Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: December 15, 2019

A. Khare, Ed.
J. Scudder
Juniper Networks, Inc.
L. Jalil
M. Gallagher
Verizon
K. Kasavchenko
NetScout

June 13, 2019

BGP FlowSpec Payload Matching
draft-khare-idr-bgp-flowspec-payload-match-04

Abstract

The rise in frequency, volume, and pernicious effects of DDoS attacks has elevated them from fare for the specialist to generalist press. Numerous reports detail the taxonomy of DDoS types, the varying motivations of their attackers, as well as the resulting business and reputation loss of their targets.

BGP FlowSpec (RFC 5575, "Dissemination of Flow Specification Rules") can be used to rapidly disseminate filters that thwart attacks, being particularly effective against the volumetric type. Operators can use existing FlowSpec components to match on pre-defined packet header fields. However recent enhancements to forwarding plane filter implementations allow matches at arbitary locations within the packet header and, to some extent, the payload. This capability can be used to detect highly amplified attacks whose attack signature remains relatively constant while values in the packet header vary, as well as the burgeoning variety of tunneled traffic.

We define a new FlowSpec component, "Flexible Match Conditions", with similar matching semantics to those of existing components. This component will allow the operator to define bounded match conditions using offsets and a variety of match types.

Copyright Notice

Table of Contents

1. Introduction

BGP FlowSpec [RFC5575] can be used to rapidly disseminate filters that thwart attacks, being particularly effective against the volumetric type. Operators can use existing FlowSpec components to match on pre-defined packet header fields. However recent enhancements to forwarding plane filter implementations allow matches at arbitary locations within the packet header and, to some extent, the payload. This capability can be used to detect highly amplified attacks whose attack signature remains relatively constant, while values in packet header vary. Varying values in packet headers generally make it challenging to mitigate such attacks.

We define a new FlowSpec component, "Flexible Match Conditions", with similar matching semantics to those of existing components. This component will allow the operator to define bounded match conditions using offsets and a variety of match types.

1.1. 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].

2. Motivation

BGP FlowSpec couples both the advertisement of NLRI-specific match conditions, as well as the forwarding instance to which the filter is attached. This makes sense since BGP FlowSpec advertisements are most commonly generated, or at least verified, by human operators. The operator finds it intuitive to configure match conditions as human-readable values, native to each address family.

It is much friendlier, for instance, to define a filter that matches a source address of 192.168.1.1/32, than it is to work with the equivalent binary representation of that IPv4 address. Further, it is easier to use field names such as 'IPv4 source address' as part of the match condition, than it is to demarc that field using byte offsets.

However, there are a number of use cases that benefit from the latter, more machine-readable approach.

2.1. Machine analysis of DDoS attacks

Launching a DDoS is easier and more cost-effective than ever. The will to attack matters more than wherewithal. Those with the inclination can initiate one from the comfort of their homes [1], or even buy DDoS-as-a-Service [2], complete with 24x7 support and flexible payment plans.

Despite their effectiveness, such attacks are easily thwarted - once identified. The challenge lies in fishing out a generally unvarying attack signature from a data stream. Machine analysis may prove superior here, given the size of input involved. The resulting pattern may not lie within a well-defined field; even if it happens to, it may be a more straight-forward workflow to have machine analysis result in a machine-readable filter.

Below we illustrate the need for the suggested approach with two use cases.

2.1.1. Matching based on payload

A vast majority of volumetric DDoS attacks are of reflection/ amplification nature. They can often be identified by the UDP source port of a service that reflects and amplifies the attack traffic. However, there exist DDoS attack methodologies such as SSDP Diffraction or Bittorent amplification where values in most of layer 3 and layer 4 header fields, including source and destination UDP ports, are varied. That makes it challenging if not impossible to classify and mitigate a DDoS attack based on existing Flow

Specification components. At the same time these attacks very often have a constant pattern in payload. Using the pattern in payload as a matching criteria would help in mitigating such DDoS attacks.

