

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 23, 2022

A. Clemm  
J. Strassner  
Futurewei  
J. Francois  
Inria  
October 20, 2021

High-Precision Service Metrics  
draft-csfx-ippm-hipmetrics-00

Abstract

This document defines a set of metrics for high-precision networking services. These metrics can be used to assess the service levels that are being delivered for a networking flow. Specifically, they can be used to determine the degree of compliance with which service levels are being delivered relative to service level objectives that were defined for the flow. The metrics can be used as part of flow records and/or accounting records. They can also be used to continuously monitor the quality with which high-precision networking service are being delivered.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 23, 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Key Words . . . . .	3
3. Definitions and Acronyms . . . . .	3
4. Metrics . . . . .	3
5. Discussion Items . . . . .	7
6. IANA Considerations . . . . .	7
7. Security Considerations . . . . .	7
8. Normative References . . . . .	8
Authors' Addresses . . . . .	9

## 1. Introduction

Many networking applications increasingly rely on high-precision networking services that have clearly defined service level objectives (SLOs), for example with regards to end-to-end latency. Applications requiring such services include industrial networks, for example cloud-based industrial controllers for precision machinery, vehicular applications, for example tele-driving in which a vehicle is remotely controlled by a human operators, or Augmented Reality / Virtual Reality (AR/VR) applications involving rendering of point clouds remotely. Many of those applications are not tolerant of degrading service levels. A slight miss in SLOs does not merely result in a slight deterioration of the Quality of Experience to end users, but may render the application inoperable. At the same time, many of those applications are mission critical, in which sudden failures can jeopardize safety or have other adverse consequences. However, clearly those applications represent significant business opportunity demanding dependable technical solutions.

Because of this, efforts such as Deterministic Networking (DetNet) [RFC8655] are attempting to create solutions in which clear bounds on parameters such as end-to-end latency and jitter can be defined in order to make service levels being delivered predictable and, ideally, deterministic. However, one area that has not kept pace concerns metrics that can account for service levels with which services are delivered, specifically the degree of precision for agreed-upon service level objectives. Such metrics, and the instrumentation to support them, are important for a number of purposes, including monitoring (to ensure that networking services

are performing according to their objectives) as well as accounting (to maintain a record of service levels actually delivered, important for monetization of such services as well as for triaging of problems).

The current state-of-the-art of such metrics includes (for example) interface metrics, useful to obtain data on traffic volume and behavior that can be observed at an interface [RFC2863] [RFC8343] but agnostic of actual end-to-end service levels and not specific to distinct flows. Flow records [RFC7011] [RFC7012] maintain statistics about flows, including flow volume and flow duration, but again contain very little information about end-to-end service levels, let alone whether the service levels delivered meet their targets, i.e. their associated SLOs.

This specification introduces a new set of metrics aimed at capturing end-to-end service levels for a flow, specifically the degree to which flows comply with the SLOs that are in effect.

It should be noted that at this point, the set of metrics proposed here is intended as a "starter set" that is intended to spark further discussion. Other metrics are certainly conceivable; we expect that the list of metrics will evolve over time as part of Working Group discussions.

## 2. Key Words

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 3. Definitions and Acronyms

MTBF: Mean Time Between Failures

SL: Service Level

SLA: Service Level Agreement

SLO: Service Level Objective

## 4. Metrics

The following section proposes a set of accounting metrics focus on end-to-end latency objectives. They indicate whether any violations of end-to-end latency occurred at the packet level. These metrics

are intended to be applied on a per-flow basis and are intended to assess the degree to which a flow's end-to-end service levels comply with the SLO in effect for that flow.

While the focus in this document concerns end-to-end latency objectives, analogous metrics could also be defined for other end-to-end service level parameters, such as loss (which is distinct from loss occurring at any one given interface) or delay variation.

- o Violated Packets. This indicates the number of packets for which a violation of a latency SLO occurred.
- o Violated Time Units (e.g. violated seconds, violated milliseconds). This indicates the number of time units during which one or more violations of SLOs were observed, regardless of how many violations took place during the same interval. This measure is useful in scenarios where bursts of violations might suddenly occur (e.g. due to temporary network congestion, during route convergence etc.) and the count of violated packets by itself might paint a misleading picture.

The following additional set of metrics may be useful in certain scenarios as well. However, their precise definition may be subject to policy and further discussion is needed:

- o Significantly Violated Packets. This indicates the number of packets for which a "significant" violation occurred, where "significant" implies an SLO that was not merely a near-miss but that missed the objective by a degree determined especially significant.
- o Significantly Violated Time Units (e.g. significantly violated seconds, significantly violated milliseconds). This indicates the number of time units during which any significant violation occurred.
- o Severely Violated Time Units (e.g. severely violated seconds, severely violated milliseconds). "Severe" here refers to the occurrence of multiple violations within the same time unit. The definition of "severe" may be subject to policy; it may also take into account the significance of the violations that occur.

Note that there is no definition of Severely Violated Packets. The term "severe" is used in conjunction with the occurrence of multiple violations related to multiple packets, not any one packet in isolation.

From these first-order metrics, second-order metrics can be defined that build on the first set of metrics. Some of these metrics are modeled after Mean Time Between Failure, or MTBF metrics - a "failure" in this context referring to a failure to deliver a packet according to its SLO.

- o Time since last violated time unit (i.e., since last violated ms, since last violated second). (This parameter is particularly useful for the monitoring of the current health.)
- o Packets since last violated packet. (This parameter is particularly useful for the monitoring of the current health.)
- o Mean time between violated time units (i.e. between violated milliseconds, between violated seconds). This refers to the arithmetic mean of time between violations such as violated time units.
- o Mean packets between violations. This refers to the arithmetic mean of the number of SLO-compliant packets between SLO violations. (Another variation of "MTBF" in a service setting.)

The same set of metrics can also be applied to significant violations, and to severe violations:

- o Time since last significantly violated time unit (i.e., since last significantly violated ms, since last significantly violated second).
- o Time since last severely violated time unit (i.e., since last severely violated ms, since last severely violated second).
- o Packets since last significantly violated packet.
- o Mean time between significantly violated time units (i.e. between significantly violated milliseconds, between significantly violated seconds).
- o Mean time between severely violated time units (i.e. between severely violated milliseconds, between severely violated seconds).
- o Mean packets between significant violations. This refers to the arithmetic mean of the number of SLO-compliant packets between significant SLO violations.

The next set of metrics puts the violations in relationship to non-violations. It is intended to provide an analogous measure to that

of availability, typically defined as the number of time units during which a system (or service) is unavailable divided by the total number of time units. In analogy, a time unit that is "violated" can be viewed as one in which a service is not available with the advertised precision:

- o Precision availability (of milliseconds, of seconds): the ratio between violated time units (seconds, milliseconds) and the total time units for the duration of the service.
- o Analogous metrics for precision availability re: severely violated time units, re: significantly violated time units.

It should be noted that certain Service Level Agreements may be statistical in nature, requiring the service levels of packets in a flow to adhere to certain distributions. For example, an SLA might state that any given SLO applies only to a certain percentage of packets, allowing for a certain amount of violations to take place. A "violated packet" in that case does not necessarily constitute an SLO violation. However, it is still useful to maintain those statistics, as the number of violated packets still matters when looked at in proportion to the total number of packets.

Along that vein, an SLA might establish an SLO of, say, end-to-end latency to not exceed 20ms for 99% of packets, to not exceed 25ms for 99.999% of packets, and to never exceed 30ms for anything beyond. In that case, any individual packet missing the 20 ms latency target cannot be considered an SLO violation in itself, but compliance with the SLO may need to be assessed after the fact.

To support statistical SLAs more directly, it is feasible to support additional metrics, such as metrics that represent histograms for service level parameters with buckets corresponding to individual service level objectives. For the example just given, a histogram for a given flow could be maintained with three buckets: one containing the count of packets within 20ms, a second with a count of packets between 20 and 25ms (or simply all within 25ms), a third with a count of packet between 25 and 30ms (or simply all packets within 30ms, and a fourth with a count of anything beyond (or simply a total count). Of course, the number of buckets and the boundaries between those buckets should correspond to the needs of the application respectively SLA, i.e. to the specific guarantees and SLOs that were provided. The definition of histogram metrics is for further study.

## 5. Discussion Items

The following is a list of items for which further discussion is needed as to whether they should be included in the scope of this specification:

- o A YANG data model
- o A set of IPFIX Information Elements
- o Statistical metrics: e.g. histograms/buckets
- o Policies regarding the definition of "significant" and "severe" violations
- o Additional second-order metrics, such as "longest disruption of service time" (measuring consecutive time units with violations)

## 6. IANA Considerations

TBD

## 7. Security Considerations

Instrumentation for metrics that are used to assess compliance with SLOs constitute an interesting target for an attacker. By interfering with the maintaining of such metrics, services could be falsely identified as being in compliance (when they are not), or vice-versa flagged as being non-compliant (when indeed they are). While this document does not specify how networks should be instrumented to maintain the identified metrics, such instrumentation needs to be properly secured to ensure accurate measurements and prohibit tampering with metrics being kept.

Where metrics are being defined relative to an SLO, the configuration of those SLOs needs to be properly secured. Likewise, where SLOs can be adjusted, it needs to be clear which particular SLO any given metrics instance refers to. The same service levels that constitute SLO violations for one flow, and that should be maintained as part of the "violated time units", "violated packets", and related metrics, may be perfectly compliant for another flow. Where it is not possible to properly tie together SLOs and violation metrics, it will be preferable to merely maintain statistics about service levels that were delivered (for example, overall histograms of end-to-end latency), without assessing which of these constitute violations.

By the same token, where the definition of what constitutes a "severe" violation or a "significant" violation depends on policy or

context, the configuration of such policy or context needs to be specially secured and the configuration of this policy be bound to the metrics being maintained. This way it will be clear which policy was in effect when those metrics were being assessed. An attacker that is able to tamper with such policies will render the corresponding metrics useless (in the best case) or misleading (in the worst case).

## 8. 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>>.
- [RFC2863] McCloghrie, K. and F. Kastenholtz, "The Interfaces Group MIB", RFC 2863, DOI 10.17487/RFC2863, June 2000, <<https://www.rfc-editor.org/info/rfc2863>>.
- [RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <<https://www.rfc-editor.org/info/rfc7011>>.
- [RFC7012] Claise, B., Ed. and B. Trammell, Ed., "Information Model for IP Flow Information Export (IPFIX)", RFC 7012, DOI 10.17487/RFC7012, September 2013, <<https://www.rfc-editor.org/info/rfc7012>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <<https://www.rfc-editor.org/info/rfc8343>>.
- [RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <<https://www.rfc-editor.org/info/rfc8655>>.



Authors' Addresses

Alexander Clemm  
Futurewei  
2330 Central Expressway  
Santa Clara CA 95050  
USA

Email: ludwig@clemm.org

John Strassner  
Futurewei  
2330 Central Expressway  
Santa Clara CA 95050  
USA

Email: strazpdj@gmail.com

Jerome Francois  
Inria  
615 Rue du Jardin Botanique  
Villers-les-Nancy 54600  
France

Email: jerome.francois@inria.fr

AVTCORE/MMUSIC Working Groups  
Internet-Draft  
Intended status: Standards Track  
Expires: 12 March 2022

S. Dawkins  
Tencent America LLC  
8 September 2021

SDP Offer/Answer for RTP using QUIC as Transport  
draft-dawkins-sdp-rtp-quic-00

Abstract

This document describes these new SDP "proto" attribute values: "QUIC", "QUIC/RTP/SAVP", "QUIC/RTP/AVPF", and "QUIC/RTP/SAVPF", and describes how SDP Offer/Answer can be used to set up an RTP connection using QUIC as a transport protocol.

These proto values are necessary to allow the use of QUIC as an underlying transport protocol for applications that commonly use SDP as a session signaling protocol to set up RTP connections with UDP as its underlying transport protocol, such as SIP and WebRTC.

Status of This Memo

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

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

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

This Internet-Draft will expire on 12 March 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components

extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
1.1. Notes for Readers . . . . .	3
1.2. Terminology . . . . .	3
1.3. Contribution and Discussion Venues for this draft. . . . .	3
1.4. Scope of this document . . . . .	3
1.5. Assumptions for this document . . . . .	3
2. Open Questions (probably not for this draft, but could have implications on SDP Offer/Answer) . . . . .	4
3. Identifiers and Attributes . . . . .	4
3.1. Protocol Identifiers . . . . .	4
3.1.1. The QUIC proto . . . . .	4
3.1.2. The QUIC/RTP/SAVP proto . . . . .	5
3.1.3. The QUIC/RTP/AVPF proto . . . . .	5
3.1.4. The QUIC/RTP/SAVPF proto . . . . .	6
3.2. A QUIC/RTP/AVPF Offer . . . . .	6
4. IANA Considerations . . . . .	8
4.1. Proto Registrations . . . . .	8
5. Security Considerations . . . . .	9
6. Acknowledgments . . . . .	9
7. References . . . . .	9
7.1. Normative References . . . . .	9
7.2. Informative References . . . . .	11
Author's Address . . . . .	11

## 1. Introduction

This document describes these new SDP "proto" attribute values: "QUIC", "QUIC/RTP/SAVP", "QUIC/RTP/AVPF", and "QUIC/RTP/SAVPF", and describes how SDP Offer/Answer ([RFC3264]) can be used to set up an RTP ([RFC3550]) connection using QUIC ([RFC9000]) as a transport protocol.

These proto values are necessary to allow the use of QUIC as an underlying transport protocol for applications that commonly use SDP as a session signaling protocol to set up RTP connections with UDP as its underlying transport protocol, such as SIP ([RFC3261]) and WebRTC ([RFC8825]).

### 1.1. Notes for Readers

This document is intended for publication as a standards-track RFC in the IETF stream, but has not been adopted by any IETF working group, and does not carry any special status within the IETF.

### 1.2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 ([RFC2119]) ([RFC8174]) when, and only when, they appear in all capitals, as shown here.

### 1.3. Contribution and Discussion Venues for this draft.

(Note to RFC Editor - if this document ever reaches you, please remove this section)

This document is under development in the Github repository at <https://github.com/SpencerDawkins/sdp-rtp-quic>.

Readers are invited to open issues and send pull requests with contributed text for this document, or to send them to the author via email.

### 1.4. Scope of this document

This document focuses on the IANA registration and description of the RTP sessions using SDP Offer/Answer, as would be the case for many current RTP applications in common use, such as SIP ([RFC3261]) and WebRTC ([RFC8825]).

This document is intended as complementary to [I-D.engelbart-rtp-over-quic], which largely focuses on RTP/RTCP encapsulation in QUIC datagrams, so that the SDP experts can focus on SDP offer/answer aspects, and the RTP experts can focus on RTP/RTCP encapsulation aspects.

### 1.5. Assumptions for this document

This document assumes that for RTP-over-QUIC, it is useful to register these AVP profiles using QUIC, in order to allow existing SIP and RTCWEB RTP applications to migrate more easily to QUIC:

- \* RTP/SAVP ("The Secure Real-time Transport Protocol (SRTP)", as defined in [RFC3711]).

- \* RTP/AVPF ("Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)'), as defined in [RFC4585].
- \* RTP/SAVPF ("Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)'), as defined in [RFC5124].

This document assumes that any implementation adding support for RTP-over-QUIC could reasonably add support for BUNDLE ([RFC8843]), "rtcp-mux" ([RFC5761]).

## 2. Open Questions (probably not for this draft, but could have implications on SDP Offer/Answer)

- \* RTP (and RTCP) headers and payloads will be entirely encrypted using QUIC ([RFC9000]), as secured by TLS 1.3 handshake ([RFC9001]), between QUIC endpoints. It's worth thinking more about how that maps onto expected deployment scenarios like centralized multiparty conferencing, and also whether WebRTC really requires SAVPF with double encryption (i.e. SRTP encryption, and then QUIC encryption). No opinions here yet, just noting the question for now.
- \* When QUIC establishes connections, it uses IP addresses but then expects applications to use connection IDs to refer to connections, even if the underlying IP addresses change because of NAT binding, and even if the QUIC implementation performs QUIC connection migration itself, so the underlying IP addresses change. RTP applications expect to use IP addresses, not QUIC connection IDs. Must we specify an RTP/RTCP adaptation layer, similar to [I-D.ietf-quic-http] for HTTP/3?

## 3. Identifiers and Attributes

As much as possible, these are reused from other specifications, with references to the original definitions.

### 3.1. Protocol Identifiers

#### 3.1.1. The QUIC proto

The 'QUIC' protocol identifier is similar to the 'UDP' and 'TCP' protocol identifiers in that it only describes the transport protocol, and not the upper-layer protocol.

An 'm' line that specifies 'QUIC' MUST further qualify the application-layer protocol using an fmt identifier, such as "QUIC/RTP/AVPF". Media described using an 'm' line containing the 'QUIC' protocol identifier are carried using QUIC ([RFC9000]).

The following is an update to the ABNF for an 'm' line, as specified by [RFC8866], that defines a new value for the QUIC protocol.

```
media-field =          %s"m" "=" media SP port \["/" integer\]
                      SP proto 1*(SP fmt) CRLF
```

```
m= line parameter      parameter value(s)
```

```
-----
<media>:                (unchanged from {{RFC8866}})
<proto>:                'QUIC'
<port>:                 UDP port number
<fmt>:                  (unchanged from {{RFC8866}})
```

### 3.1.2. The QUIC/RTP/SAVP proto

The following is an update to the ABNF for an 'm' line, as specified by [RFC8866], that defines a new value for the QUIC/RTP/SAVP protocol.

```
media-field =          %s"m" "=" media SP port \["/" integer\]
                      SP proto 1*(SP fmt) CRLF
```

```
m= line parameter      parameter value(s)
```

```
-----
<media>:                (unchanged from {{RFC8866}})
<proto>:                'QUIC/RTP/SAVP'
<port>:                 UDP port number
<fmt>:                  (unchanged from {{RFC8866}})
```

### 3.1.3. The QUIC/RTP/AVPF proto

The following is an update to the ABNF for an 'm' line, as specified by [RFC8866], that defines a new value for the QUIC/RTP/AVPF protocol.

```
media-field =          %s"m" "=" media SP port \["/" integer\]
                        SP proto 1*(SP fmt) CRLF
```

```
m= line parameter      parameter value(s)
-----
```

```
<media>:               (unchanged from {{RFC8866}})
<proto>:               'QUIC/RTP/AVPF'
<port>:                UDP port number
<fmt>:                 (unchanged from {{RFC8866}})
```

#### 3.1.4. The QUIC/RTP/SAVPF proto

The following is an update to the ABNF for an 'm' line, as specified by [RFC8866], that defines a new value for the QUIC/RTP/SAVPF protocol.

```
media-field =          %s"m" "=" media SP port \["/" integer\]
                        SP proto 1*(SP fmt) CRLF
```

```
m= line parameter      parameter value(s)
-----
```

```
<media>:               (unchanged from {{RFC8866}})
<proto>:               'QUIC/RTP/SAVPF'
<port>:                UDP port number
<fmt>:                 (unchanged from {{RFC8866}})
```

#### 3.2. A QUIC/RTP/AVPF Offer

A complete example of an SDP offer using QUIC/RTP/AVPF might look like:

SDP line	Notes
v=0	Same as [RFC8866]
o=jdoe 3724394400 3724394405 IN IP4 198.51.100.1	Same as [RFC8866]
s=Call to John Smith	Same as [RFC8866]
i=SDP Offer #1	Same as [RFC8866]
u=http://www.jdoe.example.com/ home.html	Same as [RFC8866]
e=Jane Doe jane@jdoe.example.com (mailto:jane@jdoe.example.com)	Same as [RFC8866]
p=+1 617 555-6011	Same as [RFC8866]
c=IN IP4 198.51.100.1	Same as [RFC8866]
t=0 0	Same as [RFC8866]
m=audio 49170 RTP/AVP 0	Same as [RFC8866]
m=audio 49180 RTP/AVP 0	Same as [RFC8866]
m=video 51372 QUIC/RTP/AVPF 99	QUIC transport
a=setup:passive	will wait for QUIC handshake (setup attribute from [RFC4145])
a=connection:new	don't want to reuse an existing QUIC connection (connection attribute from [RFC4145])
c=IN IP6 2001:db8::2	Same as [RFC8866]
a=rtpmap:99 h266/90000	H.266 VVC codec [I-D.ietf-avtcore-rtp-vvc]

Table 1



This example is largely based on an example appearing in [RFC8866], Section 5, but is using QUIC/RTP/AVPF to support a newer codec.

Because QUIC uses connections for both streams and datagrams, we are reusing two session- and media-level SDP attributes from [SDP-attribute-name] that were defined in [RFC4145] for use with TCP: setup and connection.

This example SDP offer might be included in a SIP Invite.

#### 4. IANA Considerations

This document registers these protocols in the proto registry ([SDP-parameters]).

- \* QUIC (Section 3.1.1)
- \* QUIC/RTP/SAVP (Section 3.1.2)
- \* QUIC/RTP/AVPF (Section 3.1.3)
- \* QUIC/RTP/SAVPF (Section 3.1.4)

##### 4.1. Proto Registrations

IANA is requested to add these protocols to the Session Description Protocol (SDP) Parameters proto registry ([SDP-parameters]).

Type	SDP Name	Reference
proto	QUIC	RFCXXXX
proto	QUIC/RTP/SAVP	RFCXXXX
proto	QUIC/RTP/AVPF	RFCXXXX
proto	QUIC/RTP/SAVPF	RFCXXXX

Table 2

\*Note to the RFC Editor\*

Please replace "RFCXXXX" with the assigned RFC number, when that is available, and remove this note.

## 5. Security Considerations

Security considerations for the QUIC protocol are described in the corresponding section in [RFC9000].

Security considerations for the TLS handshake used to secure QUIC are described in [RFC9001].

Security considerations for SDP are described in the corresponding section in [RFC8866].

Security considerations for SDP offer/answer are described in the corresponding section in [RFC3264].

## 6. Acknowledgments

My appreciation to the authors of [RFC4145], which served as a model for the initial structure of this document.

Your name could appear here. Please comment and contribute, as per Section 1.3.

## 7. References

### 7.1. 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://doi.org/10.17487/RFC2119>>.
- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <<https://doi.org/10.17487/RFC3261>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<https://doi.org/10.17487/RFC3264>>.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <<https://doi.org/10.17487/RFC3550>>.

- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<https://doi.org/10.17487/RFC3711>>.
- [RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, DOI 10.17487/RFC4585, July 2006, <<https://doi.org/10.17487/RFC4585>>.
- [RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <<https://doi.org/10.17487/RFC5124>>.
- [RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and Control Packets on a Single Port", RFC 5761, DOI 10.17487/RFC5761, April 2010, <<https://doi.org/10.17487/RFC5761>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://doi.org/10.17487/RFC8174>>.
- [RFC8825] Alvestrand, H., "Overview: Real-Time Protocols for Browser-Based Applications", RFC 8825, DOI 10.17487/RFC8825, January 2021, <<https://doi.org/10.17487/RFC8825>>.
- [RFC8843] Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", RFC 8843, DOI 10.17487/RFC8843, January 2021, <<https://doi.org/10.17487/RFC8843>>.
- [RFC8866] Begen, A., Kyzivat, P., Perkins, C., and M. Handley, "SDP: Session Description Protocol", RFC 8866, DOI 10.17487/RFC8866, January 2021, <<https://doi.org/10.17487/RFC8866>>.
- [RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://doi.org/10.17487/RFC9000>>.

[RFC9001] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <<https://doi.org/10.17487/RFC9001>>.

[SDP-attribute-name]  
"SDP Parameters - attribute-name", September 2021, <<https://www.iana.org/assignments/sdp-parameters/sdp-parameters.xhtml#sdp-att-field>>.

[SDP-parameters]  
"SDP Parameters - Proto", September 2021, <<https://www.iana.org/assignments/sdp-parameters/sdp-parameters.xhtml#sdp-parameters-2>>.

## 7.2. Informative References

[I-D.engelbart-rtp-over-quic]  
Ott, J. and M. Engelbart, "RTP over QUIC", Work in Progress, Internet-Draft, draft-engelbart-rtp-over-quic-00, 12 July 2021, <<https://datatracker.ietf.org/doc/html/draft-engelbart-rtp-over-quic-00>>.

[I-D.ietf-avtc core-rtp-vvc]  
Zhao, S., Wenger, S., Sanchez, Y., and Y. Wang, "RTP Payload Format for Versatile Video Coding (VVC)", Work in Progress, Internet-Draft, draft-ietf-avtc core-rtp-vvc-10, 9 July 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-avtc core-rtp-vvc-10>>.

[I-D.ietf-quic-http]  
Bishop, M., "Hypertext Transfer Protocol Version 3 (HTTP/3)", Work in Progress, Internet-Draft, draft-ietf-quic-http-34, 2 February 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-quic-http-34>>.

[RFC4145] Yon, D. and G. Camarillo, "TCP-Based Media Transport in the Session Description Protocol (SDP)", RFC 4145, DOI 10.17487/RFC4145, September 2005, <<https://doi.org/10.17487/RFC4145>>.

## Author's Address

Spencer Dawkins  
Tencent America LLC  
United States of America

Email: [spencerdawkins.ietf@gmail.com](mailto:spencerdawkins.ietf@gmail.com)

AVTCORE/MMUSIC Working Groups  
Internet-Draft  
Intended status: Informational  
Expires: 28 April 2022

S. Dawkins  
Tencent America LLC  
25 October 2021

SDP Offer/Answer for RTP using QUIC as Transport - Design Questions  
draft-dawkins-sdp-rtp-quic-questions-01

## Abstract

This document is a companion document to "SDP Offer/Answer for RTP using QUIC as Transport". That document focuses on the description and registration of SDP "proto" attribute parameters with IANA, to allow applications that rely on SDP Offer/Answer to negotiate the QUIC protocol as an encapsulation for RTP.

In writing that document, it became obvious that decisions about an appropriate SDP description would depend on decisions about the way RTP would be encapsulated in QUIC, and different proposals for those encapsulations had made different assumptions. Given that none of these proposals have been adopted by an IETF working group yet, it's not appropriate to try to base a general-purpose set of "QUIC/RTP" IANA registrations on any one of them, so this document includes the questions that have come up, and (as discussions progress) suggested answers for those questions.

## Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the "Media Over QUIC" non-working group mailing list (MOQ), which is archived at <https://mailarchive.ietf.org/arch/browse/moq/>. Subscription information is at <https://www.ietf.org/mailman/listinfo/Moq/>.

Source for this draft and an issue tracker can be found at <https://github.com/SpencerDawkins/sdp-rtp-quic-questions>.

## Status of This Memo

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

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

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

This Internet-Draft will expire on 28 April 2022.

#### Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

#### Table of Contents

1. Introduction . . . . .	3
1.1. Notes for Readers . . . . .	3
1.2. Scope of this document . . . . .	3
2. Questions (and, Eventually, Answers) . . . . .	4
2.1. Useful AVP Profiles . . . . .	4
2.1.1. Can We Assume the Use of "Extended Audiovisual Profiles"? . . . . .	4
2.1.2. Is "Secure RTP Encapsulated in UDP" Equivalent to "RTP Encapsulated in QUIC"? . . . . .	4
2.1.3. Encapsulations in Datagrams and Streams . . . . .	5
2.2. Useful Support For Existing RTP Extensions included in SDP Offer/Answer . . . . .	5
2.3. Feedback Mechanisms . . . . .	6
2.4. Potential Extensions To QUIC and QUIC-related Specifications . . . . .	7
2.4.1. QUIC Datagram Multiplexing . . . . .	7
2.4.2. RTP Destination Transport Addresses, Bundles, and QUIC Connection-IDs . . . . .	8
2.4.3. Support for NAT Traversal . . . . .	8
3. IANA Considerations . . . . .	9
4. Security Considerations . . . . .	9
5. Acknowledgments . . . . .	9
6. References . . . . .	10
6.1. Normative References . . . . .	10
6.2. Informative References . . . . .	12

Author's Address . . . . . 12

## 1. Introduction

This document is a companion document to "SDP Offer/Answer for RTP using QUIC as Transport" ([I-D.dawkins-sdp-rtp-quic]). That document focuses on the description and registration of SDP ([RFC8866]) "proto" attribute parameters with IANA ([SDP-parameters]), to allow applications that rely on SDP Offer/Answer ([RFC3264]) to negotiate the QUIC protocol ([RFC9000]) as an encapsulation for RTP ([RFC3550]).

In writing that document, it became obvious that decisions about an appropriate SDP description would depend on decisions about the way RTP would be encapsulated in QUIC, and different proposals for those encapsulations ([I-D.engelbart-rtp-over-quic], [I-D.hurst-quic-rtp-tunnelling], and [I-D.rtpfolks-quic-rtp-over-quic]) had made different assumptions. Given that none of these proposals have been adopted by an IETF working group yet, it's not appropriate to try to base a general-purpose set of "QUIC/RTP" IANA registrations on any one of them, so this document includes the questions that have come up, and (as discussions progress) suggested answers for those questions.

### 1.1. Notes for Readers

(Note to RFC Editor - if this document ever reaches you, please remove this section)

This document is intended to stimulate discussion about how proponents of "RTP over QUIC" expect that to work, recognizing that not everyone has the same goals in mind, but it understanding what the choices are will likely be helpful in making those choices, especially when the results of a choice provide direction that will allow implementers to agree on strategies and reuse as much code as possible.

### 1.2. Scope of this document

[I-D.dawkins-sdp-rtp-quic] will almost certainly reflect answers to the questions contained in this document, but the discussion material in this document will not be appropriate for inclusion in a draft that focuses on SDP description and IANA registration. This document might be worth publishing on its own, but is primarily intended to guide discussion that will feed into [I-D.dawkins-sdp-rtp-quic].

## 2. Questions (and, Eventually, Answers)

This version of this document is still very much a starting point for discussion, and additional questions are welcomed, even as we converge on answers.

### 2.1. Useful AVP Profiles

[SDP-parameters] contains four classes of AVP profiles:

- \* RTP/AVP ("RTP Profile for Audio and Video Conferences with Minimal Control"), as defined in [RFC3551],
- \* RTP/SAVP ("The Secure Real-time Transport Protocol (SRTP)", as defined in [RFC3711],
- \* RTP/AVPF ("Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)"), as defined in [RFC4585], and
- \* RTP/SAVPF ("Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)"), as defined in [RFC5124].

We could register all four over QUIC, but if we can cut down on the number of options for implementers, we might achieve better interoperability.

#### 2.1.1. Can We Assume the Use of "Extended Audiovisual Profiles"?

We could register both AVP and AVPF profiles, but do we need to register both?

#### 2.1.2. Is "Secure RTP Encapsulated in UDP" Equivalent to "RTP Encapsulated in QUIC"?

RTP that is encapsulated in QUIC payloads will always be encrypted [RFC9000]. So we could register (for example)

- \* QUIC/RTP/AVPF, knowing that any RTP payload using the QUIC protocol is encrypted, but is not encrypted using SDP, or
- \* QUIC/RTP/SAVPF, because QUIC encryption provides at least an equivalent level of protection to SDP, or
- \* both QUIC/RTP/AVPF and QUIC/RTP/SAVPF, to minimize the changes necessary for existing RTP applications to add support for QUIC encapsulation?



#### 2.1.2.1. Proposed Answer

We note that it is possible, and perhaps likely, that RTP-over-QUIC would be used in "mixed" environments, where one of the functions of an RTP mixer/translator is to connect RTP endpoints that are RTP-over-QUIC-enabled with RTP endpoints that are not.

In this case, the only way to ensure encryption over every hop between two endpoints is to use one of the RTP profiles that includes security.

#### 2.1.3. Encapsulations in Datagrams and Streams

We note that [SDP-parameters] contains registrations for both RTP encapsulated in UDP datagrams and RTP encapsulated in TCP streams.

If we wanted to allow the same level of flexibility for QUIC/RTP, we could register (for example) QUIC/DGRAM/RTP, mapped onto QUIC datagrams ([I-D.ietf-quic-datagram]), and QUIC/STREAM/RTP, mapped onto QUIC streams ([RFC9000]), reusing terminology from the Berkeley Sockets API.

Should we do that? If so, starting out that way would be better than starting out with QUIC/RTP and then adding QUIC/STREAM/RTP later.

#### 2.2. Useful Support For Existing RTP Extensions included in SDP Offer/Answer

At least one of the goals for QUIC/RTP encapsulation is that QUIC/RTP applications would not require more UDP ports than existing RTP applications. For this reason,

- \* It seems useful to confirm that we can assume support for "Multiplexing RTP Data and Control Packets on a Single Port", as described in [RFC5761].
- \* It seems useful to confirm that we can assume support for Media Multiplexing ("BUNDLE"), as described in [RFC8843].

*\*Editor's Note\** We recognize that the usage of RTP/RTCP ports in (for example) RTP/SAVPF, which runs over UDP, will be more nuanced in (for example) QUIC/RTP/SAVPF, which can use UDP ports in the same way as RTP/SAVPF, but can also use mechanisms such as Application-Layer Protocol Negotiation (ALPN) Protocol IDs to multiplex multiple applications on a single port, as further discussed in Section 2.4.2 below. This section should be read with Section 2.4.2 in mind. One possibility is that this discussion turns out to be about minimizing the need for more QUIC connections, which does translate directly to minimizing UDP ports.

Are there other RTP extensions that we should assume support for?

### 2.3. Feedback Mechanisms

RTP has relied on RTCP as its feedback mechanism for decades, as that mechanism has evolved over time, with the addition of AVPF feedback ([RFC4585]), and subsequent extensions (for example, the codec control messages defined in [RFC5104] and extended in [RFC7728] and [RFC8082]).

Should we assume that RTP applications using QUIC as their transport encapsulation will continue to use AVPF as the basis for feedback mechanisms, largely unchanged?

It seems likely that many implementations that already utilize (S)AVPF as their feedback mechanisms will continue to do so for QUIC/RTP/AVPF sessions. These implementations already work, and continuing to use the same feedback mechanism for QUC/RTP/AVPF sessions will minimize the amount of new development and testing required for these implementations to add support for QUIC/RTP/AVPF.

We also recognize that [RIST-Simple-Prof] extends the RTP/AVPF bitmasked-based retransmission request with its own range-based retransmission request, as an indication that this feedback mechanism remains in the mainstream of RTP/RTCP protocol usage.

However, [I-D.engelbart-rtp-over-quic] proposes that QUIC/RTP implementations may not need to support some RTCP messages, if QUIC itself provides equivalent functionality.

If there's not one answer to the feedback mechanism question, the necessary feedback mechanism will need to be included in the SDP Offer/Answer exchange.

## 2.4. Potential Extensions To QUIC and QUIC-related Specifications

Because the topics in this section are speculative, it's not clear whether they would have any impact on SDP description and IANA registration in [I-D.dawkins-sdp-rtp-quic]. They are included in this document for completeness.

### 2.4.1. QUIC Datagram Multiplexing

[I-D.ietf-quic-datagram] does not provide support for a standardized datagram multiplexing identifier, for reasons described in Section 5.1, and at greater length in the Github issue about this (at <https://github.com/quicwg/datagram/issues/6>).

The high-level explanation is that there was considerable support for adding a datagram multiplexing identifier to the datagram frame type, but much less agreement about what effect that identifier would have, and whether the QUIC transport layer would be responsible for any particular processing, or whether this processing would necessarily be performed by the application. For example, some proponents of adding datagram multiplexing expected to use multiplexing identifiers as ways of distinguishing between datagrams of different priorities, and other proponents expected to use multiplexing identifiers to distinguish between datagrams that were part of multi-datagram conversations (more like streams, but using the datagram frame types).

Given that there was no agreement on the functionality that datagram multiplexer identifiers would be used for, or what requirements this addition to the protocol would place to QUIC transport processing, the best decision for the QUIC protocol seemed to be that any application that needed this capability could trivially add its own multiplexing identifiers at the beginning of the Datagram Data field ([I-D.ietf-quic-datagram]).

In general, that's a fine plan. The question for this document is whether there is enough similarity among various applications using QUIC/RTP that specifying an RTP-specific datagram multiplier would provide better predictability and ability to multiplex datagrams from different applications without modifying those applications when they encounter each other for the first time.

#### 2.4.2. RTP Destination Transport Addresses, Bundles, and QUIC Connection-IDs

RTP has more than one way to identify endpoints, whether [RFC3550]-style destination three-tuples, or [RFC4961]-style Symmetric RTP and RTCP, or [RFC8843]-style BUNDLE transport, but all are based on IP addresses and port addresses.

The QUIC protocol also starts out with five-tuple awareness as it establishes a connection and performs TLS 1.3 handshake between the client and server, as described in [RFC9001], and performs path validation, as described in [RFC9000], but can also create multiple connection identifiers using different transport addresses that will be associated with the same connection, and when another path has been validated and associated with a known connection identifier, the QUIC endpoint can begin receiving packets for the same connection from an entirely different transport address, with no other signaling. This change of transport addresses might be the result of QUIC-level "connection migration", but might also be the result of NAT rebinding.

Applications using QUIC/RTP will need some way of managing QUIC connection identifiers, rather than transport addresses. This points to at least one potential need for a mapping of RTP semantics over QUIC, equivalent to [I-D.ietf-quic-http].

If QUIC/RTP applications will be making use of something like Application Layer Protocol Negotiation (ALPN) ([RFC7301]) identifiers to select QUIC/RTP processing, as HTTP/3 does, this may point to another potential need for a mapping.

#### 2.4.3. Support for NAT Traversal

It's worth noting that the driving use cases for the first version of the IETF QUIC protocol have been for HTTP-based web access, where the capability described in Section 8.2 of [RFC9000] was sufficient:

Path validation is not designed as a NAT traversal mechanism. Though the mechanism described here might be effective for the creation of NAT bindings that support NAT traversal, the expectation is that one endpoint is able to receive packets without first having sent a packet on that path. Effective NAT traversal needs additional synchronization mechanisms that are not provided here.

Some existing RTP applications share this characteristic - at least one RTP endpoint can receive a packet without having previously sent packet on that path. For these applications, current QUIC functionality will be sufficient.

