DNS Zone Transfer using DNS Stateful Operations (XuD)

draft-zatda-dprive-xfr-using-dso

Han Zhang
Pallavi Aras
Willem Toorop
Sara Dickinson
Allison Mankin
Matthijs Mekking

XuD - Background

Why XuD?

- XoT solves major problems for **privacy** of zone transfers, but still depends on bi-directional exchange for NOTIFY triggered IXFRs
- XoT is still therefore a polling mechanism for IXFR, but there is potential to use a **subscribe/publish mechanism** for zone updates.

What is XuD?

- DNS Zone transfer encryption using DNS Stateful Operations (DSO) [RFC8490]

What are DNS Stateful Operations?

- **RFC8490 - DSO Basics**
    - Communicate **operations within persistent stateful sessions** (TCP/TLS)
    - DSO uses a **new OPCODE**
    - New message format - uses **Type Length Value** (TLV) syntax (not RRs)

- **RFC8490 Defines 3 TLVs:**
    - Keepalive (specifies the Keepalive Interval and Inactivity Timeout)
    - Retry Delay (close the connection/operation failed & don't retry for X ms)
    - Encryption Padding (equivalent to EDNS0 padding [RFC7830])

- Other TLVs already defined on other drafts….

More on DNS Stateful Operations

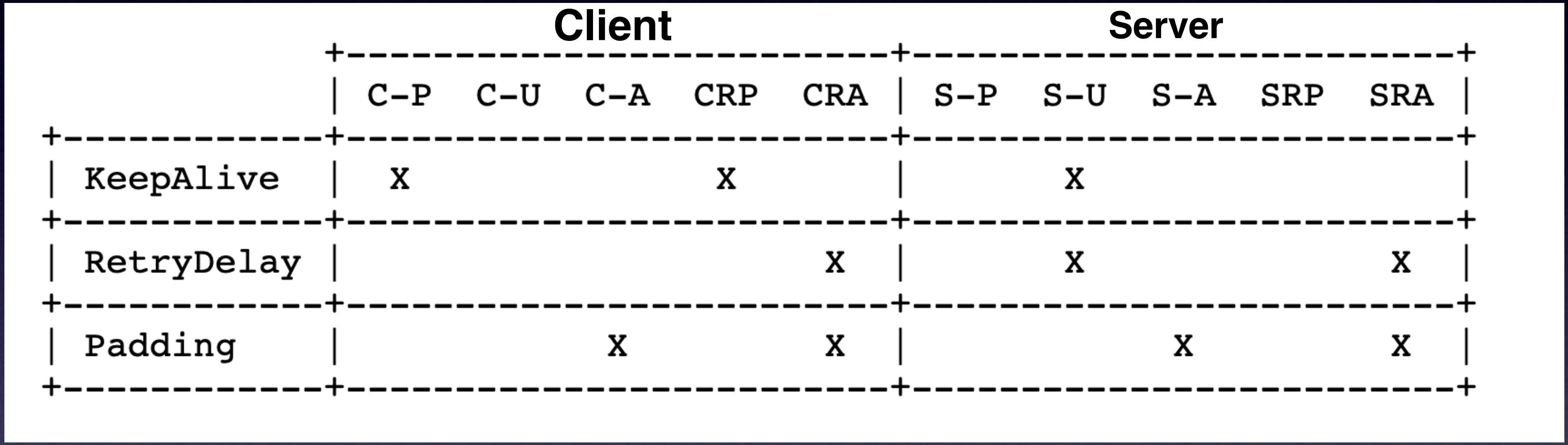
- **DSO Sessions (RFC8490)**
  - Client sends **Keepalive DSO** message to signal support, server acknowledges
  - After that ‘DSO Session’ rules apply (NOT RFC7766 rules)
    - Normal DNS message exchange can take place
- **DSO message types**
  - DSO messages (require a response)
  - DSO uni-directional messages (that don’t)
- **DSO message exchange**
  - Either client or server can initiate DSO messages
  - DSO TLV’s can be Primary or Additional (>1 per message)

