       Requirements for the design of a Substrate Protocol for User Datagrams
                                    (SPUD)
                        draft-trammell-spud-req-04

Abstract

   We have identified the potential need for a UDP-based encapsulation
   protocol to allow explicit cooperation with middleboxes while using
   new, encrypted transport protocols. This document proposes an
   initial set of requirements for such a protocol, and discusses
   tradeoffs to be made in further refining these requirements.

Status of This Memo

Copyright Notice

Table of Contents

1. Motivation

    A number of efforts to create new transport protocols or experiment
    with new network behaviors in the Internet have been built on top of
    UDP, as it traverses firewalls and other middleboxes more readily
    than new protocols do. Each such effort must, however, either manage
    its flows within common middlebox assumptions for UDP or train the
    middleboxes on the new protocol (thus losing the benefit of using
    UDP). A common Substrate Protocol for User Datagrams (SPUD) would
    allow each effort to re-use a set of shared methods for notifying
    middleboxes of the flows' semantics, thus avoiding both the
    limitations of current flow semantics and the need to re-invent the
    mechanism for notifying the middlebox of the new semantics.

    As a concrete example, it is common for some middleboxes to tear down
    required state (such as NAT bindings) very rapidly for UDP flows. By
    notifying the path that a particular transport using UDP maintains
    session state and explicitly signals session start and stop using the
    substrate, the using protocol may reduce or avoid the need for
    heartbeat traffic.

    The intention of this work is to make it possible to define and
    deploy new transport protocols that use encryption to protect their
    own operation as well as the confidentiality, authenticity,
    integrity, and linkability resistance of their payloads. The
    accelerating deployment of encryption will render obsolete network
    operations techniques that rely on packet inspection and modification
    based upon assumptions about the protocols in use. This work will
    allow the replacement the current regime of middlebox inspection and
    modification of transport and application-layer headers and payload
    with one that allows inspection only of information explicitly
    exposed by the endpoints, and modification of such information only
    under endpoint control.

    Any selective exposure of traffic metadata outside a relatively
    restricted trust domain must be advisory, non-negotiated, and
    declarative rather than imperative. As with other signaling systems,
    exposure of specific elements must be carefully assessed for privacy
    risks and the total of exposed elements must be so assessed. Each
    exposed parameter should also be independently verifiable, so that
    each entity can assign its own trust to other entities. Basic
    transport over the substrate must continue working even if signaling
    is ignored or stripped, to support incremental deployment. These
    restrictions on vocabulary are discussed further in
    [I-D.trammell-stackevo-explicit-coop]. This discussion includes
    privacy and trust concerns as well as the need for strong incentives
    for middlebox cooperation based on the information that are exposed.

Within this document, requirements are presented for a facility
implementable as an encapsulation protocol, atop which new transports
("superstrates") can be built. Alternately, these could be viewed as
a set of requirements for future transport protocol development
without a layer separation between the transport and the superstrate.

This document defines a specific set of requirements for a SPUD
facility, based on analysis on a target set of applications. It is
intended as the basis for determining the next steps to make progress
in this space, including possibly chartering a working group for
specific protocol engineering work.

2. History

An outcome of the IAB workshop on Stack Evolution in a Middlebox
Internet (SEMI) [RFC7663], held in Zurich in January 2015, was a
discussion on the creation of a substrate protocol to support the
deployment of new transport protocols in the Internet. Assuming that
a way forward for transport evolution in user space would involve
encapsulation in UDP datagrams, the workshop noted that it may be
useful to have a facility built atop UDP to provide minimal signaling
of the semantics of a flow that would otherwise be available in TCP.
At the very least, indications of first and last packets in a flow
may assist firewalls and NATs in policy decision and state
maintenance. This facility could also provide minimal application-
to- path and path-to-application signaling, though there was less
agreement about what should or could be signaled here. Further
transport semantics would be used by the protocol running atop this
facility, but would only be visible to the endpoints, as the
transport protocol headers themselves would be encrypted, along with
the payload, to prevent inspection or modification. This encryption
might be accomplished by using DTLS [RFC6347] as a subtransport
[I-D.huitema-tls-dtls-as-subtransport] or by other suitable methods.