For other RTP applications, we may need a QUIC extension that provides NAT traversal, and we may need to include information about NAT traversal in SDP Offer/Answer to enable QUIC/RTP communication.

### 3. IANA Considerations

This document makes no requests of IANA.

### 4. Security Considerations

This document is intended as the basis for discussion about protocol mechanisms that will be described in other documents. As a discussion paper, this document introduces no new security considerations, and any new security considerations resulting from those discussions should be included in the documents that actually describe protocol mechanisms.

### 5. Acknowledgments

Thanks to these folks, for authoring various "QUIC over RTP" drafts for specific use cases, which included their understanding of SDP aspects of their proposals:

- \* Joerg Ott, Roni Even, Colin Perkins, and Varun Singh, authors of [I-D.rtpfolks-quic-rtp-over-quic]
- \* Sam Hurst, author of [I-D.hurst-quic-rtp-tunnelling]
- \* Joerg Ott and Mathis Engelbart, authors of [I-D.engelbart-rtp-over-quic]

Thanks to these folks for helping to improve this draft by commenting, proposing text, or correcting confusion:

- \* Colin Perkins
- \* Richard Bradbury
- \* Stephan Wegner

(Your name also could appear here. Please comment and contribute, as described in "Contribution and Discussion Venues for this draft" above)

## 6. References

### 6.1. Normative References

- [I-D.engelbart-rtp-over-quic]  
Ott, J. and M. Engelbart, "RTP over QUIC", Work in Progress, Internet-Draft, draft-engelbart-rtp-over-quic-00, 12 July 2021, <<https://datatracker.ietf.org/doc/html/draft-engelbart-rtp-over-quic-00>>.
- [I-D.hurst-quic-rtp-tunnelling]  
Hurst, S., "QRT: QUIC RTP Tunnelling", Work in Progress, Internet-Draft, draft-hurst-quic-rtp-tunnelling-01, 28 January 2021, <<https://datatracker.ietf.org/doc/html/draft-hurst-quic-rtp-tunnelling-01>>.
- [I-D.ietf-quic-datagram]  
Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable Datagram Extension to QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-datagram-06, 5 October 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-quic-datagram-06>>.
- [I-D.ietf-quic-http]  
Bishop, M., "Hypertext Transfer Protocol Version 3 (HTTP/3)", Work in Progress, Internet-Draft, draft-ietf-quic-http-34, 2 February 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-quic-http-34>>.
- [I-D.rtpfolks-quic-rtp-over-quic]  
Ott, J., Even, R., Perkins, C., and V. Singh, "RTP over QUIC", Work in Progress, Internet-Draft, draft-rtpfolks-quic-rtp-over-quic-01, 1 September 2017, <<https://datatracker.ietf.org/doc/html/draft-rtpfolks-quic-rtp-over-quic-01>>.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <<https://www.rfc-editor.org/rfc/rfc3264>>.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <<https://www.rfc-editor.org/rfc/rfc3550>>.

- [RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, DOI 10.17487/RFC3551, July 2003, <<https://www.rfc-editor.org/rfc/rfc3551>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<https://www.rfc-editor.org/rfc/rfc3711>>.
- [RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, DOI 10.17487/RFC4585, July 2006, <<https://www.rfc-editor.org/rfc/rfc4585>>.
- [RFC4961] Wing, D., "Symmetric RTP / RTP Control Protocol (RTCP)", BCP 131, RFC 4961, DOI 10.17487/RFC4961, July 2007, <<https://www.rfc-editor.org/rfc/rfc4961>>.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<https://www.rfc-editor.org/rfc/rfc5104>>.
- [RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <<https://www.rfc-editor.org/rfc/rfc5124>>.
- [RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and Control Packets on a Single Port", RFC 5761, DOI 10.17487/RFC5761, April 2010, <<https://www.rfc-editor.org/rfc/rfc5761>>.
- [RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <<https://www.rfc-editor.org/rfc/rfc7301>>.
- [RFC7728] Burman, B., Akram, A., Even, R., and M. Westerlund, "RTP Stream Pause and Resume", RFC 7728, DOI 10.17487/RFC7728, February 2016, <<https://www.rfc-editor.org/rfc/rfc7728>>.

- [RFC8082] Wenger, S., Lennox, J., Burman, B., and M. Westerlund, "Using Codec Control Messages in the RTP Audio-Visual Profile with Feedback with Layered Codecs", RFC 8082, DOI 10.17487/RFC8082, March 2017, <<https://www.rfc-editor.org/rfc/rfc8082>>.
- [RFC8843] Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", RFC 8843, DOI 10.17487/RFC8843, January 2021, <<https://www.rfc-editor.org/rfc/rfc8843>>.
- [RFC8866] Begen, A., Kyzivat, P., Perkins, C., and M. Handley, "SDP: Session Description Protocol", RFC 8866, DOI 10.17487/RFC8866, January 2021, <<https://www.rfc-editor.org/rfc/rfc8866>>.
- [RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://www.rfc-editor.org/rfc/rfc9000>>.
- [RFC9001] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <<https://www.rfc-editor.org/rfc/rfc9001>>.
- [RIST-Simple-Prof]  
"Reliable Internet Stream Transport (RIST) Protocol Specification Simple Profile", September 2021, <[https://vsf.tv/download/technical\\_recommendations/VSF\\_TR-06-1\\_2020\\_06\\_25.pdf](https://vsf.tv/download/technical_recommendations/VSF_TR-06-1_2020_06_25.pdf)>.
- [SDP-parameters]  
"SDP Parameters - Proto", September 2021, <<https://www.iana.org/assignments/sdp-parameters/sdp-parameters.xhtml#sdp-parameters-2>>.

## 6.2. Informative References

- [I-D.dawkins-sdp-rtp-quic]  
Dawkins, S., "SDP Offer/Answer for RTP using QUIC as Transport", Work in Progress, Internet-Draft, draft-dawkins-sdp-rtp-quic-00, 8 September 2021, <<https://datatracker.ietf.org/doc/html/draft-dawkins-sdp-rtp-quic-00>>.

Author's Address



Spencer Dawkins  
Tencent America LLC  
United States of America

Email: [spencerdawkins.ietf@gmail.com](mailto:spencerdawkins.ietf@gmail.com)

MOQ Mailing List  
Internet-Draft  
Intended status: Informational  
Expires: 8 September 2022

J. Guessing  
Nederlandse Publieke Omroep  
S. Dawkins  
Tencent America LLC  
7 March 2022

Media Over QUIC - Use Cases and Considerations for Media Transport  
Protocol Design  
draft-guessing-moq-requirements-01

Abstract

This document describes use cases that have been discussed in the IETF community under the banner of "Media Over QUIC", provides analysis about those use cases, recommends a subset of use cases that cover live media ingest, syndication, and streaming for further exploration, and describes considerations that should guide the design of protocols to satisfy these use cases.

Note to Readers

\_RFC Editor: please remove this section before publication\_

Source code and issues for this draft can be found at  
<https://github.com/fiestajetsam/draft-guessing-moq-requirements>  
(<https://github.com/fiestajetsam/draft-guessing-moq-requirements>).

Discussion of this draft should take place on the IETF Media Over QUIC (MOQ) mailing list, at <https://www.ietf.org/mailman/listinfo/moq>  
(<https://www.ietf.org/mailman/listinfo/moq>).

Status of This Memo

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

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

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

This Internet-Draft will expire on 8 September 2022.

## Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

## Table of Contents

1. Introduction . . . . .	3
1.1. For The Impatient Reader . . . . .	3
1.2. Why QUIC For Media? . . . . .	3
2. Terminology . . . . .	4
2.1. The Many Meanings of "Media Over QUIC" . . . . .	5
2.2. Media Transport Protocol . . . . .	5
2.3. Latency Requirement Categories . . . . .	6
3. Prior and Existing Specifications . . . . .	7
3.1. QRT: QUIC RTP Tunnelling . . . . .	7
3.2. RTP over QUIC . . . . .	7
3.3. RUSH - Reliable (unreliable) streaming protocol . . . . .	7
3.4. Tunnelling SRT over QUIC . . . . .	7
3.5. Warp - Segmented Live Video Transport . . . . .	8
3.6. Comparison of Existing Specifications . . . . .	8
4. Use Cases Informing This Proposal . . . . .	8
4.1. Interactive Media . . . . .	9
4.1.1. Gaming . . . . .	10
4.1.2. Remote Desktop . . . . .	10
4.1.3. Video Conferencing/Telephony . . . . .	10
4.2. Live Media . . . . .	11
4.2.1. Live Media Ingest . . . . .	11
4.2.2. Live Media Syndication . . . . .	12
4.2.3. Live Media Streaming . . . . .	12
4.3. On-Demand Media . . . . .	13
4.3.1. On-Demand Ingest . . . . .	13
4.3.2. On-Demand Media Streaming . . . . .	13
5. Proposed Scope for "Media Over QUIC" . . . . .	14
5.1. Analysis for Interactive Use Cases . . . . .	14
5.2. Analysis for Live Media Use Cases . . . . .	14
5.3. Analysis for On-Demand Use Cases . . . . .	14
6. Considerations for Protocol Work . . . . .	15
6.1. Here Be Dragons . . . . .	15

6.2.	Codec Agility . . . . .	15
6.3.	Support an Appropriate Range of Latencies . . . . .	16
6.4.	Migration of Sessions . . . . .	16
6.5.	Appropriate Congestion Control . . . . .	16
6.6.	Support Lossless and Lossy Media Transport . . . . .	17
6.7.	Flow Directionality . . . . .	17
6.8.	WebTransport . . . . .	17
6.9.	Authentication . . . . .	17
6.10.	Considerations Implying QUIC Extensions . . . . .	17
6.10.1.	NAT Traversal . . . . .	18
6.10.2.	Multicast . . . . .	18
7.	IANA Considerations . . . . .	18
8.	Security Considerations . . . . .	18
9.	Informative References . . . . .	18
Appendix A.	Acknowledgements . . . . .	22
Authors' Addresses	. . . . .	23

## 1. Introduction

This document describes use cases that have been discussed in the IETF community under the banner of "Media Over QUIC", provides analysis about those use cases, recommends a subset of use cases that cover live media ingest, syndication, and streaming for further exploration, and describes considerations that should guide the design of protocols to satisfy these use cases.

### 1.1. For The Impatient Reader

- \* Our proposal is to focus on live media use cases, as described in Section 5, rather than on interactive media use cases or on-demand use cases.
- \* The reasoning behind this proposal can be found in Section 5.1.
- \* The considerations for protocol work to satisfy the proposed use cases can be found in Section 6.

Most of the rest of this document provides background for these sections.

### 1.2. Why QUIC For Media?

It is not the purpose of this document to argue against proposals for work on media applications that do not involve QUIC. Such proposals are simply out of scope for this document.

When work on the QUIC protocol ([RFC9000]) was chartered ([QUIC-goals]), the key goals for QUIC were:

- \* Minimizing connection establishment and overall transport latency for applications, starting with HTTP,
- \* Providing multiplexing without head-of-line blocking,
- \* Requiring only changes to path endpoints to enable deployment,
- \* Enabling multipath and forward error correction extensions, and
- \* Providing always-secure transport, using TLS 1.3 by default.

These goals were chosen with HTTP ([I-D.draft-ietf-quic-http]) in mind.

While work on "QUIC version 1" (version codepoint 0x00000001) was underway, protocol designers considered potential advantages of the QUIC protocol for other applications. In addition to the key goals for HTTP applications, these advantages were immediately apparent for at least some media applications:

- \* QUIC endpoints can create bidirectional or unidirectional ordered byte streams.
- \* QUIC will automatically handle congestion control, packet loss, and reordering for stream data.
- \* QUIC streams allow multiple media streams to share congestion and flow control without otherwise blocking each other.
- \* QUIC streams also allow partial reliability, since either the sender or receiver can terminate the stream early without affecting the overall connection.
- \* With the DATAGRAM extension ([I-D.draft-ietf-quic-datagram]), further partially reliable models are possible, and applications can send congestion controlled datagrams below the MTU size.
- \* QUIC connections are established using an ALPN.
- \* QUIC endpoints can choose and change their connection ID.
- \* QUIC endpoints can migrate IP address without breaking the connection.
- \* Because QUIC is encapsulated in UDP, QUIC implementations can run in user space, rather than in kernel space, as TCP typically does. This allows more room for extensible APIs between application and transport, allowing more rapid implementation and deployment of new congestion control, retransmission, and prioritization mechanisms.
- \* QUIC is supported in browsers via HTTP/3 or WebTransport.
- \* With WebTransport, it is possible to write libraries or applications in JavaScript.

The specific advantages of interest may vary from use case to use case, but these advantages justify further investigation of "Media Over QUIC".

## 2. Terminology

## 2.1. The Many Meanings of "Media Over QUIC"

Protocol developers have been considering the implications of the QUIC protocol ([RFC9000]) for media transport for several years, resulting in a large number of possible meanings of the term "Media Over QUIC", or "MOQ". As of this writing, "Media Over QUIC" has had at least these meanings:

- \* any kind of media carried directly over the QUIC protocol, as a QUIC payload
- \* any kind of media carried indirectly over the QUIC protocol, as an RTP payload ([RFC3550])
- \* any kind of media carried indirectly over the QUIC protocol, as an HTTP/3 payload
- \* any kind of media carried indirectly over the QUIC protocol, as a WebTransport payload
- \* the encapsulation of any Media Transport Protocol (Section 2.2) in a QUIC payload
- \* an IETF mailing list ([MOQ-ml]), which was requested "... for discussion of video ingest and distribution protocols that use QUIC as the underlying transport", although discussion of other Media Over QUIC proposals have also been discussed there.

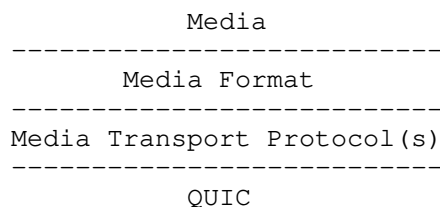
There may be IETF participants using other meanings as well.

As of this writing, the second bullet ("any kind of media carried indirectly over the QUIC protocol, as an RTP payload"), seems to be in scope for the IETF AVTCORE working group, and was discussed at some length at the February 2022 AVTCORE working group meeting [AVTCORE-2022-02], although no drafts in this space have yet been adopted by the AVTCORE working group.

## 2.2. Media Transport Protocol

This document describes considerations for work on extensions to existing "Media Transport Protocols" or creation of new "Media Transport Protocols".

Within this document, we use the term "Media Transport Protocol" to describe the protocol of interest. This is easier to understand if the reader assumes that we are talking about a protocol stack that looks something like this:



where "Media Format" would be something like RTP payload formats or ISOBMFF [ISOBMFF], and "Media Transport Protocol" would be something like RTP or HTTP. Not all possible proposals for "Media Over QUIC" follow this model, but for the ones that do, it seems useful to have names for "the protocol layers between Media and QUIC".

It is worth noting explicitly that the "Media Transport Protocol" layer might include more than one protocol. For example, a new Media Transport Protocol might be defined to run over HTTP, or even over WebTransport and HTTP.

### 2.3. Latency Requirement Categories

Within this document, we extend the latency requirement categories for streaming media described in [I-D.draft-ietf-mops-streaming-opcons]:

- \* ultra low-latency (less than 1 second)
- \* low-latency live (less than 10 seconds)
- \* non-low-latency live (10 seconds to a few minutes)
- \* on-demand (hours or more)

These latency bands were appropriate for streaming media, which was the target for [I-D.draft-ietf-mops-streaming-opcons], but some interactive media may have requirements that are significantly less than "ultra-low latency". Within this document, we are also using

- \* Ull-50 (less than 50 ms)
- \* Ull-200 (less than 200 ms)

Perhaps obviously, these last two latency bands are the shortened form of "ultra-low latency - 50 ms" and "ultra-low-latency - 200 ms". Perhaps less obviously, bikeshedding on better names and more useful values is welcomed.

### 3. Prior and Existing Specifications

Several draft specifications have been proposed which either encapsulate existing Media Transport Protocols in QUIC, or define their own new Media Transport Protocol on top of QUIC. With the exception of RUSH (Section 3.3), it is unknown if the other specifications listed in this section have had any deployments or interop with multiple implementations.

#### 3.1. QRT: QUIC RTP Tunnelling

[I-D.draft-hurst-quic-rtp-tunnelling]

QRT encapsulates RTP and RTCP and define the means of using QUIC datagrams with them, defining a new payload within a datagram frame which distinguishes packets for a RTP packet flow vs RTCP.

#### 3.2. RTP over QUIC

[I-D.draft-engelbart-rtp-over-quic]

This specification also encapsulates RTP and RTCP but unlike QRT which simply relies on the default QUIC congestion control mechanisms, it defines a set of requirements around QUIC implementation's congestion controller to permit the use of separate control algorithms.

#### 3.3. RUSH - Reliable (unreliable) streaming protocol

[I-D.draft-kpugin-rush]

Whilst RUSH predates the datagram specification, it uses its own frame types on top of QUIC to take advantage of QUIC implementations reassembling messages greater than MTU. In addition individual media frames are given their own stream identifiers to remove HoL blocking from processing out-of-order.

It defines its own registry for signalling codec information with room for future expansion but presently is limited to a subset of popular video and audio codecs and doesn't include other types (such as subtitles, transcriptions, or other signalling information) out of bitstream.

#### 3.4. Tunnelling SRT over QUIC

[I-D.draft-sharabayko-srt-over-quic]



Secure Reliable Transport (SRT) ([I-D.draft-sharabayko-srt]) itself is a general purpose transport protocol primarily for ingest transport use cases and this specification covers the encapsulation and delivery of SRT on top of QUIC using datagram frame types. This specification sets some requirements regarding how the two interact and leaves considerations for congestion control and pacing to prevent conflict between the two protocols. Apart from that, SRT provides a native support for stream multiplexing, thus contributing this missing functionality to QUIC datagrams.

### 3.5. Warp - Segmented Live Video Transport

[I-D.draft-lcurley-warp]

Warp's specification attempts to map Group of Picture encoding of video on top of QUIC streams. It depends on ISOBMFF containers to encapsulate both media as well as messaging, and defines prioritisation with separate considerations for audio and video. It doesn't yet define bi-directionality of media flows, and can be run over protocols like WebTransport [I-D.draft-ietf-webtrans-overview].

### 3.6. Comparison of Existing Specifications

\*\* Additional details for this comparison could usefully be added here. \*\*

- \* Some drafts attempt to use existing payloads of RTP, RTCP, and SDP, while others do not.
- \* Some use QUIC Datagram frames, while others use QUIC streams.
- \* All drafts take differing approaches to flow/stream identification and management. Some address congestion control and others just defer this to QUIC to handle.
- \* Some drafts specify ALPN identification, while others do not.

## 4. Use Cases Informing This Proposal

Our goal in this section is to understand the range of use cases that have been proposed for "Media Over QUIC".

Although some of the use cases described in this section came out of "RTP over QUIC" proposals, they are worth considering in the broader "Media Over QUIC" context, and may be especially relevant to MoQ, depending on whether "RTP over QUIC" requires major changes to RTP and RTCP, in order to meet the requirements arising out of the corresponding use cases.

An early draft in the "media over QUIC" space, [I-D.draft-rtpfolks-quic-rtp-over-quic], defined several key use cases. Some of the following use cases have been inspired by that document, and others have come from discussions with the wider MoQ community (among other places, a side meeting at IETF 112).

For each use case in this section, we also define

- \* the number of senders or receiver in a given session transmitting distinct streams,
- \* whether a session has bi-direction flows of media from senders and receivers, and
- \* the expected lowest latency requirements using the definitions specified in Section 2.

It is likely that we should add other characteristics, as we come to understand them.

#### 4.1. Interactive Media

The use cases described in this section have one particular attribute in common - the target latency for these cases are on the order of one or two RTTs. In order to meet those targets, it is not possible to rely on protocol mechanisms that require multiple RTTs to function effectively. For example,

- \* When the target latency is on the order of one RTT, it makes sense to use FEC [RFC6363] and codec-level packet loss concealment [RFC6716], rather than selectively retransmitting only lost packets. These mechanisms use more bytes, but do not require multiple RTTs in order to recover from packet loss.
- \* When the target latency is on the order of one RTT, it is impossible to use congestion control schemes like BBR [I-D.draft-cardwell-iccr-g-bbr-congestion-control], since BBR has probing mechanisms that rely on temporarily inducing delay and amortizing the consequences of that over multiple RTTs.

This may help to explain why these use cases often rely on protocols such as RTP [RFC3550], which provide low-level control of packetization and transmission.

## 4.1.1. Gaming

Attribute	Value
*Senders/Receivers*	One to One
*Bi-directional*	Yes
*Latency*	U11-50

Table 1

Where media is received, and user inputs are sent by the client. This may also include the client receiving other types of signalling, such as triggers for haptic feedback. This may also carry media from the client such as microphone audio for in-game chat with other players.

## 4.1.2. Remote Desktop

Attribute	Value
*Senders/Receivers*	One to One
*Bi-directional*	Yes
*Latency*	U11-50

Table 2

Where media is received, and user inputs are sent by the client. Latency requirements with this usecase are marginally different than the gaming use case. This may also include signalling and/or transmitting of files or devices connected to the user's computer.

## 4.1.3. Video Conferencing/Telephony

Attribute	Value
*Senders/Receivers*	Many to Many
*Bi-directional*	Yes

*Latency*	U11-50 to U11-200
-----	-----

Table 3

Where media is both sent and received; This may include audio from both microphone(s) or other inputs, or may include "screen sharing" or inclusion of other content such as slide, document, or video presentation. This may be done as client/server, or peer to peer with a many to many relationship of both senders and receivers. The target for latency may be as large as U11-200 for some media types such as audio, but other media types in this use case have much more stringent latency targets.

#### 4.2. Live Media

The use cases in this section, unlike the use cases described in Section 4.1, still have "humans in the loop", but these humans expect media to be "responsive", where the responsiveness is more on the order of 5 to 10 RTTs. This allows the use of protocol mechanisms that require more than one or two RTTs - as noted in Section 4.1, end-to-end recovery from packet loss and congestion avoidance are two such protocol mechanisms that can be used with Live Media.

To illustrate the difference, the responsiveness expected with videoconferencing is much greater than watching a video, even if the video is being produced "live" and sent to a platform for syndication and distribution.

##### 4.2.1. Live Media Ingest

Attribute	Value
*Senders/Receivers*	One to One
*Bi-directional*	No
*Latency*	U11-200 to Ultra-Low

Table 4

Where media is received from a source for onwards handling into a distribution platform. The media may comprise of multiple audio and/or video sources. Bitrates may either be static or set dynamically by signalling of connection information (bandwidth, latency) based on data sent by the receiver.

## 4.2.2. Live Media Syndication

Attribute	Value
*Senders/Receivers*	One to One
*Bi-directional*	No
*Latency*	U11-200 to Ultra-Low

Table 5

Where media is sent onwards to another platform for further distribution. The media may be compressed down to a bitrate lower than source, but larger than final distribution output. Streams may be redundant with failover mechanisms in place.

## 4.2.3. Live Media Streaming

Attribute	Value
*Senders/Receivers*	One to Many
*Bi-directional*	No
*Latency*	U11-200 to Ultra-Low

Table 6

Where media is received from a live broadcast or stream. This may comprise of multiple audio or video outputs with different codecs or bitrates. This may also include other types of media essence such as subtitles or timing signalling information (e.g. markers to indicate change of behaviour in client such as advertisement breaks). The use of "live rewind" where a window of media behind the live edge can be made available for clients to playback, either because the local player falls behind edge or because the viewer wishes to play back from a point in the past.

#### 4.3. On-Demand Media

Finally, the "On-Demand" use cases described in this section do not have a tight linkage between ingest and streaming, allowing significant transcoding, processing, insertion of video clips in a news article, etc. The latency constraints for the use cases in this section may be dominated by the time required for whatever actions are required before media are available for streaming.

##### 4.3.1. On-Demand Ingest

Attribute	Value
*Senders/Receivers*	One to Many
*Bi-directional*	No
*Latency*	On Demand

Table 7

Where media is ingested and processed for a system to later serve it to clients as on-demand media. This media provided from a pre-recorded source, or captured from live output, but in either case, this media is not immediately passed to viewers, but is stored for "on-demand" retrieval, and may be transcoded upon ingest.

##### 4.3.2. On-Demand Media Streaming

Attribute	Value
*Senders/Receivers*	One to Many
*Bi-directional*	No
*Latency*	On Demand

Table 8

Where media is received from a non-live, typically pre-recorded source. This may feature additional outputs, bitrates, codecs, and media types described in the live media streaming use case.

## 5. Proposed Scope for "Media Over QUIC"

Our proposal is that "Media Over QUIC" discussions focus first on the use cases described in Section 4.2, which are Live Media Ingest (Section 4.2.1), Syndication (Section 4.2.2), and Streaming (Section 4.2.3). Our reasoning for this suggestion follows.

Each of the above use cases in Section 4 fit into one of three classifications of solutions.

### 5.1. Analysis for Interactive Use Cases

The first group, Interactive Media, as described in Section 4.1, and covering gaming (Section 4.1.1), screen sharing (Section 4.1.2), and general video conferencing (Section 4.1.3), are largely covered by RTP, often in conjunction with WebRTC [WebRTC], and related protocols today.

Whilst there may be benefit in these use cases having a QUIC based protocol it may be more appropriate given the size of existing deployments to extend the RTP protocols and specifications.

### 5.2. Analysis for Live Media Use Cases

The second group of classifications, in Section 4.2, covering Live Media Ingest (Section 4.2.1), Live Media Syndication (Section 4.2.2), and Live Media Streaming (Section 4.2.3) are likely the use cases that will benefit most from this work.

Existing ingest and streaming protocols such as HLS [RFC8216] and DASH [DASH] are reaching limits towards how low they can reduce latency in live streaming and for scenarios where low-bitrate audio streams are used, these protocols add a significant amount of overhead compared to the media bitstream itself.

For this reason, we suggest that work on "Media Over QUIC" protocols target these use cases at this time.

### 5.3. Analysis for On-Demand Use Cases

The third group, Section 4.3, covering On-Demand Media Ingest (Section 4.3.1) and On-Demand Media streaming (Section 4.3.2) is unlikely to benefit from work in this space. Without the same "Live Media" latency requirements that would motivate deployment of new protocols, existing protocols such as HLS and DASH are probably "good enough" to meet the needs of these use cases.

This does not mean that existing protocols in this space are perfect. Segmented protocols such as HLS and DASH were developed to overcome the deficiencies of TCP, as used in HTTP/1.1 [RFC7230] and HTTP/2 [RFC7540], and do not make full use of the possible congestion window along the path from sender to receiver. Other protocols in this space have their own deficiencies. For example, RTSP [RFC7826] does not have easy ways to add support for new media codecs.

Our expectation is that these use cases will not drive work in the "Media Over QUIC" space, but as new protocols come into being, they may very well be taken up for these use cases as well.

## 6. Considerations for Protocol Work

Even a cursory examination of the existing proposals listed in Section 3 shows that there are fundamental differences in the approaches being used. This section is intended to "up-level" the conversation beyond specific protocols, so that we can more likely agree on what is important for protocol design work.

Please note that the considerations in this section are focused especially on the use cases described in Section 4.2, although other use cases are mentioned for comparison and contrast.

### 6.1. Here Be Dragons

The discussion in Section 6 is less mature than in most other sections of this document. The good news is that this section is fertile ground for people who would like to contribute to future revisions of this document. Comments are even more welcome for this section than for the rest of the document, for which they are welcome. The authors suggest that high-level comments are most appropriate at this time.

### 6.2. Codec Agility

When initiating a media session, both the sender and receiver will need to agree on the codecs, bitrates, resolution, and other media details based on capabilities and preferences. This agreement needs to take place before commencing media transmission, but might also take place during media transmission, perhaps as a result of changes to device output or network conditions (such as reduction in available network bandwidth).

It may be preferred to use existing ecosystem for such purposes, e.g. SDP [RFC4566].



### 6.3. Support an Appropriate Range of Latencies

Support for a nominal latency appropriate for the use cases that are in scope should be achievable, with consideration for the minimum buffer that a receiver playing content may need to handle congestion, packet loss, and other degradation in network quality.

### 6.4. Migration of Sessions

Handling of migration of a session between hosts, either of sender or receiver should be supported. This may either happen because the sender is undergoing maintenance or a rebalancing of resource, because the either is experiencing a change in network connectivity (such as a device moving from WiFi to cellular connectivity) or other reasons.

This may depend on QUIC capabilities such as [I-D.draft-ietf-quic-multipath] but support for full QUIC operation over multiple paths between senders and receivers is by no means essential.

### 6.5. Appropriate Congestion Control

An appropriate congestion control mechanism will depend upon the use cases under consideration.

It's worth remembering that we have more experience with QUIC carrying HTTP traffic than with any other type of application at this time, and consequently, we have more experience with congestion control mechanisms such as NewReno [RFC9002], Cubic [RFC8312], and BBR [I-D.draft-cardwell-iccr-g-bbr-congestion-control] being used with QUIC than with any other congestion control mechanisms. These congestion control mechanisms may also be appropriate for the on-demand use cases described in Section 4.3.

Conversely, for the interactive use cases described in Section 4.1, these congestion control mechanisms are very likely inappropriate, especially when QUIC is being used with a Media Transport Protocol such as RTP, which provides its own congestion control mechanism, and which does not seem to interact well with a second, QUIC-level congestion control mechanism. Congestion control mechanisms such as SReAM [RFC8298] or NADA [RFC8698] may be more appropriate for media. "Congestion Control Requirements for Interactive Real-Time Media" [RFC8836] is a useful reference.

Awkwardly, the live media use cases described in Section 4.2 live somewhere in the middle, and work will be needed to understand the characteristics of an appropriate congestion control mechanism for these use cases.

#### 6.6. Support Lossless and Lossy Media Transport

TODO: confirm scope of this draft to describe lossless media transport, lossy media transport, or both lossless and lossy transport.

#### 6.7. Flow Directionality

Media should be able to flow in either direction from client to server or vice-versa, either individually or concurrently but should only be negotiated at the start of the session.

#### 6.8. WebTransport

TODO: Unsure of the importance of this consideration for live media use cases. If this is critical, we have to consider two things:

- \* WebTransport supports HTTP/2, are we going to explicitly exclude it?
- \* Also, WebTransport [I-D.draft-ietf-webtrans-overview] has normative language around congestion control, which may be at odds with the considerations described in Section 6.5.

#### 6.9. Authentication

In order to allow hosts to authenticate one another, capabilities beyond what QUIC provides may be necessary. This should be kept simple but robust in nature to prevent attacks like credential brute-forcing.

TODO: More details are required here

#### 6.10. Considerations Implying QUIC Extensions

Most of the discussion of protocol work in this document has avoided mentioning capabilities that may be useful for some use cases, but seem to imply the need for extensions to the QUIC protocol, beyond what is already being considered in the IETF QUIC working group. These are included in this section, for completeness' sake.

#### 6.10.1. NAT Traversal

From Section 8.2 of [RFC9000]:

Path validation is not designed as a NAT traversal mechanism. Though the mechanism described here might be effective for the creation of NAT bindings that support NAT traversal, the expectation is that one endpoint is able to receive packets without first having sent a packet on that path. Effective NAT traversal needs additional synchronization mechanisms that are not provided here.

Although there are use cases that would benefit from a mechanism for NAT traversal, a QUIC protocol extension would be needed to support those use cases.

#### 6.10.2. Multicast

Even if multicast and other network broadcasting capabilities are often used in delivering media in our use cases, QUIC doesn't yet support multicast, and a QUIC protocol extension would be needed to do so. In addition, the inclusion of multicast would introduce more complexity in both the specification and client implementations. On the other hand, UDP multicast may be considered as the last mile delivery transport outside of QUIC transport, thus it would be beneficial for a protocol to provide such an opportunity (e.g. RTP/QUIC -> RTP/UDP).

### 7. IANA Considerations

This document makes no requests of IANA.

### 8. Security Considerations

As this document is intended to guide discussion and consensus, it introduces no security considerations of its own.

### 9. Informative References

[AVTCORE-2022-02]

"AVTCORE 2022-02 interim meeting materials", February 2022, <<https://datatracker.ietf.org/meeting/interim-2022-avtcore-01/session/avtcore>>.

[DASH]

"ISO/IEC 23009-1:2019: Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment formats (2nd edition)", n.d., <<https://www.iso.org/standard/79329.html>>.

- [I-D.draft-cardwell-iccr-g-bbr-congestion-control]  
Cardwell, N., Cheng, Y., Yeganeh, S. H., Swett, I., and V. Jacobson, "BBR Congestion Control", Work in Progress, Internet-Draft, draft-cardwell-iccr-g-bbr-congestion-control-01, 7 November 2021, <<https://datatracker.ietf.org/doc/html/draft-cardwell-iccr-g-bbr-congestion-control-01>>.
- [I-D.draft-engelbart-rtp-over-quic]  
Ott, J. and M. Engelbart, "RTP over QUIC", Work in Progress, Internet-Draft, draft-engelbart-rtp-over-quic-01, 25 October 2021, <<https://datatracker.ietf.org/doc/html/draft-engelbart-rtp-over-quic-01>>.
- [I-D.draft-hurst-quic-rtp-tunnelling]  
Hurst, S., "QRT: QUIC RTP Tunnelling", Work in Progress, Internet-Draft, draft-hurst-quic-rtp-tunnelling-01, 28 January 2021, <<https://datatracker.ietf.org/doc/html/draft-hurst-quic-rtp-tunnelling-01>>.
- [I-D.draft-ietf-mops-streaming-opcons]  
Holland, J., Begen, A., and S. Dawkins, "Operational Considerations for Streaming Media", Work in Progress, Internet-Draft, draft-ietf-mops-streaming-opcons-09, 1 March 2022, <<https://datatracker.ietf.org/doc/html/draft-ietf-mops-streaming-opcons-09>>.
- [I-D.draft-ietf-quic-datagram]  
Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable Datagram Extension to QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-datagram-10, 4 February 2022, <<https://datatracker.ietf.org/doc/html/draft-ietf-quic-datagram-10>>.
- [I-D.draft-ietf-quic-http]  
Bishop, M., "Hypertext Transfer Protocol Version 3 (HTTP/3)", Work in Progress, Internet-Draft, draft-ietf-quic-http-34, 2 February 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-quic-http-34>>.

- [I-D.draft-ietf-quic-multipath]  
Liu, Y., Ma, Y., Coninck, Q. D., Bonaventure, O., Huitema, C., and M. Kuehlewind, "Multipath Extension for QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-multipath-00, 2 February 2022, <<https://datatracker.ietf.org/doc/html/draft-ietf-quic-multipath-00>>.
- [I-D.draft-ietf-webtrans-overview]  
Vasiliev, V., "The WebTransport Protocol Framework", Work in Progress, Internet-Draft, draft-ietf-webtrans-overview-02, 28 July 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-webtrans-overview-02>>.
- [I-D.draft-kpugin-rush]  
Pugin, K., Frindell, A., Cenzano, J., and J. Weissman, "RUSH - Reliable (unreliable) streaming protocol", Work in Progress, Internet-Draft, draft-kpugin-rush-00, 12 July 2021, <<https://datatracker.ietf.org/doc/html/draft-kpugin-rush-00>>.
- [I-D.draft-lcurley-warp]  
Curley, L., "Warp - Segmented Live Video Transport", Work in Progress, Internet-Draft, draft-lcurley-warp-00, 9 February 2022, <<https://datatracker.ietf.org/doc/html/draft-lcurley-warp-00>>.
- [I-D.draft-rtpfolks-quic-rtp-over-quic]  
Ott, J., Even, R., Perkins, C., and V. Singh, "RTP over QUIC", Work in Progress, Internet-Draft, draft-rtpfolks-quic-rtp-over-quic-01, 1 September 2017, <<https://datatracker.ietf.org/doc/html/draft-rtpfolks-quic-rtp-over-quic-01>>.
- [I-D.draft-sharabayko-srt]  
Sharabayko, M., Sharabayko, M., Dube, J., Kim, J., and J. Kim, "The SRT Protocol", Work in Progress, Internet-Draft, draft-sharabayko-srt-01, 7 September 2021, <<https://datatracker.ietf.org/doc/html/draft-sharabayko-srt-01>>.
- [I-D.draft-sharabayko-srt-over-quic]  
Sharabayko, M. and M. Sharabayko, "Tunnelling SRT over QUIC", Work in Progress, Internet-Draft, draft-sharabayko-srt-over-quic-00, 28 July 2021, <<https://datatracker.ietf.org/doc/html/draft-sharabayko-srt-over-quic-00>>.

- [ISOBMFF] "ISO/IEC 14496-12:2022 Information technology Coding of audio-visual objects Part 12: ISO base media file format", January 2022,  
<<https://www.iso.org/standard/83102.html>>.
- [MOQ-ml] "Moq -- Media over QUIC", n.d.,  
<<https://www.ietf.org/mailman/listinfo/moq>>.
- [QUIC-goals] "Initial Charter for QUIC Working Group", October 2016,  
<<https://datatracker.ietf.org/doc/charter-ietf-quic/01/>>.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <<https://www.rfc-editor.org/rfc/rfc3550>>.
- [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <<https://www.rfc-editor.org/rfc/rfc4566>>.
- [RFC6363] Watson, M., Begen, A., and V. Roca, "Forward Error Correction (FEC) Framework", RFC 6363, DOI 10.17487/RFC6363, October 2011, <<https://www.rfc-editor.org/rfc/rfc6363>>.
- [RFC6716] Valin, JM., Vos, K., and T. Terriberry, "Definition of the Opus Audio Codec", RFC 6716, DOI 10.17487/RFC6716, September 2012, <<https://www.rfc-editor.org/rfc/rfc6716>>.
- [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <<https://www.rfc-editor.org/rfc/rfc7230>>.
- [RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <<https://www.rfc-editor.org/rfc/rfc7540>>.
- [RFC7826] Schulzrinne, H., Rao, A., Lanphier, R., Westerlund, M., and M. Stiemerling, Ed., "Real-Time Streaming Protocol Version 2.0", RFC 7826, DOI 10.17487/RFC7826, December 2016, <<https://www.rfc-editor.org/rfc/rfc7826>>.
- [RFC8216] Pantos, R., Ed. and W. May, "HTTP Live Streaming", RFC 8216, DOI 10.17487/RFC8216, August 2017, <<https://www.rfc-editor.org/rfc/rfc8216>>.

