ASR9K型号驱动遥测白皮书

目录

简介

受众

本白皮书旨在帮助客户快速了解一般模式驱动遥测(MDT)功能及其在聚合服务路由器9000(ASR9K)中的实施方式，包括一些设计指南和配置详细信息。它还包括一些部署注意事项，这对使用ASR9K部署此功能很有帮助。总之，本白皮书可作为任何从事此功能工作的人员的快速参考指南。

虽然遥测作为通用功能引入，但重点是ASR9K实施；即，ASR9K平台并不支持其他思科平台支持的所有功能，某些功能实施可能特定于ASR9K。

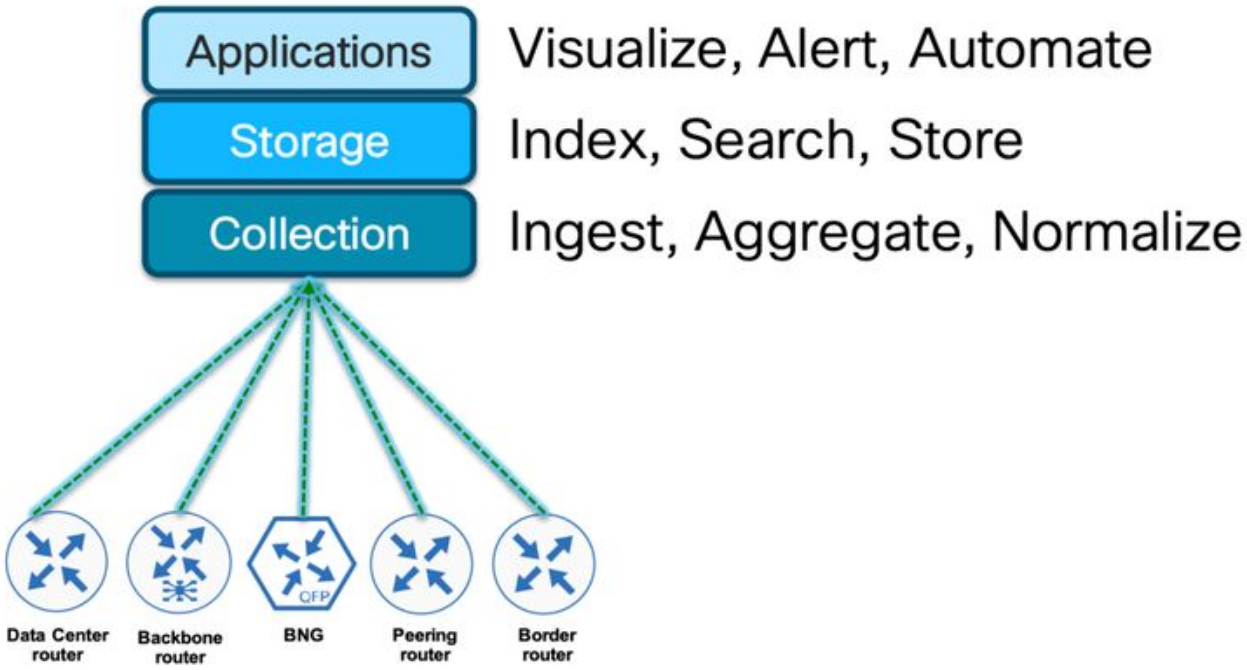
遥测简介

首先，简单地说，遥测是收集有用操作数**据的过程**。根据维基百科，遥测是一个自动通信过程，通过它可以在远程或不可达的点收集测量数据和其他数据，并将其传输到接收设备以供监控。遥测词本身源自希腊语根：tele = remote，metron = measure。

对于网络管理，网络运营商长期以来一直依赖简单网络管理协议(SNMP)。尽管SNMP被广泛用于网络监控，但从未用于配置，即使使用SNMP进行配置的能力始终存在。操作员已编写自动化脚本来处理日常配置任务，但脚本对于这类任务具有挑战性，并且难以管理。

因此，操作员逐渐转向数据模型驱动管理。例如，网络配置基于netconf等协议推送的YANG数据模型。现在，仅推送配置并不意味着配置的服务正在运行，必须存在一种机制，能够在配置时监控服务操作数据。这是操作员数据模型的地方；遥测使用哪些信息从设备推送信息；有帮助。因此，配置是以YANG数据模型驱动的，因此必须同时验证服务；对于遥测来说，为了具有相同的对象语义。因此，该术语称为**模型驱动遥测**或流遥测。

cXR（32位IOS XR）自6.1.1版开始引入模型驱动遥测(MDT)，它允许以近实时方式收集和测量关键数据，从而快速解决现代网络的大多数运营问题。



高级遥测架构

MDT利用网络设备支持的结构化数据模型，并提供在这些数据模型中定义的关键数据。**遥测**技术可帮助客户使用一个通用网络管理系统、流程和应用来管理其多供应商网络，因为从网络收集的数据是基于标准的，并且在整个供应商实施中是统一的。

而不是等待从集中管理站（通常是SNMP NMS）检索（拉取）数据；使用MDT，网络设备主动发送(推**送)与网络重要功能有关的性能数据，例如数据包转发信息、错误统计、系统状态、CPU和内存资源等**等。

