cisco

tail-f

RESTCONF

Per Andersson

ConfD Developer Days 2017

RESTCONF Talk

- RESTCONF Protocol
- ConfD RESTCONF
  - confd.conf
  - Before using RESTCONF
  - Limitations and Deviations
- How does RESTCONF compare?
  - RESTCONF and Legacy REST API
  - RESTCONF and NETCONF
- RESTCONF Demo
- Future plans

RESTCONF Protocol

RESTCONF Protocol

- Developed in the IETF NETCONF Working Group
- Defined in RFC 8040
- HTTP-based programmatic interface
- Access data defined in YANG
- Using datastore concepts from NETCONF

RESTCONF Protocol

- Not meant to replace NETCONF
- Provide a simplified interface following REST principles
  - Compatible with resource-oriented network element abstractions
- Envisioned for use cases like SDN controller integration, application automation, and OSS/BSS integration

RESTCONF Protocol

- HTTP methods provides CRUD operations on a conceptual datastore containing YANG defined data
  - GET: receives data
  - DELETE, PATCH, POST, PUT: modifies data
  - Payload is encoded with XML or JSON
    - `application/yang-data+xml`
    - `application/yang-data+json`

RESTCONF Protocol

- RESTCONF implements a subset of NETCONF concepts
  - Doesn’t implement e.g. datastores and locking
- RESTCONF provides event capability similar to the NETCONF event stream
- Access control mechanism used in RESTCONF is compatible with NETCONF Access Control Model (NACM)

RESTCONF Protocol

- RESTCONF messages
  - HTTP methods, URL:s, query parameters, headers, and payload
  - Resource part of URL, query parameters, headers, and payload are optional

```
GET /restconf/data/ietf-restconf-monitoring:restconf-state?depth=1 HTTP/1.1
Accept: application/yang-data+xml
```

RESTCONF Protocol

- RESTCONF error messages
  - HTTP status code
  - `error-type`
  - `error-path`
  - `error-tag`
  - `error-message`

RESTCONF Protocol

```
HTTP/1.1 400 Bad Request
Content-Type: application/yang-data+xml

<errors xmlns="urn:ietf:params:xml:ns:yang:ietf-restconf">
  <error>
    <error-type>protocol</error-type>
    <error-tag>invalid-value</error-tag>
    <error-path
     xmlns:ops="https://example.com/ns/example-ops">/ops:input/ops:delay</error-path>
    <error-message>Invalid input parameter</error-message>
  </error>
</errors>
```

RESTCONF Protocol

- Event Stream Resource
  - Client can retrieve a stream resource
  - Client can initiate long-poll server-sent event stream
  - Functions according to NETCONF Event Notifications (RFC 5277)
  - Available streams can be retrieved from “`streams`” list, which specifies syntax and semantics of the stream resources

ConfD RESTCONF

ConfD RESTCONF

- Supported since ConfD 6.3
  - Implements all mandatory parts of RFC 8040
- Sections from the RFC not implemented yet
  - Notifications (long-poll server-sent event stream)
    - Related optional query parameters: `filter`, `start-time`, `stop-time`, `replay`
  - Authentication with TLS client certificates

ConfD RESTCONF

- Extra functionality
  - Available under `/restconf/tailf`
    - `/restconf/tailf/query-api`
    - `/restconf/tailf/modules` (serving YANG schemas)
  - Collections
    - Necessary for XML encoded lists, payload needs to be wrapped in a top element
    - `applications/vnd.yang.collections+xml`

ConfD RESTCONF

- Extra functionality
  - REST Query API
    - Operations: `start-query`, `fetch-query-result`, `reset-query`, `stop-query`
  - New model-based method to view and apply rollback files
    - `tailf-rollback.yang`
    - Rollback file list
    - Actions: `get-rollback-file`, `apply-rollback-file`

confd.conf

- The webui must be enabled
  - HTTP and TLS ports are shared with the webui
- `/confdConfig/restconf/rootResource`
  - First part of RESTCONF API path, i.e. `{+restconf}`
  - Default: `/restconf`
- `/confdConfig/restconf/schemaServerUrl`
  - When behind a proxy server, report the proxied URL to YANG modules
- `/confdConfig/auth/rest/authCacheTTL`
  - RESTCONF caches AAA requests 10 seconds by default

Before using RESTCONF

- Evaluate: Are RESTCONF's capabilities enough for your application?
  - E.g. Web UI applications could probably benefit from using JSON-RPC (transactional, `get-schema` can return JSON etc.)
- Disable unencrypted HTTP transport
  - Since RESTCONF authorizes over HTTP Basic Auth, credentials are sent in plain text

Limitations and Deviations

- ConfD doesn’t yet support RESTCONF event streams (i.e. notifications), optional feature
- ConfD currently only supports plain patch
  - YANG Patch Media Type (RFC 8072) coming soon
- ConfD only supports HTTP Basic Auth
- Timestamps and Entity Tags apply to datastores only, not to resources within the datastore

How does RESTCONF compare?

