NETCONF Transaction ID
and related drafts

2023-03-27

IETF 116

Jan Lindblad <jlindbla@cisco.com>

Roque Gagliano <rogaglia@cisco.com>

Transaction ID

draft-ieft-netconf-transaction-id-00

draft-ieft-netconf-transaction-id-00 changes (vs. draft-lindblad-…-02)

- Changed the logic around how txids are handled in the candidate datastore
- Changed the logic of copy-config to be similar to edit-config
- Clarified how txid values interact with when-dependencies together with default values

- Added content to security considerations
- Added a high-level example for YANG-Push subscriptions with txid
- Updated language about error-info sent at txid mismatch in an edit-config
- Some rewording and minor additions for clarification, based on mailing list feedback

New Txid Candidate Datastore Behavior

Consider a <get-config> towards :candidate. Most recent txid="3456"

```
<get-config txid="?"/>
```

The :candidate may hold modified elements (vs. :running)

- Some implementations may know before <commit> what the new txid would be if candidate was committed. They can return this, future, txid.

```
<data txid="4711">
    <some-leaf txid="4711">updated-value</some-leaf>
```

- Some implementations may not know. They can return txid "!". This indicates the txid will change at commit.

```
<data txid="!">
    <some-leaf txid="!">updated-value</some-leaf>
```

