The bridge to possible

Deploying XR Programmability in Production Networks

Mike Korshunov, Provider Connectivity, TME

cisco Live!

BRKSPG-2031

Cisco Webex App

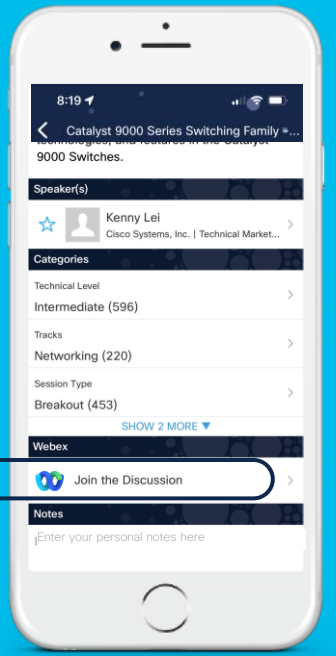
Questions?

Use Cisco Webex App to chat
with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space

Webex spaces will be moderated until February 24, 2023.



Agenda

- Adoption of Programmability in IOS XR
- Configure with Models
- gNMI/gNOI
- Monitor with Telemetry
- Conclusion

Cisco IOS XR – Industry's #1 Network OS

- Feature Releases – 3/year
- Maintenance Releases for Hardening

Cisco NCS 5xx

Cisco NCS 5xxx

Cisco ASR9000

Cisco 8000

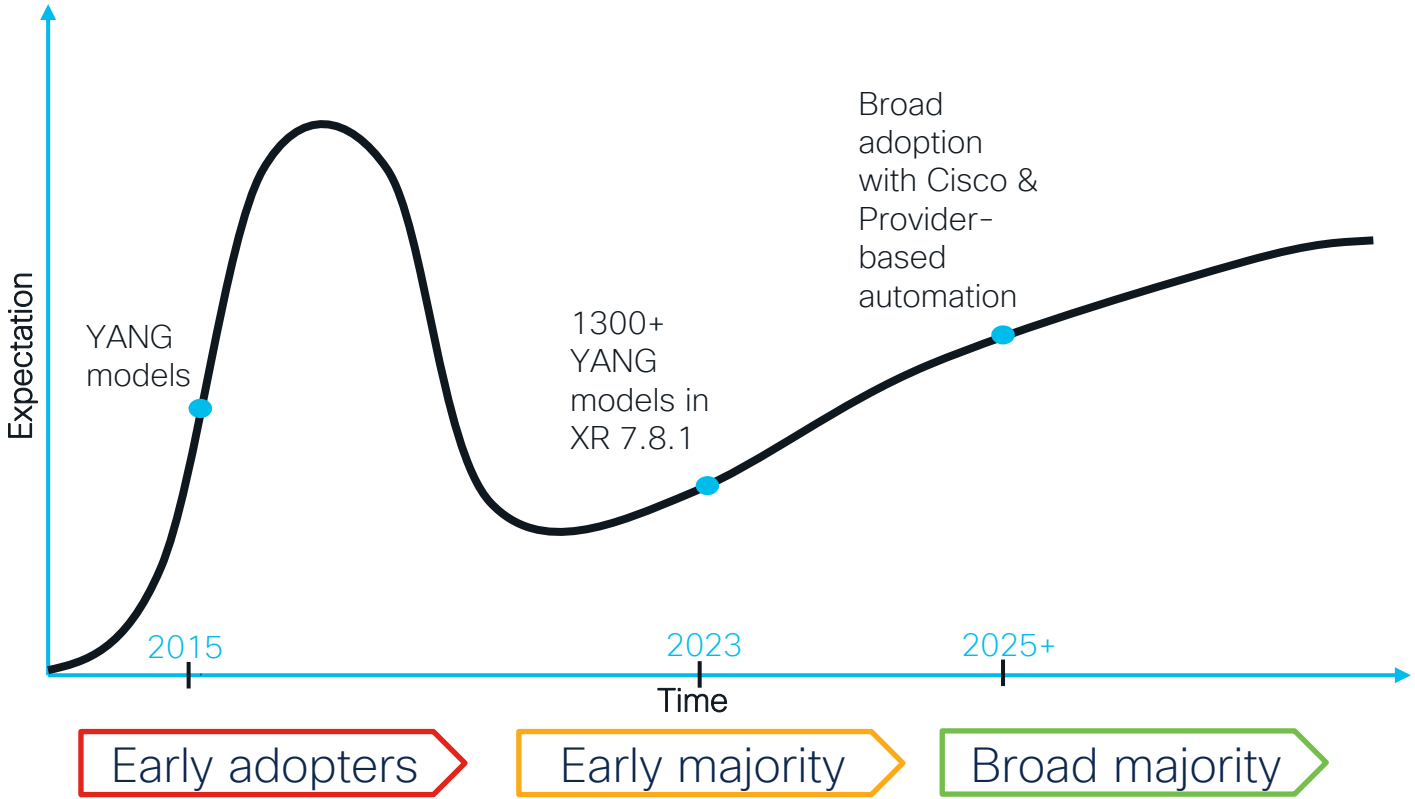
Merchant Silicon

Custom Silicon

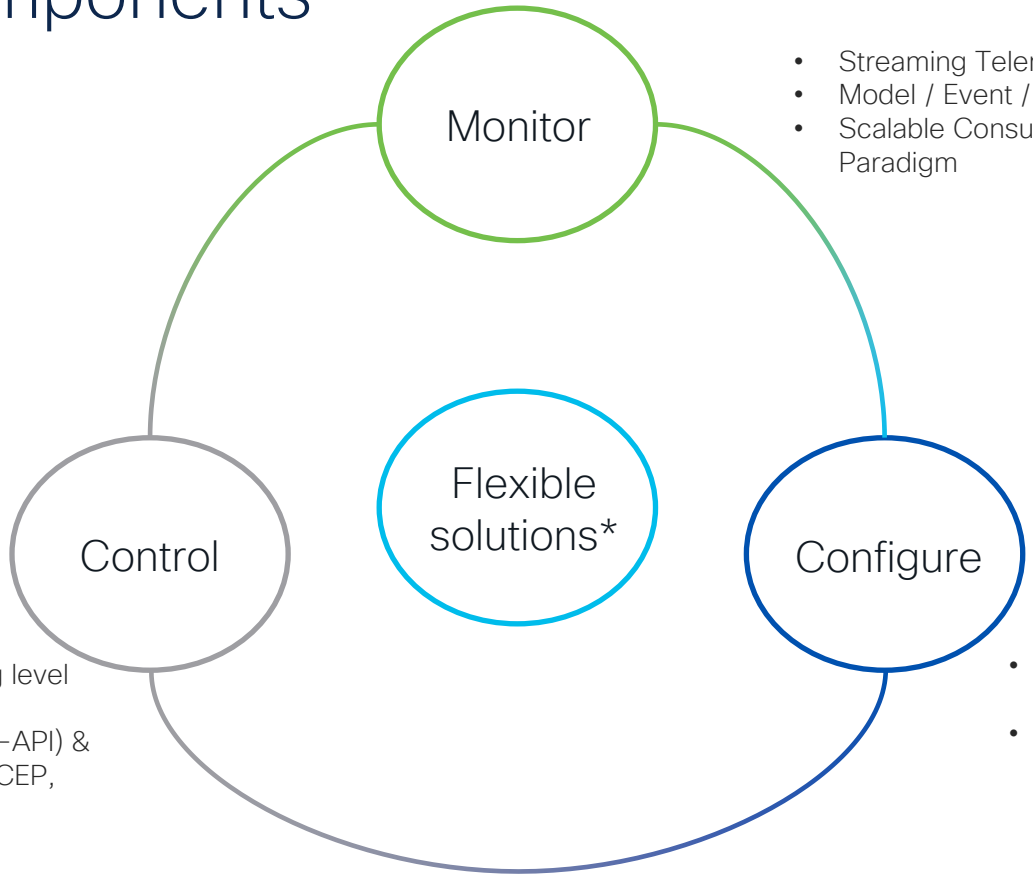
Legacy Platforms – GSR, CRS, NCS6K

Flexible Platforms – Virtual & more...

XR Model-Driven Programmability Evolution



Key Components



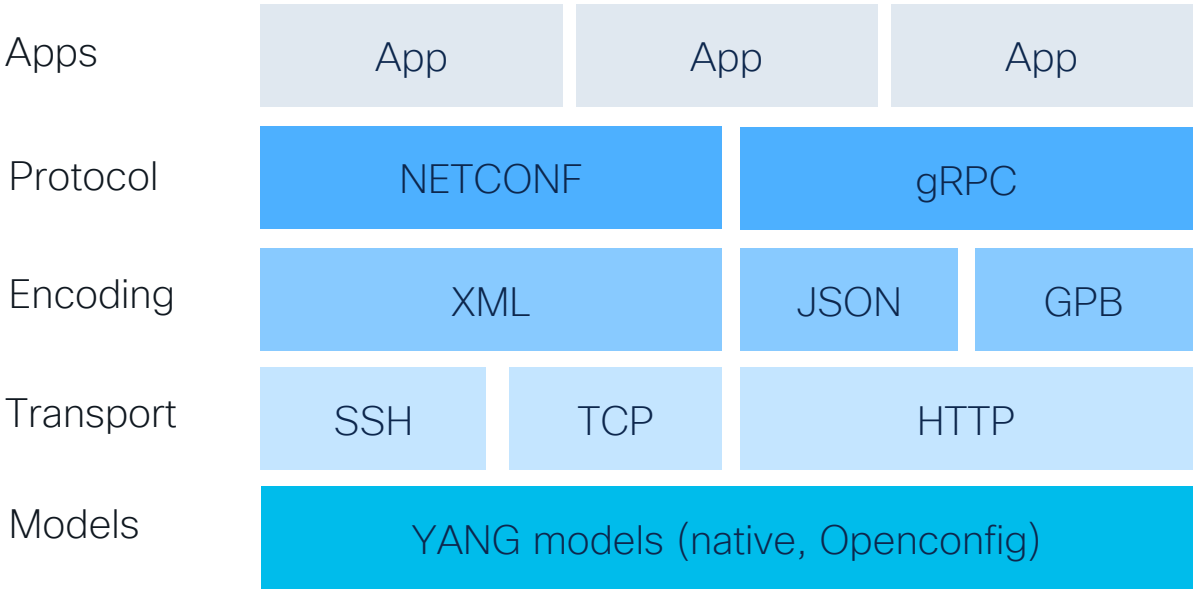
- Streaming Telemetry
- Model / Event / AI Driven
- Scalable Consumption driven Paradigm

