ASR9K model gedreven telemetrie whitepaper

Inhoud

Inleiding

Publiek

Dit witboek is bedoeld om klanten te helpen een snel begrip te verkrijgen van de functie Model Driven Telemetry (MDT) in het algemeen en hoe het is geïmplementeerd in Aggregation Services Router 9000 (ASR9K) met inbegrip van bepaalde ontwerprichtlijnen en configuratiedetails. Het omvat ook enige implementatieoverweging, die tijdens de implementatie van deze functie met behulp van ASR9K nuttig zal zijn. Over het geheel genomen kan dit artikel een snelle naslaggids zijn voor iedereen die aan deze functie werkt.

Hoewel telemetrie wordt geïntroduceerd als een generieke functie, is de focus gericht op ASR9K-implementatie, dat wil zeggen dat niet alle functies die worden ondersteund door andere Cisco-platforms worden ondersteund door ASR9K-platform en sommige functies kunnen specifiek zijn voor ASR9K.

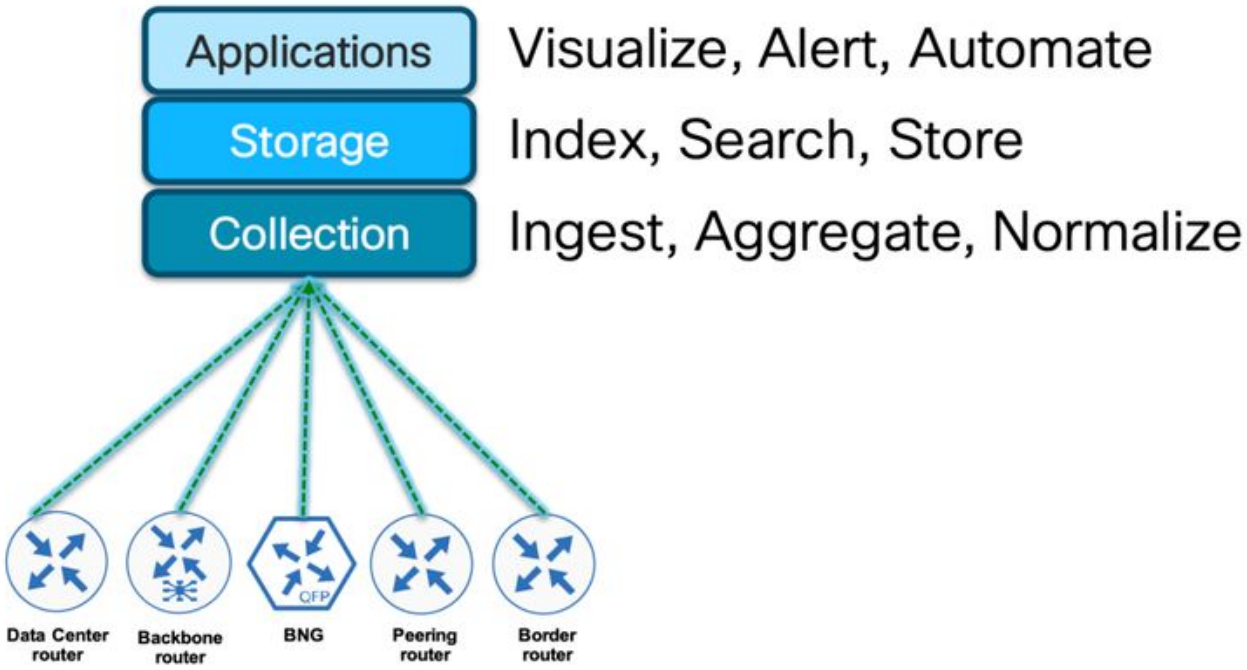
Een korte inleiding tot telemetrie

Om te beginnen, in eenvoudige termen, is Telemetrie het verzamelen van **nuttige operationele gegevens**. Volgens Wikipedia is telemetrie een geautomatiseerd communicatieproces waarbij metingen en andere gegevens worden verzameld op afgelegen of ontoegankelijke punten en verzonden naar ontvangstapparatuur voor monitoring. Telemetrie woord zelf is afgeleid van Griekse wortels: tele = ver, en metron = maatregel.

Voor het netwerkbeheer hebben netwerkbeheerders een lange geschiedenis van het vertrouwen op Simple Network Management Protocol (SNMP). Terwijl SNMP wijd voor netwerk controle wordt goedgekeurd, werd het nooit gebruikt voor configuratie alhoewel het vermogen van configuratie met SNMP altijd daar was. Operatoren hebben automatiseringsscripts geschreven om de dagelijkse configuratietaken te verwerken, maar scripts zijn uitdagend voor dergelijke taken en moeilijk te beheren.

De exploitanten zijn dan ook overgestapt op datamodelgestuurd management. De netwerkconfiguratie is gebaseerd op YANG datamodellen die door protocollen zoals netconf worden geduwd. Nu impliceert het duwen van de configuratie niet dat de gevormde dienst loopt, moet er een mechanisme zijn dat de diensten operationele gegevens kan controleren tezelfdertijd als de configuratie. Dit is waar oper data modellen; die Telemetry gebruikt om informatie uit apparaat te duwen; helpt. Daarom is de configuratie YANG data model gedreven dus moet de verificatie van de dienst ook zijn; zoals het geval met Telemetrie, om hetzelfde object semantisch te hebben. Vandaar wordt de term **Model Driven Telemetry** genoemd of streaming Telemetry.

Model Driven Telemetry (MDT) werd geïntroduceerd in **cXR** (32 bit IOS XR) sinds release **6.1.1** en maakt het mogelijk kritieke gegevens te verzamelen en te meten in bijna realtime, waardoor een snel antwoord kan worden gegeven op de meeste operationele problemen van het moderne netwerk.



Telemetriearchitectuur op hoog niveau

MDT maakt gebruik van gestructureerde gegevensmodellen die worden ondersteund door het netwerkapparaat en biedt essentiële gegevens die in deze gegevensmodellen zijn gedefinieerd. **Telemetrie** helpt klanten om **hun netwerk van meerdere leveranciers** te **beheren** met behulp van één gemeenschappelijk netwerkbeheersysteem, proces en toepassingen, aangezien de gegevens die van het netwerk worden verzameld op standaarden gebaseerd zijn en uniform zijn over de implementatie van verschillende leveranciers.

In plaats van te wachten op het ophalen van gegevens (pull) van een gecentraliseerd beheerstation (meestal SNMP NMS); met MDT sturen netwerkapparaten proactief (**push**) prestatiegegevens met betrekking tot de vitale netwerkfuncties, zoals pakketdoorsturen informatie, foutstatistieken, systeemstatus, CPU- en geheugenbronnen, enzovoort.

Waarom telemetrie

Het verzamelen van gegevens voor analytische en probleemoplossingsdoeleinden is altijd een belangrijk aspect geweest in de controle van de gezondheid van een netwerk. Er zijn verschillende beschikbare mechanismen zoals SNMP, CLI en Syslog om gegevens van een netwerk te verzamelen. Terwijl deze methodes het netwerk voor zeer lange tijd dienden maar niet voor modern netwerk past waar de vraag naar automatisering, de diensten op schaal fundamenteel is. De gezondheidsinformatie van het netwerk, de verkeersstatistieken en de kritieke infrastructuurinformatie worden verzonden naar een ver station in NMS, waar zij worden gebruikt om operationele prestaties te verbeteren en het oplossen van problemen te verminderen tijd. Een pull-model zoals een SNMP waarbij een client alle netwerkknooppunten opzoekt, is niet efficiënt. De verwerkingsbelasting op de netwerkknooppunten neemt toe wanneer er meer geen clients zijn om te pollen. In tegendeel, een push-model heeft een mogelijkheid om continu gegevens uit het netwerk te streamen en de klant te informeren. Telemetrie maakt het push-model mogelijk, dat bijna realtime toegang tot monitoringgegevens biedt.

Streamingtelemetrie biedt een mechanisme om gegevens te selecteren die van belang zijn voor routers en om deze in een standaardformaat te verzenden naar een extern beheerstation voor bewaking. Dit mechanisme maakt een fijnafstemming van het netwerk mogelijk op basis van real-time data, wat cruciaal is voor een naadloze werking. De fijnere granulariteit en hogere frequentie

van gegevens beschikbaar via telemetrie maakt betere prestatiebewaking mogelijk en leidt dus tot betere probleemoplossing.