More on DNS Stateful Operations

- **DSO Sessions (RFC8490)**
  - Client sends **Keepalive DSO** message to signal support, server acknowledges
  - After that ‘DSO Session’ rules apply (NOT RFC7766 rules)
    - Normal DNS message exchange can take place
- **DSO message types**
  - DSO messages (require a response)
  - DSO uni-directional messages (that don’t)
- **DSO message exchange**
  - Either client or server can initiate DSO messages
  - DSO TLV’s can be Primary or Additional (>1 per message)

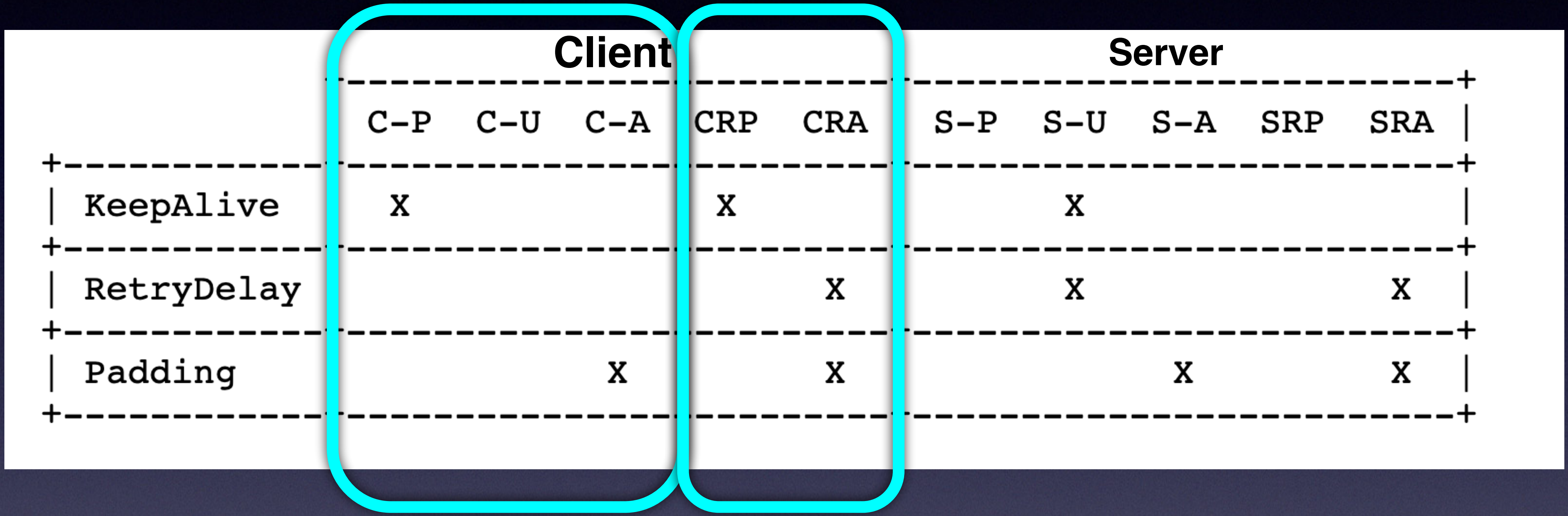
Specification of TLV defines usage

