OpenConfig

steve ulrich

22-sep, 2022

agenda

- overview of OpenConfig
- models
- OpenConfig (complementary) protocols
- practicalities
- conclusion

new fon - who dis?

steve ulrich

disclaimer / warning

- there are opinions in this presentation
- embrace your "bad minnesotan" and ask questions…

* fun fact in nearly 30 years (now i feel old) i've increased my baseline unit of connectivity measure by $1e^9$

what is OpenConfig?

open source, operator-driven working group generating the following:

- vendor-neutral data models describing network devices & protocol operation
- protocols and microservices for device management
- tools for interacting with network devices programmatically

operator participants

(MN)NUG
Minnesota Networking User Group

OpenConfig elements

models

- YANG (1.0) data models
- configuration data
- operational data
- vendor neutral

gNMI

(gRPC Network Management Interface)

- gRPC-based protocol for applying configuration and streaming telemetry
- concise set of RPCs
  - GET
  - SET
  - capabilities
  - subscribe
- supports streaming OC and vendor-specific data

miscellaneous

- gNOI - gRPC Network Operations Interface
  - ping/traceroute
  - OS upgrade
  - File operations
  - healthz
  - etc
- gRIBI - gRPC RIB Interface
  - *program* the RIB (finally)
- gNSI - gRPC Network Security Interface
  - a parallel world of AAA

models

why are models interesting?

- provide a consistent mechanism for representing a network element and its features and protocols
- can be validated with tooling for conformance and consistency
- can be wrapped with APIs/SDKs
  - actually *program* system state
- can reference other models (DRY)
- models open the door for new protocols and interactions

so. many. models.

a couple minutes poking at the yangcatalog api*

```
# get the dump of vendor specific models available
krustini(~)% curl -s  https://yangcatalog.org/api/search/vendors > yc-vendors.json
krustini(~)% ll yc-vendors.json
-rw-rw-r--  1 sulrich  staff    49M Sep  9 14:43 yc-vendors.json
# well, let's not do that again, interactively
# dump vendors publishing into the catalog
krustini(~)% jq '.vendor[].name' yc-vendors.json
"cisco"
"huawei"
"fujitsu"
"nonietf"
# dump the vendor specific model list…  scratch that, let's just count the models
krustini(~)% jq '.vendor[].platforms[][].name' yc-vendors.json | wc
      62      62     605   # 62 OS/platform specific models
```

