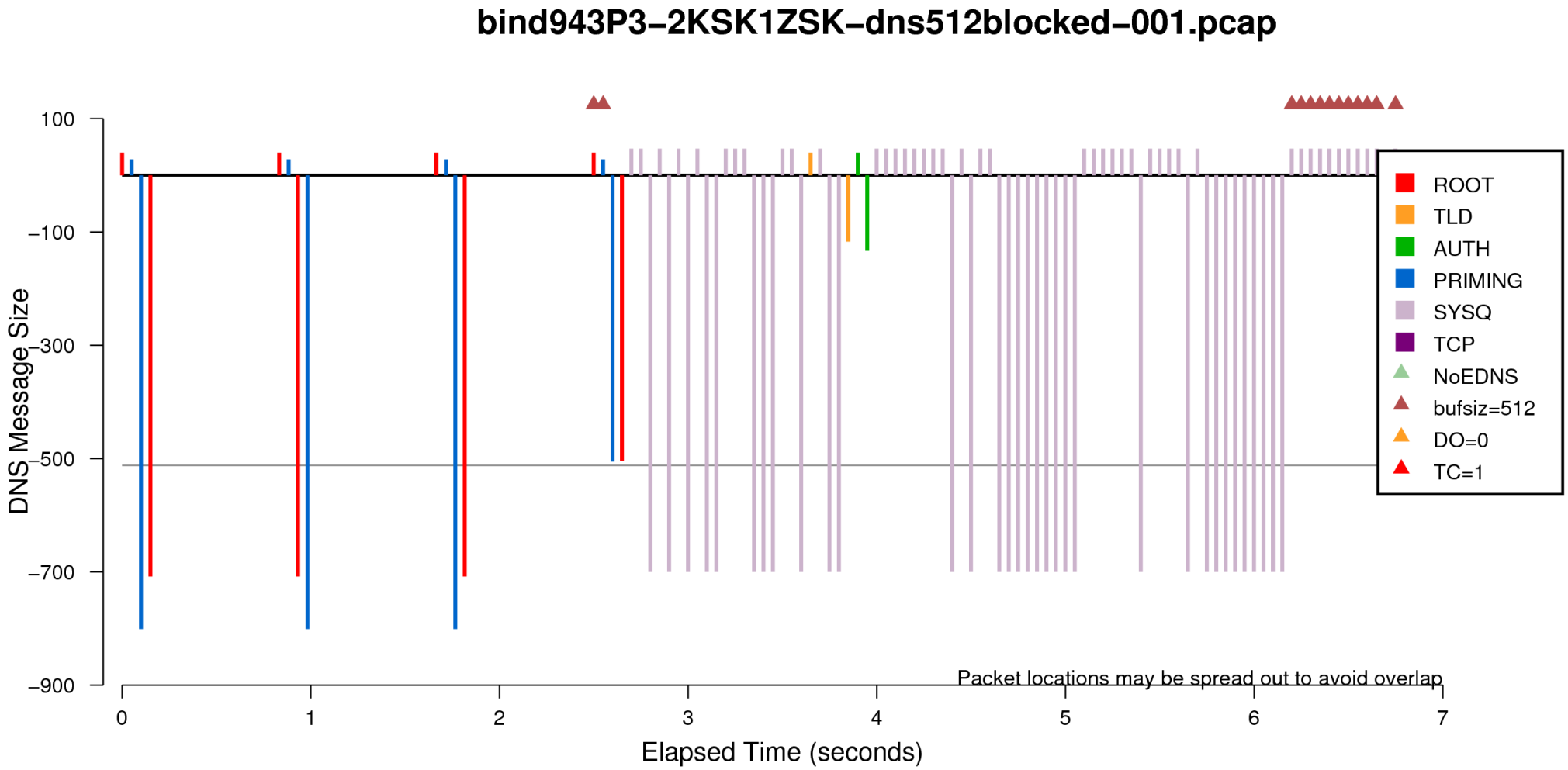
Here's what happens if you misunderstand or mis-remember the specs and block larger-than-512-byte messages based on the **UDP** size, rather than the **DNS** size.