Next Steps

- Implementation experience

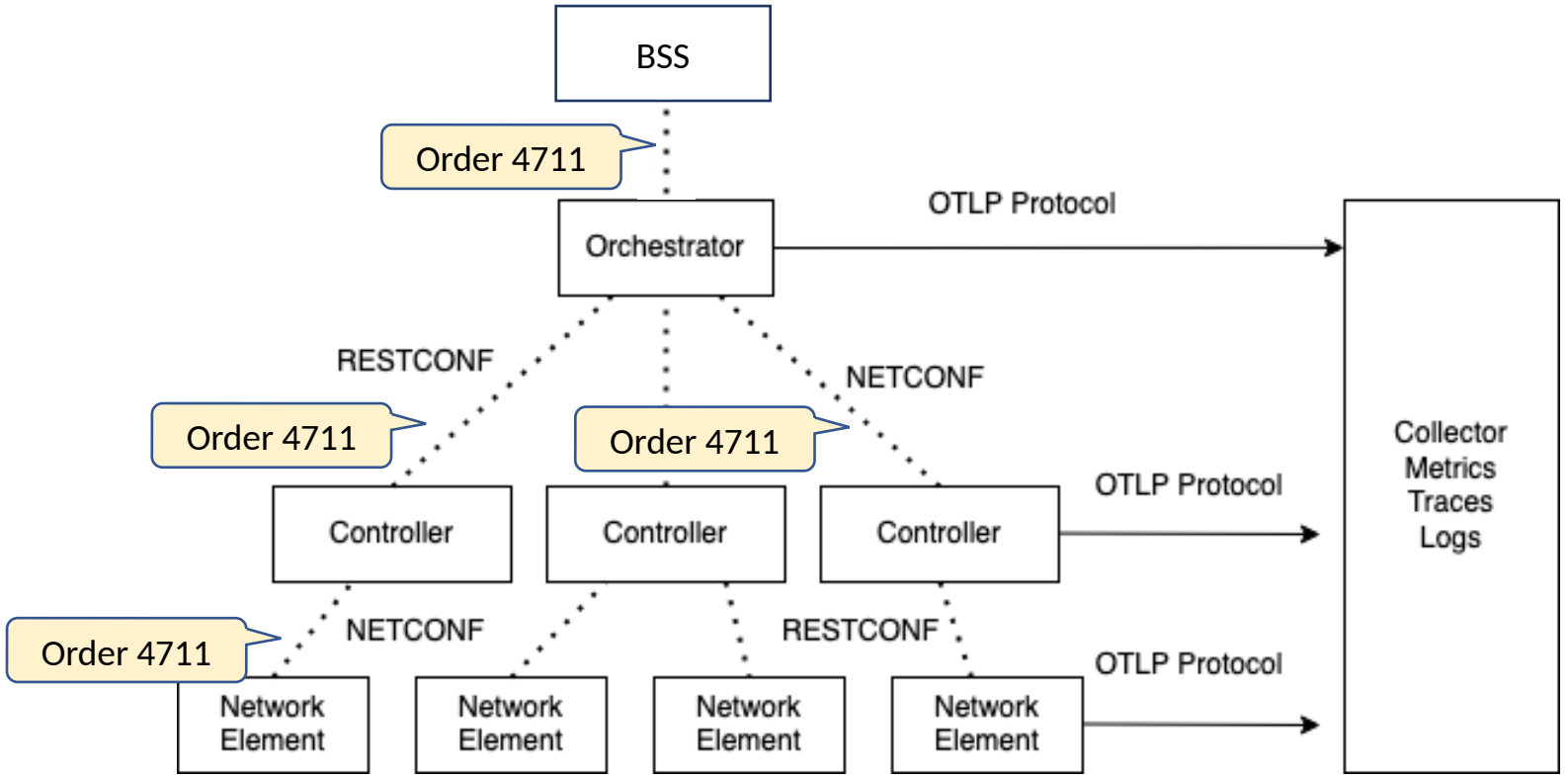
Trace Context Extension

draft-rogaglia-netconf-trace-ctx-extension-02

draft-rogaglia-netconf-trace-ctx-extension-02 changes (vs. -00)

- Added Implementation example 2: YANG DataStore to facilitate emphasize complementary with other drafts
- Added new use case: Billing and auditing
- Improved text related to: [I-D.ietf-netconf-transaction-id]
- Added Error Handling initial section
- Added how to manage versioning by defining YANG modules for each traceparent and tracestate versions as defined by W3C
- Added 'YANG Module Names' to IANA Considerations

Decorating RPCs with Trace IDs



Collector apps can:

- Correlate resource usage on all levels to individual customers and orders
- Trace the effects of service input through all layers, for billing, debugging or forensic purposes

Format Defined by W3C

https://www.w3.org/TR/2021/REC-trace-context-1-20211123/

Headers directly usable in RESTCONF:

```
traceparent: 00-4bf92f3577b34da6a3ce929d0e0e4736-00f067aa0ba902b7-01
tracestate: rojo=00f067aa0ba902b7,congo=t61rcWkgMzE
```

