OpenDaylight Summit
July 2015

OpenConfig: collaborating to enable programmable network management

Anees Shaikh
Google Network Operations
on behalf of the OpenConfig working group

www.openconfig.net

Challenges of managing a large-scale network

- 20+ network device roles
- more than half dozen vendors, multiple platforms
- 4M lines of configuration files
- up to ~30K configuration changes per month
- more than 8M OIDs collected every 5 minutes
- more than 20K CLI commands issued and scraped every 5 minutes
- many tools, and multiple generations of software

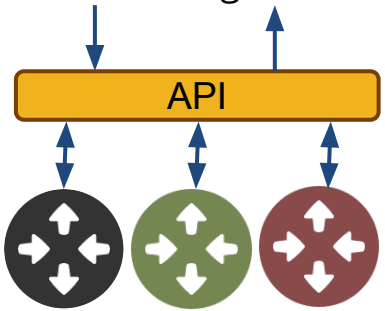
Opportunity for significant OPEX savings: reduced outage impact, simplification of management stack, automation / self-healing, better scaling ...

Network operations in the age of open networking

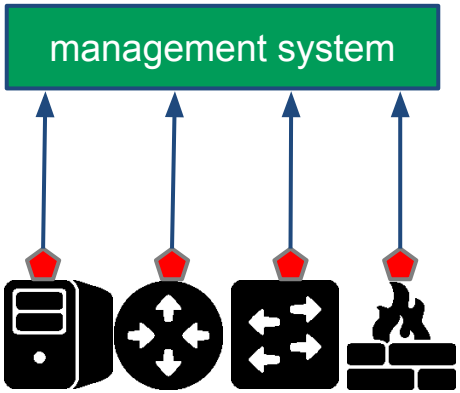
- many proprietary integrations
  - CLIs, scripts, templates, modules, cookbooks, minions, ...
- lack of available abstractions and common APIs
- configuration scraping from devices
- SNMP monitoring -- start with standard, end with enterprise

Elements of an open management plane

common API for configuration and monitoring

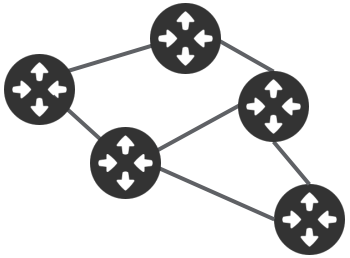


multiple vendor devices



transport and RPC protocols that are open, streaming, secure

interoperable network-wide view



model-driven network management

Configuration

- describes configuration data structure and content

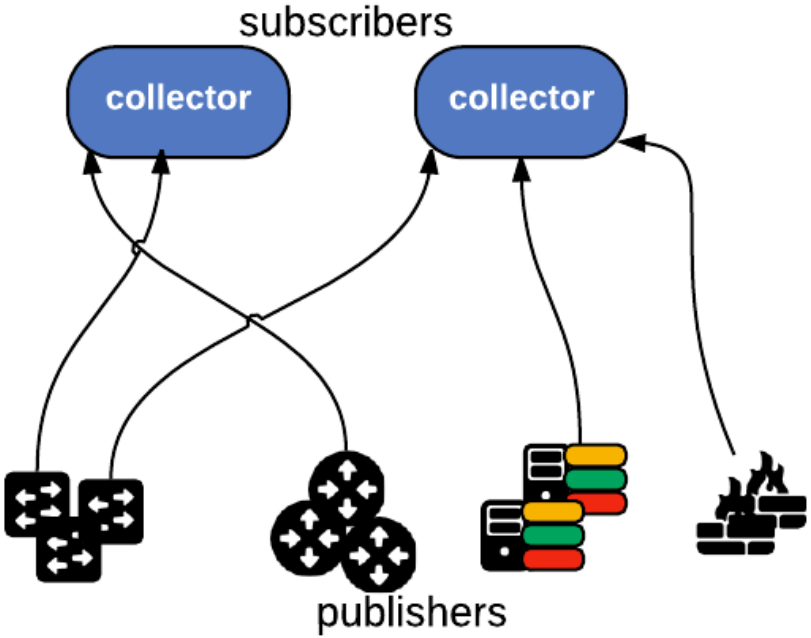
Telemetry

- describes monitoring data structure and attributes

Topology

- describes structure of the network

Telemetry framework requirements



- network elements stream data to collectors (push model)
- data populated based on vendor-neutral models
- pub/sub API to select desired data
- scale for next 10 years of density growth with high data freshness
- modern transport mechanisms with active development communities
  - e.g., gRPC (HTTP/2), Thrift, protobuf over UDP

