Network Working Group   E. Crabbe
Internet-Draft
Intended status: Standard Track   L. Yong
                                                        Huawei USA
                                                            X. Xu
                                               Huawei Technologies
                                                       T. Herbert
                                                           Google

Expires: January 2015   July 4, 2015

GRE-in-UDP Encapsulation
draft-ietf-tsvwg-gre-in-udp-encap-07

Abstract

   This document describes a method of encapsulating network protocol
   packets within GRE and UDP headers. In this encapsulation, the
   source UDP port can be used as an entropy field for purposes of load
   balancing, while the protocol of the encapsulated packet in the GRE
   payload is identified by the GRE Protocol Type. Usage restrictions
   apply to GRE-in-UDP usage for traffic that is not congestion
   controlled and to UDP zero checksum usage with IPv6.

Status of This Document

   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 January 4, 2015.

Copyright Notice

Table of Contents

[1](#). **Introduction**

   Load balancing, or more specifically statistical multiplexing of
   traffic using Equal Cost Multi-Path (ECMP) and/or Link Aggregation
   Groups (LAGs) in IP networks is a widely used technique for creating
   higher capacity networks out of lower capacity links. Most existing
   routers in IP networks are already capable of distributing IP
   traffic flows over ECMP paths and/or LAGs on the basis of a hash
   function performed on flow invariant fields in IP packet headers and
   their payload protocol headers. Specifically, when the IP payload is
   a User Datagram Protocol (UDP)[RFC768] or Transmission Control
   Protocol (TCP) [RFC793] packet, router hash functions frequently
   operate on the five-tuple of source IP address, destination IP
   address, source port, destination port, and protocol/next-header

   Several encapsulation techniques are commonly used in IP networks,
   such as Generic Routing Encapsulation (GRE) [RFC2784], MPLS
   [RFC4023] and L2TPv3 [RFC3931]. GRE is an increasingly popular
   encapsulation choice. Unfortunately, use of common GRE endpoints may
   reduce the entropy available for use in load balancing, especially
   in environments where the GRE Key field [RFC2890] is not readily
   available for use as entropy in forwarding decisions.

   This document defines a generic GRE-in-UDP encapsulation for
   tunneling network protocol packets across an IP network. The GRE
   header provides payload protocol type as an EtherType in the
   protocol type field [RFC2784][GREIPV6], and the UDP header provides
   additional entropy by way of its source port. GRE-in-UDP offers the
   additional possibility of using GRE across networks that might
   otherwise disallow it; for instance GRE-in-UDP may be used to bridge
   two islands where GRE is used natively across the Internet.

   This encapsulation method requires no changes to the transit IP
   network. Hash functions in most existing IP routers may utilize and
   benefit from the use of a GRE-in-UDP tunnel without needing any
   change or upgrade to their ECMP implementation. The encapsulation
   mechanism is applicable to a variety of IP networks including Data
   Center and wide area networks.

   1.1. Applicability Statement

   GRE encapsulation is widely used for many applications. For example,
   to redirect IP traffic to traverse a different path instead of the
   default path in an operator network, to tunnel private network
   traffic over a public network by use of public IP network addresses,

to tunnel IPv6 traffic over an IPv4 network, tunnel Ethernet traffic
over IP networks, and etc.

When using GRE-in-UDP encapsulation, encapsulated traffic will be
treated as a UDP application in an IP network. Thus GRE-in-UDP
tunnel needs to meet UDP application guidelines specified in
[RFC5405bis], which constrains GRE-in-UDP tunnel usage. The concerns
of GRE-in-UDP as a UDP application are addressed in Section 5 and 6.
As a result, GRE-in-UDP encapsulation MUST be used when one of
following conditions is true:

1) Within a single operator network or networks of an adjacent set
   of cooperating network operators where traffic is managed to
   avoid congestion.

2) The payload type is IP or a type of network protocol that has
   congestion control capability when the encapsulated traffic is
   over the Internet.

GRE-in-UDP encapsulation may be used to encapsulate already tunneled
traffic, i.e. tunnel-in-tunnel. The tunneled traffic may use GRE-in-
UDP or other tunnel encapsulation. In this case, GRE-in-UDP tunnel
end points treat other tunnel endpoints as of the end hosts for the
traffic and do not differentiate such end hosts from other end hosts.

2. Terminology

The terms defined in [RFC768][RFC2784] are used in this document.

2.1. 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 [RFC2119].