Het helpt in een meer service-efficiënte bandbreedte gebruik in het netwerk, link gebruik, risico beoordeling en schaalbaarheid. Met Streamingtelemetrie beschikken netwerkexploitanten over meer bijna realtime gegevens, wat de besluitvorming helpt te verbeteren.

De noodzaak om weg te gaan van SNMP

SNMP bestaat al drie decennia en de manier waarop het werkt is niet gewijzigd om te voldoen aan de controlebehoeften van de moderne netwerken. Het echte probleem is de snelheid van de uitvoering van SNMP.

De drie primaire uitdagingen die SNMP stelt, maken deel uit van zijn fundamentele operationele gedrag en SNMP biedt dus weinig of geen ruimte voor verbetering en Telemetrie richt zich op alle drie inherent.

- **Snelheid van uitvoering en noodzaak van realtime bewaking**

SNMP maakt gebruik van PULL Model - GetBulk / GetNext operaties die werken op een lineaire manier door de tabellen van de ene kolom naar een andere tot. Daarnaast zijn er meerdere aanvragen nodig voor grote tabellen die niet in één pakket kunnen passen. Dit is het grootste knelpunt waardoor SNMP vertraagt en gegevens die worden verzonden vaak verouderd zijn door een bepaalde tijdfactor in minuten. Deze vertraging is simpelweg niet acceptabel voor moderne eisen met betrekking tot netwerkbewaking.

MDT (Model Driven Telemetry) maakt gebruik van het PUSH-model en is inherent vrij van bovenvermelde beperking omdat het weet welke gegevens naar wie moeten worden verzonden en met welk interval. Het heeft slechts één lookup nodig om gegevens te verzamelen en gebruikt pre-gebouwde interne sjablonen voor ultrasnelle snelheid van interne bewerkingen, waardoor het mogelijk maken van veel meer gegevens in aanzienlijk minder tijd.

- **Aanvullende overheadkosten en een gebrek aan optimalisatieopties**

De gegevens die door SNMP worden getrokken, worden feitelijk opgeslagen als interne gegevensstructuren en moeten intern door de knooppunt worden geconverteerd. Dit is extra werk achter de schermen waar het netwerkknooppunt interne datastructuren in SNMP-formaat in kaart brengt. Er worden interne optimalisaties uitgevoerd, maar ze zijn nog niet genoeg.

Aan de andere kant haalt Telemetrie direct de interne datastructuren eruit en voert een minimale verwerking uit voordat deze gegevens worden verzonden, waardoor de meest recente gegevens zo weinig mogelijk tijd en moeite krijgen.

- **Lineair karakter van werkbelasting**

Elk extra stembureau leidt tot extra werkbelasting op het knooppunt, zelfs als we op hetzelfde exacte tijdstip dezelfde exacte gegevens aan het pollen zijn. Parallelle toegang van dezelfde MIB van meerdere stembureaus kan leiden tot een langzamere respons en een hoger CPU-gebruik. Dit is vooral duidelijk in het geval van grote tabellen, waar meerdere stations toegang hebben tot verschillende delen van dezelfde MIB-tabel.

Telemetrie aan de andere kant moet gegevens eenmaal ophalen en de pakketten repliceren als dezelfde gegevens worden vereist door meerdere bestemmingen. Push model verslaat SNMP

Pull voor Speed & Schale.

Met MDT, de benadering van gegevensverzameling radicaal verandert en zijn fundamentele principes worden vermeld in de lijst hieronder en vergeleken met SNMP technologie zeer belangrijke punten.

Simple Network Management Protocol (SNMP)	Model Driven Telemetry (MDT)
Niet-realtime informatie	Realtime informatie
Slecht schaalbaar	Zeer schaalbaar
trekmodel	push-model
Niet geautomatiseerd	Klaar voor automatisering/Data-Model Driven

Voordelen van streaming telemetrie

Gestreamde real-time telemetriegegevens zijn nuttig in:

Optimalisatie van capaciteitsplanning/verkeer: wanneer bandbreedtegebruik en pakketdalingen in een netwerk vaak worden gecontroleerd, is het eenvoudiger om links toe te voegen of te verwijderen, verkeer opnieuw te sturen, toezicht aan te passen enzovoort. Dankzij technologieën zoals snelle omleiding kan het netwerk switches naar een nieuw pad en sneller omleiden dan het SNMP-pollinterfacemechanisme. Streaming telemetriegegevens helpen bij het leveren van snelle responstijd voor sneller verkeer.

Betere zichtbaarheid: helpt bij het snel detecteren en voorkomen van storingssituaties die resulteren na een problematische conditie in het netwerk.

Technische specificaties voor model aangedreven telemetrie

Het volgende gedeelte gaat over technische functies en hoofdcomponenten van IOS XR Model Driven Telemetry aka MDT.

Telemetriefuncties

Het telemetriekader is onderverdeeld in drie afzonderlijke en onderling verbonden functionele blokken.

Het eerste blok gaat over **dataweergave**, dat is hoe de informatie die verwijst naar analyse of metingen aan boord is georganiseerd.

Het tweede blok gaat over het **coderen**. Bij elk voorbeeldinterval vertaalt Telemetrie de bovenstaande meetgegevens in een formaat dat over de draad kan worden geserialiseerd. Natuurlijk moet de controller aan de andere kant in staat zijn om de gegevens te decoderen om een identieke kopie van de oorspronkelijke gegevens die door het apparaat worden verzonden te hebben.

Het laatste blok gaat over **vervoer**. Dit is de protocolstapel die wordt gebruikt om gegevens tussen apparaten over te brengen.

De volgende tabel geeft een overzicht van de hoofdstructuur van de bouwstenen van Model Driven Telemetry:

Functie	Componenten
Gegevensvertegenwoordiging	YANG Data-modellen
Codering	GPB / GPB zelfbeschrijvend
Vervoer	TCP/gRPC

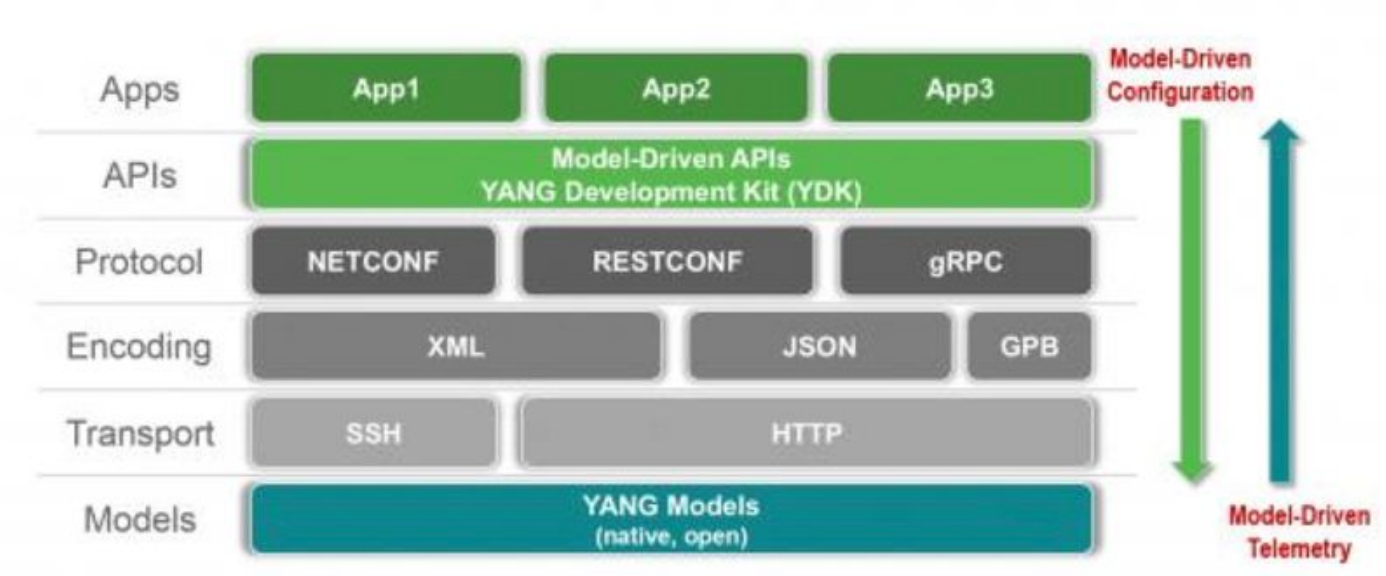
Tabel 3 Bouwstenen voor telemetrie

Telemetriecomponenten

Alvorens te begrijpen hoe Telemetrie en de onderliggende configuratiestukken werken, is het belangrijk om de verschillende componenten van Telemetrie te begrijpen om een optimale opstelling te evalueren. Telemetrie vertrouwt op de IOS XR-programmeerbaarheidsstack waar een nieuw infrastructuurframework de essentiële mogelijkheden biedt voor netwerkautomatisering.

