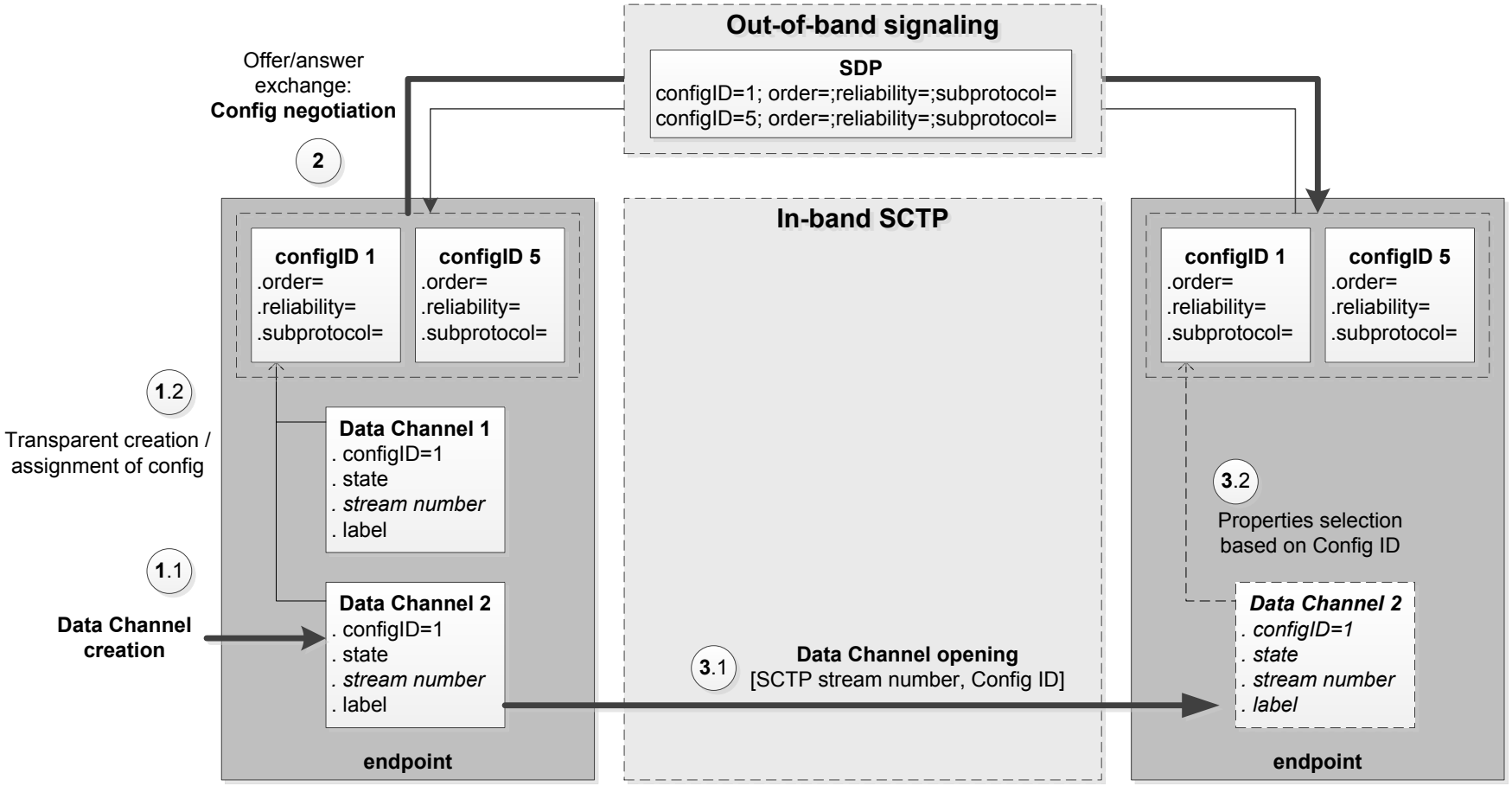
RTCWeb Data Channel Management

draft-marcon-rtcweb-data-channel-management-00

Problem:

- For best interoperability, data channel setup should be negotiated through SDP offer/answer always
- But then:
  - opening a data channel would each time trigger SDP renegotiation
  - opening a high number of data channels could cause out-of-band scalability problems
  - data channel setup would anyway be slower than an in-band data channel control protocol

Solution: Figure



Solution (1/3)

Concept of "data channel configuration"

- A set of data channel properties that can be shared by multiple data channels
  - order of delivery, reliability, subprotocol, [PPID ?]
- Each property is generic : its name/value does not matter
- When at least one property name or value differs, the configuration differs

1. Creating a data channel [& configuration]

- The application handles individual data channels and is unaware of configurations
- The application creates a data channel "as usual" and provides as distinct arguments: 'configuration properties' and 'data channel-specific properties'
- The agent detects if the passed configuration properties match an existing configuration
- If not, the agent creates a new configuration and triggers SDP renegotiation
- The resulting (existing or new) configuration ID is assigned to the data channel
- The data channel has no SCTP stream number yet
- Relationships: Configuration → 0..N Data Channels; Data Channel → 1 Configuration

Solution (2/3)

2. Negotiating out-of-band the data channel configurations

- negotiation only triggered when the app creating a data channel happens to create a new configuration
- the m-line describing the SCTP association is followed by 1 attribute per configuration
  - configuration ID, followed by configuration properties (order, reliability, subprotocol…)
- negotiated config properties could be assymetric in offer/answer
- *(to be deleted: the peer must create one data channel per new offered configuration)*

3. Opening data channels in-band

- the SCTP stream number is chosen by the first endpoint sending a msg on the channel
- no properties are carried in-band, only the configuration ID
  - so only data channels with properties matching a negotiated config can be created
- an endpoint opens a data channel by signaling "somehow" to the peer: {SCTP stream number, configuration ID}
- the peer on its side opens either an existing data channel (with same config ID but no SCTP stream number) or a new data channel
- the peer applies to the data channel the properties given by Configuration ID

Solution (3/3)

METHODS for signaling in-band the data channel info { SCTP stream number, configuration ID}

- **METHOD detailed in draft 00**
  - –The agent includes in each user message a 2-byte header signaling: Configuration ID and (then why not) data framing type - text/binary
  - –A data channel is opened by the first application message received on this channel

- **OTHER METHODS** (avoiding data framing issues discussed on the list)
  - –A convention by which the peer can infer Configuration ID from SCTP Stream Identifier, like:
    - The 16-bit SCTP Stream Identifier is split into: Configuration ID (N bits) and Data Channel Instance ID (M bits)
    - The SCTP association is initialized with 65535 inbound & outbound streams
  - –An in-band protocol sending the [list of] [SCTP stream number &] configuration ID to the peer

Benefits & Issues

BENEFITS

- Interoperability ensured by SDP Offer/Answer exchange
- Extensibility of config properties
  - when a new property is added, no need to extend an in-band protocol
  - and no change of processing as properties are generic
- Possible assymetry of offer/answer negotiated properties
- Scalability of out-of-band negotiation
  - small SDP footprint, even for a high number of data channels (of same usage)
  - SDP renegotiation rarely required
- Scalability of in-band data channel creation
- Deterministic selection of properties for data channels created in-band
- No/minimal impact on existing API

ISSUES

- Properties unique to a data channel cannot be transmitted to the peer
  - e.g. labels