3. Encapsulation in UDP
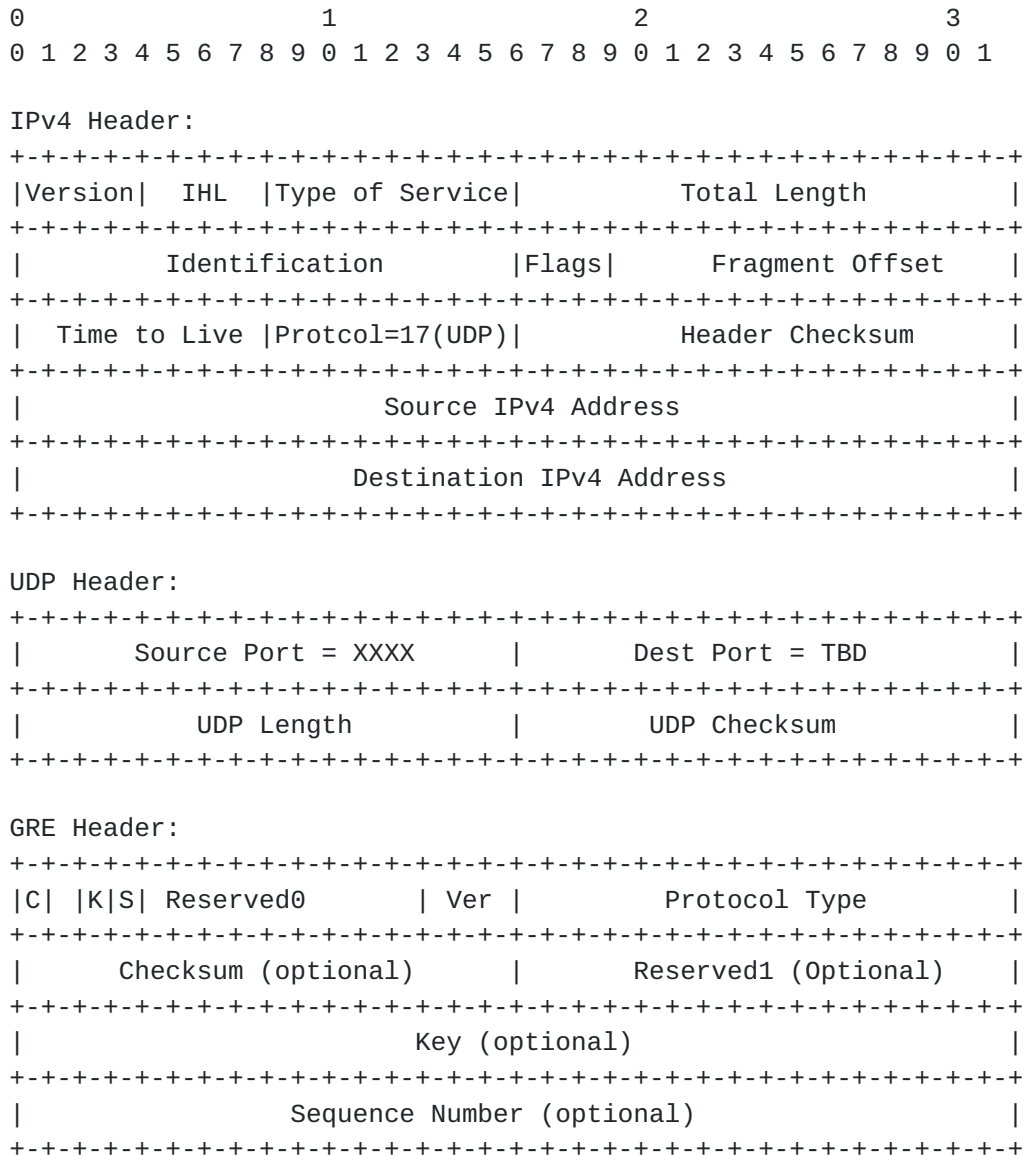
GRE-in-UDP encapsulation format is shown as follows:

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

       IPv4 Header:
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |Version|  IHL  |Type of Service|         Total Length          |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |         Identification        |Flags|     Fragment Offset     |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |  Time to Live |Protcol=17(UDP)|        Header Checksum         |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                      Source IPv4 Address                      |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                   Destination IPv4 Address                    |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

       UDP Header:
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |       Source Port = XXXX      |      Dest Port = TBD          |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |          UDP Length           |       UDP Checksum            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

       GRE Header:
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |C|  |K|S| Reserved0       | Ver |         Protocol Type        |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |       Checksum (optional)     |      Reserved1 (Optional)     |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                       Key (optional)                          |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                  Sequence Number (optional)                   |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                   Figure 1 UDP+GRE Headers in IPv4