YANG werd onlangs een norm voor gegevensmodellering, en dit wordt gebruikt door Cisco programmeerbaarheidstack om gestructureerde dataset te vormen die kan worden gecodeerd en zo snel mogelijk over het netwerk worden gedragen. De flexibiliteit van YANG biedt het grote voordeel om ook als configuratietool voor automatiseringsprocessen te worden gebruikt. Deze gegevensmodellen worden gekoppeld aan specifieke coderingsformaten en transportprotocollen om MDT tot een complete oplossing voor Network Analytics te maken.

Voor de installatie van Model Driven Telemetry wordt het YANG-datamodel een cruciaal onderdeel om de nodige gegevensstreaming voor verzameling en analyse mogelijk te maken.



IOS XR-programmeerbaarheidsstack

YANG

Yang wordt gedefinieerd als "een taal voor het modelleren van gegevens, die wordt gebruikt om configuratiegegevens, staatsgegevens en meldingen voor netwerkbeheerprotocollen te modelleren." Door zijn losgekoppelde karakter van een typische programmeertaalarchitectuur kan YANG worden geïmplementeerd om te communiceren met een grote verscheidenheid aan tools.

YANG modelleringsdatastructuur is gebouwd rond het concept van modules en submodules die een hiërarchie van gegevens in een boom als mode definieert, die kan worden gebruikt voor

verschillende bewerkingen, waaronder configuratieacties en notificatieverwerking.

Er zijn meerdere bronnen van YANG-modellen beschikbaar voor gebruik, waarvan er drie als primair worden beschouwd:

- Native modellen/Cisco specifiek
- OpenConfig
- IETF

Cisco-specifieke modellen: Deze worden ook wel **native model** genoemd en worden gepubliceerd door verschillende apparaatleveranciers, inclusief Cisco. Bijvoorbeeld Cisco IOS-XR-ptp-oper.yang

OpenConfig-modellen: OpenConfig is een informele werkgroep van netwerkbeheerders. OpenConfig definieert algemene YANG-modellen die door alle leveranciers moeten worden ondersteund om bedrijfskritieke functies te configureren. bijvoorbeeld openconfig-interfaces.yang

IETF-modellen: IETF definieert ook een paar algemene YANG-modules die basisconfiguratie voor interfaces beschrijven, QOS, en andere algemene datatypen definiëren (zoals IPv4, IPv6, enzovoort). bijvoorbeeld ietf-syslog-types.yang

Cisco ondersteunt geen beschikbare OpenCong-modellen. Verkopers komen samen aan een gestandaardiseerde manier om gegevens te modelleren ter ondersteuning van een omgeving met meerdere leveranciers.

Er zijn **drie soorten** Yang-modellen:

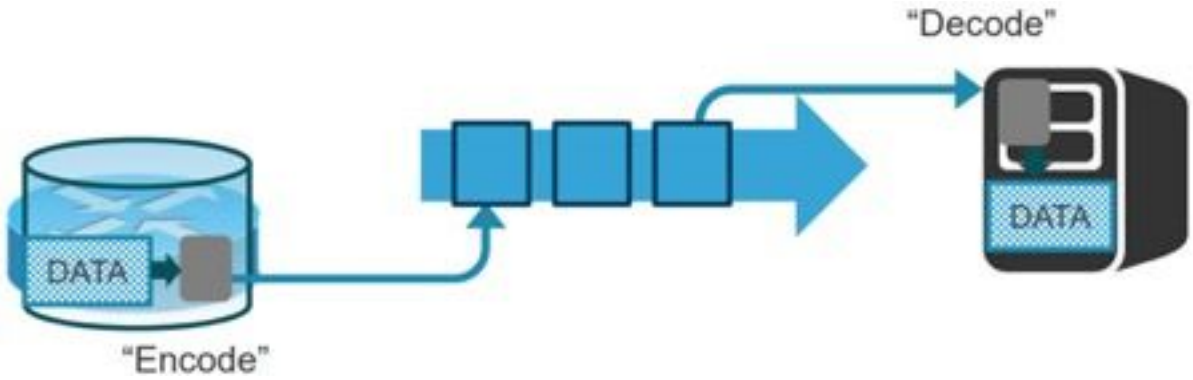
1. Operationeel
2. Configuratie
3. Actie

Telemetrie geeft alleen om **operationele Yang modellen** die kunnen worden geïdentificeerd als ***-oper-*.yang.**

YANG wordt gedefinieerd in RFC 7950: https://tools.ietf.org/html/rfc7950.

Codering

Het coderen (of "rangschikking") vertaalt gegevens (voorwerpen, staat) in een formaat dat over het netwerk kan worden overgebracht. Wanneer de ontvanger de gegevens decodeert ("de-serializeert"), heeft hij een semantisch identieke kopie van de oorspronkelijke gegevens.

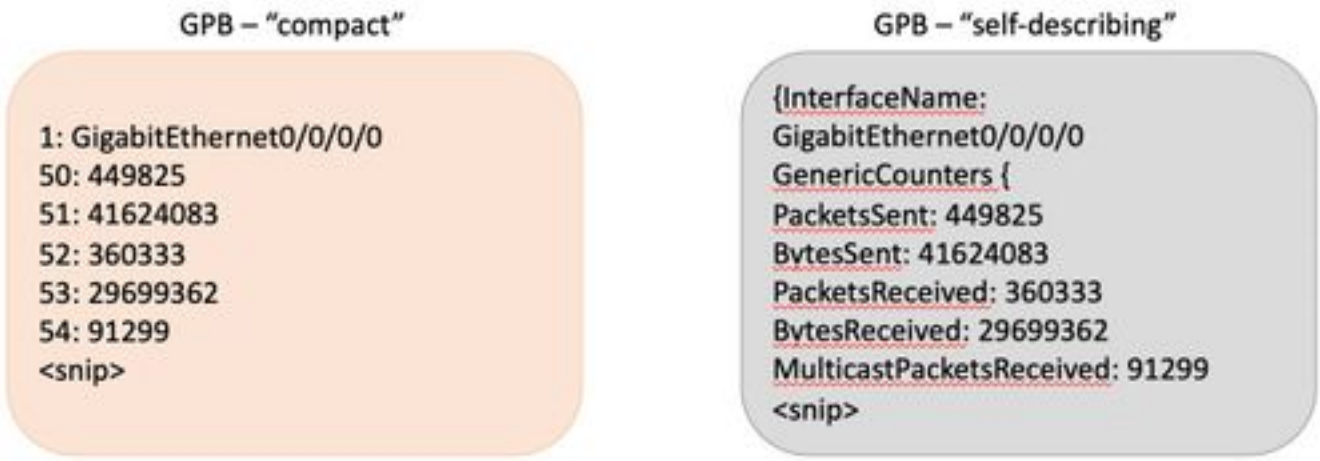


In de vroege ontwikkelingsfasen van telemetrie werd XML aanvankelijk beschouwd als een eerste keuze coderingsformaat vanwege de op tags gebaseerde structuur. Het probleem bij XML was echter de niet-compacte coderingsstructuur. GPB (Google Protocol Buffers) is uiteindelijk goedgekeurd door Cisco omdat dit de efficiëntie en snelheid van coderingsbewerkingen verbetert.

Er zijn twee smaken van GPB als coderingsopties voor Telemetry streaming:

1. Compacte GPB
2. zelfbeschrijvende GPB

Het belangrijkste verschil tussen de twee GPB-telemetrieformaten is hoe zij de sleutels binnen een telemetriestroom van gegevens vertegenwoordigen en coderen.



JSON is een ander menselijk vriendelijk coderingsschema beschikbaar dat zeer gemakkelijk te begrijpen is en bijna elke toepassing zal kunnen decoderen.

Vanuit het implementatieperspectief zijn er weinig voor- en nadelen van een coderingsschema. De vergelijking over divers coderingsschema wordt gegeven in de sectie Telemetry Design Guidelines.

Vervoer

Telemetrie biedt drie mogelijke keuzes voor transportprotocollen:

- TCP
- gRPC
- UDP

Telemetrie definieert ook twee verschillende initiatiemodi om een sessie te starten tussen het knooppunt en de collector:

- Uitbellen
- Inbellen

Het verschil tussen de twee vervoerswijzen bestaat alleen uit de wijze waarop de vervoerssessie tot stand wordt gebracht.