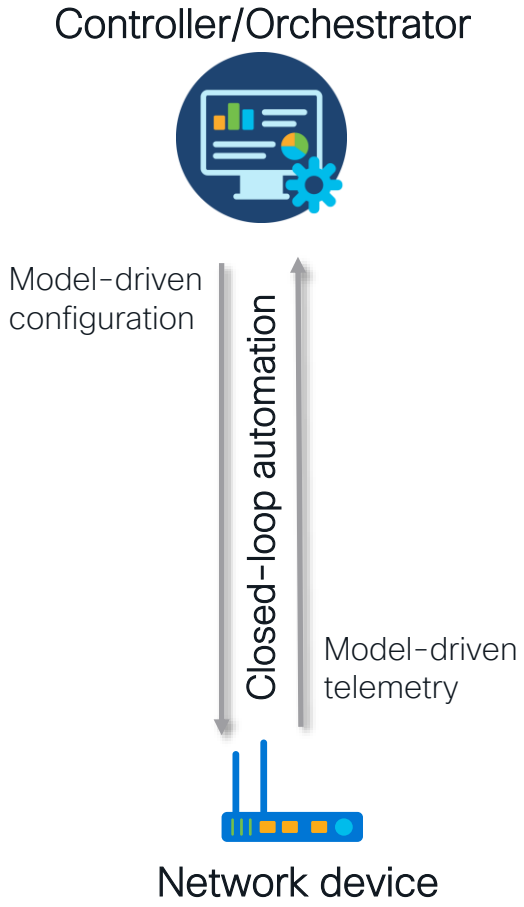
- Route/Forwarding level Control functions
- RIB/FIB Level (SL-API) & Protocol Level (PCEP, BGP FS)

- Device & Network level Configuration
- Scope
  - Day 0 (ZTP)
  - Day 1 (Operate)
  - Day 2 (Service)

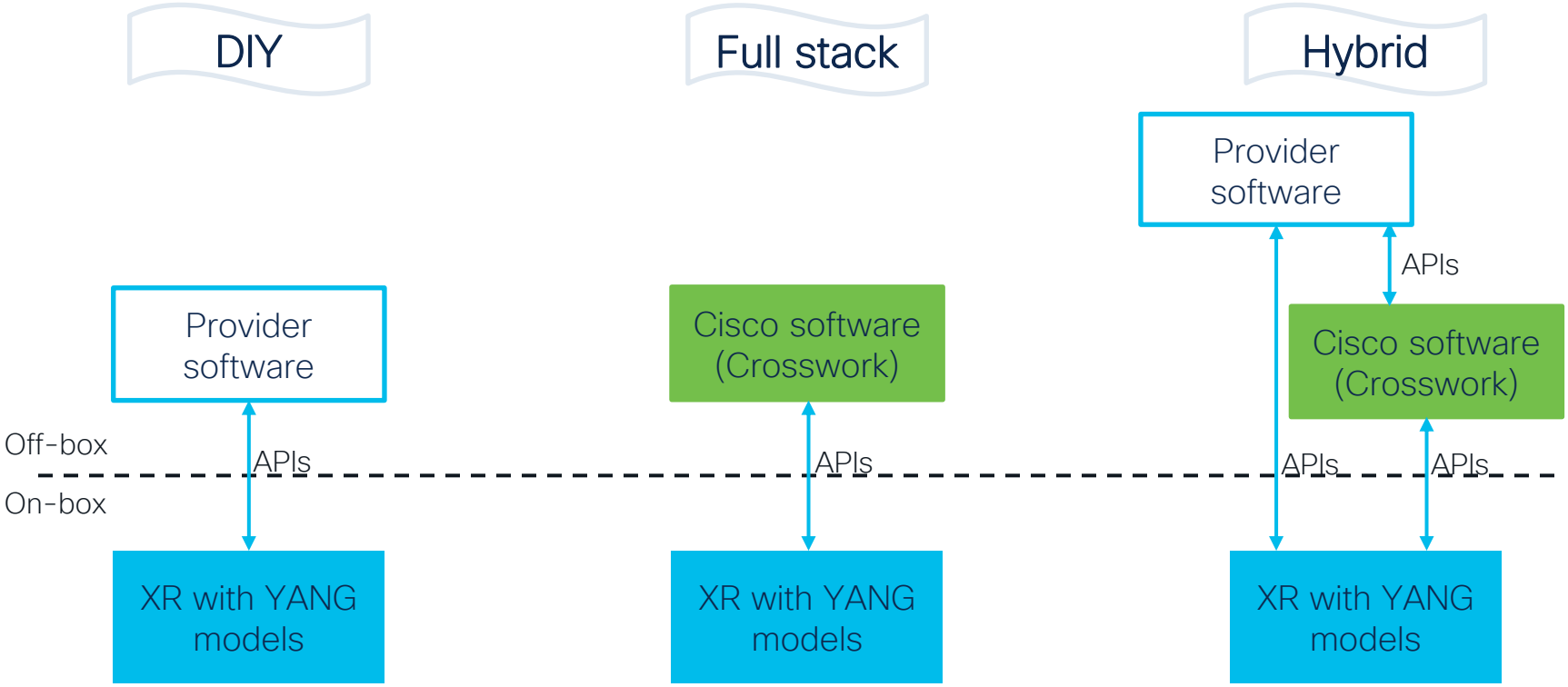
*Flexible solutions: Cisco Internal (Crosswork, NSO) or External (custom-made / 3rd party)

Integration Layers





Customer Deployment Styles



Configure with Models

YANG Model Types

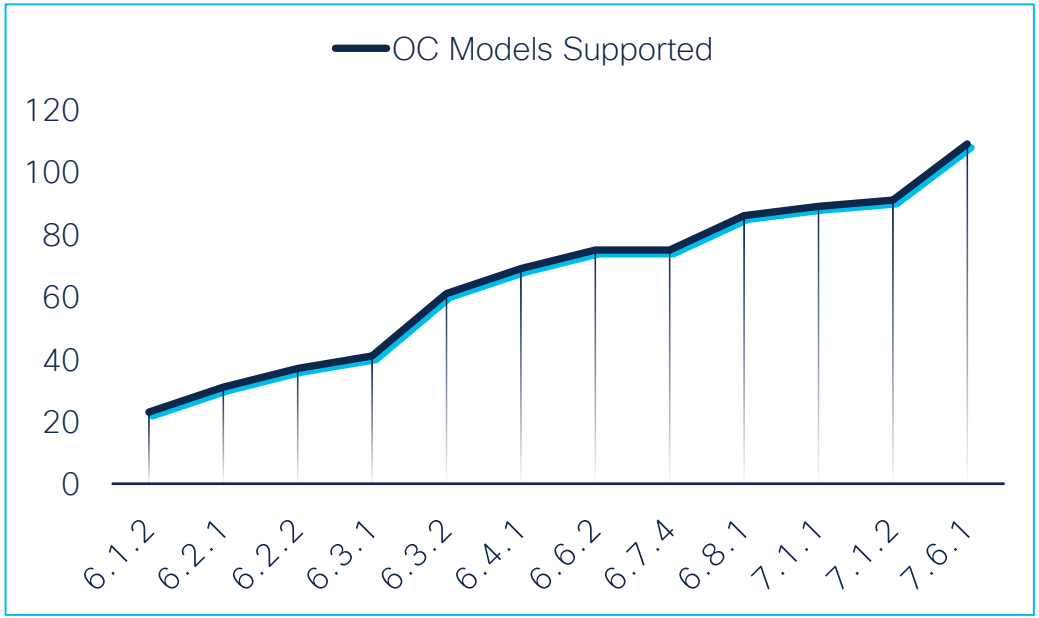
Native models

OpenConfig models

Integrated in IOS XR today (~1300 Native and ~120 OC models – XR 7.6.1)
https://github.com/YangModels/yang/tree/master/vendor/cisco/xr

OpenConfig model support

- Back in 2016 with IOS XR 6 release we started to support OpenConfig Models.
- Fast forward 6 years there are around 120 models supported on XR based devices out of box

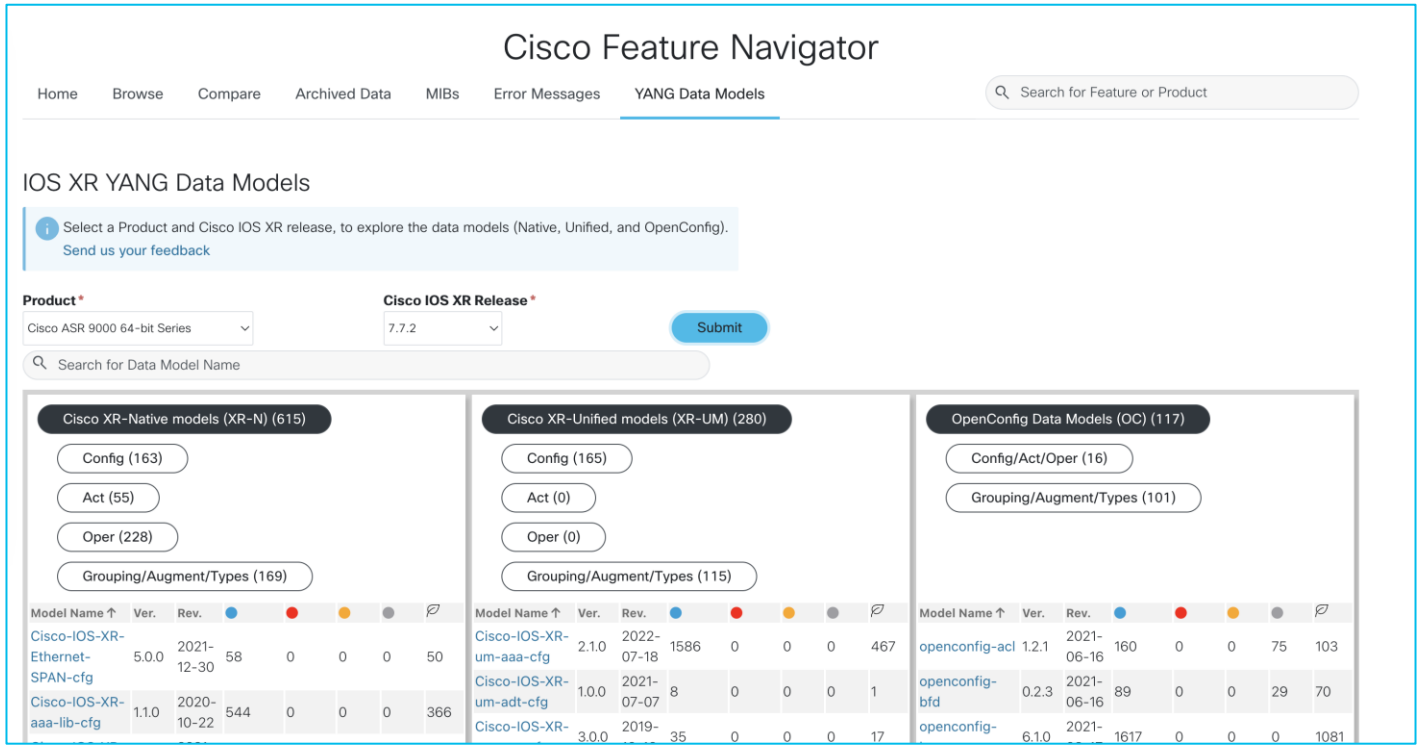


Ways to explore YANG models for IOS XR based routers

- Cisco Feature Navigator. Per product view, no device needed. Release specific. Model can be explored through UI.
- GitHub repo. Initial way to share supported models, it's going to be maintained. No ways to explore models on GitHub directly.
- Cisco YANG Suite or ANX. Two different products, available for free. Install required along with the router. Can get info for specific product from specific device. Advanced functionality to do diffs, explore models and etc.
- Engineering excel files with nitty gritty details.