DSO TLV descriptions



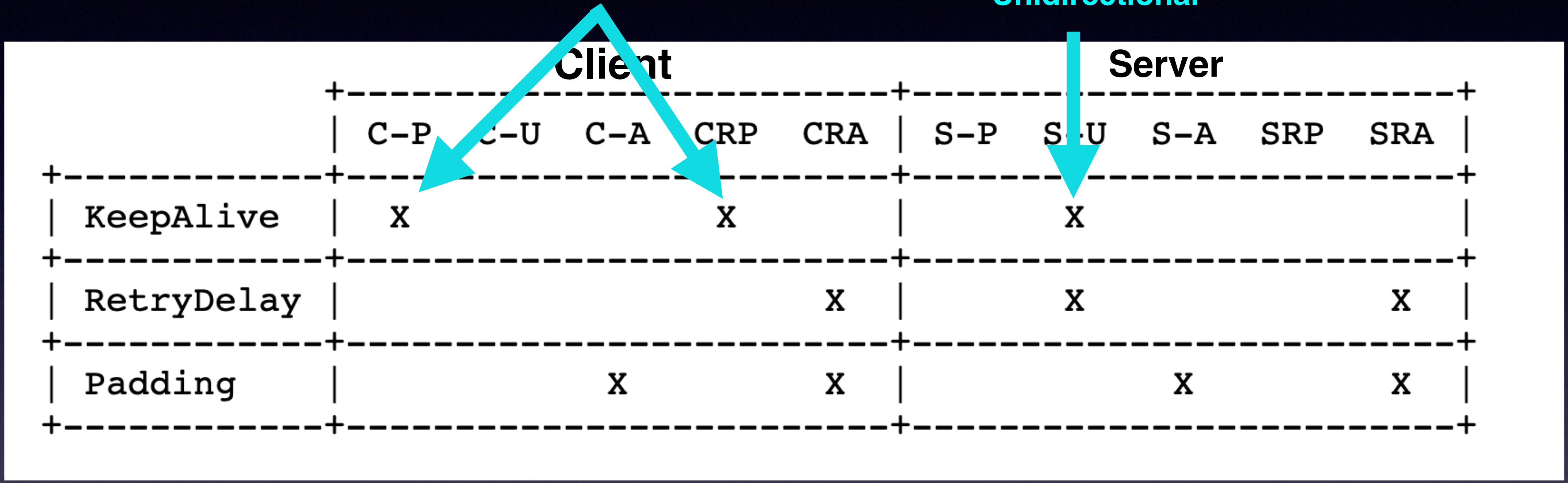
	Client					Server				
	C-P	C-U	C-A	CRP	CRA	S-P	S-U	S-A	SRP	SRA
KeepAlive	X			X			X			
RetryDelay					X		X			X
Padding			X		X			X		X

DSO TLV descriptions



	Client					Server				
	C-P	C-U	C-A	CRP	CRA	S-P	S-U	S-A	SRP	SRA
KeepAlive	X			X			X			
RetryDelay					X		X			X
Padding			X		X			X		X

DSO TLV descriptions



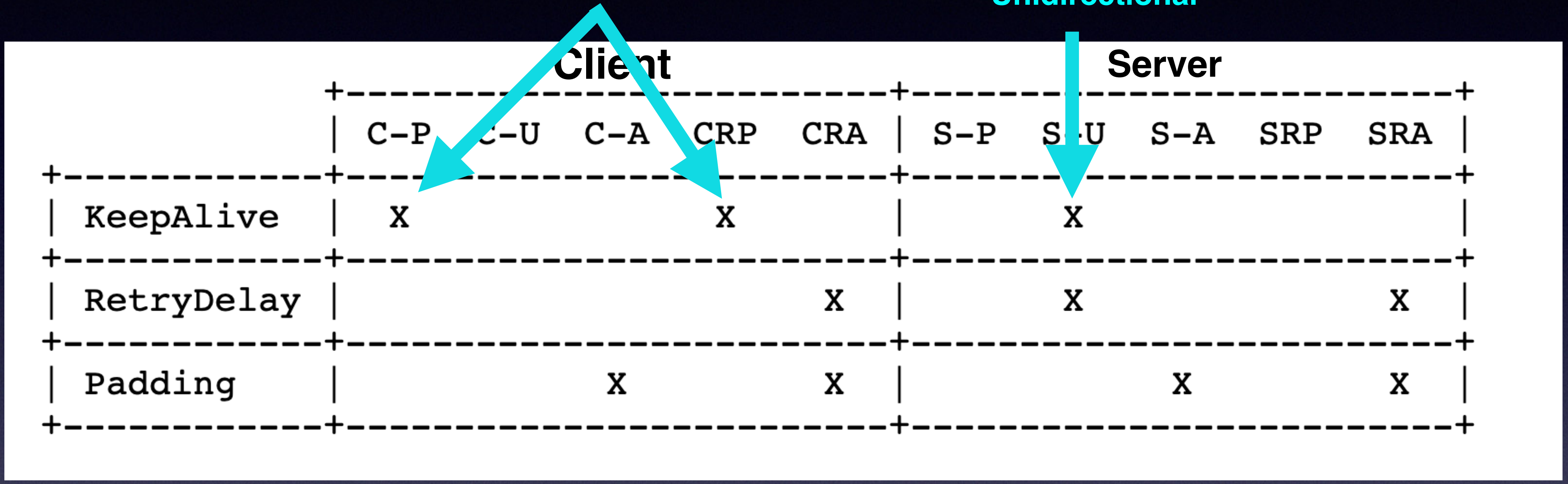
	Client					Server				
	C-P	C-U	C-A	CRP	CRA	S-P	S-U	S-A	SRP	SRA
KeepAlive	X			X			X			
RetryDelay					X		X			X
Padding			X		X			X		X