- [RFC8298] Johansson, I. and Z. Sarker, "Self-Clocked Rate Adaptation for Multimedia", RFC 8298, DOI 10.17487/RFC8298, December 2017, <<https://www.rfc-editor.org/rfc/rfc8298>>.
- [RFC8312] Rhee, I., Xu, L., Ha, S., Zimmermann, A., Eggert, L., and R. Scheffenegger, "CUBIC for Fast Long-Distance Networks", RFC 8312, DOI 10.17487/RFC8312, February 2018, <<https://www.rfc-editor.org/rfc/rfc8312>>.
- [RFC8698] Zhu, X., Pan, R., Ramalho, M., and S. Mena, "Network-Assisted Dynamic Adaptation (NADA): A Unified Congestion Control Scheme for Real-Time Media", RFC 8698, DOI 10.17487/RFC8698, February 2020, <<https://www.rfc-editor.org/rfc/rfc8698>>.
- [RFC8836] Jesup, R. and Z. Sarker, Ed., "Congestion Control Requirements for Interactive Real-Time Media", RFC 8836, DOI 10.17487/RFC8836, January 2021, <<https://www.rfc-editor.org/rfc/rfc8836>>.
- [RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://www.rfc-editor.org/rfc/rfc9000>>.
- [RFC9002] Iyengar, J., Ed. and I. Swett, Ed., "QUIC Loss Detection and Congestion Control", RFC 9002, DOI 10.17487/RFC9002, May 2021, <<https://www.rfc-editor.org/rfc/rfc9002>>.
- [WebRTC] "Web Real-Time Communications Working Group", n.d., <<https://www.w3.org/groups/wg/webrtc>>.

#### Appendix A. Acknowledgements

The authors would like to thank the many authors of the specifications referenced in Section 3 for their work:

- \* Alan Frindell
- \* Colin Perkins
- \* Jake Weissman
- \* Joerg Ott
- \* Jordi Cenzano
- \* Kirill Pugin
- \* Maria Sharabayko
- \* Mathis Engelbart
- \* Maxim Sharabayko
- \* Roni Even
- \* Sam Hurst

\* Varun Singh

The authors would like to thank Alan Frindell, Luke Curley, and Maxim Sharabayko for text contributions to this draft.

James Gruessing would also like to thank Francesco Illy and Nicholas Book for their part in providing the needed motivation.

#### Authors' Addresses

James Gruessing  
Nederlandse Publieke Omroep  
Netherlands  
Email: james.ietf@gmail.com

Spencer Dawkins  
Tencent America LLC  
United States of America  
Email: spencerdawkins.ietf@gmail.com



Internet Area Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 26 April 2022

K. Xu  
X. Wang  
Y. Guo  
Tsinghua University  
23 October 2021

Control Plane of Inter-Domain Source Address Validation Architecture  
draft-guo-intarea-savax-control-00

## Abstract

Because the Internet forwards packets according to the IP destination address, packet forwarding typically takes place without inspection of the source address and malicious attacks have been launched using spoofed source addresses. The inter-domain source address validation architecture is an effort to enhance the Internet by using state machine to generate consistent tags. When communicating between two end hosts at different ADs of IPv6 network, tags will be added to the packets to identify the authenticity of the IPv6 source address.

This memo focus on the control plane of the SAVA-X mechanism.

## Status of This Memo

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

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

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

This Internet-Draft will expire on 26 April 2022.

## Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document.

Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology and Abbreviation . . . . .	3
3. The design of the Consortium Blockchain . . . . .	4
3.1. Trust Alliance . . . . .	4
3.2. Consortium Blockchain . . . . .	5
3.3. Joining and Exiting . . . . .	6
3.3.1. Node Joining . . . . .	6
3.3.2. Node Exiting . . . . .	7
4. Alliance information and state machine maintenance based on the consortium blockchain . . . . .	7
4.1. AD Registration Information Record . . . . .	8
4.2. AD Prefix Information Record . . . . .	9
5. Time synchronization . . . . .	10
6. Security Consideration . . . . .	12
7. IANA Consideration . . . . .	12
8. Acknowledgements . . . . .	12
9. Normative References . . . . .	12
Authors' Addresses . . . . .	12

## 1. Introduction

The Inter-Domain Source Address Validation (SAVA-X) mechanism establishes a trust alliance among Address Domains (AD), maintains a one-to-one state machine among ADs, generates a consistent tag, and deploys the tag to the ADs' border router (AER). The AER of the source AD adds a tag to identify the identity of the AD to the packet originating from one AD and sinking in another AD. The AER of the destination AD verifies the source address by validating the correctness of the tag to determine whether it is a packet with a forged source address.

In the process of packet forwarding, if the source address and the destination address of this packet both are addresses in the trust alliance, however the tag is not added or incorrectly added, AER of the destination AD determines that the source address is forged and directly discards this packet. The destination AD forwards the packet directly for packets whose source address is an address outside the trust alliance.

This document mainly studies the relevant specifications of the control plane of the inter-domain source address validation architecture mechanism between ADs, which will protect IPv6 networks from being forged source address. You could see [RFC8200] for more details about IPv6. It stipulates the design of the consortium blockchain, the nodes' joining and exiting, the maintenance of trust alliance information based on the consortium blockchain, and the maintenance of the state machine. Its promotion and application can realize the standardization of the control plane in the SAVA-X to facilitate the related equipment developed by different manufacturers and organizations to cooperate to accomplish the inter-domain source address validation jointly.

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, BCP 14 [RFC2119] and indicate requirement levels for compliant CoAP implementations.

## 2. Terminology and Abbreviation

Abbreviation	Description
AD	Address Domain, the unit of a trust alliance, which is an address set consisting of all IPv6 addresses corresponding to an IPv6 address prefix.
TA	Trust Alliance, the IPv6 network that uses the SAVA-X mechanism.
STA	sub-Trust Alliance, parts of TA.
STA-admin	STA Administrator, the administrator of STA.
ACS	AD Control Server, the server that maintains state machine with other ACS and distribute information to AER.
AER	AD border router, which is placed at the boundary of an AD of STA.
ADID	The identity of an AD.
ADID_Rec	The record of a number of an AD.

ARI_Rec	The record with relavent information of an AD or STA.
API_Rec	The record of prefix of an AD or STA.
CSR	Certificate Signing Request, which is used for an AD or STA to join or exit the consortium blockchain.
SM	State Machine, which is maintained by a pair of ACS to generate tags.
Tag	The authentic identification of source address of a packet.

Table 1

### 3. The design of the Consortium Blockchain

As discussed in the introduction, consortium blockchain will be used in SAVA-X mechanism.

#### 3.1. Trust Alliance

Trust Alliance (TA) is a hierarchical structure. Address domains (AD) are assigned into different sub-trust alliances (STA) according to geographic location, economic relationship, political relationship, social relationship, and military relationship. AD is the minimum unit for trust. The one-to-one maintenance state machine between ADs located in the same layer of sub-trust alliance generates consistent tags and deploys the tags to their AERs. The ADs in each sub-trust alliance elect a master AD node. The master AD node represents the sub-trust alliance and maintains the alliance-level state machine with other master AD nodes to generate alliance-level tags. When communicating across sub-trust alliances, it is necessary to achieve the feature of tag replacement.

The AD in the SAVA-X must be located in a specific sub-trust alliance. According to its position in the SAVA-X, AD can be divided into three roles: primary address domain, boundary address domain, and ordinary address domain which is neither primary nor boundary address domain.

- \* The primary address domain is the representative node of the aforementioned sub-trust alliance and is used to establish connection with the primary address domain of other sub-trust alliances. In this way, the relationship between trust alliances finally forms a tree-like relationship, and there will be no direct relationship between address domains under the same branch.
- \* The boundary address domain is the address domain located at the boundary of the sub-trust alliance. It sends the packet to other sub-trust alliances or outside the trust alliance.
- \* The ordinary address domain is neither the primary address domain nor the address domain of the boundary address domain.

Due to the uncontrollable packet forwarding path, in SAVA-X, a virtual address domain needs to be set up as a non-boundary AD to communicate with the sub-trust alliance outside or receive packets sent from outside the sub-trust alliance to maintain the state machine. The virtual AD is recorded as AD\_V (Virtual Address Domain). When a packet from an AD in a sub-trust alliance needs to be sent outside the sub-trust alliance, but there are multiple paths to the destination AD in the sub-trust alliance, the sub-trust alliance may have multiple boundary ADs to reach the destination AD. The sub-trust alliance does not know which boundary AD will be selected during the packet forwarding. Therefore, the primary function of AD\_V is to prevent this by specifying the specific tag that should be added to the packet sent to the external address domain of the sub-trust alliance.

What's more, the tag needs to be verified by the boundary address domain of the sub-trust alliance. Therefore, the boundary AD also needs to receive the tags maintained by the AD and AD\_V in the trust alliance. As a tag for communicating data between the non-primary address domain and the external address domain of the sub-trust alliance.

### 3.2. Consortium Blockchain

To simplify the control plane's design and avoid the single point failure to subvert the SAVA-X, we design the SAVA-X with a decentralized infrastructure which will store the information of the trust alliance.

The consortium blockchain is composed of the trust alliance management committee chain and several sub-chains. Among them, the management committee chain is composed of several nodes to manage the administrator nodes of each sub-chain. The consortium blockchain records information of the sub-trust alliance administrator node,

named as STA-admin (Sub Trust Alliance administrator), and member list of each sub-chain which are packaged and submitted by the STA-admin. Each sub-trust alliance has one STA-admin that is assumed by a specific elected AD in the sub-trust alliance. The AD in the same sub-trust alliance forms a private chain to maintain the information of the members of the sub-trust alliance jointly. The STA-admin in each sub-trust alliance is responsible for managing the joining and exiting of the sub-trust alliance node. The STA-admin of each sub-trust alliance maintains the relationship of the members in each sub-trust alliance through the trust alliance management committee chain.

### 3.3. Joining and Exiting

#### 3.3.1. Node Joining

This is the admission of joining of the sub-trustalliance member AD. The prerequisite for the AD to join the sub-trust alliance is to have a certificate issued by the STA-admin firstly. AD's Address Control Server (ACS), which will maintain state machine with other ACS and distribute alliance information and other information to AER, submits a Certificate Signing Request file to the STA-admin of the sub-trust alliance that it wants to join to request the certificate. The CSR file includes ADID, ACS address information, IPv6 address prefix information, and its public key information. If the file is valid, STA-admin verifies the file, generates a node certificate, packages the AD's information into a block, and updates the list of members of the sub-consortium. STA-admin submits the latest block to the consortium blockchain, and the consortium blockchain updates the list of alliance members of the entire trust alliance.

When a sub-trust alliance want to join the trust alliance, STA-admin submits a CSR file to the consortium blockchain, including the member information list in the sub-chain and the information of the STA-admin. It requires offline negotiation and cooperation to apply for joining the consortium blockchain. The consortium blockchain management committee verifies the validity of the request, issues administrator certificates, and updates block information. The STA-admins in the current trust alliance jointly maintain a management committee chain, manage the administrator certificates of each sub-trust alliance, and use the certificates for encrypted communication. STA-admin submits the list of members of the sub-trust alliance to the consortium blockchain and joins the entire trust alliance.

After a node joins the consortium blockchain, there is an Effecting Time and an Expiration Time in the CSR file. STA-admin will assign the sub-trust alliance member with an ADID number if it does not contain the ADID information in the submitted information. The consortium blockchain will give the permitted sub-trust alliance a

sub-trust alliance number if the information submitted by the sub-trust alliance does not have the sub-trust alliance number. If there is a conflict between the submitted information and the returned information, the returned ADID or sub-trust alliance number will be selected.

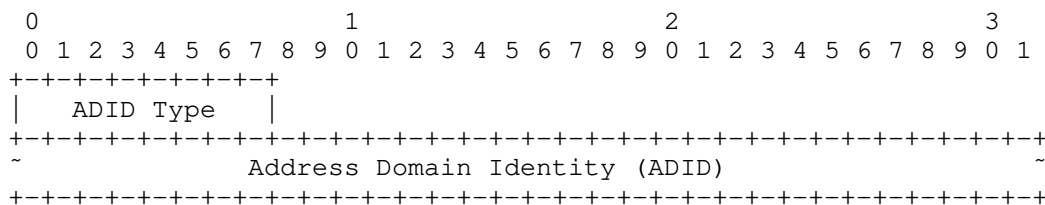
### 3.3.2. Node Exiting

The member node needs to submit the CSR file again to delete its relevant information. Its STA-admin decides whether to allow it exit or not. After passing the validation, nodes of the sub-blockchain delete the relevant member information. It also needs to submit a CSR file for the exit of the sub-trust alliance node, which the alliance management committee decides whether to allow it. After validating the receiving exit request, other sub-trust alliance administrator nodes need to delete their maintenance-related sub alliance node information.

## 4. Alliance information and state machine maintenance based on the consortium blockchain

On the AER of the destination AD, to validate the tag, it is first necessary to find out the sub-trust alliance number from the source address of the arriving packet and find out its corresponding source Address Domain Identity (ADID) number. Then find the currently valid tag according to the ADID number. The generation of the tag requires the maintenance of the state machine between the ACSes. In SAVA-X, member ADs need to inform each other of their sub-trust alliance number, ADID number, AD role, ACS address, and IPv6 address prefix. The members interact with each other with the state machine information according to the hierarchical division structure after obtaining the basic information of the other members. And use the tags generated by the state machine during the packet forwarding after the specified time to add and validate the tags.

The relevant information needs to be stored in the sub-chains, where the node is located after joining the consortium blockchain. The information stored on the consortium blockchain needs the content specified in the following three message formats, namely ADID\_Rec, ARI\_Rec, and API\_Rec. We give the packet format required by SAVA-X in the control plane as follows. ## Address Domain Identity Record Address Domain Identity Record (ADID\_Rec) is used to identify an address domain uniquely in the trust alliance.

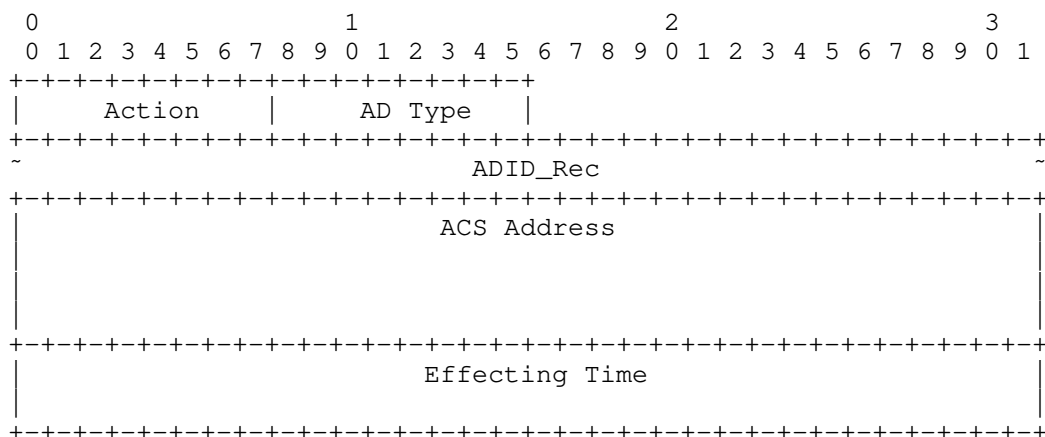


ADID Type: 8-bit Type of ADID, 1 for 16-bits AS number, 2 for 32-bits AS number and other unassigned.

ADID: The 16-bit or 32-bit ADID number. Its value can be the AS number or the number assigned by the consortium blockchain, and the specific length is determined by the ADID Type field. When each bit of ADID is 1, it represents that the AER requests the information of all members from ACS.

#### 4.1. AD Registration Information Record

AD Registration Information Record (ARI\_Rec) is the registration information record of AD, which is used to record the necessary information required to register a specific member AD. The ACS and AD establish connections.



Action: 8-bit instruction to add (ADD=1) or delete (DEL=2) this record. Others are unassigned.

AD Type: 8-bit unsigned number indicating the role of AD. 0 for ordinary AD, 1 for primary AD and 2 for boundary AD. Others are unassigned.

ADID\_Rec: Reference the ADID\_Rec packet.



ACS Address: 128-bit the IPv6 address of ACS.

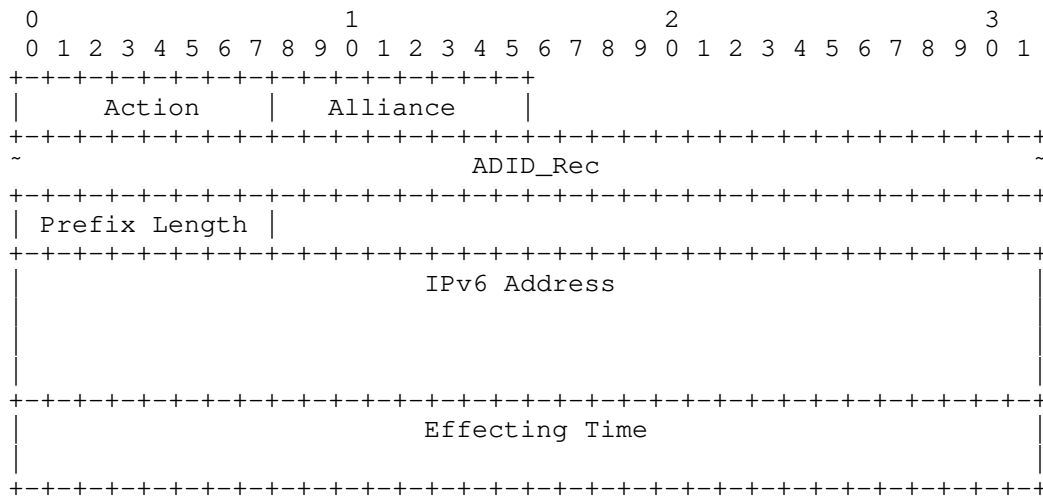
Effecting Time: 64-bit time specifies when this record is applied.

It is recommended to use the 64 bits struct timeval format for the effecting time of the execution of this record. If all bits of this field are 0 or earlier than the current time, it means that it takes effect immediately.

The role of address domain is essential, and each address domain needs to be assigned a corresponding role according to its position in the sub-trust alliance. A sub-trust alliance needs to set one (and only one) primary address domain. It serves as the representative of the sub-trust alliance. The tag generated by its ACS and the ACSes of other sub-trust alliances' primary address domain maintains the state machine to generate the tag of the sub-trust alliance, and it issues the tag to the boundary address domain of the sub-trust alliance. A specific recommendation of a consensus algorithm could generate the primary address domain or be directly specified by the operator of the address domain with offline negotiation. The boundary address domain means that packet forwards outside the address domain is no longer in the current sub-trust alliance. The primary address domain can be a boundary address domain or not. The tag replacement may occur in the boundary address domain. The ordinary address domain is neither a primary address domain nor a boundary address domain.

#### 4.2. AD Prefix Information Record

AD Prefix Information Record (API\_Rec) is the prefix information corresponds to the prefix of a specific AD. An AD may have more than one prefix, so registration information and prefix information records must be separated.



Action: 8-bit instruction to add (ADD=1) or delete (DEL=2) this record. Others are unassigned.

Alliance: 8-bit the sub-trust alliance number.

ADID\_Rec: Reference the ADID\_Rec packet.

Prefix Length: 8-bit the length of the IPv6 prefix.

IPv6 Address: 128-bit indicates a certain address prefix operated by the corresponding member AD using together with Prefix Length.

Effecting Time: 64-bit time specifies when this record is applied. It is recommended to use the 64 bits struct timeval format for the effecting time of the execution of this record. If all bits of this field are 0 or earlier than the current time, it means that it takes effect immediately.

ARI\_Rec and API\_Rec are required to store the AD information in the consortium blockchain and send it to all AERs of their AD with the communication protocol between ACS and AER.

## 5. Time synchronization

Due to the time error between the border routers of the member ADs, to ensure the correct operation of the tag validation, it is necessary to make each device including ACS and AER in the trust alliance calibrate to the same clock reference. The NTP protocol could achieve this goal. You could see [RFC5905] for more details.

Although the NTP protocol can guarantee the time synchronization between AERs, there may inevitably still have a slight time difference between AERs and ACSes. Therefore, each AER sets a shared time slice. With this time slice, both the expired tag and the new tag are considered valid. That is, more than one tag is valid for a while. The destination AD needs to validate all valid tags belonging to a specific source AD. The tag is correct if one of the tags is validated.

Assuming that the maximum time difference between AER and ACS is  $t_e$ , we set a shared time slice with a length of  $2t_e$  between two adjacent tags. In this shared time slice, the two tags before and after are valid. The validity period of the tag with the shared time slice is shown below, see Figure 1.

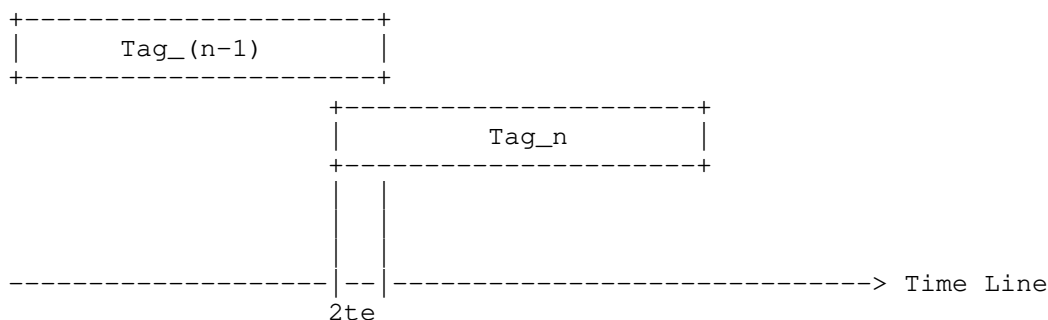


Figure 1: Validity period of tag with the shared time slice

In addition to the time difference, we should also take into account the packet transmission delay in the network. Set the minimum delay to  $td\_min$  and the maximum delay to  $td\_max$ . The expiration of  $Tag\_n$  should be extended  $td\_max$  later, and the beginning of  $Tag\_n$  validity period should be delayed  $td\_min$  later. The shared time slice and tag validity period corrected according to transmission delay are shown as follows, see Figure 2.

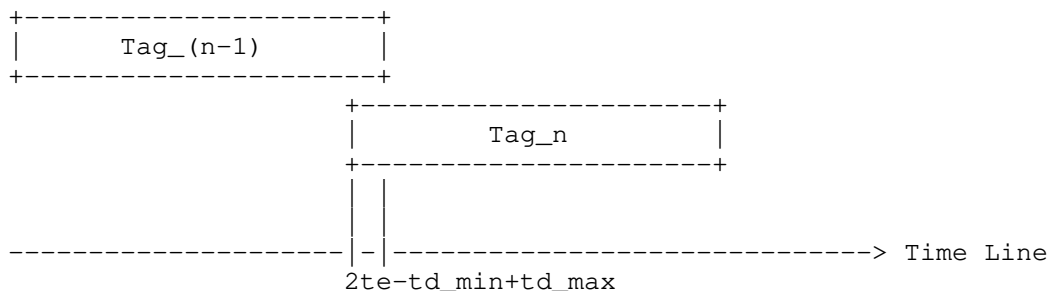


Figure 2: Validity period of tag with the shared time slice after modified

The expiration of the Tag<sub>n</sub> is extended to  $te+td_{max}$ , and the beginning of the Tag<sub>(n+1)</sub> is extended to  $te-td_{min}$ . The parameters such as  $te$ ,  $td_{min}$ ,  $td_{max}$ , the period of the shared time slice, and tag validity period are determined by the destination based on the actual network environment. Therefore, the lifecycle of a tag is as:  $lifecycle = Transition\ Interval + 2te - td_{min} + td_{max}$ .

## 6. Security Consideration

This present memo doesnot find any security problem.

## 7. IANA Consideration

This document makes no requests of IANA.

## 8. Acknowledgements

Much of the content of this document is the expansion of the IETF [RFC5210] in inter-domain level. Thanks to the effort of pioneers.

## 9. 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>>.
- [RFC5210] Wu, J., Bi, J., Li, X., Ren, G., Xu, K., and M. Williams, "A Source Address Validation Architecture (SAVA) Testbed and Deployment Experience", RFC 5210, DOI 10.17487/RFC5210, June 2008, <<https://www.rfc-editor.org/info/rfc5210>>.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.

## Authors' Addresses

Ke Xu  
Computer Science, Tsinghua University  
Qinghuayuan street, Haidian District  
Beijing  
100084  
China

Email: xuke@tsinghua.edu.cn

Xiaoliang Wang  
Computer Science, Tsinghua University  
Qinghuayuan street, Haidian District  
Beijing  
100084  
China

Email: wangxiaoliang0623@foxmail.com

Yangfei Guo  
Institute for Network Sciences and Cyberspace, Tsinghua University  
Qinghuayuan street, Haidian District  
Beijing  
100084  
China

Email: guoyangf19@mails.tsinghua.edu.cn

Internet Area Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 26 April 2022

K. Xu  
X. Wang  
Y. Guo  
Tsinghua University  
23 October 2021

Data Plane of Inter-Domain Source Address Validation Architecture  
draft-guo-intarea-savax-data-00

## Abstract

Because the Internet forwards packets according to the IP destination address, packet forwarding typically takes place without inspection of the source address and malicious attacks have been launched using spoofed source addresses. The inter-domain source address validation architecture is an effort to enhance the Internet by using state machine to generate consistent tags. When communicating between two end hosts at different ADs of IPv6 network, tags will be added to the packets to identify the authenticity of the IPv6 source address.

This memo focus on the data plane of the SAVA-X mechanism.

## Status of This Memo

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

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

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

This Internet-Draft will expire on 26 April 2022.

## Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document.

Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology and Abbreviation . . . . .	3
3. State Machine Mechanism . . . . .	4
4. Tag . . . . .	5
4.1. Tag Generation Algorithm . . . . .	5
4.1.1. Pseudo-Random Number Algorithm . . . . .	5
4.1.2. Hash Chain Algorithm . . . . .	6
4.2. Tag Update . . . . .	7
5. Packet Processing at AER . . . . .	8
5.1. Port Classification . . . . .	9
5.2. Source Address Validation . . . . .	9
5.3. Packet Classification . . . . .	9
5.4. Tag Addition . . . . .	10
5.5. Tag Verification . . . . .	10
5.6. Tag Replacement . . . . .	11
6. Packet Signature . . . . .	12
7. Security Consideration . . . . .	13
8. IANA Considerations . . . . .	13
9. Acknowledgements . . . . .	14
10. Normative References . . . . .	14
Authors' Addresses . . . . .	15

## 1. Introduction

The Inter-Domain Source Address Validation (SAVA-X) mechanism establishes a trust alliance among Address Domains (AD), maintains a one-to-one state machine among ADs, generates a consistent tag, and deploys the tag to the ADs' border router (AER). The AER of the source AD adds a tag to identify the identity of the AD to the packet originating from one AD and sinking in another AD. The AER of the destination AD verifies the source address by validating the correctness of the tag to determine whether it is a packet with a forged source address.

In the process of packet forwarding, if the source address and the destination address of this packet both are addresses in the trust alliance, however the tag is not added or incorrectly added, AER of the destination AD determines that the source address is forged and directly discards this packet. The destination AD forwards the packet directly for packets whose source address is an address outside the trust alliance.

This document mainly studies the relevant specifications of the data plane of the inter-domain source address validation architecture mechanism between ADs, which will protect IPv6 networks from being forged source address. You could see [RFC8200] for more details about IPv6. It stipulates the state machine, tag generation and update, tag processing in AER, and packet signature. Its promotion and application can realize the standardization of the data plane in the SAVA-X to facilitate the related equipment developed by different manufacturers and organizations to cooperate to accomplish the inter-domain source address validation jointly.

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, BCP 14 [RFC2119] and indicate requirement levels for compliant CoAP implementations.

## 2. Terminology and Abbreviation

Abbreviation	Description
AD	Address Domain, the unit of a trust alliance, which is an address set consisting of all IPv6 addresses corresponding to an IPv6 address prefix.
TA	Trust Alliance, the IPv6 network that uses the SAVA-X mechanism.
ACS	AD Control Server, the server that maintains state machine with other ACS and distribute information to AER.
AER	AD border router, which is placed at the boundary of an AD of STA.
ADID	The identity of an AD.
ADID_Rec	The record of a number of an AD.



ARI_Rec	The record with relavent information of an AD or STA.
API_Rec	The record of prefix of an AD or STA.
SM	State Machine, which is maintained by a pair of ACS to generate tags.
Tag	The authentic identification of source address of a packet.

Table 1

### 3. State Machine Mechanism

In SAVA-X, state machine mechanism is used to generate, update, and manage the tags.

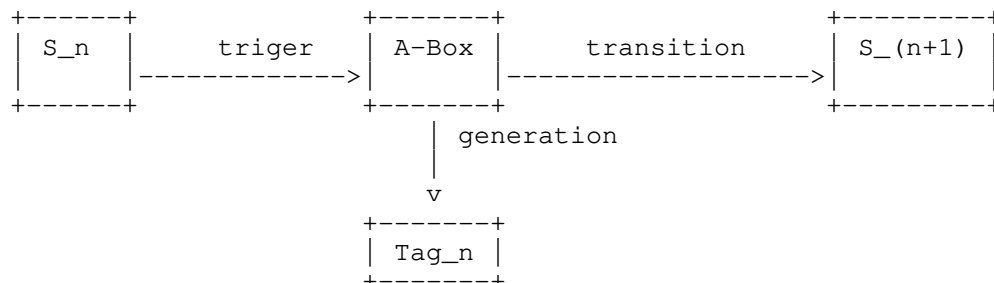


Figure 1: State machine mechanism.

State:  $S_n$  and  $S_{(n+1)}$  represent the current state and next state of the SM respectively.

Tag:  $Tag_n$  is generated in the progress of state transiting from  $S_n$  to  $S_{(n+1)}$ .

Algorithm Box: A-Box is Alogorithm Box. It is used to transite the State and generate the tag. It takes the current State as the input and the following State and current tag as the output. The algorithm box consists of two parts: one is the transition function  $transit()$ ,  $S_{(n+1)} = transit(S_n)$ ; the second is the function  $generate()$  to generate tags.  $Tag_n = generate(S_n)$ . Algorithm box (A-Box) is the core of state machine. It determines the data structure of state and tag, the specific mode of state machine implementation, as well as its security and complexity.

Trigger: It is used to trig the transition of State.

Transition: It reprents the progress of state transiting from  $S_n$  to  $S_{(n+1)}$ .

Generation: It reprents the progress of calculating the current tag from current State.

#### 4. Tag

##### 4.1. Tag Generation Algorithm

There are two ways to generate tags: pseudo-random number algorithm and hash chain algorithm.

##### 4.1.1. Pseudo-Random Number Algorithm

In the pseudo-random number generation algorithm, an initial number or string is usually used as the "seed", which corresponds to the initial state of the state machine. Using seeds, a pseudo-random number sequence is generated as a tag sequence through some algorithm. Next, we would take KISS (keep it simple stub), a pseudo-random number generation algorithm, as an example to introduce how to apply it to the state machine mechanism. For the algorithm details of KISS, you could refer to the following reference pseudo code:

```
/* Seed variables */
uint x = 123456789, y = 362436000, z = 521288629, c = 7654321;
uint KISS(){
    const ulong a = 698769069UL;
    ulong t;
    x = 69069*x+12345;
    y ^= (y<<13); y ^= (y>>17); y ^= (y<<5);
    t = a*z+c; c = (t>>32);
    z=cast(uint)t;
    return x+y+z;
}
```

Figure 2: KISS99: Pseudo-random number generatation

In this algorithm, State  $S$  can be expressed as  $(x, y, z, c)$ . The algorithm box is KISS(). After each calculation, the state undergoes a transition from  $S_n$  to  $S_{(n+1)}$ , that is, the four variables  $x$ ,  $y$ ,  $z$  and  $c$  are all changed. At the same time, a pseudo-rng number  $(x + y + z)$  is generated.

As the state machine shown above, the initial state is  $S_0 = (123456789, 362436000, 521288629, 7654321)$ . In fact, the initial state can be arbitrarily selected by the algorithm shown below:

```
void init_KISS() {
    x = devrand();
    while (!(y = devrand())); /* y must not be zero */
    z = devrand();
    /* Don't really need to seed c as well
       but if you really want to... */
    c = devrand() % 698769069; /* Should be less than 698769069 */
}
```

Figure 3: KISS99: Initial state selection

The basic design goal of pseudo-random number generation algorithm is mainly long cycle and pretty distribution, however, without or little consideration of safety factors. The backstepping security and prediction ability of KISS algorithm have not been proved.

#### 4.1.2. Hash Chain Algorithm

For the design of hash chain based tag generating algorithm, one can see S/Key in [RFC1760]. In the S/Key system, there is an encryption end and an authentication end. The encryption end generates an initial state  $W$ , and then uses some hash algorithm  $H()$  to iterate on  $W$  to obtain a string sequence:  $H_0(W), H_1(W), \dots, H_N(W)$ , where  $H_n(W)$  represents the iterative operation of  $H()$  on  $W$   $n$  times,  $H_0(W) = W$ . The state sequence  $\{S\}$  is defined as the reverse order of the hash chain, that is,  $S_n = H_{(N-n)}(W)$ . For example, the initial state  $S_0 = H_N(W)$  and the final state  $S_N = H_0(W) = W$ , so the transfer function  $\text{transit}()$  is represented as the inverse  $H()$ . Different from the KISS pseudo-random number generation algorithm mentioned in the previous section, in the hash chain, the tag is the state itself, that is, the output and input of  $\text{generate}()$  are consistent, and  $\text{Tag}_n = S_n$ . In the following discussion,  $S_n$  is temporarily used instead of  $\text{Tag}_n$  for the convenience of expression.

The encryption end sends the initial state  $S_0$  to the verification end, and maintains  $S_1 \sim S_n$ , which is also the tag sequence used. The encryption end sends  $S_{(n+1)}$  to the verification end every time. The verification end uses the  $S_n$  maintained by itself to verify the tag correctness of the encryption end by calculating  $S_{(n+1)} = \text{transit}(S_n)$ . As explained above,  $\text{transit}()$  is the inversion of  $H()$ . In practice, a secure hash algorithm is usually used as  $H()$ , such as SHA-256. For these hash algorithms, it is easy to calculate  $H()$ , but it is difficult to calculate the inversion of  $H()$ . Therefore, the actual operation is as follows: after receiving  $S_{(n+1)}$ , the verification end calculates whether  $H(S_{(n+1)})$  is equal to  $S_n$ . If it is equal, the verification is successful, otherwise it fails.

Hash chain algorithm has high security. It can prevent backstepping and prediction well. Not only the attacker can't backstep or predict, but also the verification end cannot do that. The disadvantage of hash chain algorithm is that before using tags, the encryption end needs to calculate all tag sequences, and then send the last of the sequence to the verification end as the initial state. At the same time, the encryption end needs to save a complete tag sequence, although it can be deleted after each tag is used up. The cost of storage of hash chain algorithm can not be ignored.

#### 4.2. Tag Update

After the state machine is enabled, the source AD uses the initial state  $S_0$  to transfer to the state  $S_1$  through the algorithm box, and generates the tag  $\text{Tag}_1$ . In the subsequent state transition interval, the AER of the source AD uses the same tag,  $\text{Tag}_1$ , to add to the message sent from this AD to the destination AD. The source AD does not transfer from the state  $S_1$  to the state  $S_2$  until the transition interval passes, and starts to use tag  $\text{Tag}_2$ . In this cycle, the state sequence  $S_1 \sim S_N$  and tag sequence  $\text{Tag}_1 \sim \text{Tag}_N$  are experienced, in which  $\text{Tag}_1 \sim \text{Tag}_N$  are used as tags in turn and added to the message by the source AER. Similarly, the destination AER uses the same state machine to calculate the tag sequence, so as to verify the tag in the message. If the source AD and the destination AD can ensure the synchronization of the state machine, it would guarantee the synchronization of the tags. So the tags can be verified correctly.

Each state machine has an activation time and an Expiration Time. After the expiration time comes, the current state machine is deactivated. If a new state machine is available, the new state machine will be used and performs the same label verification process.

## 5. Packet Processing at AER

SAVA-X does not require the intermediate router to recognize and process the SAVA-X option, which we will describe at Section 8, as long as the intermediate router correctly implements the extension header and option processing method described in IPv6 protocol [RFC8200]. The intermediate router could correctly forward the packet regardless of its specific content even if it does not recognize the SAVA-X option well.

The border router, AER, needs to handle tag correctly. The AER of the source AD judges whether the IPv6 destination address belongs to the trust alliance. If no, the packet will be forwarded directly. If yes, the AER continues to judge the hierarchical relationship between the source AD and the member ADs to which the packet's destination IP address belongs. If the source AD and the destination AD are under the same sub-trust alliance, the AER would add the tag between the two ADs, otherwise add the AD\_V tag.

Note that the packet will not be processed at other AERs in the sub-trust alliance.

At the AER of the boundary of sub-trust alliance, the packet is classified according to the IPv6 destination address. If the destination address is not within the trust alliance, it will be forwarded directly. If the destination address belongs to this sub-trust alliance, it will be classified according to the source IP address. If the source address also belongs to this sub-trust alliance, it will be forwarded directly. If the source address does not belong to this sub-trust alliance, the AER needs to verify the sub-trust alliance tag and replace it with the AD\_V tag in this sub-trust alliance for following forwarding. If the destination IP address of packet belongs to other sub-trust alliance, it SHALL be classified according to the source address. If the source address belongs to this sub-trust alliance, verify the AD\_V tag. If consistent, replace with sub-trust alliance tag. If the source address is not in this sub-trust alliance, it will be forwarded directly. Otherwise, the packet will be discarded.