A NETCONF client might send this:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1"
     xmlns:w3ctc="urn:ietf:params:xml:ns:netconf:w3ctc:1.0"
     w3ctc:tracestate="rojo=00f067aa0ba902b7,congo=t61rcWkgMzE"
     w3ctc:traceparent=
         "00-4bf92f3577b34da6a3ce929d0e0e4736-00f067aa0ba902b7-01">
  <edit-config>…</edit-config>
</rpc>
```

W3C's got this

- Initially, trans-id-00 draft defined NETCONF attributes for this purpose (but only for config operations)
- Since then, W3C defined their HTTP trace headers for all operations (GET/PUT/PATCH/POST/DELETE)
- W3C have tracing solution vendors on board

Þ We expect straight-forward integration with mainstream tools

if we base our solution on W3C's headers and encoding

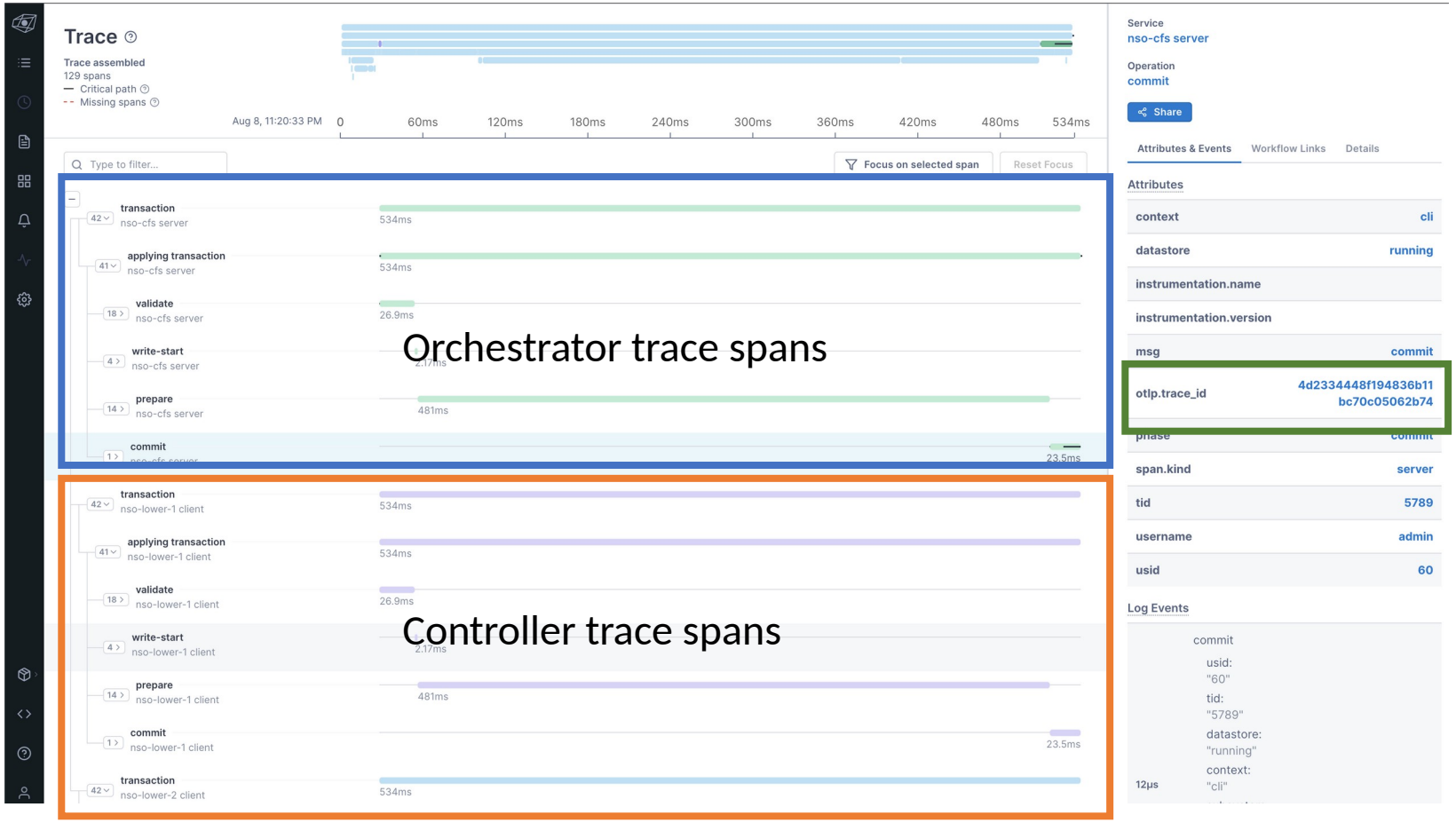
The end goal is to tap into OTLP ecosystem
Example with ServiceNow/LightStep backend

Orchestrator

<edit-config>
trace-context

Controller

OTLP Backend



Same trace-id

Objective: register NETCONF and RESTCONF at trace Context registry

ref: https://www.w3.org/TR/trace-context-protocols-registry/

§ 3. Registry

This section is the registry of identified formats of trace context [TRACE-CONTEXT] serialization and deserialization for protocols.

Protocol	Public Specification(s)	Requestor Contact