2.1.2. Matching based on any protocol header field or across fields

BGP FlowSpec [RFC5575] defines 12 Flow Specification component types that can be used to match traffic. However, a DDoS attack might result in illegitimate traffic of a specific pattern in a layer 3 or layer 4 header, and this pattern would not have a respective component type. Examples are Time to Live field of IP header or Window field of TCP header. In order to avoid extending BGP FlowSpec [RFC5575] with all theoretically possible component types, this document proposes divorcing the search space from having to align with header fields. This allows flexibly matching patterns regardless of whether they have a currently matching component type as well as patterns that span fields.

2.2. Tunneled traffic

Tunnels continue to proliferate due to the benefits they provide. They can help reduce state in the underlay network. Tunnels allow bypassing routing decisions of the transit network. Traffic that is tunneled is often done so to obscure or secure. Common tunnel types include IPsec [RFC4301], Generic Routing Encapsulation (GRE) [RFC2890], Virtual eXtensible Local Area Network (VXLAN) [RFC7348], GPRS Tunneling Protocol (GTP) [GTPv1-U], et al.

By definition, transit nodes that are not the endpoints of the tunnel hold no attendant control or management plane state. These very qualities make it challenging to filter tunneled traffic at non-endpoints. Often though, the forwarding hardware at these transit-only nodes is capable of reading the byte stream that comprises the protocol being tunneled. Despite this capability, it is usually infeasible to filter based on the content of this passenger protocol's header since BGP FlowSpec does not provide the operator a way to address arbitrary locations within a packet.

2.3. Non-IP traffic

Not all traffic is forwarded as IP packets. Layer 2 services abound, including flavors of BGP-signaled Ethernet VPNs such as BGP-EVPN, BGP-VPLS, FEC 129 VPWS (LDP-signaled VPWS with BGP Auto-Discovery).

Ongoing efforts such as [I-D.ietf-idr-flowspec-l2vpn] offer one approach, which is to add layer 2 fields as additional match conditions. This may suffice if a filter needs to be applied only to layer 2, or only to layer 3 header fields.

3. Procedures

The flexible match operator requires three inputs:

- Where the match should begin: Where in the datagram or packet the search for matching values is initiated. This allows skipping over parts of the packet that are not of interest.

- Where the match should end: Where in the datagram or packet the search ends.

- What should be matched: A variety of search types, including exact numeric matches, matching a range of numeric values, and string-based matches.

3.1. Match Location

While intuitive to grasp, determining the match location requires explication. A canonical forwarding engine parses an incoming packet header and identifies it as belonging to a single Network Layer Reachability Information (NLRI), or address family. The contents of the header are parsed with address family specificity, in order to extract a forwarding lookup key. In the case of IPv4 unicast forwarding, this key is the IPv4 destination address. The key is used to look up the corresponding action in an address family specific forwarding table.

This does not preclude implementations from exposing additional packet headers to the operator, both encapsulating and encapsulated, to provide additional forwarding functionality. For instance, common

stateless load balancing techniques involve reading fields in additional headers in order to increase entropy and preserve flow ordering. As another example, in the case of Ethernet encapsulated IPv4 packets, a forwarding engine could allow filtering using the source or destination MAC address even though the forwarding decision is ultimately based only on the IPv4 header.

As yet another example, consider that a Virtual eXtensible Local Area Network (VXLAN) [RFC7348] packet has the following headers:

- Outer Ethernet Header: Source MAC address of the originating VXLAN Tunnel End Point (VTEP). - Outer IPv4/IPv6 Header: Source IP address of the originating VXLAN Tunnel End Point (VTEP).

- Outer UDP Header: Random source port used to generate entropy for load balancing, and destined to the IANA-assigned VXLAN port 4789.

- VXLAN Header: Used to identify a specific VXLAN overlay network.

- Inner Ethernet Header and payload: Original MAC frame encapsulated.