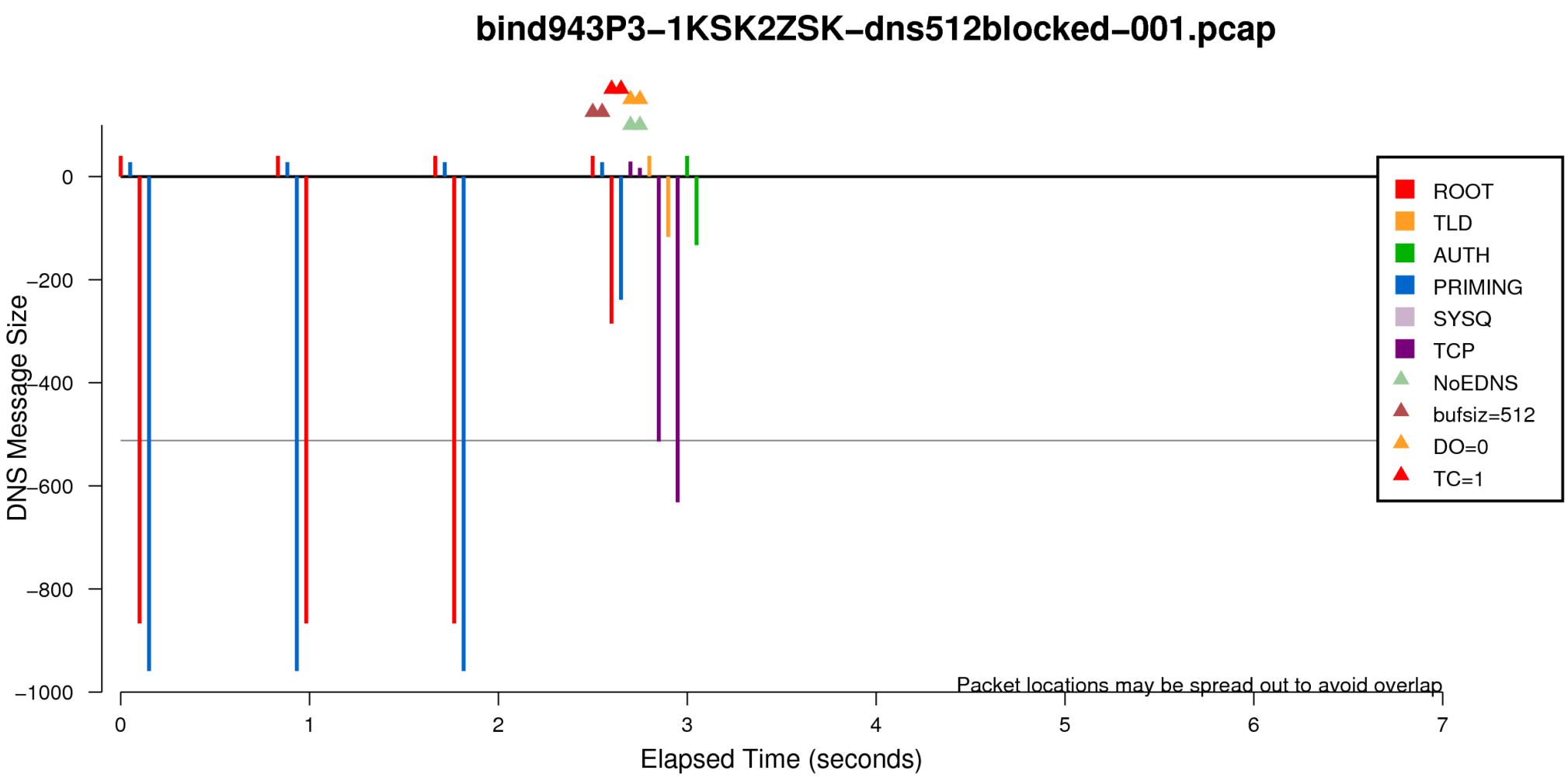
Normal transaction with a signed zone and no blocked rsponses. Responses are slightly larger.