The AER of the destination AD classifies packet according to the source address of the packet to be forwarded to determine whether it originates from a member AD. If yes, enter the label check. Otherwise it will be forwarded directly. Tag verification process: if the tag carried by the packet is consistent with the tag used by the source AD, remove the tag and forward the packet. Otherwise the packet will be discarded.

### 5.1. Port Classification

In order to classify packets correctly to complete tag addition, inspection and packet forwarding, it is necessary to classify the ports (interfaces) of AER. Any connected port of AER must belong to and only belong to the following types of ports:

- \* Ingress Port: Connect to the port of non-SAVA-X router in this AD. Generally connected to IGP router in the domain.
- \* Egress Port: Connect to other AD ports.
- \* Trust Port: Connect to the port of SAVA-X router in this AD.

### 5.2. Source Address Validation

In SAVA-X, AER must check the source address of the packet. Only the packet passing the check will be subject to the Section 5.3 step, and the packet using the fake source IP address will be discarded. The source address is checked using the ingress filtering method. AER only checks the source address according to the following three rules:

- \* The packet entering an AER from the Ingress Port SHALL only carry the source address prefix belonging to this AD.
- \* The packet entering an AER from the Egress Port SHALL NOT carry the source address prefix belonging to this AD.
- \* Packets entering an AER from Trust Port are not checked.

The prefix of IP address owned by one AD SHALL be configured by the administrator or obtained from the control plane, and deployed to AER by ACS of this AD.

### 5.3. Packet Classification

It SHALL be classified after the packet entering an AER passes the source address validation. There are three types of packets: packets that SHOULD be tagged, packets that SHOULD check tags, and other messages. The judgment rules of the three packets are as follows:

- \* Packets entering AER from Ingress Port. If the source address belongs to this AD and the IPv6 destination address belongs to another AD in the same sub-trust alliance, tag must be added. If the source IP address belongs to another AD in the same sub-trust alliance and the IPv6 destination address belongs to another sub-trust alliances, the tag must be verified and replaced with the sub-trust alliance tag. Other packets are forwarded directly.
- \* Packets entering AER from the Egress Port. Tag must be checked if the source address belongs to another AD in the same sub-trust alliance and the IPv6 destination address belongs to this AD. If the source address belongs to other sub-trust alliance and the IPv6 destination address belongs to another AD in the same sub-trust alliance, the tag must be checked and replaced. And other packets can be forwarded directly.
- \* Packets entering AER from Trust Port. These packets SHOULD be forwarded directly.

The relationship between IP address and ADs SHALL be obtained from the control plane and deployed to the AER by the ACS of the AD. When the SAVA-X option of the packet received from the progress port carries the active AD number, you can skip the "mapping from address to AD number" process and directly use the AD number carried in the message.

#### 5.4. Tag Addition

AER SHOULD add destination option header and add SAVA-X option into the packet according to the requirements of IETF [RFC8200].

According to [RFC8200], the destination option header SHOULD be filled so that its length is an integer multiple of 8 bytes, including the Next Hader and Hdr Ext Len fields of the destination option header, the Next Header and Payload Length fields of the IPv6 packet header, and the upper protocol header (such as TCP, UDP, etc.). If it is necessary, AER SHOULD recalculate the Checksum field.

#### 5.5. Tag Verification

AER will process the first option with Option Type equals to the binary code of 00111011 in the destination header. We would talk more about that at Section 8.

1. If the packet does not contain destination option header or SAVA-X option. the packet SHOULD be discarded.

2. If the packet contains SAVA-X option but the parameters or tag are incorrect, the packet SHOULD be discarded.
3. If the packet contains SAVA-X option, and the parameters and tag are correct, AER must replace the tag or remove the tag when needed before forwarding the message.

In the following scenarios, the tag needs to be removed. If there are only SAVA-X option, Padl and PadN options in the destination option header of the message, AER SHOULD remove the whole destination option header. If there are other options besides SAVA-X option, Padl and PadN option in the destination option header, AER SHOULD remove SAVA-X option and adjust the alignment of other options according to the relevant protocols of IPv6. In order to removing the sava-x option, the destination option header may also be filled, or some Padl and PadN may be removed, to make its length be multiple of 8 bytes. At the same time, the Next Header field and Payload Length field deployed in the IPv6 message header, and the Checksum field of the upper protocol header (such as TCP, UDP, etc.) SHALL be rewritten as necessary.

#### 5.6. Tag Replacement

Tag needs to be replaced when packet pass through different sub-trust alliance. Tag replacement needs to be done on the AER of the boundary address domain of the sub-trust alliance. This feature is not necessary to realize on the AER of each non-boundary address domain in the sub-trust alliance.

When packet is arrived at the AER of the sub-trust alliance boundary, it is classified according to the destination address.

1. If the destination address does not belong to the trust alliance, it will be forwarded directly.
2. If the destination address belongs to this sub-trust alliance, it will be classified according to the source address of the packet.
  - \* If the source address also belongs to this sub-trust alliance, the packet will be forwarded directly.
  - \* If the source address does not belong to this sub-trust alliance, AER should verify the sub-trust alliance tag and replace it with the AD\_V tag in this sub-trust alliance for forwarding.



3. If the destination address of the packet belongs to other sub-trust alliance, it shall be classified according to the source address.
  - \* If the source address belongs to this sub-trust alliance, AER should verify the AD\_V tag and replace the tag with sub-trust alliance tag when it is consistent.
  - \* If the source address is not in this sub-trust alliance, it will be forwarded directly.
4. Otherwise, the packet will be discarded.

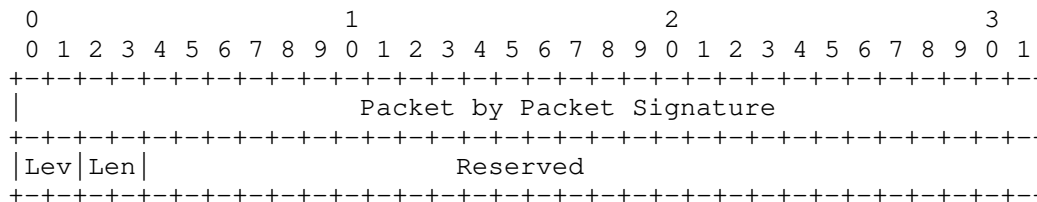
Alliance tag will be used when the packet crosses the upper AD which is at the higher level of source AD and destination AD. Alliance tag is the tag maintained between the source AD corresponding to the AD in the parent AD and the destination AD corresponding to the address domain in the parent AD.

## 6. Packet Signature

It is difficult to accurately synchronize time among the trust alliance members. So we propose a shared time slice, which means that there are two tags effecting at the same time in a period of time. But it may suffer from replay attack. Therefore, a packet signature mechanism is proposed to prevent replay attack and cancel the original tag.

Tag is time-dependent. The state machine triggers state transition by time and generates a new tag. In a short period of time, all data packets are labeled with the same tag. Moreover, due to the subtle differences in time synchronization, both old and new tags can be used for this short period of time, so attackers can reuse tags for replay attack by simply copying tags.

The packet signature mechanism joins 8-bit part of the payload in the packet and the tags generated by the state machine. And then it calculates hash value with parameters above to achieve the effect of packet by packet signature and resist the attackers reuse of tags. Its processing flow is shown below.



Packet by Packet Signature: Hash value of original tag, source address and destination address and first 8-bit of payload, credible level and credible prefix length.

Lev: 2-bit of credible level.

Len: 7-bit of credible prefix length.

Reserved: 23-bit of reserved field. 0 will be padded.

Firstly, it takes the source address, destination address and the first 8-bit of the data part of the data packet from the data packet, joins them in the way of (src-ip, dst-ip, first 8-bit of payload), and then joins the tag generated by the state machine at this time, the credible level of the SAVA architecture adopted by this AD and the length of the credible prefix to hash the concatenated string with the hash algorithm to get a new message digest. Then it is reduced to 32-bit packet signature by clipping and folding algorithm. The AER adds the 32-bit packet signature together with the 2-bit credible level and the 7-bit credible prefix length to the SAVA-X option, fills the option into 64-bit, and forwards it. At the AER of the destination AD, the same splicing and the same hash operation are performed to verify whether the generated string is consistent with the signature of the data packet. If they are consistent, they are forwarded. Otherwise, it is considered that the source address is forged and the data packet is discarded.

Due to the problem of time synchronization, when both old and new tags are valid, both old and new tags need to be verified. As long as one of them passes the verification, the packet should be forwarded. The original tag generated by the state machine will not appear in the packet. The attackers does not know the tag generated by the state machine at this time, so they can not forge the packet signature in the same way, which ensures the security of the data communication plane.

## 7. Security Consideration

This present memo doesnot find any security problem.