HTTP	[TRACE-CONTEXT]	W3C
Binary	Trace Context: binary protocol	W3C
[AMQP]	Trace Context: AMQP protocol	W3C
[MQTT]	Trace Context: MQTT protocol	W3C

Add RESTCONF and NETCONF to this list!

Next Steps

- More details around error handling
- Publish RESTCONF document “should” implement W3C headers
- Publish “Baggage” draft
- Figure out process for W3C protocol registration
- Implementation experience

Overview of drafts related to Transaction ID

Overview of Related Drafts

TRANS-ID	draft-**ietf**-netconf-transaction-id-**00**
CFG-TRACE	draft-quilbeuf-opsawg-configuration-tracing-**01**
W3C-TRACE	draft-rogaglia-netconf-trace-ctx-extension-**02**
PRIV-CAND	draft-jgc-netconf-privcand-**01** (not yet published)
ETAGS	RFC 8040 (RESTCONF)

Massive Use Case Overlaps

* = Not in recent versions

	TRANS-ID	CFG-TRACE	W3C-TRACE	PRIV-CAND	ETAGS
Increase transaction throughput by reducing lock time	✔	-	-	✔	-
Allow clients to get config changes at top level or within subtree ("Sync")	✔	Assumed in other doc	-	-	✔
Allow clients to make config changes conditional on no conflicts ("No overwrite")	✔	Assumed in other doc	-	Maybe	✔
Allow clients to recognize their own echo in YANG Push updates	✔	-	-	-	-