Tijdens uitbel sessies, het apparaat initieert de verbinding door een syn-pakket naar de vooraf geconfigureerde serverpoort te sturen. Nadat de verbinding tot stand is gebracht, worden de gegevensstromen onmiddellijk van het apparaat weggeduwd.

Voor inbelsessies luistert de router passief naar een TCP-poort die wacht op een serververbinding.

Echter, zodra de sessie is ingesteld, wordt de router niet door de server zelf ondervraagd omdat het apparaat nog steeds verantwoordelijk is voor data-pushing operaties. In de MDT bestaat het concept van data polling niet eens.

TCP is standaard de vooraf gedefinieerde transportmethode voor telemetrie omdat het betrouwbaar is en zeer eenvoudig te configureren als een optie.

gRPC is een modern opensourcekader dat ontworpen is om in elke omgeving te werken. Het is gebaseerd op HTTP/2 en biedt een uitgebreide en uitgebreide set functies.

Op ervaring gebaseerde telemetrie versus op gebeurtenissen gebaseerde telemetrie

De gegevens van de geabonneerde gegevensset worden naar de bestemming gestreamd met een geconfigureerd periodiek interval of alleen wanneer een gebeurtenis plaatsvindt. Dit gedrag is gebaseerd op de vraag of MDT is geconfigureerd voor cadence-gebaseerde telemetrie of event-based telemetrie.

De configuratie voor event-based telemetrie is vergelijkbaar met cadence-gebaseerde telemetrie, met alleen het sample interval als differentiator. Als u de waarde van het voorbeeldinterval op nul instelt, wordt het abonnement voor op gebeurtenissen gebaseerde telemetrie ingesteld, terwijl bij het configureren van het interval voor een niet-nulwaarde het abonnement voor op cadans gebaseerde telemetrie wordt ingesteld.

Het wordt aanbevolen om Event Driven Telemetry te gebruiken voor aan verandering gerelateerde gebeurtenissen.

Richtlijnen voor telemetrieontwerp

Zoals uitgelegd, zijn er vele componenten in de telemetriestack, hier zijn een paar richtlijnen te overwegen terwijl het implementeren van telemetrie op XR-apparaten.

Hoe u een coderingsschema selecteert

Zoals vermeld, vertaalt codering of serialisatie gegevens (objecten, status) naar een formaat dat over het netwerk kan worden verzonden. Wanneer de ontvanger de gegevens decodeert of de-serializeert, heeft het een semantisch identieke kopie van de oorspronkelijke gegevens.

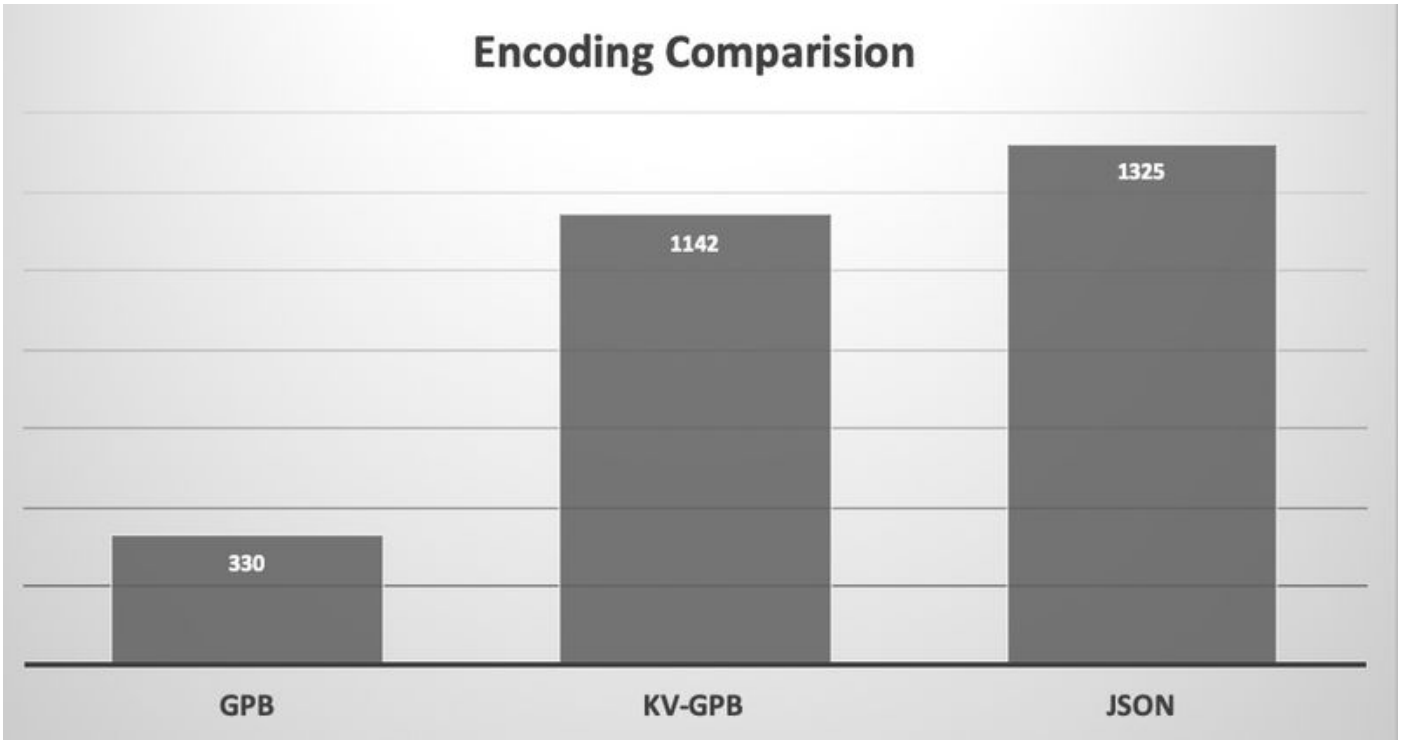
Verschillende coderingsopties variëren in efficiëntie van de draad en gebruiksgemak.

Codering	Korte beschrijving	Efficiëntie voor bekabeling	Andere overweging
GPB (compact)	Alles Binair (Behalve Waarden die koorden zijn) 2X sneller, operationeel	Hoog	Protobestand per mo

	complexer (maar niet ten opzichte van SNMP)		
GPB - KV (sleutelwaardepaar)	String-toetsen en binaire waarden (behalve waarden die tekenreeksen zijn) 3x groter, Native modellen: nog steeds nodig heuristiek voor sleutelnamen	Gemiddeld tot laag	Eén .proto-bestand v decodering
JSON	Alles Koorden: Sleutels en Waarden	Laag	Vriendelijk. Menselijk leesbaar, Toepassingsvriendeli gemakkelijk te ontled

GPB-KV geeft een goed & balans middelpunt voor het coderen van schema.

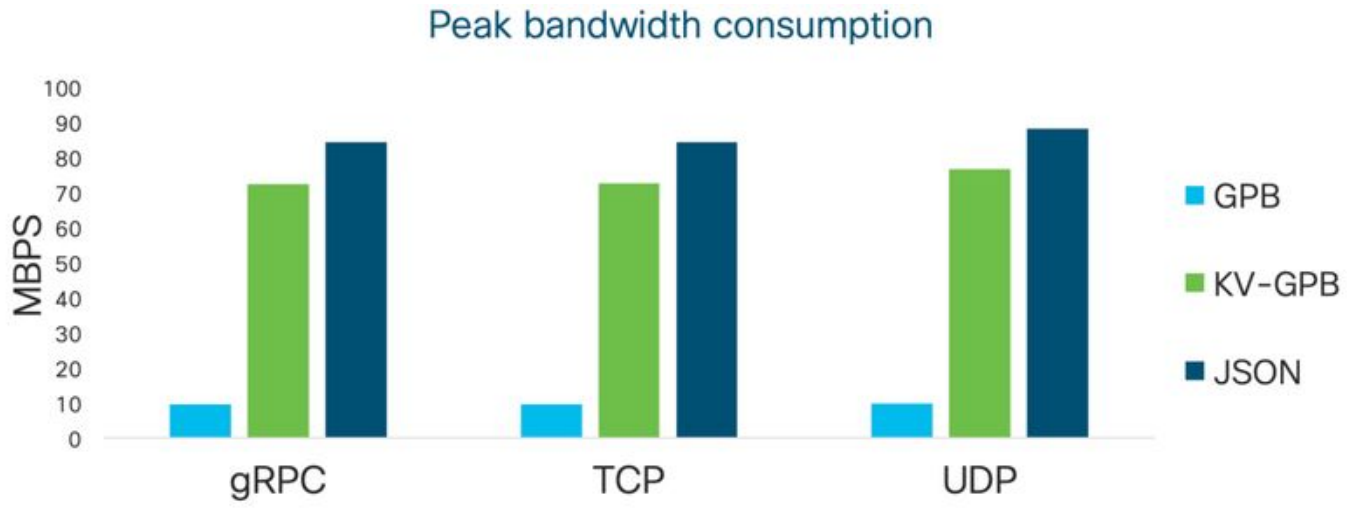
Met betrekking tot berichtlengte voor gekozen coderingsschema, hieronder is de vergelijking op de draad.