Cisco Feature Navigator – released in Oct 2022

https://cfnng.cisco.com/



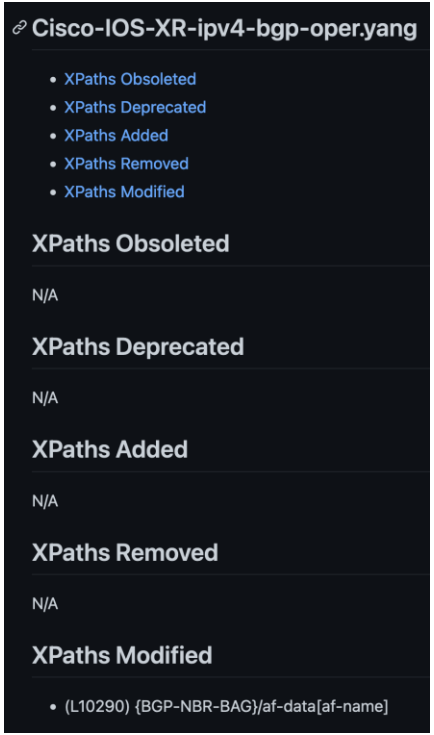
YANG Models Documentation - Github

- List of models per XR release: https://github.com/YangModels/yang/blob/main/vendor/cisco/xr/761/Available-Content.md

- Backwards-incompatible changes based on RFC 6020, Section 10 (since 7.0.2): https://github.com/YangModels/yang/tree/main/vendor/cisco/xr/761/BIC

- Check backwards-incompatibility

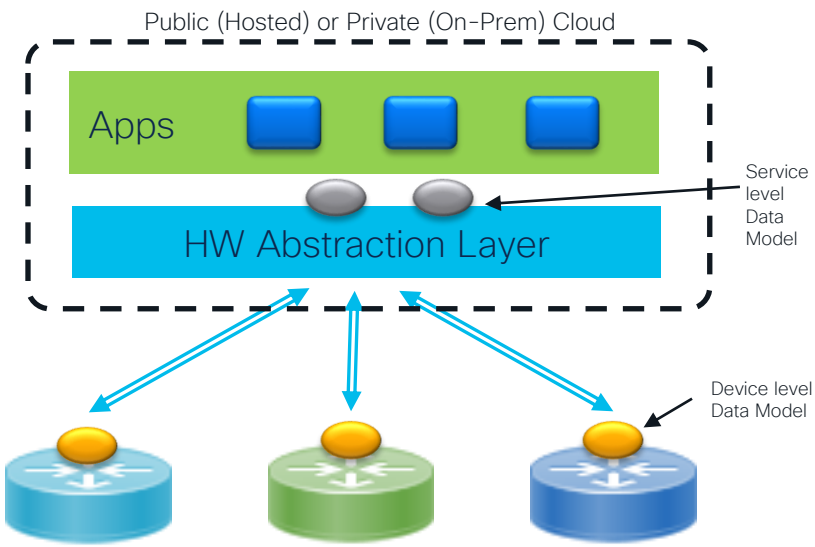
```
$ ./check-models.sh -b 751 # Check incompatibility
between 7.6.1 and 7.5.1
```



XR 7.6.1

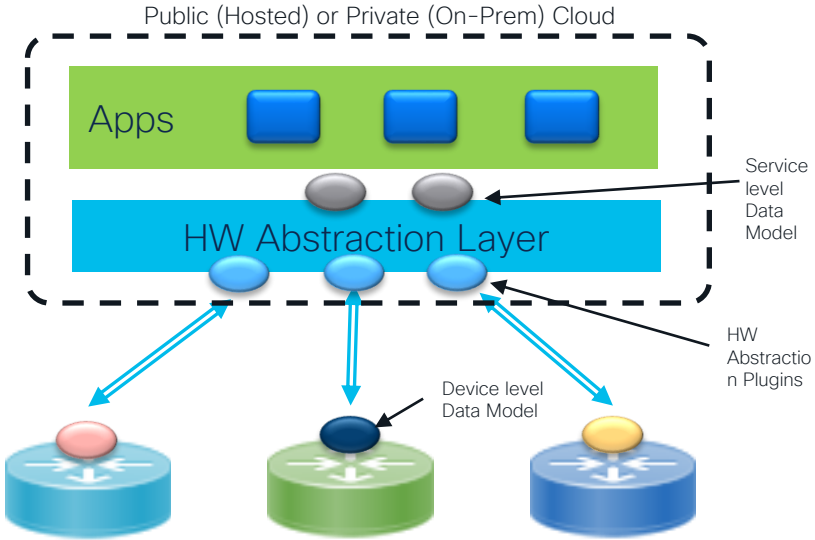
Vendor Neutrality – Options for Providers

Device Level Vendor Neutrality



= E.g. OpenConfig

Network Level Vendor Neutrality



= E.g. NSO NED

Configuration Models

XR Native

Unified

- XR or platform specific
- Full coverage of device functionality
- Single abstraction for YANG and CLI
- Full parity, deterministic coverage
- Same help/doc strings
- Expected to be current

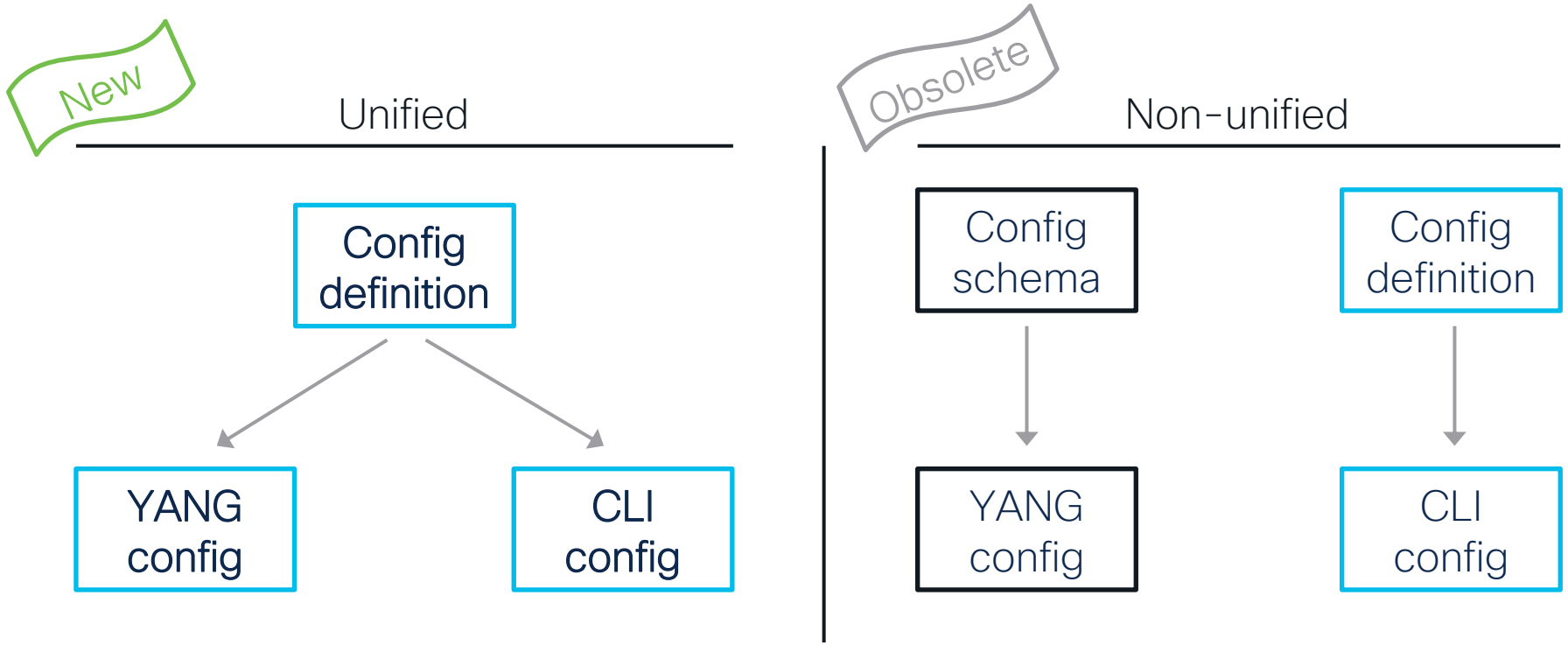
Non-unified

- XR or platform specific
- Full coverage of device functionality
- Different abstractions for YANG and CLI
- Independent testing of parity and coverage
- Expected to be obsolete

OpenConfig

- Generic
- Vendor neutral
- Partial coverage of device functionality
- Different abstraction from native model and CLI

Configuration Models - Unified versus Non-unified



Unified Native Model – BGP Neighbor Group Configuration

```
router bgp 65001
  neighbor-group IBGP
    remote-as 65001
    update-source Loopback0
    address-family ipv4 unicast
    !
  !
!
```

CLI

```
<router xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-um-router-bgp-cfg">
  <bgp>
    <as>
      <as-number>65001</as-number>
      <neighbor-groups>
        <neighbor-group>
          <neighbor-group-name>IBGP</neighbor-group-name>
          <remote-as>65001</remote-as>
          <update-source>Loopback0</update-source>
          <address-families>
            <address-family>
              <af-name>ipv4-unicast</af-name>
            </address-family>
          </address-families>
        </neighbor-group>
      </neighbor-groups>
    </as>
  </bgp>
</router>
```

XML

XR 7.3.2

Usability enhancements: XR Config Scripts

Exec Script - An exec script is a script that gets triggered via a XR-CLI or a rpc over netconf. The script should be able to do whatever a management script can do from an external controller: connect to netconf/gNMI server on the same box, configure the box, query oper data, all model based, or XR CLI driven.

