Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 19, 2020

A. Przygienda
Juniper
A. Lingala
AT&T
C. Mate
NIIF/Hungarnet
J. Tantsura
Nuage Networks
August 18, 2019

Compressed BGP Update Message
draft-przygienda-idr-compressed-updates-07

Abstract

This document provides specification of an optional compressed BGP update message format to allow family independent reduction in BGP control traffic volume.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 https://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 February 19, 2020.

Table of Contents

1. Introduction

BGP as a protocol evolved over the years to carry larger and larger volumes of information and this trend seems to continue unabated. And while lots of the growth can be contributed to the advent of new address families spurred by [RFC2283], steady increase in attributes and their size amplifies this tendency. Recently, even the same NLRI may be advertised multiple times by the means of ADD-PATH [RFC7911] extensions. All those developments drive up the volume of information BGP needs to exchange to synchronize RIBs of the peers.

Although BGP update format provides a simple "semantic" compression mechanism that avoids the repetition of attributes if multiple NLRIs share them already, in practical terms, the packing of updates has proven a difficult challenge. The packing attempts are further undermined by the plethora of "per NLRI-tagging" attributes such as extended communities [RFC4360].

One could of course dismiss the growing, raw volume of the data necessary to exchange BGP information between two peers as a mere trifle given the still rising link bandwidths, alas we are facing other sustained trends that would make the reduction of data volume exchanged by BGP highly desirable:

- Link delays will remain constant until radically new transmission mechanisms become common place [QUANT]. Bare those developments, and given the prevailing constant ethernet MTU, increasing volume of BGP traffic will cause more and more IP packets to be sent with the BGP synchronization speed being limited by the expanding bandwith-delay product.

- The data volume, which for one peer may be reasonable, becomes less so when many of those need to be refreshed due to [RFC4724] and [RFC7313] interactions. Use of those techniques is expected to increase due to increasing demands on BGP reliability and novel variants of state synchronization between peers.

- BGP message length is limited to 4K which in itself is a recognized problem. Extensions to the message length [ID.draft-ietf-idr-bgp-extended-messages-21] are being worked on but this puts its own requirements and memory pressure on the implementations and ultimately will not help with attributes exceeding 4K size limit in mixed environments.

- Virtualization techniques introduce an increasing amount of context switches an IP packet has to cross between two BGP instances. Coupled with difficulties in estimating a reasonable TCP MSS in virtualized environments and the number of IP packets TCP generates, more and more context switching overhead per update is necessary before good-put BGP processing can happen.

Obviously, unless we change BGP encoding drastically by e.g. introducing more context to allow for semantic compression, we cannot expect a reduction in data volume without paying some kind of price. Ideas such as changing BGP format to allow for decoupling of attribute value updates from the NLRI updates could be a viable course of action. The challenges of such a scheme are significant and since such "compression" would extend the semantics and formats of the updates as we have them today, former and future drafts may interact with such an approach in ways not discernible today. Last but not least, attempting to introduce a smarter, context-rich encoding is likely to cause dependency problems and slow-down in BGP encoding procedures.

Fortunately, some observations can be made and emerging trends exploited to attempt a reduction in BGP data volumes without the mentioned disadvantages:

- BGP updates are very repetitive. Smallest change in attribute values causes extensive repetition of all attributes and any difference prevents packing of NLRIs in same update. On top, each update message BGP still carries a marker that largely lost its practical value some time ago. One could generalize those facts by saying that BGP updates tend to exhibit very low entropy.

- CPU cycles available to run control protocols are getting more and more abundant as does to a certain extent memory. They tend to not be available anymore in easily harvested "single core with higher frequency" form factors but as multiple cores that introduce the usual pitfalls of parallelization. In short, getting a lot of independent work done is getting cheaper and cheaper while speeding up a single strain of execution depending on previous results less so. This opens nevertheless the possibility to apply different filters on BGP streams, possibly even executing in parallel threads. One possible filter can compress the data in a manner completely transparent to the rest of existing implementation.

Hence, we suggest in this document the removal of redundancy in the BGP update stream via Huffman codes which can be applied as filter to a BGP update stream concurrently to the rest of the BGP processing and per peer. Subsequently, this document describes an optional scheme to compress BGP update traffic with a deflate variant of Huffman encoding [RFC1950], [RFC1951].