Coderingsvergelijking - Berichtlengte in bytes

Ontwerpoverweging voor transportnetwerk

Een andere coderingsoptie stelt een ander bandbreedtevereiste. Bij het overwegen van telemetrie moet de netwerkoperator zorgen voor voldoende bandbreedte provisioning volgens het gekozen coderingsschema. Enkel om een eerlijk idee te hebben, onder bandbreedteverbruik per het coderen van schemavergelijking.



Vergelijking van netwerkbandbreedte

Cisco raadt het gebruik van KV-GPB aan. Het fungeert als een goed middenpunt tussen efficiëntie en gemak.

Configuratieopties voor telemetrie evalueren

Bij het configureren van telemetrie op basis van het model moet de operator inzicht hebben in alle verschillende componenten die bij telemetrie betrokken zijn. Gebaseerd op de opties die beschikbaar zijn voor transport, codering en de richting van streamen zoals hierboven beschreven, kan men verder de combinatie kiezen die beter past bij een omgeving.

De vier hoofdcomponenten zijn:

1. Vervoer
2. Codering
3. Sessierichting
4. YANG Data-modellen

Vervoer: Zoals vermeld, kan knooppunt telemetriegegevens leveren via TCP, UDP of gRPC via HTTP/2.

Hoewel TCP de voorkeurskeuze voor eenvoud is, biedt gRPC optionele TLS-mogelijkheid die vanuit beveiligingsoogpunt als een extra voordeel kan worden beschouwd.

Codering: De router kan telemetriegegevens in twee verschillende smaken van de buffers van het Protocol van Google leveren: Compacte en zelf-beschrijft GPB.

Compacte GPB is de meest efficiënte codering, maar vereist een unieke .proto voor elk YANG-model dat wordt gestreamd. Zelfbeschrijvend GPB is minder efficiënt maar het gebruikt één enkel .proto bestand om alle YANG modellen te decoderen omdat de sleutels als koorden in .proto worden doorgegeven.

Sessierichting: Er zijn twee opties voor sessie initiëren in telemetrie-implementatie. De router kan "uitbellen" naar de collector of de collector kan "inbellen" naar de router.

YANG is de industrie toegelaten norm voor gegevensmodellering en de programmeerbaarheidsstapel van Cisco gebruikt het ook om gestructureerde dataset te vormen

die kan worden gecodeerd en zo snel mogelijk over het netwerk worden gedragen.

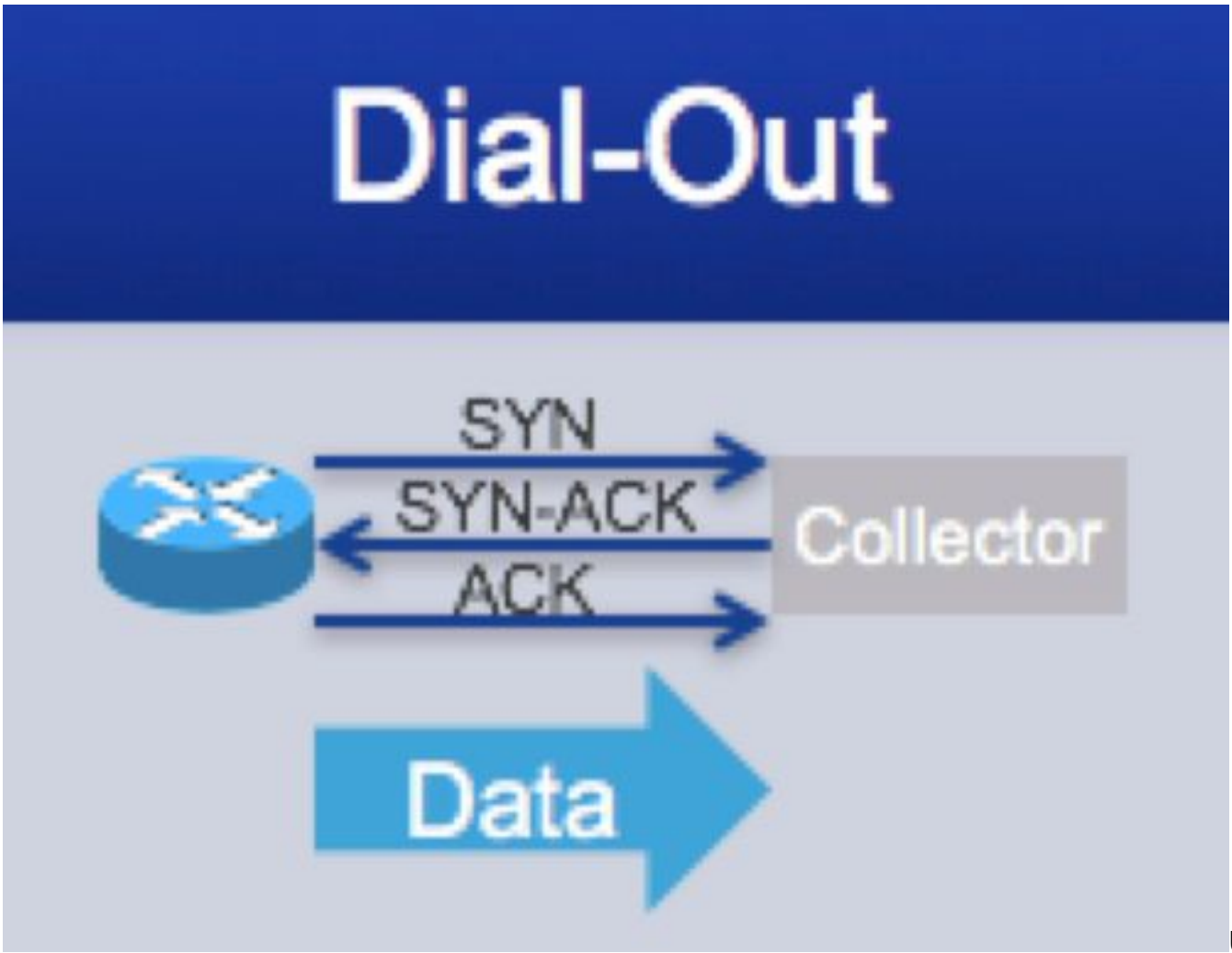
Deze gegevensmodellen in combinatie met specifieke coderingsformaten en transportprotocollen zoals hierboven besproken, maken Model Driven Telemetry (MDT) een complete oplossing voor Analytics.

Configuratievoorbeelden voor telemetrie

IOS-XR

Scheiding van uitbel-configuratie

In de Uitbel-modus is de router verantwoordelijk voor het starten van een TCP-sessie naar de collector en het verzenden van gegevens die door de sensorgroep in het abonnement worden gespecificeerd.



Uitbel voor telemetrieDe Telemetry Configuration is vanuit het configuratiestandpunt een proces in drie stappen. Ten eerste identificeren we de informatie die we willen streamen en vastleggen onder de configuratie van de Sensor Group. Ten tweede, identificeren we de bestemming waarnaar de informatie moet worden gestreamd en nemen het op onder Bestemmingsgroep configuratie. Ten derde gebruiken we de informatie die in de vorige twee stappen is geïdentificeerd om het feitelijke abonnement te configureren.

1. Sensorgroepen definiëren

2. Doelgroepen definiëren
3. Abonnement definiëren

Sensorgroepen definiëren

De groepsconfiguratie Sensor identificeert de informatie die moet worden gestreamd. Na configuratiesjabloon verstrekt de configuratie die wordt vereist om sensorgroepen te vormen.