为什么使用遥测

为了分析和故障排除而收集数据一直是监控网络运行状况的一个重要方面。有多种机制（例如SNMP、CLI和Syslog）可用于从网络收集数据。虽然这些方法为网络服务了很长时间，但并不适用于自动化需求的现代网络，大规模服务是现代网络的基础。网络运行状况信息、流量统计信息和关键基础设施信息被发送到NMS中的远程站点，用于增强运营性能并减少故障排除时间。客户端轮询所有网络节点的snmp等拉模式效率不高。当要轮询的客户端数量较多时，网络节点上的处理负载会增加。相反，推送模式能够连续将数据流从网络传出并通知客户端。遥测启用推送模型，提供对监控数据的近实时访问。

流遥测提供了一种机制，可以从路由器中选择所需数据，并以标准格式将其传输到远程管理站进行监控。该机制可根据实时数据对网络进行微调，这对网络的无缝运行至关重要。通过遥感勘测可获得更精细的粒度和更频繁的数据，从而实现了更好的性能监控，从而可更好地进行故障排除。

它有助于提高网络中服务效率的带宽利用率、链路利用率、风险评估和可扩展性。有了流传输遥感勘测，网络运营商可以使用更多接近实时的数据，这有助于改善决策。

从SNMP转移的需要

SNMP问世已有三十年，其运行方式没有改变以符合现代网络的监控需求。真正的问题在于SNMP的执行速度。

SNMP带来的三个主要挑战实际上是其基本操作行为的一部分，因此SNMP提供很少/没有改进空间，遥测技术可从根本上解决所有这三大挑战。

- **执行速度和实时监控需求**

SNMP使用PULL Model - GetBulk/GetNext操作，这些操作通过从一个列遍历到另一个列来线性工作。此外，如果大型表无法容纳在一个数据包中，则需要多个请求。 这是导致SNMP速度减慢的最大瓶颈，而且发送的数据通常在几分钟内因特定时间因素而过时。这种延迟是现代网络监控要求无法接受的。

MDT（模型驱动遥测）使用PUSH模型，并且固有不受以上列出的限制，因为它知道以什么时间间隔向谁发送什么数据。它只需一次查找即可收集数据，并使用预构建的内部模板实现极快的内部操作速度，从而能够以极少的时间交付更多数据。

- **额外开销和缺乏优化选项**

由SNMP提取的数据实际上被存储为内部数据结构，需要由节点进行内部转换。这是网络节点将内部数据结构映射为SNMP格式的幕后额外工作。执行了一些内部优化，但是这些优化仍然不够。

另一方面，遥测直接提取内部数据结构，并在发送该数据之前执行最小的处理，从而以尽可能最少的时间和精力提供最更新的数据。

- **工作负荷的线性性质**

即使我们在同一时间轮询相同的准确数据，每个额外的轮询站都会增加节点的工作负载。 从多个轮询站并行访问同一MIB可能导致响应缓慢和CPU使用率较高。这一点在大型表的情况下尤其明显，在该情况下，多个站点访问同一MIB表的不同部分。

另一方面，如果多个目标需要相同数据，则遥测需要一次提取数据并复制数据包。 推动模式优于SNMP拉动模式，有利于提高速度和规模。

使用MDT时，数据收集方法及其基本原则在下表中列出，并与SNMP技术要点进行比较。

简单网络管理协议 (SNMP)	**模型驱动遥测(MDT)**
非实时信息	实时信息
扩展性差	高度可扩展
拉模型	推送模型
非自动化	自动化就绪/数据模型驱动

流遥测的优点

流式实时遥测数据在以下方面非常有用：

容量规划/流量优化：当网络中经常监控带宽利用率和数据包丢弃时，更易于添加或删除链路、重定向流量、修改策略等。使用快速重新路由等技术，网络可以切换到新路径，并且重新路由的速度比SNMP轮询间隔机制更快。流传输遥测数据有助于为更快的流量提供快速响应时间。

更好的可视性：帮助快速检测和避免网络中出现问题情况后出现的故障情况。

型号驱动遥测技术规格

以下部分介绍IOS XR型号驱动遥感勘测（也称为MDT）的技术功能和主要组件。

遥测功能

遥测框架分为三个独立且相互关联的功能块。

第一个模块是关于**数据表示**的，也就是如何组织信息中的分析或测量。

第二个块是关于**编码**。每个采样间隔，遥测都会将以上测量数据转换为可在线路上序列化的格式。当然，另一端的控制器必须能够解码数据，以便拥有设备发送的原始数据的相同副本。