OpenConfig motivation

- management interfaces are vendor-, platform-, and generation-specific
  - NETCONF / RESTCONF, CIM, SNMP have not solved the problem
  - automation frameworks (Puppet, Chef, Ansible, etc.) do not solve the problem
- complexity and cost have been pushed to operators
  - must build, integrate, and test tools for all these proprietary variations
  - unnecessary differences for configuring and monitoring standard protocols and services
  - specialized skills required to handle proprietary differences

OpenConfig: users defining the APIs

- informal industry collaboration of network operators
- focus: define vendor-neutral configuration and operational state models based on real operations
- primary output is model code, published as open source via public github repo
- partnerships with major vendors to drive native implementations
  - fully supported and maintained as part of the platform software
  - available to all customers, no “specials”
- engagement in standards (IETF, ONF) and OSS projects (ODL, ONOS, NTT)

Why an industry collaboration

- broaden use cases beyond any single operator / customer
- simplification for vendors -- consolidate requirements from customers
- improved models through wide review and an open process
- collective effort to drive model development
- ensure relevance for different management / NMS approaches

OpenConfig participants

broad range of use cases, network environments, vendor deployments, service and business models

OpenConfig governance

short version: there is none

- no board, steering committees, bylaws, ...
  - avoid legal agreements, certifications, etc.
  - rely on good behavior, transparency, and shared goals
- OpenConfig participants join weekly working meetings
  - ‘participants’ == engineers / architects committing and reviewing model code
- raise issues / discuss models on github or mailing lists
- publish model code and tools under an Apache license

“Collaboration innovation”

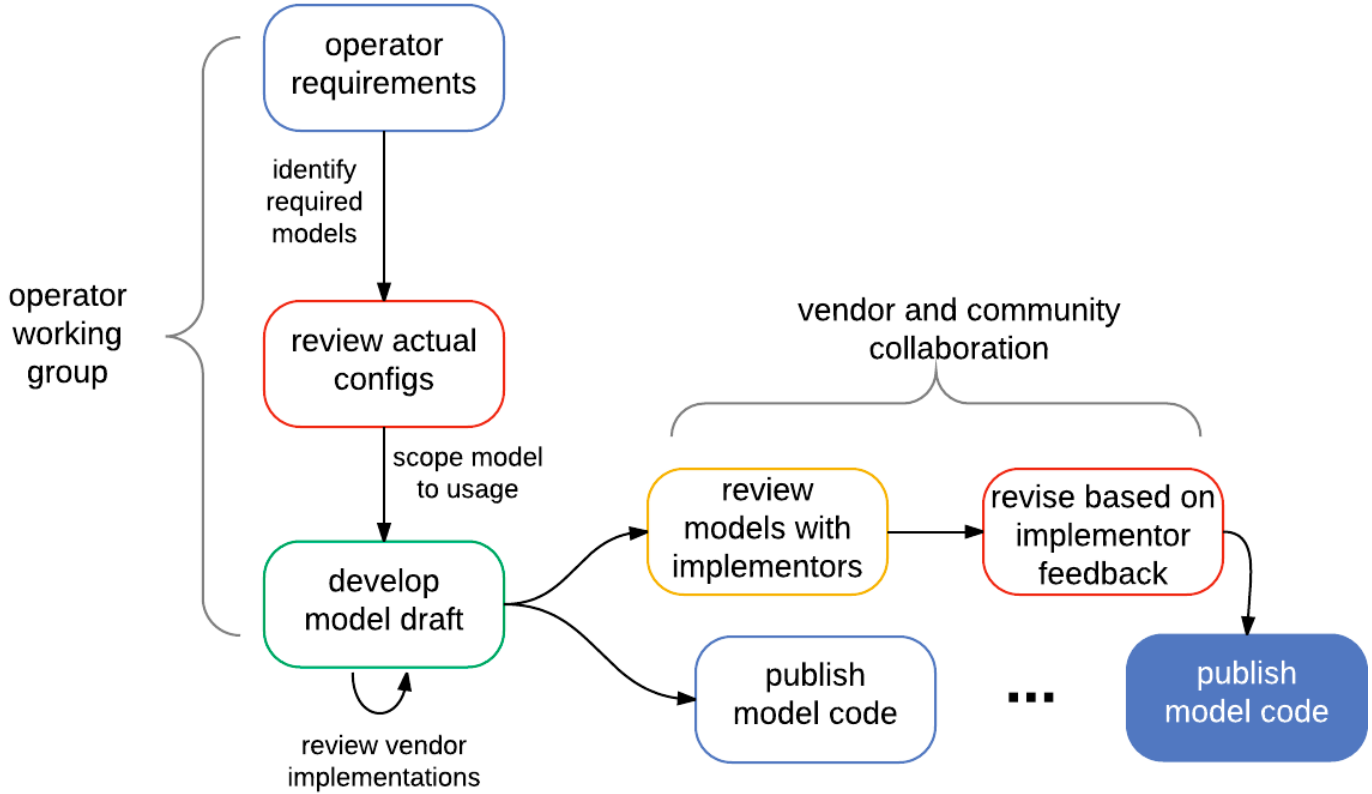
“ The fact that these distinctly different -- and often competitive -- service providers are working together is an indication of the urgency they feel … ”