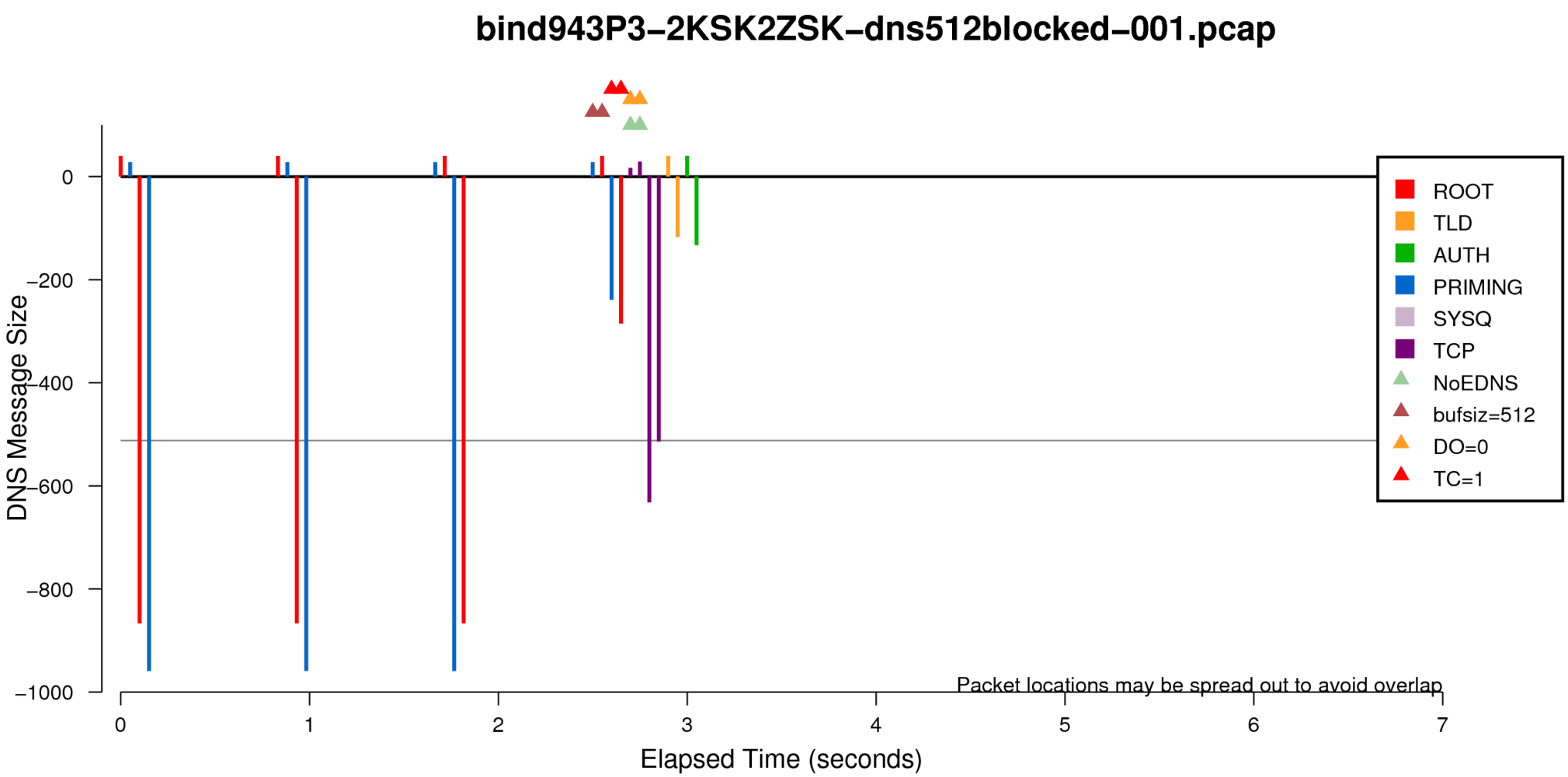
Signed zone and responses larger than 512 octets blocked. Three retransmits with the last having bufsiz=512. Note large number of NS name address lookups and corresponding blocked responses.