Config Script - A config script is used to enforce that the router configuration adheres to one or more customer-defined constraints. It is triggered automatically during configuration commits, and may either reject the commit (if invalid) or make changes to the contents of the commit (to make the resulting configuration valid).

EEM Script - An EEM (Embedded Event Manager) script is triggered via a predefined set of events. Events supported are syslog, timer, traceback and telemetry events along with logical correlation of events, rate-limit, occurrence and period.

Process Script - A process script will try to run forever as part of its design and it's lifecycle managed by IOS XR.

Documentation

Github: https://github.com/CiscoDevNet/xr-python-scripts

XR 7.4.1

Equivalency Tool (CLI to YANG)

XR tool that shows YANG paths for show and config CLI commands

```
RP/0/RP0/CPU0:Lyon-23#yang-describe operational show int brief
Mon May  9 13:49:32.556 UTC
YANG Paths:
  Cisco-IOS-XR-pfi-im-cmd-oper:interfaces/interface-briefs/interface-brief

RP/0/RP0/CPU0:Lyon-23#yang-describe operational show inventory
Mon May  9 13:46:44.878 UTC
YANG Paths:
  Cisco-IOS-XR-invmgr-oper:inventory/racks/rack/attributes/inv-basic-bag

RP/0/RP0/CPU0:Lyon-23#yang-describe configuration telemetry model-driven
Mon May  9 14:04:58.519 UTC
YANG Paths:
  Cisco-IOS-XR-um-telemetry-model-driven-cfg:telemetry/model-driven
```

Show config in XML format - Unified or OC

XR shows YANG model view for your existing config

```
RP/0/RP0/CPU0:MID-NODE-R4#show run router isis  | xml unified-model
Thu Feb  9 09:04:01.076 UTC
<data>
 <isis xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-clns-isis-cfg">
  <instances>
   <instance>
    <instance-name>1</instance-name>
    <running></running>
    <nets>
     <net>
      <net-name>49.0001.0000.0000.0004.00</net-name>
     </net>
    </nets>
    <distribute/>
    <afs>
     <af>
      <af-name>ipv4</af-name>
      <saf-name>unicast</saf-name>
```

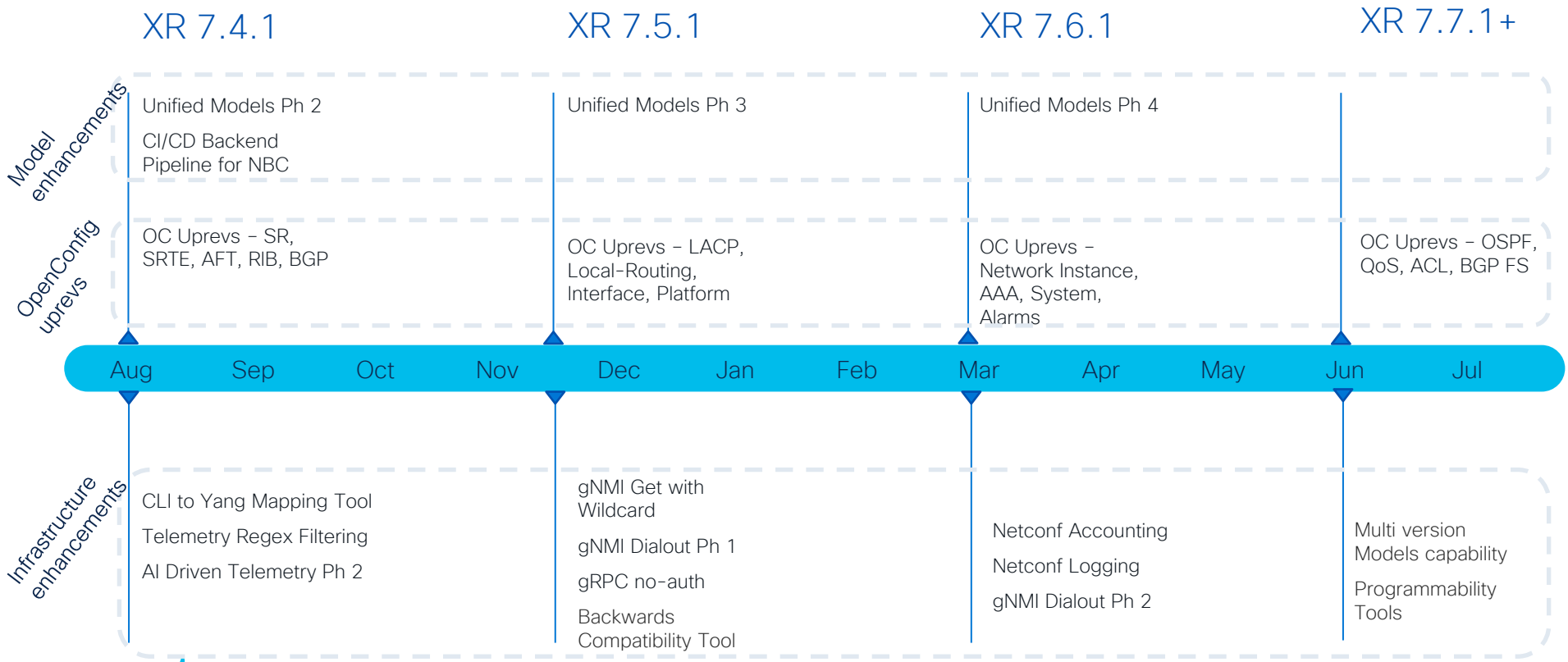
Check Streamed data (run mdt_exec)

XR tool that creates subscription from CLI to check streamed data.

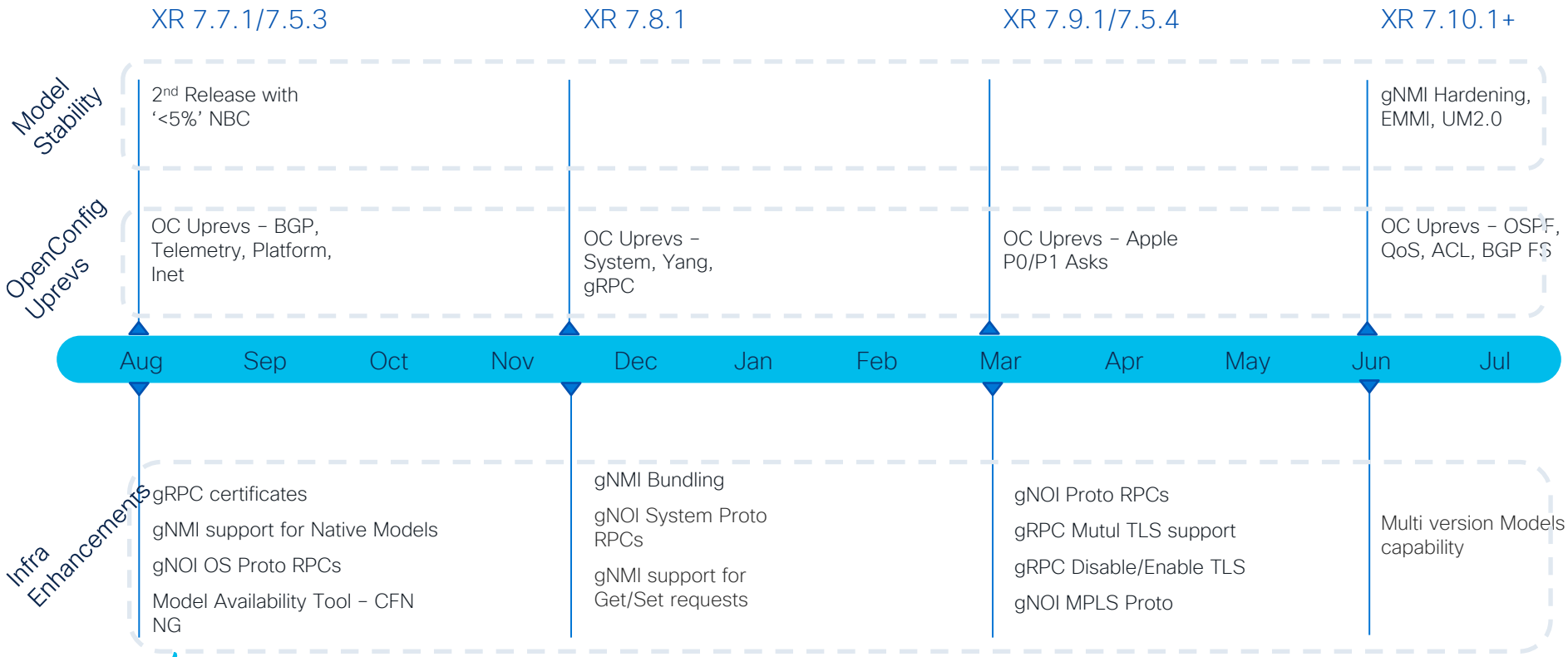
```
RP/0/RP0/CPU0:Macrocarpa#run mdt_exec -s Cisco-IOS-XR-shellutil-oper:system-time/uptime
Thu Dec 13 16:55:52.947 PST
Enter any key to exit...
 Sub_id 200000001, flag 0, len 0
 Sub_id 200000001, flag 4, len 370
--------
{"node_id_str":"NCS5501_top","subscription_id_str":"app_TEST_200000001","encoding_path":"Cisco-IOS-XR-shellutil-oper:system-time/uptime","collection_id":9701112,"collection_start_time":1544748953202,"msg_timestamp":1544748953211,"data_json":[{"timestamp":1544748953209,"keys":[],"content":{"hostname":"NCS5501_top","uptime":546074}}],"collection_end_time":1544748953211}
--------
```

https://xrdocs.io/telemetry/tutorials/2018-08-07-how-to-check-what-will-be-streamed/

XR Roadmap by Focus Areas



Programmability FY23 Roadmap – By Focus Areas



gNMI

gNMI Introduction

- Network management interface defined by OpenConfig (mostly lead by Google)
- Configuration management and streaming telemetry in a single protocol
- Data model independent
- Based on Google RPC framework
- Rich tooling and high performance

The main goal is to provide a “standard” approach for encoding and transport protocols support across different vendors.

gNMI RPCs

- Capabilities – Initial handshake to exchange capability info (e.g. supported data models)
- Set – Modifies data from server (network device)
- Get – Retrieves data on server (network device)
- Subscribe – Control data subscriptions on server (network device)

https://github.com/openconfig/reference/blob/master/rpc/gnmi/gnmi-specification.md

Is It Enough To State gRPC Support?