## 8. IANA Considerations

SAVA-X is designed for IPv6 enabled networks. It takes a destination option, SAVA-X option, header to carry the Tag. We recommend to use 00111011, i.e. 59, for SAVA-X option. Here we give our SAVA-X option format in use. 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  
 +-+ |

```

Option Type | Opt Data Len | Tag Len | AI Type | Reserved |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
TAG ~
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
Additional Information ~
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Option Type: 8-bit field. The destination option type of SAVA-X = 59.

Opt Data Len: 8-bit field. The bytes length of SAVA-X option. Its value is  $2 + \text{LenOfAI} + (\text{TagLen} + 1)$ , where LenOfAI is 2 when AI Type is 1, or 4 when AI Type is 2, or 0 default.

Tag Len: 4-bit field. The bytes length of TAG equals to  $(\text{Tag Len} + 1) * 8$ , e.g. if Tag Len = 7, it means SAVA-X uses 64 bits long TAG. It guarantees the length of TAG would be an integral multiple of 8 bits. The maximum length of TAG is 128 bits and the minimum length of TAG is 32 bits.

AI Type: 4-bit field. The type of Additional Information. 0 for no Additional Information, 1 for 16-bit long Additional Information and 2 for 32-bit long Additional Information. Others are not assigned.

Reserverd: These bits are not used now and must be zero.

TAG: Variable-length field The actual tag, its length is determined by Tag Len field.

Additional Information: As defined in AI Type field.

## 9. Acknowledgements

Much of the content of this document is the expansion of the IETF [RFC5210] in inter-domain level. Thanks to the effort of pioneers.

## 10. Normative References

- [RFC1760] Haller, N., "The S/KEY One-Time Password System", RFC 1760, DOI 10.17487/RFC1760, February 1995, <<https://www.rfc-editor.org/info/rfc1760>>.
- [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>>.

- [RFC5210] Wu, J., Bi, J., Li, X., Ren, G., Xu, K., and M. Williams,  
"A Source Address Validation Architecture (SAVA) Testbed  
and Deployment Experience", RFC 5210,  
DOI 10.17487/RFC5210, June 2008,  
<<https://www.rfc-editor.org/info/rfc5210>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6  
(IPv6) Specification", STD 86, RFC 8200,  
DOI 10.17487/RFC8200, July 2017,  
<<https://www.rfc-editor.org/info/rfc8200>>.

## Authors' Addresses

Ke Xu  
Computer Science, Tsinghua University  
Qinghuayuan street, Haidian District  
Beijing  
100084  
China

Email: [xuke@tsinghua.edu.cn](mailto:xuke@tsinghua.edu.cn)

Xiaoliang Wang  
Computer Science, Tsinghua University  
Qinghuayuan street, Haidian District  
Beijing  
100084  
China

Email: [wangxiaoliang0623@foxmail.com](mailto:wangxiaoliang0623@foxmail.com)

Yangfei Guo  
Institute for Network Sciences and Cyberspace, Tsinghua University  
Qinghuayuan street, Haidian District  
Beijing  
100084  
China

Email: [guoyangf19@mails.tsinghua.edu.cn](mailto:guoyangf19@mails.tsinghua.edu.cn)

Internet Area Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 26 April 2022

K. Xu  
X. Wang  
Y. Guo  
Tsinghua University  
23 October 2021

Communication Protocol Between the AD Control Server and the AD Edge  
Router of Inter-Domain Source Address Validation Architecture  
draft-guo-intarea-savax-protocol-00

## Abstract

Because the Internet forwards packets according to the IP destination address, packet forwarding typically takes place without inspection of the source address and malicious attacks have been launched using spoofed source addresses. The inter-domain source address validation architecture is an effort to enhance the Internet by using state machine to generate consistent tags. When communicating between two end hosts at different ADs of IPv6 network, tags will be added to the packets to identify the authenticity of the IPv6 source address.

This memo focus on the data plane of the SAVA-X mechanism.

## Status of This Memo

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

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

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

This Internet-Draft will expire on 26 April 2022.

## Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Terminology and Abbreviation . . . . .	3
3. Communication Protocol Format . . . . .	4
4. ACS-ACS Communication Protocol . . . . .	7
4.1. Announcement, Query and Response of State Machine Information . . . . .	7
4.1.1. State Machine Information Announcement . . . . .	9
4.1.2. State Machine Information Request . . . . .	11
4.2. Request and Response of Diagnose Information . . . . .	13
5. ACS-AER Communication Protocol . . . . .	16
5.1. Deployment, Request and Response of AD Registration information . . . . .	16
5.1.1. Deployment of AD Registration Information . . . . .	16
5.1.2. Request of AD Registration Information . . . . .	18
5.1.3. Reponse of AD Registration Information . . . . .	19
5.2. Deployment, Request and Reply of AD Prefix Information . . . . .	22
5.2.1. Deployment of AD Prefix Information . . . . .	22
5.2.2. Request of AD Prefix Information . . . . .	25
5.2.3. Response of AD Prefix Information . . . . .	27
5.3. Deployment, Request and Response of State Machine Information . . . . .	30
5.3.1. Deployment of State Machine Information . . . . .	30
5.3.2. Request of State Machine Information . . . . .	33
5.3.3. Response of State Machine Information . . . . .	35
5.4. Request and Response of Keep-alive Information . . . . .	38
5.4.1. Response of Keep-alive Information . . . . .	39
6. Deployment of Tag Information . . . . .	40
7. Security Consideration . . . . .	42
8. IANA Consideration . . . . .	42
9. Acknowledgements . . . . .	43
10. References . . . . .	43
10.1. Normative References . . . . .	43
10.2. Informative References . . . . .	43
Authors' Addresses . . . . .	43

## 1. Introduction

The Inter-Domain Source Address Validation Architecture (SAVA-X) mechanism establishes a trust alliance among Address Domains (AD), maintains a one-to-one state machine among ADs, generates a consistent tag, and deploys the tag to the ADs' border router (AER). The AER of the source AD adds a tag to identify the identity of the AD to the packet originating from one AD and sinking in another AD. The AER of the destination AD verifies the source address by validating the correctness of the tag to determine whether it is a packet with a forged source address.

In the process of packet forwarding, if the source address and the destination address of this packet both are addresses in the trust alliance, however the tag is not added or incorrectly added, AER of the destination AD determines that the source address is forged and directly discards this packet. The destination AD forwards the packet directly for packets whose source address is an address outside the trust alliance.

This document mainly studies the relevant specifications of the data plane of the inter-domain source address validation architecture mechanism between ADs, which will protect IPv6 networks from being forged source address. You could see [RFC8200] for more details about IPv6. It stipulates the state machine, tag generation and update, tag processing in AER, and packet signature. Its promotion and application can realize the standardization of the data plane in the SAVA-X to facilitate the related equipment developed by different manufacturers and organizations to cooperate to accomplish the inter-domain source address validation jointly.

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, BCP 14 [RFC2119] and indicate requirement levels for compliant CoAP implementations.

## 2. Terminology and Abbreviation

Abbreviation	Description
AD	Address Domain, the unit of a trust alliance, which is an address set consisting of all IPv6 addresses corresponding to an IPv6 address prefix.
TA	Trust Alliance, the IPv6 network that

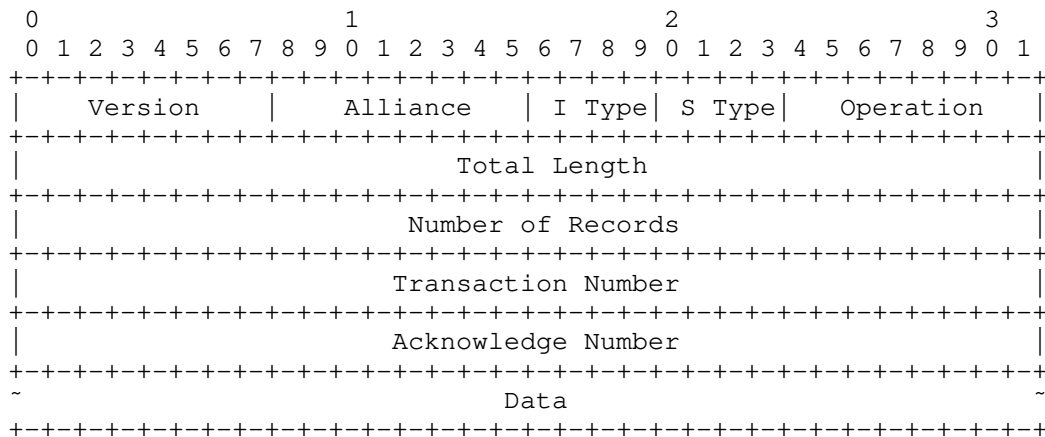
	uses the SAVA-X mechanism.
ACS	AD Control Server, the server that maintains state machine with other ACS and distribute information to AER.
AER	AD border router, which is placed at the boundary of an AD of STA.
ADID	The identity of an AD.
ADID_Rec	The record of a number of an AD.
ARI_Rec	The record with relevant information of an AD or STA.
API_Rec	The record of prefix of an AD or STA.
SM	State Machine, which is maintained by a pair of ACS to generate tags.
SMI_Rec	The record of the state machine information.
Tag	The authentic identification of source address of a packet.

Table 1

### 3. Communication Protocol Format

Every AD should be placed at least one ACS, which is mainly responsible for maintaining the relationship between ADs of the trust alliance, establishing connections with other ACS, maintaining the synchronous state machine, and sending the generated tags to the AER.





Version: 8-bit, the current version=0b1 of SAVA-X.

Alliance: 8-bit, the sub-trust alliance number.

I Type: 4-bit, Information type, 0 for G\_REF\_INFO, 1 for AD\_REG\_INFO, 2 for AD\_PREFIX\_INFO, 3 for STATE\_MACHINE\_INFO, 4 for DIAGNOSIS\_INFO, 5 for RUNNING\_STATE\_INFO, 6 for STRATEGY\_INFO, 7 for ALIVE\_INFO, 8 for TAG\_INFO, 9 for ALLI\_TAG\_INFO, 10 for AD\_V\_TAG\_INFO and others are unassigned.

S Type: 4-bit, Session type, 1 for ANNOUNCEMENT or DEPLOYMENT, 2 for REQUEST, 3 for REQUEST\_ALL, 4 for ACK, 5 for NAK, 6 for AACK, 7 for ANAK, 8 for RACK, 9 for RNAK and others are unassigned.

Operation: 8-bit, the first 3 bits means for whether RENEW Type or not. First bit: 0 for non-RENEW packet, 1 for RENEW packet. Second bit: 0 for the first non-RENEW packet, 1 for the first RENEW packet. Third bit: 0 for the last non-RENEW packet, 1 for the last RENEW packet.

Total Length: 32-bit, the length of this packet: from Version to Data.

Number of Records: 32-bit, he records in Data.

Transaction Number: 32-bit, this is the identification of a publication, query or response, and the value should increase monotonically. And different I Types MUST have its own Transaction Number.

Acknowledge Number: 32-bit, it is only be filled when S Type is ACK,

NAK, AACK, ANAK, RACK or RNAK. Otherwise it should be filled as 0.

Data: Variable-length field. I Type and S Type specifies data jointly.

When S Type is ANNOUNCEMENT:

- \* If I Type = AD\_REG\_INFO, Data field SHOULD be one or more ARI\_Rec.
- \* If I Type = AD\_PREFIX\_INFO, Data field SHOULD be one or more API\_Rec.
- \* If I Type = STATE\_MACHINE\_INFO, Data field SHOULD be one or more SMI\_Rec.
- \* If I Type = TAG\_INFO, ALLI\_TAG\_INFO or AD\_V\_TAG\_INFO, Data field SHOULD be one or more TAG\_Rec.

When S Type is REQUEST or REQUEST\_ALL:

- \* If I Type = REG\_INFO, Data field SHOULD be one or more ADID\_Rec.
- \* If I Type = AD\_PREFIX\_INFO, Data field SHOULD be none or one or more ADID\_Rec.
- \* If I Type = STATE\_MACHINE\_INFO, Data field SHOULD be none or one or more ADID\_Rec.
- \* If I Type = DIAGNOSE\_INFO, Data field SHOULD be a 32-bit diagnose request code.
- \* If I Type = ALIVE\_INFO, Data field SHOULD be none.

When S Type is ACK, AACK or RACK:

- \* If I Type = REG\_INFO, Data field SHOULD be one or more ARI\_Rec.
- \* If I Type = AD\_PREFIX\_INFO, Data field SHOULD be one or more API\_Rec.
- \* If I Type = STATE\_MACHINE\_INFO, Data field SHOULD be one or more SMI\_Rec.
- \* If I Type = DIAGNOSE\_INFO, Data field SHOULD be one 32-bit diagnose response code.
- \* If I Type = ALIVE\_INFO, Data field SHOULD be none.

When S Type is NAK, ANAK or RNAK, Data field SHOULD be one 32-bit error code:

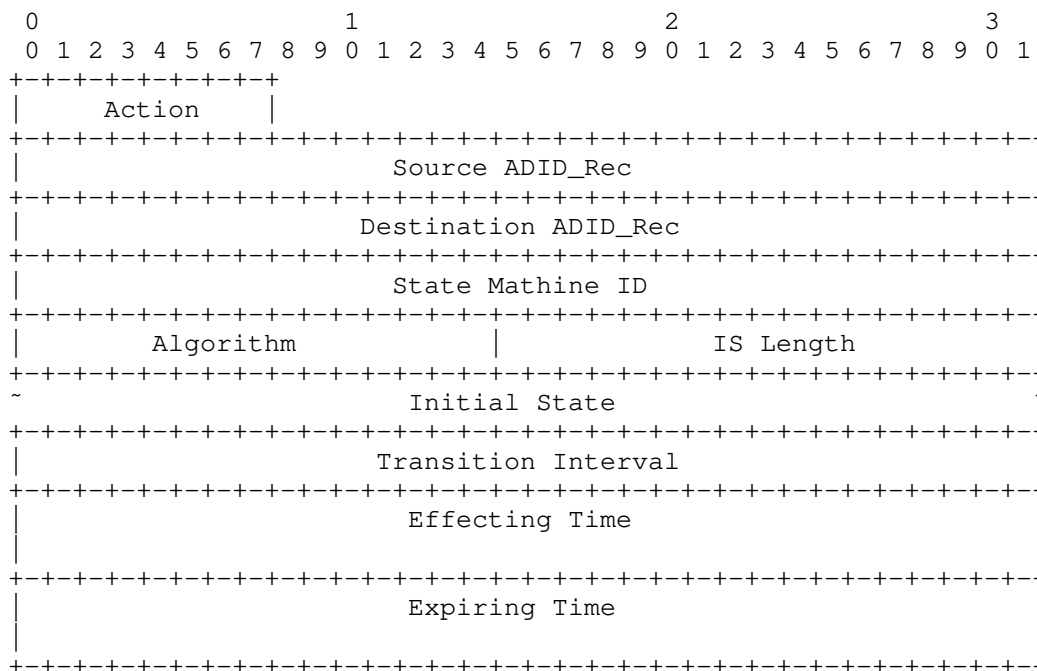
- \* 1 for parameters are wrong which means the packet cannot resolve correctly.
- \* 2 for member AD(s) in request packet does not exist in the designative sub-trust alliance.
- \* 3 for algorithm for State Machine set by source ACS cannot support by the destination ACS.

#### 4. ACS-ACS Communication Protocol

Since the blockchain is adopted in SAVA-X to maintain the information of the trust alliance, ACS can query the address domain information of relevant ADes of the trust alliance and the AD prefix information corresponding to the address domain from the blockchain.

##### 4.1. Announcement, Query and Response of State Machine Information

State machine information record (SMI\_Rec) represents the packet format used when state machine is negotiated between different ordered pairs of ADs. When an ordered pair of ADs is negotiating the state machine, ACS of AD with smaller ADID initiates the communication and ACS of AD with larger ADID uses SMI\_Rec determines the information to be used, such as initial state, tag generation algorithm, state transition interval, etc. Compared to ARI\_Rec and API\_Rec, SMI\_Rec also needs an Expiring Time in addition to the Effecting Time. Expiration Time stands when the negotiated state machine is no longer valid.



Action: 8-bit, 1 for add or update this SMI\_Rec.

Source ADID\_Rec: Variable-length field. Refer to ADID\_Rec [SAVA-X-Control].

Destination ADID\_Rec: Variable-length field. Refer to ADID\_Rec in [SAVA-X-Control].

State Mathine ID: 32-bit, the ID used to identify the state machine, which is unique to a specific ordered AD pair and grows monotonically in use. It is used to distinguish the sequence before and after the generation of multiple state machines.

Algorithm: 16-bit, algorithm used in A-Box. 1 for KISS-99 32-bit, 2 for KISS-99 64-bit Joint, 3 for OTP-2289 MD5 and others are unassigned.

IS Length: 16-bit, the length of Initial State field.

Initial State: Variable-length field, the length of this filed is determined by IS Length.

Transition Interval: 32-bit, the milliseconds of interval of state transition.

Effecting Time: 64-bit, when this field is 0, it means this State Machine should be enabled after the last State Machine expired.

Expiring Time: 64-bit, the end of this State Machine.

#### 4.1.1. State Machine Information Announcement

State machine information announcement (SM\_INFO-Announce) is sent from source ACS to destination ACS. Source ACS fills in the following values for each field:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	SM_INFO
S Type	ANNOUNCEMENT
Operation	NULL: source ACS updates part of the state machines information to destination ACS. RENEW: source ACS updates all the state machines information to destination ACS.
Total Length	The length of this message.
Number of Records	The number of SMI_Recs in Data field.
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent out where I Type is SM_INFO and ACS would keep it increasing monotonic.
Acknowledgement Number	0
Data	One or more SMI_Recs.

Table 2

All SMI\_Recs in Data field should have an unique SM\_ID. When Action is ADD and SM\_ID bigger than current used SM\_ID, ACS should add the state machine defined in SMI\_Rec. When Action is ADD and SM\_ID

equals to current used SM\_ID, ACS should modify the state machine defined in SMI\_Rec. Only Transition Interval and Expiring Time can be modified. Other SMI\_Rec should be discarded and destination ACS should send a NAK message to source ACS.

When receiving a non-RENEW packet, if it cannot resolve this message, destination ACS should send a NAK message to source ACS. When destination ACS can resolve the packet correctly, it SHOULD:

1. Compare the Transaction Number in this packet with Transaction Number received from the same ACS before. Otherwise, destination ACS would discard this packet and send a SM\_INFO-Request to request the latest information of state machine. SM\_INFO-Request is defined at Section 4.1.2. If bigger, destination ACS WOULD:
2. Accept every SMI\_Rec and process them as following: - If the SM\_ID in SMI\_Rec equals to current used SM\_ID, destination ACS would update the current used SM\_ID. - If the SM\_ID in SMI\_Rec bigger than current used SM\_ID, destination ACS would add this state machine to its following used state machine list.
3. And then destination ACS will send an SM\_INFO-AACK message to source ACS.

When receiving a RENEW packet, if it cannot resolve this message, destination ACS should send a SM\_INFO-ANAK message to source ACS. When destination ACS can resolve the packet correctly, it SHOULD:

1. Compare the Transaction Number in this packet with Transaction Number received from the same ACS before. Otherwise, destination ACS would discard this packet and send a SM\_INFO-Request to request the latest information of state machine. If bigger, destination ACS WOULD:
2. Accept every SMI\_Rec and process them as following: - If the SM\_ID in SMI\_Rec equals to current used SM\_ID, destination ACS would update the current used SM\_ID. - If the SM\_ID in SMI\_Rec bigger than current used SM\_ID, destination ACS would add this state machine to its following used state machine list. Especially, state machines will be removed right now when they are not listed in the SMI\_Recs but they are in using.
3. And then destination ACS will send an SM\_INFO-AACK message to source ACS.

There are two types of reply of SM\_INFO-Annouce message. That is SM\_INFO-AACK representing affirmative acknowledgement and SM\_INFO-ANAK representing negative acknowledgement. These are sent from destination ACS to source ACS. The mainly part of packet is filled by destination ACS as follows:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	SM_INFO
S Type	AACK if it is affirmative acknowledgement or ANAK if it is negative acknowledgement.
Operation	NULL
Total Length	The length of this message.
Number of Records	0
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent out where I Type is SM_INFO and ACS would keep it increasing monotonic.
Acknowledgement Number	The Transaction Number of the response corresponding request.
Data	S Type = AACK: None. S Type = ANAK: a 32-bit error code defined in Section 3.

Table 3

Nothing needs to do when source ACS receives an SM\_INFO-AACK message while it should regenerate a new state machine and announces to destination ACS when source ACS receives an SM\_INFO-ANAK message.

#### 4.1.2. State Machine Information Request

State machine information request (SM\_INFO-Request) is sent from source ACS to destination ACS. Source ACS fills in the following values for each field:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	SM_INFO
S Type	REQUEST
Operation	NULL: announce all state machine information to source ACS.
Total Length	The length of this message.
Number of Records	0
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent out where I Type is SM_INFO and ACS would keep it increasing monotonic.
Acknowledgement Number	0
Data	None

Table 4

When source ACS receives a SM\_INFO-Request message, it would send a SM\_INFO-RNAK message to destination ACS if some fields are wrong. Otherwise, source ACS would send an SM\_INFO-RACK message to destination ACS and process this SM\_INFO-Request message. Source ACS should compare the Transaction Number in this message with Transaction Number received from the same destination ACS before. Otherwise, source ACS would discard this packet. If bigger, source ACS would send an SM\_INFO-RACK message to destination ACS.



There are two types of reply of SM\_INFO-Request message, i.e. SM\_INFO-RACK representing affirmative acknowledgement and SM\_INFO-RNAK representing negative acknowledgement. These are sent from source ACS to destination ACS. The mainly part of packet is filled by source ACS as follows: I Type is SM\_INFO. S Type is RACK if it is affirmative acknowledgement or RNAK if it is negative acknowledgement. Operation is NULL. When S Type is RACK, Data field is a few of SMI\_Recs. When S Type is RNAK, Data field is a 32-bit error code.

When receiving a SM\_INFO-RACK message, if it cannot resolve this message, destination ACS should send a SM\_INFO-Request message to source ACS to acquire another state machine. When destination ACS can resolve the message correctly, it SHOULD:

1. Compare the Transaction Number in this packet with Transaction Number received from the same source ACS before. Otherwise, destination ACS would discard this packet and send a SM\_INFO-Request to request the latest information of state machine. If bigger, destination ACS WOULD:
2. Accept every SMI\_Rec and process them as following: - If the SM\_ID in SMI\_Rec equals to current used SM\_ID, destination ACS would update the current used SM\_ID. - If the SM\_ID in SMI\_Rec bigger than current used SM\_ID, destination ACS would add this state machine to its following used state machine list.
3. And then destination ACS will send an SM\_INFO-AACK message to source ACS.

When receiving a SM\_INFO-RNAK message, if it cannot resolve this message, destination ACS should send a SM\_INFO-Request message to source ACS to acquire new state machine. When destination ACS can resolve the message correctly, it SHOULD compare the Transaction Number in this packet with Transaction Number received from the same source ACS before. Otherwise, destination ACS would discard this packet and send a SM\_INFO-Request to request the latest information of state machine. If bigger, destination ACS WOULD send a new correct SM\_INFO-Request message to source ACS.

#### 4.2. Request and Response of Diagnose Information

Sent by destination ACS, request of diagnose information (DIAG\_INFO-Request) is used to require the source ACS to check its configuration and source AERs' settings. Source ACS will response with its result. Destination ACS fills in the following values for each field:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	DIAG_INFO
S Type	REQUEST
Operation	NULL
Total Length	The length of this message.
Number of Records	0
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent out where I Type is DIAG_INFO and ACS would keep it increasing monotonic.
Acknowledgement Number	0
Data	a 32-bit error code defined below.

Table 5

Response of diagnose information (DIAG\_INFO-Response) replies from source ACS to destination ACS.

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	DIAG_INFO
S Type	ACK
Operation	NULL
Total Length	The length of this message.
Number of Records	0
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent out where I Type is DIAG_INFO and ACS would keep it increasing monotonic.
Acknowledgement Number	The Transaction Number of the response corresponding request.
Data	a 32-bit error code defined below.

Table 6

Before it sends the DIAG\_INFO-Request message, the destination ACS should check its own configuration and gurantee they are correct.

If it receives a DIAG\_INFO-Request message, the source ACS would check the communication with its own AER whether correct or not.

1. If it's wrong, source ACS would reply a DIAG\_INFO-Response message in which its Data filed is filled with 2 for fault cannot be repaired and alarm to administrator to deal with this problem.
2. If it's right, source ACS would RENEW all the registration information, prefix information and state machine information to its all AERs. After that, source ACS will reply a DIAG\_INFO-Response message in which its Data filed is filled with 1 for all runs correctly after repairing.

## 5. ACS-AER Communication Protocol

ACS would periodically deploy AD registration information, AD prefix information, and state machine information of relevant ADes to its all AERs to guarantee all information are latest. And ACS also would deploy the tag information to its all AERs periodically.

### 5.1. Deployment, Request and Response of AD Registration information

#### 5.1.1. Deployment of AD Registration Information

After connecting with AER, ACS deploys the AD Registration Information (REG\_INFO-Deploy) to AER periodically. I Type is REG\_INFO. S Type is Announcement. Operation is NULL when some ADes' information are joined, left or updated and Operation is RENEW when all ADes' information are deployed. Acknowledgement is 0. Data field is one or more ARI\_Rec.

It should be noted that when there are two ARI\_Recs in Data fields responding to the same AD, one may effect right now and the other effects after passing Effecting Time. When AER receives this message, all of them should be restored in the trust alliance list and AER MUST process them orderly. Since the protocol processes the records in sequence, it is required that the ARI\_Rec effecting at the current time for the same member AD should appear in front of another updating ARI\_Rec.

When receiving a non-RENEW packet, if it cannot resolve this message, AER could send a REG\_INFO-Request message to acquire the latest AD registration information.

When AER can resolve this message correctly, it SHOULD:

1. Compare the Transaction Number in this packet with Transaction Number received from the same ACS before. If bigger, AER WOULD accept every ARI\_Rec and process them as follows. Otherwise, AER would discard this packet and send a REG\_INFO-RequestAll message to acquire the latest information of AD registration information.
2. Process every ARI\_Rec: - If Action is ADD and the record does not exist in its maintained trust alliance list, AER would add this record to its trust alliance list. - If Action is ADD and the record exists in its maintained trust alliance list but ACS Address is changed, AER would add this record to its trust alliance list and delete original record after passing Effecting Time in this ARI\_Rec. - If Action is ADD and the record exists in its maintained trust alliance list and ACS Address is not

changed, AER would do nothing. - If Action is DEL and the record exists in its maintained trust alliance list, AER would remove this record from its trust alliance list after passing Effecting Time in this ARI\_Rec.

3. If a change is made in step 2, the update should take effect after passing the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than the current time or is all 0, it will take effect immediately.

AER acts as following when receiving a RENEW packet. When ACS initiates RENEW, it would send a RENEW message with which the first bit of Operation field is 1. The second bit of Operation field identifies the begin of a procedure of RENEW and the third bit of Operation field identifies the end of a procedure of RENEW. ACS MUST NOT send a RENEW packet with which the first bit of Operation field is 0 in RENEWing. AER MUST process this procedure of RENEW after received all RENEW packets. When AER can resolve this packet correctly, it SHOULD:

1. Compare the Transaction Number in this packet with Transaction Number received from the same ACS before. If bigger, AER would accept every ARI\_Rec and process them as follows. Otherwise, AER would discard this packet and send a REG\_INFO-RequestAll message to acquire the latest information of AD registration information.
2. Process every ARI\_Rec: - If the record does not exist in its maintained trust alliance list, AER would add this record to its trust alliance list. - If the record exists in its maintained trust alliance list but ACS Address is changed, AER would add this record to its trust alliance list and delete original record after passing Effecting Time in this ARI\_Rec. - If the record exists in its maintained trust alliance list and ACS Address is not changed, AER would do nothing. - If there are some records in the original trust alliance list that do not appear in the Data field during this RENEW process, they will be deleted immediately.
3. If a change is made in step 2, the update should take effect after passing the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than the current time or is all 0, it will take effect immediately.

## 5.1.2. Request of AD Registration Information

The request is sent by AER to ACS. There are two types of request of AD Registration Information message. When querying the information of all member ADs of the trust alliance, the type is REG\_INFO-RequestAll and REG\_INFO-Request is used when querying the information of partial member ADs of the trust alliance.

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	REG_INFO
S Type	REQUEST: for querying partial member ADs and S Type is REQUEST_ALL: for querying all member ADs.
Operation	NULL
Total Length	The length of this message.
Number of Records	S Type = REQUEST: the number of ADID_Recs in Data field. S Type = REQUEST_ALL: 0.
Transaction Number	The last Transaction Number add 1. AER would maintain a global Transaction Number for packets sent out to ACS where I Type is REG_INFO and AER would keep it increasing monotonic.
Acknowledgement Number	0
Data	S Type = REQUEST: one or more ADID_Recs. S Type = REQUEST_ALL: None.

Table 7

When processing REG\_INFO-Request(ALL) message, ACS would reply REG\_INFO-NAK to AER if it holds some fields are wrong. For example, AER requests one ARI\_Rec that does not exist. Otherwise, REG\_INFO-ACK message will be replied. ACS WOULD process as follows:

1. ACS SHOULD compare the Transaction Number in this packet with Transaction Number received from the same AER before. If bigger, ACS would process as step 2. Otherwise, AER WOULD discard this packet and send a REG\_INFO-NAK message to AER.
2. ACS processes every ADID\_Rec. If the AD exists in its maintained trust alliance list, ACS would mark this record as "Reply". Otherwise ACS would mark this record as "Negative Reply". Especially, all records would be marked with "Reply" when Operation field is REQUEST\_ALL.
3. If any case in step 2 is marked with "Negative Reply", ACS would construct a REG\_INFO-NAK message to reply to the AER. Otherwise, a REG\_INFO-ACK message is constructed to reply the AD registration information of all members marked with "Reply" to the AER.

#### 5.1.3. Reponse of AD Registration Information

AD registration information response includes two types. That is REG\_INFO-ACK and REG\_INFO-NAK. ACS will reply to AER according to the request of registration information sent by AER to ACS.

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	REG_INFO
S Type	ACK: representing affirmative acknowledgement. NAK: representing negative acknowledgement.
Operation	NULL: REG_INFO-Request message. RENEW: REG_INFO-RequestAll.
Total Length	The length of this message.
Number of Records	S Type = ACK: the number of ARI_Recs in Data field. S Type = REQUEST_ALL: 0.
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent out to AER where I Type is REG_INFO and ACS would keep it increasing monotonic.
Acknowledgement Number	The Transaction Number of the response corresponding request.
Data	S Type = ACK: one or more ARI_Recs. S Type = NAK: a 32-bit error code defined at Section 3. There is no boundary identification between these ARI_Recs, which requires that the implementation of the protocol can process each record sequentially until the end of this message.

Table 8



It should be noted that when there are two ARI\_Recs in Data fields responding to the same AD, one may effect right now and the other effects after passing Effecting Time. When AER receives this message, all of them should be restored in the trust alliance list and AER MUST process them orderly. Since the protocol processes the records in sequence, it is required that the ARI\_Rec effecting at the current time for the same member AD should appear in front of another updating ARI\_Rec.

When receiving a non-RENEW REG\_INFO-ACK message, if it holds that some fields are wrong, AER could send a REG\_INFO-RequestAll message to acquire the latest AD registration information. Otherwise, AER would act as follows.

1. AER SHOULD compare the Transaction Number in this packet with Transaction Number received from the same ACS before. If bigger, AER would process them as follows. Otherwise, AER would discard this packet and send a REG\_INFO-RequestAll message to acquire the latest information of AD registration information.
2. AER WOULD processes every ARI\_Rec: - If Action is ADD and the record does not exist in its maintained trust alliance list, AER would add this record to its trust alliance list. - If Action is ADD and the record exists in its maintained trust alliance list but ACS Address is changed, AER would add this record to its trust alliance list and delete original record after passing Effecting Time in this ARI\_Rec. - If Action is ADD and the record exists in its maintained trust alliance list and ACS Address is not changed, AER would do nothing. - If Action is DEL and the record exists in its maintained trust alliance list, AER would remove this record from its trust alliance list after passing Effecting Time in this ARI\_Rec.
3. If a change is made in step 2, the update should take effect after passing the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than the current time or is all 0, it will take effect immediately.

AER acts as following when receiving a RENEW REG\_INFO-ACK message. When ACS initiates RENEW, it would send a RENEW meesge with which the first bit of Operation field is 1. The second bit of Operation field identifies the begin of a procedure of RENEW and the third bit of Operation field identifies the end of a procedure of RENEW. ACS MUST NOT send a RENEW packet with which the first bit of Operation field is 0 in RENEWing. AER MUST process this procedure of RENEW after received all RENEW packets. When AER can resolve this packet correctly, it SHOULD:

1. Compare the Transaction Number in this packet with Transaction Number received from the same ACS before. If bigger, AER would accept every ARI\_Rec and process them as step 2. Otherwise, AER would discard this packet and send a REG\_INFO-RequestAll message to acquire the latest information of AD registration information.
2. Process every ARI\_Rec: - If the record does not exist in its maintained trust alliance list, AER would add this record to its trust alliance list. - If the record exists in its maintained trust alliance list but ACS Address is changed, AER would add this record to its trust alliance list and delete original record after passing Effecting Time in this ARI\_Rec. - If the record exists in its maintained trust alliance list and ACS Address is not changed, AER would do nothing. -If there are some records in the original trust alliance list that do not appear in the Data field during this RENEW process, they will be deleted immediately.
3. If a change is made in step 2, the update should take effect after passing the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than the current time or is all 0, it will take effect immediately.

When AER receives an REG\_INFO-NAK message, it could send a REG\_INFO-RequestAll message to ACS to acquire the latest AD registration information.

## 5.2. Deployment, Request and Reply of AD Prefix Information

### 5.2.1. Deployment of AD Prefix Information

AD prefix information deployment (PFX\_INFO-Deploy) is sent from ACS to AER. ACS fills in the following values for each field:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	AD_PREFIX_INFO
S Type	DEPLOYMENT
Operation	NULL: to publish partial update information of member ADs' prefix. RENEW: to publish all member ADs' prefix.
Total Length	The length of this message.
Number of Records	The number of API_Recs in Data field.
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent out to AER where I Type is AD_PREFIX_INFO and ACS would keep it increasing monotonic.
Acknowledgement Number	0
Data	One or more API_Recs. There is no boundary identification between these API_Recs, which requires that the implementation of the protocol can process each record sequentially until the end of this message.

Table 9

It should be noted that when there are two ARI\_Recs in Data fields responding to the same AD, one may effect right now and the other is update message for ADD or DEL effecting after the Effecting Time. For example, if the current time is 5 and there are two records corresponding to the prefix P, in which the Effecting Time of record R1 is 1, the action is ADD, the Effecting Time of record R2 is 7 and the action is DEL, then it indicates that the prefix P is currently valid effective from time 1 and becomes invalid at time 7. When ACS or AER receives this message, all of them should be restored in the database and ACS should send them all when deploying. Since the

protocol processes the records in sequence, it is required that the API\_Rec effecting at the current time for the same member AD should appear in front of another updating API\_Rec.

When receiving a non-RENEW PFX\_INFO-Deploy message, if it holds that some fields are wrong, for example, it requires to delete a API\_Rec that does not exist or to add some prefix that is conflict with other member ADs, AER could send a request message to acquire the latest AD prefix information. Otherwise, AER would act as follows.

1. AER SHOULD compare the Transaction Number in this packet with Transaction Number received from the same ACS before. If bigger, AER WOULD process them as step 2. Otherwise, AER would discard this packet and send a PFX\_INFO-RequestAll message to acquire the latest information of AD prefix information.
2. AER processes every API\_Rec: - If Action is ADD and the record does not exist in its maintained prefix list, AER would add this record to its prefix list. - If Action is ADD and the record exists in its maintained prefix list, AER would do nothing. - If Action is DEL and the record exists in its maintained prefix list, AER would remove this record from its prefix list after Effecting Time.
3. If a change is made in step 2, the update should take effect after the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than the current time or is all 0, it will take effect immediately.

AER acts as following when receiving a RENEW PFX\_INFO-Deploy message. When ACS initiates RENEW, it would send a RENEW message with which the first bit of Operation field is 1. The second bit of Operation field identifies the begin of a procedure of RENEW and the third bit of Operation field identifies the end of a procedure of RENEW. ACS MUST NOT send a RENEW packet with which the first bit of Operation field is 0 in RENEWing. AER SHOULD uniformly process all packets in this RENEW process after receiving all RENEW packets.

1. AER SHOULD compare the Transaction Number in this packet with Transaction Number received from the same ACS before. If bigger, AER WOULD process as step 2. Otherwise, AER would discard this message and send a PFX\_INFO-RequestAll message to acquire the latest information of AD prefix information.
2. AER processes every API\_Rec: - If the record does not exist in its maintained prefix list, AER would add this record to its trust alliance list. - If the record exists in its maintained prefix list, AER would do nothing. - If there are some records

in the original prefix list that do not appear in the Data field during this RENEW process, these records will be deleted immediately.

3. If a change is made in step 2, the update should take effect after passing the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than the current time or is all 0, it will take effect immediately.

#### 5.2.2. Request of AD Prefix Information

AD prefix information request (PFX\_INFO-RequestAll) is sent from AER to ACS to query some member ADs', including itself, all latest AD prefix information. AER fills in the following values for each field:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	AD_PREFIX_INFO
S Type	REQUEST_ALL: querying from ACS the latest AD prefix information of all member ADs.
Operation	NULL
Total Length	The length of this message.
Number of Records	0
Transaction Number	The last Transaction Number add 1. AER would maintain a global Transaction Number for packets sent out to ACS where I Type is AD_PREFIX_INFO and AER would keep it increasing monotonic.
Acknowledgement Number	0
Data	None

Table 10

When receiving a PFX\_INFO-RequestAll message, if it holds that some fields are wrong, ACS could send a PFX\_INFO-NAK. Otherwise, ACS would act as follows. The specific construction methods of PFX\_INFO-ACK and PFX\_INFO-NAK are described in Section 5.2.3.

1. ACS SHOULD compare the Transaction Number in this packet with Transaction Number whose I Type is PFX\_INFO received from the same AER before. If bigger, ACS WOULD process them as step 2. Otherwise, ACS would discard this packet and send a PFX\_INFO-NAK message.

2. ACS processes every ADID\_Rec. If AD exists in the maintained trust alliance list, ACS would mark this record as "Reply". Otherwise, ACS would mark this record as "Negative Reply". Particularly, all records are marked with "Reply" when S Type is REQUEST\_ALL.
3. If any case in step 2 is marked with "Negative Reply", ACS would construct a PFX\_INFO-NAK message to reply to the AER. Otherwise, a PFX\_INFO-ACK message is constructed to reply the AD prefix information of all members marked with "Reply" to the AER.

#### 5.2.3. Response of AD Prefix Information

AD prefix information response includes two types. That is PFX\_INFO-ACK and PFX\_INFO-NAK. According to the request sent by AER, if some fields are wrong, ACS will reply NAK, in which the error code is "parameter error". If a non-existent member AD is queried, the error code is "the requested member AD does not exist", which defined as before will not be repeated. The following mainly introduces PFX\_INFO-ACK response. ACS fills in the following values for each field:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	AD_PREFIX_INFO
S Type	ACK: representing affirmative acknowledgement. NAK: representing negative acknowledgement.
Operation	RENEW: replying the latest AD prefix information to AER.
Total Length	The length of this message.
Number of Records	S Type = ACK: the number of API_Rec in Data field. S Type = NAK: 0
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent out to AER where I Type is AD_PREFIX_INFO and ACS would keep it increasing monotonic.
Acknowledgement Number	The Transaction Number of the response corresponding request.
Data	S Type = ACK: One or more latest requested API_Rec. S Type = NAK: a 32-bit error code defined in Section 3. There is no boundary identification between these API_Recs, which requires that the implementation of the protocol can process each record sequentially until the end of this message.

Table 11

When receiving a non-RENEW PFX\_INFO-ACK message which is the positive reply to the request of AD prefix sent from ACS to AER, if it holds that some fields are wrong, AER could send a request message to acquire the latest AD prefix information. Otherwise, AER would act as follows.



1. AER SHOULD compare the Transaction Number in this packet with Transaction Number whose I Type is PFX\_INFO received from the same ACS before. If bigger, AER would process them as follows. Otherwise, AER would discard this packet and send REG\_INFO-RequestAll and PFX\_INFO-RequestAll messages to acquire the latest information.
2. AER processes every API\_Rec: - If Action is ADD and the record does not exist in its maintained prefix list, AER would add this record to its prefix list. - If Action is ADD and the record exists in its maintained prefix list, AER would do nothing. - If Action is DEL and the record exists in its maintained prefix list, AER would remove this record from its prefix list after Effecting Time.
3. If a change is made in step 2, the update should take effect after the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than the current time or is all 0, it will take effect immediately.

AER acts as following when receiving a RENEW PFX\_INFO-ACK message. When ACS initiates RENEW process, it would send a RENEW message with which the first bit of Operation field is 1. The second bit of Operation field identifies the begin of a procedure of RENEW and the third bit of Operation field identifies the end of a procedure of RENEW. ACS MUST NOT send a RENEW packet with which the first bit of Operation field is 0 in RENEW process. AER SHOULD uniformly process all packets in this RENEW process after receiving all RENEW packets.

1. AER SHOULD compare the Transaction Number in this packet with Transaction Number whose I Type is PFX\_INFO received from the same ACS before. If bigger, AER WOULD process as step 2. Otherwise, AER would discard this message and send REG\_INFO-RequestAll and PFX\_INFO-RequestAll messages to acquire the latest information.
2. AER processes every API\_Rec. All Action in API\_Recs is ADD during RENEW process. - If the record does not exist in its maintained prefix list, AER would add this record to its trust alliance list. - If the record exists in its maintained prefix list, AER would do nothing. - If there are some records in the original prefix list that do not appear in the Data field during this RENEW process, these records will be deleted immediately.
3. If a change is made in step 2, the update message should take effect after the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than current time or is all 0, it will take effect immediately.

When AER receives an PFX\_INFO-NAK message, it could send REG\_INFO-RequestAll and PFX\_INFO-RequestAll messages to ACS to acquire the latest AD registration information and AD prefix information.

### 5.3. Deployment, Request and Response of State Machine Information

#### 5.3.1. Deployment of State Machine Information

State machine information deployment (SM\_INFO-Deploy) is sent from ACS to AER. ACS fills in the following values for each field:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	SM_INFO
S Type	DEPLOYMENT
Operation	NULL: to publish the partial update of state machine maintained by the pair of this AD and another AD and Operation is RENEW: to publish wholesome update of state machine maintained by the pair of this AD and another AD.
Total Length	The length of this message.
Number of Records	The number of SMI_Recs in Data field
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent out to AER where I Type is SM_INFO and ACS would keep it increasing monotonic.
Acknowledgement Number	0
Data	One or more SMI_Recs. There is no boundary identification between these ARI_Recs, which requires that the implementation of the protocol can process each record sequentially until the end of this message.

Table 12

It should be noted that state machine is responding to a ordered AD pair. The state machine information mastered by ACS includes the state machine information from this AD to another member AD, and the state machine information from another member AD to this AD. When ACS deployment is partially updated, only some changed or newly added state machines are deployed. When ACS deploys the update of RENEW message, it is necessary to deploy all existing and updated information. For the same ordered AD pair, there cannot be two or

more SMI\_Recs using the same SM\_ID in Data field. In addition, there are two actions for SMI\_Rec: one is to add a SM whose SM\_ID is bigger than current using state machine. The second is to modify an existing state machine whose SM\_ID equals to current using state machine. Both of them are using Action ADD. Here we requires only Transition Interval and Expiring Time can be updated.

When receiving a non-RENEW SM\_INFO-Deploy message sent from ACS to AER, if it holds that some fields are wrong, for example, Action is DEL or SM\_ID is smaller than current state machine in using, AER could send a request message to acquire the latest information. Otherwise, AER would act as follows.

1. AER SHOULD compare the Transaction Number in this packet with Transaction Number whose I Type is SM\_INFO received from the same ACS before. If bigger, AER WOULD process them as step 2. Otherwise, AER would discard this packet and send REG\_INFO-RequestAll and request messages to acquire the lastest information.
2. AER processes every SMI\_Rec: - If SM\_ID equals to the current using state machine, AER should update the state machine in use. - If SM\_ID bigger than the current using state machine, AER should add this state machine to its list.
3. If a change is made in step 2, the update message should take effect after the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than the current time or is all 0, it will take effect immediately.

AER acts as following when receiving a RENEW SM\_INFO-Deploy message. When ACS initiates RENEW process, it would send a RENEW meesge with which the first bit of Operation field is 1. The second bit of Operation field identifies the begin of a procedure of RENEW and the third bit of Operation field identifies the end of a procedure of RENEW. ACS MUST NOT send a RENEW packet with which the first bit of Operation field is 0 in RENEW process. AER SHOULD uniformly process all packets in this RENEW process after receiving all RENEW packets.

1. AER SHOULD compare the Transaction Number in this packet with Transaction Number whose I Type is SM\_INFO received from the same ACS before. If bigger, AER WOULD process as step 2. Otherwise, AER would discard this message and send a request messages to acquire the lastest information.
2. AER processes every SMI\_Rec. - If SM\_ID equals to the current using state machine, AER should update the state machine in use. - If SM\_ID bigger than the current using state machine, AER

should add this state machine to its list. - If there are some records of state machines in use that do not appear in the Data field during this RENEW process, these state machines will be deleted immediately.

3. If a change is made in step 2, the update message should take effect after the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than current time or is all 0, it will take effect immediately.

#### 5.3.2. Request of State Machine Information

State machine information request (SM\_INFO-Request) is sent from AER to ACS. AER fills in the following values for each field:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	SM_INFO
S Type	REQUEST: querying the state machines maintained by the pair of this AD to another member AD and vice versa. These member ADs are specified by ADID_Rec defined in Data field. REQUEST_ALL: querying all state machines maintained by this AD with other member ADs.
Operation	NULL
Total Length	The length of this message.
Number of Records	S Type = REQUEST: the number of ADID_Rec in Data field. S Type = REQUEST_ALL: 0.
Transaction Number	The last Transaction Number add 1. AER would maintain a global Transaction Number for packets sent out to ACS where I Type is SM_INFO and AER would keep it increasing monotonic.
Acknowledgement Number	0
Data	S Type = REQUEST: One or more ADID_Recs. S Type = REQUEST_ALL: none. There is no boundary identification between these ADID_Recs, which requires that the implementation of the protocol can process each record sequentially until the end of this message.

Table 13

For example, let this AD is AD1. When any ADID\_Rec including in Data field, defined as AD2, it means that AER will request the SM(AD1, AD2) and SM(AD2, AD1). When ACS replies, it will reply these two state machines both.

When receiving a SM\_INFO-Request(All) message, if it holds that some fields are wrong, ACS could send a PFX\_INFO-NAK. Otherwise, ACS would act as follows. The specific construction methods of SM\_INFO-ACK and SM\_INFO-NAK are described in section 3.2.3.3.

1. ACS SHOULD compare the Transaction Number in this packet with Transaction Number whose I Type is SM\_INFO received from the same AER before. If bigger, ACS WOULD process them as step 2. Otherwise, ACS would discard this packet and send a SM\_INFO-NAK message.
2. ACS processes every ADID\_Rec. If AD exists in the maintained trust alliance list, ACS would mark this record as "Reply". Otherwise, ACS would mark this record as "Negative Reply". Particularly, all records are marked with "Reply" when S Type is REQUEST\_ALL.
3. If any case in step 2 is marked with "Negative Reply", ACS would construct a SM\_INFO-NAK message to reply to the AER. Otherwise, a SM\_INFO-ACK message is constructed to reply the state machine information of all members marked with "Reply" to the AER.

#### 5.3.3. Response of State Machine Information

State machine information response includes two types. That is SM\_INFO-ACK and SM\_INFO-NAK. Both of them are sent from ACS to AER. ACS fills in the following values for each field:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	SM_INFO
S Type	ACK: representing affirmative acknowledgement. NAK: representing negative acknowledgement.