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

    IPv6 Header:
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Version| Traffic Class |           Flow Label                  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |        Payload Length         | NxtHdr=17(UDP)|   Hop Limit   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    +                                                               +
    |                                                               |
    +                  Outer Source IPv6 Address                    +
    |                                                               |
    +                                                               +
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    +                                                               +
    |                                                               |
    +               Outer Destination IPv6 Address                 +
    |                                                               |
    +                                                               +
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    UDP Header:
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      Source Port = XXXX       |       Dest Port = TBD         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          UDP Length           |        UDP Checksum           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    GRE Header:
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |C|  |K|S| Reserved0       | Ver |        Protocol Type         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      Checksum (optional)      |      Reserved1 (Optional)     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        Key (optional)                         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                   Sequence Number (optional)                  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2   UDP+GRE Headers in IPv6

The contents of the IP, UDP, and GRE headers that are relevant in
this encapsulation are described below.

3.1. IP Header

An encapsulator MUST encode its own IP address as the source IP
address and the decapsulator's IP address as the destination IP
address. The TTL field in the IP header must be set to a value
appropriate for delivery of the encapsulated packet to the peer of
the encapsulation.

3.2. UDP Header

3.2.1. Source Port

The UDP source port contains a 16-bit entropy value that is
generated by the encapsulator to identify a flow for the
encapsulated packet. The port value SHOULD be within the ephemeral
port range. IANA suggests this range to be 49152 to 65535, where the
high order two bits of the port are set to one. This provides
fourteen bits of entropy for the inner flow identifier. In the case
that an encapsulator is unable to derive flow entropy from the
payload header, it should set a randomly selected constant value for
UDP source port to avoid payload packet flow reordering.

The source port value for a flow set by an encapsulator MAY change
over the lifetime of the encapsulated flow. For instance, an
encapsulator may change the assignment for Denial of Service (DOS)
mitigation or as a means to effect routing through the ECMP network.
An encapsulator SHOULD NOT change the source port selected for a
flow more than once every thirty seconds.

How an encapsulator generates entropy from the payload is outside
the scope of this document.

3.2.2. Destination Port

The destination port of the UDP header is set the GRE/UDP port (TBD)
(see Section 8).

3.2.3. Checksum

The UDP checksum is set and processed per [RFC768] and [RFC1122] for
IPv4, and [RFC2460] for IPv6. Requirements for checksum handling and
use of zero UDP checksums are detailed in Section 5.

3.2.4. Length

The usage of this field is in accordance with the current UDP
specification in [RFC768]. This length will include the UDP header
(eight bytes), GRE header, and the GRE payload (encapsulated packet).

3.3. GRE Header

An encapsulator sets the protocol type (EtherType) of the packet
being encapsulated in the GRE Protocol Type field.

An encapsulator may set the GRE Key Present, Sequence Number Present,
and Checksum Present bits and associated fields in the GRE header as
defined by [RFC2784] and [RFC2890].

The GRE checksum MAY be enabled to protect the GRE header and
payload. An encapsulator SHOULD NOT enable both the GRE checksum and
UDP checksum simultaneously as this would be mostly redundant. Since
the UDP checksum covers more of the packet including the GRE header
and payload, the UDP checksum SHOULD have preference to using GRE
checksum.

An implementation MAY use the GRE keyid to authenticate the
encapsulator. In this model, a shared value is either configured or
negotiated between an encapsulator and decapsulator. When a
encapsulated packet is received with the keyid present, it is
checked to see if it is valid for the source to have set for the
tunnel packet was sent on. An implementation MAY enforce that a
keyid be used for source authentication on selected tunnels. When a
decapsulator determines a presented keyid is not valid for the
source to send or the keyid is absent and is considered required for
authenticating the encapsulator for a tunnel, the packet MUST be
dropped.

4. Encapsulation Process Procedures

The GRE-in-UDP encapsulation allows encapsulated packets to be
forwarded through "GRE-UDP tunnels". When performing GRE-in-UDP
encapsulation by the encapsulator, the entropy value would be
generated by the encapsulator and then be filled in the Source Port
field of the UDP header. The Destination Port field is set to a
value (TBD) allocated by IANA to indicate that the UDP tunnel
payload is a GRE packet. The Protocol Type header field in GRE
header is set to the EtherType value corresponding to the protocol
of the encapsulated packet.

Intermediate routers, upon receiving these UDP encapsulated packets, could balance these packets based on the hash of the five-tuple of UDP packets.

Upon receiving these UDP encapsulated packets, the decapsulator would decapsulate them by removing the UDP and GRE headers and then process them accordingly.

Note: Each UDP tunnel is unidirectional, as GRE-in-UDP traffic is sent to the IANA-allocated UDP Destination Port, and in particular, is never sent back to any port used as a UDP Source Port (which serves solely as a source of entropy). This is at odds with a common middlebox (e.g., firewall) assumption that bidirectional traffic uses a common pair of UDP ports. As a result, arranging to pass bidirectional GRE-in-UDP traffic through middleboxes may require separate configuration for each direction of traffic.

