External Transaction ID for Configuration Tracing

draft-quilbeuf-opsawg-configuration-tracing-01

J. Quilbeuf (Huawei), B. Claise (Huawei), T. Graf (Swisscom), D. Lopez (Telefonica), S. Qiong (China Telecom)

IETF 116, NETCONF
Motivation: Tracing Configuration

Something wrong with last change of configuration on NE2.
Where does the change comes from?

Use cases:

• **Configuration Mistake** “Which service request, if any, caused the mistake?”

• **Concurrent NMS modification** “Both NMS assume that they are in charge of the NE and regularly overwrite each other configuration.”

• **Conflicting Intents** ”Conflicting configuration changes are cause by two conflicting service requests.”
1 Slide recap of -00

Yang module + behavior for storing:
• Assign a transaction-id to each transaction, map with local-commit-id
• Store parent-transaction-id and parent-id when configured (i.e. as a NE)
• Store child-transaction-id when configuring (i.e. as Orchestrator)
• Store both when configured and configuring (i.e. as Controller)

Behavior for tracing:
• Find local-commit-id for “Bad” change on NE
• Match parent-transaction-id on NE to child-transaction-id on device identified by parent-id
• Recurse until no more parent-transaction-id is found.
Clarification: renaming and defining fields

Northbound Transaction -> Parent Transaction
Southbound Transaction -> Child transaction
Northbound Client ID -> Client ID
Clarification: Link with Existing on most devices and Role of Client ID

4.1. Existing configuration metadata on device

This document assumes that NETCONF clients or servers (orchestrators, controllers, devices, ...) have some kind of mechanism to record the modifications done to the configuration. For instance, routers typically have an history of configuration change and this configuration associates a locally unique identifier to some metadata, such as the timestamp of the modification, the user doing the modification or the protocol used for the modification. Such a locally unique identifier is a Local Commit ID, we assume that it exists on the platform. This Local Commit ID is the link between the module presented in this draft and the device-specific way of storing configuration changes.

4.2. Client ID

This document assumes that each NETCONF client for which configuration must be traced (for instance orchestrator and controllers) has a unique client ID among the other NETCONF clients in the network. Such an ID could be an IP address or a host name. The mechanism for providing and defining this client ID is out of scope of the current document.

4.3. Instantiating the YANG module
Link with draft-rogaglia-netconf-trace-ctx-extension/

• Traceparent contents (from W3C Trace Context):

```
00-4bf92f3577b34da6a3ce929d0e0e4736-00f067aa0ba902b7-01
```

version | trace-id | parent-id | flags

seems aligned with our case:
• trace-id is a good candidate for playing the role of the transaction id
  • Every configuration resulting from a single service request gets the same trace-id

• parent-id is actually a transient ID generated in the context of the trace, not the same as the client-id
• Client-id could be encoded in the tracestate, or some connection metadata (for instance IP address of the client)

The NETCONF extension proposed in trace-ctx-extension implements a way to pass the transaction-id and client-id between the NETCONF client and the NETCONF server.
Link with [draft-rogaglia-netconf-trace-ctx-ctx-extension](#)

<table>
<thead>
<tr>
<th></th>
<th>configuration-tracing</th>
<th>trace-ctx-extension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>Configuration</td>
<td>Generic</td>
</tr>
<tr>
<td>Global Infrastructure Needed</td>
<td>Inventory mapping client-id to access credentials</td>
<td>Collector (OTLP or YANG Push)</td>
</tr>
<tr>
<td>Lookup for configuration tracing</td>
<td>Every element involved in configuration</td>
<td>Only collectors</td>
</tr>
<tr>
<td>History kept until</td>
<td>Element removes it</td>
<td>Collectors/DB removes it</td>
</tr>
</tbody>
</table>

Tracing with the context might require to lookup several collectors if spawning over several domains.

Seems that we have two complementary alternatives, makes sense to depend on the extension proposed by trace-ctx-extension in configuration-tracing at it covers the NETCONF extension part and simplifies the draft.
Open questions/items to consider

• Draft to be updated from transaction-id to trace-id: switch dependency from netconf-transaction-id to netconf-trace-ctx-extension.
• Where to get client ID from: tracestate or other metadata?

• Risk of collision between southbound transaction-ids from different southbound elements
  • Trace-id should be unique

• Is NETCONF the right scope? Should we include RESTCONF as well?
  Other configuration protocols?
  • The Trace Context approach can be used with RESTCONF as well
Conclusion

• Thanks for reviews (Med!)
• Is the problem of mapping configuration changes in NEs to a service request a valid problem?
• Is our solution a good approach?

Draft repo is at
https://github.com/JeanQuilbeufHuawei/draft-quilbeuf-opsawg-configuration-tracing