ASR9K 모델 기반 텔레메트리 백서

목차

소개

대상

이 백서는 고객이 MDT(Model Driven Telemetry) 기능에 대한 전반적인 이해 및 일부 설계 지침 및 컨피그레이션 세부사항을 포함하여 ASR9K(Aggregation Services Router 9000)에서 어떻게 구현되었는지를 쉽게 이해할 수 있도록 돕기 위한 것입니다. 또한 ASR9K를 사용하여 이 기능을 구축하는 데 도움이 되는 몇 가지 구축 고려 사항도 포함되어 있습니다. 전반적으로 이 백서는 이 기능에 대해 작업하는 모든 사람에게 빠른 참조 가이드가 될 수 있습니다.

텔레메트리는 일반 기능으로 소개되지만 ASR9K 구현에 초점을 맞춥니다. 즉, 다른 cisco 플랫폼에서 지원하는 모든 기능이 ASR9K 플랫폼에서 지원되지는 않으며 일부 기능 구현이 ASR9K에 특정될 수 있습니다.

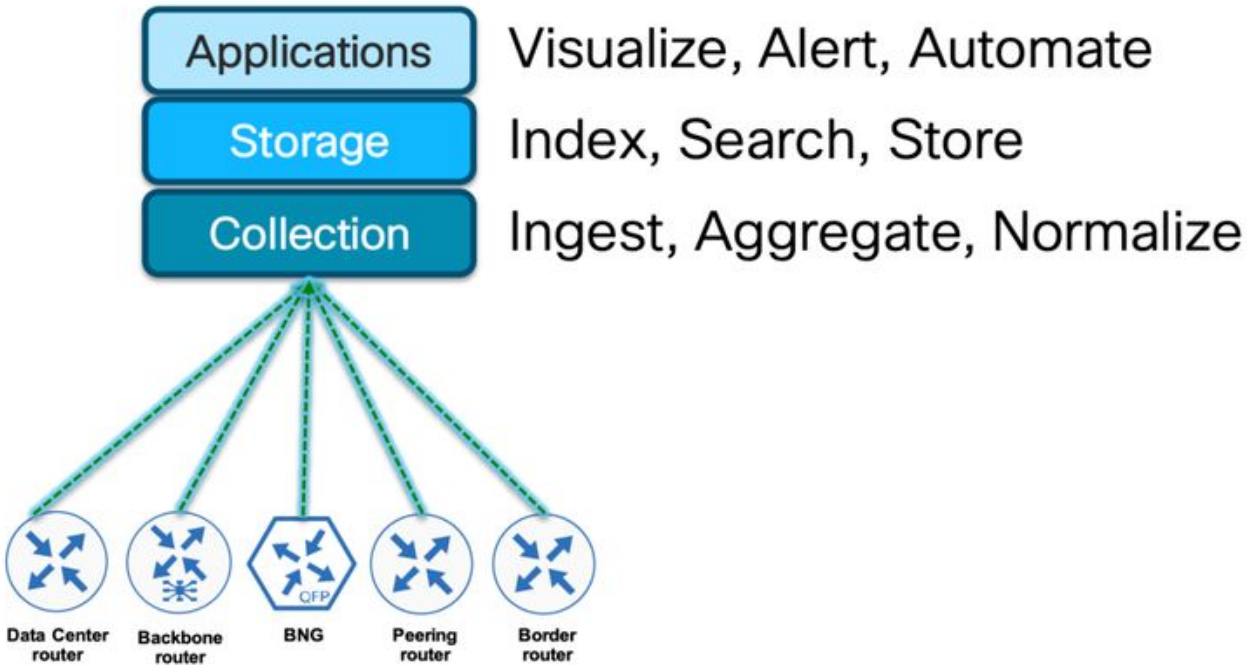
텔레메트리 소개

간단히 말해, 텔레메트리는 **유용한 운영** 데이터의 수집 **프로세스입니다**. Wikipedia에 따르면, 텔레메트리는 측정 및 기타 데이터를 원격 또는 액세스 할 수없는 지점에서 수집하고 모니터링을 위해 수신 장비로 전송하는 자동화된 통신 프로세스입니다. 텔레메트리 단어 자체는 그리스어 뿌리에서 파생됩니다: tele = remote, metron = measure.

네트워크 관리를 위해 네트워크 운영자는 SNMP(Simple Network Management Protocol)에 의존한지 오래되었습니다. SNMP는 네트워크 모니터링에 널리 채택되지만, snmp를 사용한 구성 기능이 항상 존재함에도 불구하고 구성에 사용되지 않았습니다. 운영자는 일상적인 컨피그레이션 작업을 처리하기 위해 자동화 스크립트를 작성했지만 스크립트는 그러한 작업에 대해 까다롭고 관리하기가 어렵습니다.

따라서 운영자는 데이터 모델 기반 관리로 전환했습니다. 네트워크 컨피그레이션은 예를 들어 netconf와 같은 프로토콜에서 푸시하는 YANG 데이터 모델을 기반으로 합니다. 이제 컨피그레이션을 푸시한다고 해서 구성된 서비스가 실행 중이라는 의미는 아닙니다. 컨피그레이션과 동시에 서비스 운영 데이터를 모니터링할 수 있는 메커니즘이 있어야 합니다. 바로 여기에 오퍼레이션 데이터 모델이 있습니다. 이 모델은 텔레메트리를 사용하여 정보를 디바이스 외부로 내보내며 도움이 됩니다. 따라서 컨피그레이션은 YANG 데이터 모델 기반이므로 동일한 개체 의미 체계를 갖기 위해서는 Telemetry의 경우와 마찬가지로 서비스 검증이어야 합니다. 따라서 이 용어를 **모델 기반 텔레메트리 또**는 스트리밍 텔레메트리라고 합니다.

MDT(Model Driven Telemetry)는 **cXR**(32비트 IOS XR)에서 릴리스 **6.1.1** 이후 도입되었으며 거의 실시간으로 중요 데이터를 수집하고 측정하여 최신 네트워크의 운영 문제에 대한 빠른 해결책을 제공합니다.



하이 레벨 텔레메트리 아키텍처

MDT는 네트워킹 디바이스에서 지원하는 정형 데이터 모델을 활용하고 이러한 데이터 모델에 정의된 중요한 데이터를 제공합니다. **텔레메트리**는 네트워크에서 수집된 데이터**가 표준을 기반으로 하며 공급업체 구현**에서 통일되어 있기 때문에 고객이 하나의 공통 네트워크 관리 시스템, 프로세스 및 애플리케이션을 사용하여 멀티벤더 네트워크를 **관리할** 수 있도록 지원합니다.

중앙 집중식 관리 스테이션(일반적으로 SNMP NMS)에서 데이터 검색(풀)을 기다리는 대신, MDT를 통해 네트워크 디바이스는 패킷 전달 정보, 오류 통계, 시스템 상태, CPU 및 메모리 리소스 등과 같은 네트워크 중요 기능과 관련된 성능 데이터를 능동적으로 전송(**푸시**)합니다.