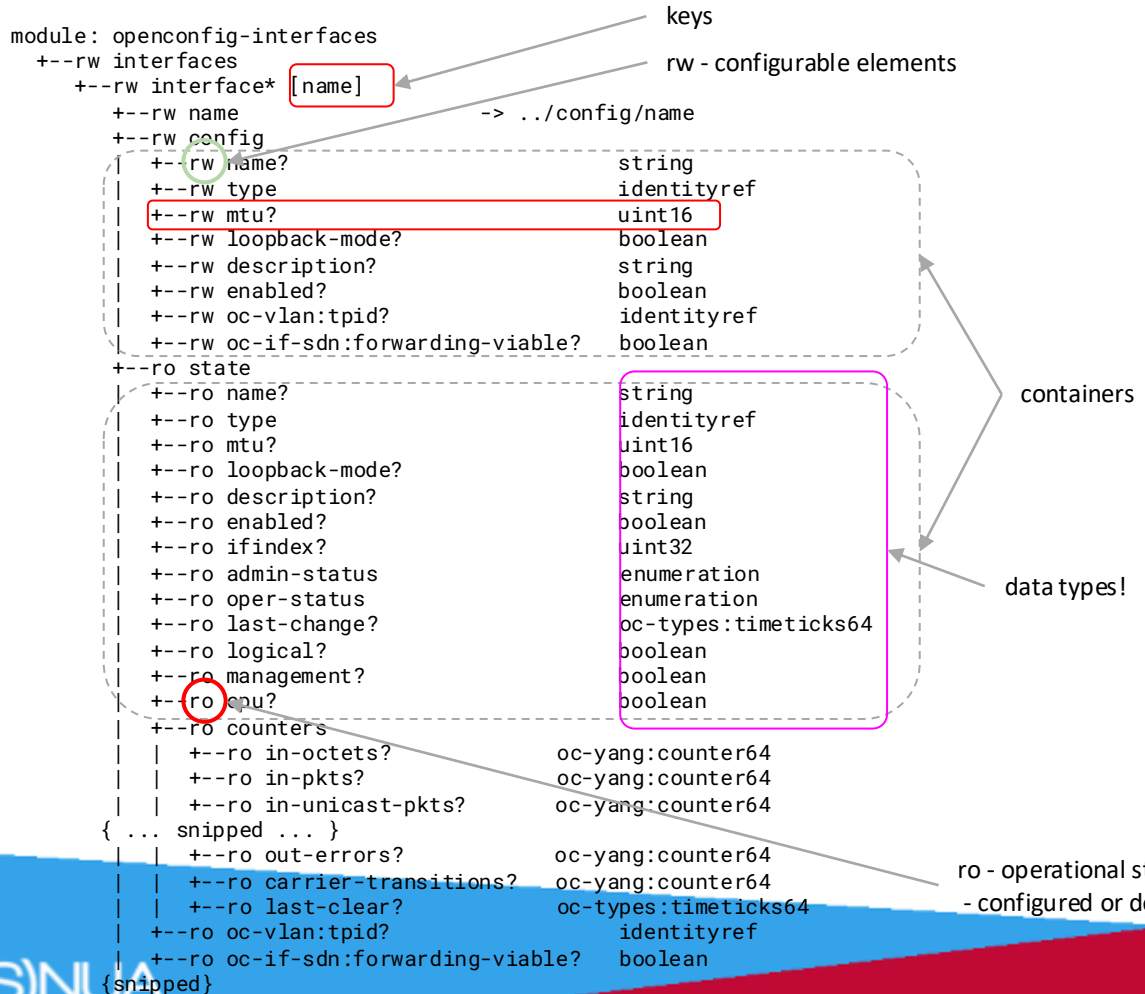

* where most of the time is spent detangling jq query params

models - continued

- vendors are publishing YANG models aplenty …
  - vendor models will (theoretically) have all of the features represented in a NETCONF-frobbable format for your ingest delight
  - these might (🤞) be internally consistent
  - good news! now you have the CLI in a programmatically frobbable format
- so is the IETF / IEEE / etc. … these will have varying degrees of support and internal consistency

¯_(ツ)_/¯

OpenConfig models



YANG is a data modeling language used to model configuration and state data manipulated by NETCONF (or some other mechanism)

the code

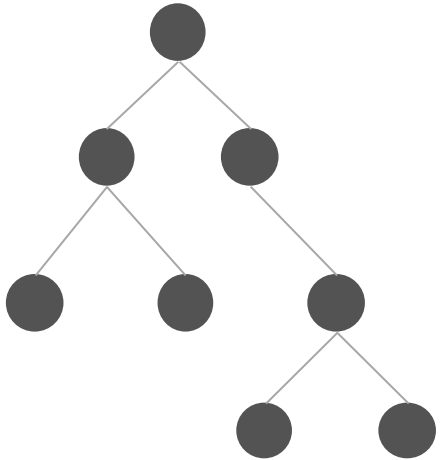
```
module openconfig-interfaces {

  yang-version "1";

  // namespace
  namespace "http://openconfig.net/yang/interfaces";

  prefix "oc-if";

  // import some basic types
  import ietf-interfaces { prefix ietf-if; }
  import openconfig-yang-types { prefix oc-yang; }
  import openconfig-types { prefix oc-types; }
  import openconfig-extensions { prefix oc-ext; }

  // meta
  organization "OpenConfig working group";

 ... snipped, lots ...

  // OpenConfig specific extensions for module metadata.
  oc-ext:regexp-posix;
  oc-ext:catalog-organization "openconfig";
  oc-ext:origin "openconfig";

  // typedef statements

  typedef base-interface-ref {
    type leafref {
      path "/oc-if:interfaces/oc-if:interface/oc-if:name";
    }
    description
      "Reusable type for by-name reference to a base interface.
      This type may be used in cases where ability to reference
      a subinterface is not required.";
  }

  typedef interface-id {
    type string;
    description
      "User-defined identifier for an interface, generally used to
      name a interface reference.  The id can be arbitrary but a
      useful convention is to use a combination of base interface
      name and subinterface index.";
  }

 ... snipped, lots ...
```