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)#telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#sensor-group <Sensor-Group-Name>
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# sensor-path <Sensor-Path>
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# commit
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# end
RP/0/RP0/CPU0:XR#
```

Het volgende voorbeeld toont feitelijk voorbeeld van de router CLI waar een echt voorbeeld van de configuratie van de Sensorgroep:

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)#telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#sensor-group SensorGroup101
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# sensor-path Cisco-IOS-XR-infra-statsd-
oper:infra-statistics/interfaces/interface/latest/generic-counters
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# commit
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# end
RP/0/RP0/CPU0:XR#
```

We kunnen meerdere Sensorpaden hebben als onderdeel van dezelfde SensorGroup-definitie:

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)#telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#sensor-group SensorGroup101
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# sensor-path sensor-path Cisco-IOS-XR-infra-
statsd-oper:infra-statistics/interfaces/interface/data-rate
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# sensor-path Cisco-IOS-XR-infra-statsd-
oper:infra-statistics/interfaces/interface/latest/generic-counters
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# commit
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# end
RP/0/RP0/CPU0:XR#
```

Doelgroepen definiëren

De groepsconfiguratie Bestemming identificeert de bestemming waarnaar de informatie moet worden gestreamd.

Het heeft drie belangrijke parameters

1. Sessierichting
2. Te gebruiken codering

3. Te gebruiken transportprotocol

Hieronder zie je een voorbeeld:

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)# telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)# destination-group DestGroup101
RP/0/RP0/CPU0:XR(config-model-driven-dest)#  address family ipv4 10.1.1.1 port 5432
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#   encoding self-describing-gpb
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#   protocol tcp
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)# commit
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# end
RP/0/RP0/CPU0:XR#
```

Het abonnement definiëren

Het abonnement bindt de Sensor-groep en de bestemming-groep informatie samen als het definitieve stuk van de configuratie. Het voorbeeldinterval is gedefinieerd als een deel van het abonnement.

Hieronder zie je een voorbeeld:

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#subscription Subscription101
RP/0/RP0/CPU0:XR(config-model-driven-subs)#sensor-group-id SensorGroup101 sample-interval 30000
RP/0/RP0/CPU0:XR(config-model-driven-subs)#destination-id DestGroup101
RP/0/RP0/CPU0:XR(config-model-driven-subs)# commit
RP/0/RP0/CPU0:XR(config-model-driven-subs)# end
RP/0/RP0/CPU0:XR#
```

Voorbeeld van volledige configuratie

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#conf
RP/0/RP0/CPU0:XR(config)#
RP/0/RP0/CPU0:XR(config)#telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#sensor-group SensorGroup101
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# sensor-path Cisco-IOS-XR-infra-statsd-
oper:infra-statistics/interfaces/interface/latest/generic-counters
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)#destination-group DestGroup101
RP/0/RP0/CPU0:XR(config-model-driven-dest)#address family ipv4 10.1.1.2 port 5432
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#encoding self-describing-gpb
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#protocol tcp
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#subscription Subscription101
RP/0/RP0/CPU0:XR(config-model-driven-subs)#sensor-group-id SensorGroup101 sample-interval 30000
RP/0/RP0/CPU0:XR(config-model-driven-subs)#destination-id DestGroup101
RP/0/RP0/CPU0:XR(config-model-driven-subs)#commit
RP/0/RP0/CPU0:XR(config-model-driven-subs)#end
RP/0/RP0/CPU0:XR#
```

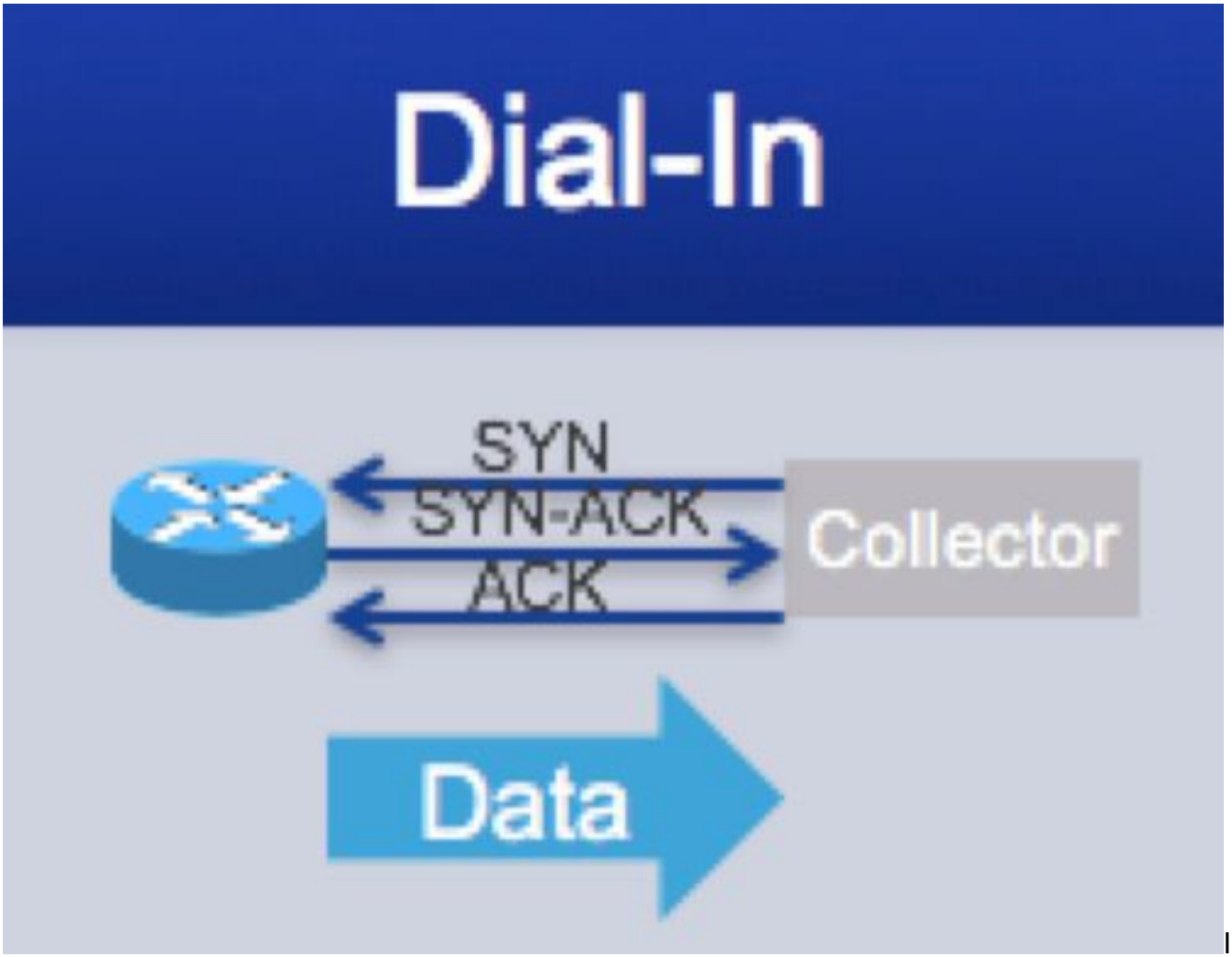
Voordelen voor uitbellen

- een grotere flexibiliteit voor vervoersopties.
- Geen noodzaak om poorten te openen voor inkomend beheerverkeer.
- Anycast en taakverdeling.

Scheiding van inbel-configuratie

In de inbelmodus wordt een MDT-collector / ontvanger / orchestrator ingeschakeld op de router en wordt dynamisch een abonnement genomen op een of meer sensorpaden of -abonnementen. De router fungeert als de server en de client als de ontvanger.

Slechts één sessie wordt gevormd en de router stroomt telemetriegegevens door dezelfde sessie. Dit dynamische abonnement eindigt wanneer de ontvanger het abonnement annuleert of wanneer de sessie wordt beëindigd.



Inb
ellen via telemetrie

Sinds de collector "inbellen" naar de router is het niet nodig om elke MDT-bestemming in de configuratie op te geven. Schakel gewoon de gRPC-service in op de router, sluit uw client aan en schakel dynamisch het gewenste telemetrieabonnement in.

Vanuit het configuratiestandpunt is Telemetry Configuration een proces in drie stappen, dat vergelijkbaar is met het proces dat hierboven wordt beschreven. Ten eerste moeten we gRPC mogelijk maken. Ten tweede, identificeren we de bestemming waarnaar de informatie moet worden gestreamd en nemen het op onder Sesnor Group configuratie. Ten derde gebruiken we de informatie die in de vorige twee stappen is geïdentificeerd om het feitelijke abonnement te

configureren.

