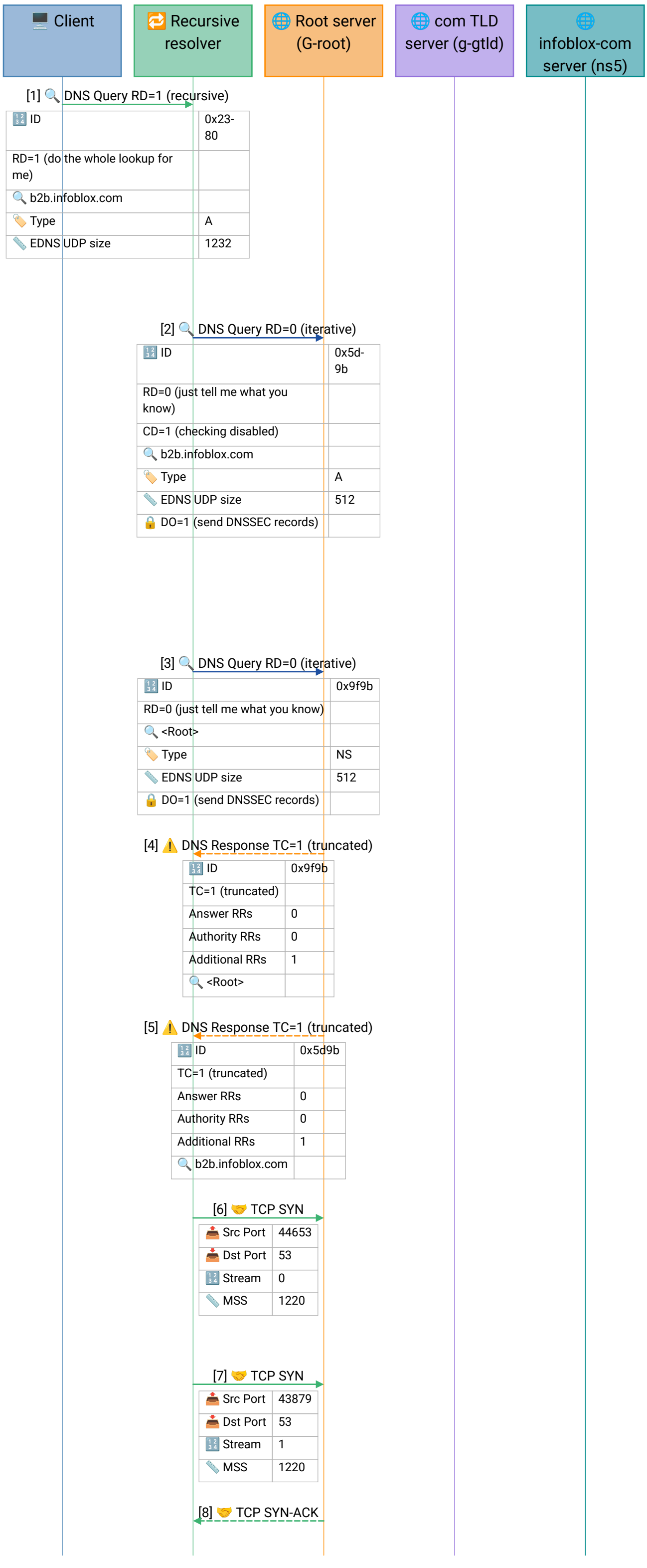


[Frame 1]

Phase 1: the client asks once

The client asks its resolver for the IPv4 address (A record) of `b2b.infoblox.com.` `RD=1` (recursion desired) means "do the whole lookup for me": the client sends nothing else, and every frame until the answer in frame 31 is work the resolver does on its behalf, out of the client's sight.

The query also sets `AD=1`, asking the resolver to report whether it validated the answer (RFC 6840), and carries an EDNS client cookie (RFC 7873). It advertises a 1,232-byte UDP buffer and leaves `DO` clear, so it does not want DNSSEC records back.

[Frame 2]

Phase 2: a cold resolver starts at the root

The resolver starts at the top of the DNS tree and asks a root server. The capture later proves this address is G-root (frame 15). Nothing in the capture supplied that address before this query, so it comes from the resolver's own built-in list of root server addresses (its root hints).

`RD=0`: an iterative query. The root answers only from its own zone and will not chase the name further.