The Substrate Protocol for User Datagrams (SPUD) BoF was held at IETF
92 in Dallas in March 2015 to develop this concept further.
Restrictions on vocabulary assumed in these requirements are derived
from discussions during this BoF, based on experience with previous
endpoint-to-middle and middle-to- endpoint signaling approaches as
well as concerns about the privacy implications of endpoint-to-middle
signaling.

3. Terminology

This document uses the following terms:

o Superstrate: The transport protocol or protocol stack "above"
   SPUD, that uses SPUD for explicit path cooperation and path

traversal. The superstrate usually consists of a security layer
(e.g. TLS, DTLS) and a transport protocol, or a transport
protocol with integrated security features, to protect headers and
payload above SPUD.

o Endpoint: One end of a communication session, located on a single
   node that is a source or destination of packets in that session.
   In this document, this term may refer to either the SPUD
   implementation at the endpoint, the superstrate implementation
   running over SPUD, or the applications running over that
   superstrate.

o Path: The sequence of Internet Protocol nodes and links that a
   given packet traverses from endpoint to endpoint.

o Middlebox: As defined in [RFC3234], a middlebox is any
   intermediary device performing functions other than the normal,
   standard functions of an IP router on the datagram path between a
   source host and destination host; e.g. making decisions about
   forwarding behavior based on other than addressing information,
   and/or modifying a packet before forwarding.

4. Use Cases

   Use cases are outlined in more detail in
   [I-D.kuehlewind-spud-use-cases]. We summarize some of the primary
   use cases below.

   The primary use case for endpoint to path signaling in the Internet
   making use of packet grouping, as described in the use case document,
   is the binding of limited related semantics (start, ack, and stop) to
   a flow or a group of packets within a flow that are semantically
   related in terms of the application or superstrate. By explicitly
   signaling start and stop semantics, a flow allows middleboxes to use
   those signals for setting up and tearing down their relevant state
   (NAT bindings, firewall pinholes), rather than requiring the
   middlebox to infer this state from continued traffic. At best, this
   would allow the application to reduce heartbeat traffic, which might
   result in reduced radio utilization and thus greater battery life on
   mobile platforms.

   SPUD could also be used to provide information relevant for network
   treatment for middleboxes as a replacement for deep packet inspection
   for traffic classification purposes, rendered ineffective by
   superstrate encryption. In this application, properties would be
   expressed in terms of network-relevant parameters (intended
   bandwidth, latency and loss sensitivity, etc.) as opposed to
   application-relevant semantics. See

[I-D.trammell-stackevo-explicit-coop] for discussion on limitations in signaling in untrusted environments.

SPUD may also provide some facility for SPUD-aware nodes on the path to signal some property of the path to the endpoints and other SPUD-aware nodes on the path. The primary use case for path to application signaling is parallel to the use of ICMP [RFC0792] and ICMPv6 [RFC4443], in that it describes a set of conditions (including errors) that applies to the datagrams as they traverse the path. Since the signals here would traverse NATs in the same way as the traffic related to them, this use case would sidestep problems with ICMP availability in the deployed Internet.

Link-layer characteristics of use to the transport layer (e.g., whether a high-transient-delay, highly-buffered link such as LTE is present on the path) could also be signaled using this path-to-endpoint facility.

5. Functional Requirements

The following requirements detail the services that SPUD must provide to superstrates, endpoints, and middleboxes using SPUD.

5.1. Grouping of Packets (into "tubes")

Transport semantics and many properties of communication that endpoints may want to expose to middleboxes are bound to flows or groups of flows (5-tuples). SPUD must therefore provide a basic facility for associating packets together (into what we call a "tube", for lack of a better term) and associate information to these groups of packets. Each packet in a SPUD "flow" (determined by 5-tuple) belongs to exactly one tube. Notionally, a tube consists of a set of packets with a set of common properties, that should therefore receive equivalent treatment from the network; these tubes may or may not be related to separate semantic entities in the superstrate (e.g. SCTP streams), at the superstrate's discretion.

The simplest mechanisms for association involve the addition of an identifier to each packet in a tube. Other mechanisms that don't directly encode the identifier in a packet header, but instead provide it in a way that it is simple to derive from other information available in the packet at the endpoints and along the path, are also possible. In any cases, for the purposes of this requirement we treat this identifier as a simple vector of N bits. The properties of the tube identifier are subject to tradeoffs on the requirements for privacy, security, ease of implementation, and header overhead efficiency.