Cisco gRPC call proto

```
service gRPCConfigOper {
    // Configuration related commands
    rpc GetConfig(ConfigGetArgs) returns(stream ConfigGetReply) {};
    rpc MergeConfig(ConfigArgs) returns(ConfigReply) {};
    rpc DeleteConfig(ConfigArgs) returns(ConfigReply) {};
    rpc ReplaceConfig(ConfigArgs) returns(ConfigReply) {};
    rpc CliConfig(CliConfigArgs) returns(CliConfigReply) {};
    rpc CommitReplace(CommitReplaceArgs)
      returns (CommitReplaceReply) {};
    // Do we need implicit or explicit commit
    rpc CommitConfig(CommitArgs) returns(CommitReply) {};
    rpc ConfigDiscardChanges(DiscardChangesArgs)
      returns(DiscardChangesReply) {};
    // Get only returns oper data
    rpc GetOper(GetOperArgs) returns(stream GetOperReply) {};
    // Get Telemetry Data
    rpc CreateSubs(CreateSubsArgs) returns(stream CreateSubsReply)
{};
}
```

https://github.com/cisco/bigmuddy-network-telemetry-proto/blob/master/staging/mdt_grpc_dialin/mdt_grpc_dialin.proto

Juniper gRPC call proto

```
service OpenConfigTelemetry {
    // Request an inline subscription for data at the specified path.
    // The device should send telemetry data back on the same
    // connection as the subscription request.
    rpc telemetrySubscribe(SubscriptionRequest)
      returns (stream OpenConfigData) {}
    // Terminates and removes an exisiting telemetry subscription
    rpc cancelTelemetrySubscription(CancelSubscriptionRequest)
      returns (CancelSubscriptionReply) {}
    // Get the list of current telemetry subscriptions from the
    // target. This command returns a list of existing subscriptions
    // not including those that are established via configuration.
    rpc getTelemetrySubscriptions(GetSubscriptionsRequest)
      returns (GetSubscriptionsReply) {}
    // Get Telemetry Agent Operational States
    rpc getTelemetryOperationalState(GetOperationalStateRequest)
      returns (GetOperationalStateReply) {}
    // Return the set of data encodings supported by the device for telemetry
    rpc getDataEncodings(DataEncodingRequest)
      returns (DataEncodingReply) {}
```

https://github.com/Juniper/jtimon/blob/master/telemetry/telemetry.proto

GNMI Should Be The Answer. Right?

Defines the gRPC call

```
service gNMI {
  // Capabilities allows the client to retrieve the set of capabilities that
  // is supported by the target. This allows the target to validate the
  // service version that is implemented and retrieve the set of models that
  // the target supports. The models can then be specified in subsequent RPCs
  // to restrict the set of data that is utilized.
  // Reference: gNMI Specification Section 3.2
  rpc Capabilities(CapabilityRequest) returns (CapabilityResponse);
  // Retrieve a snapshot of data from the target. A Get RPC requests that the
  // target snapshots a subset of the data tree as specified by the paths
  // included in the message and serializes this to be returned to the
  // client using the specified encoding.
  // Reference: gNMI Specification Section 3.3
  rpc Get(GetRequest) returns (GetResponse);
  // Set allows the client to modify the state of data on the target. The
  // paths to modified along with the new values that the client wishes
  // to set the value to.
  // Reference: gNMI Specification Section 3.4
  rpc Set(SetRequest) returns (SetResponse);
  // Subscribe allows a client to request the target to send it values
  // of particular paths within the data tree. These values may be streamed
  // at a particular cadence (STREAM), sent one off on a long-lived channel
  // (POLL), or sent as a one-off retrieval (ONCE).
  // Reference: gNMI Specification Section 3.5
  rpc Subscribe(stream SubscribeRequest) returns (stream SubscribeResponse);
}
```

Defines the message

```
message Update {
  Path path = 1;                        // The path (key) for the update.
  Value value = 2 [deprecated=true];    // The value (value) for the update.
  TypedValue val = 3;                   // The explicitly typed update value.
  uint32 duplicates = 4;                // Number of coalesced duplicates.
}

// TypedValue is used to encode a value being sent between the client and
// target (originated by either entity).
message TypedValue {
  // One of the fields within the val oneof is populated with the value
  // of the update. The type of the value being included in the Update
  // determines which field should be populated. In the case that the
  // encoding is a particular form of the base protobuf type, a specific
  // field is used to store the value (e.g., json_val).
  oneof value {
    string string_val = 1;              // String value.
    int64 int_val = 2;                  // Integer value.
    uint64 uint_val = 3;                // Unsigned integer value.
    bool bool_val = 4;                  // Bool value.
    bytes bytes_val = 5;                // Arbitrary byte sequence value.
    float float_val = 6;                // Floating point value.
    Decimal64 decimal_val = 7;          // Decimal64 encoded value.
    ScalarArray leaflist_val = 8;       // Mixed type scalar array value.
    google.protobuf.Any any_val = 9;    // protobuf.Any encoded bytes.
    bytes json_val = 10;                // JSON-encoded text.
    bytes json_ietf_val = 11;           // JSON-encoded text per RFC7951.
    string ascii_val = 12;              // Arbitrary ASCII text.
```

gNOI

GNOI Intro

- Suite of microservices – each corresponding to a set of operations.
  - Allows adoption of only the services that the device supports.
  - Reflection service (in gRPC library) can be used to discover which services a device supports.
- Growing coverage:
  - BGP, Certificate management, MPLS, interface, layer 2, system (ping, traceroute etc.)
- Natively described in protobuf.
  - No YANG model for operations contents.
  - Path within data tree used to relate to other state on the device.

gNOI Support – Cisco IOS XR

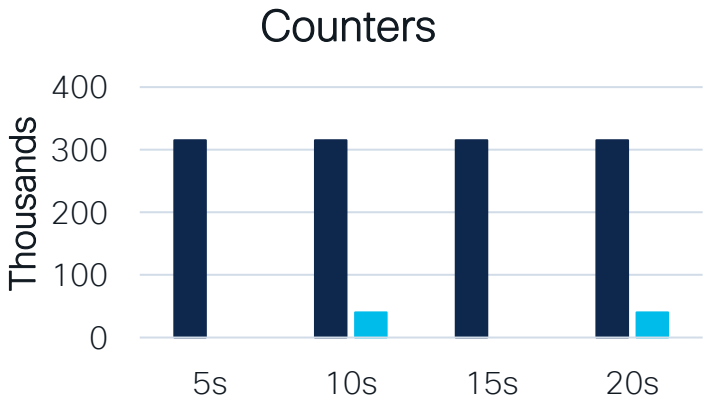
XR 7.0.1 & Prior	XR 7.1.1	XR 7.2.1	XR 7.3 .1	XR 7.5.2	In Development
System • Reboot • RebootStatus • SetPackage **File** • Get • Remove	**System** • Ping • Traceroute • Time • SwitchControlProcessor **File** • Put • Stat **Cert(0.1.0)** • Rotate • Install • GetCertificates • Revoke • CanGenerateCSR	BGP • ClearBGPNeighbor Layer2 • ClearLLDPInterface Interface • SetLoopbackMode • GetLoopbackMode • ClearInterfaceCounters	Diag • StartBERT • StopBERT • GetBERT MPLS • ClearLSP • ClearLSPCounters • MPLSPing OS • Install • Activate • Verify	Cert(0.2.0) • Install • LoadCertificateAuthorityBundle • GetCertificates • RevokeCertificates • CanGenerateCSR	MPLS • ClearLSP • ClearLSPCounters • MPLSPing Active discussions on other RPCs

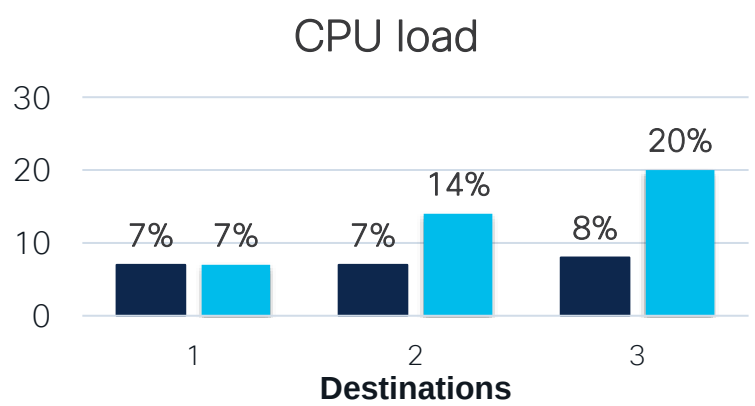
gNOI Support – Cisco IOS XR

XR 7.0.1 & Prior	XR 7.1.1	XR 7.2.1	XR 7.3 .1	XR 7.5.2	XR 7.8.1 & Beyond
System • Reboot • RebootStatus • SetPackage **File** • Get • Remove	**System** • Ping • Traceroute • Time • SwitchControlProcessor **File** • Put • Stat **Cert(0.1.0)** • Rotate • Install • GetCertificates • Revoke • CanGenerateCSR	BGP • ClearBGPNeighbor Layer2 • ClearLLDPInterface Interface • SetLoopbackMode • GetLoopbackMode • ClearInterfaceCounters	Diag • StartBERT • StopBERT • GetBERT MPLS • ClearLSP • ClearLSPCounters • MPLSPing OS • Install • Activate • Verify	Cert(0.2.0) • Install • LoadCertificateAuthorityBundle • GetCertificates • RevokeCertificates • CanGenerateCSR	MPLS(Unsupported) • ClearLSP • ClearLSPCounters • MPLSPing

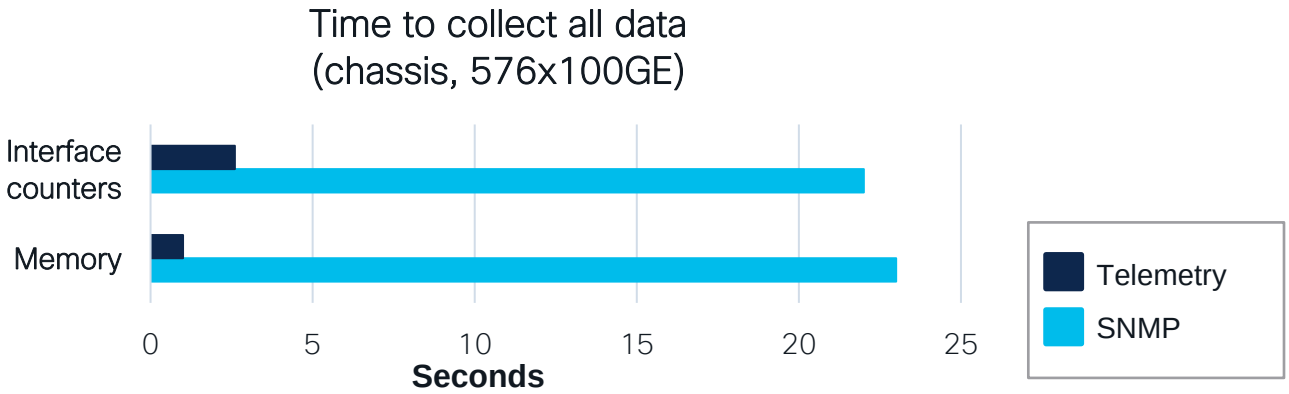
Monitor with Telemetry

MDT versus SNMP – No Competition





- ✓ More counter data
- ✓ Reduction in CPU load
- ✓ Faster collection



cisco Live!

