INTRODUCTORY PRESENTATION: NETCONF & CORECONF

With YANG Examples, CBOR Encoding, and CoAP Transport

Scott Mansfield (Ericsson)
+ help from Copilot

Network Configuration Protocols Matter

- Modern networks are no longer static.

- Devices must be configured, monitored, and updated programmatically.
- Traditional CLI automation is proprietary
- SNMP is historically limited to monitoring.

- **Model-driven management** using **Standardized YANG Modules**, **NETCONF**, and **CORECONF** is the **solution.**

YANG: The Foundation of Model-Driven Networking

- YANG is a data modeling language used to define:
  - Configuration data
  - State data
  - RPCs
  - Notifications

```
module example-interfaces {
  namespace "urn:example:interfaces";
  prefix if;

  container interfaces {
    list interface {
      key "name";
      leaf name {
        type string;
      }
      leaf enabled {
        type boolean;
        default "true";
      }
      leaf mtu {
        type uint16;
      }
    }
  }
}
```

NETCONF Overview

- NETCONF (RFC 6241) is a protocol designed for:
  - Reliable configuration changes
  - Transactional updates
  - Schema-validated operations
  - Secure transport (SSH)

Operation (not exhaustive list)	Purpose
<get>	Retrieve state data
<get-config>	Retrieve configuration
<edit-config>	Modify configuration
<lock> <unlock>	Apply changes transactionally
<validate> <commit>	Capabilities

CORECONF Overview

- NETCONF is powerful but overkill for:
  - IoT devices
  - Constrained networks
  - Low-power sensors
  - Embedded systems

- CORECONF (RFC 9254) adapts NETCONF concepts to the **CoRE (Constrained RESTful Environments)** ecosystem.

- CORECONF Goals
  - Uses **YANG**, but encode data in **CBOR (Concise Binary Object Representation)**
  - Uses **CoAP (Constrained Application Protocol)** instead of SSH
  - Reduce message size
  - Support constrained devices

CORECONF Architecture

- CORECONF = YANG + CBOR encoding + CoAP Transport

Feature	NETCONF	CORECONF
Encoding	XML	CBOR
Transport	SSH	CoAP
Target devices	Routers, switches	IoT, constrained nodes
Operations	RPC-based	REST-like CoAP methods
Efficiency	Medium	Very high

YANG to CBOR Mapping

- CORECONF uses SID (Semantic IDs) to compress YANG paths

```
module example-interfaces {
  namespace "urn:example:interfaces";
  prefix if;

  container interfaces {
    list interface {
      key "name";
      leaf name {
        type string;
      }
      leaf enabled {
        type boolean;
        default "true";
      }
      leaf mtu {
        type uint16;
      }
    }
  }
}
```

YANG Path	SID
/interfaces/interface/name	1001
/interfaces/interface/enabled	1002
/interfaces/interface/mtu	1003

<name>"eth0"</name>
<enabled>true</enabled>
<mtu>1500</mtu>

YANG

XML

CBOR

```
{
  1001: "eth0",
  1002: true,
  1003: 1500
}
```

Semantic ID Workflow

- Parse YANG modules
- Enumerate all schema nodes
- Assign SIDs deterministically
- Ensure no collisions
- Output a SID file (JSON or CBOR)
- Register ranges with IANA (optional but recommended)

```
{
  "module-name": "example-interfaces",
  "revision": "2024-01-01",
  "sid-file-version": 1,
  "items": [
    { "sid": 1001, "xpath": "/interfaces/interface/name" },
    { "sid": 1002, "xpath": "/interfaces/interface/enabled"
},
    { "sid": 1003, "xpath": "/interfaces/interface/mtu" }
  ]
}
```

Tooling Available!

Why CBOR?

- **Concise Binary Object Representation (CBOR - RFC 8949)**
  - Binary
  - Compact
  - Deterministic
  - Schema-friendly
  - Ideal for IoT

- Example
  - XML: "<mtu>1500</mtu>" is 23 bytes
  - CBOR: "1003: 1500" is 5 bytes

CORECONF Using CoAP

- CORECONF uses CoAP methods:

Operation	Purpose
GET	Retrieve data
PUT	Update configuration
POST	RPC-like operations
FETCH, PATCH	Partial Updates

CoAP PUT
coap://device/config/interfaces/interface/eth0
Payload (CBOR):
{
  1003: 1500
}

NETCONF vs CORECONF Summary

- Both share the same conceptual model but serve different environments.

Feature	NETCONF	CORECONF
Encoding	XML	CBOR
Transport	SSH	CoAP
Message Size	Large	Very small
Target	Full routers/switches	IoT, constrained devices
Security	SSH	DTLS/OSCORE
Data Model	YANG	YANG (SID compressed)

Decision Tree

NETCONF

- Managing routers, switches, firewalls
- When transactions and rollbacks are needed
- XML tooling is acceptable
- SSH-based security is required

CORECONF

- Devices are memory/CPU constrained
- Ultra-compact encoding is required
- CoAP is already used in your IoT stack
- YANG modeling consistency across all devices is desired

Where To Go From Here?

Standards

- RFC 6241 — Network Configuration Protocol (NETCONF)
- RFC 7950 — The YANG 1.1 Data Modeling Language
- RFC 8040 — RESTCONF Protocol
- RFC 8949 — Concise Binary Object Representation (CBOR)
- RFC 9200 — Constrained RESTful Environments (CoRE) Problem Statement
- RFC 9254 — CORECONF: A CBOR and CoAP-based Protocol for YANG Data
- RFC 9255 — YANG Schema Item iDentifier (SID) Specification
- RFC 9290 — IANA CBOR Tag Registries

Open Source

- aiocoap (Python) — https://github.com/chrysn/aiocoap
- libcoap (C) — https://github.com/obgm/libcoap
- CoAPthon — https://github.com/Tanganelli/CoAPthon
- cbor2 (Python) — https://github.com/agronholm/cbor2
- tinycbor — https://github.com/intel/tinycbor
- pyang SID plugin (bundled with pyang)