NETCONF and RESTCONF Client/Server Models

Drafts covered:

- draft-ietf-netconf-keystore-01
- draft-ietf-netconf-ssh-client-server-02
- draft-ietf-netconf-tls-client-server-02
- draft-ietf-netconf-netconf-client-server-02
- draft-ietf-netconf-restconf-client-server-02

NETCONF WG

IETF 98 (Chicago)

Recap

- In the IETF 97 (Seoul), we reported little progress on any of the drafts.
- The only real change made to the drafts then was to address the keystore-renaming issue.
- But we had said that, with zerotouch winding down, that the expectation was that these drafts would start to get more attention.

Updates since IETF 97

- While zerotouch did NOT wind down as expected, these drafts still got a fair amount of attention.

- Keystore:
  - Replaced cert-chain idiom with PKCS#7 structures
  - Added 'private-key' as a configurable data node, and removed the 'generate-private-key' and 'load-private-key' actions.
  - Moved 'user-auth-credentials' to the ietf-ssh-client module.

- SSH Client/Server
  - removed transport-specific grouping (module only defines one grouping now)
  - Simplified the "client-auth" part in the ietf-ssh-client module. It now inlines what it used to point to keystore for.
  - Added cipher suites for various SSH-specific algorithms.

- TLS Client/Server
  - removed transport-specific grouping (module only defines one grouping now)
  - Filled in previously incomplete 'ietf-tls-client' module.
  - Added cipher suites for various TLS-specific algorithms

Updates since IETF 97 (cont.)

- NETCONF Client/Server
  - Added to ietf-netconf-client ability to connected to a cluster of endpoints, including a reconnection-strategy.
  - Added to ietf-netconf-client the ability to configure connection- type and also keep-alive strategy.
  - Updated both modules to accommodate new groupings in the ssh/tls drafts.
- RESTCONF Client/Server
  - Filled in previously missing 'ietf-restconf-client' module.
  - Updated the ietf-restconf-server module to accommodate new grouping 'ietf-tls-server-grouping’
- Other drafts are planning to use these models:
  - draft-ietf-netmod-syslog-model
  - draft-ietf-pce-pcep-yang

Open Issues

- Keystore:
  - Should 'private key' be a union?
  - Add back `generate-private-key` action?
- SSH Client/Server:
  - Simplified client-auth okay for call-home apps?
- TLS Client/Server:
  - Simplified client-auth okay for call-home apps?

$\left.\right\}$ **Same Issue**

- NETCONF Client/Server:
  - Should NETCONF-client be a grouping?
- RESTCONF Client/Server:
  - Should RESTCONF-client be a grouping?

$\left.\right\}$ **Same Issue**

Should ‘private-key’ be a union?

What should be the treatment for when NACM hides a value, resulting in an invalid response?

```
leaf private-key {
  nacm:default-deny-all;
    type union {
      type binary;
      type enumeration {
        enum "RESTRICTED" {
          description
           "The private key is restricted due to access-control.";
        }
        enum "INACCESSIBLE" {
          description
           "The private key is inaccessible due to being protected
            by the cryptographic hardware modules (e.g., a TPM).";
        }
      }
    }
    mandatory true;
    description
      "A binary string that contains the value of the private
       key. The interpretation of the content is defined in the
       registration of the key algorithm.  For example, a DSA key
       is an INTEGER, an RSA key is represented as RSAPrivateKey
       as defined in [RFC3447], and an Elliptic Curve Cryptography
       (ECC) key is represented as ECPrivateKey as defined in
       [RFC5915]";
}
```

Add back `generate-private-key` action?

This action was removed when we added ‘private-key’, protected by “nacm:default-deny-all” (see previous slide).

But:

1. It is still best practice to have a device generate the private key
   - so it never leaves the device)
2. The private key needs to be generated in hardware sometimes
   - no option to set via configuration

My plan is a add this action statement back, with the explanation that it only updates the “operational” datastore, so that certificates can be configured on top of these system-generated private keys.

Any concerns?

Simplified client-auth okay for call-home apps?

- Works great for traditional clients, and also for call-home apps that want to use the same client-auth for *ALL* devices.
- For more complicated call-home apps, is it okay to assume that the app would use business logic to handle special client-auth logic?

```
module: ietf-ssh-client
    groupings:
    ssh-client-grouping
        +---- server-auth
        |   ...
        +---- client-auth
        |  +---- username?      string
        |  +---- (auth-type)?
        |     +--:(certificate)
        |     |  +---- certificate?   leafref {sshcom:ssh-x509-certs}?
        |     +--:(public-key)
        |     |  +---- public-key?    -> /ks:keystore/keys/key/name
        |     +--:(password)
        |        +---- password?      union
```

Configures just a single client.

The SSH-client grouping is presented here. A similar single-client construct exists in the TLS-client grouping as well.

Should NC/RC-client be a grouping?

- Having configuration for NC/RC-servers makes sense
  - since the server's backend MUST implement the modules it claims to support.
- But clients are different
  - A client must have business logic of some sort to do something. Specifically, an NC/RC client needs to be linked into an application that orchestrates its function.
- That being the case, how can a client ever be configured on its own?
  - Shouldn't the application itself be the thing that is configured?
- Should these client models be groupings instead of a containers?

Next Steps

- Work through remaining issues
- Complete Call Home reference implementation
  - exercises ietf-ssh-server call-home configuration
- Wait for other implementations
  - Syslog?
  - PCE-PCEP?
- Then Last Call

Questions, Comments, Concerns?