DNS Transport over TCP - Implementation Requirements

J. Dickinson, Sinodun Internet Technologies
R. Bellis, Nominet
A. Mankin and D. Wessels, Verisign Labs

DNS Transport over TCP

- This is a -bis of RFC5966
- Aim of draft is to put TCP on the same footing as UDP for use as a DNS transport
- In support of
  - Privacy efforts
  - Preventing amplification attacks
  - Packet size limitations

DNS Transport over TCP

- Major changes in -bis include:
  - DNS implementations are recommended not only to support TCP but to support it on an equal footing with UDP
  - DNS implementations are recommended to support reuse of TCP connections
  - DNS implementations are recommended to support pipelining and out of order processing of the query stream
  - A non-normative discussion of use of TCP Fast Open

Connection Handling

- One perceived disadvantage to DNS over TCP is the added connection setup latency, generally equal to one RTT.
  - Both clients and servers SHOULD support connection reuse by sending multiple queries and responses over a single TCP connection.
- DNS currently has no connection signalling mechanism. Clients and servers may close a connection at any time. Clients MUST be prepared to retry failed queries on broken connections.

Connection Handling

- To mitigate the risk of unintentional server overload, it is RECOMMENDED that for any given client - server interaction there SHOULD be no more than one connection for
  - regular queries [One for each client application]
  - one for zone transfers
  - one for each protocol that is being used on top of TCP, for example, if the resolver was using TLS.

Query Pipelining

- In order to achieve performance on par with UDP, it is RECOMMENDED that DNS clients pipeline their queries.
  - Do not wait for an outstanding reply before sending the next query.
- DNS servers SHOULD expect to receive pipelined queries. The server should process TCP queries in parallel, just as it would for UDP.

Query Pipelining

- Authoritative servers and recursive resolvers are RECOMMENDED to support the sending of responses in parallel and/or out-of-order, regardless of the transport protocol in use.
- Stub and recursive resolvers MUST be able to process responses that arrive in a different order to that in which the requests were sent, regardless of the transport protocol in use.
- Recursive resolvers SHOULD process TCP queries in parallel and return individual responses as soon as they are available, possibly out-of-order.

TCP Fast Open

- This section is non-normative.
- TCP fastopen [I-D.ietf-tcpm-fastopen] (TFO) allows data to be carried in the SYN packet.
  - It saves up to one RTT compared to standard TCP.
- Currently Linux only. 3.16.0 added IPv6 support.