Forwarding at the tunnel midpoints, i.e., not the where tunnel imposition or disposition occur, makes use of the outer IPv4 header. In order to differentiate itself, a midpoint may provide the ability to parse and take the VXLAN header into account. This functionality could be used to implement access control or perform traffic telemetry.

In order to normalize behavior across forwarding implementations, the match location MUST be aligned with the FlowSpec AFI/SAFI to which the flexible match rule belongs. For any such AFI/SAFI, there will be an associated well-known header. The match offset is computed

from either the beginning, or the end, of that header, determined by the "a" flag. For instance, with FlowSpec for IPv4 traffic, the match offset can only start at the beginning of the IPv4 header, or data immediately following that header. Even if the forwarding implementation has the capability to read outer and inner headers, the match location is anchored at the IPv4 header.

3.2. How Much Data to Match

Similarly, the amount of data to match MUST be the lesser of the amount of data specified by the match operator itself, the last data in a packet or the Maximum Readable Length (MRL) that a forwarding implementation can parse from a packet and make available for filtering. As the MRL will be implementation-dependent, it needs to be known to the flexible filtering rules engine. That can be communicated out-of-band via configuration or signaled using future BGP or IGP extensions.

It is not required that all nodes in a flexible filtering domain be required to have a common or minimum MRL. This does not obviate the need for a rules engine to take MRL into account when creating flexible filters. This is especially important as the rules engine may not have direct BGP peering with all FlowSpec enforcers and may not receive a BGP Notification if it advertises a flexible match that exceeds the MRL of a given node.

3.3. Match Operators

We define a new FlowSpec component, Type TBD, named "Flexible Match Conditions".

Encoding: <type TBD (1 octet), match offset (2 octets), match type (1 octet), match length (1 octet), match term (variable)>

match offset - How far to offset from the anchor point. The format and use are detailed below (Section 3.3.1).

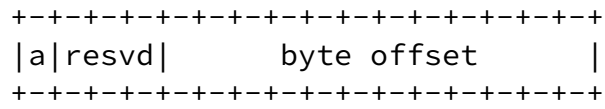
match type - A type code indicating what kind of match to perform. Type codes are their uses are detailed below (Section 3.3.2).

match length - An unsigned integer giving the length in octets of the match term that follows.

match term - The term to be matched. Format and use are detailed below (Section 3.3.2).

3.3.1. Match Offset

The match offset is encoded as:



a - Anchor flag, indicates if the offset is computed from the beginning of the relevant header (if the "a" flag is 0), or the beginning of the data following the relevant header (if the "a" flag is 1).

resvd - Reserved. Must be set to zero on transmission and ignored on receipt.

byte offset - The number of bytes to offset, beginning from the location specified by the "a" flag.

3.3.2. Match Type

This document defines the following match types:

Value	Match Type
0	Bitmask match
1	Numeric range match
2	Regular expression (regex) string match

Match Types

Match types 0 and 1 MUST be implemented. All other types are optional.

3.3.2.1. Bitmask match

This is encoded as {target, mask}, where target and mask are fields of equal length, inferred from the match length field.

target - Provides a bit string to be matched.

mask - Paired with the target field to create a bit string match. An unset bit is treated as a 'do not care' bit in the corresponding position in the target field. When a bit is set in the mask, the value of the bit in the corresponding location in the target field must match exactly.

The mask field is bitwise AND'ed with the packet data, and the result compared to the target field.

3.3.2.2. Integer range match

This is encoded as {low value, high value}, treated as an inclusive range of unsigned integers. The size of integers can be inferred from the match length field; the size is half the match length. For example, if length is 8, then the range is given by two four-octet unsigned integers.

low - The low value of the desired numeric range, expressed as an unsigned integer. This value MUST be numerically lower than the high value.

high - The high value of the desired numeric range, expressed as an unsigned integer. This value MUST be numerically higher than the low value.