GRE-in-UDP allows encapsulation of unicast, broadcast, or multicast traffic. Entropy may be generated from the header of encapsulated unicast or broadcast/multicast packets at an encapsulator. The mapping mechanism between the encapsulated multicast traffic and the multicast capability in the IP network is transparent and independent to the encapsulation and is otherwise outside the scope of this document.

To provide entropy for ECMP, GRE-in-UDP does not rely on GRE keep-alive. It is RECOMMENED no use of GRE keep-alive in the GRE-in-UDP tunnel. This aligns with middlebox traversal guidelines in Section 3.5 of [RFC5405bis].

4.1. MTU and Fragmentation

Regarding fragmentation, an encapsulator SHOULD perform fragmentation [GREMTU] on a packet before encapsulation and factor in both GRE and UDP header bytes in the effective Maximum Transmission Unit (MTU) size. Not performing the fragmentation will cause the packets exceeding network MTU size to be dropped or fragmented in the network. An encapsulator MUST use the same source UDP port for all packet fragments to ensure that the transit routers will forward the packet fragments on the same path. An operator should factor in the additional bytes of overhead when considering an MTU size for the payload to avoid the likelihood of fragmentation.

Fragmented packets MUST be reassembled at the decapsulator prior to being sent to a (payload) application. Packet fragmentation and reassembling process is outside the scope of the document.

4.2. Differentiated Services

To ensure that tunneled traffic gets the same treatment over the IP
network, prior to the encapsulation process, an encapsulator should
process the payload to get the proper parameters to fill into the IP
header such as DiffServ [RFC2983]. Encapsulation end points that
support ECN must use the method described in [RFC6040] for ECN
marking propagation. This process is outside of the scope of this
document.

5. UDP Checksum Handling

5.1. UDP Checksum with IPv4

For UDP in IPv4, the UDP checksum MUST be processed as specified in
[RFC768] and [RFC1122] for both transmit and receive. An
encapsulator MAY set the UDP checksum to zero for performance or
implementation considerations. The IPv4 header includes a checksum
which protects against mis-delivery of the packet due to corruption
of IP addresses. The UDP checksum potentially provides protection
against corruption of the UDP header, GRE header, and GRE payload.
Enabling or disabling the use of checksums is a deployment
consideration that should take into account the risk and effects of
packet corruption, and whether the packets in the network are
protected by other, possibly stronger mechanisms such as the
Ethernet CRC.

When a decapsulator receives a packet, the UDP checksum field MUST
be processed. If the UDP checksum is non-zero, the decapsulator MUST
verify the checksum before accepting the packet. By default a
decapsulator SHOULD accept UDP packets with a zero checksum. A node
MAY be configured to disallow zero checksums per [RFC1122]; this may
be done selectively, for instance disallowing zero checksums from
certain hosts that are known to be sending over paths subject to
packet corruption. If verification of a non-zero checksum fails, a
decapsulator lacks the capability to verify a non-zero checksum, or
a packet with a zero-checksum was received and the decapsulator is
configured to disallow, the packet MUST be dropped and an event MAY
be logged.

5.2. UDP Checksum with IPv6

For UDP in IPv6, the UDP checksum MUST be processed as specified in
[RFC768] and [RFC2460] for both transmit and receive.

When UDP is used over IPv6, the UDP checksum is relied upon to
protect both the IPv6 and UDP headers from corruption, and so MUST
used with the following exceptions:

   a. Use of GRE-in-UDP in networks under single administrative
      control (such as within a single operator's network) where it
      is known (perhaps through knowledge of equipment types and
      lower layer checks) that packet corruption is exceptionally
      unlikely and where the operator is willing to take the risk of
      undetected packet corruption.

   b. Use of GRE-in-UDP in networks under single administrative
      control (such as within a single operator's network) where it
      is judged through observational measurements (perhaps of
      historic or current traffic flows that use a non-zero checksum)
      that the level of packet corruption is tolerably low and where
      the operator is willing to take the risk of undetected packet
      corruption.

   c. Use of GRE-in-UDP for traffic delivery for applications that
      are tolerant of mis-delivered or corrupted packets (perhaps
      through higher layer checksum, validation, and retransmission
      or transmission redundancy) where the operator is willing to
      rely on the applications using the tunnel to survive any
      corrupt packets.

For these exceptions, the UDP zero-checksum mode can be used.
However the use of the UDP zero-checksum mode must meet the
requirements specified in [RFC6935] and [RFC6936] as well at the
additional requirements stated below.