The resolver sends the full name `b2b.infoblox.com` to the root, not just `com`, so it is not using QNAME minimization (RFC 9156). Upstream traffic is IPv6, while the client side was IPv4. The query carries a fresh random ID (`0x5d9b`) and source port (51976), which makes spoofed replies hard to land.

Remember the advertised EDNS UDP size: 512 bytes, with `DO=1` asking for DNSSEC records.

[Frame 3] A priming query (RFC 8109): "who are the root servers?" (`NS` for the root, `<Root>`) . It goes to the same server 0.1 ms after frame 2; the resolver does not wait for it before sending the real query.

Priming swaps the static hints list for the live, signed root NS set. This query also advertises 512 bytes with `DO=1` .

[Frame 4] 27 ms later the root answers the priming query, but with `TC=1` (truncated) and empty answer and authority sections. The full answer is 1,109 bytes (frame 15 shows it), far over the 512 bytes the resolver offered.

The truncated reply is just the header, the question, and the OPT record: 28 bytes, the same size as the query. `AA=1` survives because the root is authoritative for its own NS set.

A resolver that gets `TC=1` must retry over TCP, which is why RFC 7766 makes DNS over TCP mandatory.

[Frame 5] Same outcome for the real query: `TC=1` and no answer. The full referral is 1,179 bytes (frame 19).

The CD bit from frame 2 is copied back, as RFC 4035 requires. Both answers would have fit in the 1,232-byte buffer the client itself used in frame 1. Offering only 512 bytes with `DO=1` against the signed root zone makes this detour close to certain.

[Frame 6]

