YANG Extension and Metadata Annotation for Immutable Flag

draft-ma-netmod-immutable-flag-09

Qiufang Ma (Huawei) Presenter
Qin Wu (Huawei)
Balazs Lengyel (Ericsson)
Hongwei Li (HPE)
Document Status

• Derived from “system-defined configuration” work
  • Document the immutability of some system configuration
    • HW-related configuration, built-in keys/certs, server-provided users/policies, server capabilities.

• Adoption poll after IETF 117
  • Some concern/objections received

• Some major concern to highlight in this presentation:
Concern #1: The server can dynamically decide what is considered immutable

• Never be the case in this draft
  • Immutability and its instantiated value are relatively stable.
    • must only change during upgrade or due to licensing or the addition or removal of hardware
    • independent on the operational state

• Some text pointed out to be ambiguous, removed in the latest revision.

• Further comments, suggestions are always welcome.
Concern #2: The server’s behavior follows the YANG model (immutable yang extension), not the other way round

• We’ve now removed the immutable YANG extension definition to simplify the solution
  
  • Only immutable annotation [RFC7952] in the current draft
    
    • descriptive information to annotate system config which cannot be overridden
    
    • Not included in server’s response unless being explicitly asked via a “with-immutable” parameter
Concern #3: The relationship of the proposed system datastore and this immutable mechanism is unclear

• A new section has been added to discuss the relationship between immutable flag and <system>
  • Immutable config refers to the node instantiated by system yet cannot be overridden by clients
  • System config (immutable or not) exists regardless of the implementation of <system>
  • Immutable config is generated in <system> (if implements) but doesn’t appear in <running> until being referenced (on the premise that <running> needs to satisfy referential integrity)
    • Immutable config being present in <running> is achieved by either the client copying it explicitly or the server doing it automatically via “resolve-system”.
  • Immutable mechanism allows a server to annotate non-modifiable system config with metadata annotation when it is retrieved
    • A client aware of the “immutable” annotation can avoid making unnecessary modification attempts
    • Legacy clients don’t see any changes and encounter the error response as always.
Comments, Questions, Concerns?