– LightReading, The New IP, February 2015

“ [OpenConfig] serves to provide a testing ground for working out kinks before turning the specifications over to the official consortia … ”

– siliconAngle, June 2015

OpenConfig development process



OpenConfig progress I

Data models (configuration and operational state)

- BGP and routing policy
  - multiple vendor implementations in progress
  - BGP model adopted by IETF for standards track
- Local routing (locally generated static routes, aggregates, etc.)
- MPLS / TE consolidated model
  - RSVP / TE and segment routing as initial focus
- device model -- common structure for composing models

Design patterns and usability improvements

- design patterns for operational state and model composition
- model catalog proposal

OpenConfig progress II

Models currently in review

- updated interfaces and system models
- RIB model -- represent routing tables in common format
- optical transport devices (transport SDN)

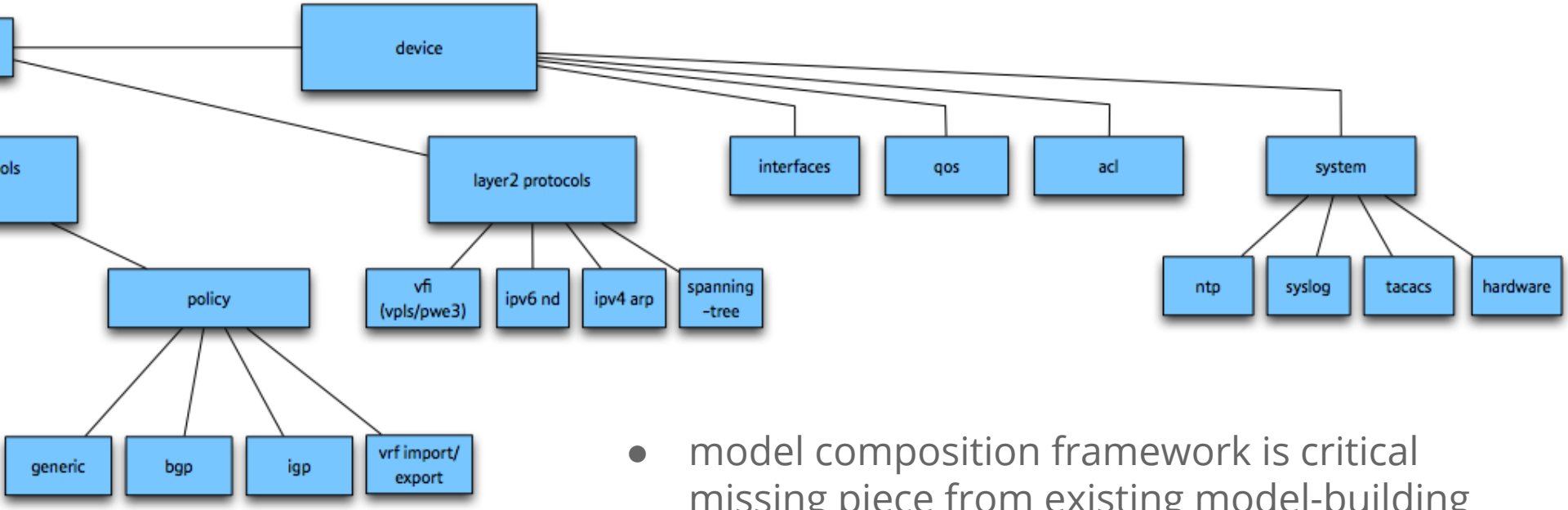
Tools and APIs

- pyangbind -- generates Python classes from YANG models
- protocol-independent specs for config and telemetry RPCs

Native implementations -- BGP+policy models

  - Cisco IOS-XR
  - Juniper JUNOS
  - additional vendors with implementations underway

Models must be composed to be useful



- model composition framework is critical missing piece from existing model-building efforts
- how to build composition support into the modeling language

Modeling operational state

Types of operational state data

- derived, negotiated, set by a protocol, etc. (negotiated BGP hold-time)
- operational state data for counters or statistics (interface counters)
- operational state data representing applied configuration (actual vs. configured)

Clear benefits from using YANG to model both configuration and operational state in the same data model

- provides monitoring data in a common structure across devices
- allows easy association of configuration with corresponding state
- but … YANG focus has primarily been config, NETCONF-centric, lack of common conventions

Observations on YANG / NETCONF

- YANG and NETCONF should be decoupled -- each are independently useful
- YANG needs to evolve more rapidly at this early phase, stabilize as real usage increases
- YANG needs review and input from a much broader set of users with different perspectives