最后一个块是关于**transport**。这是用于在设备之间传输数据的协议栈。

下表总结了模型驱动遥测构建块的主要结构：

功能	**组件**
数据表示	YANG数据模型
编码	GPB/GPB自我描述
传输	TCP/gRPC

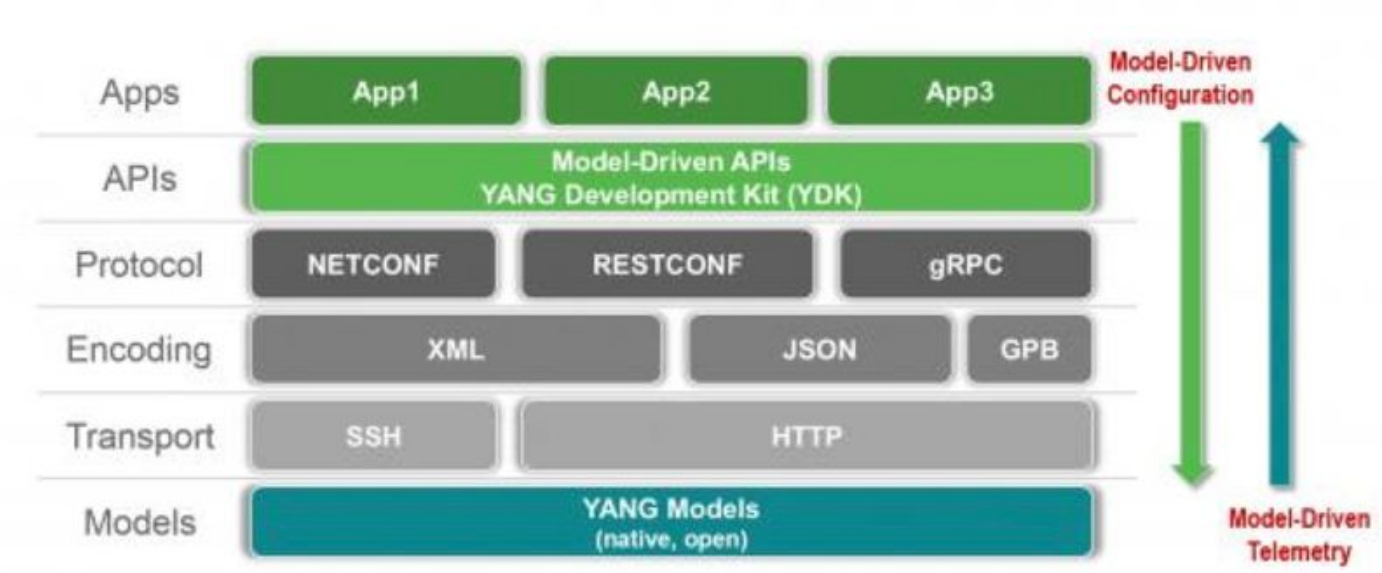
表 3 遥测构建块

遥测组件

在了解遥测和基础配置部分如何工作之前，了解遥测的不同组件非常重要，以便评估最佳设置。遥测依赖于IOS XR可编程性堆栈，其中新的基础设施框架提供了网络自动化的基本功能。

YANG最近成为数据建模的标准，思科可编程性堆栈使用此标准构建结构化数据集，这些数据集可以编码并在网络上尽可能快地传输。YANG的灵活性为同时用作自动化流程配置工具带来了巨大优势。这些数据模型结合特定的编码格式和传输协议，使MDT成为网络分析的完整解决方案。

对于模型驱动遥测设置，YANG数据模型将成为关键组件，以支持收集和分析所需的数据流。



IOS XR可编程性堆栈

杨

Yang被定义为“数据建模语言，用于为网络管理协议对配置数据、状态数据和通知进行建模。” 由于

YANG与典型编程语言架构的分离性，它可以与各种工具进行交互。

YANG建模数据结构是围绕模块和子模块的概念构建的，它以树形方式定义数据的层次结构，可用于多种操作，包括配置操作和通知处理。

有多种可用的YANG模型来源，其中以下三种被视作主要来源：

- 本机型号/思科特定型号
- OpenConfig
- IETF

思科专用型号：这些也称为**本地模式**，由包括思科在内的各种设备供应商发布。 例如Cisco-IOS-XR-ptp-oper.yang

OpenConfig型号： OpenConfig是一个由网络操作员组成的非正式工作组。OpenConfig定义了常用的YANG模型，所有供应商都应该支持这些模型来配置任务关键型功能。 例如openconfig-interfaces.yang

IETF模型:IETF还定义了几种常见YANG模块，这些模块描述了接口、QOS的基本配置，并定义了其他常用数据类型（如Ipv4、IPv6等）。 例如ietf-syslog-types.yang