Map transaction ids from client to server and server controlled children	*	✔	✔	-	-
Finding Source of configuration mistakes	*	✔	✔	-	-
Detecting conflicting intents	*	✔	✔	Maybe	-
Provisioning root cause analysis	*	✔	✔	-	-
System performance profiling	*	✔	✔	-	-
Billing and auditing	*	✔	✔	-	-

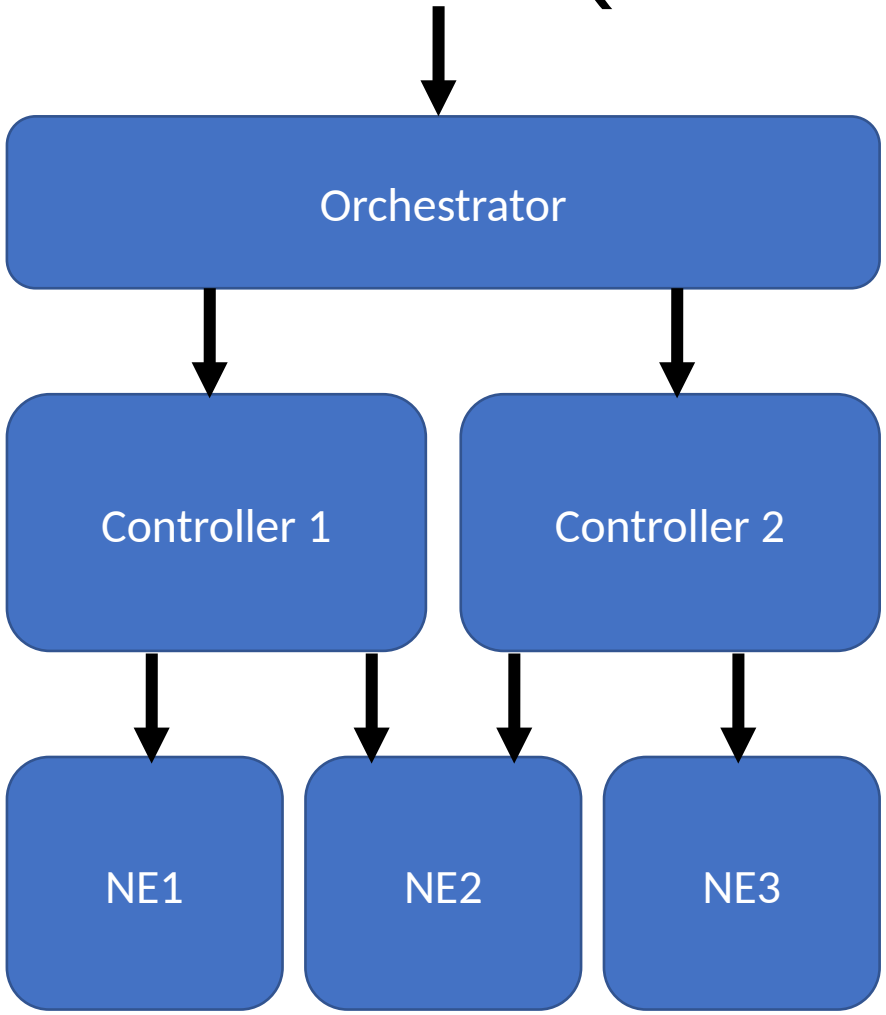
… but not much overlap in functionality

Great overlap in use cases + not much overlap in functionality

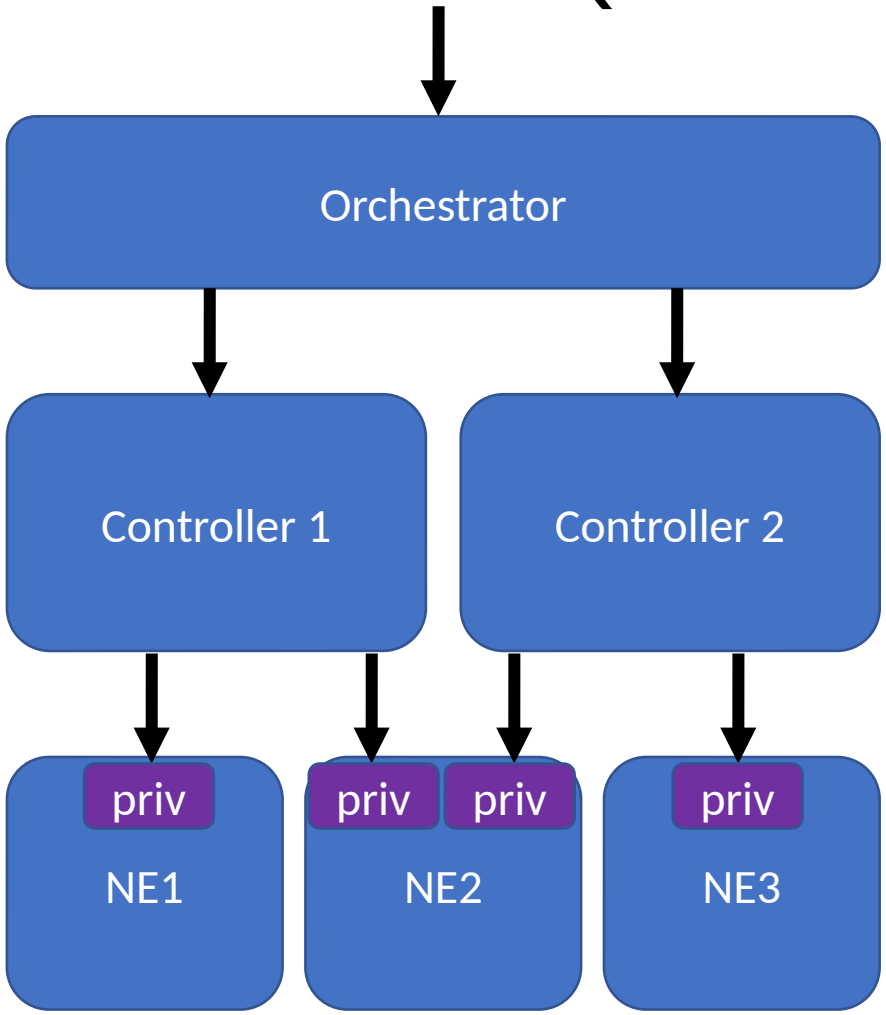
=>

Drafts could be made into a coherent framework

Drafts are (mostly) Complementary

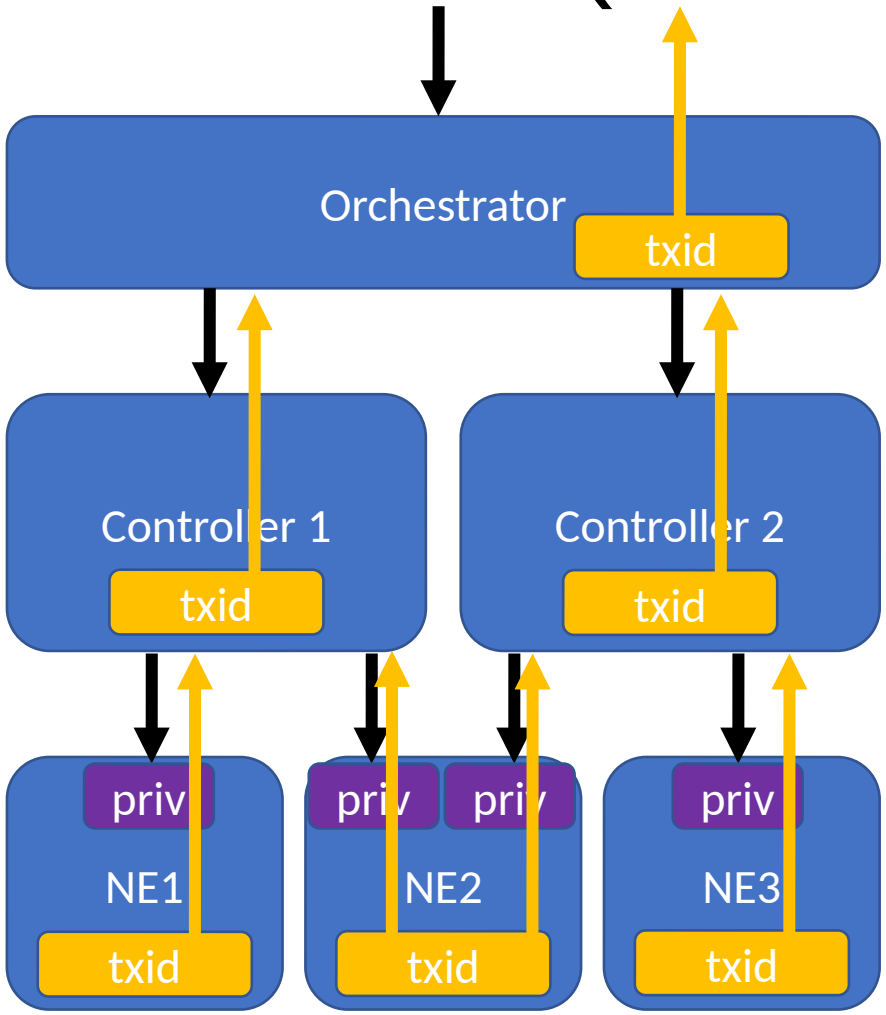


Drafts are (mostly) Complementary