Configuration is easy

```
telemetry model-driven
 destination-group DGroup1
  address-family ipv4 10.30.110.38 port 5432
   encoding self-describing-gpb
   protocol tcp
  !
  !
  address-family ipv6 2001::1 port 2345
   encoding json
   protocol grpc no-tls
```

WHERE and HOW

```
!
!
sensor-group SGroup1
 sensor-path Cisco-IOS-XR-qos-ma-oper:qos/interface-table/interface/input/statistics
 sensor-path Cisco-IOS-XR-infra-statsd-oper:infra-
statistics/interfaces/interface/latest/generic-counters
!
!
```

WHAT

```
subscription Sub1
 sensor-group-id SGroup1 sample-interval 1000
 destination-id DGroup1
```

HOW OFTEN

No need to install any package or license!
IOS-XR is fully ready to run telemetry out of the box!

Examples with Native Models:
https://goo.gl/Adjcgu
Examples with OC Models:
https://goo.gl/Ubb2ZN

Encoding Options

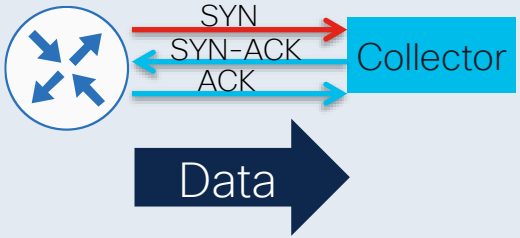
Vary in Efficiency and Ease of Use

Encoding	Description	Wire Efficiency	Other Considerations
GPB (-Compact)	Everything binary (except values that are strings)	High	Proto file per model.
GPB-KV	String keys and binary values (except values that are strings)	Medium Low	Single .proto file for decoding. Can use GPB tooling.
JSON	Everything strings: keys and values	Low	Friendly. Human readable, easy for humans and code to parse

Strings are readable and self-describing but not efficient!

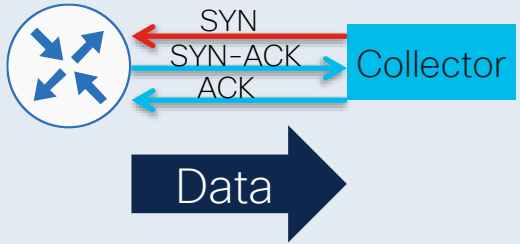
Transport Options

Dial-Out



- TCP & gRPC (from 6.1.1)
- UDP (from 6.2.1)

Dial-In

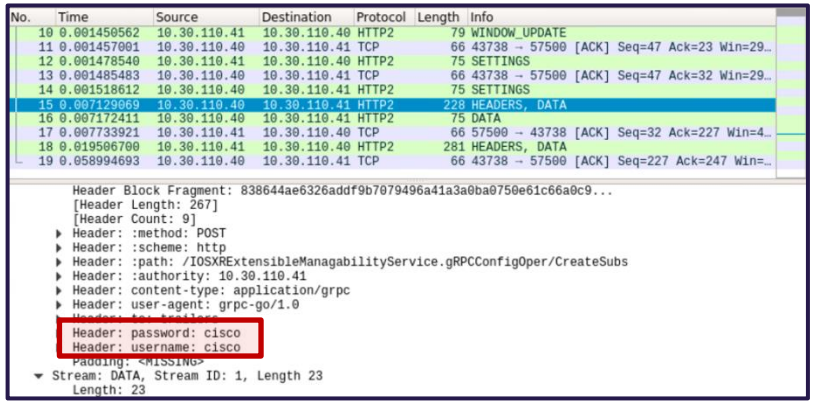


- gRPC only (from 6.1.1)
- gNMI (from 6.5.1)

Security is a Very Popular Question

gRPC Dial-in (NO-TLS)

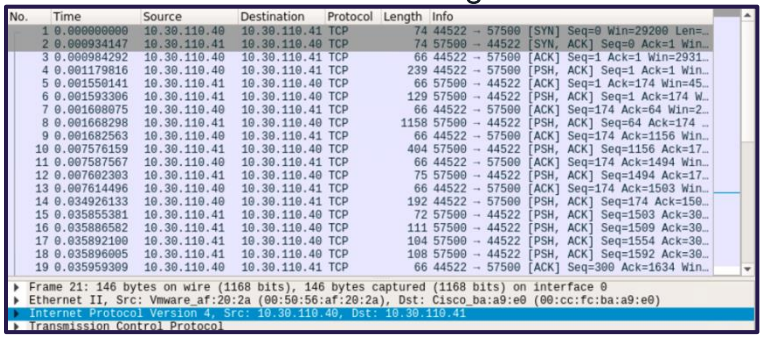
Password exchange



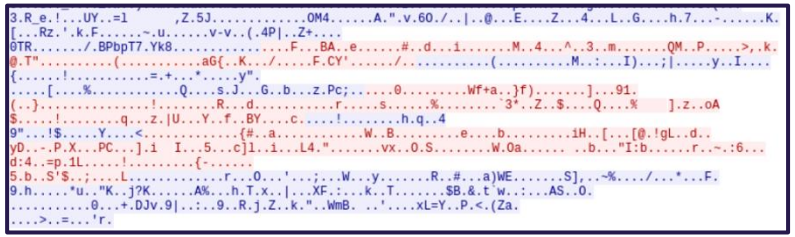
Message content

gRPC Dial-in (TLS)

Password exchange



Message content



Telemetry – Best Practices

Telemetry requires Collector based architectures

- Limited processing of data on the router due to limited compute
- Generic Server compute with Data Lake type approach

Key factors for scaling Telemetry

- Cadence – Interval between Sensor path updates
- Interfaces/Sensor paths – Amount of data to be streamed out of each device
- Devices to Collector Ratio
  - Compute or Bandwidth should not be constrained
  - Distributed Collectors across the network

Beachhead Deployments

- In production: ”Metro Area” with 30 Devices per Collector with aggregate of 1000 interfaces with 30 second cadence
- In discussions: Varied Devices/Collector ratio at 1–10 min cadence

Migrating from SNMP?

- Check this document: https://www.cisco.com/c/en/us/support/docs/ios-nx-os-software/ios-xr-software/216947-snmp-migration-to-telemetry-on-ios-xr.html
- It covers basic steps on migration to Streaming Telemetry

ifSpeed	1.3.6.1.2.1.2.2.1.5	An estimate of the interface's current bandwidth in bits per second. For interfaces which do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth. If the bandwidth of the interface is greater than the maximum value reportable by this object then this object should report its maximum value (4,294,967,295) and ifHighSpeed must be used to report the interace's speed. For a sub-layer which has no concept of bandwidth, this object should be zero.	Cisco-IOS-XR-pfi-im-cmd-oper:interfaces/interface-xr/interface/bandwidth
ifOperStatus	1.3.6.1.2.1.2.2.1.8	The current operational state of the interface. The testing(3) state indicates that no operational packets can be passed. If ifAdminStatus is down(2) then ifOperStatus should be down(2). If ifAdminStatus is changed to up(1) then ifOperStatus should change to up(1) if the interface is ready to transmit and receive network traffic; it should change to dormant(5) if the interface is waiting for external actions (such as a serial line waiting for an incoming connection); it should remain in the down(2) state if and only if there is a fault that prevents it from going to the up(1) state; it should remain in the notPresent(6) state if the interface has missing (typically, hardware) components.	Cisco-IOS-XR-pfi-im-cmd-oper:interfaces/interface-non-dynamics/interface-non-dynamic/oper-state

Sample OID to sensor-path match

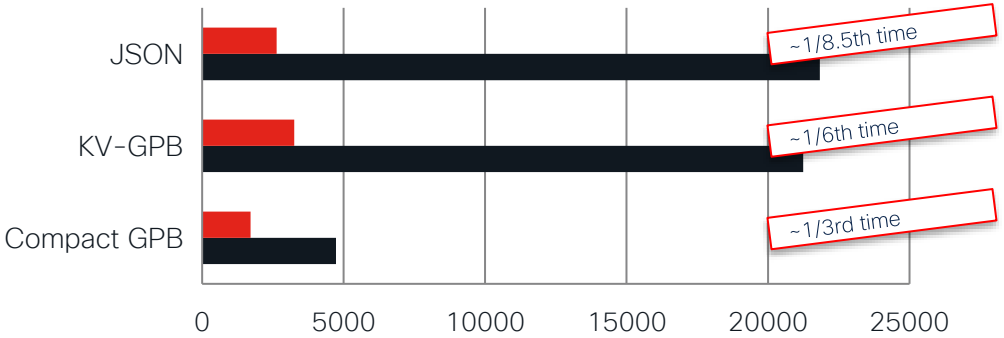
Deploy

XR 7.1.2 / 7.2.1

gRPC Compression

- Support for compression has been added to XR gRPC implementation
- No configuration required for gNMI clients
  - Clients use CallOption **UseCompressor**
- New configuration under **`protocol grpc`**, per destination **(dial-out)**

```
telemetry model-driven
 destination-group notls
  address-family ipv4 192.168.122.1 port 9902
   encoding self-describing-gpb
   protocol grpc no-tls gzip
  !
 !
!
```



Red – with compression, Black – without compression
Y axis – encodings w / w/o compression, X axis – milliseconds on transport