- current YANG model versioning is not helpful -- treat models like software artifacts, not dated documents
- current “standard” models should be open for revisiting and revising; avoid rush to standardize more models until they are deployed and used in production

these are not necessarily OpenConfig consensus views

OpenDaylight opportunities

ODL NMS	• NMS as first-class use case for OpenDaylight platform • encourage additional management and operations focus
OpenConfig support	• use OpenConfig published models as interfaces to OpenDaylight capabilities
YANG tooling and ecosystem	• generalized toolchain for YANG modeling • enable experimentation with modeling language features

some of these are already happening :-)

similar opportunities with other SDN-related OSS projects (e.g., OPNFV, ONOS)

OpenDaylight NMS

- ODL already supports some management / operations features
  - monitoring and path management: SNMP, BGP-LS / PCEP
  - network configuration: NETCONF / RESTCONF, OVSDB
  - data management and modeling: YANG tools, time-series data repo
- Potential additional capabilities
  - streaming telemetry collector, with pub/sub
  - support for additional data transports and encodings
  - configuration validation

Support for OpenConfig models in ODL

Configuration and monitoring APIs based on OpenConfig models

- BGP and routing policy
  - interface to ODL BGP implementation
  - some progress underway (e.g., IETF 93 hackathon)
- MPLS / TE
  - integration with PCEP, segment routing

Developing the YANG ecosystem

- cross validation of YANG tools and models
- make it easier to visualize and experiment with YANG models
- consistency in code artifacts generated from YANG models
  - e.g., class bindings from Java, Python, Go, etc.
- **improvements to the YANG modeling language**
  - user / implementor perspective to complement IETF standard
  - address major shortcomings (lists, versioning, choices, model composition)
    - also see Colin Dixon's ONS 2015 talk on YANG/ODL

Summary

- network management needs a model-driven approach to bring it into the age of SDN and programmable networking
- OpenConfig is a new kind of industry collaboration
  - network operators directly contributing open data models, tools, and design patterns
- as native implementations become available, potential to significantly transform network monitoring and configuration
- major role for OpenDaylight and other OSS projects to help realize the vision

Thank you!