Status and Issues for the “Client-Server” Suite of Drafts

draft-ietf-netconf-crypto-types-05
draft-ietf-netconf-trust-anchors-03
draft-ietf-netconf-keystore-08
draft-ietf-netconf-ssh-client-server-11
draft-ietf-netconf-tls-client-server-10
draft-ietf-netconf-netconf-client-server-10
draft-ietf-netconf-restconf-client-server-10
+ draft-kwatsen-netconf-tcp-client-server-00
draft-kwatsen-netconf-http-client-server-00

NETCONF WG
IETF 104 (Prague)
Since IETF 103

All drafts updated and submitted as a set

Progress made on the two issues discussed before:

1. Finalizing the "crypto-types" identities.
2. Adding support for TCP Keep-alives.

This presentation focuses on these two issues

- plus a few additional issues that have surfaced....
Begin discussion #1

Finalizing the "crypto-types" Identities.
Updates to Crypto Types Draft

(From the Change Log)

- **Renamed** base identity 'asymmetric-key-encryption-algorithm' to 'asymmetric-key-algorithm'.

- **Added** new 'asymmetric-key-algorithm' identities for secp192r1, secp224r1, secp256r1, secp384r1, and secp521r1.


- For all -cbc and -ctr identities, **renamed** base identity 'symmetric-key-encryption-algorithm' to 'encryption-algorithm'.

- For all -ccm and -gcm identities, **renamed** base identity 'symmetric-key-encryption-algorithm' to 'encryption-and-mac-algorithm' and **renamed** the identities to remove the "enc-

- For all the 'signature-algorithm' based identities, **renamed** from 'rsa-*' to 'rsassa-*'.

- **Removed** all of the "x509v3-" prefixed 'signature-algorithm' based identities.

- **Added** 'key-exchange-algorithm' based identities for 'rsaes-oaep' and 'rsaes-pkcs1-v1_5'.

- **Renamed** typedef 'symmetric-key-encryption-algorithm-ref' to 'symmetric-key-algorithm-ref'.

- **Renamed** typedef 'asymmetric-key-encryption-algorithm-ref' to 'asymmetric-key-algorithm-ref'.

- **Added** typedef 'encryption-and-mac-algorithm-ref'.

Questions

Currently in the crypto-types draft, we have defined following identities for crypto and mac algorithms:

- hash-algorithm
- asymmetric-key-algorithm
- mac-algorithm
- encryption-algorithm
- encryption-and-mac-algorithm
- signature-algorithm
- key-exchange-algorithm

1. Is there any suggestion to the classification of the algorithms?

2. Is there new category to be added in the classification?
Begin discussion #2

Adding support for TCP Keep-alives.
Current **Adopted** Solution

crypto-types

<table>
<thead>
<tr>
<th>trust-anchors</th>
<th>keystore</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssh-client-server</td>
<td>tls-client-server</td>
</tr>
</tbody>
</table>

netconf-client-server

restconf-client-server
Previously Discussed Proposal

Adding in the missing tcp/http/https-client-server Layers
Current/Published Proposal

crypto-types

trust-anchors

keystore

tcp-client-server

ssh-client-server

tls-client-server

http-client-server

netconf-client-server

restconf-client-server
The Good

Has-a (not Is-a)

1. Enables application-level models to compose stack via 'uses' statements.

- **HTTP Client**
  - uses tcpc:tcp-client-grouping
  - uses httpc:http-client-grouping

- **HTTPS Client**
  - uses tcpc:tcp-client-grouping
  - uses tlsc:tcp-client-grouping
  - uses httpc:http-client-grouping

- **HTTPS Call Home Client**
  - uses tcps:tcp-server-grouping
  - uses tlsc:tcp-client-grouping
  - uses httpc:http-client-grouping

2. Avoids "devil diamonds"
   - Multiple-inheritance problem where based class is used twice
   - Example:

```
+---------------------+---------------------+---------------------+
| tcp-client          | tls-client          | http-client         |
|                     |                     | TCP base instantiated twice! |
|                     |                     | (current proposal avoids this issue) |
|                     |                     |                      |
|                     |                     |                      |
```
The Bad

Top-level groupings nodes are in same namespace
- thus names may conflict!

In order to demux node names, either:

1. Prefix the top-level nodes: *(current approach, mostly...)*

   grouping tcp-client-grouping {
     leaf remote-address {...}
     _container tcp-keepalives {...}
     _anydata tcp-bar {...}
   }

2. Wrap everything in a prefixed container:

   grouping tcp-client-grouping {
     _container tcp-client-params {
       leaf remote-address {...}
       _container keepalives {...}
       _anydata foo {...}
     }
   }

   grouping tls-client-grouping {
     _container tls-client-params {
       _container tls-keepalives {...}
       _anydata tls-bar {...}
     }
   }

   grouping http-client-grouping {
     _container http-client-params {
       _container http-keepalives {...}
       _anydata http-baz {...}
     }
   }
The Ugly

Should the NC/RC models also follow the "Has-A" pattern?

- Do we care about possible protocols built on top of NC/RC?

- Presumably we'd isolate that which is configured per-socket/session, from the larger multi-socket/session model supporting, e.g., "listen" and "call-home"

- Both could be in same draft by:
  1. Factor-out the inner per-session data models to their own groupings
     - Let these have the, e.g., "ietf-netconf-client-grouping" names
  2. Rename the original/larger models to something more appropriate:
     - e.g. "ietf-netconf-client-application-grouping"

Thoughts?
Begin discussion #3

Other Issues that have Surfaced...
And Other Issues

1. Protocol specific parameters are per-socket (redundant)
   - E.g., TCP keepalive must be set for each client/server
   - To be fair, this is inherent in any list of like-items
   - Proposed fix: do nothing, wait for a TBD templating mechanism
     - YANG-Next Issue #18 (importance-med, backcompat-high, complexity-high)
       - I.e. Juniper’s “apply-groups” statement

2. Keepalive config MAY be present for "periodic" connections
   - We previously removed keepalives from periodic config...
   - Proposed fix: add "must not" expressions...
More Other Issues

3. Are all the protocol-specific keepalives models correct?
   - TCP is okay (model after POSIX), but what about the others?

4. Any desire for other protocol-specific config?
   - e.g., HTTP proxy settings

5. Not all HTTP auth schemes are defined

   ```
   container hoba { // FIXME
      description
      "The 'hoba' HTTP scheme credentials.";
      reference
      "RFC 7486: HTTP Origin-Bound Authentication (HOBA)";
   }
   ```

6. Why do we have breakout groupings again?

   ```
   grouping ssh-client-grouping {
      uses client-identity-grouping;
      uses server-auth-grouping;
      uses transport-params-grouping;
      uses keepalives-grouping;
   }
   ```

7. TLS draft references obsolete RFCs! TLS 1.0, 1.1, 1.2
   - needed?
Thanks for the input!