In determining the optimal size and scope for this tube identifier, we first assume that the 5-tuple of source and destination IP address, UDP port, and IP protocol identifier (17 for UDP) is used in the Internet as an existing flow identifier, due to the widespread deployment of network address and port translation. We conclude that SPUD tube IDs should be scoped to this 5-tuple.

While a globally-unique identifier would allow easier state comparison and migration for mobility use cases, it would have two serious disadvantages. First, N would need to be sufficiently large to minimize the probability of collision among multiple tubes having the same identifier along the same path during some period of time. A 128-bit UUID [RFC4122] or an identifier of equivalent size generated using an equivalent algorithm would probably be sufficient, at the cost of 128 bits of header space in every packet. Second, globally unique tube identifiers would also introduce new possibilities for user and node tracking, with a serious negative impact on privacy. We note that global identifiers for mobility, when necessary to expose to the path, can be supported separately from the tube identification mechanism, by using a generic tube-grouping application-to-path signaling bound to the tube.

Even when tube IDs are scoped to 5-tuples, N must still be sufficiently large, and the bits in the identifier sufficiently random, that possession of a valid tube ID implies that a node can observe packets belonging to the tube. This reduces the chances of success of blind packet injection attacks of packets with guessed valid tube IDs.

5.2. Bidirectionality of Tubes

When scoped to 5-tuples, the forward and backward directions of a bidirectional connection will have different tube IDs, since these will necessarily take different paths and may interact with a different set of middleboxes due to asymmetric routing. SPUD will therefore require some facility to note that one tube is the "reverse" direction of another, a general case of the tube grouping signal above.

5.3. Signaling of Per-Tube Properties

SPUD must be able to provide information scoped to a tube from the end- point(s) to all SPUD-aware nodes on the path about the packets in that tube.

We note that in-band signaling would meet this requirement.

5.4. Path to Receiver Signaling under Sender Control

   SPUD must be able to provide information about from a SPUD-aware
   middlebox to the endpoint. This information is associated with a
   tube, in terms of "the properties of the path(s) the packets in this
   tube will traverse". This signaling must happen only with explicit
   sender permission and be sent to the receiver of packets in the tube.

   We note that in-band signaling would meet this requirement, if the
   sender created a "placeholder" in-band that could be filled in by the
   middlebox(es) on path. In-band signaling has the advantage that it
   does not require foreknowledge of the identity and addresses of
   devices along the path by endpoints and vice versa, but does add
   complexity to the signaling protocol. Piggybacked signaling uses
   some number of bits in each packet generated by the overlying
   transport. It requires either reducing the MTU available to the
   encapsulated transport and/or opportunistically using "headroom" as
   it is available: bits between the network-layer MTU and the bits
   actually used by the transport. For use cases that accumulate
   information from devices on path in the SPUD header, piggybacked
   signaling also requires a mechanism for endpoints to create "scratch
   space" for potential use of the on-path devices.

   In contrast, interleaved signaling uses signaling packets on the same
   5-tuple and tube ID, which don't carry any superstrate data. These
   interleaved packets could also contain scratch space for on-path
   device use. This reduces complexity and sidesteps MTU problems, at
   the cost of sending more packets per flow.

5.5. Receiver to Sender Feedback

   SPUD must be able send information collected from SPUD-aware
   middleboxes along the path to a receiver back to the sender that gave
   permission; see Section 6.4 for restrictions on this facility.

5.6. Direct Path to Sender Signaling

   SPUD must provide a facility for a middlebox to send a packet
   directly in response to a sending endpoint, primarily to signal error
   conditions (e.g. "packet administratively prohibited" or "no route
   to destination", as in present ICMP).

   In this case, the direct return packet generated by the middlebox
   uses the reversed end-to-end 5-tuple in order to receive equivalent
   NAT treatment, though the reverse path might not be the same as the
   forward path. Endpoints have control over this feature: A SPUD-aware
   middlebox must not emit a direct return packet unless it is in direct
   response to a packet from a sending endpoint, and must not forward a

packet for which it has sent a direct return packet; see Section 6.6
and Section 7.9.