In broadest terms, such a scheme will be beneficial if a BGP implementation finds itself in an I/O constrained scenario while having spare CPU cycles disponible. Compression will ease the pressure on TCP processing and synchronization as well as reduce raw number of IP packets exchanged between peers.

2. Terminology

3. IANA Considerations

This document will request IANA to assign new BGP message type value and and a new optional capability value in the BGP Capability Codes registry. The suggested value for the Compressed Updates message type in this process will be 7 and for the Capability Code the suggested value will be 76.

IANA will be requested as well to assign a new subcode in the "BGP Cease NOTIFICATION message subcodes" registry. The suggested name for the code point will be "Decompression Error". The suggested value will be 10.

4. Procedures

4.1. Decompression Capability Negotiation

The capability to *decompress* a new, optional message type carrying compressed updates is advertised via the usual BGP optional capability negotiation technique.

A peer MUST NOT send any compressed updates towards peers that did not advertise the capability to decompress. A peer MAY send compressed updates towards peers that advertised such capability.

4.2. Compressed BGP Update Messages

A new BGP message is introduced under the name of "Compressed BGP Update". It contains inside arbitrary number of following message types

- normal BGP updates
- Enhanced Route Refresh [RFC7313] subtype 1 and 2 (BoRR and EoRR)
- Route Refresh with Options [ID.draft-idr-bgp-route-refresh-options-03] subtype 4 and 5 (BoRR and EoRR with options)

following each other and compressed while following the rules below:

1. Compressed and uncompressed BGP updates MAY follow each other in arbitrary order with exception of compressor overflow scenario per Section 4.3.
2. After decompression of the stream of interleaved compressed and uncompressed BGP update messages the resulting uncompressed sequence does not have to be identical to the sequence in a stream that would be generated without compression. However, the processing of the uncompressed sequence MUST ensure that the ultimate semantics of the message stream is the same to the peer as of a correct uncompressed case.
3. The sender is explicitly permitted to generate outgoing updates in a manner that reorders them as compared to uncompressed stream, but if it does so it MUST ensure that the resulting

stream of updates retains the original semantics as if compression was not in use.

4. The updates and refreshes contained within the compressed BGP update message MUST be stripped of the initial marker while preserving the BGP update or route refresh message header. The length field in the BGP header retains its original value.

5. Each compressed BGP Update MUST carry a sequence of non-fragmented original messages, i.e. it cannot e.g. contain a part of an original BGP update. Section 4.3 presents the only exception to this rule.

6. Each compressed BGP Update MUST be sent as a block, i.e. the decompression MUST be able to yield decompressed results of the update without waiting for further compressed updates. This is different from the normally used stream compression mode. Section 4.3 presents the only exception to this rule.

7. The compressed update message MAY exceed the maximum message size but in such case compressor overflow per Section 4.3 MUST be invoked.

4.3. Compressor Overflow

To achieve optimal compression rates it is desirable to provide to the compressor enough data so the resulting compressed update is as close to the maximum BGP update size as possible. Unfortunately, a Huffman with adapting dictionary compresses at always varying ratio which can lead to an overflow unless it is used very conservatively. A special provision, optionally to be used at the sender's discretion, allows for such overruns and simplifies the handling of overflow events.

In case the compressed block size exceeds the maximum BGP update size, the compressing peer MUST set the according bit in the compressed update generated and MUST proceed it with one and only one compressed update with the overflow and compressor restart bit cleared and the remainder of the block. No other BGP update messages are allowed in the TCP stream between the compressed update of a certain compressor and its overflow fragment. In case of any deviations, the error procedures of Section 4.5 MUST be followed.

The receiving peer MUST concancenate the first compressed update and the following overflow update as a single compressed block and apply decompression to it.

The first update MAY be smaller than the maximum BGP update size.

4.4. Compressor Restarts

In certain scenarios it is beneficial for the compressing peer to be able to restart any of the compressors at any point in the ongoing BGP session. To indicate such an occurrence, each compressed update CAN carry a flag signaling to the decompressing peer that it MUST restart the given de-compressor before attempting to handle the update.