Operation	RENEW: replying the latest state machine information to AER.
Total Length	The length of this message.
Number of Records	S Type = ACK: the number of SMI_Recs in Data field. S Type = NAK: 0.
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent to AER where I Type is SM_INFO and would keep it increasing monotonic.
Acknowledgement Number	The Transaction Number of the response corresponding request.
Data	S Type = ACK: one or more latest requested SMI_Rec. S Type = NAK: a 32-bit error code defined in Section 3. There is no boundary identification between these ADID_Recs, which requires that the implementation of the protocol can process each record sequentially until the end of this message.

Table 14

When receiving a non-RENEW SM\_INFO-ACK message which is the positive reply to the request of AD prefix sent from ACS to AER, if it holds that some fields are wrong, AER could send a request message to acquire the latest state machine information. Otherwise, AER would act as follows. 1. AER SHOULD compare the Transaction Number in this packet with Transaction Number whose I Type is PFX\_INFO received from the same ACS before. If bigger, AER WOULD process them as step 2.



Otherwise, AER would discard this packet and send a SM\_INFO-RequestAll message to acquire the latest information. 2. AER processes every SMI\_Rec: - If SM\_ID equals to the current using state machine, AER should update the state machine in use. - If SM\_ID bigger than the current using state machine, AER should add this state machine to its list. 3. If a change is made in step 2, the update should take effect after the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than the current time or is all 0, it will take effect immediately.

AER acts as following when receiving a RENEW SM\_INFO-ACK message. When ACS initiates RENEW process, it would send a RENEW message with which the first bit of Operation field is 1. The second bit of Operation field identifies the begin of a procedure of RENEW and the third bit of Operation field identifies the end of a procedure of RENEW. ACS MUST NOT send a RENEW packet with which the first bit of Operation field is 0 in RENEW process. AER SHOULD uniformly process all packets in this RENEW process after receiving all RENEW packets.

1. AER SHOULD compare the Transaction Number in this packet with Transaction Number whose I Type is SM\_INFO received from the same ACS before. If bigger, AER WOULD process as step 2. Otherwise, AER would discard this message and send a SM\_INFO-RequestAll message to acquire the latest information.
2. AER processes every API\_Rec. All Action in API\_Recs is ADD during RENEW process. - If SM\_ID equals to the current using state machine, AER should update the state machine in use. - If SM\_ID bigger than the current using state machine, AER should add this state machine to its list. - If there are some records of state machines in use that do not appear in the Data field during this RENEW process, these state machines will be deleted immediately.
3. If a change is made in step 2, the update message should take effect after the Effecting Time, which acts on the data plane. If the Effecting Time is earlier than current time or is all 0, it will take effect immediately.

When AER receives an SM\_INFO-NAK message, it could send a SM\_INFO-RequestAll message to ACS to acquire the latest state machine information.

#### 5.4. Request and Response of Keep-alive Information

In SAVA-X, ACS will periodically send Keep-alive request to query the availability of AER in SAVA-X mechanism. ### Request of Keep-alive Information Keep-alive information request (ALIVE\_INFO-Request) is sent by ACS to test the viability of AER. AER would reply to ACS when receiving a ALIVE\_INFO-Request message. ACS considers that AER has gone wrong if it does not receive a response from AER within 60 seconds and ACS notifies the AD administrator of the failure information by email. ACS would keep sending ALIVE\_INFO-Request to the fault AER at the same time. The filling values of each field in ACS request are as follows:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	ALIVE_INFO
S Type	REQUEST
Operation	NULL
Total Length	The length of this message.
Number of Records	0
Transaction Number	The last Transaction Number add 1. ACS would maintain a global Transaction Number for packets sent to AER where I Type is ALIVE_INFO and would keep it increasing monotonic.
Acknowledgement Number	0
Data	None

Table 15

ACS considers that AER has gone wrong if it does not receive a response from AER within 60 seconds and ACS notifies the AD administrator of the failure information by email. ACS would consider that AER has recovered from failure when AER reply to the request correctly. ACS performs the following steps to update AER:

1. Keep time synchronization between AER and ACS.
2. Deploy AD registration information, AD prefix information and state machine information to AER by the way of RENEW message.

#### 5.4.1. Response of Keep-alive Information

Keep-alive information response (ALIVE\_INFO-Response) is sent by AER to reply the ALIVE\_INFO-Request message.

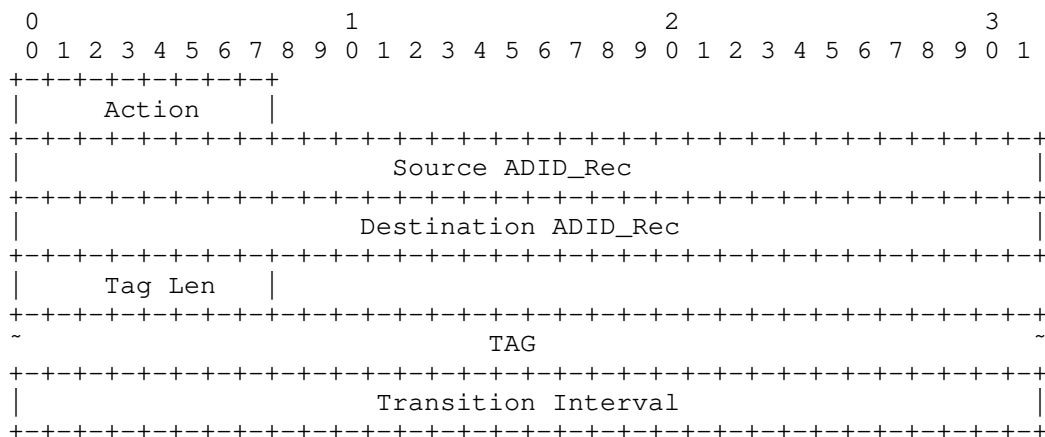
In response to ALIVE\_INFO-Request, AER fills in the following values for each field in the response:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	ALIVE_INFO
S Type	ACK
Operation	NULL
Total Length	The length of this message.
Number of Records	0
Transaction Number	The last Transaction Number add 1. AER would maintain a global Transaction Number for packets sent to ACS where I Type is ALIVE_INFO and would keep it increasing monotonic.
Acknowledgement Number	0
Data	None

Table 16

## 6. Deployment of Tag Information

Tag information deployment (TAG\_INFO-Deploy) is sent from ACS to AER and AER would add, verify and remove the tag to packet. When using sub trust alliance level tags and AD\_V tags, the primary address domain ACS needs to distribute these two tags to the ACS of the boundary address domain first, and then the boundary address domain ACS will distribute these tags to their respective address domains' AERs. The sub trust alliance tag is used in the data plane to cross different address domain levels. The AD\_V tag is used in the data plane when it is sent from the current address domain to the boundary address domain. Standard TAG\_INFO is used in the data plane at the same level and under the same direct parent address field. The three types of tags use the same message format as follows.



Action: 8-bit field. 1 for add (ADD=1) and 2 for delete (DEL=2).

Source ADID\_Rec: Variable-length field. Refer to ADID\_Rec in [SAVA-X-Control].

Destination ADID\_Rec: Variable-length field. Refer to ADID\_Rec.

Tag Len: The length of TAG. The equation for calculation is (Tag Len + 1) \* 8 bits. The length of TAG MUST be multiple times of 8 bits. The maximum length is 128 bits and the minimum length is 32 bits. So the minimum of Tag Len is 0011.

TAG: Variable-length field. The actual Tag or packet signature.

Transition Interval: 32-bit, the milliseconds of interval of state transition.

When ACS announce tag to ACS or AER, it fills in the following values for each field:

Field	Value
Version	1
Alliance	The sub-trust alliance number.
I Type	TAG_INFO, ALLI_TAG_INFO or AD_V_TAG_INFO
S Type	ANNOUNCEMENT
Operation	NULL
Total Length	The length of this message.
Number of Records	The number of TAG_Rec in Data field.
Transaction Number	ACS would maintain a global Transaction Number for packets sent to ACS or AER where I Type is TAG_INFO and would keep it increasing monotonic. Acknowledgement Number is 0.
Acknowledgement Number	0
Data	One or more TAG_Recs. There is no boundary identification between these records, which requires that the implementation of the protocol can process each record sequentially until the end of this message.

Table 17

## 7. Security Consideration

This present memo doesnot find any security problem.

## 8. IANA Consideration

There are two tcp ports, 23160 and 23161, are used in implementing SAVA-X mechanism. Port 23160 is used for the communication between ACS and ACS. Port 23161 is used for the communication between ACS and AER.

## 9. Acknowledgements

Much of the content of this document is the expansion of the IETF [RFC5210] in inter-domain level. Thanks to the effort of pioneers.

## 10. References

### 10.1. 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>>.
- [RFC5210] Wu, J., Bi, J., Li, X., Ren, G., Xu, K., and M. Williams, "A Source Address Validation Architecture (SAVA) Testbed and Deployment Experience", RFC 5210, DOI 10.17487/RFC5210, June 2008, <<https://www.rfc-editor.org/info/rfc5210>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.

### 10.2. Informative References

- [SAVA-X-Control] Computer Science, ., Computer Science, ., and . Institute for Network Sciences and Cyberspace, "Control Plane of Inter-Domain Source Address Validation Architecture", 2021.

## Authors' Addresses

Ke Xu  
Computer Science, Tsinghua University  
Qinghuayuan street, Haidian District  
Beijing  
100084  
China

Email: [xuke@tsinghua.edu.cn](mailto:xuke@tsinghua.edu.cn)

Xiaoliang Wang  
Computer Science, Tsinghua University  
Qinghuayuan street, Haidian District

Beijing  
100084  
China

Email: wangxiaoliang0623@foxmail.com

Yangfei Guo  
Institute for Network Sciences and Cyberspace, Tsinghua University  
Qinghuayuan street, Haidian District  
Beijing  
100084  
China

Email: guoyangf19@mails.tsinghua.edu.cn



Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 21 April 2022

D. von Hugo  
Deutsche Telekom  
B. Sarikaya  
18 October 2021

Problem Statement for Internet of Things Sensing  
draft-hsothers-iotsens-ps-00

Abstract

The document attempts to establish hardware based Internet of Things authentication as a future networking area beyond 5G going into 6G for standardization. The problem of hardware authentication is discussed and its relationship with Wireless Local Area network collaborative and/or multi-band sensing is established and then recent research efforts in the area are indicated.

Status of This Memo

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

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

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

This Internet-Draft will expire on 21 April 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
1.1. Conventions and Terminology . . . . .	3
2. Hardware Based Authentication . . . . .	4
3. State of the Academic Approaches to IoT Authentication . . . . .	5
4. IoT Authentication Protocols . . . . .	6
5. Hardware IoT Authentication Problem . . . . .	6
5.1. Architectural and Procedural Issues for Future IP-based IoT-Authentication . . . . .	7
6. IANA Considerations . . . . .	8
7. Security Considerations . . . . .	8
8. Acknowledgements . . . . .	8
9. References . . . . .	8
9.1. Normative References . . . . .	8
9.2. Informative References . . . . .	8
Authors' Addresses . . . . .	11

## 1. Introduction

Future networking to make full use of 5G capabilities or even resembling an evolution to beyond 5G will have to exploit a much more heterogeneous environment in terms of network and device connectivity technologies and applications. In addition ease of use for customers and human-independent operation of a multitude of devices and machines (things) has to be provided.

Therefore current authentication models like 802.1X [IEEE802.1X] which are based on human intervention do not fit well. Also this model does not scale well for the Internet of Things (IoT). What we need is hardware based admission model. Such a model will enable many new applications as we explain more in this document.

IEEE 802.11 [IEEE802.11] has a project on Wireless LAN (WLAN sensing) and 802.11bf task group (TG) in charge of this project [BFSFD]. Use cases for 802.11bf TG includes room sensing, i.e., presence detection, counting the number of people in the room, localization of active people, audio with user detection, gesture recognition at different ranges, device proximity detection, home appliance control. There are also health care related use cases like breathing/heart rate detection, surveillance of persons of interest, building a 3D picture of an environment, as, e.g., in-car sensing for driver sleepiness detection [BFUseCases].

Hardware based authentication that we address in this document builds on similar use cases. We can summarize the use cases we are currently considering here: Authenticating the device that is playing a melody, or a person has just touched; authenticating devices, i.e.

smart teapot with certain manifests, like blinking red and blue; authenticate the device when a camera is pointed at it; and the like [Henning]. 802.11bf sensing project provides proper framework for hardware based authentication because 802.11 or Wi-Fi devices are more and more diverse spanning from personal computers, smartphones, televisions, tablets, and all sorts of IoT devices or sensors.

TGbf is also working on Specification Framework Document with an outline of each of the functional blocks that will be a part of the final amendment like wireless LAN sensing procedure [BFSFD]. TGbf sensing is based on obtaining physical Channel State Information (CSI) measurements between a transmitter and receiver WLAN nodes, called stations (STA). Using these measurements, presence of obstacles between a transmitter and receiver can be detected and tracked. This way, using feature extraction and classification provided by means of artificial intelligence (AI), more higher level tasks like human activity recognition and object detection are available for authentication purposes, while hardware based authentication use cases can be achieved through computation of phase differences, etc.

TGbf Wi-Fi Sensing (SENS) is achieved by signaling between just an initiator and a responder. TGbf may also define more effective collaborative SENS (in short, CSENS) where multiple SENS-enabled devices can collaborate as a group in an orderly fashion to capture additional information about the surrounding environment [Rest21].

### 1.1. Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Sensing (SENS) is defined as the usage of received Wi-Fi signals from a Station (STA) to detect features (i.e., range, velocity, angular, motion, presence or proximity, gesture, etc.) of intended targets (i.e., object, human, animal, etc.) in a given environment (i.e., house, office, room, vehicle, enterprise, etc.).

Collaborative sensing (CSENS) defines the operation in which multiple SENS enabled devices can collaborate as a group in an orderly fashion to capture additional information about the surrounding environment and allow for more precise detection, thus enabling a more reliable authentication.

Multi-band sensing is defined as sensing using both sub-7-GHz Channel State Information (CSI) measurements that provide indication of relatively large motions and that can propagate through obstacles (e.g., walls) and 60-GHz Received Signal Strength Indicator (RSSI) measurements at mmWave that provide highly-directional information through the usage of beamforming toward a given receiver, but have small range due to the presence of blockers (e.g., walls).

## 2. Hardware Based Authentication

Aim of this document is to lay ground for the need for new authentication models in the framework of devices (e.g., machines in IoT communication) within a (wireless or wireline-based) network. Currently employed authentication models (such as e.g., 802.1X certificate model) is based on a human being using the machine and providing credentials (e.g., user name/password or a permitted digital certificate) to the authenticator. Similarly, for user equipment (UE) to access a cellular network the device has to be equipped with a USIM and the user has to provide a secret key, i.e., PIN (Personal Identification Number). With the use case of massive IoT (mIoT) as foreseen, e.g., in 5G and with an increasing amount of devices within a household (smart home) and/or in the ownership of a customer (smart watch etc.) the need for an ease-of-use hardware-based admission model arises.

Focusing on corresponding procedures starting with detection (sensing) of a new device and subsequent mutual authenticating of the device by and to the network a set of potential technologies are identified and described to allow for analysis in terms of criteria as reliable operation (working), scalability, ease of use and convenience, security, and many more. Sensing is critical to Hardware Based Authentication because sensing (together with intelligent interpretation using possibly neural network models) will allow the detection of the device playing a melody, blinking red and blue, being pointed at, or somebody just touched and the like. Furthermore, the method should be applicable to future generations of network and of users, upcoming new applications and devices, assuming that today's established standard procedures do not fulfill the requirements sufficiently.

Hardware based authentication should leverage collaborative and multi-band sensing technologies to enable sensing with much higher precision and capacity using the state-of-art equipment. Also equally important is the use of all artificial intelligence and neural networks research results developed by the academia.

### 3. State of the Academic Approaches to IoT Authentication

A detailed review on current topics in IoT Security, Device Authentication and Access Control was provided in [Inayat]. The following list of literature on sensor data and WiFi sensing for securing and authenticating a user and a device shows the wide range of approaches and interest in this topic [Rest21].

[Ma], [Wang], [Zhu], [Wang2], and [Qian] provide a holistic overview on the evolution of Wi-Fi technology and on investigations in opportunistic applications of Wi-Fi signals for gesture and motion detection.

[Henning2] is investigating geospatial access control for IoT. There are attribute, role and identity based, time based and geospatial access control techniques. Real-world IoT access control policies will be a combination of all three, leading to powerful access control techniques to use in practice such as in university campus. Such access control or authorization techniques will likely be used in conjunction with Hardware Based Authentication.

Other notable literature includes [Al-Qaness] on the so-called device-free CSI-based Wi-Fi sensing mechanism, [Pahlavan] using Wi-Fi signals for gesture and motion detection as well as for authentication and security, [Lui] distinguishing between Line-of-Sight (LOS) and Non-Line-of-Sight (NLOS) conditions in case of obstacles appearing between the transmitter and the receiver [Guo] studying HuAc (Human Activity Recognition) as a combination of WiFi-based and Kinect-based activity recognition system, [FURQAN] analyzing the wireless sensing and radio environment awareness mechanisms, highlighting their vulnerabilities such as dependency of sensing modes on external signals, and provides solutions for mitigating them, e.g., the different threats to REM (radio environment mapping) and its consequences in a vehicular communication scenario.

[Ma2] has studied reliable SENS algorithm for human and animal identification. The aim is to make it resilient to spoofing and adverse channel conditions, i.e., presence of noise and interference from other technologies.

[Restuccia] investigates data driven algorithms, neural networks, especially convolutional neural network (CNN) or digital signal processing (DSP) block to classify complex sensing phenomena. Also [Liao] and [Liao2] proposed to enhance security of industrial wireless sensor networks (IWSNs) by neural network based algorithms for sensor nodes' authentication and implementations in IWSNs have shown that an improved convolution preprocessing neural network

(CNN)-based algorithm requires few computing resources and has extremely low latency, thus enabling a lightweight multi-node PHY-layer authentication.

Further research on these and similar issues can be found in [Tian], [Bai], and [Axente].

#### 4. IoT Authentication Protocols

Since IoT applications cover a broad range of domains from smart cities, industry, and homes to personal (e.g., wearable) devices, including security and privacy sensitive areas as e-health, and can reach a huge number of entities the security requirements in terms of preventing unauthorized access to data are very high. Therefore very robust authentication mechanisms have to be applied. At the same time depending on the specific scenario a trade-off between resources as processing power and memory and security protocol complexity has to be considered. Also a plethora of attack scenarios has to be in focus as well as scalability of the considered implicit and explicit hardware- and software-based authentication procedures. [RFC8576] serves as a reference for details about IoT specific security considerations including the area of authentication and documents their specific security challenges, threat models, and possible mitigations.

A more recent work surveys secure bootstrapping and onboarding protocols [I-D.irtf-t2trg-secure-bootstrapping-00] developed by IETF as well as other standards developing organizations such as IEEE, FIDO alliance, Open Connectivity Foundation (OCF), Open Mobile Alliance (OMA).

Lastly, the Open Authorization (OAuth) [RFC6749] protocol in the area of authorization is a standard for access delegation. It extends traditional client-server authentication by providing a third party client with a token instead of allowing it to use the resource owner's credentials to access protected resources while such token resembles a different set of credentials than those of the resource owner.

#### 5. Hardware IoT Authentication Problem

Most of the state-of-art hardware identification techniques to authenticate the user use finger prints a.k.a. touch id and facial identification and they use detection by hardware i.e. touch, accelerometer, and gyro sensors or cameras. They are based on creating a signature, or the user's already stored password [Wang3].

On the other hand to authenticate a device based on a set of characteristic parameters which should be flexibly chosen by the owner and subsequently made known to the authentication system will require a certain level of processing and storage capacity either within the local system components (e.g., the device itself and the wireless point of attachment or access point) and/or within the network (e.g., an edge cloud instance or a central data base). The result of the detection process (e.g., radio wave analysis outcome in terms of parameters as modulation scheme, number of carriers, and fingerprinting) has to be compared with the required (correct) parameter values which are safely stored within the network components. On all levels of handling these data, i.e., storage, processing, and transport via a communication network, the integrity of the content has to be preserved. One should keep in mind, that any unintended authentication request should be prevented to minimize the risk of occasional attachment to networks and subsequent exposure to attack to sensitive user data.

#### 5.1. Architectural and Procedural Issues for Future IP-based IoT-Authentication

Here we will discuss possible solutions on IP level and identify benefits and potential gaps towards the requirements of next generation IoT systems. On IP or network layer for IPv6 IPsec protocol suite is mandatory and provides end-to-end security for authentication procedures, ensuring confidentiality and integrity of the transmitted data. Authentication for IoT may rely on a protocol as 6LoWPAN (Low-power Wireless Personal Area Network) which is defined for optimizing the efficient routing of IPv6 packets for resource constrained machine- type communication applications.

When compared to a fully certificate-based authentication, however, a hardware-based AAA mechanism relying e.g., on WiFi sensing gesture detection does not require the user to know any key, identifier, or password for the device to be authenticated. A pre-defined type of access to the device (e.g., physical, photographic or video representation, unique description in terms of parameters, etc.) shall be sufficient for authentication.

[RFC8995] on 'Bootstrapping Remote Secure Key Infrastructure' (BRSKI) deals with authentication of devices, including sending authorizations to the device as to what network they should join, and how to authenticate that network by specifying automated bootstrapping of an Autonomic Control Plane (ACP). Secure Key Infrastructure (SKI) bootstrapping using manufacturer-installed X.509 certificates combined with a manufacturer's authorizing service, both online and offline, is called the Bootstrapping Remote Secure Key Infrastructure (BRSKI) protocol. Bootstrapping a new device can

occur when using a routable address and a cloud service, only link-local connectivity, or limited/disconnected networks and includes support for deployment models with less stringent security requirements. When the cryptographic identity of the new SKI is successfully deployed to the device, completion of bootstrapping is achieved. A locally issued certificate can be deployed to the device via the established secure connection as well.

## 6. IANA Considerations

TBD.

## 7. Security Considerations

This document raises no new security concerns but tries to identify how to increase security in future IoT by discussing the issues of robust but easy to apply authentication mechanisms.

## 8. Acknowledgements

TBD.

## 9. References

### 9.1. 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>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

### 9.2. Informative References

- [Al-Qaness] Al-Qaness, M.A.A., Abd Elaziz, M., Kim, S., Ewees, A.A., Abbasi, A.A., Alhaj, Y.A., and A. Hawbani, "Channel State Information (CSI) from Pure Communication to Sense and Track Human Motion: A Survey", *Sensors* 2019, 19(15), 3329, July 2019.
- [Axente] Axente, M.-S., Dobre, C., Ciobanu, R.-I., and R. Purnichescu-Purtan, "Gait Recognition as an Authentication Method for Mobile Devices", *Sensors* 2020, 20, 4110, July 2020.



- [Bai] Bai, L., Zhu, L., Liu, J., Choi, J., and W. Zhang, "Physical Layer Authentication in Wireless Communication Networks: A Survey", Journal of Communications and Information Networks Vol.5, No.3, September 2020.
- [BFSFD] IEEE, "Institute of Electrical and Electronics Engineers, IEEE P802.11 - TASK GROUP BF (WLAN SENSING) 11-21/0504r2 "Specification Framework for TGBf"", July 2021.
- [BFUseCases] IEEE, "Institute of Electrical and Electronics Engineers, IEEE P802.11 - TASK GROUP BF (WLAN SENSING) 11-20/1712r2 "WiFi Sensing Use Cases"", January 2021.
- [FURQAN] Furqan, H.M., Solaija, M.S.J., Tuerkmen, H., and H. Arslan, "Wireless Communication, Sensing, and REM: A Security Perspective", IEEE Open Journal of the Communications Society Vol. 2 , January 2021.
- [Guo] Guo, L., Wang, L., Liu, J., Zhou, W., and B. Lu, "HuAc: Human Activity Recognition Using Crowdsourced WiFi Signals and Skeleton Data", Hindawi Wireless Communications and Mobile Computing, Volume 2018 , February 2021.
- [Henning] Schulzrinne, H., "Do We Still Need Wi-Fi in the Era of 5G (and 6G)?", February 2021.
- [Henning2] Jan Janak, Luoyao Hao and Henning Schulzrinne, ., "How do we program the Internet of Things at scale?", September 2021.
- [I-D.irtf-t2trg-secure-bootstrapping-00] Sethi, M., Sarikaya, B., and D. Garcia-Carrillo, "Secure IoT Bootstrapping: A Survey", Work in Progress, Internet-Draft, draft-irtf-t2trg-secure-bootstrapping-00, 7 April 2021, <<https://www.ietf.org/archive/id/draft-irtf-t2trg-secure-bootstrapping-00.txt>>.
- [IEEE802.11] IEEE, "IEEE Std. 802.11-2016", December 2016, <<https://standards.ieee.org/findstds/standard/802.11-2016.html>>.
- [IEEE802.1X] IEEE, "Institute of Electrical and Electronics Engineers, "802.1X - Port Based Network Access Control"", January 2020.

- [Inayat] Ali, I., Sabir, S., and Z. Ullah, "Internet of Things Security, Device Authentication and Access Control: A Review", International Journal of Computer Science and Information Security (IJCSIS), Vol. 14, No. 8 , August 2016.
- [Liao] Liao, R.-F., Wen, H., Wu, J., Pan, F., Xu, A., Jiang, Y., Xie, F., and M. Cao, "Deep-learning-based physical layer authentication for industrial wireless sensor networks", Sensors 2019, 19(11), 2440 , May 2019.
- [Liao2] Liao, R.-F., Wen, H., Wen, H., Xie, F., Pan, F., Pan, F., and F. Xie, "Multiuser Physical Layer Authentication in Internet of Things With Data Augmentation", IEEE Internet of Things Journal, vol. 7, no. 3, pp. 2077-2088 , March 2020.
- [Lui] Liu, J., Wang, L., Fang, J., Guo, L., Lu, B., and L. Shu, "Multi-Target Intense Human Motion Analysis and Detection Using Channel State Information", Sensors 2018, 18(10), 3379 , October 2018.
- [Ma] Ma, Y., Arshad, et al, S., and , "Location-and Person-Independent Activity Recognition with WiFi, Deep Neural Networks, and Reinforcement Learning," , 2021.
- [Ma2] Ma, Y. and G. Zhou, et al, "WiFi Sensing with Channel State Information: A Survey," , ACM Computing Surveys (CSUR), , vol. 52, no. 3, pp. 1-36, 2019.
- [Pahlavan] Pahlavan, K. and P. Krishnamurthy, "Evolution and Impact of Wi Fi Technology and Applications: A Historical Perspective", Springer Science+Business Media, LLC, part of Springer Nature 2020 , November 2020.
- [Qian] Xian, K. and C. Wu, et al, "Widar: Decimeter-level Passive Tracking via Velocity Monitoring with Commodity WiFi," , Proc. of ACM MobiCom, , 2017.
- [Rest21] Restuccia, F., "IEEE 802.11bf: Toward Ubiquitous Wi-Fi Sensing", arXiv preprint arXiv:2103.14918 7 pages, March 2021.
- [Restuccia] Restuccia, F. and T. Melodia, "Deep Learning at the Physical Layer: System Challenges and Applications to 5G and Beyond," , IEEE Communications Magazine, , vol. 58, no. 10, pp. 58-64, 2020.

- [RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <<https://www.rfc-editor.org/info/rfc6749>>.
- [RFC8576] Garcia-Morchon, O., Kumar, S., and M. Sethi, "Internet of Things (IoT) Security: State of the Art and Challenges", RFC 8576, DOI 10.17487/RFC8576, April 2019, <<https://www.rfc-editor.org/info/rfc8576>>.
- [RFC8995] Pritikin, M., Richardson, M., Eckert, T., Behringer, M., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructure (BRSKI)", RFC 8995, DOI 10.17487/RFC8995, May 2021, <<https://www.rfc-editor.org/info/rfc8995>>.
- [Tian] Tian, Q., Lin, Y., Guo, X., Wang, J., AlFarraj, O., and A. Tolba, "An Identity Authentication Method of a MIIoT Device Based on Radio Frequency (RF) Fingerprint Technology", Sensors 2020, 20(4), 1213 , February 2020.
- [Wang] Wang, X. and C. Yang, et al, "TensorBeat: Tensor Decomposition for Monitoring Multiperson Breathing Beats with Commodity WiFi", ACM Transactions on Intelligent Systems and Technology (TIST), , vol. 9, no. 1, pp. 1-27, 2017.
- [Wang2] Wang, X. and C. Yang, et al, "PhaseBeat: Exploiting CSI Phase Data for Vital Sign Monitoring with Commodity WiFi Devices", Proc. of IEEE ICDCS, , 2017.
- [Wang3] Wang, H., Lymberopoulos, D., and J. Liu, "Sensor-Based User Authentication", EWSN 2015, LNCS 8965, 168 , 2015.
- [Zhu] Zhu, H. and F. Xiao, et al, "R-TTWD: Robust device-free through-the-wall detection of moving human with WiFi", IEEE Journal on Selected Areas in Communications, , vol. 35, no. 5, pp. 1090-1103, 2017.

#### Authors' Addresses

Dirk von Hugo  
Deutsche Telekom  
Deutsche-Telekom-Allee 9  
64295 Darmstadt  
Germany

Email: [Dirk.von-Hugo@telekom.de](mailto:Dirk.von-Hugo@telekom.de)

Behcet Sarikaya

Email: sarikaya@ieee.org

Internet Area Working Group  
Internet-Draft  
Intended status: Informational  
Expires: 7 September 2022

Y. Jia  
D. Trossen  
L. Iannone  
Huawei  
P. Mendes  
Airbus  
N. Shenoy  
R.I.T.  
L. Toutain  
IMT-Atlantique  
A. Y. Chen  
Avinta  
D. Farinacci  
lispers.net  
6 March 2022

Gap Analysis in Internet Addressing  
draft-jia-intarea-internet-addressing-gap-analysis-02

Abstract

There exist many extensions to Internet addressing, as it is defined in [RFC0791] for IPv4 and [RFC8200] for IPv6, respectively. Those extensions have been developed to fill gaps in capabilities beyond the basic properties of Internet addressing. This document outlines those properties as a baseline against which the extensions are categorized in terms of methodology used to fill the gap together with examples of solutions doing so.

While introducing such extensions, we outline the issues we see with those extensions. This ultimately leads to consider whether or not a more consistent approach to tackling the identified gaps, beyond point-wise extensions as done so far, would be beneficial. The benefits are the ones detailed in the companion document [I-D.jia-intarea-scenarios-problems-addressing], where, leveraging on the gaps identified in this memo and scenarios provided in [I-D.jia-intarea-scenarios-problems-addressing], a clear problem statement is provided.

Status of This Memo

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

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

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

This Internet-Draft will expire on 7 September 2022.

#### Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

#### Table of Contents

1. Introduction . . . . .	3
2. Properties of Internet Addressing . . . . .	4
2.1. Property 1: Fixed Address Length . . . . .	4
2.2. Property 2: Ambiguous Address Semantic . . . . .	4
2.3. Property 3: Limited Address Semantic Support . . . . .	5
3. Filling Gaps through Extensions to Internet Addressing Properties . . . . .	5
3.1. Length Extensions . . . . .	5
3.1.1. Shorter Address Length . . . . .	6
3.1.2. Longer Address Length . . . . .	8
3.1.3. Summary . . . . .	10
3.2. Identity Extensions . . . . .	10
3.2.1. Anonymous Address Identity . . . . .	11
3.2.2. Authenticated Address Identity . . . . .	14
3.2.3. Summary . . . . .	15
3.3. Semantic Extensions . . . . .	16
3.3.1. Utilizing Extended Address Semantics . . . . .	17
3.3.2. Utilizing Existing or Extended Header Semantics . . . . .	20
3.3.3. Summary . . . . .	23
4. Overview of Approaches to Extend Internet Addressing . . . . .	24

5. A System View on Address . . . . .	26
6. Issues in Extensions to Internet Addressing . . . . .	27
6.1. Limiting Address Semantics . . . . .	27
6.2. Complexity and Efficiency . . . . .	27
6.2.1. Repetitive encapsulation . . . . .	28
6.2.2. Compounding issues with header compression . . . . .	29
6.2.3. Introducing Path Stretch . . . . .	29
6.2.4. Complicating Traffic Engineering . . . . .	29
6.3. Security . . . . .	30
6.4. Fragility . . . . .	30
7. Summary of issues . . . . .	31
8. Conclusions . . . . .	33
9. Security Considerations . . . . .	34
10. IANA Considerations . . . . .	34
11. Informative References . . . . .	34
Acknowledgments . . . . .	44
Authors' Addresses . . . . .	44

## 1. Introduction

[I-D.jia-intarea-scenarios-problems-addressing] outlines scenarios and problems in Internet addressing through presenting a number of cases of communication that have emerged over the many years of utilizing the Internet and for which various extensions to the network interface-centric addressing of IPv6 have been developed. In order to continue the discussion on the emerging needs for addressing, initiated with [I-D.jia-intarea-scenarios-problems-addressing], this memo aims at identifying gaps between the Internet addressing model and desirable features that have been added by various extensions, in various contexts.

The approach to identifying the gaps is guided by key properties of Internet addressing, outlined in Section 2, namely (i) the fixed length of the IP addresses, (ii) the ambiguity of IP addresses semantic, while still (iii) providing limited IP address semantic support. Those properties are derived directly as a consequence of the respective standards that provide the basis for Internet addressing, most notably [RFC0791] for IPv4 and [RFC8200] for IPv6, respectively.

Those basic properties, and the potential issues that arise from those properties, give way to extensions that have been proposed over the course of deploying new Internet technologies. Section 3 discusses those extensions, summarized as gaps against the basic properties in Section 4.

Finally, this memo outlines issues that arise with the extension-driven approach to the basic Internet addressing, discussed in Section 6, arguing that any requirements for solutions that would revise the basic Internet addressing would require to address those issues.

## 2. Properties of Internet Addressing

As the Internet Protocol adoption has grown towards the global communication system we know today, its characteristics have evolved subtly, with [RFC6250] documenting various aspects of the IP service model and its frequent misconceptions, including Internet addressing. In this section, the three most acknowledged properties related to `_Internet addressing_` are detailed. Those are (i) fixed IP address length, (ii) ambiguous IP address semantic, and (iii) limited IP address semantic support.

Section 3 elaborates on various extensions that aim to expand Internet addressing beyond those properties; those extensions are positioned as intentions to close perceived gaps against those key properties.

### 2.1. Property 1: Fixed Address Length

The fixed IP address length is specified as a key property of the design of Internet addressing, with 32 bits for IPv4 ([RFC0791]), and 128 bits for IPv6 ([RFC8200]), respectively. Given the capability of the hardware at the time of IPv4 design, a fixed length address was considered as a more appropriate choice for efficient packet forwarding. Although the address length was once considered to be variable during the design of Internet Protocol Next Generation ("IPng", cf., [RFC1752]) in the 1990s, it finally inherited the design of IPv4 and adopted a fixed length address towards the current IPv6. As a consequence, the 128-bit fixed address length of IPv6 is regarded as a balance between fast forwarding (i.e., fixed length) and practically boundless cyberspace (i.e., enabled by using 128-bit addresses).

### 2.2. Property 2: Ambiguous Address Semantic

Initially, the meaning of an IP address has been to identify an interface on a network device, although, when [RFC0791] was written, there were no explicit definitions of the IP address semantic.

With the global expansion of the Internet protocol, the semantic of the IP address is commonly believed to contain at least two notions, i.e., the explicit 'locator', and the implicit 'identifier'. Because of the increasing use of IP addresses to both identify a node and to



indicate the physical or virtual location of the node, the intertwined address semantics of identifier and locator was then gradually observed and first documented in [RFC2101] as 'locator/identifier overload' property. With this, the IP address is used as an identification for host and server, very often directly used, e.g., for remote access or maintenance.

### 2.3. Property 3: Limited Address Semantic Support

Although IPv4 [RFC0791] did not add any semantic to IP addresses beyond interface identification (and location), time has proven that additional semantics are desirable (c.f., the history of 127/8 [HISTORY127] or the introduction of private addresses [RFC1918]), hence, IPv6 [RFC4291] introduced some form of additional semantics based on specific prefix values, for instance link-local addresses or a more structured multicast addressing. Nevertheless, systematic support for rich address semantics remains limited and basically prefix-based.

## 3. Filling Gaps through Extensions to Internet Addressing Properties

Over the years, a plethora of extensions has been proposed in order to move beyond the native properties of IP addresses, outlined in the previous section. The development of those extensions can be interpreted as filling gaps between the original properties of Internet addressing and desired new capabilities that those developing the extensions identified as being missing and yet needed and desirable.

### 3.1. Length Extensions

Extensions in this subsection aim at extending the property described in Section 2.1, i.e., the fixed IP address length.

When IPv6 was designed, the main objective was to create an address space that would not lead to the same situation as IPv4, namely to address exhaustion. To this end, while keeping the same addressing model like IPv4, IPv6 adopted a 128-bit address length with the aim of providing a sufficient and future-proof address space. The choice was also founded on the assumption that advances in hardware and Moore's law would still allow to make routing and forwarding faster, and the IPv6 routing table manageable.

We observe, however, that the rise of new use cases but also the number of new, e.g., industrial/home or small footprint devices, was possibly unforeseen. Sensor networks and more generally the Internet of Things (IoT) emerged after the core body of work on IPv6, thus different from IPv6 assumptions, 128-bit addresses were costly in

certain scenarios. On the other hand, given the huge investments that IPv6 deployment involved, certain solutions are expected to increase the addressing space of IPv4 in a compatible way, and thus extend the lifespan of the sunk investment on IPv4.

At the same time, it may also be possible to use variable and longer address lengths to address current networking demands. For example in content delivery networks, longer addresses such as URLs are required to fetch content, an approach that Information-Centric Networking (ICN) applied for any data packet sent in the network, using information-based addressing at the network layer. Furthermore, as an approach to address the routing challenges faced in the Internet, structured addresses may be used in order to avoid the need for routing protocols. Using variable length addresses allow as well to have shorter addresses. So for requirements for smaller network layer headers, shorter addresses could be used, maybe alleviating the need to compress other fields of the header. Furthermore, transport layer port numbers can be considered short addresses, where the high order bits of the extended address is the public IP of a NAT. Hence, in IoT deployments, the addresses of the devices can be really small and based on the port number, but they all share the global address of the gateway to make each one have a globally unique address.

### 3.1.1. Shorter Address Length

#### 3.1.1.1. Description:

In the context of IoT [RFC7228], where bandwidth and energy are very scarce resources, the static length of 128-bit for an IP address is more a hindrance than a benefit since 128-bit for an IP address may occupy a lot of space, even to the point of being the dominant part of a packet. In order to use bandwidth more efficiently and use less energy in end-to-end communication, solutions have been proposed that allow for very small network layer headers instead.

#### 3.1.1.2. Methodology:

- \* Header Compression/Translation: One of the main approaches to reduce header size in the IoT context is by compressing it. Such technique is based on a stateful approach, utilizing what is usually called a 'context' on the IoT sensor and the gateway for communications between an IoT device and a server placed somewhere in the Internet - from the edge to the cloud.

The role of the 'context' is to provide a way to 'compress' the original IP header into a smaller one, using shorter address information and/or dropping some field(s); the context here serves as a kind of dictionary.

- \* Separate device from locator identifier: Approaches that can offer customized address length that is adequate for use in such constrained domains are preferred. Using different namespaces for the 'device identifier' and the 'routing' or 'locator identifier' is one such approach.

#### 3.1.1.3. Examples

- \* Header Compression/Translation: Considering one base station is supposed to serve hundreds of user devices, maximizing the effectiveness for specific spectrum directly improves user quality of experience. To achieve the optimal utilization of the spectrum resource in the wireless area, the RObust Header Compression (ROHC) [RFC5795] mechanism, which has been widely adopted in cellular network like WCDMA, LTE, and 5G, utilizes header compression to shrink existing IPv6 headers onto shorter ones.

Similarly, header compression techniques for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) have been around for several years now, constituting a main example of using the notion of a 'shared context' in order to reduce the size of the network layer header ([RFC6282], [RFC7400], [ITU9959]). More recently, other compression solutions have been proposed for Low Power Wide Area Networks (LPWAN - [RFC8376]). Among them, the Static Context Header Compression (SCHC - [RFC8724]) generalized the compression mechanism developed by 6lo. Instead of a standard compression behavior implemented in all 6lo nodes, SCHC introduces the notion of rule shared by two nodes. The SCHC compression technique is generic and can be applied to IPv6 and above layers. Regarding the nature of the traffic, IPv6 addresses (source and destination) can be elided, partially sent, or replaced by a small index. Instead of the versatile IP packet, SCHC defines new packet formats dedicated to specific applications. SCHC rules are equivalence functions mapping this format to standard IP packets.

Also, constraints coming from either devices or carrier links would lead to mixed scenarios and compound requirements for extraordinary header compression. For native IPv6 communications on DECT ULE and MS/TP Networks [RFC6282], dedicated compression mechanisms are specified in [RFC8105] and [RFC8163], while the transmission of IPv6 packets over NFC and PLC, specifications are being developed in [I-D.ietf-6lo-nfc] and [I-D.ietf-6lo-plc].

- \* Separate device from locator identifier: Solutions such as proposed in [EIBP] and [I-D.ietf-lisp-rfc6830bis] can utilize a separation of device from locator, where only the latter is used for routing between the different domains using the same technology, therefore enabling the use of shorter addresses in the (possibly constrained) local environment. Device IDs used within such domains are carried as part of the payload by EIBP and hence can be of shorter size suited to the domain, while, for instance, in LISP a flexible address encoding [RFC8060] allows shorter addresses to be supported in the LISP control plane [I-D.ietf-lisp-rfc6833bis].

### 3.1.2. Longer Address Length

#### 3.1.2.1. Description

Historically, obtaining adequate address space is considered as the primary and raw motivation to invent IPv6. Longer address (more than 32-bit of IPv4 address), which can accommodate almost inexhaustible devices, used to be considered as the surest direction in 1990s. Nevertheless, to protect the sunk cost of IPv4 deployment, certain efforts focus on IPv4 address space depletion question but engineer IPv4 address length in a more practical way. Such effort, i.e., NAT (Network Address Translation), unexpectedly and significantly slows IPv6 deployment because of its high cost-effectiveness in practice.

Another crucial need for longer address lengths comes from "semantic extensions" to IP addresses, where the extensions themselves do not fit within the length limitation of the IP address. Section 3.3 discusses extensions which extend address semantics that are not limited by the IP address length.

This sub-section focuses on address length extensions that aim at reducing the IPv4 addresses depletion, while Section 3.3, i.e., address semantic extensions, may still refer to extensions when longer address length are suitable to accommodate different address semantic. See Section 3.3 for details of semantic-driven address lengthening.

#### 3.1.2.2. Methodology

- \* Split address zone by network realm: This methodology first split the network realm into two types: one public realm (i.e., the Internet), and innumerable private realms (i.e., local networks, which may be embedded and/or having different scope). Then, it splits the IP address space into two type of zones: global address zone (i.e., public address) and local address zone (e.g., private address, reserved address). Based on this, it is assumed that in

public realm, all devices attached to it should be assigned an address that belongs to the global address zone. While for devices attached to private realms, only addresses belonging to the local address zone will be assigned. Local realms may have different scope or even be embedded one in another, like for instance, light switches local network being part of the building local network, which in turn connects to the Internet. In the local realms address may have a pure identification purpose. For instance in last example, addresses of the light switches identify the switches themselves, while the building local network is used to locate them.

Given that the local address zone is not globally unique, certain mechanisms are designed to express the relationship between the global address zone (in public realm) and the local address zone (in any private realm). In this case, global addresses are used for forwarding when a packet is in the public realm, and local addresses are used for forwarding when a packet is in a private realms.

#### 3.1.2.3. Examples

- \* Split address zone by network realm: Network Address Translation (NAT), which was first laid out in [RFC2663], using private address and a stateful address binding to translate between the realms. As outlined in [RFC2663], basic address translation is usually extended to include port number information in the translation process, supporting bidirectional or simple outbound traffic only. Because the 16-bits port number is used in the address translation, NAT theoretically increase IPv4 address length from 32-bit to 48-bit, i.e., 281 trillion address space.

Similarly, EzIP [EzIP] expects to utilize a reserved address block, i.e., 240/4, and an IPv4 header option to include it. Based on this, it can be regarded as EzIP is carrying a hierarchical address with two parts, where each part is a partial 32-bit IPv4 address. The first part is a public address residing in the "address field" of the header from globally routable IPv4 pool [IPv4pool], i.e., ca. 3.84 billion address space. The second part is the reserved address residing in "option field" and belongs to the 240/4 prefix, i.e., ca.  $2^{28}=268$  million. Based on that, each EzIP deployment is tethered on the existing Internet via one single IPv4 address, and EzIP then have  $3.84B * 268M$  address, ca. 1,000,000 trillion. Collectively, the 240/4 can also be used as end point identifier and form an overlay network providing services parallel to the current Internet, yet independent of the latter in other aspects.

Compared to NAT, EzIP is able to establish a communication session from either side of it, hence being completely transparent, and facilitating a full end-to-end networking configuration.

### 3.1.3. Summary

Table 1 summarizes methodologies and examples towards filling gaps on IP address length extensions.

	Methodology	Examples
Shorter Address Length	Header compression/ translation	6LoWPAN, ROHC, SCHC
	Separate device from locator identifier	EIBP, LISP, ILNP, HIP
Longer Address Length	Split address zone by network realm	NAT, EzIP

Table 1: Summary Length Extensions

### 3.2. Identity Extensions

Extensions in this subsection attempt extending the property described in Section 2.2, i.e., 'locator/identifier overload' of the ambiguous address semantic.

From the perspective of Internet users, on the one hand, the implicit identifier semantic results in a privacy issue due to network behavior tracking and association. Despite that IP address assignments may be dynamic, they are nowadays considered as 'personal data' and as such undergoes privacy protection regulations like General Data Protection Regulation ("GDPR" [GDPR]). Hence, additional mechanisms are necessary in order to protect end user privacy.