思科支持可用的Openconfg型号。 供应商正在向标准化数据建模方法融合，以支持多供应商环境。

Yang模**型有**三种类型：

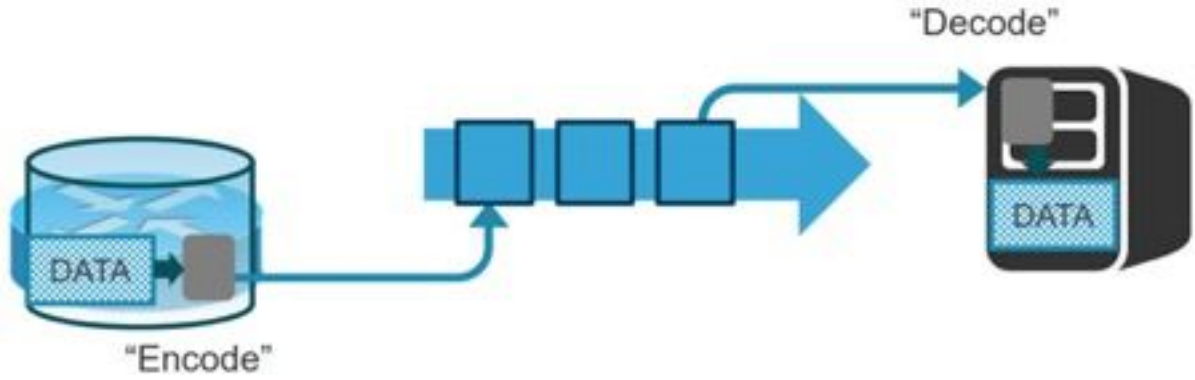
1. 运行
2. 配置
3. 操作

遥测只关**于Yang操作模**型，可标识为***-oper-*.yang。**

YANG定义在RFC 7950中：https://tools.ietf.org/html/rfc7950。

编码

编码（或“序列化”）将数据（对象、状态）转换为可通过网络传输的格式。当接收器对数据进行解码（“反序列化”）时，它拥有原始数据的语义相同的副本。

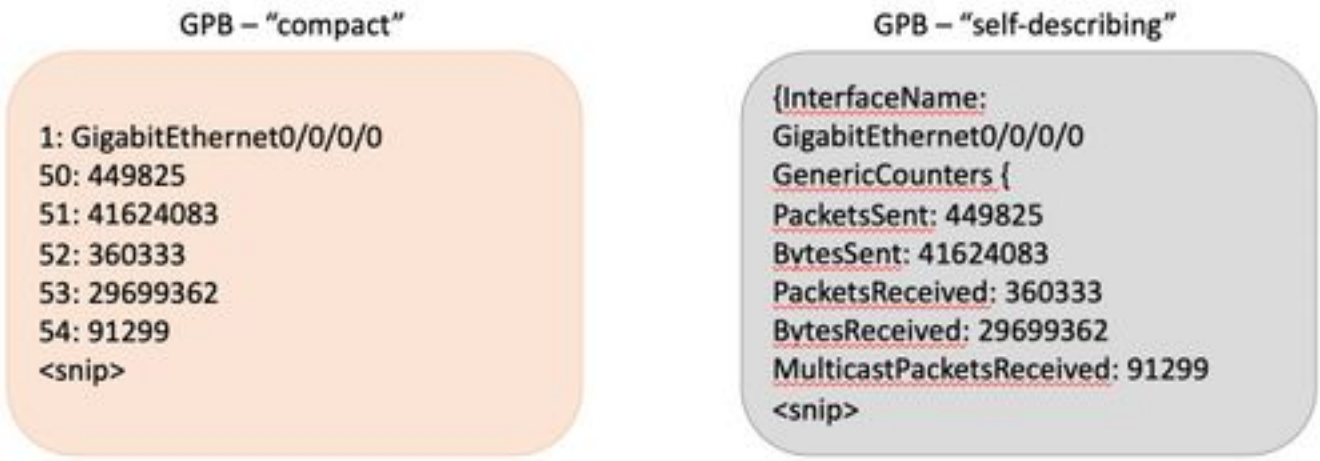


在遥测的早期开发阶段，XML因其基于标签的结构而最初被视为首选的编码格式。但是XML的问题在于其编码结构不紧凑。GPB（Google协议缓冲区）最终被思科采用，因为它提高了编码操作的效率和速度。

GPB有两种形式作为遥测流传输的编码选项：

1. 紧凑型GPB
2. 自描述GPB

两种GPB遥测格式之间的主要区别在于它们如何表示和编码遥测数据流中的密钥。



JSON是另一个人类友好的编码模式，非常容易理解，几乎任何应用程序都能解码。

从部署角度来看，编码方案的优缺点非常少。有关各种编码方案的比较见遥测设计指南一节。

传输

遥测为传输协议提供三种可能的选择：

- TCP
- gRPC
- UDP

遥测还定义了两种不同的启动模式，以启动节点和收集器之间的会话：

- 拨出
- 拨入

两种模式之间的区别仅在于传输会话的建立方式。

在拨出会话期间，设备通过向预配置的服务器端口发送syn数据包来启动连接。建立连接后，数据流会立即从设备推离。

对于拨入会话，路由器会被动侦听等待服务器连接的tcp端口。

但是，建立会话后，路由器不会由服务器自身轮询，因为设备仍负责数据推送操作。在MDT中，数据轮询的概念甚至不存在。

默认情况下，TCP是遥测的预定义传输方法，因为它非常可靠且易于配置为选项。

gRPC是一个现代开源框架，可在任何环境中运行。它构建在HTTP/2之上，提供一组增强且丰富的功能。

基于节奏的遥测与基于事件的遥测

来自订用数据集的数据以配置的周期性间隔或仅在发生事件时才被流式传输到目的地。此行为取决于MDT是配置为基于节奏的遥测还是基于事件的遥测。

基于事件的遥测的配置类似于基于节奏的遥测，仅样本间隔作为区分因素。将示例间隔值配置为零会设置基于事件的遥测的订用，而将间隔配置为任何非零值会设置基于速率的遥测的订用。