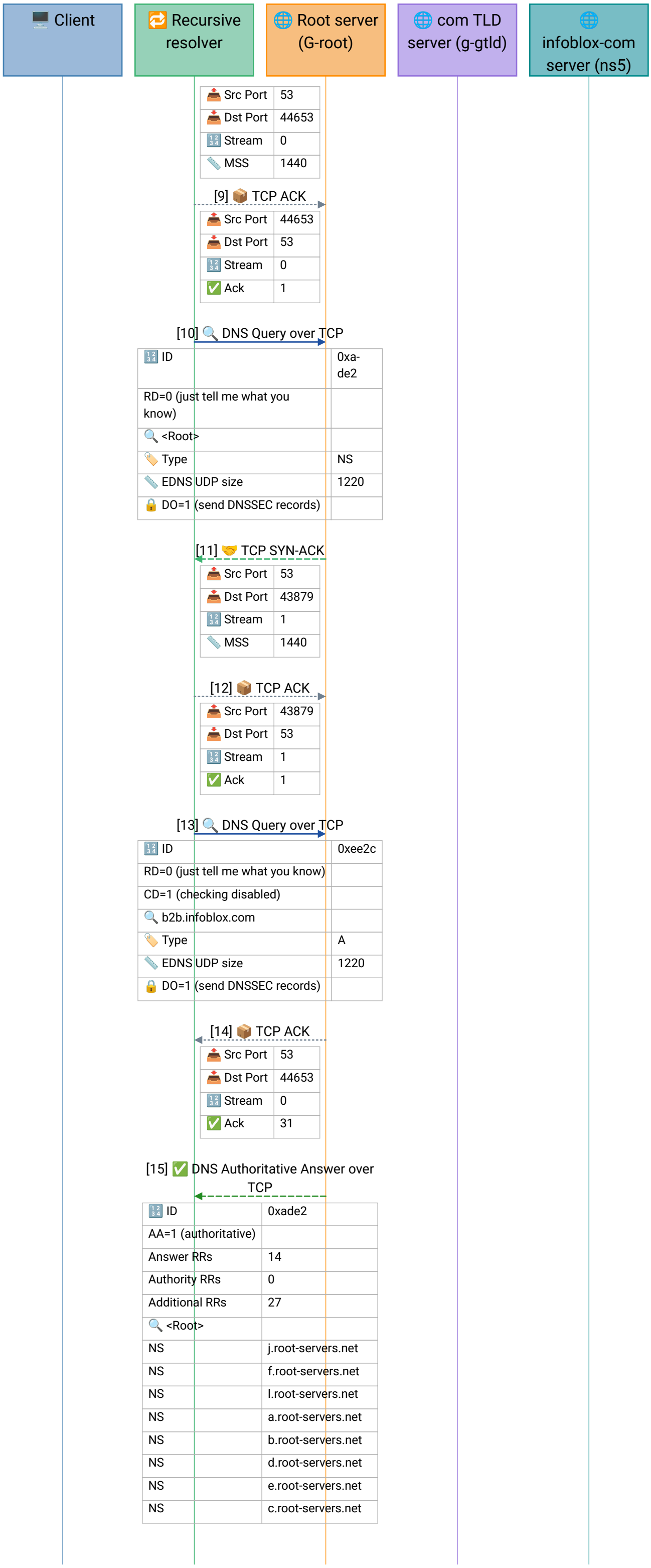
Phase 3: retry over TCP

0.4 ms after the truncated reply, the resolver opens a TCP connection to the same root server on port 53. This connection (source port 44653) will carry the priming query.

`MSS 1220` is 1,280 bytes minus 40 for the IPv6 header and 20 for TCP: consistent with a 1,280-byte MTU (the IPv6 minimum) on the resolver's path.

[Frame 7] A second TCP connection (source port 43879) for the A query. This resolver opens one connection per outstanding query rather than sending both on one connection, which RFC 7766 would allow.

[Frame 8] The root accepts the first connection, 25.9 ms after the SYN: the TCP handshake costs one full round trip



before any DNS can flow. The root's `MSS 1440` fits a 1,500-byte MTU.

[Frame 9] The resolver's ACK completes the handshake on connection 44653.

[Frame 10] The priming query again, now over TCP, with a new ID (`0xade2`). Over TCP each DNS message is preceded by a 2-byte length field (RFC 1035): 30 bytes on the wire = 2 + 28.

The EDNS UDP size now reads 1,220, but it only limits UDP answers. Over TCP the reply can be as large as it needs to be.

[Frame 11] The root accepts the second connection (43879).

[Frame 12] Handshake complete on connection 43879.

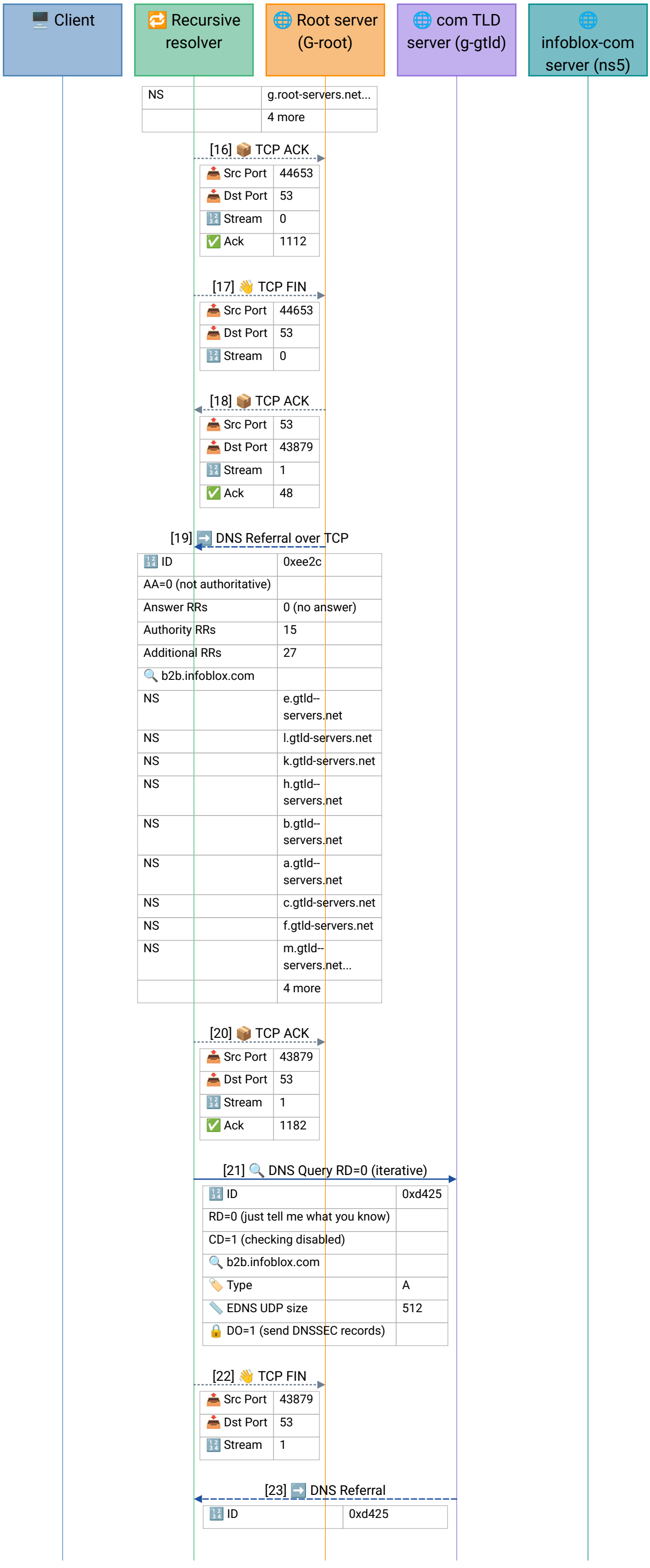
[Frame 13] The `A` query for `b2b.infoblox.com` over TCP: ID `0xee2c`, 47 bytes = 2-byte length + 45-byte message. Still `RD=0` and `CD=1`.