4.5. Error Handling

If the decompression fails for any reason, the failure MUST cause immediate CEASE notification with a newly introduced subcode of "Decompression Error" (as documented in the IANA BGP Error Codes registry). The peer which experienced the failure MAY initiate the connection again but it SHOULD NOT advertise the decompressor capability until an administrative reset of the session or re-configuration of the peer. This will achieve self-stabilization of the feature in case of implementation problems.

The compressing peer MAY send such CEASE notification as well and close the peer. It is at the discretion of the decompressing peer given such a notification to omit the decompression capability on the next OPEN.

5. Special Considerations

5.1. Impact on Network Sniffing Tools

Network sniffing tool today have the capability to monitor an ongoing BGP session and try to reconstruct the state of the peers from the updates parsed. Obviously, with compression enabled, such a monitor cannot follow the compressed updates unless the session is monitored from the first compressed update on.

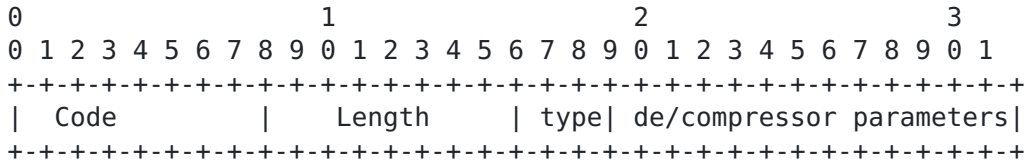
Several possibilities to deal with the problem exist, the simplest one being the restart of the compressors on a periodic basis to allow the monitoring tool to 'sync up'. It goes without saying that this will be detrimental to the compression ratio achieved.

Another possibility would have been to periodically send the Huffman dictionary over the wire but this complexity has been left out as to not overburden this specification. Moreover, at the current time, such a capability is not part of any standard Huffman implementation that could be easily referred to.

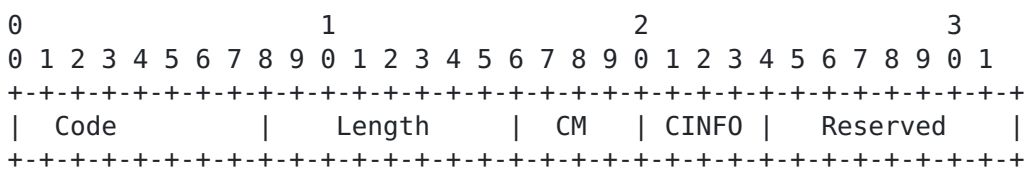
6. Packet Formats

6.1. Decompressor Capability

Decompressor Capability is following the normal procedures of [RFC5492]. In its generic form the option can support different compressors in the future.



This document specifies only DEFLATE Huffman support per [RFC1950].



Code: To be obtained by early allocation, suggested value in this process will be 76.

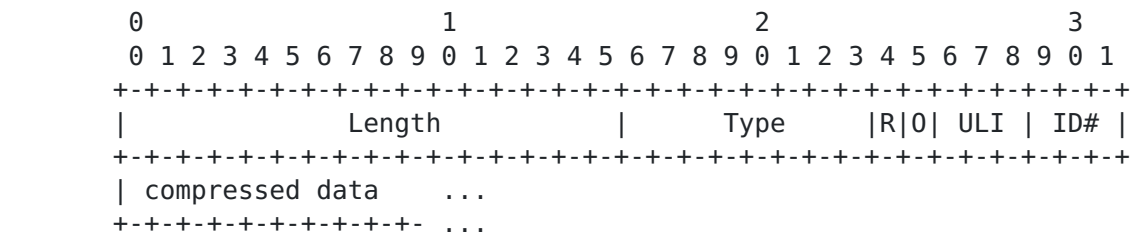
Length: 1 octet.

CM: 4 bits of CM indicating DEFLATE compressed format value as specified in [RFC1950].

CINFO: 4 bits of CINFO as specified in [RFC1950]. Invalid values MUST lead to the capability being ignored. The compressing peer MUST use this value for the parametrization of its algorithm.

6.2. Compressed Update Messages

This carries the original updates in a single message with content adhering to Section 4.2.



Type: To be obtained by early allocation, suggested value in this process will be 7.

Length: 2 octets.