建议对变更相关事件使用事件驱动遥测。

遥测设计准则

如前所述，遥测堆栈中有许多组件，以下是在XR设备上实施遥测时应考虑的一些准则。

如何选择编码方案

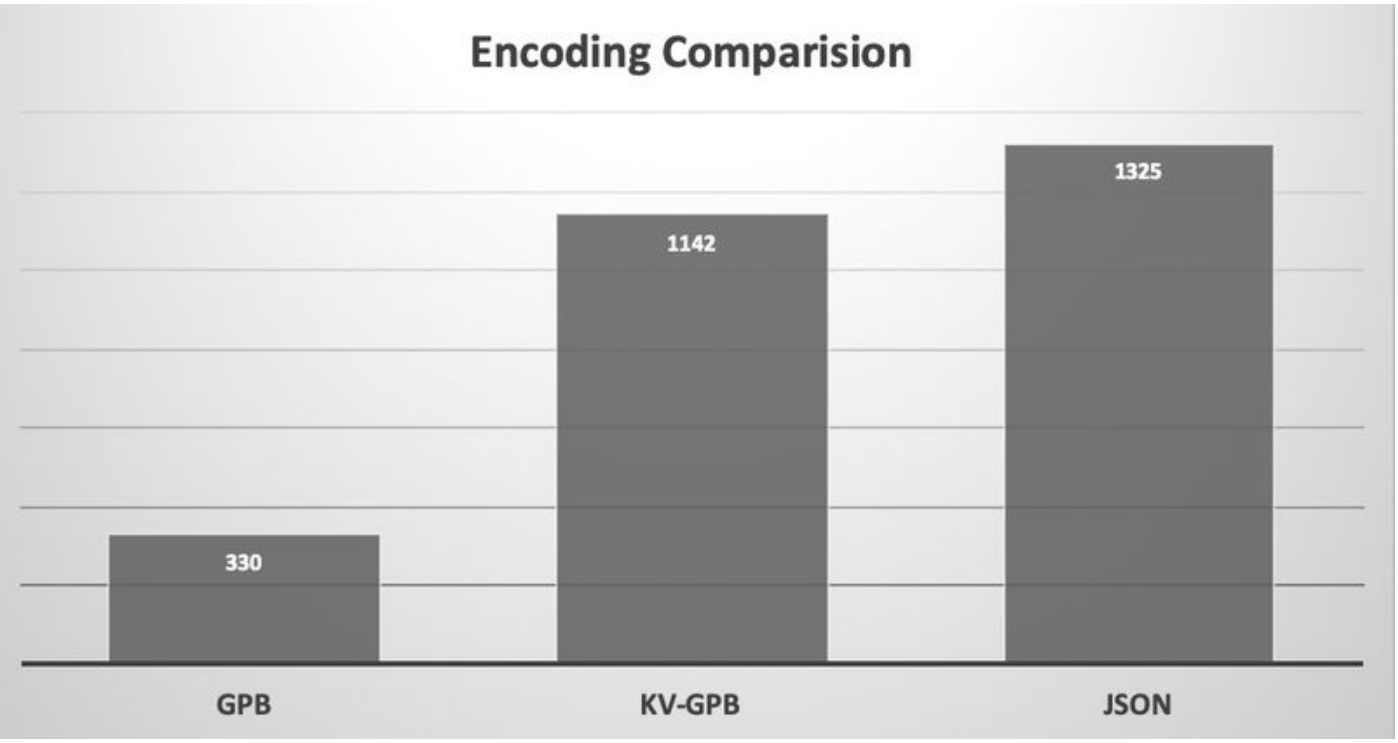
如上所述，编码或序列化将数据（对象、状态）转换为可通过网络传输的格式。当接收器对数据进行解码或反序列化时，它拥有原始数据的语义相同的副本。

各种编码选项在导线效率和易用性方面各有不同。

编码	简要说明	线上的效率	其他注意事项
GPB（紧凑型）	万物二进制 （除了字符串值） 速度提高2倍，操作更复杂 （但与SNMP无关）	高	每个型号的Proto文件
GPB - KV（密钥值对）	字符串键和二进制值 （字符串值除外） 3倍大， 原生型号：仍需要关键名称的启发式方法	中到低	用于煎煮的单个.proto
JSON	Everything字符串：键和值	低	友善。易于阅读、应用且易于分析