3.3.2.3. Regular expression string match

Not every forwarding plane that supports filtering via FlowSpec is a hardware-accelerated Network Processor Unit (NPU) or Application-Specific Integrated Circuit (ASIC). Software-only forwarding planes, while less performant, may be able to filter on more complex match types.

There is a plethora of regular expression engines and their supported flavor. The specific flavor this match type refers to is the extended regular expression (ERE) as defined by [IEEE.1003-2.1992].

4. Error Handling

Malicious, misbehaving, or misunderstanding implementations could advertise semantically incorrect values. Care must be taken to minimize fallout from attempting to parse such data. Any well-behaved implementation SHOULD verify that the minimum packet length undergoing a match equals (match start header length + byte offset + value length).

5. Security Considerations

This document introduces no additional security considerations beyond those already covered in [RFC5575] .

6. IANA Considerations

IANA is requested to assign a type from the First Come First Served range of the "Flow Spec Component Types" registry:

Type Value	Name	Reference
TBD	Flexible Match Conditions	this document

IANA is requested to create a "BGP Flow Spec" group. The existing "Flow Spec Component Types" registry is to be a member of the "BGP Flow Spec" group.

IANA is requested to create a new registry, called "Flow Spec Match Type" within the (newly created) "BGP Flow Spec" group.

Reference: this document

Registry Owner/Change Controller: IESG

Registration procedures:

Range	Registration Procedures
0-127	IETF Review
128-249	First Come First Served
250-254	Experimental
255	Reserved

Note: a separate "owner" column is not provided because the owner of all registrations, once made, is "IESG".

IANA is requested to perform the following new allocations within the "Flow Spec Match Type" registry:

Value	Description	Reference
0	Bitmask match	(This document)
1	Numeric range match	(This document)
2	Regular expression (regex) string match	(This document)

7. Acknowledgements

Thanks to Philippe Bergeon, Ron Bonica, Jeff Haas, Sudipto Nandi, and Rafal Jan Szarecki for their valuable comments and suggestions.

8. References

8.1. Normative References

[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>.

[RFC5575] Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J., and D. McPherson, "Dissemination of Flow Specification Rules", RFC 5575, DOI 10.17487/RFC5575, August 2009, <https://www.rfc-editor.org/info/rfc5575>.

8.2. Informative References

[GTPv1-U] 3GPP, "General Packet Radio System (GPRS) Tunnelling Protocol User Plane (GTPv1-U)", 3GPP TS 29.281 10.3.0, September 2011.

[I-D.ietf-idr-flowspec-l2vpn] Weiguo, H., Eastlake, D., Uttaro, J., Litkowski, S., and S. Zhuang, "BGP Dissemination of L2VPN Flow Specification Rules", draft-ietf-idr-flowspec-l2vpn-10 (work in progress), May 2019.

[IEEE.1003-2.1992]

Institute of Electrical and Electronics Engineers, "Information Technology - Portable Operating System Interface (POSIX) - Part 2: Shell and Utilities (Vol. 1)", IEEE Standard 1003.2, 1992.

[RFC2890] Dommety, G., "Key and Sequence Number Extensions to GRE", RFC 2890, DOI 10.17487/RFC2890, September 2000, <https://www.rfc-editor.org/info/rfc2890>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <https://www.rfc-editor.org/info/rfc7348>.

8.3. URIs

[1] https://github.com/649/Memcrashed-DDoS-Exploit

[2] https://www.facebook.com/PutinStresser/photos/a.1687498801469198/2024483917770683/?type=3

Authors' Addresses

Anurag Khare (editor)
Juniper Networks, Inc.
2251 Corporate Park Drive
Herndon, Virginia 20171
US

Email: anuragk@juniper.net

John Scudder
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089
US

Email: jgs@juniper.net

Luay Jalil
Verizon

Email: luay.jalil@one.verizon.com

Michael Gallagher
Verizon

Email: michael.gallagher@verizon.com

Kirill Kasavchenko
NetScout

Email: Kirill.Kasavchenko@netscout.com