draft-jgc-netconf-privcand-00

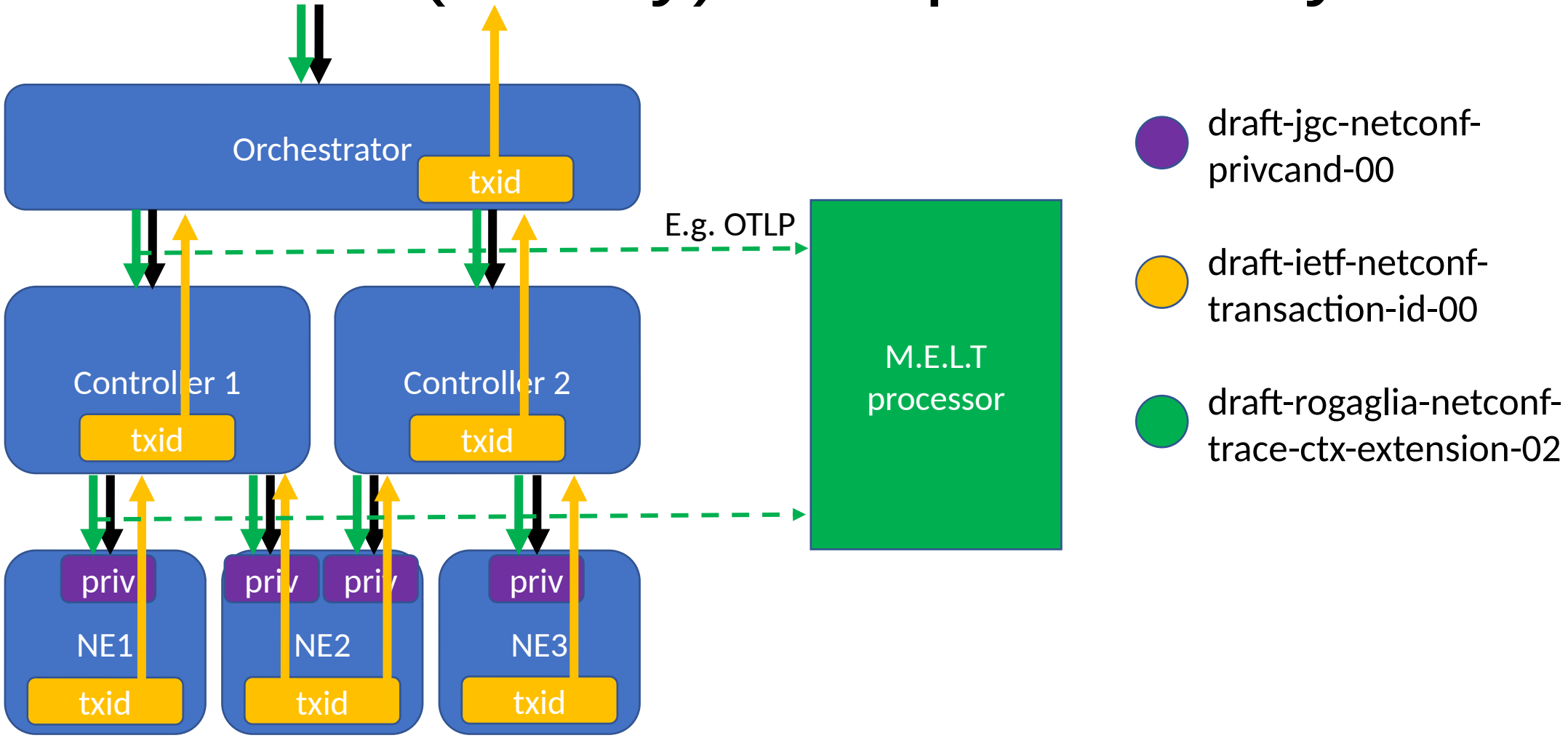
Drafts are (mostly) Complementary



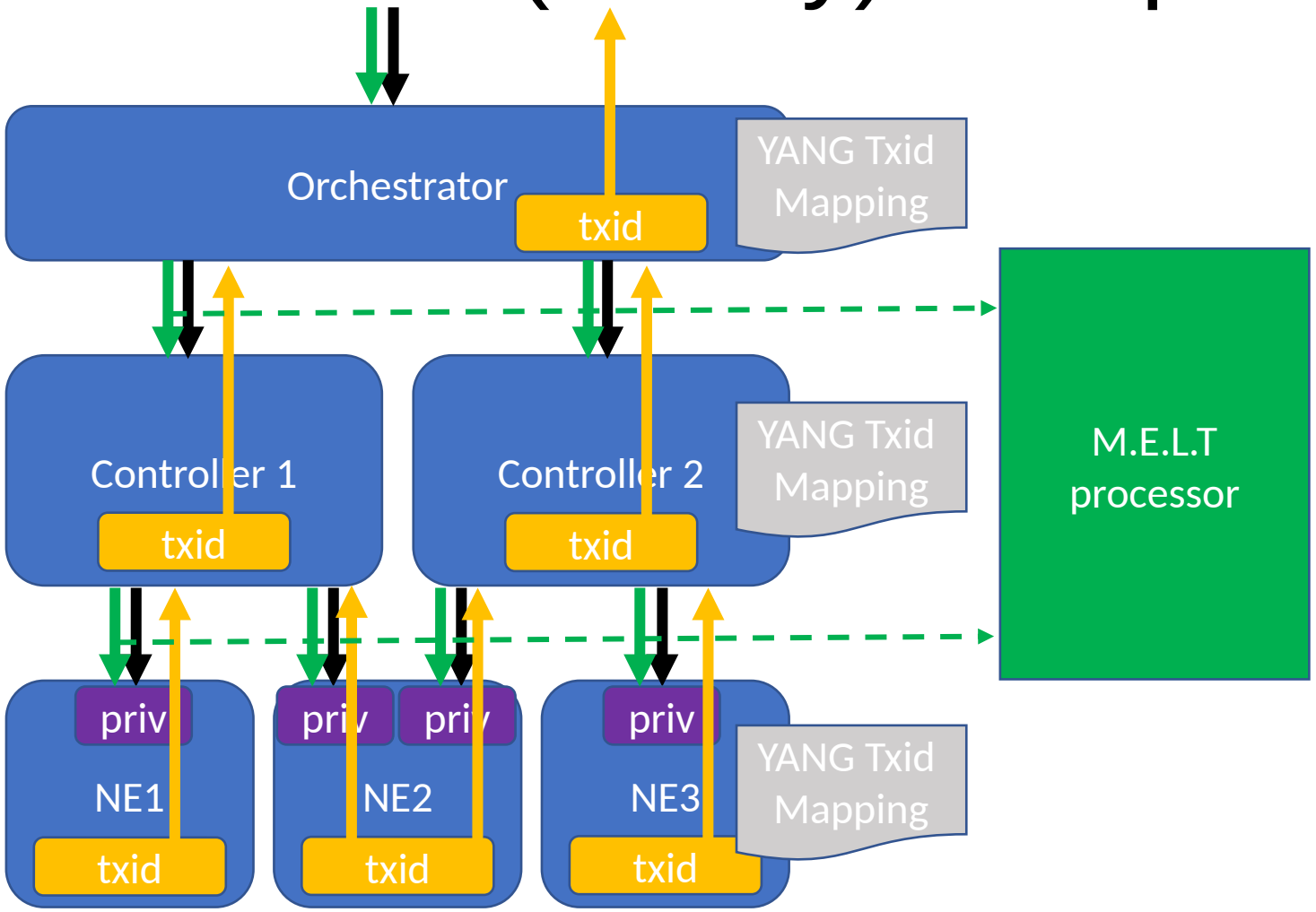
draft-jgc-netconf-privcand-00

draft-ietf-netconf-transaction-id-00

Drafts are (mostly) Complementary



Drafts are (mostly) Complementary



- draft-jgc-netconf-privcand-00
- draft-ietf-netconf-transaction-id-00
- draft-rogaglia-netconf-trace-ctx-extension-02
- draft-quilbeuf-opsawg-configuration-tracing-01

Direction Going Forward (IMHO)

Create framework of 4 separate documents

- Each one optional to implement, optional to use
- Use cases, terminology and behavior aligned

Thank you