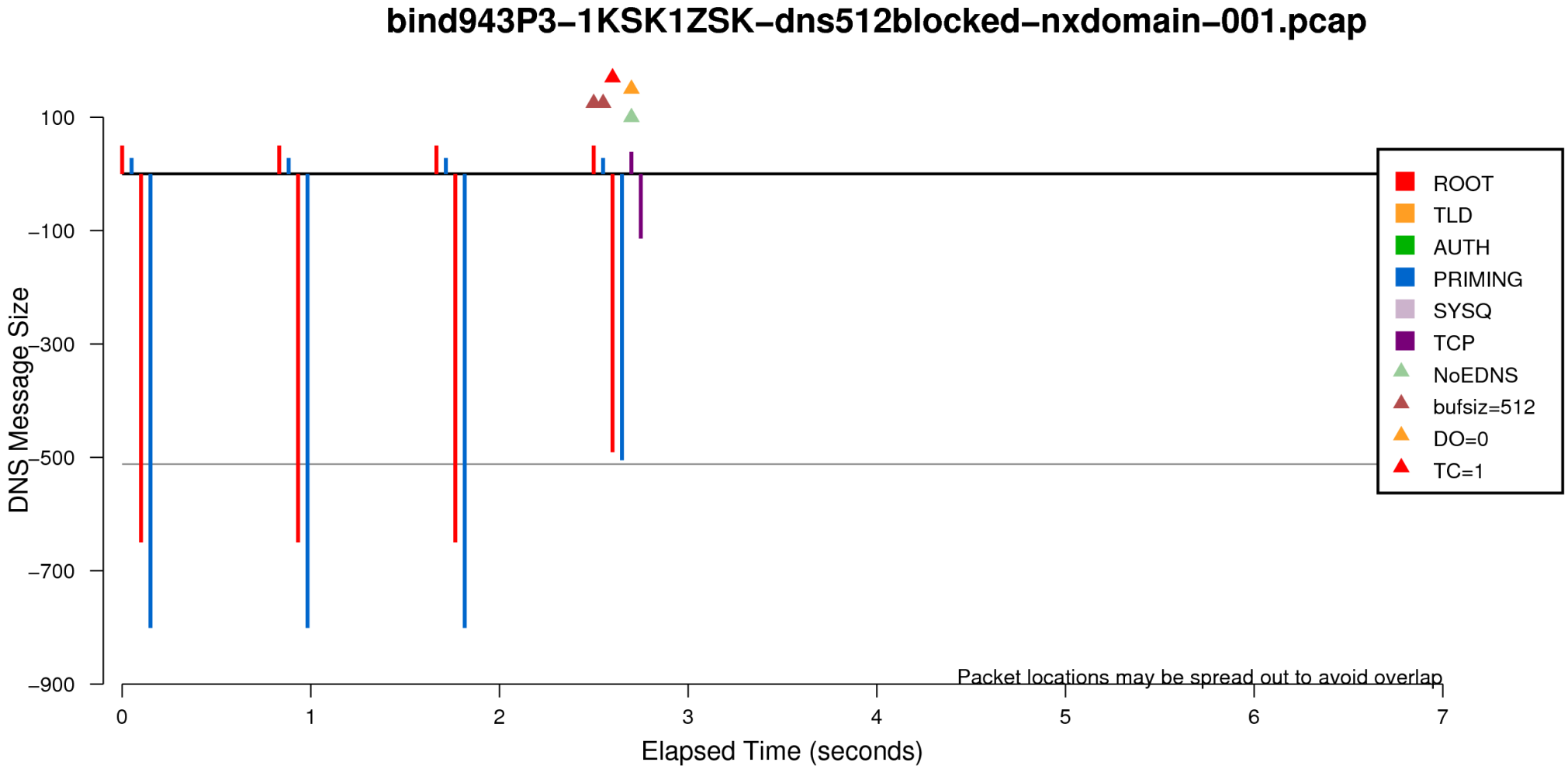
A KSK roll event.



A ZSK roll event. The extra key data in the zone makes responses large enough to force TC=1 and a retry over TCP. Also note BIND-9.4.3P3 appears omit EDNS (and therefore DNSSEC support) over TCP. Both the user query and the priming query happen over TCP.



During worst-case key roll events, there may be up to 2 KSKs and 2 ZSKs in the zone. Looks about the same as the 1KSK/2ZSK case.