this just a taste of the first part of the interface model…

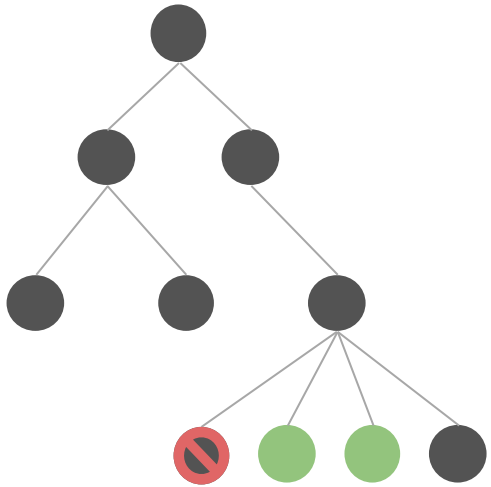
```
krustini(models/interfaces)[master] % ll
total 272
-rw-r--r--  1 sulrich  staff   8.3K Sep  7 20:53 openconfig-if-8021x.yang
-rw-r--r--  1 sulrich  staff   5.5K Sep  7 20:53 openconfig-if-aggregate.yang
-rw-r--r--  1 sulrich  staff   3.3K Sep  7 20:53 openconfig-if-ethernet-ext.yang
-rw-r--r--  1 sulrich  staff    18K Sep  7 20:53 openconfig-if-ethernet.yang
-rw-r--r--  1 sulrich  staff   4.2K Sep  7 20:53 openconfig-if-ip-ext.yang
-rw-r--r--  1 sulrich  staff    36K Sep  7 20:53 openconfig-if-ip.yang
-rw-r--r--  1 sulrich  staff   2.2K Sep  7 20:53 openconfig-if-poe.yang
-rw-r--r--  1 sulrich  staff   2.1K Sep  7 20:53 openconfig-if-sdn-ext.yang
-rw-r--r--  1 sulrich  staff   2.8K Sep  7 20:53 openconfig-if-tunnel.yang
-rw-r--r--  1 sulrich  staff    33K Sep  7 20:53 openconfig-interfaces.yang
krustini(models/interfaces)[master] % wc -l *
     318 openconfig-if-8021x.yang
     249 openconfig-if-aggregate.yang
     116 openconfig-if-ethernet-ext.yang
     683 openconfig-if-ethernet.yang
     179 openconfig-if-ip-ext.yang
    1328 openconfig-if-ip.yang
     110 openconfig-if-poe.yang
      69 openconfig-if-sdn-ext.yang
     120 openconfig-if-tunnel.yang
    1106 openconfig-interfaces.yang
    4278 total
```

extending OpenConfig models

OpenConfig standard model



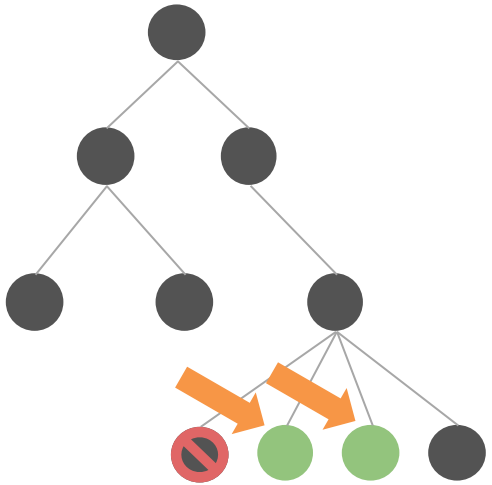
Vendor Extensions



extending OpenConfig models : augment

- YANG augment statement adds a YANG node to another node.
- example: adding a MAC learning option to a VLAN