These exceptions may also be extended to the use of GRE-in-UDP
within a set of closely cooperating network administrations (such as
network operators who have agreed to work together in order to
jointly provide specific services).

As such, for IPv6, the UDP checksum for GRE-in-UDP MUST be used as
specified in [RFC768] and [RFC2460] for tunnels that span multiple
networks whose network administrations do not cooperate closely,
even if each non-cooperating network administration independently
satisfies one or more of the exceptions for UDP zero-checksum mode
usage with GRE-in-UDP over IPv6.

The following additional requirements apply to implementation and
use of UDP zero-checksum mode for GRE-in-UDP over IPv6:

a. Use of the UDP checksum with IPv6 MUST be the default
   configuration of all GRE-in-UDP implementations.

b. The GRE-in-UDP implementation MUST comply with all requirements
   specified in Section 4 of [RFC6936] and with requirement 1
   specified in Section 5 of [RFC6936].

c. By default a decapsulator MUST disallow receipt of GRE-in-UDP
   packets with zero UDP checksums with IPv6. Zero checksums May
   selectively be enabled for certain source address. A decapsulator
   MUST check that the source and destination IPv6 addresses are
   valid for the GRE-in-UDP tunnel on which the packet was received
   if that tunnel uses UDP zero-checksum mode and discard any packet
   for which this check fails.

d. An encapsulator SHOULD use different IPv6 addresses for each GRE-
   in-UDP tunnel that uses UDP zero-checksum mode regardless of the
   decapsulator in order to strengthen the decapsulator's check of
   the IPv6 source address (i.e., the same IPv6 source address
   SHOULD NOT be used with more than one IPv6 destination address,
   independent of whether that destination address is a unicast or
   multicast address). When this is not possible, it is RECOMMENDED
   to use each source IPv6 address for as few UDP zero-checksum mode
   GRE-in-UDP tunnels as is feasible.

e. Any middlebox support for GRE-in-UDP with UDP zero-checksum mode
   for IPv6 MUST comply with requirements 1 and 8-10 in Section 5 of
   [RFC6936].[RFC6936].

f. Measures SHOULD be taken to prevent IPv6 traffic with zero UDP
   checksums from "escaping" to the general Internet; see Section 6
   for examples of such measures.

g. IPv6 traffic with zero UDP checksums MUST be actively monitored
   for errors by the network operator.

h. If a packet with a non-zero checksum is received, the checksum
   MUST be verified before accepting the packet. This is regardless
   of whether a tunnel encapsulator and decapsulator have been
   configured with UDP zero-checksum mode.

The above requirements do not change either the requirements
specified in [RFC2460] as modified by [RFC6935] or the requirements
specified in [RFC6936].

The requirement to check the source IPv6 address in addition to the
destination IPv6 address, plus the strong recommendation against

reuse of source IPv6 addresses among GRE-in-UDP tunnels collectively provide some mitigation for the absence of UDP checksum coverage of the IPv6 header. Additional assurance is provided by the restrictions in the above exceptions that limit usage of IPv6 UDP zero-checksum mode to well-managed networks for which GRE encapsulated packet corruption has not been a problem in practice.

Hence GRE-in-UDP is suitable for transmission over lower layers in the well-managed networks that are allowed by the exceptions stated above and the rate of corruption of the inner IP packet on such networks is not expected to increase by comparison to GRE traffic that is not encapsulated in UDP. For these reasons, GRE-in-UDP does not provide an additional integrity check except when GRE checksum is used when UDP zero-checksum mode is used with IPv6, and this design is in accordance with requirements 2, 3 and 5 specified in Section 5 of [RFC6936].

GRE does not accumulate incorrect state as a consequence of GRE header corruption. A corrupt GRE results in either packet discard or forwarding of the packet without accumulation of GRE state. GRE checksum MAY be used for protecting GRE header and payload. Active monitoring of GRE-in-UDP traffic for errors is REQUIRED as occurrence of errors will result in some accumulation of error information outside the protocol for operational and management purposes. This design is in accordance with requirement 4 specified in Section 5 of [RFC6936].

The remaining requirements specified in Section 5 of [RFC6936] are inapplicable to GRE-in-UDP. Requirements 6 and 7 do not apply because GRE does not have a GRE-generic control feedback mechanism. Requirements 8-10 are middlebox requirements that do not apply to GRE-in-UDP tunnel endpoints, but see Section 5.2.1 for further middle box discussion.

It is worth to mention that the use of a zero UDP checksum should present the equivalent risk of undetected packet corruption when sending similar packet using GRE-in-IPv6 without UDP and without GRE checksums.