[Frame 14] The root acknowledges the 30-byte priming query (Ack 31).

[Frame 15] The full priming answer, 1,109 bytes, now that TCP lifts the size limit.

- `AA=1`: the root is authoritative for this data.
- 14 answers: the 13 root server names, a through `m.root-servers.net`, plus an `RRSIG` signing that NS set.
- 27 additional records: an `A` and an `AAAA` address for each server, plus OPT.
- TTL 518,400 s (6 days).

This is where the capture proves our root server is G-root: the `AAAA` record for `g.root-servers.net` is `2001:500:12::d0d`.



[Frame 16] The resolver acknowledges the 1,111-byte TCP segment (Ack 1112).

[Frame 17] With its answer in hand, the resolver closes connection 44653 at once. Because it closes first, the resolver, not the root server, carries the TCP TIME_WAIT state.

[Frame 18] The root acknowledges the 47-byte A query (Ack 48).

[Frame 19] The referral. No answer (`AA=0`, 0 answer records), but a pointer: "I don't know `b2b.infoblox.com`, but `com` is delegated to these servers, and here are their addresses."

- 15 authority records: 13 NS records naming the `com` servers, a through `m.gtld-servers.net`, then a DS record and an RRSIG over it.
- 27 additional records: A and AAAA glue for the 13 servers, plus OPT.

The DS (key tag 19718, algorithm 13, SHA-256 digest) is the root vouching for `com`'s signing key: one link in the DNSSEC chain of trust. The RRSIG (algorithm 8, key tag 61809) covers only the DS; a parent never signs the NS set at a delegation point.

TTLs: 172,800 s (2 days) for the NS and glue, 86,400 s for the DS.

[Frame 20] The resolver acknowledges the 1,181-byte referral (Ack 1182).

[Frame 21]