텔레메트리를 선택해야 하는 이유

분석 및 문제 해결을 위해 데이터를 수집하는 것은 네트워크 상태를 모니터링하는 데 있어 항상 중요한 부분이었습니다. SNMP, CLI 및 Syslog와 같이 네트워크에서 데이터를 수집하는 몇 가지 메커니즘을 사용할 수 있습니다. 이러한 방법은 오랫동안 네트워크에 사용되었지만 자동화에 대한 수요가 있는 최신 네트워크에는 적합하지 않지만 규모에 맞는 서비스는 기본입니다. 네트워크 상태 정보, 트래픽 통계 및 중요 인프라 정보는 NMS의 원격 스테이션으로 전송되며, 이 원격 스테이션은 운영 성능을 개선하고 문제 해결 시간을 단축하는 데 사용됩니다. 클라이언트가 모든 네트워크 노드를 폴링하는 snmp와 같은 풀 모델은 효율적이지 않습니다. 폴링할 클라이언트 수가 많을수록 네트워크 노드의 처리 로드가 증가합니다. 반대로, 푸시 모델에는 네트워크 외부로 데이터를 지속적으로 스트리밍하고 클라이언트에 알릴 수 있는 기능이 있습니다. 텔레메트리를 통해 푸시 모델을 활성화하여 모니터링 데이터에 거의 실시간으로 액세스할 수 있습니다.

스트리밍 텔레메트리는 라우터에서 원하는 데이터를 선택하고 모니터링을 위해 원격 관리 스테이션에 표준 형식으로 전송하는 메커니즘을 제공합니다. 이 메커니즘은 실시간 데이터를 기반으로 네트워크를 정밀하게 조정할 수 있게 해주며, 이는 원활한 작동을 위해 매우 중요합니다. 텔레메트리를 통해 사용할 수 있는 데이터의 세분화 및 빈도가 높으므로 성능 모니터링이 향상되므로 문제 해결이 향상됩니다.

네트워크, 링크 활용, 위험 평가 및 확장성에서 서비스 효율성이 더 높은 대역폭 활용을 지원합니다. 스트리밍 텔레메트리를 사용하면 네트워크 운영자가 폐기하는 즉시 더 많은 실시간에 가까운 데이터를 사용할 수 있으므로 의사 결정을 개선하는 데 도움이 됩니다.

SNMP에서 벗어나야 할 필요성

SNMP는 30년 동안 사용되어 왔으며 최신 네트워크의 모니터링 요구 사항에 맞게 운영 방식이 바뀌지 않았습니다. 진짜 문제는 SNMP의 실행 속도입니다.

SNMP가 제기하는 세 가지 주요 과제는 실제로 기본적인 운영 동작의 일부이므로 SNMP는 개선의 여지가 거의 없으며 텔레메트리를 통해 세 가지 문제를 모두 해결할 수 있습니다.

- **실행 속도 및 실시간 모니터링 필요**

SNMP는 한 열에서 다른 열로 테이블을 이동하여 선형 방식으로 작동하는 PULL Model - GetBulk / GetNext 작업을 사용합니다. 또한 하나의 패킷에 들어갈 수 없는 큰 테이블의 경우 여러 요청이 필요합니다. 이는 SNMP의 속도를 떨어뜨리는 가장 큰 병목 현상으로, 몇 분 내에 특정 시간 요인으로 인해 전송되는 데이터가 오래되는 경우가 많습니다. 이러한 지연은 현대의 네트워크 모니터링 요구 사항에는 적합하지 않습니다.

MDT(Model Driven Telemetry)는 PUSH 모델을 사용하며, 어떤 데이터를 누구에게 어떤 간격으로 전송해야 하는지 알기 때문에 위에서 언급한 제약으로부터 본질적으로 벗어납니다. 데이터를 수집하기 위해 단 하나의 조회만 필요하며, 내부 작업의 초고속 수행을 위해 사전 구축된 내부 템플릿을 사용하므로 훨씬 적은 시간에 훨씬 많은 데이터를 제공할 수 있습니다.

- **추가 오버헤드 및 최적화 옵션 부재**

SNMP에서 가져오는 데이터는 실제로 내부 데이터 구조로 저장되며 노드에 의해 내부적으로 변환되어야 합니다. 이는 네트워크 노드가 내부 데이터 구조를 SNMP 형식으로 매핑하는 배후의 추가 작업입니다. 내부 최적화가 수행되지만 아직 충분하지 않습니다.

반면, 텔레메트리는 내부 데이터 구조를 직접 꺼내고 이 데이터를 전송하기 전에 최소 수준의 처리를 수행하여 가장 업데이트된 데이터를 가장 적은 시간과 노력으로 제공합니다.

- **워크로드의 선형 특성**

동일한 정확한 데이터를 동일한 시간에 폴링하는 경우에도 모든 추가 폴링 스테이션은 노드에 추가 워크로드를 초래합니다. 여러 폴링 스테이션에서 동일한 MIB를 병렬로 액세스하면 응답이 느려지고 CPU 사용률이 높아질 수 있습니다. 이는 특히 여러 스테이션이 동일한 MIB 테이블의 서로 다른 부분에 액세스하는 대형 테이블의 경우 분명합니다.

반면 여러 대상에서 동일한 데이터가 필요한 경우 텔레메트리는 데이터를 한 번 끌어와 패킷을 복제해야 합니다. Push 모델이 SNMP Pull을 능가하는 속도와 확장성을 제공합니다.

MDT를 사용하면 데이터 수집에 대한 접근 방식이 크게 달라지고 기본 원칙이 아래 표에 나열되어 SNMP 기술 핵심 사항과 비교됩니다.

SNMP(Simple Network Management Protocol)	MDT(Model Driven Telemetry)
비실시간 정보	실시간 정보
확장성이 낮음	뛰어난 확장성
끌어오기 모델	푸시 모델
자동화되지 않음	자동화 준비/데이터 모델 기반

스트리밍 텔레메트리의 장점

스트리밍된 실시간 텔레메트리 데이터는 다음과 같은 경우에 유용합니다.

용량 계획/트래픽 최적화: 네트워크의 대역폭 사용률 및 패킷 삭제를 자주 모니터링하면 링크를 추가 또는 제거하고, 트래픽을 리디렉션하고, 폴리싱을 수정하는 등의 작업을 더 쉽게 수행할 수 있습니다. 빠른 경로 재설정과 같은 기술을 통해 네트워크는 새로운 경로로 전환하고 SNMP 폴링 간격 메커니즘보다 빠르게 경로를 재설정할 수 있습니다. 텔레메트리 데이터를 스트리밍하면 더 빠른 트래픽에 빠르게 대응할 수 있습니다.

더 우수한 가시성: 네트워크에서 문제가 발생한 상황을 신속하게 탐지하고 예방할 수 있도록 지원합니다.

모델 기반 텔레메트리 기술 사양

다음 섹션에서는 MDT라고도 하는 IOS XR 모델 기반 텔레메트리의 기술 기능 및 주요 구성 요소에 대해 다룹니다.

텔레메트리 기능

텔레메트리 프레임워크는 세 개의 서로 다른 상호 연결된 기능 블록으로 구성됩니다.

첫 번째 블록은 **데이터 표현에** 관한 것으로, 이는 분석 또는 측정을 지칭하는 정보가 보드 상에 구성되는 방법이다.