In summary, UDP zero-checksum mode for IPv6 is allowed to be used with GRE-in-UDP when one of the three exceptions specified above applies, provided that additional requirements stated above are complied with. Otherwise the UDP checksum MUST be used for IPv6 as specified in [RFC768] and [RFC2460]. Use of GRE checksum favors non-use of the UDP checksum.

5.2.1. Middlebox Considerations

IPv6 datagrams with a zero UDP checksum will not be passed by any
middlebox that validates the checksum based on [RFC2460] or that
updates the UDP checksum field, such as NATs or firewalls. Changing
this behavior would require such middleboxes to be updated to
correctly handle datagrams with zero UDP checksums. The GRE-in-UDP
encapsulation does not provide a mechanism to safely fall back to
using a checksum when a path change occurs redirecting a tunnel over
a path that includes a middlebox that discards IPv6 datagrams with a
zero UDP checksum. In this case the GRE-in-UDP tunnel will be
black-holed by that middlebox. Recommended changes to allow
firewalls, NATs and other middleboxes to support use of an IPv6 zero
UDP checksum are described in Section 5 of [RFC6936].

6. **Congestion Considerations**

Section 3.1.7 of [RFC5405bis] discussed the congestion implications
of UDP tunnels. As discussed in [RFC5405bis], because other flows
can share the path with one or more UDP tunnels, congestion control
[RFC2914] needs to be considered.

A major motivation for GRE-in-UDP encapsulation is to tunnel a
network protocol over IP network and improve the use of multipath
(such as ECMP) in cases where traffic is to traverse routers which
are able to hash on UDP Port and IP address. As such, in many cases
this may reduce the occurrence of congestion and improve usage of
available network capacity. However, it is also necessary to ensure
that the network, including applications that use the network,
responds appropriately in more difficult cases, such as when link or
equipment failures have reduced the available capacity.

The impact of congestion must be considered both in terms of the
effect on the rest of the network over which packets are sent in UDP
tunnels, and in terms of the effect on the flows that are sent by
UDP tunnels. The potential impact of congestion from a UDP tunnel
depends upon what sort of traffic is carried over the tunnel, as
well as the path of the tunnel.

GRE encapsulation is widely used to carry a wide range of network
protocols and traffic. In many cases GRE encapsulation is used to
carry IP traffic. IP traffic is generally assumed to be congestion
controlled, and thus a tunnel carrying general IP traffic (as might
be expected to be carried across the Internet) generally does not
need additional congestion control mechanisms. As specified in RFC
5405:

 "IP-based traffic is generally assumed to be congestion-controlled,
i.e., it is assumed that the transport protocols generating IP-based
traffic at the sender already employ mechanisms that are sufficient
to address congestion on the path. Consequently, a tunnel carrying
IP-based traffic should already interact appropriately with other
traffic sharing the path, and specific congestion control mechanisms
for the tunnel are not necessary."

For this reason, where GRE-in-UDP tunneling is used to carry IP
traffic that is known to be congestion controlled, the UDP tunnels
MAY be used within a single network or across multiple networks,
with cooperating network operators. Internet IP traffic is
generally assumed to be congestion-controlled.

However, GRE-in-UDP tunneling can be also used to carry traffic that
is not necessarily congestion controlled. In such cases network
operators may avoid congestion by careful provisioning of their
networks, by rate limiting of user data traffic, and/or by using
Traffic Engineering tools to monitor the network segments and
dynamically steers traffic away from potentially congested links.

For this reason, where the GRE payload traffic is not congestion
controlled, GRE-in-UDP tunnels MUST only be used within a single
operator's network that utilizes careful provisioning (e.g., rate
limiting at the entries of the network while over-provisioning
network capacity) to ensure against congestion, or within a limited
number of networks whose operators closely cooperate in order to
jointly provide this same careful provisioning.

As such, GRE-in-UDP MUST NOT be used over the general Internet, or
over non-cooperating network operators, to carry traffic that is not
congestion-controlled.

Measures SHOULD be taken to prevent non-congestion-controlled GRE-
in-UDP traffic from "escaping" to the general Internet, e.g.:

o Physical or logical isolation of the links carrying GRE-in-UDP
   from the general Internet.

o Deployment of packet filters that block the UDP ports assigned
   for GRE-in-UDP.

o Imposition of restrictions on GRE-in-UDP traffic by software
   tools used to set up GRE-in-UDP tunnels between specific end
   systems (as might be used within a single data center).

   o Use of a "Managed Circuit Breaker" for the tunneled traffic as
      described in [CB].

7. Backward Compatibility

   It is assumed that tunnel ingress routers must be upgraded in order
   to support the encapsulations described in this document.

   No change is required at transit routers to support forwarding of
   the encapsulation described in this document.