XR 7.2.1

Leaf-level filtering

- Current subscriptions are internally mapped to the corresponding container (gather path)
- New feature: allow subscription at individual leaf level
  - Multiple leaves can be specified in a single subscription
  - Optimized to avoid duplicate internal collections

```
telemetry model-driven
  sensor-group intf-stats
    sensor-path Cisco-IOS-XR-pfi-im-cmd-oper:interfaces/interface-xr/
      interface/interface-statistics/full-interface-stats/bytes-sent
    !
  !
!
```

XR 7.5.1

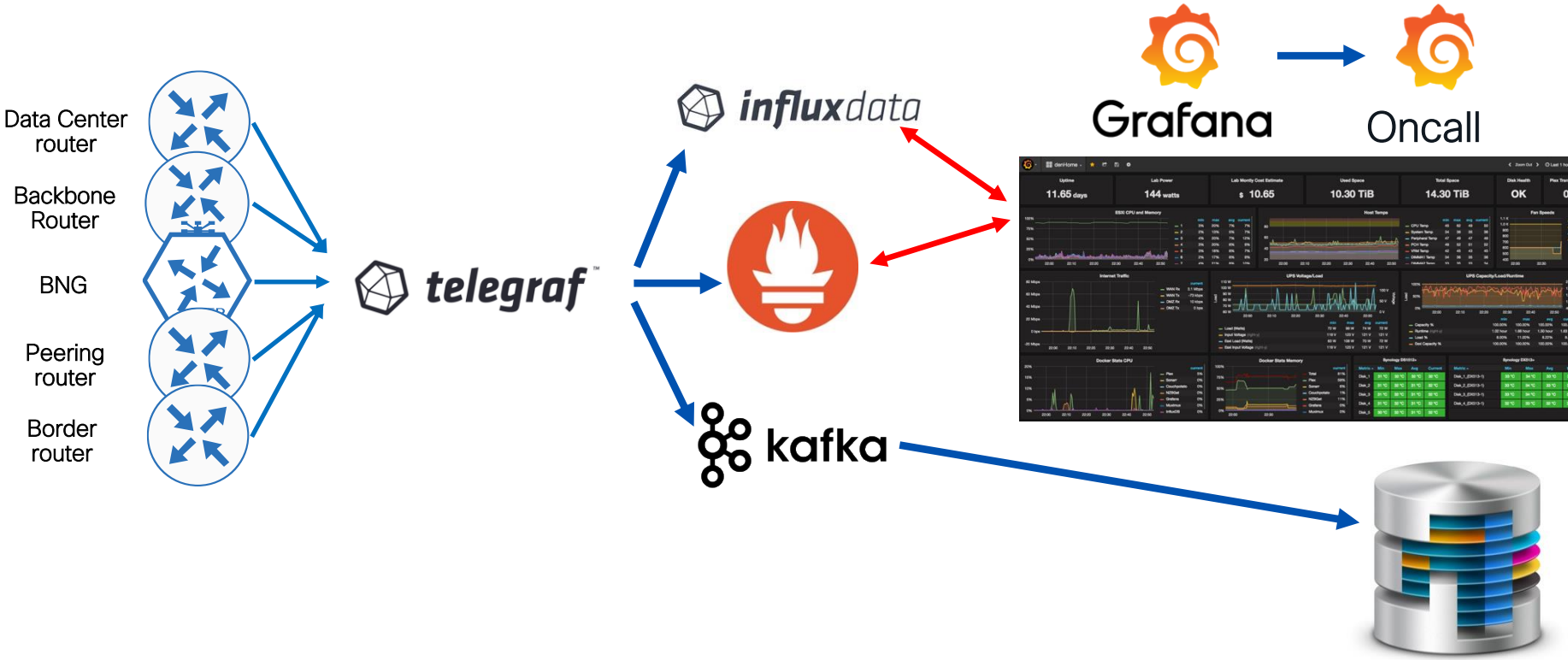
GNMI Dialout

- There are several use cases that dictate the need to reverse the direction in which a TCP session is established between a client and server.
- Configuration sample below:

```
RP/0/RP0/CPU0:macrocarpa#sh run grpc
Fri Jun 25 19:37:21.015 UTC
grpc
 port 57500
 no-tls
 tunnel
  destination 5.0.0.2 port 59510
   target TEST
   source-interface GigabitEthernet0/0/0/1
  !
  destination 2002::1:2 port 59510
    source-interface GigabitEthernet0/0/0/0
  destination 192.168.122.1 port 59500
  destination 192.168.122.1 port 59600
```

Start Exploring Telemetry Today

Go With Open Source Tools

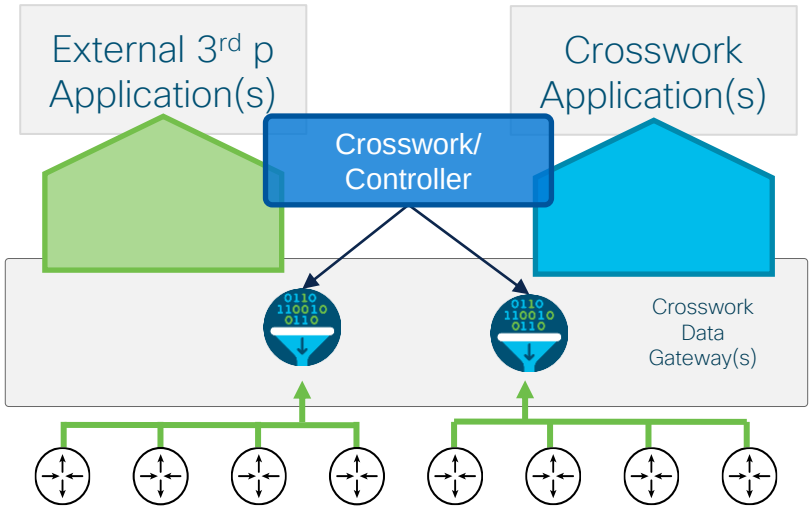


Collector Options: Telegraf

- Telegraf is the open-source server agent to help you collect metrics from your stacks, sensors, and systems.
- Multiple plugins to use: cisco_telemetry_mdt, gnmi & snmp

```
[[inputs.cisco_telemetry_mdt]]
 ## Telemetry transport can be "tcp" or "grpc".  TLS is only supported when
 ## using the grpc transport.
 transport = "tcp"

 ## Address and port to host telemetry listener
 service_address = ":57100"

 ## Grpc Maximum Message Size, default is 4MB, increase the size.
 max_msg_size = 4000000

 ## Enable TLS; grpc transport only.
 # tls_cert = "/etc/telegraf/cert.pem"
 # tls_key = "/etc/telegraf/key.pem"

 ## Enable TLS client authentication and define allowed CA certificates; grpc
 ##  transport only.
 # tls_allowed_cacerts = ["/etc/telegraf/clientca.pem"]
```

Collector Options: Crosswork Data Gateway



- Crosswork Data Gateway is part of the Crosswork Automation platform
- Crosswork Data Gateway provides a simple secure gateway between the network and applications to collect data even for applications in the cloud
- It is designed to be multi-vendor, multi-protocol, to reduce the need for multiple collection points to the network
- By scaling horizontally and offloading application logic into it, Data Gateway can help applications achieve mass scale deployments

Sensor paths used for basic tests

Data	Model
Interface Oper State	sensor-path Cisco-IOS-XR-pfi-im-cmd-oper:interfaces/interface-xr/interface
Interface Data Rate	sensor-path Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/latest/data-rate
Interfaces Stats	sensor-path Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface/latest/generic-counters
Optics Ports Info	sensor-path Cisco-IOS-XR-controller-optics-oper:optics-oper/optics-ports/optics-port/optics-Info
Uptime Info	sensor-path Cisco-IOS-XR-shellutil-oper:system-time/uptime
CPU State	sensor-path Cisco-IOS-XR-wdsysmon-fd-oper:system-monitoring/cpu-utilization
Memory Info	sensor-path Cisco-IOS-XR-nto-misc-oper:memory-summary/nodes/node/summary
Processes Memory	sensor-path Cisco-IOS-XR-procmem-oper:processes-memory/nodes
LLDP Info	sensor-path Cisco-IOS-XR-ethernet-lldp-oper:lldp/nodes/node/neighbors/summaries/summary
IPv4 RIB Info	sensor-path Cisco-IOS-XR-ip-rib-ipv4-oper:rib/vrfs/vrf/afs/af/safs/saf/ip-rib-route-table-names/ip-rib-route-table-name/routes/route
IPv6 RIB Info	sensor-path Cisco-IOS-XR-ip-rib-ipv6-oper:ipv6-rib/vrfs/vrf/afs/af/safs/saf/ip-rib-route-table-names/ip-rib-route-table-name/routes/route
BGP IPv4 Routes Info	sensor-path Cisco-IOS-XR-ip-rib-ipv4-oper:rib/vrfs/vrf/afs/af/safs/saf/ip-rib-route-table-names/ip-rib-route-table-name/protocol/bgp/as/information
BGP IPv6 Routes Info	sensor-path Cisco-IOS-XR-ip-rib-ipv6-oper:ipv6-rib/vrfs/vrf/afs/af/safs/saf/ip-rib-route-table-names/ip-rib-route-table-name/protocol/bgp/as/information
BGP ipv4 Neighbor Info	sensor-path Cisco-IOS-XR-ipv4-bgp-oper:bgp/instances/instance/instance-active/default-vrf/neighbors/neighbor
MPLS-TE Tunnels Summary Info	sensor-path Cisco-IOS-XR-mpls-te-oper:mpls-te/tunnels/summary
RSVP Interface Info	sensor-path Cisco-IOS-XR-ip-rsvp-oper:rsvp/interface-briefs/interface-brief