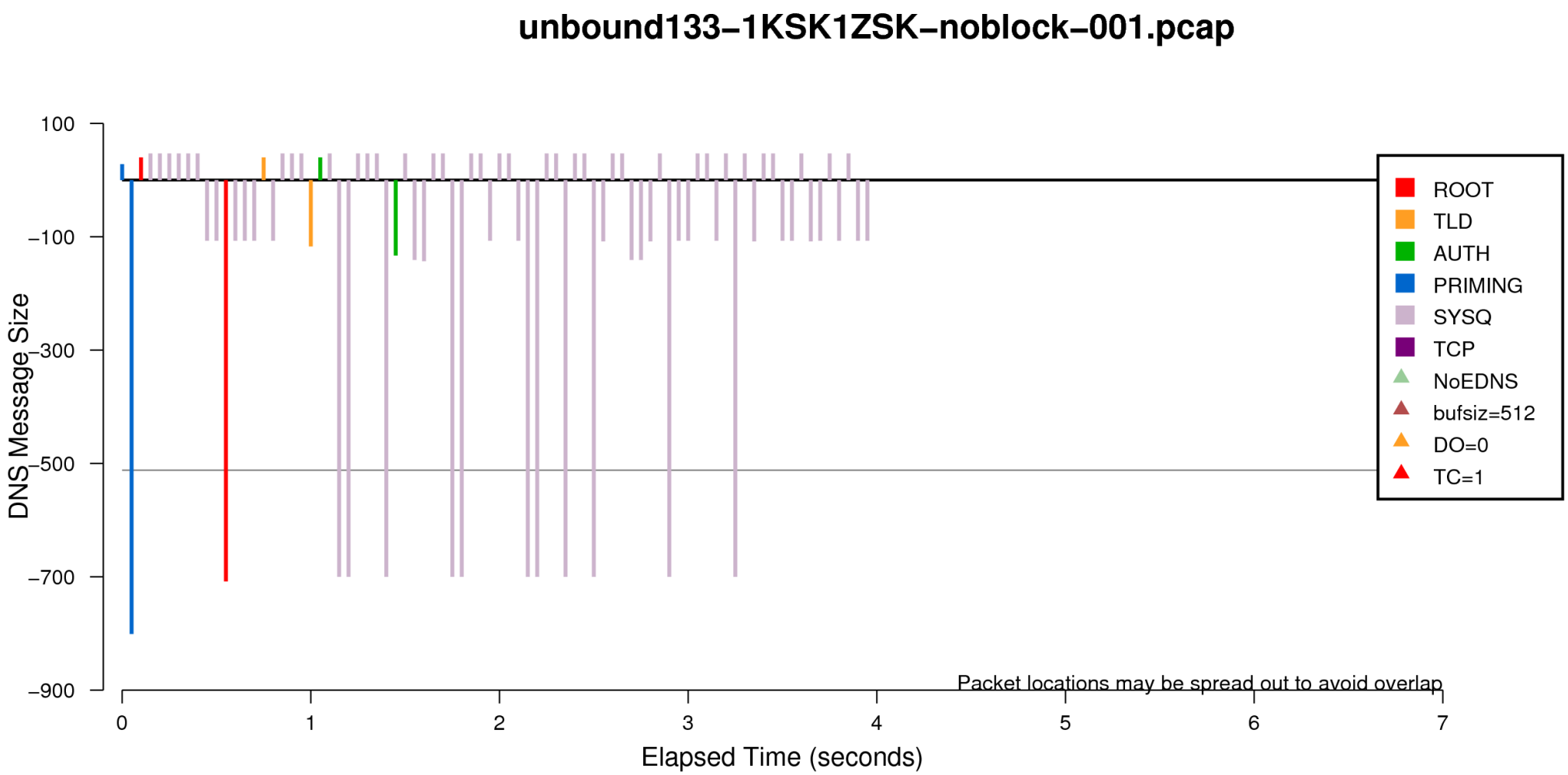


Query to a signed root that results in NXDOMAIN. The response to the user's query was truncated and retried via TCP. The priming query was not truncated.