For network regulation of sensitive information, on the other hand, dynamically allocated IP addresses are not sufficient to guarantee device or user identification. As such, different address allocation systems, with stronger identification properties are necessary where security and authentication are at highest priority. Hence, in order to protect information security within a network, additional mechanism are necessary to identify the users or the devices attached to the network.

### 3.2.1. Anonymous Address Identity

#### 3.2.1.1. Description

As discussed in Section 2.2, IP addresses reveal both 'network locations' as well as implicit 'identifier' information to both traversed network elements and destination nodes alike. This enables recording, correlation, and profiling of user behaviors and historical network traces, possibly down to individual real user identity. The IETF, e.g., in [RFC7258], has taken a clear stand on preventing any such pervasive monitoring means by classifying them as an attack on end users' right to be left alone (i.e., privacy). Regulations such as the EU's General Data Protection Regulation (GDPR) classifies, for instance, the 'online identifier' as personal data which must be carefully protected; this includes end users' IP addresses [GDPR].

Even before pervasive monitoring [RFC7258], IP addresses have been seen as something that some organizational owners of networked system may not want to reveal at the individual level towards any non-member of the same organization. Beyond that, if forwarding is based on semantic extensions, like other fields of the header, extension headers, or any other possible extension, if not adequately protected it may introduce privacy leakage and/or new attack vectors.

#### 3.2.1.2. Methodology:

- \* **Traffic Proxy:** Detouring the traffic to a trusted proxy is a heuristic solution. Since nodes between trusted proxy and destination (including the destination per se) can only observe the source address of the proxy, the 'identification' of the origin source can thereby be hidden. To obfuscate the nodes between origin and the proxy, the traffic on such route would be encrypted via a key negotiated either in-band or off-band. Considering that all applications' traffic in such route can be seen as a unique flow directed to the same 'unknown' node, i.e., the trusted proxy, eavesdroppers in such route have to make more efforts to correlate user behavior through statistical analysis even if they are capable of identifying the users via their source addresses. The protection lays in the inability to isolate single application specific flows. According to the methodology, such approach is IP version independent and works for both IPv4 and IPv6.
- \* **Source Address Rollover:** Privacy issues related to address 'identifier' semantic can be mitigated through regular change (beyond the typical 24 hours lease of DHCP). Due to the semantics of 'identifier' that an IP address carries, such approach promotes

to change the source IP address at a certain frequency. Under such methodology, the refresh cycling window may reach to a balance between privacy protection and address update cost. Due to the limited space that IPv4 contains, such approach usually works for IPv6 only.

- \* Private Address Spaces: Their introduction in [RFC1918] foresaw private addresses (assigned to specific address spaces by the IANA) as a means to communicate purely locally, e.g., within an enterprise, by separating private from public IP addresses. Considering that private addresses are never directly reachable from the Internet, hosts adopting private addresses are invisible and thus 'anonymous' for the Internet. Besides, hosts for purely local communication used the latter while hosts requiring public Internet service access would still use public IP addresses.
- \* Address Translation: The aforementioned original intention for using private IP addresses, namely for purely local communication, resulted in a lack of flexibility in changing from local to public Internet access on the basis of what application would require which type of service.

If eventually every end-system in an organization would require some form of public Internet access in addition to local one, an adequate number of public Internet addresses would be required for providing to all end systems. Instead, address translation enables to utilize many private IP addresses within an organization, while only relying on one (or few) public IP addresses for the overall organization.

In principle, address translation can be applied recursively. This can be seen in modern broadband access where Internet providers may rely on carrier-grade address translation for all their broadband customers, who in turn employ address translation of their internal home or office addresses to those (private again) IP addresses assigned to them by their network provider.

Two benefits arise from the use of (private to public IP) address translation, namely (i) the hiding of local end systems at the level of the (address) assigned organization, and (ii) the reduction of public IP addresses necessary for communication across the Internet. While the latter has been seen for long as a driver for address translation, we focus on the first issue in this section, also since we see such privacy benefit as well as objective as still being valid in addressing systems like IPv6 where address scarcity is all but gone [GNATCATCHER].



- \* Separate device from locator identifier: Solutions that make a clear separation between the routing locator and the identifier, can allow for a device ID of any size, which in turn can be encrypted by a network element deployed at the border of routing domain (e.g., access/edge router). Both source and end-domain addresses can be encrypted and transported, as in the routing domain, only the routing locator is used.

#### 3.2.1.3. Examples:

- \* Traffic Proxy: Although not initially designed as a traffic proxy approach, a Virtual Private Network (VPN [VPN]) is widely utilized for packets origin hiding as a traffic detouring methodology. As it evolved, VPN derivatives like WireGuard [WireGuard] have become a mainstream instance for user privacy and security enhancement.

With such methodology in mind, onion routing [ONION], instantiated in the TOR Project [TOR], achieves high anonymity through traffic hand over via intermediates, before reaching the destination. Since the architecture of TOR requires at least three proxies, none of them is aware of the entire route. Given that the proxies themselves can be deployed all over cyberspace, trust is not the prerequisite if proxies are randomly selected.

In addition, dedicated protocols are also expected to be customized for privacy improvement via traffic proxy. For example, Oblivious DNS over HTTPS (ODOH [ODOH]) use a third-party proxy to obscure identifications of user source addresses during DNS over HTTPS (DoH [RFC8484]) resolution. Similarly, Oblivious HTTP [OHTTP] involve proxy alike in the HTTP environment.

- \* Source Address Rollover: As for source address rollover, it has been standardized that IP addresses for Internet users should be dynamic and temporary every time they are being generated [RFC8981]. This benefits from the available address space in the case of IPv6, through which address generation or assignment should be unpredictable and stochastic for outside observers.

More radically, [EPHEMERALv6] advocates an 'ephemeral address', changing over time, for each process. Through this, correlating user behaviors conducted by different identifiers (i.e., source address) becomes much harder, if not impossible, if based on the IP packet header alone.

- \* Private Addresses: The use and assignment of private addresses for IPv4 is laid out in [RFC1918], while unique local addresses (ULAs) in IPv6 [RFC4193] take over the role of private address spaces in IPv4.

- \* Network Address Translation: Given address translation can be performed several times in cascade, NATs may exist as part of existing customer premise equipment (CPE), such as a cable or an Ethernet router, with private wired/wireless connectivity, or may be provided in a carrier environment to further translate ISP-internal private addresses to a pool of (assigned) public IP addresses. The latter is often dynamically assigned to CPEs during its bootstrapping.
- \* Separate device from locator identifier: EIBP [EIBP] utilizes a structured approach to addressing. It separates the routing ID from the device ID, where only the former is used for routing. As such, the device IDs can be encrypted, protecting the end device identity. Similarly, LISP uses separate namespaces for routing and identification allowing to 'hide' identifiers in encrypted LISP packets that expose only known routing information [RFC8061].

### 3.2.2. Authenticated Address Identity

#### 3.2.2.1. Description

In some scenarios (e.g., corporate networks) it is desirable to being able authenticate IP addresses in order to prevent malicious attackers spoofing IP addresses. This is usually achieved by using a mechanism that allows to prove ownership of the IP address.

#### 3.2.2.2. Methodology

- \* Self-certified addresses: This method is usually based on the use of nodes' public/private keys. A node creates its own interface ID (IID) by using a cryptographic hash of its public key (with some additional parameters). Messages are then signed using the nodes' private key. The destination of the message will verify the signature through the information in the IP address. Self-certification has the advantage that no third party or additional security infrastructure is needed. Any node can generate its own address locally and then only the address and the public key are needed to verify the binding between the public key and the address.

- \* Third party granted addresses: DHCP (Dynamic Host Configuration Protocol) is widely used to provide IP addresses, however, in its basic form, it does not perform any check and even an unauthorized user without the right to use the network can obtain an IP address. To solve this problem, a trusted third party has to grant access to the network before generating an address (via DHCP or other) that identifies the user. User authentication done securely either based on physical parameters like MAC addresses or based on an explicit login/password mechanism.

#### 3.2.2.3. Examples

- \* Self-certified Addresses: As an example of this methodology serves [RFC3972], defining IPv6 cryptographically Generated Addresses (CGA). A Cryptographically Generated Address is formed by replacing the least-significant 64 bits of an IPv6 address with the cryptographic hash of the public key of the address owner. Packets are then signed with the private key of the sender. Packets can be authenticate by the receiver by using the public key of the sender and the address of the sender. The original specifications have been already amended (cf., [RFC4581] and [RFC4982]) in order to support multiple (stronger) cryptographic algorithms.
- \* Third party granted addresses: [RFC3118] defines a DHCP option through which authorization tickets can be generated and newly attached hosts with proper authorization can be automatically configured from an authenticated DHCP server. Solutions exist where separate servers are used for user authentication like [UA-DHCP] and [RFC4014]. The former proposing to enhance the DHCP system using registered user login and password before actually providing an IP address lease and recording the MAC address of the device the user used to sign-in. The latter, couples the RADIUS authentication protocol ([RFC2865]) with DHCP, basically piggybacking RADIUS attributes in a DHCP sub-option, with the DHCP server contacting the RADIUS server to authenticate the user.

#### 3.2.3. Summary

Table 2, summarize the methodologies and the examples towards filling the gaps on identity extensions.

	Methodology	Examples
Anonymous Address Identity	Traffic Proxy	VPN, TOR, ODoH
	Source Address Rollover	SLAAC
	Private Address Spaces	ULA
	Address Translation	NAT
	Separate device from locator identifier	EIBP, LISP
Authenticated Address Identity	Self-certified Addresses	CGA
	Third party granted addresses	DHCP-Option

Table 2: Summary Identity Extensions

### 3.3. Semantic Extensions

Extensions in this subsection try extending the property described in Section 2.3, i.e., limited address semantic support.

As explained in Section 2.2, IP addresses carry both locator and identification semantic. Some efforts exist that try to separate these semantics either in different address spaces or through different address formats. Beyond just identification, location, and the fixed address size, other efforts extended the semantic through existing or additional header fields (or header options) outside the Internet address.

How much unique and globally routable an address should be? With the effect of centralization, edges communicate with (rather) local DCs, hence a unique address globally routable is not a requirement anymore. There is no need to use globally unique addresses all the time for communication, however, there is the need of having a unique address as a general way to communicate to any connected entity without caring what transmission networks the packets traverse.

### 3.3.1. Utilizing Extended Address Semantics

#### 3.3.1.1. Description

Several extensions have been developed to extend beyond the limited IPv6 semantics. Those approaches may include to apply structure to the address, utilize specific prefixes, or entirely utilize the IPv6 address for different semantics, while re-encapsulating the original packet to restore the semantics in another part of the network. For instance, structured addresses have the capability to introduce delimiters to identify semantic information in the header, therefore not constraining any semantic by size limitations of the address fields.

We note here that extensions often start out as being proposed as an extended header semantic, while standardization may drive the solution to adopt an approach to accommodate their semantic within the limitations of an IP address. This section does include examples of this kind.

#### 3.3.1.2. Methodology

\*Semantic prefixes: Semantic prefixes are used to separate the IPv6 address space. Through this, new address families, such as for information-centric networking [HICN], service routing or other semantically rich addressing, can be defined, albeit limited by the prefix length and structure as well as the overall length limitation of the IPv6 address.

\* Separate device/resource from locator identifier: The option to use separate namespaces for the device address would offer more freedom for the use of different semantics. For instance, the static binding of IP addresses to servers creates a strong binding between IP addresses and service/resources, which may be a limitation for large Content Distribution networks (CDNs) [FAYED21].

As an extreme form of separating resource from locator identifier, recent engineering approaches, described in [CLOUDFLARE\_SIGCOMM], decouple web service (semantics) from the routing address assignments by using virtual hosting capabilities, thereby effectively mapping possibly millions of services onto a single IP address.

\* Structured addressing: One approach to address the routing challenges faced in the Internet is the use of structured addresses, e.g., to void the need for routing protocols. Benefits of this approach can be significant, with the structured addresses

capturing the relative physical or virtual position of routers in the network as well as being variable in length. Key to the approach, however, is that the structured addresses capturing the relative physical or virtual position of routers in the network, or networks in an internetwork may not fit within the fixed and limited IP address length (cf., Section 3.1.2). Other structured approaches may be the use of application-specific structured binary components for identification, generalizing URL schema used for HTTP-level communication but utilized at the network level for traffic steering decisions.

- \* Localized forwarding semantics: Layer 2 hardware, such as SDN switches, are limited to the use of specific header fields for forwarding decisions. Hence, devising new localized forwarding mechanisms may be based on re-using differently existing header fields, such as the IPv6 source/destination fields, to achieve the desired forwarding behavior, while encapsulating the original packets in order to be restored at the local forwarding network boundary. Networks in those solutions are limited by the size of the utilized address field, e.g., 256 bits for IPv6, thereby limiting the way such techniques could be used.

#### 3.3.1.3. Examples

- \* Semantic prefixes: Newer approaches to IP anycast suggest the use of service identification in combination with a binding IP address model [SFCANYCAST] as a way to allow for metric-based traffic steering decisions; approaches for Service Function Chaining (SFC) [RFC7665] utilize the Network Service Header (NSH) information and packet classification to determine the destination of the next service.

Another example of the usage of different packet header extensions based on IP addressing is Segment Routing. In this case, the source chooses a path and encodes it in the packet header as an ordered list of segments. Segments are encoded using new Routing Extensions Header type, the Segment Routing Header (SRH), which contains the Segment List, similar to what is already specified in [RFC8200], i.e., a list of segment ID (SID) that dictate the path to follow in the network. Such segment IDs are coded as 128 bit IPv6 addresses [RFC8986].

Approaches such as [HICN] utilize semantic prefixing to allow for ICN forwarding behavior within an IPv6 network. In this case, an HICN name is the hierarchical concatenation of a name prefix and a name suffix, in which the name prefix is encoded as an IPv6 128 bits word and carried in IPv6 header fields, while the name suffix is encoded in transport headers fields such as TCP. However, it

is a challenge to determine which IPv6 prefixes should be used as name prefixes. In order to know which IPv6 packets should be interpreted based on an ICN semantic, it is desirable to be able to recognize that an IPv6 prefix is a name prefix, e.g. to define a specific address family (AF\_HICN, b0001::/16). This establishment of a specific address family allows the management and control plane to locally configure HICN prefixes and announce them to neighbors for interconnection.

- \* Separate device from locator identifier: EIBP [EIBP] separates the routing locator from the device identifier, relaxing therefore any semantic constraints on the device identifier. Similarly, LISP uses a flexible encoding named LISP Canonical Address Format (LCAF [RFC8061]), which allows to associate to routing locators any possible form (and length) of identifier. ILNP [RFC6740] introduces as well a different semantic of IP addresses, while aligning to the IPv6 address format (128 bits). Basically, ILNP introduces a sharper logical separation between the 64 most significant bits and the 64 least significant bits of an IPv6 address. The former being a global locator, while the latter being an identifier that can have different semantics (rather than just being an interface identifier).
- \* Structured addressing: Network topology captures the physical connectivity among devices in the network. There is a structure associated with the topology. Examples are the core-distribution-access router structure commonly used in enterprise networks and clos topologies that are used to provide multiple connections between Top of Rack (ToR) devices and multiple layers of spine devices. Internet service providers use a tier structure that defines their business relationships. A clear structure of connected networks can be noticed in the Internet. EIBP [EIBP] proposes to leverage the physical structure (or a virtual structure overlaid on the physical structure) to auto assign addresses to routers in a network or networks in an internetwork to capture their relative position in the physical/virtual topology. EIBP proposes to administratively identify routers/networks with a tier value based on the structure.
- \* Localized forwarding semantics: Approaches such as those outlined in [REED] suggest using a novel forwarding semantic based on path information carried in the packet itself, said path information consists in a fixed size bit-field (see [REED] for more information on how to represent the path information in said bit-field). In order to utilize existing, e.g., SDN-based, forwarding switches, the direct use of the IPv6 source/destination address is suggested for building appropriate match-action rules (over the suitable binary information representing the local output ports),

while preserving the original IPv6 information in the encapsulated packet. As mentioned above, such use of the existing IPv6 address fields limits the size of the network to a maximum of 256 bits (therefore paths in the network over which such packets can be forwarded). [ICNIP], however, goes a step further by suggesting to use the local forwarding as direct network layer mechanism, removing the IP packet and only leaving the transport/application layer, with the path identifier constituting the network-level identifier albeit limited by using the existing IP header for backward compatibility reasons (the next section outlines the removal of this limitation).

### 3.3.2. Utilizing Existing or Extended Header Semantics

#### 3.3.2.1. Description:

While the former sub-section explored extended address semantic, thereby limiting any such extended semantic with that of the existing IPv6 semantic and length, additional semantics may also be placed into the header of the packet or the packet itself, utilized for the forwarding decision to the appropriate endpoint according to the extended semantic.

Reasons for embedding such new semantics may be related to traffic engineering since it has long been shown that the IP address itself is not enough to steer traffic properly since the IP address itself is not semantically rich enough to adequately describe the forwarding decision to be taken in the network, not only impacting WHERE the packet will need to go but also HOW it will need to be sent.

#### 3.3.2.2. Methodology:

- \* In-Header extensions: One way to add additional semantics besides the address fields is to use other fields already present in the header.
- \* Headers option extensions: Another mechanism to add additional semantics is to actually add additional fields, e.g., through Header Options in IPv4 or through Extension Headers in IPv6.
- \* Re-encapsulation extension: A more radical approach for additional semantics is the use of a completely new header that is designed so to carry the desired semantics in an efficient manner (often as a shim header).
- \* Structured addressing: Similar to the methodology that structures addresses within the limitations of the IPv6 address length, outlined in the previous sub-sections, structured addressing can



also be applied within existing or extended header semantics, e.g., utilizing a dedicated (extension) header to carry the structured address information.

- \* Localized forwarding semantics: This set of solutions applies capabilities of newer (programmable) forwarding technology, such as [P4], to utilize any header information for a localized forwarding decision. This removes any limitation to use existing header or address information for embedding a new address semantic into the transferred packet.

### 3.3.2.3. Examples:

- \* In-Header extensions: In order to allow additional semantic with respect to the pure Internet addressing, the original design of IPv4 included the field 'Type of Service' [RFC2474], while IPv6 introduced the 'Flow label' and the 'Traffic Class' [RFC8200]. In a certain way, those fields can be considered 'semantic extensions' of IP addresses, and they are 'in-header' because natively present in the IP header (differently from options and extension headers). However, they proved not to be sufficient. Very often a variety of network operation are performed on the well-known 5-tuple (source and destination addresses; source and destination port number; and protocol number). In some contexts all of the above mentioned fields are used in order to have a very fine grained solution ([RFC8939]).
- \* Headers option extensions: Header options have been largely under-exploited in IPv4. However, the introduction of the more efficient extension header model in IPv6 along with technology progress made the use of header extensions more widespread in IPv6. Segment Routing re-introduced the possibility to add path semantic to the packet by encoding a loosely defined source routing ([RFC8402]). Similarly, in the aim to overcome the inherent shortcoming of the multi-homing in the IP context, SHIM6 ([RFC5533]) also proposed the use of an extension header able to carry multi-homing information which cannot be accommodated natively in the IPv6 header.

To serve a moving endpoint, mechanisms like Mobile IPv6 [RFC6275] are used for maintaining connection continuity by a dedicated IPv6 extension header. In such case, the IP address of the home agent in Mobile IPv6 is basically an identification of the on-going communication. In order to go beyond the interface identification model of IP, the Host Identity Protocol (HIP) tries to introduce an identification layer to provide (as the name says) host identification. The architecture here relies on the use of another type of extension header [RFC7401].

- \* Re-encapsulation extension: Differently from the previous approach, re-encapsulation prepends complete new IP headers to the original packet introducing a completely custom shim header between the outer and inner header. This is the case for LISP, adding a LISP specific header right after an IP+UDP header ([I-D.ietf-lisp-rfc6830bis]). A similar design is used by VxLAN ([RFC7348]) and GENEVE ([RFC8926]), even if they are designed for a data center context. IP packets can also be wrapped with headers using more generic and semantically rich names, for instance with ICN [ICNIP].
- \* Structured addressing: Solutions such as those described in the previous sub-section, e.g., EIBP [EIBP], can provide structured addresses that are not limited to the IPv6 address length but instead carry the information in an extension header to remove such limitation.

Also Information-Centric Networking (ICN) naming approaches usually introduce structures in the (information) names without limiting themselves to the IP address length; more so, ICN proposes its own header format and therefore radically breaks with not only IP addressing semantic but the format of the packet header overall. For this, approaches such as those described in [RFC8609] define a TLV-based binary application component structure that is carried as a 'name' part of the CCN messages. Such a name is a hierarchical structure for identifying and locating a data object, which contains a sequence of name components. Names are coded based on 2-level nested Type-Length-Value (TLV) encodings, where the name-type field in the outer TLV indicates this is a name, while the inner TLVs are name components including a generic name component, an implicit SHA-256 digest component and a SHA-256 digest of Interest parameters. For textual representation, URIs are normally used to represent names, as defined in [RFC3986].

In geographic addressing, position based routing protocols use the geographic location of nodes as their addresses, and packets are forwarded when possible in a greedy manner towards the destination. For this purpose, the packet header includes a field coding the geographic coordinates (x, y, z) of the destination node, as defined in [RFC2009]. Some proposals also rely on extra fields in the packet header to code the distance towards the destination, in which case only the geographic coordinates of neighbors are exchanged. This way the location of the destination is protected even if routing packets are eavesdropped.

- \* Localized forwarding semantics: Unlike the original suggestion in [REED] to use existing SDN switches, the proliferation of P4 [P4] opens up the possibility to utilize a locally limited address semantic, e.g., expressed through the path identifier, as an entirely new header (including its new address) with an encapsulation of the IP packet for E2E delivery (including further delivery outside the localized forwarding network or positioning the limited address semantic directly as the network address semantic for the packet, i.e., removing any IP packet encapsulation from the forwarded packet, as done in [ICNIP]). Removing the IPv6 address size limitation by not utilizing the existing IP header for the forwarding decision also allows for extensible length approaches for building the path identifier with the potential for increasing the supported network size. On the downside, this approach requires to encapsulate the original IP packet header for communication beyond the local domain in which the new header is being used, such as discussed in the previous point above on 're-encapsulation extension'.

### 3.3.3. Summary

Table 3, summarize the methodologies and the examples towards filling the gaps on semantic extensions.

	Methodology	Examples
Utilizing Extended Address Semantics	Semantic prefixes	HICN
	Separate device from locator identifier	EIBP, ILNP, LISP, HIP
	Structured addressing	EIBP, ILNP
	Localized forwarding semantics	REED
Utilizing Existing or Extended Header Semantics	In-Header extensions	DetNet
	Headers option extensions	SHIM6, SRv6, HIP
	Re-encapsulation extension	VxLAN, ICNIP
	Structured addressing	EIBP
	Localized forwarding semantics	REED

Table 3: Summary Semantic Extensions

#### 4. Overview of Approaches to Extend Internet Addressing

The following Table 4 describes the objectives of the extensions discussed in this memo with respect to the properties of Internet addressing (Section 2). As summarized, extensions may aim to extend one property of the Internet addressing, or extend other properties at the same time.

	Length Extension	Identity Extension	Semantic Extension
6LoWPAN	x		
ROHC	x		

EzIP	x		
TOR		x	
ODoH		x	
SLAAC		x	
CGA		x	x
NAT	x	x	
HICN		x	x
ICNIP	x	x	x
CCNx names	x	x	x
EIBP	x	x	x
Geo addressing	x		x
REED	x (with P4)		x
DetNet		x	
Mobile IP			x
SHIM6			x
SRv6			x
HIP		x	x
VxLAN		x	x
LISP		x	x
SFC		x	x

Table 4: Relationship between Extensions and Internet Addressing

## 5. A System View on Address

In the following, we investigate in which parts of the overall Internet system extensions have been proposed and developed. For this, we divide the possible innovation across two dimensions:

- \* Horizontal: Internet edge vs core. The criticality, scale, investment on the core of the Internet makes it more difficult to introduce innovation, while at the edges there is more flexibility. As general purpose processors have drastically improved in performance, data-plane features can be implemented in software. At the edge of the Internet, it is easier to introduce innovation for several reasons: Economics, faster ROI because of faster deployment; No need of large scale deployment (and hence less standardization effort); less stakeholders involved (sometimes just one, see following point). Furthermore, the fact that the edge is a place where there is less coordination and cooperation from the core, is another factor that eases the innovation.
- \* Vertical: at which layer of the protocol stack. The difficulty to innovate varies as well depending at which layer the innovation takes place. One thing is to innovate at application layer where the app developer has large degree of freedom, another is to innovate at network layer, which is more constrained because of its central point in the architecture. Innovation at higher layer sometimes leads to walled gardens (aka limited domains [RFC8799]). Indeed because of the centralization phenomena, an actor offering a certain service may very well develop and deploy a custom technology that does not need to be actually standardized because it is done for its own internal usage.
- \* Horizontal vs Vertical Innovation:
  - In the public Internet, core innovation at lower layer is harder, often reduced to app-level innovation or building an overlay limited domain (aka a walled garden).
  - At the edges it is easier to innovate at lower layers (more vertical flexibility) but some form of adaptation is needed if global reachability is wanted.

Despite these two orthogonal dimensions, innovation does not happen either horizontally or vertically, rather in both dimensions simultaneously at various degree.

## 6. Issues in Extensions to Internet Addressing

While the extensions to the original Internet properties, discussed in Section 3, demonstrate the benefits of more flexibility in addressing, they also bring with them a number of issues, which are discussed in the following section. To this end, the problems hereafter outlined link to the approaches to extensions summarized in Section 4. These issues may not be present all the time and everywhere, since as explained in Section 5, extensions are developed and deployed in different part of the Internet, which may worsen things.

### 6.1. Limiting Address Semantics

Many approaches changing the semantics of communication, e.g., through separating host identification from network node identification [RFC7401], separating the device identifier from the routing locator ([EIBP], [I-D.ietf-lisp-introduction]), or through identifying content and services directly [HICN], are limited by the existing packet size and semantic constraints of IPv6, e.g., in the form of its source and destination network addresses.

While approaches such as [ICNIP] may override the addressing semantics, e.g., by replacing IPv6 source and destination information with path identification, a possible unawareness of endpoints still requires the carrying of other address information as part of the payload.

Also, the expressible service or content semantic may be limited, as in [HICN] or the size of supported networks [REED] due to relying on the limited bit positions usable in IPv6 addresses.

### 6.2. Complexity and Efficiency

A crucial issue is the additional complexity introduced for realizing the additional addressing semantics. This is particularly an issue since we see those additional semantics particularly at the edge of the Internet, utilizing the existing addressing semantic of the Internet to interconnect the domains that require those additional semantics.

Furthermore, any additional complexity often comes with an efficiency and cost penalty, particularly at the edge of the network, where resource constraints may play a significant role. Compression processes, taking [ROHC] as an example, require additional resources both for the sender generating the compressed header but also the gateway linking to the general Internet by re-establishing the full IP header.

Conversely, the performance requirements of core networks, in terms of packet processing speed, makes the accommodation of extensions to addressing often prohibitive. This is not only due to the necessary extra processing that is specific to the extension, but also due to the complexity that will need to be managed in doing so at significantly higher speeds than at the edge of the network. The observations on the dropping of packets with IPv6 extension headers in the real world is (partially) due to such a implementation complexity [RFC7872].

Another example for lowering the efficiency of packet forwarding is the routing in systems like TOR [TOR]. As detailed before, traffic in TOR, for anonymity purposes, should be handed over by at least three intermediates before reaching the destination. Frequent relaying enhances the privacy, however, because such kind of solutions are implemented at application level, they come at the cost of lower communication efficiency. May be a different privacy enhanced address semantic would enable efficient implementation of TOR-like solutions at network layer.

#### 6.2.1. Repetitive encapsulation

Repetitive encapsulation is an issue since it bloats the packets size due to additional encapsulation headers. Addressing proposals such as those in [ICNIP] utilize path identification within an alternative forwarding architecture that acts upon the provided path identification. However, due to the limitation of existing flow-based architectures with respect to the supported header structures (in the form of IPv4 or IPv6 headers), the new routing semantics are being inserted into the existing header structure, while repeating the original, sender-generated header structure, in the payload of the packet as it traverses the local domain, effectively doubling the per-packet header overhead.

The problem is also present in a number of solutions tackling different issues, e.g., mobility [I-D.ietf-lisp-mn], DC networking ([RFC8926], [RFC7348], [I-D.ietf-intarea-gue]), traffic engineering [RFC8986], and privacy ([TOR], [SPHINX]). Certainly these solutions are able to avoid other issues, like path lengthening or privacy issues, as described before, but they come at the price of multiple encapsulations that reduce the effective payload. This, not only hampers efficiency in terms of header-to-payload ratio, but also introduces 'encapsulation points', which in turn add complexity to the (often edge) network as well as fragility due to the addition of possible failure points; this aspect is discussed in further details in Section 6.4.



#### 6.2.2. Compounding issues with header compression

IP header overhead requires header compression in constrained environments, such as wireless sensor networks and IoT in general. Together with fragmentation, both tasks constitute significant energy consumption, as shown in [HEADER\_COMP\_ISSUES1], negatively impacting resource limited devices that often rely on battery for operation. Further, the reliance on the compression/decompression points creates a dependence on such gateways, which may be a problem for intermittent scenarios.

According to the implementation of `_contiki-ng_` [CONTIKI], an example of operating system for IoT devices, the source codes for 6LowPan requires at least 600Kb to include a header compression process. In certain use cases, such requirement can be an obstacle for extremely constrained devices, especially for the RAM and energy consumption.

#### 6.2.3. Introducing Path Stretch

Mobile IP [RFC6275], which was designed for connection continuity in the face of moving endpoints, is a typical case for path stretch. Since traffic must follow a triangular route before arriving at the destination, such detour routing inevitably impacts transmission efficiency as well as latency.

#### 6.2.4. Complicating Traffic Engineering

While many extensions to the original IP address semantic target to enrich the decisions that can be taken to steer traffic, according to requirements like QoS, mobility, chaining, compute/network metrics, flow treatment, path usage, etc., the realization of the mechanisms as individual solutions likely complicates the original goal of traffic engineering when individual solutions are being used in combination. Ultimately, this may even prevent the combined use of more than one mechanism and/or policy with a need to identify and prevent incompatibilities of mechanisms. Key here is not the issue arising from using conflicting traffic engineering policies, rather conflicting realizations of policies that may well generally work well alongside ([ROBUSTSDN], [TRANSACTIONSDN]).

This not only increases fragility, as discussed separately in Section 6.4, but also requires careful planning of which mechanisms to use and in which combination, likely needing human-in-the-loop approaches alongside possible automation approaches for the individual solutions.

### 6.3. Security

The properties described in Section 2 have, obviously, also consequences in terms of security and privacy related issues, as already mentioned in other parts of this document.

For instance, in the effort of being somehow backward compatible, HIP [RFC7401] uses a 128-bit Host Identity, which may be not sufficiently cryptographically strong in the future because of the limited size (future computational power may erode 128-bit security). Similarly, CGA [RFC3972] also aligns to the 128-bit limit, but may use only 59 bits of them, hence, the packet signature may not be sufficiently robust to attacks [I-D.rafiiee-6man-cga-attack].

IP addresses, even temporary ones meant to protect privacy, have been long recognized as a 'Personal Identification Information' that allows even to geolocate the communicating endpoints [RFC8280]. The use of temporary addresses provides sufficient privacy protection only if the renewal rate is high [EPHEMERALv6]. However, this causes additional issues, like the large overhead due to the Duplicate Address Detection, the impact on the Neighbor Discovery mechanism, in particular the cache, which can even lead to communication disruption. With such drawbacks, the extensions may even lead to defeat the target, actually lowering security rather than increasing it.

The introduction of alternative addressing semantics has also been used to help in (D)DoS attacks mitigation. This leverages on changing the service identification model so to avoid topological information exposure, making the potential disruptions likely remain limited [ADDRLESS]. However, this increased robustness to DDoS comes at the price of important communication setup latency and fragility, as discussed next.

### 6.4. Fragility

From the extensions discussed in Section 3, it is evident that having alternative or additional address semantic and formats available for making routing as well as forwarding decisions dependent on these, is common place in the Internet. This, however, adds many extension-specific translation/adaptation points, mapping the semantic and format in one context into what is meaningful in another context, but also, more importantly, creating a dependency towards an additional component, often without explicit exposure to the endpoints that originally intended to communicate.

For instance, the re-writing of IP addresses to facilitate the use of private address spaces throughout the public Internet, realized through network address translators (NATs), conflicts with the end-to-end nature of communication between two endpoints. Additional (flow) state is required at the NAT middle-box to smoothly allow communication, which in turn creates a dependency between the NAT and the end-to-end communication between those endpoints, thus increasing the fragility of the communication relation.

A similar situation arises when supporting constrained environments through a header compression mechanism, adding the need for, e.g., a ROHC [RFC5795] element in the communication path, with communication-related compression state being held outside the communicating endpoints. Failure will introduce some inefficiencies due to context regeneration, which may affect the communicating endpoints, increasing fragility of the system overall.

Such translation/adaptation between semantic extensions to the original 'semantic' of an IP address is generally not avoidable when accommodating more than a single universal semantic. However, the solution-specific nature of every single extension is likely to noticeably increase the fragility of the overall system, since individual extensions will need to interact with other extensions that may be deployed in parallel, but were not designed taking into account such deployment scenario (cf., [I-D.ietf-intarea-tunnels]). Considering that extensions to traditional per-hop-behavior (based on IP addresses) can essentially be realized over almost 'any' packet field, the possible number of conflicting behaviors or diverging interpretation of the semantic and/or content of such fields, among different extensions, may soon become an issue, requiring careful testing and delineation at the boundaries of the network within which the specific extension has been realized.

## 7. Summary of issues

Table 5, derived from Section 6, summarizes the issues related to each extension. While each extension involves at least one issue, some others, like ICNIP, may create several issues at the same time.

	Limiting Address Semantics	Complexity and Efficiency	Security	Fragility
6LoWPAN		x		x
ROHC		x		x

EzIP		x		
TOR		x		x
ODoH		x		
SLAAC		x		
CGA	x		x	
NAT		x		x
HICN	x			
ICNIP	x	x		
CCNx name	x			
EIBP				x
Geo addressing	x			x
REED	x			
DetNet		x		
Mobile IP		x		x
SHIM6				x
SRv6				x
HIP			x	x
VxLAN		x		
LISP		x		x
SFC		x		x

Table 5: Issues in Extensions to Internet Addressing

## 8. Conclusions

The examples of extensions discussed in Section 3 to the original Internet addressing scheme show that extensibility beyond the original model (and its underlying per-hop behavior) is a desired capability for networking technologies and has been so for a long time. Generally, we can observe that those extensions are driven by the requirements of stakeholders, expecting a desirable extended functionality from the introduction of the specific extension. If interoperability is required, those extensions require standardization of possibly new fields, new semantics as well as (network and/or end system) operations alike.

The issues we identified in this document with the extension-specific solution approach, point to the need for a discussion on Internet addressing, as formulated in the companion document [I-D.jia-intarea-scenarios-problems-addressing] that formalizes the problem statement through scenarios that highlight the shortcomings of the Internet addressing model.

It is our conclusion that the existence of the many extensions to the original Internet addressing is clear evidence for gaps that have been identified over time by the wider Internet community, each of which come with a raft of issues that we need to deal with daily: We believe that it is time to develop an architectural but more importantly a sustainable approach to make Internet addressing extensible in order to capture the many new use cases that will still be identified for the Internet to come.

To jumpstart any such effort from an addressing perspective, it will be key to suitably define what an address is at which layer of the overall system, let alone the network layer. We argue that any answer to this question must be derived from what features we may want from the network instead of being guided by the answers that the Internet can give us today, e.g., being a mere ephemeral token for accessing PoP-based services (as indicated in related arch-d mailing list discussions).

This is not to 'second guess' the market and its possible evolution, but to outline clear features from which to derive clear principles for a design. Any such design must not skew the technical capabilities of addressing to the current economic situation of the Internet since this bears the danger of locking down innovation capabilities as an outcome of those technical limitations introduced. Instead, addressing must be aligned with enabling the model of permissionless innovation that the IETF has been promoting, ultimately enabling the serendipity of new applications that has led to many of those applications we can see in the Internet today. Most

importantly, any inaction on our side in that regard will only compound the issues identified, eventually hampering the future Internet's readiness for those new uses.

## 9. Security Considerations

The present memo does not introduce any new technology and/or mechanism and as such does not introduce any security threat to the TCP/IP protocol suite.

As an additional note, and as discussed in this document, security and privacy aspects were not considered as part of the key properties for Internet addressing, which led to the introduction of a number of extensions intending to fix those gaps. The analysis presented in this memo (non-exhaustively) shows those issues are either solved in an ad-hoc manner at application level, or at transport layer, while at network level only few extensions tackling specific aspects exist, albeit often with limitations due to the adherence to the Internet addressing model and its properties.

## 10. IANA Considerations

This document does not include any IANA request.

## 11. Informative References

[ADDRLESS] Hao, S., Liu, R., Weng, Z., Chang, D., Bao, C., and X. Li, "Addressless: A new internet server model to prevent network scanning", PLOS ONE Vol. 16, pp. e0246293, DOI 10.1371/journal.pone.0246293, February 2021, <<https://doi.org/10.1371/journal.pone.0246293>>.

[CLOUDFLARE\_SIGCOMM] Fayed, M., Bauer, L., Giotsas, V., Kerola, S., Majkowski, M., Odintsov, P., Sitnicki, J., Chung, T., Levin, D., Mislove, A., Wood, C., and N. Sullivan, "The ties that unbind: decoupling IP from web services and sockets for robust addressing agility at CDN-scale", Proceedings of the 2021 ACM SIGCOMM 2021 Conference, DOI 10.1145/3452296.3472922, August 2021, <<https://doi.org/10.1145/3452296.3472922>>.

[CONTIKI] "Contiki-NG: The OS for Next Generation IoT Devices", n.d., <<https://github.com/contiki-ng/contiki-ng>>.

- [EIBP] Shenoy, S Chandraiah, P Willis, N., "A Structured Approach to Routing in the Internet", June 2021, <First Intl Workshop on Semantic Addressing and Routing for Future Networks>.
- [EPHEMERALv6] Gont, F. and G. Gont, "IPv6 Addressing Considerations", Work in Progress, Internet-Draft, draft-gont-v6ops-ipv6-addressing-considerations-01, 21 February 2021, <<https://www.ietf.org/archive/id/draft-gont-v6ops-ipv6-addressing-considerations-01.txt>>.
- [EzIP] Chen, A. Y., Ati, R. R., Karandikar, A., and D. R. Crowe, "Adaptive IPv4 Address Space", Work in Progress, Internet-Draft, draft-chen-ati-adaptive-ipv4-address-space-10, 8 December 2021, <<https://www.ietf.org/archive/id/draft-chen-ati-adaptive-ipv4-address-space-10.txt>>.
- [FAYED21] Fayed, M., Bauer, L., Giotsas, V., Kerola, S., Majkowski, M., Odintsov, P., Sitnicki, J., Chung, T., Levin, D., Mislove, A., Wood, C., and N. Sullivan, "The ties that unbind: decoupling IP from web services and sockets for robust addressing agility at CDN-scale", Proceedings of the 2021 ACM SIGCOMM 2021 Conference, DOI 10.1145/3452296.3472922, August 2021, <<https://doi.org/10.1145/3452296.3472922>>.
- [GDPR] Voigt, P. and A. von dem Bussche, "The EU General Data Protection Regulation (GDPR)", Springer International Publishing book, DOI 10.1007/978-3-319-57959-7, 2017, <<https://doi.org/10.1007/978-3-319-57959-7>>.
- [GNATCATCHER] "Global Network Address Translation Combined with Audited and Trusted CDN or HTTP-Proxy Eliminating Reidentification", n.d., <<https://github.com/bslassey/ip-blindness>>.
- [HEADER\_COMP\_ISSUES1] Mesrinejad, F., Hashim, F., Noordin, N., Rasid, M., and R. Abdullah, "The effect of fragmentation and header compression on IP-based sensor networks (6LoWPAN)", The 17th Asia Pacific Conference on Communications, DOI 10.1109/apcc.2011.6152926, October 2011, <<https://doi.org/10.1109/apcc.2011.6152926>>.

- [HICN] Muscariello, L., "Hybrid Information-Centric Networking: ICN inside the Internet Protocol", March 2018, <<https://datatracker.ietf.org/meeting/interim-2018-icnrg-01/materials/slides-interim-2018-icnrg-01-sessa-hybrid-icn-hicn-luca-muscariello>>.
- [HISTORY127] "History of 127/8 as localhost/loopback addresses", n.d., <<https://elists.isoc.org/pipermail/internet-history/2021-January/006920.html>>.
- [I-D.ietf-6lo-nfc] Choi, Y., Hong, Y., Youn, J., Kim, D., and J. Choi, "Transmission of IPv6 Packets over Near Field Communication", Work in Progress, Internet-Draft, draft-ietf-6lo-nfc-17, 23 August 2020, <<https://www.ietf.org/archive/id/draft-ietf-6lo-nfc-17.txt>>.