5.7. Tube Start and End Signaling

SPUD must provide a facility for endpoints to signal that a tube has
started, that the start of the tube has been acknowledged and
accepted by the remote endpoint(s), and that a tube has ended and its
state can be forgotten by the path. Given unreliable signaling (see
Section 7.10), both endpoints and devices on the path must be
resilient to the loss of any of these signals. Specifically,
timeouts are still necessary to clean up stale state.

5.8. Transport Semantic Signaling

Similar to tube start and end signaling, SPUD must provide a facility
for endpoints to signal that a superstrate transport session has been
requested, set up, and/or torn down. This facility provides an
explicit replacement for the common practice in TCP-aware middleboxes
of modeling TCP state of flows by inspecting the TCP flags byte.

Given the fact that a superstrate transport session may consist of
multiple tubes, this signaling must be separate from that for tube
start and end.

5.9. Declarative signaling

All information signaled via SPUD is defined to be declarative (as
opposed to imperative). A SPUD endpoint must function correctly even
no middlebox along the path understands the signals it sends, or if
sent signals from middleboxes it does not understand. It must also
function correctly if the path (and thereby the set of middleboxes
traversed) changes during the lifetime of a tube; endpoints cannot
rely on the creation or maintenance of state even on cooperative
middleboxes. Likewise, a SPUD-aware middlebox must function
correctly if sent signals from endpoints it does not understand, or
in the absence of expected signals from endpoints.

The declarative nature of this signaling removes any requirement that
SPUD provide reliability for its signals.

5.10. Extensibility

SPUD must enable multiple new transport semantics and application/
path declarations without requiring updates to SPUD implementations
in middleboxes.

   The use of SPUD for experimental signaling must be possible either
   without the registration of codepoints or namespaces with IANA, or
   with trivially easy (First Come, First Served [RFC5226] registration
   of such codepoints.

5.11. Common Vocabulary

   For the interoperability of SPUD endpoints and middleboxes with each
   other, the use of SPUD for standard signaling must use a common
   vocabulary with registered codepoints allocated under relatively
   restrictive policy. This restrictive policy serves primarily
   security and privacy goals (i.e., reducing the risk of misuse of the
   extensibility provided by the protocol).

   We note that an IANA registry requiring Standards Action {RFC5226}}
   to modify would meet this requirement.

5.12. Additional Per-Packet Signaling

   SPUD may provide a facility for signaling semantically simple
   information (similar to tube start and end) on a per-packet as
   opposed to a per-tube basis. Properties signaled per packet reduce
   state requirements at middleboxes, but also increase per-packet
   overhead. Small signal size (in bits of entropy) and encoding
   efficiency (in bits on the wire) is therefore more important for per-
   packet signaling that per-tube signaling. If per-packet signals need
   to be used by multiple hops along a path, these will need to be
   encoded in an efficiently-implementable way (i.e., using fixed-
   length, constant-offset data structures).

   Given these constraints, per-packet signaling is necessary for
   certain use cases, it is likely that SPUD will provide a very limited
   set of per-packet signals using flags in a SPUD header, and require
   all more complex properties to be bound per-tube.

6. Security Requirements

6.1. Privacy

   SPUD must allow endpoints to control the amount of information
   exposed to middleboxes, with the default being the minimum necessary
   for correct functioning. This includes the cryptographic protection
   of transport layer headers from inspection by devices on path, in
   order to prevent ossification of these headers.

6.2. Authentication

   The basic SPUD protocol must not require any authentication or a
   priori trust relationship between endpoints and middleboxes to
   function. However, SPUD should interoperate with the presentation/
   exchange of authentication information in environments where a trust
   relationship already exists, or can be easily established, either in-
   band or out-of-band, and use this information where possible and
   appropriate.

   Given the advisory nature of the signaling it supports, SPUD may also
   support eventual authentication: authentication of a signal after the
   reception of a packet after that containing the signal.

6.3. Integrity

   SPUD must be able to provide integrity protection of information
   exposed by endpoints in SPUD-encapsulated packets, though the details
   of this integrity protection are still open.

   Endpoints should be able to detect changes to headers SPUD uses for
   its own signaling (whether due to error, accidental modification, or
   malicious modification), as well as the injection of packets into a
   SPUD flow (defined by 5-tuple) or tube by nodes other than the remote
   endpoints. Errors and accidental modifications can be detected using
   a simple checksum over the SPUD header, while detecting malicious
   modifications requires cryptographic integrity protection. Similar
   to Section 6.2, cryptographic integrity protection may also be
   eventual.