   If a router that is intended for use as a decapsulator does not
   support or enable GRE-in-UDP encapsulation described in this
   document, it will not be listening on destination port (TBD). In
   these cases, the router will conform to normal UDP processing and
   respond to an encapsulator with an ICMP message indicating "port
   unreachable" according to [RFC792]. Upon receiving this ICMP
   message, the node MUST NOT continue to use GRE-in-UDP encapsulation
   toward this peer without management intervention.

8. IANA Considerations

   IANA is requested to make the following allocations:

   One UDP destination port number for the indication of GRE

        Service Name: GRE-in-UDP
        Transport Protocol(s): UDP
        Assignee: IESG <iesg@ietf.org>
        Contact: IETF Chair <chair@ietf.org>
        Description: GRE-in-UDP Encapsulation
        Reference: [This.I-D]
        Port Number: TBD
        Service Code: N/A
        Known Unauthorized Uses: N/A
        Assignment Notes: N/A

   One UDP destination port number for the indication of GRE with DTLS

        Service Name: GRE-UDP-DTLS
        Transport Protocol(s): UDP
        Assignee: IESG <iesg@ietf.org>

```
          Contact: IETF Chair <chair@ietf.org>
          Description: GRE-in-UDP Encapsulation with DTLS
          Reference: [This.I-D]
          Port Number: TBD
          Service Code: N/A
          Known Unauthorized Uses: N/A
          Assignment Notes: N/A
```

9. Security Considerations

UDP and GRE encapsulation does not effect security for the payload
protocol. When using GRE-in-UDP, Network Security in a network is
mostly equivalent to that of a network using GRE.

Datagram Transport Layer Security (DTLS) [RFC6347] can be used for
application security and can preserve network and transport layer
protocol information. Specifically, if DTLS is used to secure the
GRE-in-UDP tunnel, the destination port of the UDP header MUST be
set to an IANA-assigned value (TBD2) indicating GRE-in-UDP with DTLS,
and that UDP port MUST NOT be used for other traffic. The UDP
source port field can still be used to add entropy, e.g., for load-
sharing purposes. DTLS usage is limited to a single DTLS session
for any specific tunnel encapsulator/ decapsulator pair (identified
by source and destination IP addresses). Both IP addresses MUST be
unicast addresses - multicast traffic is not supported when DTLS is
used. A GRE-in-UDP tunnel decapsulator implementation that supports
DTLS is expected to be able to establish DTLS sessions with multiple
tunnel encapsulators, and likewise an GRE-in-UDP tunnel encapsulator
implementation is expected to be able to establish DTLS sessions
with multiple decapsulators (although different source and/or
destination IP addresses may be involved -see Section 5.2 for
discussion of one situation where use of different source IP
addresses is important).

Use of ICMP for signaling of the GRE-in-UDP encapsulation capability
adds a security concern. Upon receiving an ICMP message and before
taking an action on it, the ingress MUST validate the IP address
originating against tunnel egress address and MUST evaluate the
packet header returned in the ICMP payload to ensure the source port
is the one used for this tunnel. The mechanism for performing this
validation is out of the scope of this document.

In an instance where the UDP source port is not set based on the
flow invariant fields from the payload header, a random port SHOULD
be selected in order to minimize the vulnerability to off-path
attacks. [RFC6056]. The random port may also be periodically changed

to mitigate certain denial of service attacks. How the source port
randomization occurs is outside scope of this document.

Using one standardized value in UDP destination port for an
encapsulation indication may increase the vulnerability of off-path
attack. To overcome this, an alternate port may be agreed upon to
use between an encapsulator and decapsulator [RFC6056]. How the
encapsulator end points communicate the value is outside scope of
this document.

This document does not require that decapsulator validates the IP
source address of the tunneled packets (with the exception that the
IPv6 source address MUST be validated when UDP zero-checksum mode is
used with IPv6), but it should be understood that failure to do so
presupposes that there is effective destination-based (or a
combination of source-based and destination-based) filtering at the
boundaries.

Corruption of GRE header can cause a privacy and security concern
for some applications that rely on the key field for traffic
segregation. When GRE key field is used for privacy and security,
ether UDP checksum or GRE checksum SHOULD be used for GRE-in-UDP
with both IPv4 and IPv6, and in particular, when UDP zero-checksum
mode is used, GRE checksum SHOULD be used.

10. Acknowledgements

Authors like to thank Vivek Kumar, Ron Bonica, Joe Touch, Ruediger
Geib, Lar Edds, Lloyd, and many others for their review and valuable
input on this draft.

Thank the design team led by David Black (members: Ross Callon,
Gorry Fairhurst, Xiaohu Xu, Lucy Yong) to efficiently work out the
descriptions for the congestion considerations and IPv6 UDP zero
checksum.