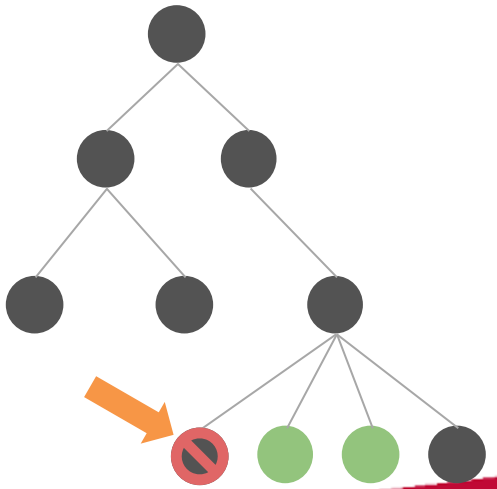
```
augment "/network-instances/network-instance/vlans/vlan/config" {
    leaf mac-learning {
      type boolean;
      default "true";
      description
        "Turn on/off mac learning on this VLAN.";
    }
}
```



extending OpenConfig models : deviation

- YANG deviation: lets a YANG node deviate from standard model
- example: changing mtu-discovery to be non-configurable and giving it a default value

```
deviation "/bgp/neighbors/neighbor/transport/config/mtu-discovery" {
    description
      "MTU discovery is implicitly always true in EOS";
    deviate replace {
      config false;
      default "true";
    }
  }
```



OpenConfig (complementary) protocols

OpenConfig - transport protocols

fyi … OpenConfig does not mandate a transport protocol.

operators can choose the transport/RPC protocols that are best suited for integration with their environments and overall automation strategy

popular options:

- **NETCONF (**RFC6241) - XML over SSH
- **RESTCONF** (RFC8040) - JSON/XML over HTTPS
- **gNMI** (gRPC Network Management Interface)
  - protobufs + JSON over HTTP2

gNMI - gRPC Network Management Interface

- gRPC-based protocol for modifying and retrieving system configuration
  - snapshots of system state (GET)
  - configuration (SET)
  - streaming telemetry data (subscribe)
- can support dial-in/out operational modes
- supports user/pass and mTLS authentication
- facilities to support hybrid schema configuration
  - i.e.: OpenConfig modeled and vendor-specific

RPC	functionality
GET	retrieve system state - supports limited XPath-like filtering mechanisms
SET	modify system state (delete, replace, update operations) - effectively the "config" RPC.
CAPABILITIES	provides info re: supported models, services, encodings, extensions, etc.
SUBSCRIBE	puts the streaming in streaming telemetry

gRPC Network Management Interface (gNMI)

- gRPC Network Management Interface
  - https://github.com/openconfig/reference
- gRPC is an RPC framework for client-server applications
  - protocol buffers (efficient binary encoding) over HTTP2
  - allows for request+response APIs and streaming
- RPCs: Capabilities, Get, Set, Subscribe
- data is organized around paths
  - `Set(system/state/hostname)`
  - `Set(system/config/hostname, “tor14”)`
  - `Subscribe(interfaces/interface/state/counters)`
  - Subscribe has options for how often to stream data
- ON_CHANGE, SAMPLED, TARGET_DEFINED
- responses contain timestamp, path, and value

gNMI: capabilities

model info available via gnmi capabilities RPC

```
gnmi -addr=10.0.66.1 -username admin -password arista123 capabilities
Version: 0.7.0
SupportedModel: name:"arista-bfd-deviations" organization:"Arista Networks, Inc."
SupportedModel: name:"arista-exp-eos-varp-net-inst" organization:"Arista Networks <http://arista.com/>"
SupportedModel: name:"arista-exp-eos-igmpsnooping" organization:"Arista Networks <http://arista.com/>"
SupportedModel: name:"arista-exp-eos-multicast" organization:"Arista Networks <http://arista.com/>"
SupportedModel: name:"openconfig-bgp-evpn" organization:"Arista Networks, Inc."
SupportedModel: name:"openconfig-lldp-types" organization:"OpenConfig working group" version:"0.1.1"
SupportedModel: name:"arista-rpc-netconf" organization:"Arista Networks, Inc."
SupportedModel: name:"openconfig-isis-lsdb-types" organization:"OpenConfig working group" version:"0.4.2"
SupportedModel: name:"openconfig-segment-routing" organization:"OpenConfig working group" version:"0.0.4"
```