GPB-KV为编码模式提供了一个良好和平衡的中间点。

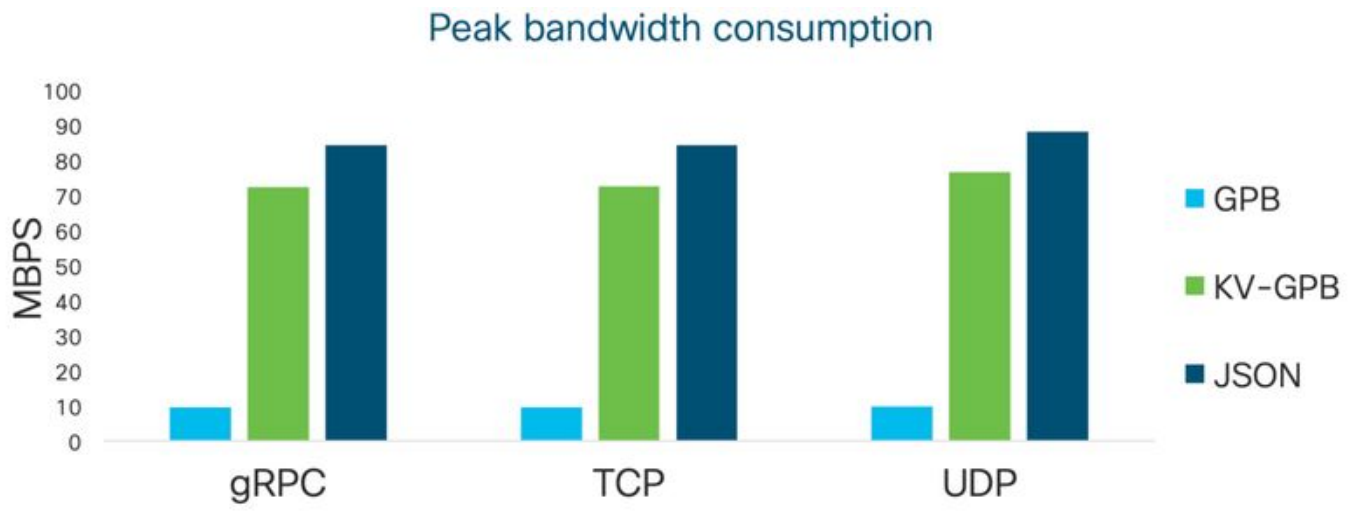
关于所选编码方案的消息长度，下面是线路上的比较。



编码比较 — 消息长度（以字节为单位）

传输网络设计注意事项

不同的编码选项有不同的带宽要求。在考虑遥测时，网络运营商需要根据所选的编码方案进行充分的带宽调配。这只是一个公平的想法，低于每个编码模式所消耗的带宽。



网络带宽比较

思科建议使用KV-GPB。它是效率和便利性之间的一个很好的中点。

评估遥测配置选项

配置模型驱动遥测时，操作员应了解遥测涉及的所有不同组件。根据上文详述的传输、编码和流方向可用选项，您可以进一步选择更适合环境的组合。

四个关键组件是

1. 传输

2. 编码
3. 会话方向
4. YANG数据模型

传输： 如上所述，节点可以使用TCP、UDP或gRPC over HTTP/2来传输遥测数据。

虽然TCP是简便性的首选方案，但gRPC提供可选的TLS功能，从安全角度来看，此功能可能是额外的优势。

编码： 路由器可以交付两种不同形式的Google协议缓冲区的遥测数据：紧凑和自描述GPB。

紧凑型GPB是最有效的编码，但对于流传输的每个YANG模型都需要一个唯一的.proto。自描述GPB的效率较低，但它使用单个.proto文件来解码所有YANG模型，因为密钥在.proto中作为字符串传递。

会话方向： 在遥测部署中启动会话有两种选项。路由器可以“拨出”到收集器，或收集器可以“拨入”到路由器。

YANG是行业认可的数据建模标准，思科可编程性堆栈还使用该标准形成结构化数据集，该数据集可通过网络进行编码并尽快传送。

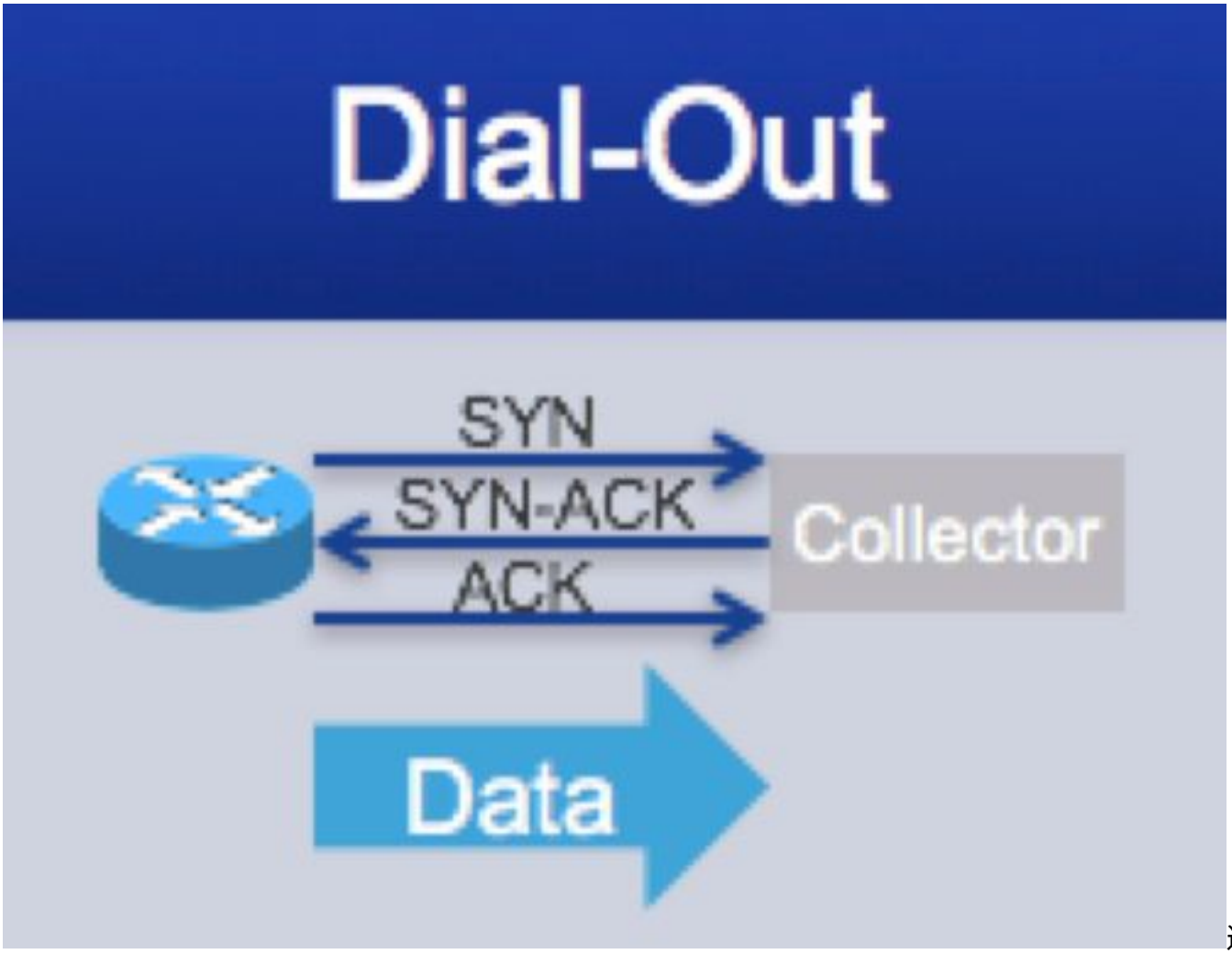
这些数据模型与上文讨论的特定编码格式和传输协议相结合，使模型驱动遥测(MDT)成为用于分析的完整解决方案。