   Integrity protection of the superstrate is left up to the
   superstrate.

6.4. Encrypted Feedback

   As feedback from a receiver to a sender (see Section 5.5) does not
   need to be exposed to the path, this feedback channel should be
   encrypted for confidentiality and authenticity, when available (see
   Section 6.2). This facility will rely on cooperation with the
   superstrate or some other out-of-band mechanism to provide these
   guarantees.

6.5. Preservation of Security Properties

   The use of SPUD must not weaken the essential security properties of
   the superstrate: confidentiality, integrity, authenticity, and
   defense against linkability. If the superstrate includes payload
   encryption for confidentiality, for example, the use of SPUD must not

allow deep packet inspection systems to have access to the plaintext.
Likewise, the use of SPUD must not create additional opportunities
for linkability not already existing in the superstrate.

6.6. Protection against trivial abuse

Malicious background traffic is a serious problem for UDP-based
protocols due to the ease of forging source addresses in UDP together
with only limited deployment of network egress filtering [RFC2827].
Trivial abuse includes flooding and state exhaustion attacks, as well
as reflection and amplification attacks. SPUD must provide minimal
protection against this trivial abuse. This implies that SPUD should
provide:

o a proof of return routability, that the endpoint identified by a
   packet's source address receives packets sent to that address;

o a feedback channel between endpoints;

o a method to probabilistically discriminiate legitimate SPUD
   traffic from reflected malicious traffic;

o a method to probabilistically discriminate SPUD traffic from on-
   path devices from devices off-path; and

o the ability to deploy mechanisms to protect against state
   exhaustion and other denial-of-service attacks against SPUD
   itself.

We note that using a "magic number" or other pattern of bits in an
encapsulation-layer header not used in any widely deployed protocol
has the nice property that no existing node in the Internet can be
induced to reflect traffic containing it. This allows the magic
number to provide probabilistic assurance that a given packet is not
reflected, assisting in meeting this requirement.

If SPUD is implemented over UDP, see [I-D.ietf-tsvwg-rfc5405bis] for
guidelines on the safe usage of UDP in the Internet, which addresses
some of these issues.

6.7. Continuum of trust among endpoints and middleboxes

There are different security considerations for different security
contexts. The end-to-end context is one; anything that only needs to
be seen by the path shouldn't be exposed in SPUD, but rather by the
superstrate. There are multiple different types of end-to-middle
context based on levels of trust between end and middle - is the
middlebox on the same network as the endpoint, under control of the

same owner? Is there some contract between the application user and
the middlebox operator? SPUD should support different levels of
trust than the default ("untrusted, but presumed honest due to
limitations on the signaling vocabulary") and fully-authenticated;
how these points along the continuum are to be implemented and how
they relate to each other needs to be explored further.

In the Internet, it is not in the general case possible for the
endpoint to authenticate every middlebox that might see packets it
sends and receives. In this case information produced by middleboxes
may enjoy less integrity protection than that produced by endpoints.
In addition, endpoint authentication of middleboxes and vice-versa
may be better conducted out-of- band (treating the middlebox as an
endpoint for the authentication protocol) than in-band (treating the
middlebox as a participant in a 3+ party communication).

7. Technical Requirements

   The following requirements detail the constraints on how the SPUD
   facility must meet its functional requirements.

7.1. Middlebox Traversal

   SPUD, including all path-to-endpoint and endpoint-to-path signaling
   as well as superstrate and superstrate payload, should be able to
   traverse existing middleboxes and firewalls, including those that are
   not SPUD-aware. Therefore SPUD must be encapsulated in a transport
   protocol that is known to be accepted on a large fraction of paths in
   the Internet, or implement some form of probing to determine in
   advance which transport protocols will be accepted on a certain path.
   This encapsulation will require port numbers to support endpoints
   connected via network address and port translation (NAPT). We note
   that UDP encapsulation would meet these requirements.