1. gRPC inschakelen
2. Sensorgroepen definiëren
3. Abonnement definiëren

Klik om uit te vouwen

De inbelmodus wordt alleen ondersteund met gRPC

De inbelmodus wordt alleen ondersteund met gRPC

gRPC inschakelen

Ten eerste moeten we gRPC server op de router in staat stellen om inkomende verbindingen van de collector te accepteren.

Klik om uit te vouwen

- Het <poortnummer>-bereik loopt van 57344 tot 57999. Als een poortnummer niet beschikbaar is, wordt een fout weergegeven.

Het <poortnummer>-bereik loopt van 57344 tot 57999. Als een poortnummer niet beschikbaar is, wordt een fout weergegeven.

```
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)# grpc
RP/0/RP0/CPU0:XR(config-grpc)#port 57890
RP/0/RP0/CPU0:XR(config-grpc)#commit
RP/0/RP0/CPU0:XR(config-grpc)#end
RP/0/RP0/CPU0:XR#
```

Sensorgroepen definiëren

De groepsconfiguratie Sensor identificeert de informatie die moet worden gestreamd. Na configuratiesjabloon verstrekt de configuratie die wordt vereist om sensorgroepen te vormen.

Het volgende voorbeeld toont feitelijk voorbeeld van de router CLI waar een echt voorbeeld van de configuratie van de Sensorgroep

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)#telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#sensor-group SensorGroup101
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# sensor-path openconfig-
interfaces:interfaces/interface
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# commit
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# end
RP/0/RP0/CPU0:XR#
```

Het abonnement definiëren

Het abonnement bindt de Sensor-groep en gRPC samen als het laatste stuk van de configuratie. Het voorbeeldinterval is gedefinieerd als een deel van het abonnement.

Na configuratiesjabloon verstrekt de configuratie die wordt vereist om het abonnement te vormen.

Het volgende voorbeeld toont het daadwerkelijke voorbeeld van de router CLI waar wij een Abonnement creëren en de Sensor-Groep en de Bestemming-Groep samen en ook het bepalen van het steekproeftarief binden.

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#subscription Subscription101
RP/0/RP0/CPU0:XR(config-model-driven-subs)#sensor-group-id SensorGroup101 sample-interval 30000
RP/0/RP0/CPU0:XR(config-model-driven-subs)# commit
RP/0/RP0/CPU0:XR(config-model-driven-subs)# end
RP/0/RP0/CPU0:XR#
```

Sjabloon voor volledige configuratie en voorbeeld

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)# grpc
RP/0/RP0/CPU0:XR(config-grpc)#port 57890
RP/0/RP0/CPU0:XR(config-grpc)telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#subscription Subscription101
RP/0/RP0/CPU0:XR(config-model-driven-subs)#sensor-group-id SensorGroup101 sample-interval 30000
RP/0/RP0/CPU0:XR(config-model-driven-subs)# commit
RP/0/RP0/CPU0:XR(config-model-driven-subs)# end
RP/0/RP0/CPU0:XR#
```

Voordelen voor inbellen

- Eén kanaal voor configuratie en streaming
- Luisterpoort op de router/het apparaat
- Tijdelijke verbinding
- Momenteel is alleen gRPC/gNMI beschikbaar

Event Driven Telemetry

In Event gedreven Telemetry, worden de gegevens van de geabonneerde gegevensreeks uitgestroomd slechts wanneer een gebeurtenis voorkomt.

Configuratie van gebeurtenisgestuurde telemetrie

De configuratie voor event-based telemetrie is vergelijkbaar met cadence-gebaseerde telemetrie, waarbij het enige verschil in configuratie van Event Driven Telemetry is dat van het sample interval. Als u de waarde van het voorbeeldinterval op nul instelt, wordt het abonnement voor op gebeurtenissen gebaseerde telemetrie ingesteld.

Compleet configuratiesjabloon en voorbeeld voor UITBELLEN

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#conf
RP/0/RP0/CPU0:XR(config)#
```

```
RP/0/RP0/CPU0:XR(config)#telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#sensor-group <Sensor-Group-Name>
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# sensor-path <Sensor-Path>
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)#destination-group <Destination-Group-Name>
RP/0/RP0/CPU0:XR(config-model-driven-dest)#address family ipv4 <Destination-IP>  port
<Destination-Port>
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#encoding <Encoding-Type>
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#protocol <Transport-Protocol>
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#subscription <Subscription-Name>
RP/0/RP0/CPU0:XR(config-model-driven-subs)#sensor-group-id <Sensor-Group-Name> sample-interval
<0>
RP/0/RP0/CPU0:XR(config-model-driven-subs)#destination-id <Destination-Group-Name>
RP/0/RP0/CPU0:XR(config-model-driven-subs)#commit
RP/0/RP0/CPU0:XR(config-model-driven-subs)#end
RP/0/RP0/CPU0:XR#
```

Na voorbeeld toont het daadwerkelijke voorbeeld van de router CLI.

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#conf
RP/0/RP0/CPU0:XR(config)#
RP/0/RP0/CPU0:XR(config)#telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#sensor-group SensorGroup101
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)# sensor-path Cisco-IOS-XR-infra-statsd-
oper:infra-statistics/interfaces/interface/latest/generic-counters
RP/0/RP0/CPU0:XR(config-model-driven-snsr-grp)#destination-group DestGroup101
RP/0/RP0/CPU0:XR(config-model-driven-dest)#address family ipv4 10.1.1.2 port 5432
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#encoding self-describing-gpb
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#protocol tcp
RP/0/RP0/CPU0:XR(config-model-driven-dest-addr)#subscription Subscription101
RP/0/RP0/CPU0:XR(config-model-driven-subs)#sensor-group-id SensorGroup101 sample-interval 0
RP/0/RP0/CPU0:XR(config-model-driven-subs)#destination-id DestGroup101
RP/0/RP0/CPU0:XR(config-model-driven-subs)#commit
RP/0/RP0/CPU0:XR(config-model-driven-subs)#end
RP/0/RP0/CPU0:XR#
```

Compleet configuratiesjabloon en voorbeeld voor INBELLEN

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)# grpc
RP/0/RP0/CPU0:XR(config-grpc)#port <port-number>
RP/0/RP0/CPU0:XR(config-grpc)telemetry model-driven
RP/0/RP0/CPU0:XR(config-model-driven)#subscription <Subscription-Name>
RP/0/RP0/CPU0:XR(config-model-driven-subs)#sensor-group-id <Sensor-Group-Name> sample-interval
<0>
RP/0/RP0/CPU0:XR(config-model-driven-subs)#commit
RP/0/RP0/CPU0:XR(config-model-driven-subs)#end
RP/0/RP0/CPU0:XR#
```

Na voorbeeld toont het daadwerkelijke voorbeeld van de router CLI.

```
RP/0/RP0/CPU0:XR#
RP/0/RP0/CPU0:XR#config
RP/0/RP0/CPU0:XR(config)# grpc
RP/0/RP0/CPU0:XR(config-grpc)#port 57890
RP/0/RP0/CPU0:XR(config-grpc)telemetry model-driven
```

```
RP/0/RP0/CPU0:XR(config-model-driven)#subscription Subscription101
RP/0/RP0/CPU0:XR(config-model-driven-subs)#sensor-group-id SensorGroup101 sample-interval 0
RP/0/RP0/CPU0:XR(config-model-driven-subs)# commit
RP/0/RP0/CPU0:XR(config-model-driven-subs)# end
RP/0/RP0/CPU0:XR#
```