Message & response

Unidirectional

Always Additional

DSO TLV descriptions



Message & response
Unidirectional
Always Additional

	Client					Server				
	C-P	C-U	C-A	CRP	CRA	S-P	S-U	S-A	SRP	SRA
KeepAlive	X			X			X			
RetryDelay					X		X			X
Padding			X		X			X		X

- **Current uses of DSO that define other TLVs (DNSSD)**
  - **DNS Push Notifications**: "client to be asynchronously notified.. .of changes to DNS records.." (i.e. publish/subscribe model for particular RRsets)
  - **Discovery Proxy for Multicast DNSSD** & **Multicast DNS Discovery Relay**

DSO for XFR?

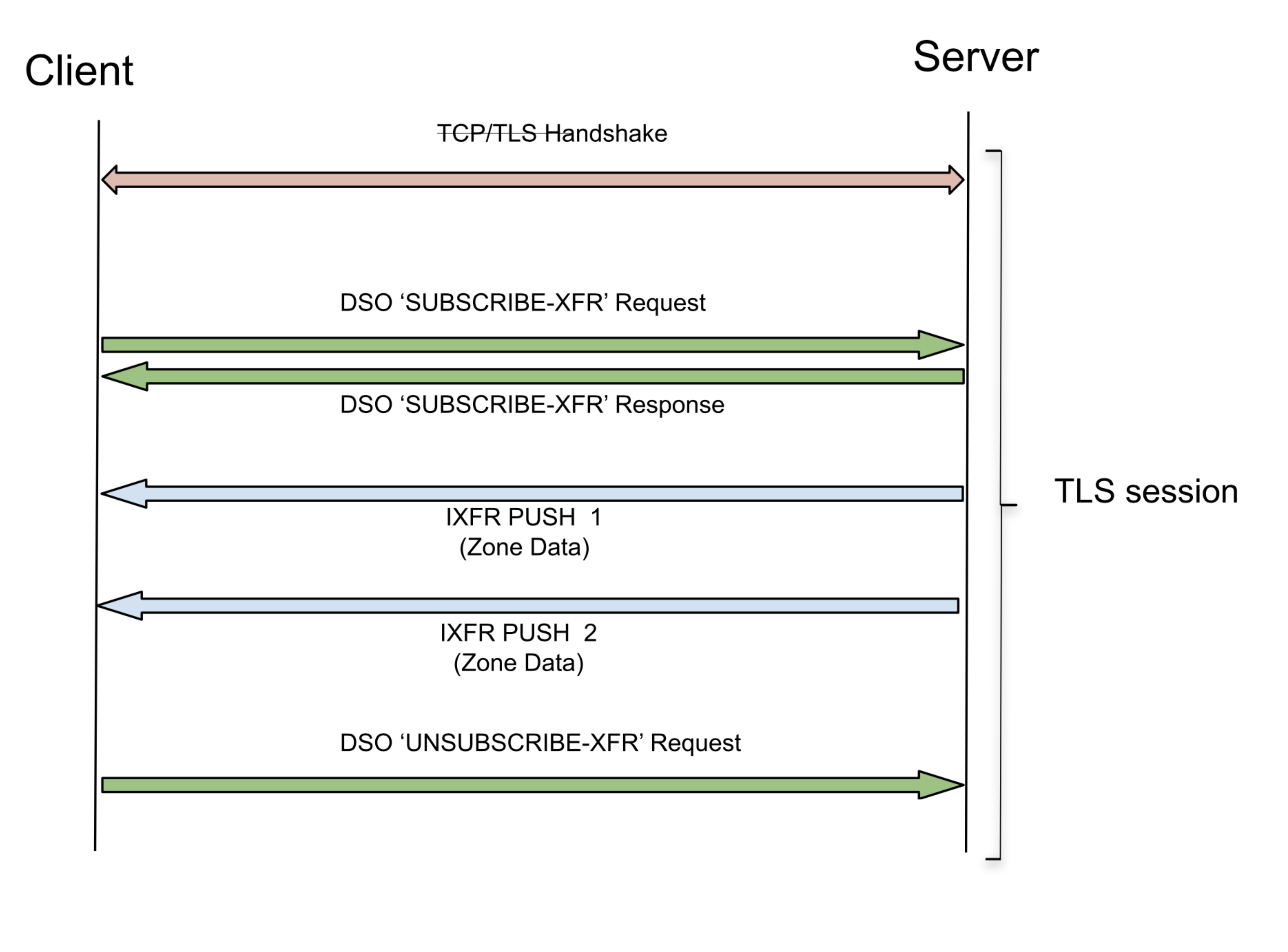
- **Build heavily on DNS Push Notifications concepts but**
  - Modify for publish/subscribe to zones