두 번째 블록은 인코딩에 **대한 것입니다**. Telemetry는 샘플 간격마다 위의 측정 데이터를 유선 전체에 걸쳐 직렬화할 수 있는 형식으로 변환합니다. 물론, 다른 쪽 끝의 컨트롤러는 디바이스가 전송하는 원본 데이터의 동일한 사본을 갖기 위해 데이터를 디코딩할 수 있어야 한다.

마지막 블록은 전송에 **대한 것입니다**. 디바이스 간에 데이터를 전송하는 데 사용되는 프로토콜 스택입니다.

다음 표에는 모델 기반 텔레메트리 빌딩 블록의 주요 구조가 요약되어 있습니다.

기능	구성 요소
자료 표현	YANG 데이터 모델
인코딩	GPB/GPB 자체 설명
전송	TCP/gRPC

표 3 텔레메트리 구성 요소

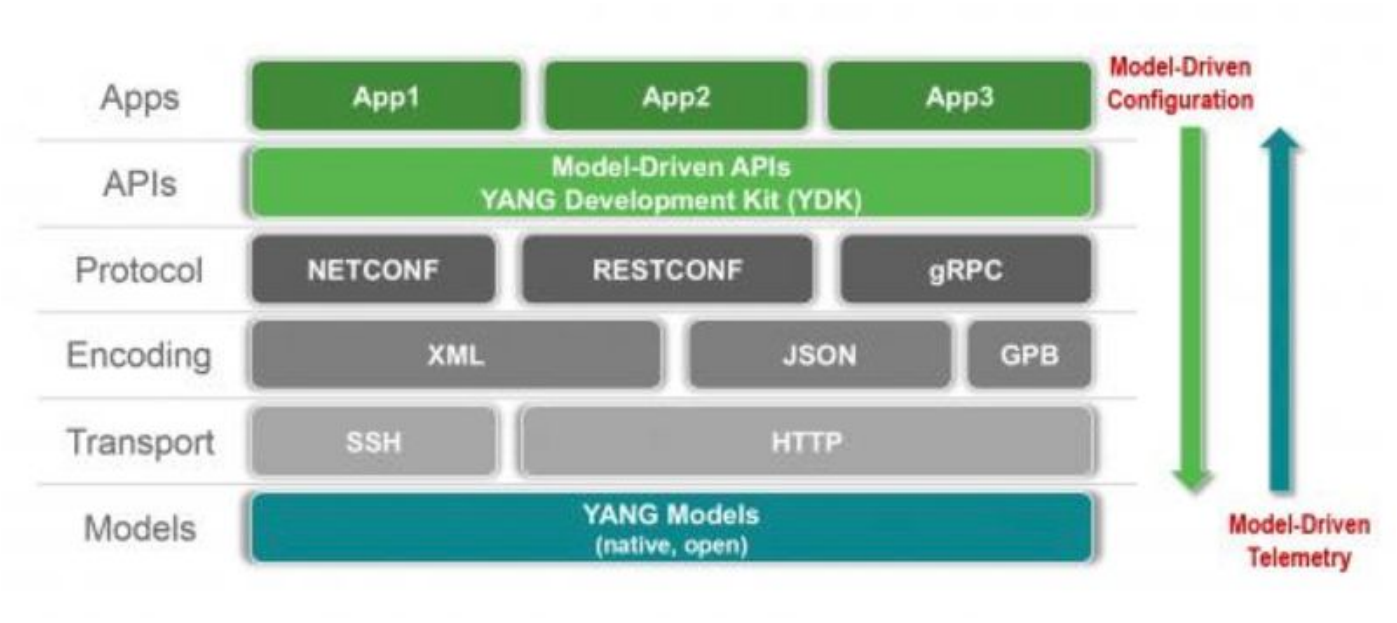
텔레메트리 구성 요소

텔레메트리 및 기본 컨피그레이션 부분이 작동하는 방식을 파악하기 전에, 최적의 설정을 평가하기 위해 텔레메트리의 다양한 구성 요소를 파악하는 것이 중요합니다. 텔레메트리는 새로운 인프라 프레임워크가 네트워크 자동화에 필수적인 기능을 제공하는 IOS XR 프로그래밍 기능 스택에 의존합니다.

YANG은 최근 데이터 모델링의 표준이 되었습니다. 이 표준은 Cisco 프로그래밍 기능 스택에서 네트워크를 통해 가능한 한 빨리 인코딩하고 전달할 수 있는 구조화된 데이터 집합을 구성하는 데 사용됩니다. YANG은 유연성이 뛰어나 자동화 프로세스를 위한 구성 툴로도 활용할 수 있는 큰 장점이 있습니다. 이러한 데이터 모델은 특정 인코딩 형식 및 전송 프로토콜과 결합되어 MDT를 네트워크 분석에 대한 완전한 솔루션으로 만듭니다.

모델 기반 텔레메트리 설정의 경우 수집 및 분석에 필요한 데이터 스트리밍을 활성화하기 위해

YANG 데이터 모델이 중요한 구성 요소가 됩니다.



IOS XR 프로그래밍 기능 스택

양

Yang은 "네트워크 관리 프로토콜에 대한 컨피그레이션 데이터, 상태 데이터 및 알림을 모델링하는 데 사용되는 데이터 모델링 언어"로 정의됩니다. 전형적인 프로그래밍 언어 아키텍처로부터 분리된 특성 때문에 YANG은 매우 다양한 도구와 상호 작용하도록 구현될 수 있습니다.

YANG 모델링 데이터 구조는 구성 작업 및 알림 처리를 비롯한 여러 작업에 사용할 수 있는 트리 형식의 데이터 계층 구조를 정의하는 모듈 및 하위 모듈 개념을 기반으로 구축됩니다.

YANG 모델의 여러 소스가 있으며 이 중 3개 미만이 기본 소스로 간주됩니다.

- 기본 모델/Cisco 관련
- 오픈컨피그레이션
- IETF

Cisco 관련 모델: 이러한 모델은 **네이티브 모델이라고도** 하며 Cisco를 비롯한 다양한 장치 벤더에서 게시합니다. 예: Cisco-IOS-XR-ptp-oper.yang

OpenConfig 모델 OpenConfig는 네트워크 운영자의 비공식 작업 그룹입니다. OpenConfig는 모든 공급업체가 미션 크리티컬 기능 구성을 위해 지원해야 하는 공통 YANG 모델을 정의합니다. 예: openconfig-interfaces.yang

IETF 모델: IETF는 인터페이스, QOS에 대한 기본 구성을 설명하고 기타 일반 데이터 유형(예: Ipv4, IPv6 등)을 정의하는 몇 가지 일반 YANG 모듈도 정의합니다. 예: ietf-syslog-types.yang

Cisco는 사용 가능한 Openconfig 모델을 지원합니다. 공급업체들은 멀티벤더 환경을 지원하기 위해 표준화된 데이터 모델링 방식으로 통합하고 있습니다.

Yang 모델에는 **세** 가지 유형이 있습니다.