Other Programmatic API:s

- SNMP
- Legacy REST API
- JSON-RPC
- API bindings
  - C, Erlang, Java, Python
- NETCONF

RESTCONF and Legacy REST API

RESTCONF and Legacy REST API

- Can be used simultaneously
- Root resource discovery via `/.well-known/host-meta`
  - Hard coded `/api` vs configurable `{+restconf}`
  - Discoverable by client
- RESTCONF use one unified datastore: `/restconf/data`
  - Similar to `/api/config` in Legacy REST
  - Legacy REST also has `/api/running`, `/api/candidate`, etc.

RESTCONF and Legacy REST API

- Datastore operations (`lock, commit, copy-running-to-startup, discard-change,` and `validate`) are not supported by RESTCONF
  - If datastore is locked by something else, RESTCONF fails the modify operation
- Query parameters differ
  - Name: `content`, `depth`, `fields` etc
  - Semantics: `?depth=N` instead of `?deep`, `?shallow`
  - Defaults: depth is `unbounded`, whereas it limited depth in Legacy REST

RESTCONF and Legacy REST API

- List keys are represented differently
  - RESTCONF
    - List with single key: `/restconf/data/example/list=key`
    - List with multiple keys:

      `/restconf/data/example/list=key1,key2`
  - Legacy REST: `/api/config/example/list/key1,key2`

RESTCONF and Legacy REST API

- Different media types
  - Legacy REST: `application/vnd.yang.api`, `application/vnd.yang.data`, `application/vnd.yang.datastore`, `application/vnd.yang.operation`
  - RESTCONF: `application/yang-data`
- YANG module `ietf-restconf-monitoring` support `ietf-restconf-streams` and capabilities
  - ConfD RESTCONF so far supports all capabilities except streams (i.e. notifications)
- RESTCONF supports schema retrieval YANG modules according to `ietf-yang-library` (RFC 7895)
  - Since ConfD 6.4, YANG source (i.e. schema) is compiled into the FXS files, making it available for download without adding the YANG module to the `loadPath`

RESTCONF and Legacy REST API

- RPC:s are found under `/restconf/operations`
- Actions are found under `/restconf/data`
- RESTCONF must use “`content`” query parameter to single out, omit, or mix-in operational data
  - Default is “`content=all`”, i.e. mixing config and operational data

RESTCONF and Legacy REST API

Query parameters

RESTCONF	Legacy REST	Description
content		Select config and/or non-config data resources
depth	deep/shallow	Request limited sub-tree depth in the reply content
fields	select	Request a subset of the target resource contents
filter		Boolean notification filter for event stream resources
insert	insert	Insertion mode for ordered-by user data resources
	limit	Specify a limited set of list entries to retrieve
	offset	Specify a limited set of list entries to retrieve
	operations	Specify whether to include/exclude operations (tailf:actions)
point	resource	Insertion point for ordered-by user data resources
start-time		Replay buffer start time for event stream resources
stop-time		Replay buffer stop time for event stream resources
	verbose	Display of the "self" and "path" attributes of the resource
with-defaults	with-defaults	Control retrieval of default values

RESTCONF and NETCONF

RESTCONF and NETCONF

- RESTCONF only supports one unified datastore
- Operations related to datastores not supported
  - `lock, commit, copy-running-to-startup, discard-changes, and validate`

RESTCONF and NETCONF

- `:writable-running`
  - If the NETCONF server supports `:writable-running`, all edits to configuration nodes in `/restconf/data` are performed in the running datastore.

RESTCONF and NETCONF

- `:candidate`
  - If the device instead supports `:candidate`, all edits are done within the candidate datastore
  - The candidate datastore is committed to running immediately after each successful edit
    - Any edits from other sources in candidate will also be committed

RESTCONF and NETCONF

- `:startup`
  - If the NETCONF server supports `:startup`, the RESTCONF server automatically updates non-volatile startup configuration datastore, after running datastore has been altered by a RESTCONF edit operation