- **Use Cases (in addition to XoT)**
  - **Confidentiality** - DSO doesn't require TLS but specs using DSO can use it
  - **Confidentiality** - Eliminate NOTIFY/SOA (or do both within the DSO session)
  - **Security** - All queries/updates can occur on one connection (client initiated)
  - **Performance** - reduced number of messages
  - **Improved error handling** - define new, specific error codes
  - **Command channel** - potential to define server (primary) initiated commands…

XuD Data flow (simple)

Secondary

Primary



Client

Server

~~TCP/TLS H~~andshake

DSO ‘SUBSCRIBE-XFR’ Request

DSO ‘SUBSCRIBE-XFR’ Response

IXFR PUSH 1
(Zone Data)

IXFR PUSH 2
(Zone Data)

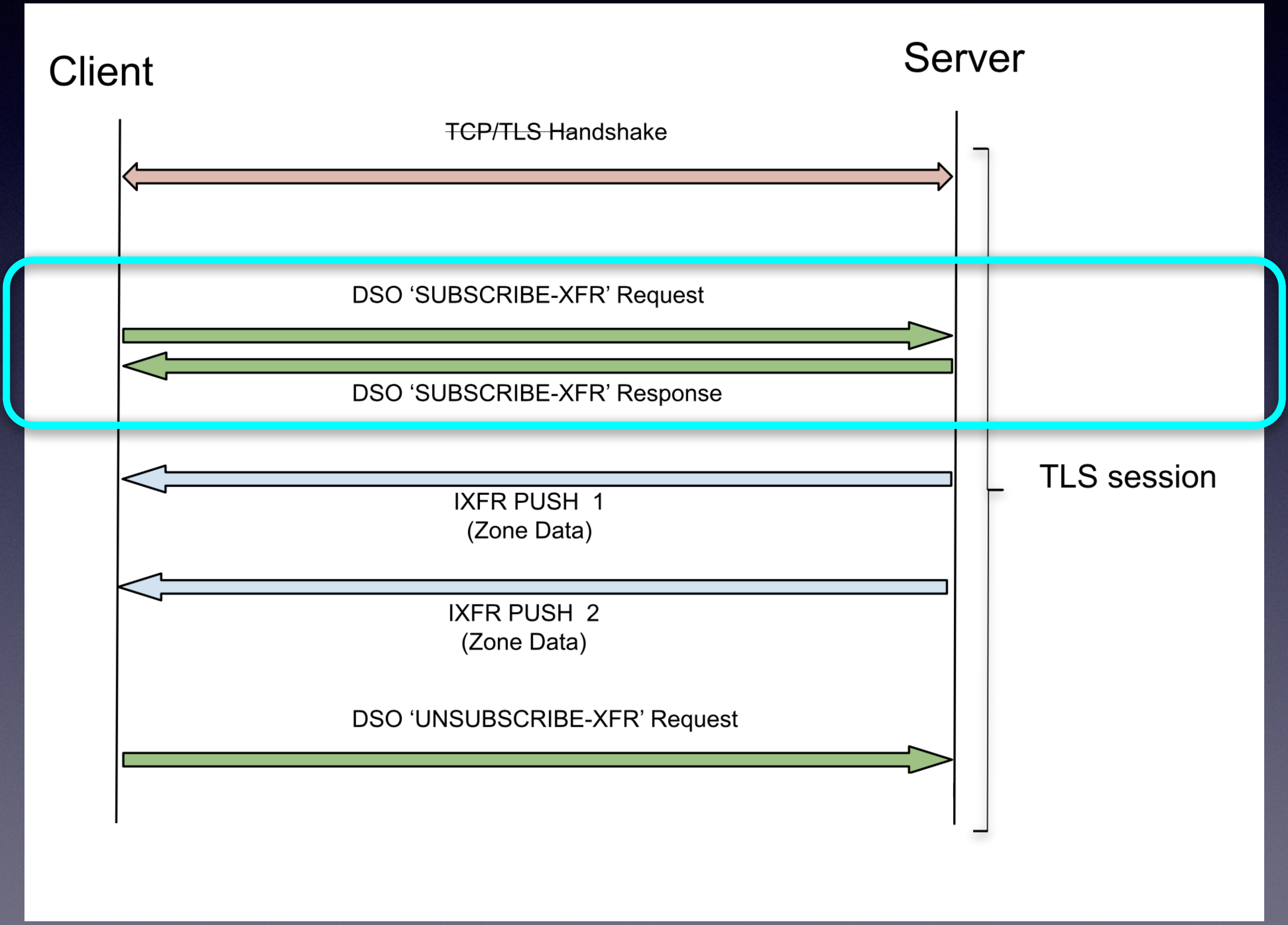
DSO ‘UNSUBSCRIBE-XFR’ Request

TLS session

XuD Data flow (simple)