1. 운영
2. 설정

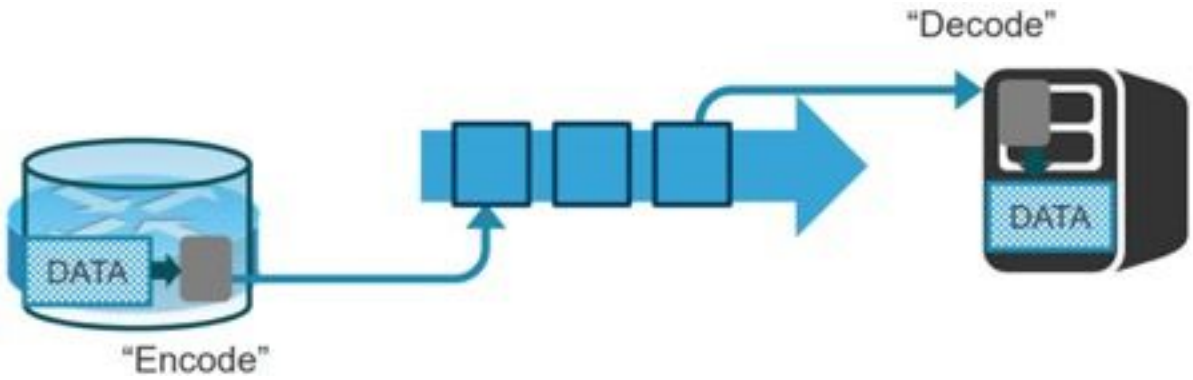
3. 작업

텔레메트리는 ***-oper-*.yang**으로 식별될 수 있는 **Operational Yang 모델만** 고려합니다.

YANG은 RFC 7950에 정의되어 있습니다. https://tools.ietf.org/html/rfc7950.

인코딩

인코딩(또는 "serialization")은 데이터(개체, 상태)를 네트워크 전체에 전송할 수 있는 형식으로 변환합니다. 수신자가 데이터를 디코딩("deserialize")할 때 원본 데이터의 의미론적으로 동일한 복사본이 있습니다.

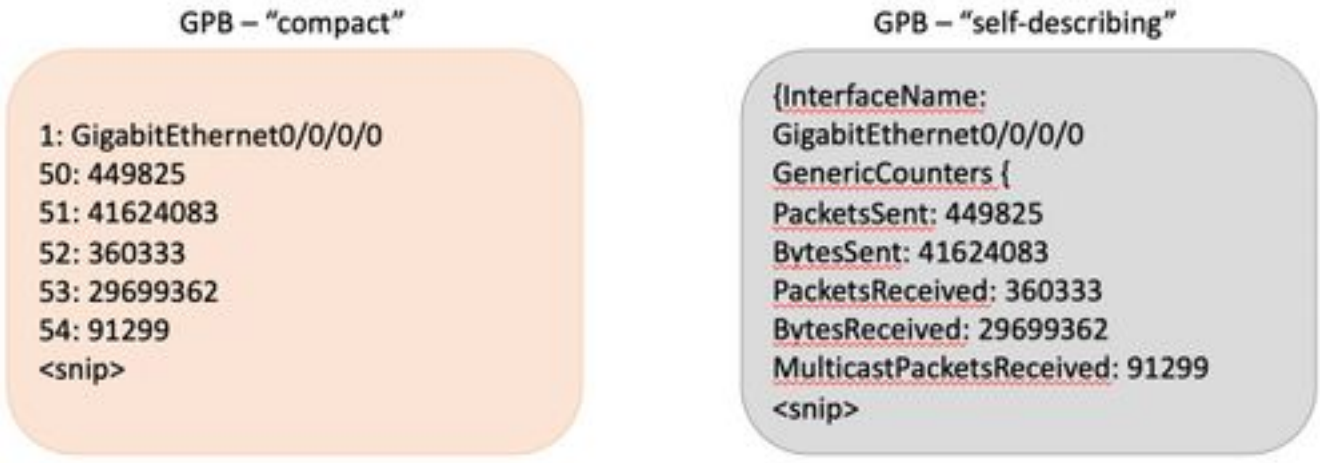


텔레메트리의 초기 개발 단계에서 XML은 태그 기반 구조 때문에 처음에 첫 번째 선택 인코딩 형식으로 간주되었습니다. 그러나 XML의 문제는 압축되지 않은 인코딩 구조였습니다. GPB(Google Protocol Buffers)는 인코딩 작업의 효율성과 속도를 향상하기 때문에 Cisco에서 최종적으로 채택했습니다.

텔레메트리 스트리밍을 위한 인코딩 옵션으로 두 가지 유형의 GPB가 있습니다.

1. 컴팩트 GPB
2. 자체 설명 GPB

두 GPB 텔레메트리 형식 간의 주요 차이점은 데이터의 텔레메트리 스트림 내에서 키를 표시하고 인코딩하는 방법입니다.



JSON은 사람이 쉽게 이해할 수 있고 거의 모든 응용 프로그램에서 디코딩할 수 있는 또 다른 사용자 친화적인 인코딩 스키마입니다.

배포 관점에서 인코딩 스키마의 장단점이 거의 없습니다. 다양한 인코딩 스키마에 대한 비교는

Telemetry Design Guidelines 섹션에서 설명합니다.

전송

텔레메트에서는 전송 프로토콜에 대한 세 가지 선택 사항을 제공합니다.

- TCP
- gRPC
- UDP

또한 텔레메트에서는 노드와 컬렉터 간의 세션을 시작하기 위해 두 가지 시작 모드를 정의합니다.

- 전화 접속
- 전화 접속

두 모드의 차이점은 전송 세션이 설정되는 방식에서만 이루어집니다.

전화 접속 세션 중에 디바이스는 사전 구성된 서버 포트를 향해 syn 패킷을 전송하여 연결을 시작합니다. 연결이 설정되면 데이터 스트림이 디바이스에서 즉시 푸시됩니다.

다이얼인 세션의 경우 라우터는 서버 연결을 기다리는 tcp 포트를 수동적으로 수신 대기합니다.

그러나 세션이 설정되면 라우터가 서버 자체에서 폴링되지 않습니다. 디바이스가 데이터 푸싱 작업을 계속 담당하기 때문입니다. MDT에서는 실제로 데이터 폴링이라는 개념도 존재하지 않습니다.

TCP는 기본적으로 텔레메트리를 위한 사전 정의된 전송 방법입니다. 이는 신뢰할 수 있고 옵션으로 구성하기가 매우 쉽기 때문입니다.

gRPC는 모든 환경에서 실행되도록 설계된 최신 오픈 소스 프레임워크입니다. HTTP/2를 기반으로 구축되며, 향상된 다양한 기능을 제공합니다.

케이던스 기반 텔레메트리 vs 이벤트 기반 텔레메트리

가입된 데이터 집합의 데이터는 구성된 주기 간격으로 또는 이벤트가 발생하는 경우에만 대상으로 스트리밍됩니다. 이 동작은 MDT가 cadence 기반 텔레메트리 또는 이벤트 기반 텔레메트리를 위해 구성되었는지 여부에 따라 달라집니다.

이벤트 기반 텔레메트리의 컨피그레이션은 샘플 간격만 차별화 요소로 하는 cadence 기반 텔레메트리와 유사합니다. 샘플 간격 값을 0으로 구성하면 이벤트 기반 텔레메트리를 위한 서브스크립션이 설정되고, 0이 아닌 값으로 구성하면 cadence 기반 텔레메트리를 위한 서브스크립션이 설정됩니다.

