Status and Issues for the "Client-Server" Suite of Drafts

draft-ietf-netconf-crypto-types-17
draft-ietf-netconf-trust-anchors-12
draft-ietf-netconf-keystore-19
draft-ietf-netconf-tcp-client-server-07
draft-ietf-netconf-ssh-client-server-21
draft-ietf-netconf-tls-client-server-21
draft-ietf-netconf-http-client-server-04
draft-ietf-netconf-netconf-client-server-20
draft-ietf-netconf-restconf-client-server-20

NETCONF WG
IETF 108 (Virtual)

Since IETF 107

High-level Updates:

crypto-types:

- Removed the IANA-maintained registries for symmetric, asymmetric, and hash algorithms.
- Removed the "generate-symmetric-key" and “generate-asymmetric-key" RPCs.
- Removed the "algorithm" node in the various symmetric and asymmetric key groupings.
- Added typedefs csr, csr-info, oscp-request, ocsp-response.
- Added "encrypted" case to both asymmetric and symmetric key groupings.
- Added "cleartext-" prefix to key nodes.

trust-anchors:

- Modified 'local-or-truststore-certs-grouping' to use a list (not a leaf-list).
- Added new example section "The Local or Truststore Groupings”.
- Clarified expected behavior for "built-in" certificates in <operational>.

keystore:

- Added new section "Encrypting Keys in Configuration".
- Clarified expected behavior for "built-in" keys in <operational>
- Clarified the "Migrating Configuration to Another Server" section.

Recently
Last Called

Since IETF 107 (cont.)

tcp-client-server:

- Added support for TCP proxies.

ssh-client-server:

- Removed algorithm-mapping tables from the "SSH Common Model" section.
- Renamed both "client-certs" and "server-certs" to "ee-certs"
- A few "must" and "mandatory" modifications.

tls-client-server:

- Removed algorithm-mapping tables from the "SSH Common Model" section.
- Renamed both "client-certs" and "server-certs" to "ee-certs"
- A few "must" and "mandatory" modifications.

http-client-server:

- Removed "protocol-versions" from ietf-http-server based on HTTP WG feedback.
- Added a parent "container" to "client-identity-grouping" so that it could be better used by the proxy model.
- Added a "choice" to the proxy model enabling selection of proxy types.
- Added 'http-client-stack-grouping' and 'http-server-stack-grouping' convenience groupings.

netconf-client-server:

- Many updates to examples.

restconf-client-server:

- Many updates to examples.

What to do about cleartext password fields?

A raw password required whenever a model configures a client to authenticate itself to a remote system

- Occurs for SSH-client, HTTP-client, and SOCKS5-client.
- Unlike when password is used to authenticate a client
  - in which case “ianach:crypt-hash” can be used

All of these nodes are tagged with “nacm:default-deny-all”

- But can we do better?

Thoughts:

1. “password” —> “cleartext-password”
   - Only helpful if an option exists
2. Add an “encrypted-password”?
   - i.e., use “ct:encrypted-key-value-grouping”
3. Use “ct:symmetric-key-grouping”?
   - Comes with the “key-format” field
     - Which makes the cleartext value be type binary
4. Define a new “ct:password-grouping”?

Hardcode the “format” based of type of the “encrypted-by” key?

```
grouping password-grouping {
  choice password-type {
    nacm:default-deny-write;
    mandatory true;
    case cleartext-password {
      leaf cleartext-password {
        nacm:default-deny-all;
        type string;
   }
    case encrypted-password {
      container encrypted-password {
       uses ct:encrypted-key-value-grouping;
      }
    }
  }
}
```

Specifying HTTP-client Paths

The current “http-client-group” is solely focused on connectivity

- e.g., the HTTP’s client’s identity
- A fully configured “stack” →

```
<tcp-client-parameters>
  <remote-address>
    corp-fw2.example.com
  </remote-address>
</tcp-client-parameters>
<tls-client-parameters>
  <server-authentication>
    <ca-certs>
      <truststore-reference>
               trusted-server-ca-certs
             </truststore-reference>
    </ca-certs>
  </server-authentication>
</tls-client-parameters>
<http-client-parameters>
  <client-identity>
    <basic>
      <user-id>local-app-1</user-id>
      <password>secret</password>
    </basic>
  </client-identity>
</http-client-parameters>
```

It is assumed that the client knows how to construct the URL path (e.g., RESTCONF)

- And query parameters, the request body, etc.

The “https-notif” draft augments-in a “path”:

```
uses httpc:http-client-stack-grouping {
  augment "transport/tls/tls/http-client-parameters" {
    leaf path {
      type string;
      description
        "Relative URI to the target resource.";
    }
    description
      "Augmentation to add a path to the target resource.";
  }
```

Any change needed?

FIXMEs in the PSK’s “id” node

```
case psk {
  if-feature psk-auth;
  container psk {
    description
      "Specifies the server identity using a PSK (pre-shared
       or pairwise-symmetric key).";
    uses ks:local-or-keystore-symmetric-key-grouping;
    leaf id {
      type string;     // FIXME: is this the right type?
      mandatory true;  // FIXME: is it mandatory?
      description
        "The key 'id' value when used in the TLS protocol.”;
      reference
        “FIXME: Where defined?”;
      }
    }
  }
}
```

All drafts primed for WGLC…

Any comments before start?