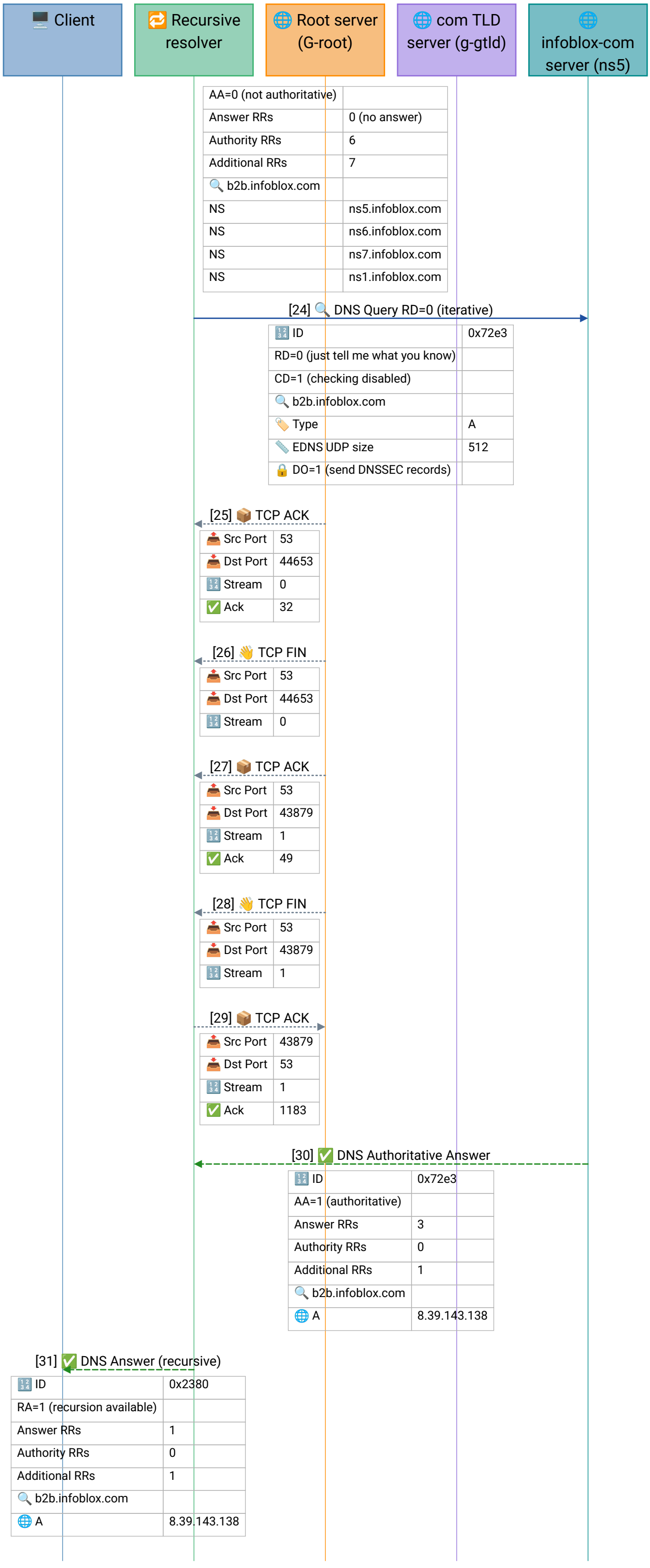
Phase 4: the com referral

0.6 ms after the referral, the resolver asks `g.gtld-servers.net` (`2001:503:eea3::30`, from frame 19's glue), one of the 13 `com` servers.

Same question, full name again, `RD=0`, back over UDP with 512 bytes offered. New ID (`0xd425`) and source port (41447).

[Frame 22] The resolver closes the second root connection (43879) now that its referral has arrived.

[Frame 23] The `com` server answers in 7.9 ms, the fastest round trip of the lookup, with another referral: `AA=0`, no answers.



- 6 authority records: 4 NS records for `infoblox.com` (`ns5`, `ns6`, `ns7`, `ns1`), a DS (key tag 19717, algorithm 8), and an `RRSIG` over the DS (algorithm 13, signed by `com`, key tag 46539).
- 7 additional records: glue for `ns5` (`AAAA` + `A`), `ns6` (`A`), `ns7` (`A` + `AAAA`), `ns1` (`A`), plus OPT.

This glue is essential, not a convenience. The name servers live inside the zone they serve, so without their addresses here the resolver could never reach them.

At 384 bytes, this referral fits in 512 bytes and needs no TCP.

[Frame 24]

Phase 5: the authoritative answer

The resolver asks `2600:1f14:3515:8202:da8:7299:3926:a2c1`, which frame 23's glue names as `ns5.infoblox.com`. Only `ns5` and `ns7` came with IPv6 glue, and this resolver uses IPv6 for every upstream query.

Same question, `RD=0`, ID `0x72e3`.

[Frame 25] Meanwhile, the root servers finish the TCP teardowns. The root acknowledges the resolver's FIN from frame 17.

[Frame 26] The root's own FIN on connection 44653. Watch this one: the capture shows no ACK from the resolver, so the root sends it again in frame 32.

[Frame 27] The root acknowledges the resolver's FIN from frame 22.

[Frame 28] The root's FIN on connection 43879.

[Frame 29] The resolver acknowledges it, and connection 43879 is fully closed.

[Frame 30] The answer, at last. `ns5` replies after 65.6 ms, the slowest single round trip of the lookup, with `AA=1`: this server owns the zone, so its word is final.