변경 관련 이벤트에는 Event Driven Telemetry를 사용하는 것이 좋습니다.

텔레메트리 설계 지침

설명한 것처럼, 텔레메트리 스택에는 많은 구성 요소가 있으며, 다음은 XR 디바이스에서 텔레메트리를 구현하는 동안 고려해야 할 몇 가지 지침입니다.

인코딩 스키마를 선택하는 방법

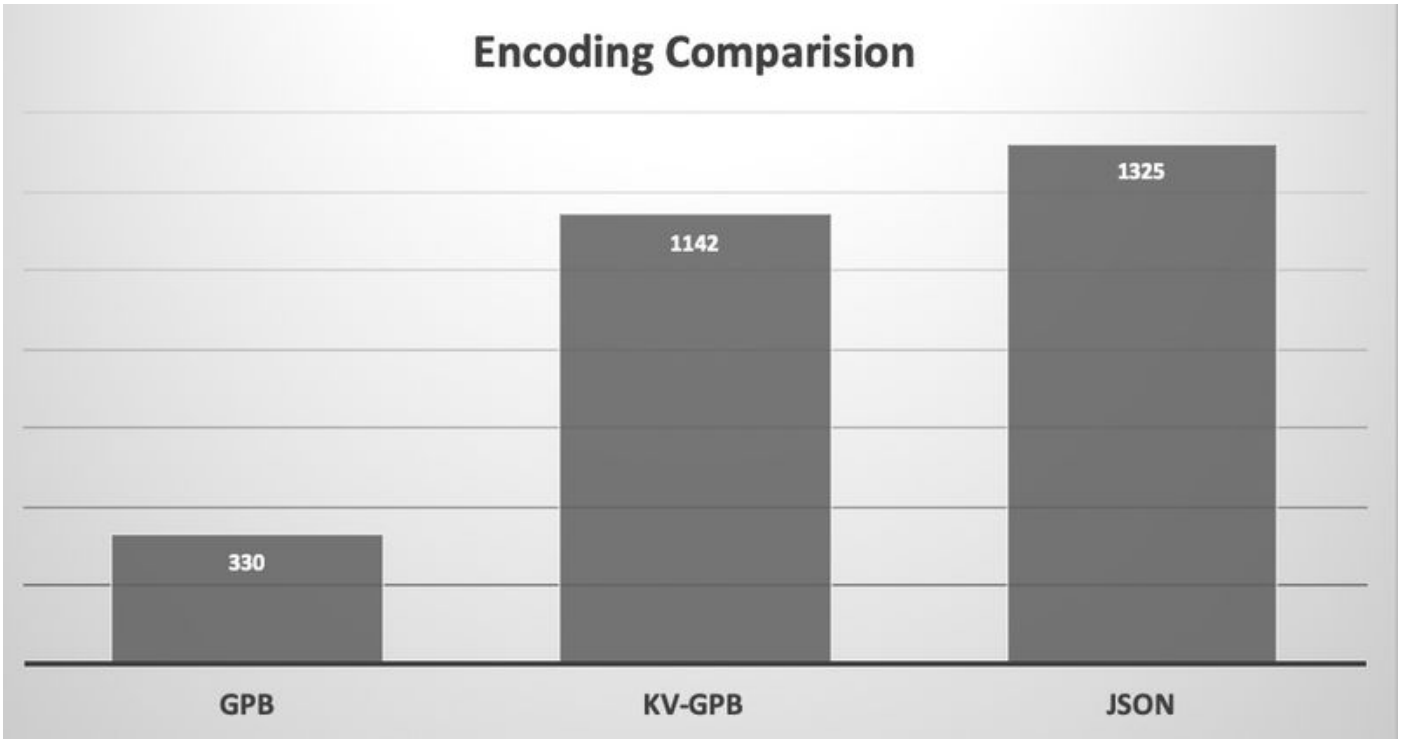
언급된 바와 같이, 인코딩 또는 직렬화는 데이터(객체, 상태)를 네트워크를 통해 전송될 수 있는 포맷으로 변환한다. 수신자가 데이터를 디코딩하거나 역직렬화할 때 원본 데이터의 의미상 동일한 복사본을 가지고 있습니다.

다양한 인코딩 옵션은 와이어 효율성 및 사용 편의성에 따라 달라집니다.

인코딩	간략한 설명	유선 상의 효율성	기타 고려 사항
GPB(컴팩트)	Everything 바이너리 (문자열인 값 제외) 2배 더 빠르고, 운영이 더 복잡함(SNMP와 관련되지는 않음)	높음	모델당 프로토콜 파일
GPB - KV(키-값 쌍)	문자열 키 및 이진 값 (문자열인 값 제외) 3배 더 크고 네이티브 모델: 여전히 키 이름에 대한 휴리스틱이 필요합니다.	중간 ~ 낮음	Deccoding을 위한 단 .proto 파일
JSON	Everything 문자열 : 키 및 값	낮음	친절하군 사람이 읽을 있고 응용 프로그램에 하며 구문 분석이 용이

GPB-KV는 인코딩 스키마에 대해 좋은 균형 중간 포인트를 제공합니다.

선택한 인코딩 스키마에 대한 메시지 길이와 관련하여, 다음은 와이어에 대한 비교입니다.

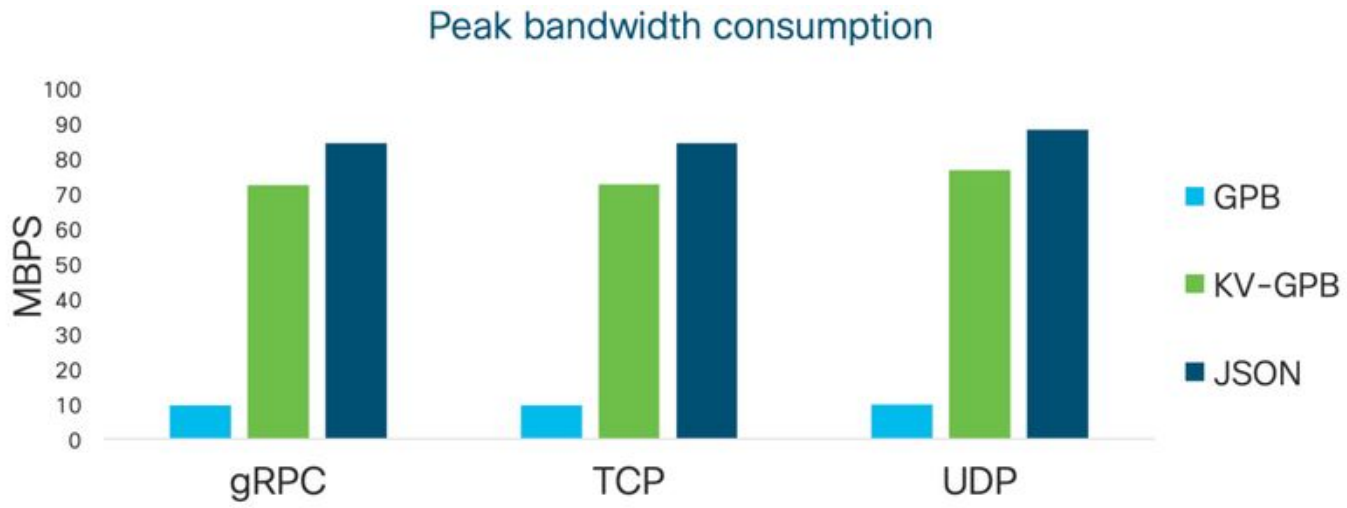


인코딩 비교 - 메시지 길이(바이트)

전송 네트워크 설계 고려 사항

인코딩 옵션이 다르면 대역폭 요구 사항이 달라집니다. 텔레메트리를 고려하는 동안 네트워크 운영

자는 선택한 인코딩 스키마에 따라 충분한 대역폭 프로비저닝을 처리해야 합니다. 인코딩 스키마 비교당 대역폭 소비량 아래에서도 공정한 아이디어를 얻을 수 있습니다.