7.2. Low Overhead in Network Processing

   SPUD must be desgined to have low overhead, specifically requiring
   very little effort to recognize that a packet is a SPUD packet and to
   determine the tube it is associated with. We note that a magic
   number as in
   Section 6.6 would also have a low probability of colliding with any
   non-SPUD traffic, therefore meeting the recognition requirement.
   Tube identifiers appearing directly in the encapsulation-layer header
   would meet the tube association requirement.

7.3. Implementability in User-Space

To enable fast deployment SPUD and superstrates must be implementable
without requiring kernel replacements or modules on the endpoints,
and without having special privilege (such as is required for raw
packet transmission, i.e. root or "jailbreak") on the endpoints.

We note here that UDP would meet this requirement, as nearly all
operating systems and application development platforms allow a
userspace application to open UDP sockets.

We additionally note that while TCP APIs are also widely available to
userspace applications, they are bound to TCP transport semantics,
and generally do not provide enough control over segmentation and
transmission to successfully implement superstrate transports.

7.4. Incremental Deployability

SPUD must be designed to operate in the present Internet, and must be
designed to encourage incremental deployment.

As endpoint implementations can change more quickly than middleboxes
can be designed and deployed, a SPUD facility that was be useful
between endpoints even before the deployment of middleboxes that
understand it would stimulate deployment. The information exposed
over SPUD must provide incentives for adoption by both endpoints and
middleboxes.

SPUD must not be designed in such a way that precludes its
deployability in multipath, multicast, and/or endpoint multi-homing
environments.

7.5. No unnecessary restrictions on the superstrate

Beyond those restrictions deemed necessary as common features of any
secure, responsible transport protocol (see Section 6.6), SPUD must
impose only minimal restrictions on the transport protocols it
encapsulates. However, to serve as a substrate, it is necessary to
factor out the information that middleboxes commonly rely on and
endpoints are commonly willing to expose. This information should be
included in SPUD, and might itself impose additional restrictions to
the superstrate.

7.6. Minimal additional start-up latency

SPUD should not introduce additional start-up latency for
superstrates. Specifically, superstrates which can send data on an
initial packet must be able to do so when encapsulated within SPUD.

7.7. Minimal header overhead

   To avoid reducing network performance, the information and coding
   used in SPUD should be designed to use the minimum necessary amount
   of additional space in encapsulation headers.

7.8. Minimal non-productive traffic

   SPUD should minimize additional non-productive traffic (e.g.
   keepalives), and should provide mechanisms to allow its superstrates
   to minimize their reliance on non-productive traffic.

7.9. Endpoint Control

   Both endpoint-to-path and path-to-endpoint signaling happen
   completely under endpoint control.

7.10. On Reliability, Fragmentation, MTU, and Duplication

   As any information provided by SPUD is anyway opportunistic, SPUD
   need not provide reliable signaling for the information associated
   with a tube. Signals must be idempotent; all middleboxes and
   endpoints must gracefully handle receiving duplicate signal
   information. SPUD must continue working in the presence of IPv4
   fragmentation on path, but in order to reduce the impact of requiring
   fragments reassembly at middleboxes for signals to be intelligible,
   endpoints using SPUD should attempt to fit all signals into single
   MTU-sized packets.

   Given the importance of good path MTU information to SPUD's own
   signaling, SPUD should implement packetization layer path MTU
   discovery [RFC4821].

   Any facilities requiring more than an MTU's worth of data in a single
   signal should use an out-of-band method which does provide
   reliability - this method may be an existing transport or
   superstrate/SPUD combination, or a "minimal transport" defined by
   SPUD for its own use.

7.11. SPUD Support Discovery

   If SPUD is not usable on a path to an endpoint, a SPUD sender needs
   to be able to fall back to some other approach to achieve the goals
   of the superstrate; a SPUD endpoint must be able to easily determine
   whether a remote endpoint with which it wants to communicate using
   SPUD as a substrate can support SPUD, and whether path to the remote
   endpoint as well as the return path from the remote endpoint will
   pass SPUD packets.

It is not clear whether this is a requirement of SPUD, or a
requirement of the superstrate / application over SPUD.

8. Security Considerations

The security-relevant requirements for SPUD are outlined in
Section 6. These will be further addressed in protocol definition
work following from these requirements.

9. IANA Considerations

This document has no actions for IANA.