RESTCONF and NETCONF

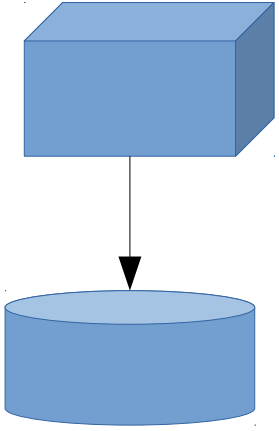
- Explicit locking
  - If a datastore that would be modified by a RESTCONF edit operation, which has an active lock from a NETCONF client, the RESTCONF edit operation will fail with HTTP 409 Conflict and error-tag “`in-use`”

RESTCONF and NETCONF

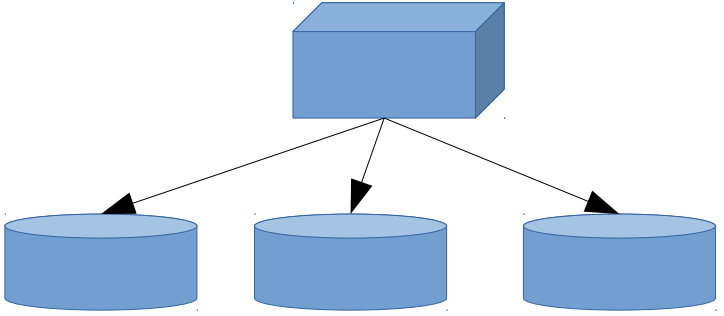
- A RESTCONF edit operation is a transaction in itself but transactions spanning multiple calls are not supported
- RESTCONF doesn’t support network wide transactions
- There is no validate operation in RESTCONF
  - Validation is implicit in every RESTCONF edit operation which either succeeds or fails

RESTCONF and NETCONF

- RESTCONF
  - Use for configuring single device



- NETCONF
  - Use for configuring multiple devices
    - Need two-phase/three-phase commit, candidate, validate



RESTCONF and NETCONF

Corresponding protocol operations

RESTCONF	NETCONF
HEAD	<none>
OPTIONS	<none>
GET	<get-config>, <get>
POST	<edit-config> (nc:operation=”create”)
POST	Invoke an RPC operation or an action
PUT	<edit-config> (nc:operation=”create/replace”)
PATCH	<edit-config> (nc:operation=”merge”)
DELETE	<edit-config> (nc:operation=”delete”)

RESTCONF Demo

Meanwhile in the IETF

- Adapting modernized datastore concepts to RESTCONF
  - draft-ietf-netconf-nmda-restconf-00
- NETCONF and RESTCONF Call Home
  - draft-ietf-netconf-server-model-09
  - draft-ietf-netconf-restconf-client-server-04
- YANG-Push
  - draft-ietf-netconf-yang-push-10
- Zero Touch Provisioning for NETCONF or RESTCONF based Management
  - draft-ietf-netconf-zerotouch-17

Future plans for ConfD

- YANG Patch Media Type
- Notifications (long-poll server-sent event streams)
- Swagger
- Other authentication (e.g. TLS client certificates)

YANG Patch Media Type

- Developed by the IETF NETCONF Working Group
- Defined in RFC 8072
- Describes a method to apply patches to configuration datastores using YANG
- Shortens the gap to NETCONF
  - Drawbacks of vanilla RESTCONF: Only one operation per request, large overhead

yang-patch request

- Unique id: “`patch-id`”
- A patch is an ordered collection of edits
  - Identified with “`edit-id`”
  - Target resource
- Available edit operations
  - `create, delete, insert, merge, move, replace, remove`

YANG Patch Demo

Swagger

- Describes REST API:s
- Idea
  - Emit swagger.json when compiling YANG models
  - First step: Enable browsing the API / YANG models with swagger-ui
  - Stretch goal: Enable API requests with swagger-ui, code generation
- Existing solutions today doesn’t follow RESTCONF

Swagger Mockup

References

- YANG – A Data Modeling Language for the Network Configuration Protocol (NETCONF)
  - https://www.rfc-editor.org/info/rfc6020
- The YANG 1.1 Data Modeling Language
  - https://www.rfc-editor.org/info/rfc7950
- YANG Module Library
  - https://www.rfc-editor.org/info/rfc7895
- Network Configuration Protocol (NETCONF)
  - https://www.rfc-editor.org/info/rfc6241
- NETCONF Event Notifications
  - https://www.rfc-editor.org/info/rfc5277
- RESTCONF Protocol
  - https://www.rfc-editor.org/info/rfc8040
- YANG Patch Media Type
  - https://www.rfc-editor.org/info/rfc8072
- https://swagger.io/

Thank you for listening!

www.tail-f.com

Opinions?

- How do you use RESTCONF?
- What do you want to use RESTCONF for?
- What are you missing in RESTCONF?
- Is there something stopping you from using RESTCONF?
- What do you use instead of RESTCONF?