네트워크 대역폭 비교

Cisco에서는 KV-GPB를 사용하는 것이 좋습니다. 효율성과 편리성 사이의 좋은 중간 지점 역할을 합니다.

텔레메트리 컨피그레이션 옵션 평가

모델 기반 텔레메트리를 구성하는 동안 오퍼레이터는 텔레메트리에 포함된 여러 구성 요소를 모두 파악해야 합니다. 위에서 설명한 전송, 인코딩 및 스트리밍 방향에 사용할 수 있는 옵션을 기반으로 환경에 더 적합한 조합을 선택할 수 있습니다.

이 4가지 주요 구성 요소는

1. 전송
2. 인코딩
3. 세션 방향
4. YANG 데이터 모델

전송: 언급된 대로, 노드는 HTTP/2를 통해 TCP, UDP 또는 gRPC를 사용하여 텔레메트리 데이터를 전달할 수 있습니다.

간소화를 위해 TCP가 선호되는 반면, gRPC는 보안 측면에서 추가적인 이점으로 간주될 수 있는 선택적 TLS 기능을 제공합니다.

인코딩: 라우터는 두 가지 유형의 Google 프로토콜 버퍼로 텔레메트리 데이터를 제공할 수 있습니다. 바로 컴팩트형 GPB와 자체 설명형 GPB입니다.

컴팩트 GPB는 가장 효율적인 인코딩이지만 스트리밍되는 각 YANG 모델에 대해 고유한 .proto가 필요합니다. 자체 설명 GPB는 효율성이 떨어지지만 키가 .proto에서 문자열로 전달되기 때문에 단일 .proto 파일을 사용하여 모든 YANG 모델을 디코딩합니다.

세션 방향: 텔레메트리 구축에서 세션을 시작하는 옵션에는 두 가지가 있습니다. 라우터가 컬렉터로 "전화 접속"하거나 컬렉터가 라우터로 "전화 접속"할 수 있습니다.

YANG은 업계에서 인정받는 데이터 모델링 표준이며 Cisco 프로그래밍 스택에서도 이를 사용하여

네트워크를 통해 가능한 한 빨리 인코딩하고 전달할 수 있는 구조화된 데이터 집합을 구성합니다.

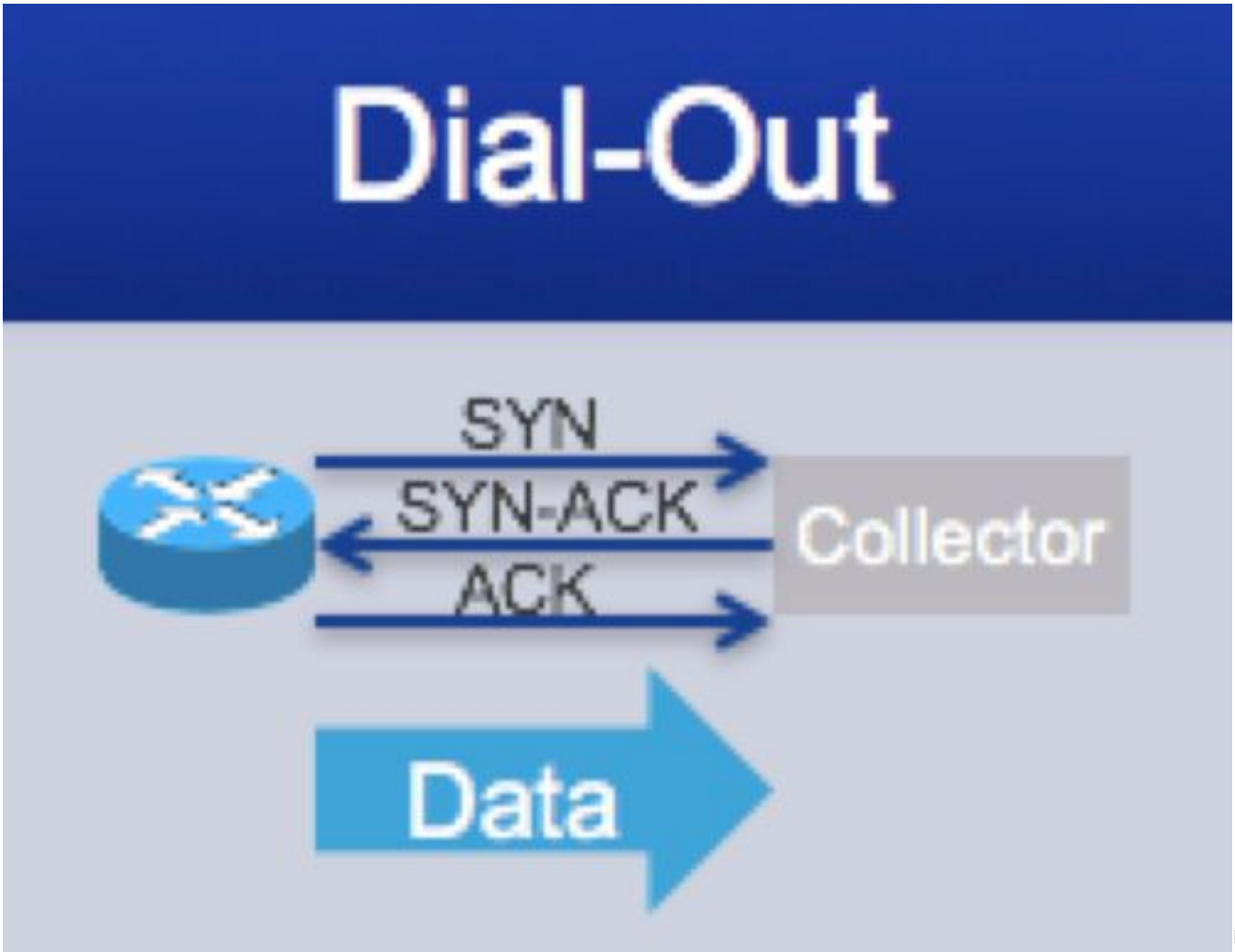
위에서 설명한 특정 인코딩 형식 및 전송 프로토콜과 결합된 이러한 데이터 모델은 MDT(Model Driven Telemetry)를 완전한 분석 솔루션으로 만듭니다.