Het bevestigen van Telemetrie met TOON Opdrachten

Vanuit routerstandpunt kunnen we de parameters verifiëren die zijn geconfigureerd voor elke sensorgroep, doelgroep en abonnement

```
// ALL CONFIGURED SUBSCRIPTIONS
RP/0/RP0/CPU0:XR#show telemetry model-driven subscription

Subscription:  Subscription101          State: ACTIVE
-------------
  Sensor groups:
  Id                Interval(ms)        State
  SensorGroup101    30000               Resolved

  Destination Groups:
  Id                Encoding            Transport   State   Port    IP
  DestGroup101      self-describing-gpb tcp         Active  5432    172.16.128.3
```

```
// DETAILS ON A PARTICULAR SUBSCRIPTION
RP/0/RP0/CPU0:XR#show telemetry model-driven subscription Subscription101

Subscription:  Subscription101
-------------
  State:       ACTIVE
  Sensor groups:
  Id: SensorGroup101
    Sample Interval:      30000 ms
    Sensor Path:          Cisco-IOS-XR-infra-statsd-oper:infra-
statistics/interfaces/interface/latest/generic-counters
    Sensor Path State:    Resolved

  Destination Groups:
  Group Id: DestGroup101
    Destination IP:       172.16.128.3
    Destination Port:     5432
    Encoding:             self-describing-gpb
    Transport:            tcp
    State:                Active
    Total bytes sent:     4893
    Total packets sent:   1
    Last Sent time:       2019-11-01 10:04:11.2378949664 +0000

    Collection Groups:
  ------------------
    Id: 1
    Sample Interval:      30000 ms
    Encoding:             self-describing-gpb
    Num of collection:    5
    Collection time:      Min:     6 ms Max:    29 ms
    Total time:           Min:     6 ms Avg:    12 ms Max:    29 ms
    Total Deferred:       0
    Total Send Errors:    0
```

```
    Total Send Drops:     0
    Total Other Errors:   0
    Last Collection Start:2019-11-01 10:06:11.2499000664 +0000
    Last Collection End:  2019-11-01 10:06:11.2499006664 +0000
    Sensor Path:          Cisco-IOS-XR-infra-statsd-oper:infra-
statistics/interfaces/interface/latest/generic-counters

RP/0/RP0/CPU0:XR#

// ALL CONFIGURED DESTINATIONS
RP/0/RP0/CPU0:XR#show telemetry model-driven destination
  Group Id        IP              Port    Encoding            Transport   State
  ------------------------------------------------------------------------------
  DestGroup101    172.16.128.3    5432    self-describing-gpb tcp         Active
RP/0/RP0/CPU0:XR#

// PARTICULAR DESTINATION
RP/0/RP0/CPU0:XR#show telemetry model-driven destination DestGroup101
  Destination Group:  DestGroup101
  -----------------
    Destination IP:       172.16.128.3
    Destination Port:     5432
    State:                Active
    Encoding:             self-describing-gpb
    Transport:            tcp
    Total bytes sent:     83181
    Total packets sent:   17
    Last Sent time:       2019-11-01 10:12:11.2859133664 +0000

    Collection Groups:
    ------------------
      Id: 1
      Sample Interval:      30000 ms
    Encoding:             self-describing-gpb
      Num of collection:    17
      Collection time:      Min:     5 ms Max:    29 ms
      Total time:           Min:     6 ms Max:    29 ms Avg:    10 ms
      Total Deferred:       0
      Total Send Errors:    0
      Total Send Drops:     0
      Total Other Errors:   0
      Last Collection Start:2019-11-01 10:12:11.2859128664 +0000
      Last Collection End:  2019-11-01 10:12:11.2859134664 +0000
      Sensor Path:          Cisco-IOS-XR-infra-statsd-oper:infra-
statistics/interfaces/interface/latest/generic-counters

RP/0/RP0/CPU0:XR#

// ALL CONFIGURED SENSOR GROUPS
RP/0/RP0/CPU0:XR#show telemetry model-driven sensor-group
  Sensor Group Id:SensorGroup101
    Sensor Path:        Cisco-IOS-XR-infra-statsd-oper:infra-
statistics/interfaces/interface/latest/generic-counters
    Sensor Path State:  Resolved

// PARTICULAR SENSOR GROUPS
RP/0/RP0/CPU0:XR#show telemetry model-driven sensor-group SensorGroup101
  Sensor Group Id:SensorGroup101
    Sensor Path:        Cisco-IOS-XR-infra-statsd-oper:infra-
statistics/interfaces/interface/latest/generic-counters
    Sensor Path State:  Resolved

RP/0/RP0/CPU0:XR#
```

TelePresence Collectiestack

Naast de routerconfiguratie waren voor een telemetriegebaseerde oplossing verschillende componenten nodig, zoals de software voor collector, database en bewaking/analyse. Deze componenten kunnen afzonderlijk worden geconfigureerd of kunnen deel uitmaken van één compleet product.

- Er moet een decoder worden geïnstalleerd om de inkomende verpakkingen in te nemen en door te geven voor verdere opslag.
- Er is een tijdreeksdatabase (ook bekend als TSDB) nodig om gestreamde informatie op te slaan.
- Er is ook een grafische tool nodig om gegevens uit de interne database te visualiseren.

Het is buiten het bereik om de collectie stack in detail te beschrijven. Cisco Crossworks Health Insights maakt zero-touch telemetrie mogelijk waarbij apparaten automatisch worden voorzien van telemetrieconfiguratie en tabellen/schema's worden gemaakt in een Time Series Database (TSDB). Het stroomlijnt de operationele en netwerkbeheeroverheadkosten van het verzamelen en reinigen van gegevens, waardoor exploitanten zich op hun bedrijfsdoelstellingen kunnen concentreren. Het gebruik van een gemeenschappelijke collector om netwerkapparaatgegevens via SNMP, CLI en model-gedreven telemetrie te verzamelen maakt het mogelijk om gegevensduplicatie te vermijden en vermindert ook de belasting op apparaten en het netwerk.

Implementatieoverwegingen voor telemetrie in een netwerk

Hieronder vindt u verschillende overwegingen die moeten worden gemaakt bij het analyseren van de implementatie van telemetrie in een netwerk.

Schalen

Telemetrie kan een aanzienlijke hoeveelheid gegevens streamen en een zorgvuldige afweging van de schaalbaarheidsaspecten wordt aanbevolen.

Alleen de vereiste gegevens streamen

Elk Yang-model heeft meerdere bladknooppunten. Het wordt aangeraden om specifiek te zijn over de benodigde informatie en de informatie die niet vereist is. Het wordt aanbevolen om de Yang-modellen te verkennen en het gegevenspad te identificeren dat vereist is bij de telemetrie-gebruikscases.

Overweeg de hoeveelheid streaming gegevens

De totale hoeveelheid telemetriegegevens die worden gestroomd zal overweging over de volgende punten nodig hebben:

1. Bandbreedtetoewijzing binnen het netwerk
2. QoS
3. Prestaties van extra applicaties/software zoals VerzamelaarssoftwareTime Series-databaseAnalyse-/visualisatiesoftware

4. Efficiëntie coderen (besproken in de sectie "Telemetry Design Guidelines") heeft direct effect op de hoeveelheid gegevens die worden gestreamd. Compacte GPB wordt waar mogelijk aanbevolen.
5. Verzamelinterval heeft direct effect op meerdere aspecten, waaronder maar niet beperkt tot het volgende. Bandbreedtegebruik in het netwerkOpslagvereiste voor de databasePrestaties van het apparaat dat de gegevens stroomt

Aanbevolen wordt de frequentie van de verzameling te evalueren op basis van de toepassingsvereisten.

In het algemeen wordt aanbevolen het filteren van ongewenste gegevens aan de bron of op de bestemming als haalbaar te beschouwen. We hebben wel de mogelijkheid om ongewenste gegevens te filteren. Er kan filtering op twee niveaus plaatsvinden: -

1. Bij de Bron - de apparaten die de gegevens stromen.
2. Bij de Bestemming - de Collector die gegevens verzamelen en normaliseren. (Het filteren op Collector valt buiten het bereik van dit document)

In het volgende voorbeeld worden gegevens alleen gefilterd voor Hundered Gig interfaces binnen de sensorpaden door het toepassen van wildcards.

```
sensor-path Cisco-IOS-XR-infra-statsd-oper:infra-statistics/interfaces/interface[interface-name='HundredGigE*']/latest/generic-counters
```

Referenties

https://blogs.cisco.com/sp/the-limits-of-snmp

https://blogs.cisco.com/sp/why-you-should-care-about-model-driven-telemetry

https://www.cisco.com/c/en/us/td/docs/iosxr/asr9000/telemetry/b-telemetry-cg-asr9000-61x.html

Over deze vertaling

Cisco heeft dit document vertaald via een combinatie van machine- en menselijke technologie om onze gebruikers wereldwijd ondersteuningscontent te bieden in hun eigen taal. Houd er rekening mee dat zelfs de beste machinevertaling niet net zo nauwkeurig is als die van een professionele vertaler. Cisco Systems, Inc. is niet aansprakelijk voor de nauwkeurigheid van deze vertalingen en raadt aan altijd het oorspronkelijke Engelstalige document (link) te raadplegen.