External Transaction ID for Configuration Tracing

draft-quilbeuf-opsawg-configuration-tracing-02

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

IETF 117, 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.”
Changes: switch to W3 Trace-Parent

Modelled trace-parent, most important part is trace-id, which is the same for every transaction in the configuration chain.

For example in the Controller, the same local-commit-id will be mapped to “tr-1”, indicating that it was created by transaction “tx-1” and resulted in transactions “tx-2” and “tx-3”.
New: Modeling W3C Trace-parent

00-4bf92f3577b34da6a3ce929d0e0e4736-00f067aa0ba902b7-01

version  trace-id  parent-id  flags

```yaml
grouping trace-parent-g {
  description
  "Trace parent frow the W3C trace-context recommendation.
    Follows the format version 00."
  leaf version {
    type hex-digits {
      length "2";
    }
    must "/version = '00'";
    description
    "Version of the trace context. Must be 00 to match the
     format described in this module.";
  }
  leaf trace-id {
    type hex-digits {
      length "32";
    }
    must "/trace-id != '00000000000000000000000000000000'";
    description
    "Trace ID that is common for every transaction that is
     part of the configuration chain. This value can be used
     to match a local commit id to a commit local to another
     system.";
  }
  leaf parent-id {
    type hex-digits {
      length "16";
    }
    description
    "ID of the request (client-side) that lead to configuring
     the server hosting this module.";
  }
  leaf trace-flags {
    type hex-digits {
      length "2";
    }
    description
    "Flags enabled for this trace. See W3C reference for the
     details about flags.";
  }
}
```
Configuration Tracing

1. Find the “bad” local-commit-id in the NE
2. Retrieve the corresponding trace-id “tr-1” and client id (Controller here)
3. Look for the local-commit-id of Controller that has “tr-1” as trace-id
4. Get the corresponding client id (Orchestrator here)
5. Look for the local-commit-id of Orchestrator that has “tr-1” as trace-id
6. Retrieve that local-commit-id, which might refer to service request for instance.

Recursive process: as long as a client-id is present, one can go a level upper in the hierarchy.
Open questions/items to consider

• Where to get client ID from: tracestate, trace Baggage?
• Should we make a more general solution (i.e. follow
  https://github.com/open-telemetry/opentelemetry-proto/blob/main/opentelemetry/proto/trace/v1/trace.proto)
  and only specify what is missing (i.e. client-id, local-commit-id)?

• 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?
  • RESTCONF supported as stated in draft-rogaglia-netconf-restconf-trace-ctx-headers-00
Conclusion

• Is our solution a good approach?
• We would like this draft to be adopted by NETCONF WG

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