텔레메트리 컨피그레이션 예

IOS-XR

전화 접속 구성 분리

다이얼 아웃 모드에서 라우터는 컬렉터에 대한 TCP 세션을 시작하고 서브스크립션의 센서 그룹에 의해 지정된 데이터를 전송합니다.



텔레메트리 다이얼 아웃컨피그레이션 관점에서 텔레메트리 컨피그레이션은 3단계 프로세스입니다. 먼저, 스트리밍할 정보를 식별하고 센서 그룹 컨피그레이션에서 캡처합니다. 둘째, 정보가 스트리밍되어야 하는 대상을 식별하고 이를 Destination Group 컨피그레이션에 캡처합니다. 셋째, 이전 두 단계에서 확인된 정보를 사용하여 실제 서브스크립션을 구성합니다.

1. 센서 그룹 정의
2. 대상 그룹 정의
3. 구독 정의

센서 그룹 정의

센서 그룹 컨피그레이션은 스트리밍할 정보를 식별합니다. 다음 컨피그레이션 템플릿은 센서 그룹을 구성하는 데 필요한 컨피그레이션을 제공합니다.

```
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#
```

다음 예에서는 라우터 CLI에서 실제 센서 그룹 컨피그레이션의 예를 보여줍니다.

```
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#
```

동일한 SensorGroup 정의에 여러 센서 경로를 포함할 수 있습니다.

```
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#
```

대상 그룹 정의

Destination 그룹 컨피그레이션은 정보가 스트리밍될 대상을 식별합니다.

여기에는 3가지 주요 매개변수가 있습니다

1. 세션 방향
2. 사용할 인코딩
3. 사용할 전송 프로토콜

다음은 그 예입니다.

```
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#
```

구독 정의

서브스크립션은 센서 그룹 및 대상 그룹 정보를 컨피그레이션의 마지막 부분으로 함께 바인딩합니다. 샘플 간격은 서브스크립션의 일부로 정의됩니다.

다음은 그 예입니다.

```
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#
```

전체 컨피그레이션 예

```
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#
```

다이얼 아웃의 장점

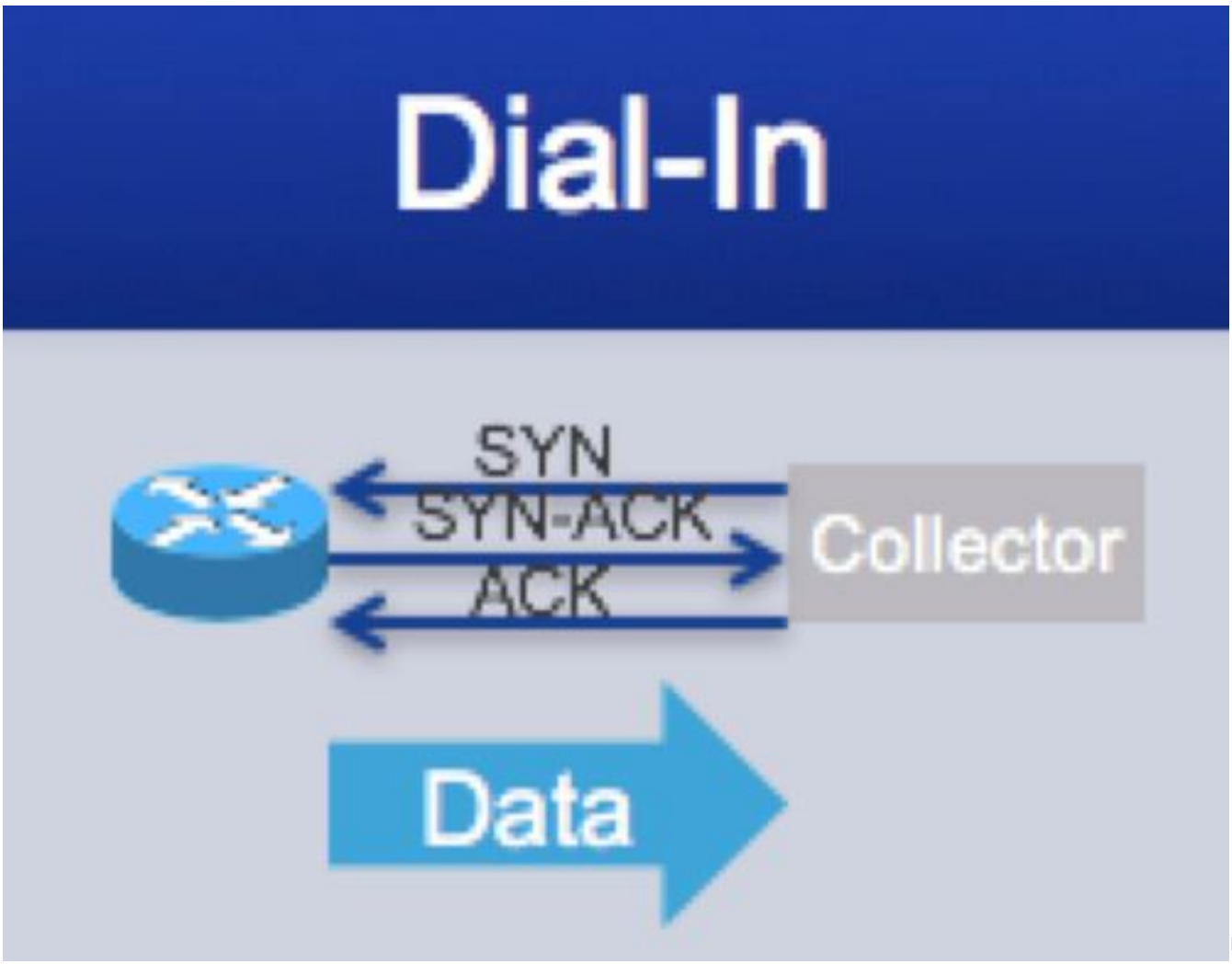
- 전송 옵션에 대한 폭넓은 유연성.
- 인바운드 관리 트래픽을 위해 포트를 열 필요가 없습니다.
- 애니캐스트 및 로드 밸런싱.

다이얼인 컨피그레이션 분리

Dial-In 모드에서는 MDT 수집기/수신기/오케스트레이터가 라우터로 전화를 걸어 하나 이상의 센서 경로 또는 가입에 동적으로 가입합니다. 라우터는 서버 역할을 하고 클라이언트는 수신자 역할을

합니다.

단일 세션만 구성되며 라우터는 동일한 세션을 통해 텔레메트리 데이터를 스트리밍합니다. 수신자가 구독을 취소하거나 세션이 종료될 때 이 동적 구독이 종료됩니다.



텔레메트리 다이얼인

컬렉터가 라우터에 "다이얼인(dial-in)"하므로 컨피그레이션에서 각 MDT 대상을 지정할 필요가 없습니다. 라우터에서 gRPC 서비스를 활성화하고, 클라이언트를 연결하고, 원하는 텔레메트리 서브스크립션을 동적으로 활성화할 수 있습니다.

컨피그레이션 관점에서 텔레메트리 컨피그레이션은 앞서 설명한 것과 유사한 3단계 프로세스입니다. 먼저, gRPC를 활성화해야 합니다. 둘째, 정보를 스트리밍해야 하는 대상을 식별하고 Sensor Group 컨피그레이션에서 이를 캡처합니다. 셋째, 이전 두 단계에서 확인된 정보를 사용하여 실제 서브스크립션을 구성합니다.