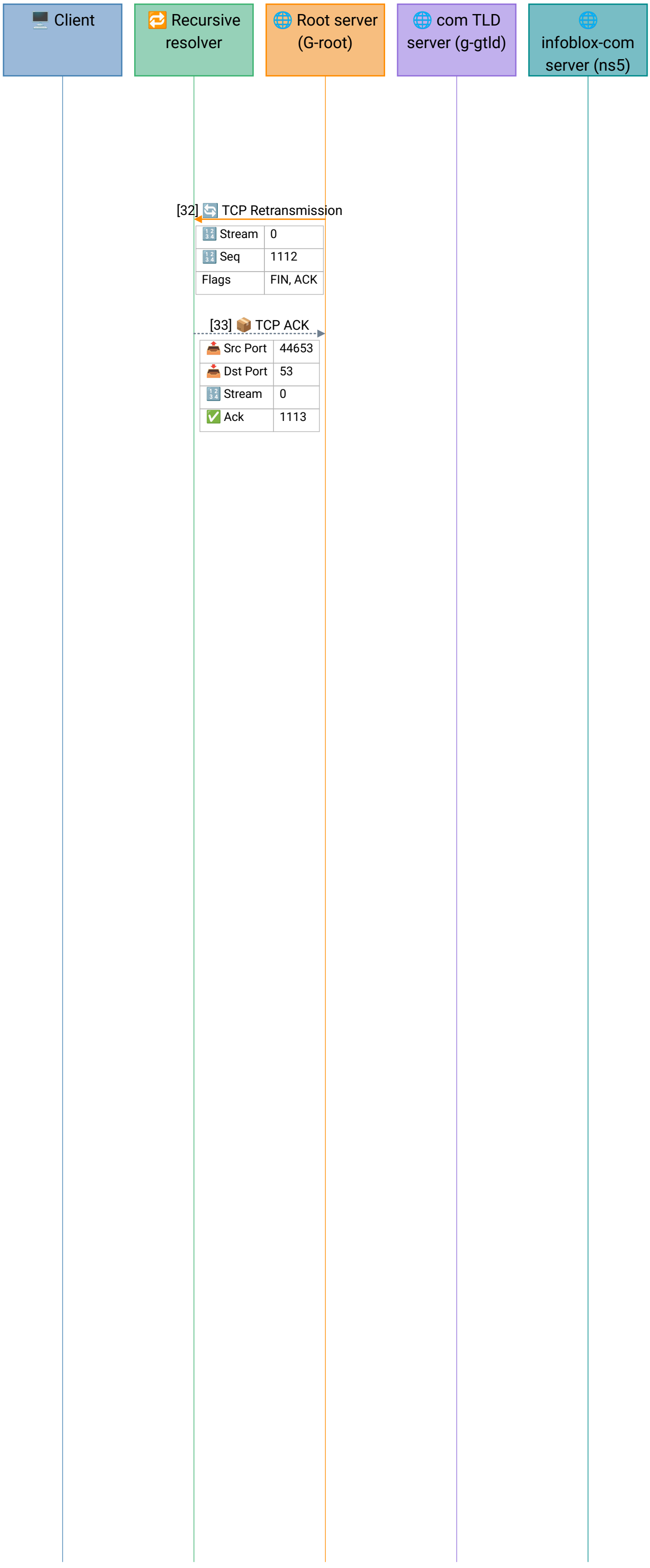
- 3 answer records: the `A` record `8.39.143.138` and two `RRSIG`s over it, both algorithm 8, but from two different keys (key tags 5349 and 33823).
- TTL 10 s: any resolver may cache this address for only 10 seconds, while the delegations above it last for days.

[Frame 31]

Phase 6: the client gets its answer

0.4 ms later the resolver answers the client, 159 ms after the question. ID `0x2380` matches frame 1.

- `RA=1`: this server offers recursion. `AA=0`: it is relaying the answer, not authoritative for it.
- Only the `A` record comes back. The client did not set `DO`, so the `RRSIG`s are dropped.
- `AD=0`: the resolver does not claim the answer was validated. Validation would need the `DNSKEY` sets for the



root, `com`, and `infoblox.com`, and the capture contains no DNSKEY queries.

- The resolver echoes the client cookie and adds its own server cookie.

From the client's side, the whole lookup was two packets. The other 31 frames happened behind the resolver.

[Frame 32] One second after frame 26, the root retransmits its FIN for connection 44653: it never saw an ACK. The capture does not show why the resolver stayed silent. None of this affects the client, which got its answer 950 ms earlier.

[Frame 33] The resolver acknowledges the retransmitted FIN (Ack 1113), and connection 44653 is finally closed.