Secondary

Primary

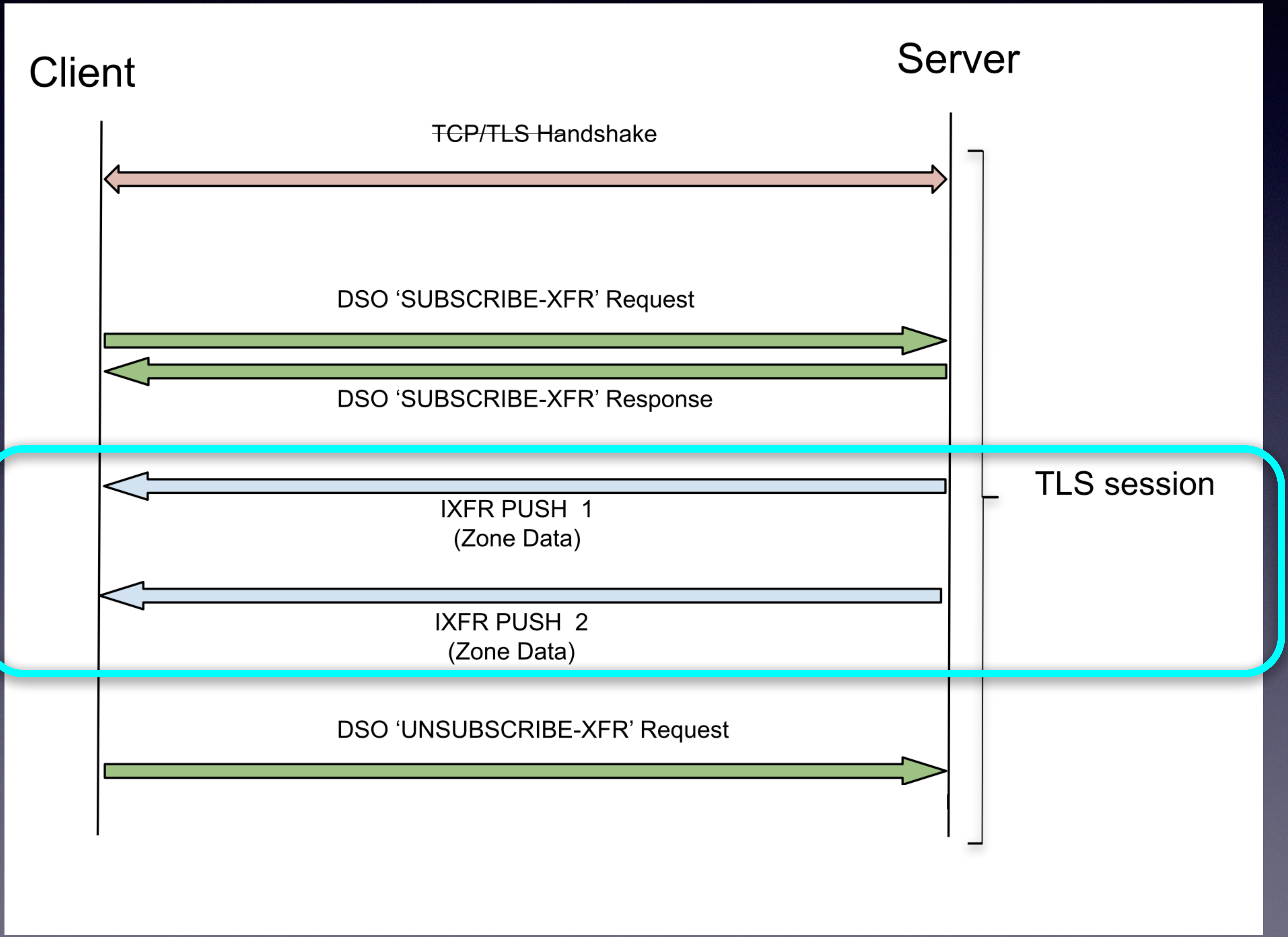


Client sends zone & SOA

XuD Data flow (simple)