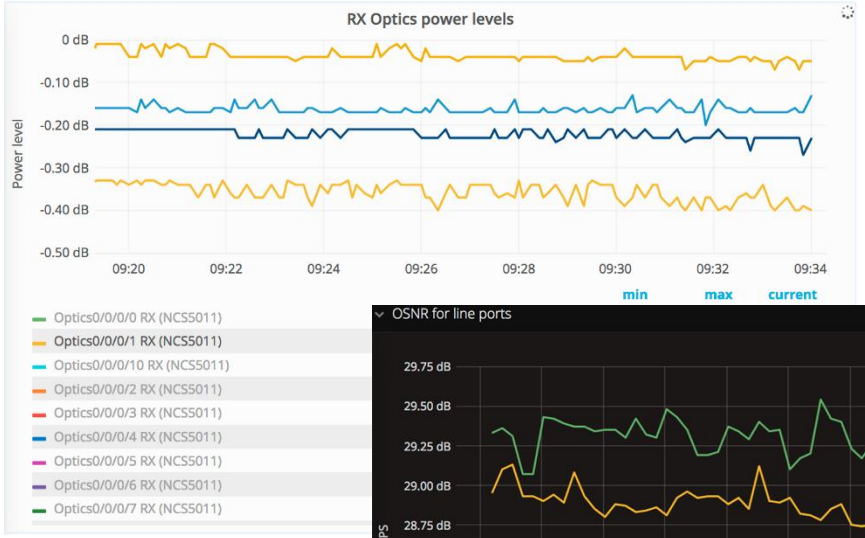
Popular Use Cases

SNMP OIDs ➡ MDT

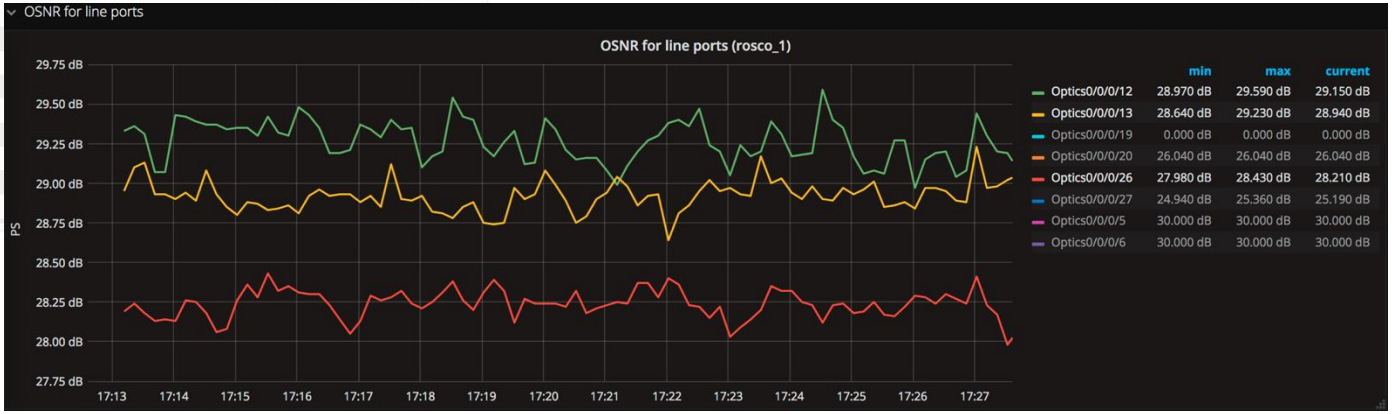
(BNG, Optical, BGP Peering, MPLS, IF stats, Drops, CPU/MEM , etc., etc.)

- ☑ Auto ticketing (port down, errors/threshold, if flapping, BGP peer lost, ISIS peer lost, etc)
- ☑ BNG Automatic IP address allocation for clients
- ☑ Predictive Load Capacity Planning
- ☑ AI-powered Anomaly Detection

Telemetry For Optical Transceivers and Platforms



sensor-path Cisco-IOS-XR-controller-optics-oper:optics-oper/optics-ports/optics-port/optics-info

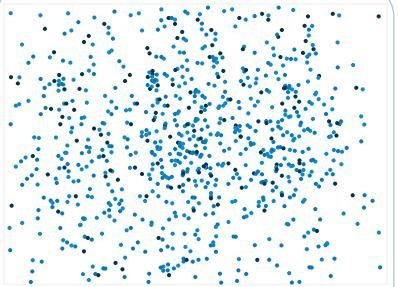


https://xrdocs.io/telemetry/tutorials/2017-10-25-ncs1002-telemetry-deep-dive

Deploy XR 7.3.1

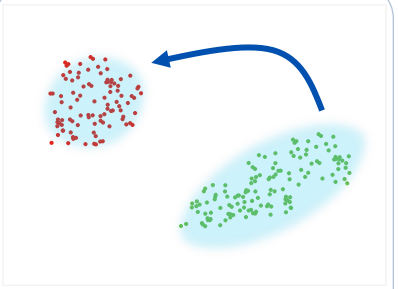
AI Driven Telemetry (ADT)

Collect



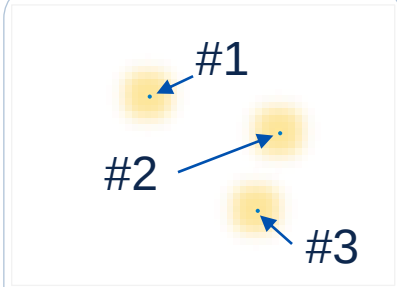
Holistic view:
Collect all counters all the time.
Currently: MDT data, Netflow

Detect



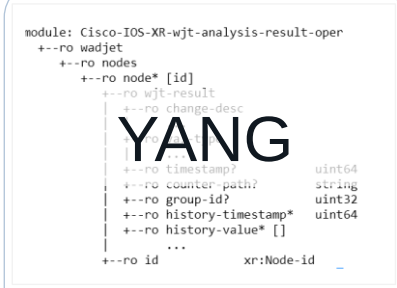
Macroscopic view:
Catch interesting state changes.
Dim.-Redux, Cluster.
Online, unsupervised.

Select



Microscopic view:
Choose counters which best describe the state change.
Online, unsupervised.

Export



Present results using existing YANG tool-chain:
Counter values, Sensor-paths

Conclusion

Check Your IOS XR Version

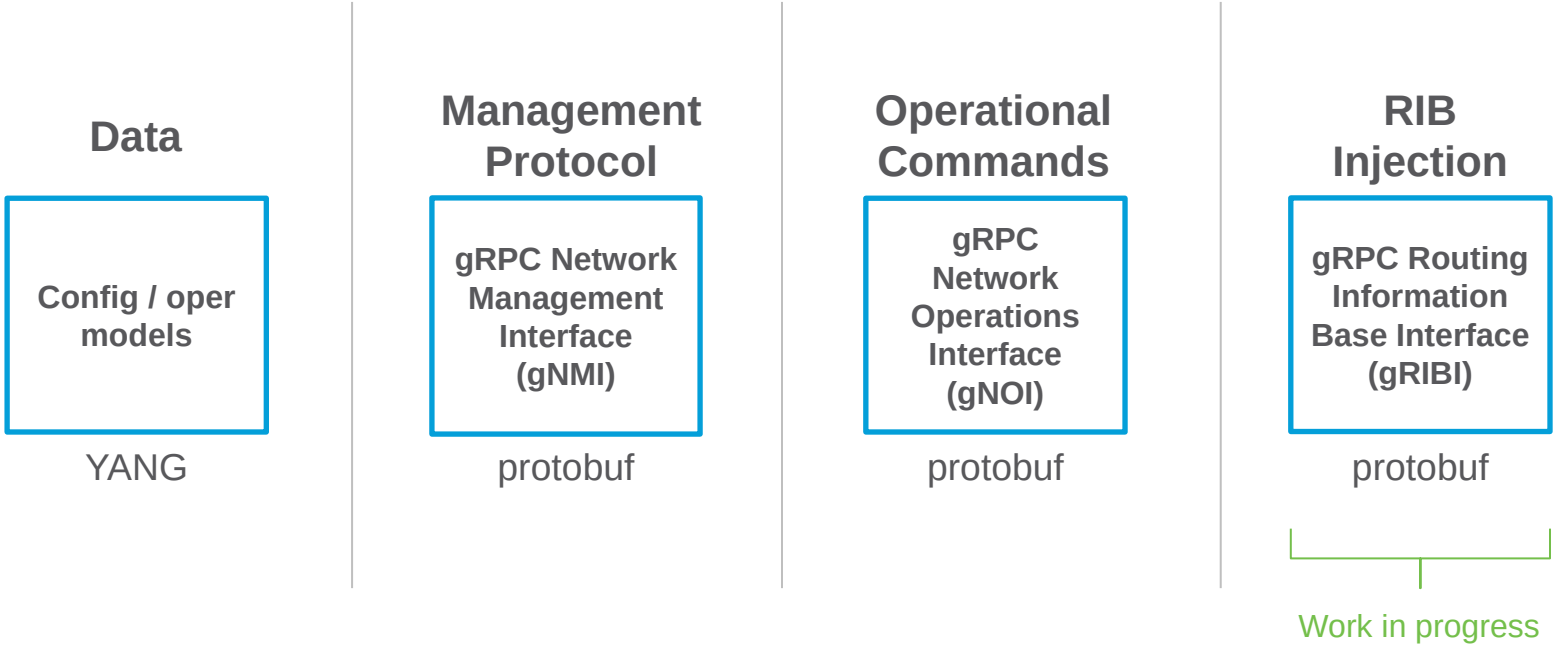
	Classic XR ASR9k	Evolved XR ASR9k	NCS 5500/5700/ 540	8000	NCS6k
MDT support	6.1.1	6.1.1	6.1.1	7.0.12	6.1.3
Data models	YANG (native, OC)	YANG (native, OC)	YANG (native, OC)	YANG (native, OC)	YANG (native, OC)
Transport (Control protocols)	TCP, UDP (6.2.1)	gRPC (dial-in, dial-out), TCP, UDP (6.2.1)	gRPC (dial-in, dial-out), TCP, UDP (6.2.1)	TCP, UDP, gRPC	TCP, UDP (6.2.1) **gRPC (mgmt port only, dial-in, dial-out, 6.5.1)**
Encoding	GPB / GPB-KV / JSON (6.3.1)	GPB / GPB-KV / JSON (6.3.1)	GPB / GPB-KV / JSON (6.3.1)	GPB / GPB-KV / JSON	GPB / GPB-KV / JSON (6.3.1)
gNMI		**6.5.1** (rev. 0.4)	**6.5.1** (rev. 0.4)	**7.0.12** (rev. 0.6)	**6.5.1** (rev. 0.4)

gNMI/NETCONF support across Cisco Products

Protocol	IOS XR	IOS XE	NX OS
gNMI	6.5.1	16.12*	9.3(x)
NETCONF	4.1	16.6*	7.x

* Feature availability is platform dependent

Have a full picture in your head



Complete your Session Survey

- Please complete your session survey after each session. Your feedback is important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (open from Thursday) to receive your Cisco Live t-shirt.

- All surveys can be taken in the Cisco Events Mobile App or by logging in to the Session Catalog and clicking the "Attendee Dashboard" at https://www.ciscolive.com/emea/learn/sessions/session-catalog.html

Continue Your Education

Visit the Cisco Showcase for related demos.

Book your one-on-one Meet the Engineer meeting.

Attend any of the related sessions at the DevNet, Capture the Flag, and Walk-in Labs zones.

Visit the On-Demand Library for more sessions at ciscolive.com/on-demand.


cisco
The bridge to possible
Thank you
cisco Live!


CISCO Live!
ALL IN