遥测配置示例

IOS-XR

拨出配置中断

在拨出模式下，路由器负责发起到收集器的TCP会话，并发送订阅中传感器组指定的数据。



遥测拨出从配置角度来看，遥测配置是一个三步流程。首先，我们识别要传输的信息，并在传感器组配置下捕获该信息。其次，我们确定信息流传输的目的地，并在目标组配置下捕获该信息。第三，我们使用前两个步骤中确定的信息来配置实际订用。

1. 定义传感器组
2. 定义目标组
3. 定义订用

定义传感器组

传感器组配置标识要流传输的信息。以下配置模板提供配置传感器组所需的配置。

```
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#
```

以下示例展示路由器CLI中的实际示例，其中显示传感器组配置的实际示例：

```
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#
```

我们可以将多个传感器路径作为同一SensorGroup定义的一部分：

```
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#
```

定义目标组

目标组配置确定信息流传输到的目标。

它有三个关键参数

1. 会话方向
2. 要使用的编码
3. 要使用的传输协议

下面是一个示例：

```
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#
```

定义订用

订阅将传感器组和目标组信息绑定在一起，作为配置的最后一部分。示例间隔被定义为订用的一部分。

下面是一个示例：

```
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#
```

完整配置示例

```
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#
```

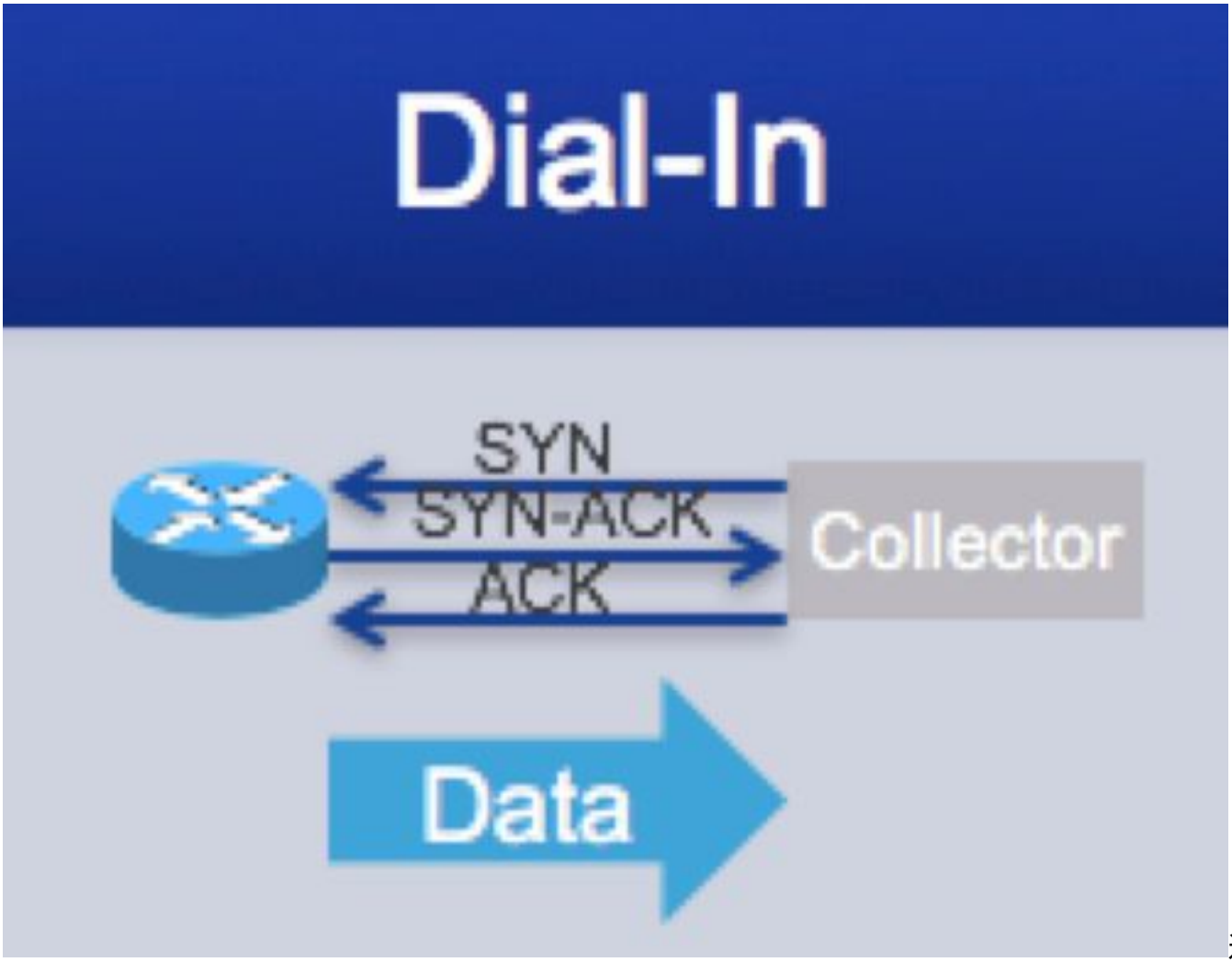
外拨的优点

- 传输选项具有更大的灵活性。
- 无需为入站管理流量打开端口。
- 任播和负载均衡。

拨入配置中断

在拨入模式下，MDT收集器/接收器/协调器拨入路由器并动态预订一个或多个传感器路径或预订。路由器充当服务器，客户端充当接收器。

仅形成一个会话，路由器通过同一会话传输遥测数据。当接收方取消订阅或会话终止时，此动态订阅终止。



遥测拨入

由于收集器“拨入”路由器，因此无需在配置中指定每个MDT目标。只需在路由器上启用gRPC服务，连接您的客户端，并动态启用所需的遥测订阅。

从配置角度来看，遥测配置是一个类似于上述过程的三步过程。首先，需要启用gRPC。其次，我们确定信息流传输的目的地，并在传感器组配置下捕获该信息。第三，我们使用前两个步骤中确定的信息来配置实际订用。

1. 启用gRPC
2. 定义传感器组
3. 定义订用

单击展开

只有gRPC支持拨入模式

只有gRPC支持拨入模式

启用gRPC

首先，我们需要在路由器上启用gRPC服务器以接受来自收集器的传入连接。

单击展开

- <port-number>的范围是从57344到57999。如果端口号不可用，则会显示错误。

<port-number>的范围是从57344到57999。如果端口号不可用，则会显示错误。

```
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#
```

定义传感器组

传感器组配置标识要流传输的信息。以下配置模板提供配置传感器组所需的配置。

以下示例展示路由器CLI中的实际示例，其中显示传感器组配置的实际示例

```
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#
```

定义订用

订阅将传感器组和gRPC绑定在一起，作为配置的最终部分。示例间隔被定义为订用的一部分。

以下配置模板提供配置订用所需的配置。

以下示例展示来自路由器CLI的实际示例，我们在其中创建预订，并将传感器组和目标组绑定在一起，同时定义采样速率。

```