OpenConfig Interface Counters

- subscription path: (interfaces/interface[name=*]/state/counters)
- Initially receive all current state, and then only paths that are modified
- gnmi command line tool output below

```
[2022-08-11T21:56:13.347333707Z] /interfaces/interface[name=Ethernet3]/state/counters/in-broadcast-pkts = 0
[2022-08-11T21:56:13.347312216Z] /interfaces/interface[name=Ethernet3]/state/counters/in-discards = 0
[2022-08-11T21:56:13.34735386Z] /interfaces/interface[name=Ethernet3]/state/counters/in-errors = 0
[2022-08-11T21:56:13.347345686Z] /interfaces/interface[name=Ethernet3]/state/counters/in-multicast-pkts = 0
[2022-08-11T21:56:13.347295652Z] /interfaces/interface[name=Ethernet3]/state/counters/in-octets = 0
[2022-08-11T21:56:13.347426105Z] /interfaces/interface[name=Ethernet3]/state/counters/in-unicast-pkts = 0
[2022-08-11T21:56:13.347325882Z] /interfaces/interface[name=Ethernet3]/state/counters/out-broadcast-pkts = 0
[2022-08-11T21:56:13.347397388Z] /interfaces/interface[name=Ethernet3]/state/counters/out-discards = 0
[2022-08-11T21:56:13.347416794Z] /interfaces/interface[name=Ethernet3]/state/counters/out-errors = 0
[2022-08-11T22:00:16.212850547Z] /interfaces/interface[name=Ethernet3]/state/counters/out-multicast-pkts = 300
[2022-08-11T22:00:16.212874987Z] /interfaces/interface[name=Ethernet3]/state/counters/out-octets = 36900
[2022-08-11T21:56:13.347365635Z] /interfaces/interface[name=Ethernet3]/state/counters/out-unicast-pkts = 0
…
[2022-08-11T22:00:20.201377743Z] /interfaces/interface[name=Ethernet3]/state/counters/out-multicast-pkts = 302
[2022-08-11T22:00:20.201355917Z] /interfaces/interface[name=Ethernet3]/state/counters/out-octets = 37146
```

origin: differences

OpenConfig

```
gnmi -addr=10.0.66.1 -username admin \
 -password arista123 \
 get "/network-instances/network-instance/protocols/protocol/bgp"
/network-instances/network-instance[name=default]/protocols/protocol[identifier=BGP][name=BGP]/bgp:
{
  "openconfig-network-instance:global": {
    "afi-safis": {
      "afi-safi": [
        {
          "add-paths": {
            "config": {
              "receive": false,
              "send": false
            },
            "state": {
              "receive": false,
              "send": false
...
```

eos_native

```
gnmi -addr=10.0.66.1 -username admin \
 -password arista123 \
 get origin=eos_native "/Sysdb/routing/bgp"

…
/Sysdb/routing/bgp/vrf/config/name: config
/Sysdb/routing/bgp/macvrfrdmap/name: macvrfrdmap
/Sysdb/routing/bgp/evpnconfig/duplicateThreshold: 5
/Sysdb/routing/bgp/evpnconfig/duplicateTimeout: 180
/Sysdb/routing/bgp/evpnconfig/name: evpnconfig
/Sysdb/routing/bgp/evpnconfig/schedClearTime: 0
/Sysdb/routing/bgp/macvrf/name: macvrf
/Sysdb/routing/bgp/testControlRequest/ribAddThreshold: 0

...
```

gNMI: config

- SetRequest() RPC enables configuration of supported OC model features
- gNMI supports hybrid or mixed-schema configuration
- enables the use of gNMI as a mechanism for configuration of vendor-native features and supported OC-model features

sample SetRequest() - OC

```
krustini(~)% gnmi -addr 10.0.66.139:6030 -username admin -password arista \
  update \
'/network-instances/network-instance[name=default]/protocols/protocol[name=BGP]/bgp/neighbors/neighbor[neighbor-address=198.51.100.43]' \
  '{"config": {"neighbor-address":"198.51.100.43", "peer-as": 123}}'
```

