Network Working Group
Internet-Draft
Intended status: Informational
Expires: July 21, 2017

V. Talwar
J. Kolhe
A. Shaikh
Google
J. George
Cisco

January 17, 2017

Use cases for gRPC in network management
draft-talwar-rtgwg-grpc-use-cases-01

Abstract

gRPC is an open, high-performance RPC framework designed for efficient low-latency cross-service communications. This document describes use cases for gRPC in network management and other services, particularly streaming telemetry.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 21, 2017.

Table of Contents

1. Introduction

gRPC is a high performance universal RPC framework to connect distributed systems [GRPC-WWW]. gRPC emerged from an internal Google framework called Stubby which has been used to connect large numbers microservices running within and across data centers for over a decade. Having a uniform, cross-platform RPC infrastructure allowed Google to deploy fleet-wide improvements in efficiency, security, reliability and behavioral analysis critical to supporting the incredible growth of these services. gRPC is the next generation of Stubby, built in the open originally for services, as well as last mile computing use cases like mobile, browser, IOT [GRPC-DESIGN]. It is based on standards like HTTP/2 [RFC7540] and is extensible and pluggable by design.

This document describes use cases for the gRPC protocol [I-D.kumar-rtgwg-grpc-protocol] in network management, including monitoring, configuration management and programmatic operations. We

also summarize a number of additional use cases where gRPC is currently being applied.

2. gRPC use cases

2.1. Network management

Below we discuss several gRPC applications related to network management with a focus on monitoring and telemetry. gRPC is already implemented by several network device vendors as a primary transport for monitoring data based on the streaming telemetry paradigm.

2.1.1. Streaming telemetry motivation and overview

Network operations depend fundamentally on the availability of accurate, near real-time data to drive a variety of management systems, including traffic control systems, fault recovery systems, and demand and capacity forecasting systems. This data consists of information about the control plane (e.g., protocol operations), management plane (e.g., system availability, statistics, and counters), and data plane (e.g., packet and flow statistics).

In addition to the variety of data, the volume of monitoring and management data continues to increase significantly. Modern, high-density platforms with thousands of interfaces and numerous hardware and software modules means potentially collecting millions of objects and running tens of thousands of CLI commands every few minutes in a large-scale network. Network monitoring data is increasingly used to manage mission-critical systems such as real-time monitoring, centralized traffic engineering, server selection and load balancing. Hence it requires efficient, secure, and scalable mechanisms for data transport, encoding, and control.

Most networks rely on traditional management protocols such as SNMP [RFC1157] [RFC3410] for collecting monitoring data about the control and management planes, and SFLOW [RFC3176] or IPFIX [RFC7011] for the data plane. For control and management data in particular, SNMP is the primary tool, despite limitations which make it ill-suited for modern, large-scale networks, especially Web- and Internet-scale backbones, and large, high-capacity data center networks.

While SNMP is widely deployed and implemented in a variety of network environment, it suffers from a number of drawbacks:

- legacy implementations -- designed for devices with limited memory and little processing power; e.g., SNMPv2 supports multiple data items in a message, but is not optimized for high-volume data collection

- lack of discoverability -- discovering new elements requires walking the SNMP MIB periodically; on high-density platforms this is extremely computationally expensive

- lack of capability advertisements -- each object ID must be checked to know whether it is supported by the target platform

- rigid data structures -- whether using standard or vendor proprietary MIBs, the structure and format of the data cannot be easily extended or augmented

To address these drawbacks, a number of network operators proposed a new approach for network monitoring based on streaming telemetry (see an early proposal in [I-D.swhyte-i2rs-data-collection-system]). Streaming telemetry is based on a pub/sub push model in which target devices send data of interest over a streaming channel to a data collection system.