ID#: 3 bits. Indicates the number of the compressor used. Up to 8 compressors MAY be used by the compressing peer to allow for multiple thread of execution to compress the BGP update stream. Accordingly the decompressing side MUST support up to 8 independent decompressors.

R: If the bit is set, the according de-compressor MUST be initialized before the following compressed data is decompressed per Section 4.4. The bit MAY be set on first compressed update sent for the compressor on the session or is otherwise implied sapienti sat. The bit MUST NOT be set on the overflow fragment in case of overflow.

O: If the bit is set, procedures in Section 4.3 MUST be applied. If both the R-bit and the O-bit are set, the de-compressor must be re-initialized before the update and its overflow is assembled and decompression attempted.

ULI: Original uncompressed length indication as to be interpreted as 2**(11+ULI). This MUST indicate a buffer large enough the decompressed data (including overflow) will fit in. The indication MAY be ignored by the receiver but should allow for efficient buffer allocation. The field MUST be ignored on overflow fragment.

7. Security Considerations

This document introduces no new security concerns to BGP or other specifications referenced in this document.

8. Acknowledgements

Thanks to John Scudder for some bar discussions that primed the creative process. Thanks to Eric Rosen, Jeff Haas and Acee Lindem for their careful reviews. Thanks to David Lamperter for discussions on reordering issues.

9. Normative References

[ID.draft-idr-bgp-route-refresh-options-03]
Patel et al., K., "Extension to BGP's Route Refresh Message", internet-draft draft-idr-bgp-route-refresh-options-03.txt, May 2017.

[ID.draft-ietf-idr-bgp-extended-messages-21]
Bush et al., R., "Extended Message support for BGP", internet-draft draft-ietf-idr-bgp-extended-messages-21.txt, May 2016.

[QUANT] Zyga, L., "Worldwide Quantum Web May Be Possible with Help from Graphs", New Journal on Physics , June 2016.

[RFC1950] Deutsch, P. and J-L. Gailly, "ZLIB Compressed Data Format Specification version 3.3", RFC 1950, DOI 10.17487/RFC1950, May 1996, <https://www.rfc-editor.org/info/rfc1950>.

[RFC1951] Deutsch, P., "DEFLATE Compressed Data Format Specification version 1.3", RFC 1951, DOI 10.17487/RFC1951, May 1996, <https://www.rfc-editor.org/info/rfc1951>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2283] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 2283, DOI 10.17487/RFC2283, February 1998, <https://www.rfc-editor.org/info/rfc2283>.

[RFC4360] Sangli, S., Tappan, D., and Y. Rekhter, "BGP Extended Communities Attribute", RFC 4360, DOI 10.17487/RFC4360, February 2006, <https://www.rfc-editor.org/info/rfc4360>.

[RFC4724] Sangli, S., Chen, E., Fernando, R., Scudder, J., and Y. Rekhter, "Graceful Restart Mechanism for BGP", RFC 4724, DOI 10.17487/RFC4724, January 2007, <https://www.rfc-editor.org/info/rfc4724>.

[RFC5492] Scudder, J. and R. Chandra, "Capabilities Advertisement with BGP-4", RFC 5492, DOI 10.17487/RFC5492, February 2009, <https://www.rfc-editor.org/info/rfc5492>.

[RFC7313] Patel, K., Chen, E., and B. Venkatachalapathy, "Enhanced Route Refresh Capability for BGP-4", RFC 7313, DOI 10.17487/RFC7313, July 2014, <https://www.rfc-editor.org/info/rfc7313>.

[RFC7911] Walton, D., Retana, A., Chen, E., and J. Scudder, "Advertisement of Multiple Paths in BGP", RFC 7911, DOI 10.17487/RFC7911, July 2016, <https://www.rfc-editor.org/info/rfc7911>.

Authors' Addresses

Tony Przygienda
Juniper
1137 Innovation Way
Sunnyvale, CA
USA

Email: prz@juniper.net

Avinash Lingala
AT&T
200 S Laurel Ave
Middletown, NJ
USA

Email: ar977m@att.com

Csaba Mate
NIIF/Hungarnet
18-22 Victor Hugo
Budapest 1132
Hungary

Email: matecs@niif.hu

Jeff Tantsura
Nuage Networks
755 Ravendale Drive
Mountain View, CA 94043
USA

Email: jefftant.ietf@gmail.com