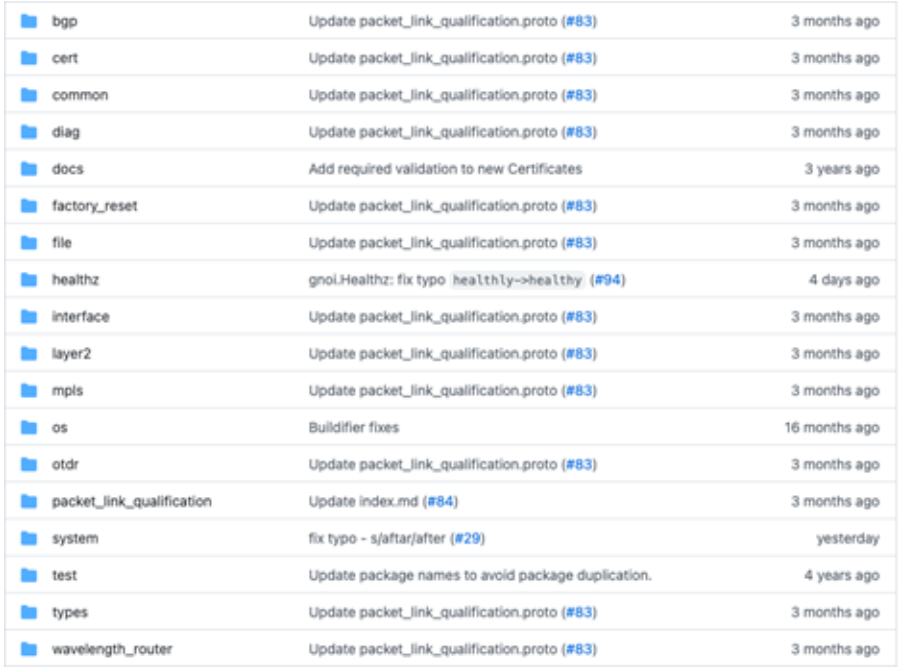
sample SetRequest() - vendor CLI

```
krustini(~)% gnmic -a 192.168.1.21:6030 --encoding ASCII set \
   --update-path "cli:" --update-value "logging host 192.168.1.13 514"
Set Response:
{
  "timestamp": 1598346946666698662,
  "time": "2022-08-25T04:15:46.666698662-05:00",
  "results": [
    {
      "operation": "UPDATE",
      "path": "cli:"
    }
  ]
   }
```

gNOI - gRPC Network Operations Interface

microservices to support operational tasks associated with a network element

- OS upgrades
- reboots
- OAM: ping / traceroute / BERT
- diagnostics / health
- link qualification
- "clear" operations
  - neighbor discovery protocol
  - BGP neighbor
  - spanning-tree
  - LSPs (counters)



bgp	Update packet_link_qualification.proto (#83)	3 months ago
cert	Update packet_link_qualification.proto (#83)	3 months ago
common	Update packet_link_qualification.proto (#83)	3 months ago
diag	Update packet_link_qualification.proto (#83)	3 months ago
docs	Add required validation to new Certificates	3 years ago
factory_reset	Update packet_link_qualification.proto (#83)	3 months ago
file	Update packet_link_qualification.proto (#83)	3 months ago
healthz	gnoi.Healthz: fix typo `healthly->healthy` (#94)	4 days ago
interface	Update packet_link_qualification.proto (#83)	3 months ago
layer2	Update packet_link_qualification.proto (#83)	3 months ago
mpls	Update packet_link_qualification.proto (#83)	3 months ago
os	Buildifier fixes	16 months ago
otdr	Update packet_link_qualification.proto (#83)	3 months ago
packet_link_qualification	Update index.md (#84)	3 months ago
system	fix typo - s/aftar/after (#29)	yesterday
test	Update package names to avoid package duplication.	4 years ago
types	Update packet_link_qualification.proto (#83)	3 months ago
wavelength_router	Update packet_link_qualification.proto (#83)	3 months ago

gRIBI - gRPC RIB Interface

- gRPC interface to manipulate the Abstract Forwarding Tables, aka AFT (OpenConfig modeled FIB) to effectively provide a routing protocol
- allows definition of:
  - next-hops (with weights)
  - next-hop groups
  - IPv[46] routes with these structures
  - specification of LFIB
    - NHs comprised of label stacks
- supports RIB and FIB ACK
- AFT can be "read-out" for validation