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#
```

完整配置模板和示例

```
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#
```

拨入的优点

- 用于配置和流传输的单一信道
- 路由器/设备上的侦听端口
- 临时连接
- 当前只有gRPC/gNMI可用

事件驱动遥测

在事件驱动遥测中，只有在发生事件时，才会从订用数据集中流式传输数据。

事件驱动遥测配置

基于事件的遥测的配置类似于基于时间的遥测，而事件驱动遥测的配置唯一区别是采样间隔的配置。将示例间隔值配置为零会将订用设置为基于事件的遥测。

完整的配置模板和拨出示例

```
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#
```

以下示例展示路由器CLI的实际示例。

```
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#
```

拨入的完整配置模板和示例

```
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#
```

以下示例展示路由器CLI的实际示例。

```
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#
```

使用SHOW命令验证遥测

从路由器的角度来说，我们可以验证为每个传感器组、目标组和订用配置的参数

```
// 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#
```

遥测收集堆栈

除了路由器配置之外，基于遥测的解决方案还需要多个组件，如收集器、数据库和监控/分析软件。这些组件可以单独配置，也可以作为单个综合产品的一部分。

- 应安装解码器以接收入站数据包并将它们传送以进一步存储。
- 需要时序数据库（也称为TSDB）来存储流传输的信息。
- 还需要图形工具来显示从内部数据库获取的数据。

它超出了详细描述集合堆栈的范围。Cisco Crossworks Health Insights支持零接触遥测，自动为设备调配遥测配置，并在时序数据库(TSDB)中创建表/模式。它简化了收集和清理数据的运营和网络管理开销，从而使运营商能够专注于其业务目标。使用通用收集器通过SNMP、CLI和模型驱动的遥感勘测收集网络设备数据，可以避免数据重复，还可以减少设备和网络的负载。

网络中遥测的部署注意事项

以下是分析网络中遥测部署时需要考虑的各种因素。

扩展

遥测可以传输大量数据，建议仔细考虑可扩展性方面。

仅流式传输所需数据

每个Yang模型将具有多个枝叶节点。建议具体说明所需的信息和不要求的信息。建议探索Yang模型并确定遥测使用案例所需的数据路径。

考虑流数据量

流传输的遥测数据总量需要考虑以下方面：

1. 网络中的带宽分配
2. 服务质量
3. 其他应用程序/软件(如 收集器软件时序数据库分析/可视化软件
4. 编码效率（在“遥测设计指南”一节中讨论）直接影响流传输的数据量。如果可行，建议使用紧凑型GPB。
5. 收集间隔直接影响多个方面，包括但不限于以下方面。 网络中的带宽利用率数据库存储要求流式传输数据的设备的性能

建议根据应用要求评估收集的频率。

总之，建议考虑在认为可行的情况下在源或目标位置过滤不需要的数据。我们确实可以选择过滤不需要的数据。过滤可以在两个级别上执行： -

1. 源设备 — 传输数据的设备。
2. 在目的地 — 收集器收集并规范化数据。（在收集器进行过滤不在本文档的讨论范围之内）

以下示例显示通过应用通配符仅对传感器路径内的搜索千兆接口过滤数据。

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

参考

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

关于此翻译
思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。