10. Contributors

In addition to the editors, this document is the work of David Black,
Ken Calvert, Ted Hardie, Joe Hildebrand, Jana Iyengar, and Eric
Rescorla.

11. Acknowledgments

Thanks to Ozgu Alay, Roland Bless, Cameron Byrne, Toerless Eckert,
Gorry Fairhurst, Daniel Kahn Gillmor, Tom Herbert, Christian Huitema,
Iain Learmonth, Diego Lopez, and Matteo Varvello for feedback and
comments on these requirements, as well as to the participants at the
SPUD BoF at IETF 92 meeting in Dallas and the IAB SEMI workshop in
Zurich for the discussions leading to this work.

This work is supported by the European Commission under Horizon 2020
grant agreement no. 688421 Measurement and Architecture for a
Middleboxed Internet (MAMI), and by the Swiss State Secretariat for
Education, Research, and Innovation under contract no. 15.0268. This
support does not imply endorsement.

12. Informative References

   [RFC0792] Postel, J., "Internet Control Message Protocol", STD 5,
              RFC 792, DOI 10.17487/RFC0792, September 1981,
              <http://www.rfc-editor.org/info/rfc792>.

   [RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering:
              Defeating Denial of Service Attacks which employ IP Source
              Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827,
              May 2000, <http://www.rfc-editor.org/info/rfc2827>.

   [RFC3234] Carpenter, B. and S. Brim, "Middleboxes: Taxonomy and
              Issues", RFC 3234, DOI 10.17487/RFC3234, February 2002,
              <http://www.rfc-editor.org/info/rfc3234>.

   [RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally
              Unique IDentifier (UUID) URN Namespace", RFC 4122,
              DOI 10.17487/RFC4122, July 2005,
              <http://www.rfc-editor.org/info/rfc4122>.

   [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet
              Control Message Protocol (ICMPv6) for the Internet
              Protocol Version 6 (IPv6) Specification", RFC 4443,
              DOI 10.17487/RFC4443, March 2006,
              <http://www.rfc-editor.org/info/rfc4443>.

   [RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU
              Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007,
              <http://www.rfc-editor.org/info/rfc4821>.

   [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an
              IANA Considerations Section in RFCs", BCP 26, RFC 5226,
              DOI 10.17487/RFC5226, May 2008,
              <http://www.rfc-editor.org/info/rfc5226>.

   [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer
              Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347,
              January 2012, <http://www.rfc-editor.org/info/rfc6347>.

   [RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black,
              "Encapsulating MPLS in UDP", RFC 7510,
              DOI 10.17487/RFC7510, April 2015,
              <http://www.rfc-editor.org/info/rfc7510>.

   [RFC7663] Trammell, B., Ed. and M. Kuehlewind, Ed., "Report from the
              IAB Workshop on Stack Evolution in a Middlebox Internet
              (SEMI)", RFC 7663, DOI 10.17487/RFC7663, October 2015,
              <http://www.rfc-editor.org/info/rfc7663>.

   [I-D.ietf-tsvwg-rfc5405bis]
              Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage
              Guidelines", draft-ietf-tsvwg-rfc5405bis-11 (work in
              progress), April 2016.

   [I-D.kuehlewind-spud-use-cases]
              Kѐѡhlewind, M. and B. Trammell, "Use Cases for a
              Substrate Protocol for User Datagrams (SPUD)", draft-
              kuehlewind-spud-use-cases-01 (work in progress), March
              2016.

   [I-D.huitema-tls-dtls-as-subtransport]
              Huitema, C., Rescorla, E., and J. Jana, "DTLS as
              Subtransport protocol", draft-huitema-tls-dtls-as-
              subtransport-00 (work in progress), March 2015.

   [I-D.trammell-stackevo-explicit-coop]
              Trammell, B., "Architectural Considerations for Transport
              Evolution with Explicit Path Cooperation", draft-trammell-
              stackevo-explicit-coop-00 (work in progress), September
              2015.

Authors' Addresses

   Brian Trammell (editor)
   ETH Zurich
   Gloriastrasse 35
   8092 Zurich
   Switzerland

   Email: ietf@trammell.ch

   Mirja Kuehlewind (editor)
   ETH Zurich
   Gloriastrasse 35
   8092 Zurich
   Switzerland

   Email: mirja.kuehlewind@tik.ee.ethz.ch