gNSI - gRPC Network Security Interface

- effectively a parallel AAA infrastructure for gNxI services
- oriented around gRPC and tightly aligned with gRPC-core capabilities
  - support for mTLS and SPIFFE-IDs w/policy
  - gRPC core CRL and cert management
- provides AAA on a per-RPC (authz) and service path (pathz) basis
- provides for certificate rotation and credential management
- updated accounting for gRPC services
- ssh and console credential rotation

practicalities

OpenConfig support

- broad support from the major networking equipment vendors
  - arista / cisco / juniper / nokia / PANW / others
  - all are shipping reasonably robust levels of OpenConfig model/gNMI support
- adoption by other OpenSource projects (e.g.: SONiC) as a means of collecting telemetry data (sonic-gnmi)
- growing ecosystem of tooling and support
  - popular open source projects have gNMI plugins
  - influxdb / gnmi-gateway / NAPALM, etc

telemetry - immediate win

stop scraping your devices down with SNMP

the year is 2022 …

we've been at this (OpenConfig) for 8 years it's time to move forward

that said …

- don't inflict more pain on yourself than strictly necessary
  - OpenConfig is probably what you're after for telemetry
  - gaps can be augmented (no pun intended) with vendor-specific model coverage
    - you will find gaps - these can be incrementally addressed
      - most vendors have some mechanism for you to obtain vendor specific model data in parallel
      - some will even stream it for you
- take care when pulling in additional model sources (IEEE, IETF, etc)
  - assuming your platforms even support them
  - additional dependencies create new problems
    - incompatible versioning, open references, etc.

the tyranny of SNMP

- expensive discoverability
  - need to re-walk the MIBS to discover new elements
- no capabilities / discovery
  - poke OID space to figure out what's supported
- data modeling and transmission from the 1990s
- no timestamps
- horrid scaling behaviors
- internal marshaling and caching layers induce measurement skew
- fragmented data models
  - partial walk then re-assembly / correlation required
- poor extensibility

admin interactions, are interesting …

currently - gNMI makes assumptions re: operational environment

- all operations are effectively as `admin`
- telemetry, has implications for visibility into `/system/aaa`
  - you may need to disable read/write paths for OpenConfig
- review vendor options for providing authz controls
  - minimally, apply some form of authz for SET RPCs
- logging has remained unspecified
  - if you have auditing considerations, talk to your vendor.

gNSI will address much of this, but …

config: it's rough out there

- can you generate a full device configuration in OpenConfig?
- if wireless APs are your thing … yes.
- if not, the answer is more complicated
  - common to find notable GET vs. SET differences
  - vendors adding model support at a pretty solid clip
    - feature coverage is largely driven by the needs of specific operators
  - enterprise features may not necessarily have the same level of OpenConfig model definition
  - mixed-schema may be an option - use with caution [more on this in a bit]
- you may want to have a low-priority project to continuously evaluate adoption options

stuff folks don't talk (enough) about

versioning

- officially, OpenConfig models use semantic versioning
- practically, it's up to each vendor to ensure that something reasonable happens in terms of support and packaging
  - internal resolution of references, etc.
- models are still iterating at a pretty rapid clip
  - make sure you test from version to version against your collection infrastructure
  - while care is taken to limit backwards incompatible updates, some things will change (sometimes in non-obvious ways)

stuff folks don't talk (enough) about (cont'd)

hybrid-schema management

- considerable work remains to be done here
  - <insert previous comments re: model development activity>
- [poor] visibility into (co-)dependencies
  - what does OpenConfig, config - i.e.: realize in terms of translated (vendor-specific) configuration?
  - what is the result if you need to overlay vendor nibbly bits on top of OC applied configuration?
  - will inheritance be preserved?

conclusion - telemetry

come on in …

the water's great(ish)

- significant gains to be had with telemetry and … buzzword["observability"]
- practically, SNMP has been in maintenance mode for years now
- if you really want to scrape your network elements down…
  - there's GET / and you can filter … so that's a win.
  - there's POLL

either way you moved into the 21st century from a tooling perspective

conclusion - config / operations / etc

- config
  - well, let's talk about this …
- gNOI
  - potentially some solid wins
  - check your vendor's gNOI u-service support to see if you can simplify / streamline your ops tooling
- gNSI
  - wait on this for a while, but talk to your vendors re: admin considerations

thanks!