Secondary

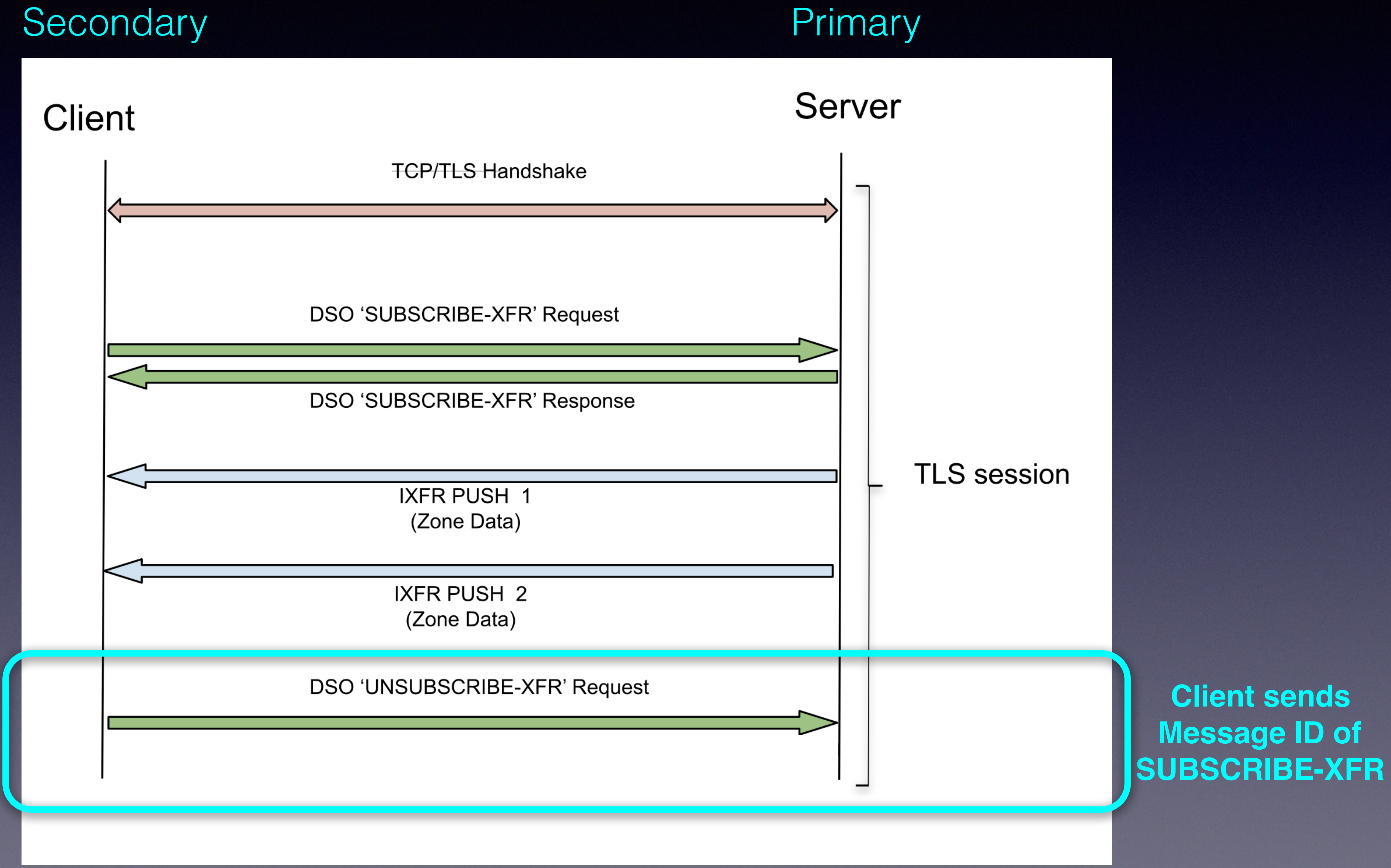
Primary



Client
Server
TCP/TLS Handshake
DSO ‘SUBSCRIBE-XFR’ Request
DSO ‘SUBSCRIBE-XFR’ Response
TLS session
IXFR PUSH 1
(Zone Data)
IXFR PUSH 2
(Zone Data)
DSO ‘UNSUBSCRIBE-XFR’ Request

Server pushes IXFRs as required

XuD Data flow (simple)



XuD characteristics

- **Specification details**
    - Server can **refuse a subscription** with e.g. NOTIMP, REFUSED, NOTAUTH
    - Clients can subscribe to **multiple zones on the same connection**
    - Client can request a **full zone transfer** by omitting the SOA in the SUBSCRIBE-XFR
    - Server can still send a **full zone transfer** if it can’t offer an incremental one
    - Clients can **unsubscribe and re-subscribe** for on the same connection
    - **Need a new TLV for TSIG** over a SUBSCRIBE-XFR request and DSO-IXFR

- **Implementation**
    - More complex to implement (A bigger delta on existing implementations than XoT)
    - No major open source authoritative implementations currently support DSO
    - But, cleaner data flow and naturally extensible/flexible

Open questions

- **Major: Current spec REQUIRES TLS. But TCP use case exists too…. (DNSOP?)**
    - Implementations MUST support XuD over TLS?
- **Minor**:
    - Should we support multiple zone in a single SUBSCRIBE-XFR request?
    - More signalling while subscriptions are active
        - Restart at a different SOA, or send an AXFR
        - What happens if SOA refresh timer expires?
    - Should there be a DSO-AXFR message defined?
    - Use case for master connecting to client?
    - Command channel uses: ‘stop serving zone’, ‘delete zone’
- **Is the WG interested in working on this?**