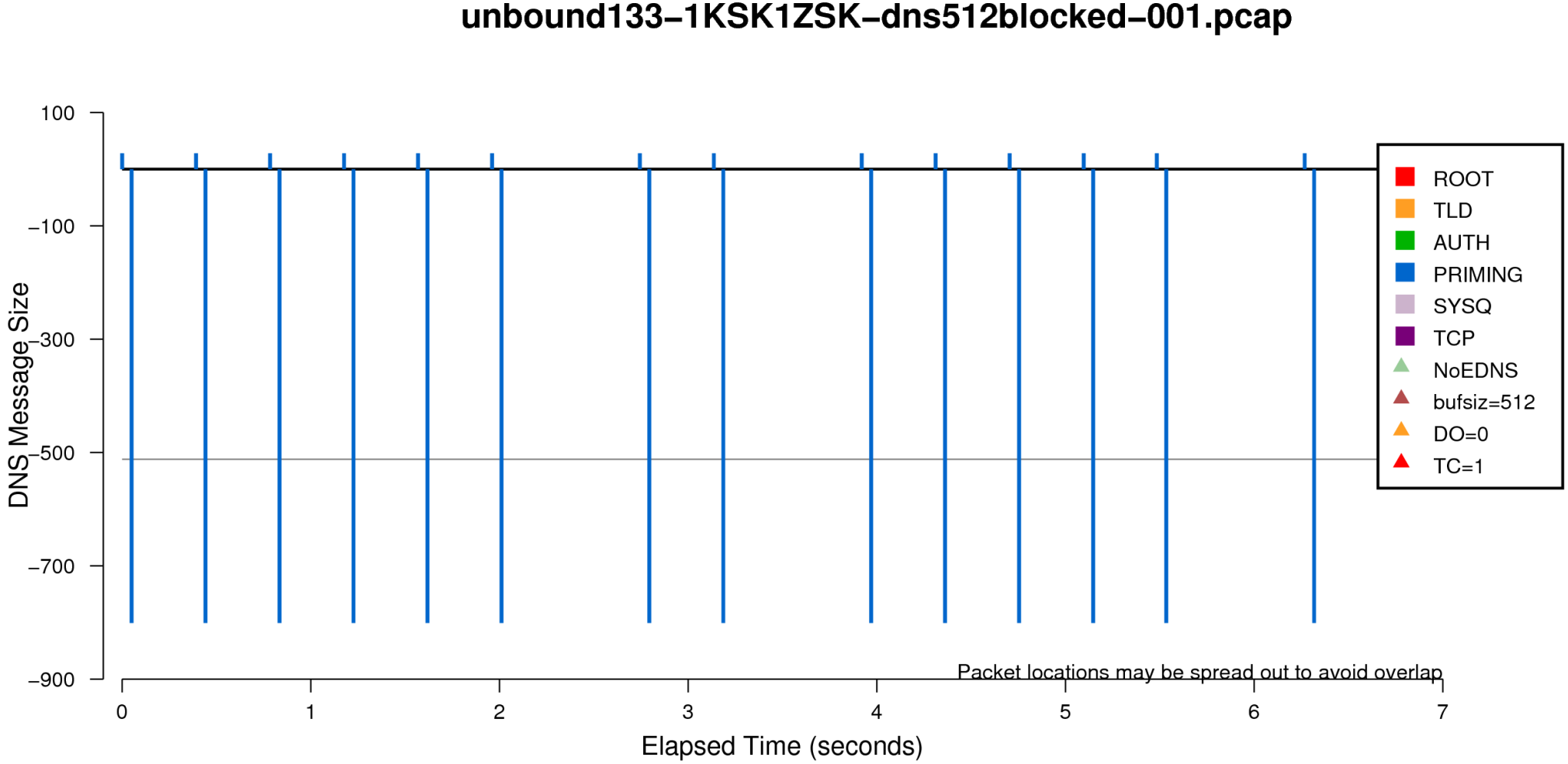
Notes and Thoughts on BIND

- Roots should expect TCP for NXDOMAINs
- Root should expect TCP for referrals during key rollovers
- Need more testing on whether BIND sends EDNS and DO=1 over TCP
- Would you rather have 1 TCP or numerous (25?) UDPs?

Unbound 1.3.3



Normal transaction to a signed root. The unsigned case looks this way as well, with lots of "sysqueries."



Unbound does not fall back to smaller EDNS buffer sizes, nor disable EDNS altogether. The user's query is never sent to a root because the priming query always fails. All cases where large responses are blocked look just like this.

Questions?

Feedback?