11. Contributors

The following people all contributed significantly to this document
and are listed below in alphabetical order:

David Black
EMC Corporation
176 South Street

      Hopkinton, MA 01748
      USA

      Email: david.black@emc.com

      Ross Callon
      Juniper Networks
      10 Technology Park Drive
      Westford, MA 01886
      USA

      Email: rcallon@juniper.net

      John E. Drake
      Juniper Networks

      Email: jdrake@juniper.net

      Gorry Fairhurst
      University of Aberdeen

      Email: gorry@erg.abdn.ac.uk

      Yongbing Fan
      China Telecom
      Guangzhou, China.
      Phone: +86 20 38639121

      Email: fanyb@gsta.com

      Adrian Farrel
      Juniper Networks

      Email: adrian@olddog.co.uk

      Vishwas Manral
      Hewlett-Packard Corp.
      3000 Hanover St, Palo Alto.

      Email: vishwas.manral@hp.com

      Carlos Pignataro
      Cisco Systems

7200-12 Kit Creek Road
Research Triangle Park, NC 27709 USA

EMail: cpignata@cisco.com

12. References

12.1. Normative References

[RFC768]   Postel, J., "User Datagram Protocol", STD 6, RFC 768,
           August 1980.

[RFC1122] Braden, R., "Requirements for Internet Hosts --
           Communication Layers", RFC1122, October 1989.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
           Requirement Levels", BCP 14, RFC2119, March 1997.

[RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P.
           Traina, "Generic Routing Encapsulation (GRE)", RFC 2784,
           March 2000.

[RFC2890] Dommety, G., "Key and Sequence Number Extensions to GRE",
           RFC2890, September 2000.

[RFC2983] Black, D., "Differentiated Services and Tunnels", RFC2983,
           October 2000.

[RFC5405bis] Eggert, L., "Unicast UDP Usage Guideline for
           Application Designers", draft-ietf-tsvwg-rfc5405bis, work
           in progress.

[RFC6040] Briscoe, B., "Tunneling of Explicit Congestion
           Notification", RFC6040, November 2010.

[RFC6347] Rescoria, E., Modadugu, N., "Datagram Transport Layer
           Security Version 1.2", RFC6347, 2012.

[RFC6438] Carpenter, B., Amante, S., "Using the IPv6 Flow Label for
           Equal Cost Multipath Routing and Link Aggregation in
           tunnels", RFC6438, November, 2011.

[RFC6935] Eubanks, M., Chimento, P., and M. Westerlund, "IPv6 and
           UDP Checksums for Tunneled Packets", RFC 6935, April 2013.

   [RFC6936] Fairhurst, G. and M. Westerlund, "Applicability Statement
              for the Use of IPv6 UDP Datagrams with Zero Checksums",
              RFC 6936, April 2013.

   12.2. Informative References

   [RFC792] Postel, J., "Internet Control Message Protocol", STD 5, RFC
              792, September 1981.

   [RFC793] DARPA, "Transmission Control Protocol", RFC793, September
              1981.

   [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6
              (IPv6) Specification", RFC 2460, December 1998.

   [RFC2914] Floyd, S.,"Congestion Control Principles", RFC2914,
              September 2000.

   [RFC3931] Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling
              Protocol - Version 3 (L2TPv3)", RFC 3931, March 2005.

   [RFC4023] Worster, T., Rekhter, Y., and E. Rosen, "Encapsulating
              MPLS in IP or Generic Routing Encapsulation (GRE)", RFC
              4023, March 2005.

   [RFC6056] Larsen, M. and Gont, F., "Recommendations for Transport-
              Protocol Port Randomization", RFC6056, January 2011.

   [GREMTU]   Bonica, R., "A Fragmentation Strategy for Generic Routing
              Encapsulation (GRE)", draft-ietf-intarea-gre-mtu, work in
              progress.

   [CB]   Fairhurst, G., "Network Transport Circuit Breakers",
              draft-fairhurst-tsvwg-circuit-breaker-01, work in
              progress.

   [GREIPV6] Pignataro, C., Bonica, R., Krishnan, S., "IPv6 Support for
              Generic Routing Encapsulation (GRE)", draft-ietf-intarea-
              gre-ipv6, work in progress.

13. Authors' Addresses

   Edward Crabbe

   Email: edward.crabbe@gmail.com

   Lucy Yong

Huawei Technologies, USA

Email: lucy.yong@huawei.com

Xiaohu Xu
Huawei Technologies,
Beijing, China

Email: xuxiaohu@huawei.com

Tom Herbert
Google
1600 Amphitheatre Parkway
Mountain View, CA
Email : tom@herbertland.com