1. gRPC 사용
2. 센서 그룹 정의
3. 구독 정의

확장하려면 클릭

전화 접속 모드는 gRPC에서만 지원됩니다.

전화 접속 모드는 gRPC에서만 지원됩니다.

gRPC 사용

먼저, 라우터에서 gRPC 서버를 활성화하여 컬렉터에서 들어오는 연결을 수락해야 합니다.

확장하려면 클릭

- <port-number> 범위는 57344~57999입니다. 포트 번호를 사용할 수 없는 경우 오류가 표시됩니다.

<port-number> 범위는 57344~57999입니다. 포트 번호를 사용할 수 없는 경우 오류가 표시됩니다.

```
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#
```

센서 그룹 정의

센서 그룹 컨피그레이션은 스트리밍할 정보를 식별합니다. 다음 컨피그레이션 템플릿은 센서 그룹을 구성하는 데 필요한 컨피그레이션을 제공합니다.

다음 예에서는 라우터 CLI의 실제 예를 보여줍니다. 여기서 센서 그룹 컨피그레이션의 실제 예는

```
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#
```

구독 정의

서브스크립션은 Sensor-Group 및 gRPC를 컨피그레이션의 최종 부분으로 함께 바인딩합니다. 샘플 간격은 서브스크립션의 일부로 정의됩니다.

다음 컨피그레이션 템플릿은 서브스크립션을 구성하는 데 필요한 컨피그레이션을 제공합니다.

다음 예에서는 서브스크립션을 생성하고 Sensor-Group과 Destination-Group을 함께 바인딩하며 샘플 속도를 정의하는 라우터 CLI의 실제 예를 보여줍니다.

```
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#
```

전체 구성 템플릿 및 예

```
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#
```

전화 접속 기능의 장점

- 컨피그레이션 및 스트리밍을 위한 단일 채널
- 라우터/디바이스의 수신 대기 포트
- 과도 연결
- 현재 gRPC/gNMI만 사용 가능

이벤트 중심 텔레메트리

이벤트 기반 텔레메트에서는 이벤트가 발생하는 경우에만 가입된 데이터 집합의 데이터가 스트리밍됩니다.

이벤트 중심 텔레메트리 컨피그레이션

이벤트 기반 텔레메트리의 컨피그레이션은 cadence 기반 텔레메트리와 유사하며, Event Driven Telemetry의 컨피그레이션에서는 샘플 간격의 컨피그레이션만 다릅니다. 샘플 간격 값을 0으로 구성하면 이벤트 기반 텔레메트리를 위한 서브스크립션이 설정됩니다.

완전한 컨피그레이션 템플릿 및 다이얼 아웃 예

```
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#
```

다음 예에서는 라우터 CLI의 실제 예를 보여줍니다.

```
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#
```

다이얼 인 전체 구성 템플릿 및 예

```
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#
```

다음 예에서는 라우터 CLI의 실제 예를 보여줍니다.

```
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#
```

SHOW 명령으로 원격 분석 유효성 검사

라우터 관점에서 각 센서 그룹, 대상 그룹 및 서브스크립션에 대해 구성된 매개변수를 확인할 수 있습니다

```
// 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#
```

텔레메트리 수집 스택

라우터 컨피그레이션 외에 텔레메트리 기반 솔루션에는 컬렉터, 데이터베이스, 모니터링/분석 소프트웨어 등 여러 구성 요소가 필요했습니다. 이러한 구성 요소는 별도로 구성할 수도 있고 하나의 포괄적인 제품에 포함될 수도 있습니다.

- 인바운드 패킷을 가져와서 추가 저장을 위해 전달하기 위한 디코더를 설치해야 합니다.
- 스트리밍된 정보를 저장하려면 TSDB라고도 하는 시계열 데이터베이스가 필요합니다.
- 내부 데이터베이스에서 가져온 데이터를 시각화하는 그래픽 도구도 필요합니다.

수집 스택을 자세히 설명하는 것은 범위를 벗어납니다. Cisco Crossworks Health Insights를 사용하면 디바이스가 원격 분석 컨피그레이션으로 자동으로 프로비저닝되고 TSDB(Time Series Database)에 테이블/스키마가 생성되는 제로 터치 원격 분석이 가능합니다. 데이터 수집 및 정리에 따른 운영 및 네트워크 관리 오버헤드를 간소화하여 운영자가 비즈니스 목표에 집중할 수 있도록 합니다. SNMP, CLI 및 모델 기반 텔레메트리를 통해 네트워크 디바이스 데이터를 수집하기 위해

공통 컬렉터를 사용하면 데이터 중복을 방지하고 디바이스 및 네트워크에 대한 로드를 줄일 수 있습니다.

네트워크의 텔레메트리를 위한 구축 고려 사항

다음은 네트워크에서 텔레메트리 구축을 분석하는 동안 고려해야 할 다양한 사항입니다.

크기 조정

텔레메트리는 상당한 양의 데이터를 스트리밍할 수 있으며 확장성 측면을 신중하게 고려하는 것이 좋습니다.

필요한 데이터만 스트리밍

각 Yang 모델에는 여러 리프 노드가 있습니다. 필요한 정보와 필요하지 않은 정보를 구체적으로 명시하는 것이 좋습니다. Yang 모델을 탐색하고 텔레메트리 활용 사례에 필요한 데이터 경로를 식별하는 것이 좋습니다.

스트리밍 데이터의 양 고려

스트리밍되는 텔레메트리 데이터의 총량은 다음 사항에 대해 고려해야 합니다.

1. 네트워크 내 대역폭 할당
2. QoS
3. 추가 애플리케이션/소프트웨어 성능 컬렉터 소프트웨어시계열 데이터베이스분석/시각화 소프트웨어
4. 인코딩 효율성("텔레메트리 설계 지침" 섹션에서 설명)은 스트리밍되는 데이터의 양에 직접적인 영향을 미칩니다. 가능한 경우 컴팩트 GPB가 권장됩니다.
5. 수집 간격은 다음을 비롯한 여러 측면에 직접적인 영향을 미칩니다. 네트워크에서의 대역폭 사용률데이터베이스의 저장소 요구 사항데이터를 스트리밍하는 디바이스의 성능

애플리케이션 요구 사항에 따라 수집 빈도를 평가하는 것이 좋습니다.

전반적으로, 실행 가능한 것으로 간주되는 소스 또는 목적지에서 원치 않는 데이터를 필터링하는 것을 고려하는 것이 좋습니다. 원하지 않는 데이터를 필터링할 수 있습니다. 필터링은 두 가지 레벨에서 수행할 수 있습니다. -

1. At the Source(소스) - 데이터를 스트리밍하는 디바이스입니다.
2. At the Destination(대상) - 컬렉터에서 데이터를 수집하고 표준화합니다. (컬렉터에서 필터링하는 것은 이 문서의 범위를 벗어납니다.)

다음 예에서는 와일드카드를 적용하여 센서 경로 내의 Hundered Gig 인터페이스에 대해서만 필터링되는 데이터를 보여 줍니다.

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

참조

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

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.