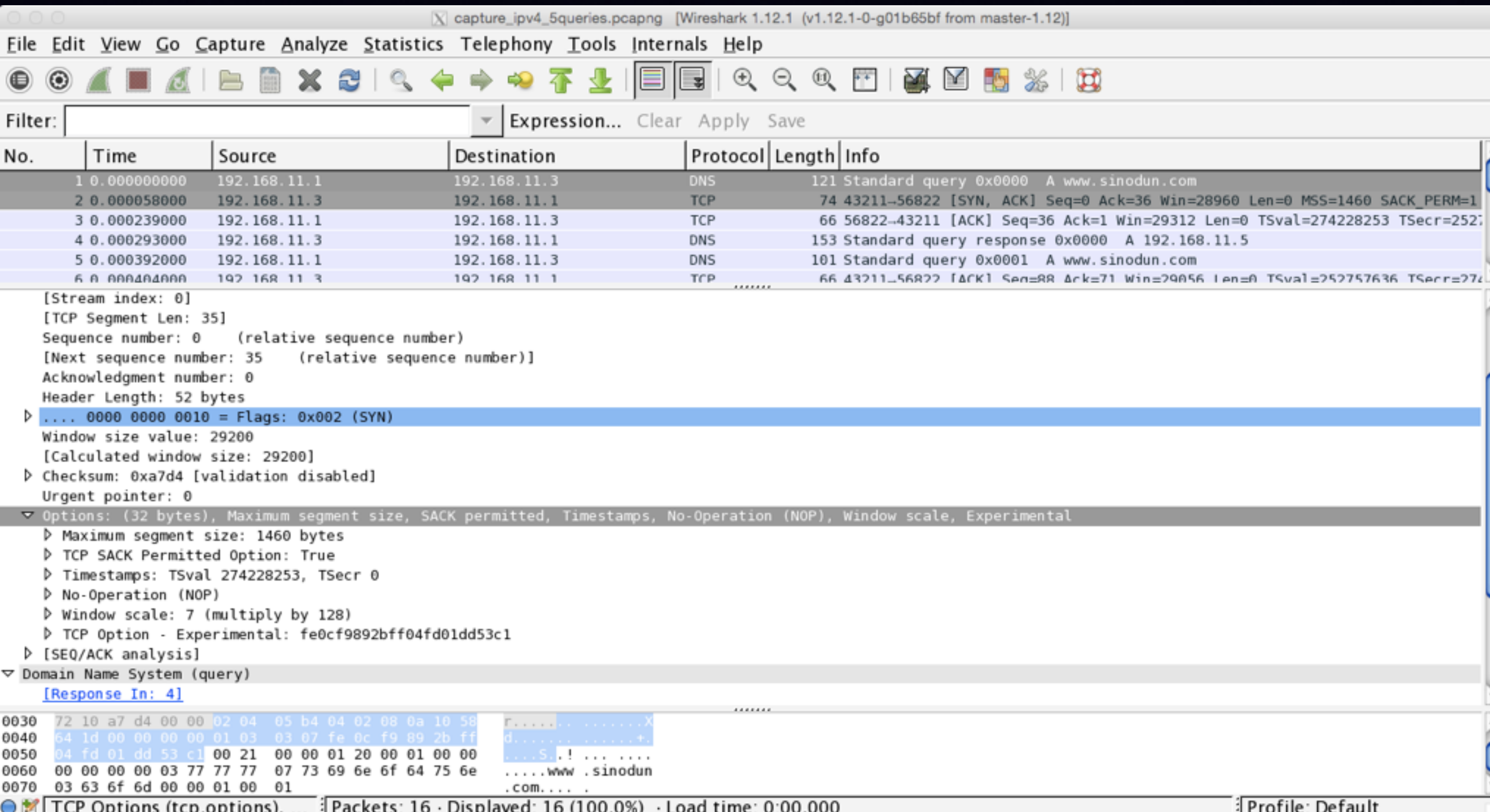
TCP Fast Open

- TFO Code changes
  - On the client, the call to connect() is replaced with a TFO aware version of sendmsg() or sendto().
  - On server, set a socket option between the bind() and listen() calls.

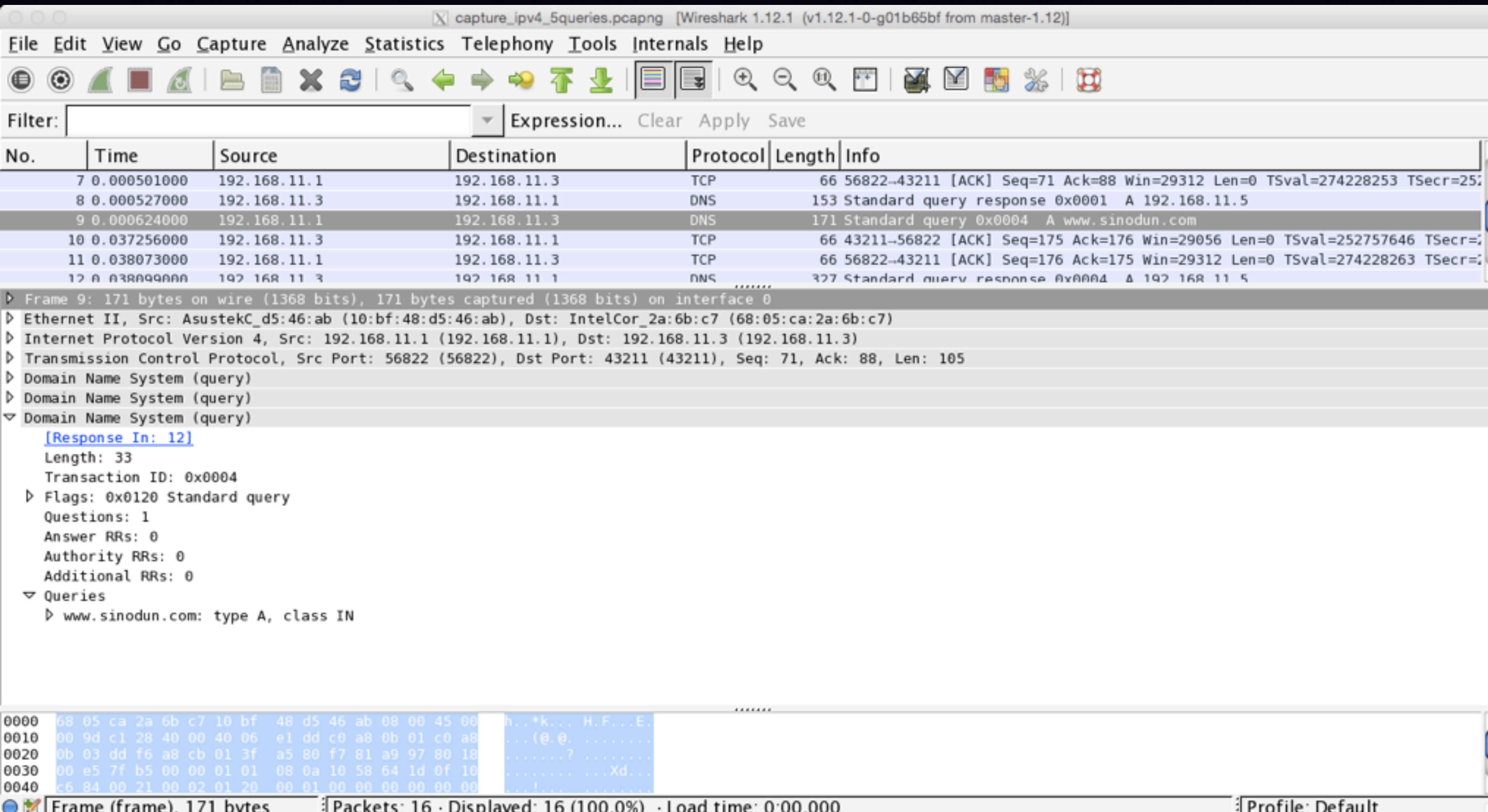
TCP Fast Open

- TFO kernel config
  - change the kernel parameter net.ipv4.tcp_fastopen (A bitmap)
    - 1= client
    - 2 = server

TCP Fast Open - query



TCP Pipelining
Multiple queries in one packet



DNS Transport over TCP

- We are seeking adoption of this draft…