Some notable features of a streaming telemetry system include:

- targets stream data continuously based on a specified period (or as frequently as the target supports), or on a state change

- data is sent as soon as it is available, reducing the need to buffer, or to handle a single large for all data at once

- data may be sent incrementally, e.g., only for those data items that have changed

- ability to distribute the telemetry sources (e.g., directly to linecards) to avoid burdening the management CPU

- users issue subscription requests via RPC to the target to request only the data of interest

- data is exported in a well-structured, common format, e.g., based on YANG models of operational state data [I-D.openconfig-netmod-opstate]

- the target and collector communicate over a secure, authenticated, reliable channel that is long-lived and efficient

Streaming telemetry allows the network behavior to be observed through a time-series data stream. This is in contrast to the polling mechanism used in SNMP in which a monitoring client must periodically request the set of desired data, and walk the MIB to discover changes. The polling frequency is limited since the device

must be able to handle large requests for all interface or QoS counters, for example.

Open source implementations of streaming telemetry are currently being developed by several network vendors, including adapters to deliver data into time-series databases, messaging systems, and data visualization systems [ST-CISCO] [ST-JUNIPER] [ST-ARISTA].

2.1.2. Streaming telemetry with gRPC

gRPC provides a number of capabilities that makes it well-suited for network telemetry. Since its underlying transport is based on HTTP/2, it can exploit several key features:

- binary framing and header compression -- highly efficient encoding on the wire to enable bulk data transfer
- bidirectional streaming RPCs -- the target and collector can stream their data independently, and leverage application-level flow control
- flexible data encoding -- gRPC is payload agnostic, and can be used to transfer data encoded as XML, JSON, protocol buffer, or Thrift; as new data formats and encodings emerge for network data, the RPC layer can be easily adapted
- multi-language support -- open source gRPC IDLs are available for 10 programming languages, and service endpoints can be created on a number of operating systems, giving device vendors flexibility in implementation

gRPC-based telemetry stacks are now being implemented with some available as open source [ST-ARISTA]. A protocol specification for streaming telemetry based on gRPC is also available [GNMI-SPEC].

2.1.3. Network configuration management

gRPC offers a non-proprietary, modern alternative to vendor-specific configuration protocols or standards such as NETCONF [RFC6241] or TL1 [TL1]. Some of the benefits of using gRPC for configuration management include more flexible data encodings (e.g., no requirement to use XML), easier integration based on the large number of language implementations available, and more options for securing connections.

Several platforms now support gRPC configuration protocols using data based on YANG models [GRPC-CISCO] [GRPC-JUNIPER].

2.2. Additional use cases

2.2.1. Client Libraries for connecting polyglot systems

gRPC generates client libraries in 10 languages and thus allows developers to operate in their language of choice and system to communicate with any other system. These libraries offer idiomatic-to-language API surface such that every developer feels they are in their language native environment.

2.2.2. MicroServices

Designed as a general, high performance protocol to interconnect polyglot systems, gRPC is ideal for microservices communication, independent of where the services are deployed. A protocol that offers flow control, bidirectional streaming and a very compact serialization mechanism is ideally suited for connecting microservices at scale. It is already being adopted by large organizations like Square and Netflix for their microservices communications.

2.2.3. Browser and mobile applications communicating to gRPC Services

Mobile and Browser applications are becoming feature rich and more demanding by the day. User expectation is that apps are performant in various network conditions and drain minimal battery and computing power of device. gRPC provides native iOS and Android Java libraries for more efficient communication for applications with backend services such that battery, data are efficiently used and developers have more control of communication with servers using gRPC APIs.

2.2.4. High performance access to Cloud Services

The expectations from high request-rate cloud services like storage and pub/sub messaging systems are to be very efficient and low cost from a compute and networking point of view. Hence, gRPC based APIs are being used for services like Google Cloud BigTable and Google Cloud PubSub. External products like etcd (underlying storage system for kubernetes) also relies on gRPC.

2.2.5. Secure and low overhead communications in embedded systems

With its integrated authentication model and a IDL like nano-protobuf, gRPC could be ideal for secure device-to-device and device-to-cloud communication as well. This use case is still under development.

2.2.6. Unified inter-process and remote communication

gRPC can provide a unified programming model for both inter-process communication and remote service communication. This use case is still under development.

3. Security Considerations

As applied to network configuration and monitoring, any transport protocol and RPC framework must have support for secure, authenticated communication. gRPC supports a number of security mechanisms that are suitable for use in network management, including TLS-based transport, and client and server authentication. These will be detailed further in subsequent drafts.

4. IANA Considerations

None at this time. In the future, there may be proposals to designate specific application ports for gRPC-based telemetry and configuration traffic.

5. References

5.1. Normative references

[RFC1157] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol (SNMP)", RFC 1157, DOI 10.17487/RFC1157, May 1990, <http://www.rfc-editor.org/info/rfc1157>.

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, DOI 10.17487/RFC3410, December 2002, <http://www.rfc-editor.org/info/rfc3410>.

[RFC3176] Phaal, P., Panchen, S., and N. McKee, "InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks", RFC 3176, DOI 10.17487/RFC3176, September 2001, <http://www.rfc-editor.org/info/rfc3176>.

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <http://www.rfc-editor.org/info/rfc7011>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <http://www.rfc-editor.org/info/rfc7540>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

5.2. Informative references

[GRPC-DESIGN]
Ryan, L., "gRPC Motivation and Design Principles", September 2015, <http://www.grpc.io/posts/principles>.

[GRPC-WWW]
"gRPC Web site (grpc.io)", September 2015, <http://www.grpc.io>.

[ST-ARISTA]
"Arista Networks goarista GitHub repository", July 2016, <https://github.com/aristanetworks/goarista>.

[ST-CISCO]
"Cisco Systems bigmuddy GitHub repository", April 2016, <https://github.com/cisco/bigmuddy-network-telemetry-stacks>.

[GRPC-CISCO]
"Cisco Systems grpc GitHub repository", February 2016, <https://github.com/CiscoDevNet/grpc-getting-started>.

[ST-JUNIPER]
"Juniper Networks open-nti GitHub repository", July 2016, <https://github.com/Juniper/open-nti>.

[GRPC-JUNIPER]
Juniper Networks, "Next-Generation Network Configuration and Management", May 2016, <https://www.juniper.net/assets/us/en/local/pdf/whitepapers/2000632-en.pdf>.

[GNMI-SPEC]
Borman, P., Hines, M., Lebsack, C., Morrow, C., Shaikh, A., and R. Shakir, "gRPC Network Management Interface", November 2016, <https://github.com/openconfig/reference/blob/master/rpc/gnmi/gnmi-specification.md>.

[TL1] Telcordia, "GR-831-CORE - Operations Application Messages - Language for Operations Application Messages", November 1996, <http://telecom-info.telcordia.com/site-cgi/ido/docs.cgi?ID=SEARCH&DOCUMENT=GR-831&>.

[I-D.kumar-rtgwg-grpc-protocol]
Kumar, A., Kolhe, J., Ghemawat, S., and L. Ryan, "gRPC Protocol", kumar-rtgwg-grpc-protocol-00 (work in progress), July 2016.

[I-D.swhyte-i2rs-data-collection-system]
Whyte, S., Hines, M., and W. Kumari, "Bulk Network Data Collection System", draft-swhyte-i2rs-data-collection-system-00 (work in progress), October 2013.

[I-D.openconfig-netmod-opstate]
Shakir, R., Shaikh, A., and M. Hines, "Consistent Modeling of Operational State Data in YANG", draft-openconfig-netmod-opstate-01 (work in progress), July 2015.

Appendix A. Change summary

A.1. Changes between revisions -00 and -01

- Added reference to gRPC Network Management Interface specification.
- Updated author contact information.

Authors' Addresses

Varun Talwar
Google
1600 Amphitheatre Pkwy
Mountain View, CA 94043
US

Email: varuntalwar@google.com

Jayant Kolhe
Google
1600 Amphitheatre Pkwy
Mountain View, CA 94043
US

Email: jkolhe@google.com

Anees Shaikh
Google
1600 Amphitheatre Pkwy
Mountain View, CA 94043
US

Email: aashaikh@google.com

Joshua George
Cisco
170 W Tasman Dr
San Jose, CA 95134
US

Email: jgeorge@cisco.com