- [I-D.ietf-6lo-plc] Hou, J., Liu, B., Hong, Y., Tang, X., and C. E. Perkins, "Transmission of IPv6 Packets over PLC Networks", Work in Progress, Internet-Draft, draft-ietf-6lo-plc-10, 17 February 2022, <<https://www.ietf.org/archive/id/draft-ietf-6lo-plc-10.txt>>.
- [I-D.ietf-intarea-gue] Herbert, T., Yong, L., and O. Zia, "Generic UDP Encapsulation", Work in Progress, Internet-Draft, draft-ietf-intarea-gue-09, 26 October 2019, <<https://www.ietf.org/archive/id/draft-ietf-intarea-gue-09.txt>>.
- [I-D.ietf-intarea-tunnels] Touch, J. and M. Townsley, "IP Tunnels in the Internet Architecture", Work in Progress, Internet-Draft, draft-ietf-intarea-tunnels-10, 12 September 2019, <<https://www.ietf.org/archive/id/draft-ietf-intarea-tunnels-10.txt>>.
- [I-D.ietf-lisp-introduction] Cabellos, A. and D. S. (Ed.), "An Architectural Introduction to the Locator/ID Separation Protocol (LISP)", Work in Progress, Internet-Draft, draft-ietf-lisp-introduction-15, 20 September 2021, <<https://www.ietf.org/archive/id/draft-ietf-lisp-introduction-15.txt>>.



[I-D.ietf-lisp-mn]

Farinacci, D., Lewis, D., Meyer, D., and C. White, "LISP Mobile Node", Work in Progress, Internet-Draft, draft-ietf-lisp-mn-11, 30 January 2022, <<https://www.ietf.org/archive/id/draft-ietf-lisp-mn-11.txt>>.

[I-D.ietf-lisp-rfc6830bis]

Farinacci, D., Fuller, V., Meyer, D., Lewis, D., and A. Cabellos, "The Locator/ID Separation Protocol (LISP)", Work in Progress, Internet-Draft, draft-ietf-lisp-rfc6830bis-36, 18 November 2020, <<https://www.ietf.org/archive/id/draft-ietf-lisp-rfc6830bis-36.txt>>.

[I-D.ietf-lisp-rfc6833bis]

Farinacci, D., Maino, F., Fuller, V., and A. Cabellos, "Locator/ID Separation Protocol (LISP) Control-Plane", Work in Progress, Internet-Draft, draft-ietf-lisp-rfc6833bis-30, 18 November 2020, <<https://www.ietf.org/archive/id/draft-ietf-lisp-rfc6833bis-30.txt>>.

[I-D.jia-intarea-scenarios-problems-addressing]

Jia, Y., Trossen, D., Iannone, L., Shenoy, N., Mendes, P., 3rd, D. E. E., and P. Liu, "Challenging Scenarios and Problems in Internet Addressing", Work in Progress, Internet-Draft, draft-jia-intarea-scenarios-problems-addressing-02, 23 October 2021, <<https://www.ietf.org/archive/id/draft-jia-intarea-scenarios-problems-addressing-02.txt>>.

[I-D.rafiiee-6man-cga-attack]

Rafiiee, H. and C. Meinel, "Possible Attack on Cryptographically Generated Addresses (CGA)", Work in Progress, Internet-Draft, draft-rafiiee-6man-cga-attack-03, 8 May 2015, <<https://www.ietf.org/archive/id/draft-rafiiee-6man-cga-attack-03.txt>>.

[ICNIP]

Trossen, D., Robitzsch, S., Reed, M., Al-Naday, M., and J. Riihijarvi, "Internet Services over ICN in 5G LAN Environments", Work in Progress, Internet-Draft, draft-trossen-icnrg-internet-icn-5glan-04, 1 October 2020, <<https://www.ietf.org/archive/id/draft-trossen-icnrg-internet-icn-5glan-04.txt>>.

- [IPv4pool] "IANA IPv4 Address Space Registry", n.d.,  
<<https://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml>>.
- [ITU9959] Badenhop, C., Fuller, J., Hall, J., Ramsey, B., and M. Rice, "Evaluating ITU-T G.9959 Based Wireless Systems Used in Critical Infrastructure Assets", IFIP Advances in Information and Communication Technology pp. 209-227, DOI 10.1007/978-3-319-26567-4\_13, 2015,  
<[https://doi.org/10.1007/978-3-319-26567-4\\_13](https://doi.org/10.1007/978-3-319-26567-4_13)>.
- [ODoH] Kinnear, E., McManus, P., Pauly, T., Verma, T., and C. A. Wood, "Oblivious DNS Over HTTPS", Work in Progress, Internet-Draft, draft-pauly-dprive-oblivious-doh-11, 17 February 2022, <<https://www.ietf.org/archive/id/draft-pauly-dprive-oblivious-doh-11.txt>>.
- [OHTTP] Thomson, M. and C. A. Wood, "Oblivious HTTP", Work in Progress, Internet-Draft, draft-thomson-http-oblivious-02, 24 August 2021, <<https://www.ietf.org/archive/id/draft-thomson-http-oblivious-02.txt>>.
- [ONION] Goldschlag, D., Reed, M., and P. Syverson, "Onion routing", Communications of the ACM Vol. 42, pp. 39-41, DOI 10.1145/293411.293443, February 1999,  
<<https://doi.org/10.1145/293411.293443>>.
- [P4] Bosshart, P., Daly, D., Gibb, G., Izzard, M., McKeown, N., Rexford, J., Schlesinger, C., Talayco, D., Vahdat, A., Varghese, G., and D. Walker, "P4: programming protocol-independent packet processors", ACM SIGCOMM Computer Communication Review Vol. 44, pp. 87-95, DOI 10.1145/2656877.2656890, July 2014,  
<<https://doi.org/10.1145/2656877.2656890>>.
- [REED] Reed, M., Al-Naday, M., Thomos, N., Trossen, D., Petropoulos, G., and S. Spirou, "Stateless multicast switching in software defined networks", 2016 IEEE International Conference on Communications (ICC), DOI 10.1109/icc.2016.7511036, May 2016,  
<<https://doi.org/10.1109/icc.2016.7511036>>.
- [RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981,  
<<https://www.rfc-editor.org/info/rfc791>>.

- [RFC1752] Bradner, S. and A. Mankin, "The Recommendation for the IP Next Generation Protocol", RFC 1752, DOI 10.17487/RFC1752, January 1995, <<https://www.rfc-editor.org/info/rfc1752>>.
- [RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G. J., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <<https://www.rfc-editor.org/info/rfc1918>>.
- [RFC2009] Imielinski, T. and J. Navas, "GPS-Based Addressing and Routing", RFC 2009, DOI 10.17487/RFC2009, November 1996, <<https://www.rfc-editor.org/info/rfc2009>>.
- [RFC2101] Carpenter, B., Crowcroft, J., and Y. Rekhter, "IPv4 Address Behaviour Today", RFC 2101, DOI 10.17487/RFC2101, February 1997, <<https://www.rfc-editor.org/info/rfc2101>>.
- [RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <<https://www.rfc-editor.org/info/rfc2474>>.
- [RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, DOI 10.17487/RFC2663, August 1999, <<https://www.rfc-editor.org/info/rfc2663>>.
- [RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, DOI 10.17487/RFC2865, June 2000, <<https://www.rfc-editor.org/info/rfc2865>>.
- [RFC3118] Droms, R., Ed. and W. Arbaugh, Ed., "Authentication for DHCP Messages", RFC 3118, DOI 10.17487/RFC3118, June 2001, <<https://www.rfc-editor.org/info/rfc3118>>.
- [RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, DOI 10.17487/RFC3972, March 2005, <<https://www.rfc-editor.org/info/rfc3972>>.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <<https://www.rfc-editor.org/info/rfc3986>>.

- [RFC4014] Droms, R. and J. Schnizlein, "Remote Authentication Dial-In User Service (RADIUS) Attributes Suboption for the Dynamic Host Configuration Protocol (DHCP) Relay Agent Information Option", RFC 4014, DOI 10.17487/RFC4014, February 2005, <<https://www.rfc-editor.org/info/rfc4014>>.
- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <<https://www.rfc-editor.org/info/rfc4193>>.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <<https://www.rfc-editor.org/info/rfc4291>>.
- [RFC4581] Bagnulo, M. and J. Arkko, "Cryptographically Generated Addresses (CGA) Extension Field Format", RFC 4581, DOI 10.17487/RFC4581, October 2006, <<https://www.rfc-editor.org/info/rfc4581>>.
- [RFC4982] Bagnulo, M. and J. Arkko, "Support for Multiple Hash Algorithms in Cryptographically Generated Addresses (CGAs)", RFC 4982, DOI 10.17487/RFC4982, July 2007, <<https://www.rfc-editor.org/info/rfc4982>>.
- [RFC5533] Nordmark, E. and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6", RFC 5533, DOI 10.17487/RFC5533, June 2009, <<https://www.rfc-editor.org/info/rfc5533>>.
- [RFC5795] Sandlund, K., Pelletier, G., and L-E. Jonsson, "The RObusT Header Compression (ROHC) Framework", RFC 5795, DOI 10.17487/RFC5795, March 2010, <<https://www.rfc-editor.org/info/rfc5795>>.
- [RFC6250] Thaler, D., "Evolution of the IP Model", RFC 6250, DOI 10.17487/RFC6250, May 2011, <<https://www.rfc-editor.org/info/rfc6250>>.
- [RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <<https://www.rfc-editor.org/info/rfc6275>>.
- [RFC6282] Hui, J., Ed. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, DOI 10.17487/RFC6282, September 2011, <<https://www.rfc-editor.org/info/rfc6282>>.

- [RFC6740] Atkinson, R.J. and SN. Bhatti, "Identifier-Locator Network Protocol (ILNP) Architectural Description", RFC 6740, DOI 10.17487/RFC6740, November 2012, <<https://www.rfc-editor.org/info/rfc6740>>.
- [RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <<https://www.rfc-editor.org/info/rfc7228>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<https://www.rfc-editor.org/info/rfc7258>>.
- [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>>.
- [RFC7400] Bormann, C., "6LoWPAN-GHC: Generic Header Compression for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 7400, DOI 10.17487/RFC7400, November 2014, <<https://www.rfc-editor.org/info/rfc7400>>.
- [RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <<https://www.rfc-editor.org/info/rfc7401>>.
- [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <<https://www.rfc-editor.org/info/rfc7665>>.
- [RFC7872] Gont, F., Linkova, J., Chown, T., and W. Liu, "Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World", RFC 7872, DOI 10.17487/RFC7872, June 2016, <<https://www.rfc-editor.org/info/rfc7872>>.
- [RFC8060] Farinacci, D., Meyer, D., and J. Snijders, "LISP Canonical Address Format (LCAF)", RFC 8060, DOI 10.17487/RFC8060, February 2017, <<https://www.rfc-editor.org/info/rfc8060>>.

- [RFC8061] Farinacci, D. and B. Weis, "Locator/ID Separation Protocol (LISP) Data-Plane Confidentiality", RFC 8061, DOI 10.17487/RFC8061, February 2017, <<https://www.rfc-editor.org/info/rfc8061>>.
- [RFC8105] Mariager, P., Petersen, J., Ed., Shelby, Z., Van de Logt, M., and D. Barthel, "Transmission of IPv6 Packets over Digital Enhanced Cordless Telecommunications (DECT) Ultra Low Energy (ULE)", RFC 8105, DOI 10.17487/RFC8105, May 2017, <<https://www.rfc-editor.org/info/rfc8105>>.
- [RFC8163] Lynn, K., Ed., Martocci, J., Neilson, C., and S. Donaldson, "Transmission of IPv6 over Master-Slave/Token-Passing (MS/TP) Networks", RFC 8163, DOI 10.17487/RFC8163, May 2017, <<https://www.rfc-editor.org/info/rfc8163>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC8280] ten Oever, N. and C. Cath, "Research into Human Rights Protocol Considerations", RFC 8280, DOI 10.17487/RFC8280, October 2017, <<https://www.rfc-editor.org/info/rfc8280>>.
- [RFC8376] Farrell, S., Ed., "Low-Power Wide Area Network (LPWAN) Overview", RFC 8376, DOI 10.17487/RFC8376, May 2018, <<https://www.rfc-editor.org/info/rfc8376>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <<https://www.rfc-editor.org/info/rfc8484>>.
- [RFC8609] Mosko, M., Solis, I., and C. Wood, "Content-Centric Networking (CCNx) Messages in TLV Format", RFC 8609, DOI 10.17487/RFC8609, July 2019, <<https://www.rfc-editor.org/info/rfc8609>>.
- [RFC8724] Minaburo, A., Toutain, L., Gomez, C., Barthel, D., and JC. Zuniga, "SCHC: Generic Framework for Static Context Header Compression and Fragmentation", RFC 8724, DOI 10.17487/RFC8724, April 2020, <<https://www.rfc-editor.org/info/rfc8724>>.

- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.
- [RFC8926] Gross, J., Ed., Ganga, I., Ed., and T. Sridhar, Ed., "Geneve: Generic Network Virtualization Encapsulation", RFC 8926, DOI 10.17487/RFC8926, November 2020, <<https://www.rfc-editor.org/info/rfc8926>>.
- [RFC8939] Varga, B., Ed., Farkas, J., Berger, L., Fedyk, D., and S. Bryant, "Deterministic Networking (DetNet) Data Plane: IP", RFC 8939, DOI 10.17487/RFC8939, November 2020, <<https://www.rfc-editor.org/info/rfc8939>>.
- [RFC8981] Gont, F., Krishnan, S., Narten, T., and R. Draves, "Temporary Address Extensions for Stateless Address Autoconfiguration in IPv6", RFC 8981, DOI 10.17487/RFC8981, February 2021, <<https://www.rfc-editor.org/info/rfc8981>>.
- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.
- [ROBUSTSDN] Canini, M., Kuznetsov, P., Levin, D., and S. Schmid, "A distributed and robust SDN control plane for transactional network updates", 2015 IEEE Conference on Computer Communications (INFOCOM), DOI 10.1109/infocom.2015.7218382, April 2015, <<https://doi.org/10.1109/infocom.2015.7218382>>.
- [ROHC] Fitzek, F., Rein, S., Seeling, P., and M. Reisslein, "RObust Header Compression (ROHC) Performance for Multimedia Transmission over 3G/4G Wireless Networks", Wireless Personal Communications Vol. 32, pp. 23-41, DOI 10.1007/s11277-005-7733-2, January 2005, <<https://doi.org/10.1007/s11277-005-7733-2>>.
- [SFCANYCAST] Wion, A., Bouet, M., Iannone, L., and V. Conan, "Distributed Function Chaining with Anycast Routing", Proceedings of the 2019 ACM Symposium on SDN Research, DOI 10.1145/3314148.3314355, April 2019, <<https://doi.org/10.1145/3314148.3314355>>.

- [SPHINX] Danezis, G. and I. Goldberg, "Sphinx: A Compact and Provably Secure Mix Format", 2009 30th IEEE Symposium on Security and Privacy, DOI 10.1109/sp.2009.15, May 2009, <<https://doi.org/10.1109/sp.2009.15>>.
- [TOR] "The Tor Project", n.d., <<https://www.torproject.org/>>.
- [TRANSACTIONSDN] Curic, M., Despotovic, Z., Hecker, A., and G. Carle, "Transactional Network Updates in SDN", 2018 European Conference on Networks and Communications (EuCNC), DOI 10.1109/eucnc.2018.8442793, June 2018, <<https://doi.org/10.1109/eucnc.2018.8442793>>.
- [UA-DHCP] Komori, T. and T. Saito, "The secure DHCP system with user authentication", 27th Annual IEEE Conference on Local Computer Networks, 2002. Proceedings. LCN 2002., DOI 10.1109/lcn.2002.1181774, n.d., <<https://doi.org/10.1109/lcn.2002.1181774>>.
- [VPN] Khanvilkar, S. and A. Khokhar, "Virtual private networks: an overview with performance evaluation", IEEE Communications Magazine Vol. 42, pp. 146-154, DOI 10.1109/mcom.2004.1341273, October 2004, <<https://doi.org/10.1109/mcom.2004.1341273>>.
- [WireGuard] Donenfeld, J., "WireGuard: Next Generation Kernel Network Tunnel", Proceedings 2017 Network and Distributed System Security Symposium, DOI 10.14722/ndss.2017.23160, 2017, <<https://doi.org/10.14722/ndss.2017.23160>>.

#### Acknowledgments

Thanks to all the people that shared insightful comments both privately to the authors as well as on various mailing list, especially on the INTArea Mailing List. Also thanks for the interesting discussions to Carsten Borman, Brian E. Carpenter.

#### Authors' Addresses

Yihao Jia  
Huawei Technologies Co., Ltd  
156 Beiqing Rd.  
Beijing  
100095  
P.R. China  
Email: [jiayihao@huawei.com](mailto:jiayihao@huawei.com)



Dirk Trossen  
Huawei Technologies Duesseldorf GmbH  
Riesstr. 25C  
80992 Munich  
Germany  
Email: dirk.trossen@huawei.com

Luigi Iannone  
Huawei Technologies France S.A.S.U.  
18, Quai du Point du Jour  
92100 Boulogne-Billancourt  
France  
Email: luigi.iannone@huawei.com

Paulo Mendes  
Airbus  
Willy-Messerschmitt Strasse 1  
81663 Munich  
Germany  
Email: paulo.mendes@airbus.com

Nirmala Shenoy  
Rochester Institute of Technology  
New-York, 14623  
United States of America  
Email: nxsvks@rit.edu

Laurent Toutain  
IMT-Atlantique  
2 rue de la Chataigneraie  
CS 17607  
35576 Cesson-Sevigne Cedex  
France  
Email: laurent.toutain@imt-atlantique.fr

Abraham Y. Chen  
Avinta Communications, Inc.  
142 N. Milpitas Blvd.  
Milpitas, CA, 95035-4401  
United States of America  
Email: AYChen@Avinta.com

Dino Farinacci  
lispers.net  
United States of America  
Email: farinacci@gmail.com

Internet Area Working Group  
Internet-Draft  
Intended status: Informational  
Expires: 7 September 2022

Y. Jia  
D. Trossen  
L. Iannone  
Huawei  
N. Shenoy  
R.I.T.  
P. Mendes  
Airbus  
D. Eastlake 3rd  
Futurewei  
P. Liu  
China Mobile  
D. Farinacci  
lispers.net  
6 March 2022

Challenging Scenarios and Problems in Internet Addressing  
draft-jia-intarea-scenarios-problems-addressing-03

Abstract

The Internet Protocol (IP) has been the major technological success in information technology of the last half century. As the Internet becomes pervasive, IP has been replacing communication technology for many domain-specific solutions. However, domains with specific requirements as well as communication behaviors and semantics still exist and represent what [RFC8799] recognizes as "limited domains".

This document describes well-recognized scenarios that showcase possibly different addressing requirements, which are challenging to be accommodated in the IP addressing model. These scenarios highlight issues related to the Internet addressing model and call for starting a discussion on a way to re-think/evolve the addressing model so to better accommodate different domain-specific requirements.

The issues identified in this document are complemented and deepened by a detailed gap analysis in a separate companion document [I-D.jia-intarea-internet-addressing-gap-analysis].

Status of This Memo

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

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

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

This Internet-Draft will expire on 7 September 2022.

#### Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

#### Table of Contents

1. Introduction . . . . .	3
2. Communication Scenarios in Limited Domains . . . . .	5
2.1. Communication in Constrained Environments . . . . .	5
2.2. Communication within Dynamically Changing Topologies . . . . .	7
2.3. Communication among Moving Endpoints . . . . .	10
2.4. Communication Across Services . . . . .	13
2.5. Communication Traffic Steering . . . . .	14
2.6. Communication with built-in security . . . . .	15
2.7. Communication protecting user privacy . . . . .	16
2.8. Communication in Alternative Forwarding Architectures . . . . .	16
3. Desired Network Features . . . . .	18
4. Issues in Addressing . . . . .	21
5. Problem Statement . . . . .	23
6. Security Considerations . . . . .	25
7. IANA Considerations . . . . .	25
8. References . . . . .	25
8.1. Normative References . . . . .	25
8.2. Informative References . . . . .	25
Acknowledgments . . . . .	33
Authors' Addresses . . . . .	33

## 1. Introduction

The Internet Protocol (IP), positioned as the unified protocol at the (Internet) network layer, is seen by many as key to the innovation stemming from Internet-based applications and services. Even more so, with the success of TCP/IP protocol stack, IP has been gradually replacing existing domain-specific protocols, evolving into the core protocol of the entire communication eco-system. At its inception, roughly 40 years ago [RFC0791], the Internet addressing system, represented in the form of the IP address and its locator-based (topological) semantics, has brought the notion of a 'common namespace for all communication'. Compared to proprietary technology-specific solutions, such 'common namespace for all communication' advance ensures end-to-end communication from any device connected to the Internet to another.

However, use cases, associated services, node behaviors, and requirements on packet delivery have since been significantly extended, with the Internet technology being developed to accommodate them in the framework of addressing that stood at the beginning of the Internet's development. This evolution is reflected in the concept of "Limited Domains", first introduced in [RFC8799]. It refers to a single physical network, attached to or running in parallel with the Internet, or is defined by a set of users and nodes distributed over a much wider area, but drawn together by a single virtual network over the Internet. Key to a limited domain is that requirements, behaviors, and semantics could be noticeable local and, more importantly, specific to the limited domain. Very often, the realization of a limited domain is defined by specific communication scenario(s) and/or use case(s) that exhibit the domain-specific behaviors and pose the requirements that lead to the establishment of the limited domain. Identifying limited domains may sometime be not obvious because of blurry boundaries depending on the point of view. For instance, from an end user perspective there is no vision at all on limited domains, hence for end users the dichotomy Internet vs limited domains more transparent. In such cases, it is harder to ensure (and detect) that no limited domain specific semantics leak in the Internet or other limited domains.

One key architectural aspect, when communicating within limited domains, is that of addressing and, therefore, the address structure, as well as the semantic that is being used for packet forwarding (e.g., service identification, content location, device type). The topological location centrality of IP is fundamental when reconciling the often differing semantics for 'addressing' that can be found in those limited domains. The result of this fundamental role of the single IP addressing is that limited domains have to adopt specific solutions, e.g., translating/mapping/converting concepts, semantics, and ultimately, domain-specific addressing, into the common IP addressing used across limited domains.

This document advocates flexibility in addressing in order to accommodate limited domain specific semantics, while, if possible, ensuring a single holistic addressing scheme able to reduce, or even entirely remove, the need for aligning the address semantics of different limited domains, such as the current topological location semantic of the Internet. Ultimately, such holistic addressing could be beneficial to those communication scenarios realized within limited domains by improving efficiency, removing of constraints imposed by needing to utilize the limited semantics of IP addressing, and/or in other ways.

In other words, this document revolves around the following question:

"Should interconnected limited domains purely rely on IP addresses and therefore deal with the complexity of translating any semantic mismatch themselves, or should flexibility for supporting those limited domains be a key focus for an evolved Internet addressing?"

To that end, this document describes well-recognized scenarios in limited domains that could benefit from greater flexibility in addressing and overviews the problems encountered throughout these scenarios due to the lack of that flexibility. A detailed gap analysis can be found in {I-D.jia-intarea-internet-addressing-gap-analysis}}, which elaborates on the issues identified in this memo in reference to extensions to Internet addressing that have attempted to address those issues. The purpose of this memo is rather to stimulate discussion on the emerging needs for addressing at large with the possibility to fundamentally re-think the addressing in the Internet beyond the current objectives of IPv6 [RFC8200].

It is important to remark that any change in the addressing, hence at the data plane level, leads to changes and challenges at the control plane level, i.e., routing. The latter is an even harder problem than just addressing and might need more research efforts that are beyond the objective of this document, which focuses solely on the data plane.

## 2. Communication Scenarios in Limited Domains

The following sub-sections outline a number of scenarios, all of which belong to the concept of "limited domains" [RFC8799]. While the list of scenarios may look long, this document focuses on scenarios with a number of aspects that can be observed in those limited domains, captured in the sub-section titles. For each scenario, possible challenges are highlighted, which are then picked upon in Section 4, when describing more formally the existing shortcomings in current Internet addressing.

### 2.1. Communication in Constrained Environments

In a number of communication scenarios, such as those encountered in the Internet of Things (IoT), a simple, communication network demanding minimal resources is required, allowing for a group of IoT network devices to form a network of constrained nodes, with the participating network and end nodes requiring as little computational power as possible and having small memory requirements in order to reduce the total cost of ownership of the network. Furthermore, in the context of industrial IoT, real-time requirements and scalability make IP technology not naturally suitable as communication technology ([OCADO]).

In addition to IEEE 802.15.4, i.e., Low-Rate Wireless Personal Area Network [LR-WPAN], several limited domains exist through utilizing link layer technologies such as Bluetooth Low Energy (BLE) [BLE], Digital European Cordless Telecommunications (DECT) - Ultra Low Energy (ULE) [DECT-ULE], Master-Slave/Token-Passing (MS/TP) [BACnet], Near-Field-Communication (NFC) [ECMA-340], and Power Line Communication (PLC) [IEEE\_1901.1].

The end-to-end principle (detailed in [RFC2775]) requires IP addresses (e.g., IPv6 [RFC8200]) to be used on such constrained nodes networks, allowing IoT devices using multiple communication technologies to talk on the Internet. Often, devices located at the edge of constrained networks act as gateway devices, usually performing header compression ([RFC4919]). To ensure security and reliability, multiple gateways must be deployed. IoT devices on the network must select one of those gateways for traffic passthrough by the devices on the (limited domain) network.

Given the constraints imposed on the computational and possibly also communication technology, the usage of a single addressing semantic in the form of a 128-bit endpoint identifier, i.e., IPv6 address, may pose a challenge when operating such networks.

Another type of (differently) constrained environment is an aircraft, which encompasses not only passenger communication but also the integration of real-time data exchange to ensure that processes and functions in the cabin are automatically monitored or actuated. The goal for any aircraft network is to be able to send and receive information reliably and seamlessly. From this perspective, the medium with which these packets of information are sent is of little consequence so long as there is a level of determinism to it. However, there is currently no effective method in implementing wireless inter- and intra-communications between all subsystems. The emerging wireless sensor network technology in commercial applications such as smart thermostat systems, and smart washer/dryer units could be transposed onto aircraft and fleet operations. The proposal for having an Wireless Avionics Intra-Communications (WAIC) system promises reduction in the complexity of electrical wiring harness design and fabrication, reduction in wiring weight, increased configuration, and potential monitoring of otherwise inaccessible moving or rotating aircraft parts. Similar to the IoT concept, WAIC systems consist of short-range communications and are a potential candidate for passenger entertainment systems, smoke detectors, engine health monitors, tire pressure monitoring systems, and other kinds of aircraft maintenance systems.

While there are still many obstacles in terms of network security, traffic control, and technical challenges, future WAIC can enable real-time seamless communications between aircraft and between ground teams and aircraft as opposed to the discrete points of data leveraged today in aircraft communications. For that, WAIC infrastructure should also be connected to outside IP based networks in order to access edge/cloud facilities for data storage and mining. However, the restricted capacity (energy, communication) of most aircraft devices (e.g. sensors) and the nature of the transmitted data - periodic transmission of small packets - may pose some challenges for the usage of a single addressing semantic in the form of a 128-bit endpoint identifier, i.e., an IPv6 address. Moreover, most of the aircraft applications and services are focused on the data (e.g. temperature of gas tank on left wing) and not on the topological location of the data source. This means that the current topological location semantic of IP addresses is not beneficial for aircraft applications and services.



Greater flexibility in Internet addressing may avoid complex and energy hungry operations, like header compression and fragmentation, necessary to translate protocol headers from one limited domain to another, while enabling semantics different from locator-based addressing may better support the communication that occurs in those environments.

## 2.2. Communication within Dynamically Changing Topologies

Communication may occur over networks that exhibit dynamically changing topologies. One such example is that of satellite networks, providing global Internet connections through a combination of inter-satellite and ground station communication. With the convergence of space-based and terrestrial networks, users can experience seamless broadband access, e.g., on cruise ships, flights, and within cars, often complemented by and seamlessly switching between Wi-Fi, cellular, or satellite based networks at any time [WANG19].

The satellite network service provider will plan the transmission path of user traffic based on the network coverage, satellite orbit, route, and link load, providing potentially high-quality Internet connections for users in areas that are not, or hard to be, covered by terrestrial networks. With large scale LEO (Low Earth Orbit) satellites, the involved topologies of the satellite network will be changing constantly while observing a regular flight pattern in relation to other satellites and predictable overflight patterns to ground users [CHEN21].

Although satellite bearer services are capable of transporting IPv4 and IPv6, as well as associated protocols such as IP Multicast, DNS services and routing information, no IP functionality is implemented on-board the spacecraft limiting the capability of leveraging for instance large scale satellite constellations.

One of the major constraints of deploying routing capability on board of a satellite is power consumption. Due to this, space routers may end up being intermittently powered up during a daytime sunlit pass. Another limitation of the first generation of IP routers in space was the lack of capability to remotely manage and upgrade software while in operation.

The limitations faced in early development of IP based satellite communication payloads, showed the need to develop a flexible networking solution that would enable delay tolerant communications in the presence of intermittent connectivity. Further, in order to reduce latency, which is the major impairment of satellite networks, there was a need of a networking solution able to perform in a scenario encompassing mobile devices with the capability of storing data, leading to a significant reduction of latency, which is the major impairment of satellite networks.

Moreover, due to the current IP addressing scheme and its focus on IP unicast addressing with extended deployment of IP multicast and some IP anycast, current deployments do not take advantage of the broadcast nature of satellite networks.

Moreover networking platforms based on a name (data or service) based addressing scheme would bring several potential benefits to satellite networks aiming to tackle their major challenges, including high propagation delay and changing network topology in the case of LEO constellations.

Another example is that of vehicular communication, where services may be accessed across vehicles, such as self-driving cars, for the purpose of collaborative objection recognition (e.g., for collision avoidance), road status conveyance (e.g., for pre-warning of road-ahead conditions), and other purposes. Communication may include Road Side Units (RSU) with the possibility to create ephemeral connections to those RSUs for the purpose of workload offloading, joint computation over multiple (vehicular) inputs, and other purposes [I-D.ietf-lisp-nexagon]. Communication here may exhibit a multi-hop nature, not just involving the vehicle and the RSU over a direct link. Those topologies are naturally changing constantly due to the dynamic nature of the involved communication nodes.

The advent of Flying Ad-hoc NETworks (FANETs) has opened up an opportunity to create new added-value services [CHRIKI19]. Although these networks share common features with vehicular ad hoc networks, they present several unique characteristics such as energy efficiency, mobility degree, the capability of swarming, and the potential large scale of swarm networks. Due to high mobility of FANET nodes, the network topology changes more frequently than in a typical vehicular ad hoc network. From a routing point of view, although ad-hoc reactive and proactive routing approaches can be used, there are other type of routing protocols that have been developed for FANETS, such as hybrid routing protocols and position based routing protocols, aiming to increase efficiency in large scale networks with dynamic topologies.

Both type of protocols challenge the current Internet addressing semantic: in the case of hybrid protocols, two different routing strategies are used inside and outside a network zone. While inside a zone packets are routed to a specific destination IP address, between zones, query packets are routed to a subset of neighbors as determined by a broadcast algorithm. In the case of position based routing protocol, the IP addressing scheme is not used at all, since packets are routed to a different identifier, corresponding to the geographic location of the destination and not its topological location. Hence, what is needed is to consolidate the geo-spatial addressing with that of a locator-based addressing in order to optimize routing policies across the zones.

Moreover most of the application/services deployed in FANETs tend to be agnostic of the topological location of nodes, rather focusing on the location of data or services. This distinction is even more important because in dynamic network such as FANET robust networking solutions may rely on the redundancy of data and services, meaning that they may be found in more than one device in the network. This in turn may bring into play a possible service-centric semantic for addressing the packets that need routing in the dynamic network towards a node providing said service (or content).

In the aforementioned network technologies, there is a significant difference between the high dynamics of the underlying network topologies, compared to the relative static nature of terrestrial network topology, as reported in [HANDLEY]. As a consequence, the notion of a topological network location becomes restrictive in the sense that not only the relation between network nodes and user endpoint may change, but also the relation between the nodes that form the network itself. This may lead to the challenge of maintaining and updating the topological addresses in this constantly changing network topology.

In attempts to utilize entirely different semantics for the addressing itself, geographic-based routing, such as in [CARTISEAN], has been proposed for MANETs (Mobile Ad-hoc NETWORKs) through providing geographic coordinates based addresses to achieve better routing performance, lower overhead, and lower latency [MANET1].

Flexibility in Internet addressing here would allow for accommodating such geographic address semantics into the overall Internet addressing, while also enabling name/content-based addressing, utilizing the redundancy of many network locations providing the possible data.

### 2.3. Communication among Moving Endpoints

When packet switching was first introduced, back in the 60s/70s, it was intended to replace the rigid circuit switching with a communication infrastructure that was more resilient to failures. As such, the design never really considered communication endpoints as mobile. Even in the pioneering ALOHA [ALOHA] system, despite considering wireless and satellite links, the network was considered static (with the exception of failures and satellites, which fall in what is discussed in Section 2.2). Ever since, a lot of efforts have been devoted to overcome such limitations once it became clear that endpoint mobility will become a main (if not THE main) characteristic of ubiquitous communication systems.

The IETF has for a long time worked on solutions that would allow extending the IP layer with mobility support. Because of the topological semantic of IP addresses, endpoints need to change addresses each time they visit a different network. However, because routing and endpoint identification is also IP address based, this leads to a communication disruption.

To cope with such a situation, sometimes, the transport layer gets involved in mobility solutions, either by introducing explicit in-band signaling to allow for communicating IP address changes (e.g., in SCTP [RFC5061] and MPTCP [RFC6182]), or by introducing some form of connection ID that allows for identifying a communication independently from IP addresses (e.g., the connection ID used in QUIC [RFC9000]).

Concerning network layer only solutions, anchor-based Mobile IP mechanisms have been introduced ([RFC5177], [RFC6626] [RFC5944], [RFC5275]). Mobile IP is based on a relatively complex and heavy mechanism that makes it hard to deploy and it is not very efficient. Furthermore, it is even less suitable than native IP in constrained environments like the ones discussed in Section 2.1.

Alternative approaches to Mobile IP often leverage the introduction of some form of overlay. LISP [I-D.ietf-lisp-introduction], by separating the topological semantic from the identification semantic of IP addresses, is able to cope with endpoint mobility by dynamically mapping endpoint identifiers with routing locators [I-D.ietf-lisp-mn]. This comes at the price of an overlay that needs its own additional control plane [I-D.ietf-lisp-rfc6833bis].

Similarly, the NV03 (Network Virtualization Overlays) Working Group, while focusing on Data Center environments, also explored an overlay-based solution for multi-tenancy purposes, but also resilient to mobility since relocating Virtual Machines (VMs) is common practice.

NVO3 considered for a long time several data planes that implement slightly different flavors of overlays ([RFC8926], [RFC7348], [I-D.ietf-intarea-gue]), but lacks an efficient control plane specifically tailored for DCs.

Alternative mobility architectures have also been proposed in order to cope with endpoint mobility outside the IP layer itself. The Host Identity Protocol (HIP) [RFC7401] introduced a new namespace in order to identify endpoints, namely the Host Identity (HI), while leveraging the IP layer for topological location. On the one hand, such an approach needs to revise the way applications interact with the network layer, by modifying the DNS (now returning an HI instead of an IP address) and applications to use the HIP socket extension. On the other hand, early adopters do not necessarily gain any benefit unless all communicating endpoints upgrade to use HIP. In spite of this, such a solution may work in the context of a limited domain.

Another alternative approach is adopted by Information-Centric Networking (ICN) [RFC7476]. By making content a first class citizen of the communication architecture, the "what" rather than the "where" becomes the real focus of the communication. However, as explained in the next sub-section, ICN can run either over the IP layer or completely replace it, which in turn can be seen as running the Internet and ICN as logically completely separated limited domains.

Unmanned Aircraft Systems (UAS) are examples of moving devices that require a stable mobility management scheme since they consist of a number of Unmanned Aerial Vehicles (UAV) subordinated to a Ground Control Station (GCS) [MAROJEVIC20]. The information produced by the different sensors and electronic devices available at each UAV is collected and processed by a software or hardware data acquisition unit, being transmitted towards the GCS, where it is inspected and/or analyzed. Analogously, control information transmitted from the GCS to the UAV enables the execution of control operations over the aircraft, such as changing the route planning or the direction pointed by a camera.

Although UAVs may have redundant links to maintain communications in long-range missions (e.g., satellite), most of the communications between the GCS and the UAVs take place over wireless data links, e.g., based on a radio line-of-sight technology, Wi-Fi or 3G/4G/5G. While in some scenarios, UAVs will operate always under the range of the same cellular base station, in missions with large range, UAVs will move between different cellular or wireless ground infrastructure, meaning that the UAV needs to upload its topological locator and re-start the ongoing communication sessions. In such cases, most of existing Mobile IP approaches may play a role, as well as approaches to split the UAV identifier and the topological locator, such as HIP.

However, while the industry is given the first steps towards evolved UAS architectures and communication models, the data-centric communication plays an increasing role, where information is named and decoupled from its location, and applications/services operate over these named data rather than on host-to-host communications.

In this context, the Data Distribution Service ([DDS]) has emerged as an industry-oriented open standard that follows this approach. The space and time decoupling allowed by DDS is very relevant in any dynamic and distributed system, since interacting entities are not forced to know each other and are not forced to be simultaneously present to exchange data. Time decoupling can significantly simplify the management of intermittent data-links, in particular for wireless connectivity between UAS, as well as facilitate seamless UAV mobility between GCSs. This model of communication, in turn, questions the locator-based addressing used in IP and instead utilizes a data-centric naming.

In the case of using TCP/IP, mobility of UAVs introduces a significant challenge. Consider the case where a GCS is receiving telemetry information from a specific UAV. Assuming that the UAV moves and changes its point of attachment to the network, it will have to configure a new IP address on its wireless interface. However, this is problematic, as the telemetry information is still being sent by to the previous IP address of the UAV. This simple example illustrates the necessity to deploy mobility management solutions to handle this type of situations.

However, mobility management solutions increase the complexity of the deployment and may impact the performance of data distribution, both in terms of signaling/data overhead and communication path delay. Considering the specific case of multicast data streams, mobility of content producers and consumers is inherently handled by multicast routing protocols, which are able to react to changes of location of mobile nodes by reconstructing the corresponding multicast delivery

trees. Nevertheless, this comes with a cost in terms of signaling and data overhead (data may still flow through branches of a multicast delivery tree where there are no receivers while the routing protocol is still converging).

Another alternative is to perform the mobility management of producers and consumers not at the application layer based on IP multicast trees, but on the network layer based on an Information Centric Network approach, which was already mentioned in this section.

Greater flexibility in addressing may help in dealing with mobility more efficiently, e.g., through an augmented semantic that may fulfil the mobility requirements [RFC7429] in a more efficient way or through moving from a locator- to a content or service-centric semantic for addressing.

#### 2.4. Communication Across Services

As a communication infrastructure spanning many facets of life, the Internet integrates services and resources from various aspects such as remote collaboration, shopping, content production as well as delivery, education, and many more. Accessing those services and resources directly through URIs has been proposed by methods such as those defined in ICN [RFC7476], where providers of services and resources can advertise those through unified identifiers without additional planning of identifiers and locations for underlying data and their replicas. Users can access required services and resources by virtue of using the URI-based identification, with an ephemeral relationship built between user and provider, while the building of such relationship may be constrained with user- as well as service-specific requirements, such as proximity (finding nearest provider), load (finding fastest provider), and others.

While systems like ICN [CCN] provide an alternative to the topological addressing of IP, its deployment requires an overlay (over IP) or native deployment (alongside IP), the latter with dedicated gateways needed for translation. Underlay deployments are also envisioned in [RFC8763], where ICN solutions are being used to facilitate communication between IP addressed network endpoints or URI-based service endpoints, still requiring gateway solutions for interconnection with ICN-based networks as well as IP routing based networks (cf., [ICN5G][ICNIP]).

Although various approaches combining service and location-based addressing have been devised, the key challenge here is to facilitate a "natural", i.e., direct communication, without the need for gateways above the network layer.

Another aspect of communication across services is that of chaining individual services to a larger service. Here, an identifier would be used that serves as a link to next hop destination within the chain of single services, as done in the work on Service Function Chaining (SFC). With this, services are identified at the level of Layer 2/3 ([RFC7665], [RFC8754], [RFC8595]) or at the level of name-based service identifiers like URLs [RFC8677] although the service chain identification is carried as a Network Service header (NSH) [RFC7665], separate to the packet identifiers. The forwarding with the chain of services utilizes individual locator-based IP addressing (for L3 chaining) to communicate the chained operations from one Service Function Forwarder [RFC7665] to another, leading to concerns regarding overhead incurred through the stacking of those chained identifiers in terms of packet overhead and therefore efficiency in handling in the intermediary nodes.

Greater flexibility in addressing may allow for incorporating different information, e.g., service as well as chaining semantics, into the overall Internet addressing.

## 2.5. Communication Traffic Steering

Steering traffic within a communication scenario may involve at least two aspects, namely (i) limiting certain traffic towards a certain set of communication nodes and (ii) restraining the sending of packets towards a given destination (or a chain of destinations) with metrics that would allow the selection among one or more possible destinations.

One possibility for limiting traffic inside limited domains, towards specific objects, e.g., devices, users, or group of them, is subnet partition with techniques such as VLAN [RFC5517], VxLAN [RFC7348], or more evolved solution like TeraStream [TERASTREAM] realizing such partitioning. Such mechanisms usually involve significant configuration, and even small changes in network and user nodes could result in a repartition and possibly additional configuration efforts. Another key aspect is the complete lack of correlation of the topological address and any likely more semantic-rich identification that could be used to make policy decisions regarding traffic steering. Suitably enriching the semantics of the packet address, either that of the sender or receiver, so that such decision could be made while minimizing the involvement of higher layer mechanisms, is a crucial challenge for improving on network operations and speed of such limited domain traffic.

When making decisions to select one out of a set of possible destinations for a packet, IP anycast semantics can be applied albeit being limited to the locator semantic of the IP address itself.



Recent work in [SFCANYCAST] suggests utilizing the notion of IP anycast address to encode a "service identifier", which is dynamically mapped onto network locations where service instances fulfilling the service request may be located. Scenarios where this capability may be utilized are provided in [SFCANYCAST] and include, but are not limited to, scenarios such as edge-assisted VR/AR, transportation, smart cities, smart homes, smart wearables, and digital twins.

The challenge here lies in the possible encoding of not only the service information itself but the constraint information that helps the selection of the "best" service instance and which is likely a service-specific constraint in relation to the particular scenario. The notion of an address here is a conditional (on those constraints) one where this conditional part is an essential aspect of the forwarding action to be taken. It needs therefore consideration in the definition of what an address is, what is its semantic, and how the address structure ought to look like.

As outlined in the previous sub-section, chaining services are another aspect of steering traffic along a chain of constituent services, where the chain is identified through either a stack of individual identifiers, such as in Segment Routing [RFC8402], or as an identifier that serves as a link to next hop destination within the chain, such as in Service Function Chaining (SFC). The latter can be applied to services identified at the level of Layer 2/3 ([RFC7665], [RFC8754], [RFC8595]) or at the level of name-based service identifiers like URLs [RFC8677]. However, the overhead incurred through the stacking of those chained identifiers is a concern in terms of packet overhead and therefore efficiency in handling in the intermediary nodes.

Flexibility in addressing may enable more semantic rich encoding schemes that may help in steering traffic at hardware level and speed, without complex mechanisms usually resulting in handling packets in the slow path of routers.

## 2.6. Communication with built-in security

Today, strong security in the Internet is usually implemented as a general network service ([PILA], [RFC6158]). Among the various reasons for such approach is the limited semantic of current IP addresses, which do not allow to natively express security features or trust relationships. Efforts like Cryptographically Generated Addresses (CGA) [RFC3972], provide some security features by embedding a truncated public key in the last 57-bit of IPv6 address, thereby greatly enhancing authentication and security within an IP network via asymmetric cryptography and IPsec [RFC4301]. The

development of the Host Identity Protocol (HIP) [RFC7401] saw the introduction of cryptographic identifiers for the newly introduced Host Identity (HI) to allow for enhanced accountability, and therefore trust. The use of those HIs, however, is limited by the size of IPv6 128bit addresses.

Through a greater flexibility in addressing, any security-related key, certificate, or identifier could instead be included in a suitable address structure without any information loss (i.e., as-is, without any truncation or operation as such), avoiding therefore compromises such as those in HIP. Instead, CGAs could be created using full length certificates, or being able to support larger HIP addresses in a limited domain that uses it. This could significantly help in constructing a trusted and secure communication at the network layer, leading to connections that could be considered as absolute secure (assuming the cryptography involved is secure). Even more, anti-abuse mechanisms and/or DDoS protection mechanisms like the one under discussion in PEARG ([PEARG]) Research Group may leverage a greater flexibility of the overall Internet addressing, if provided, in order to be more effective.

## 2.7. Communication protecting user privacy

See Comments in Section "Issues".

## 2.8. Communication in Alternative Forwarding Architectures

The performance of communication networks has long been a focus for optimization due to the immediate impact on cost of ownership for communication service providers. Technologies like MPLS [RFC3031] have been introduced to optimize lower layer communication, e.g., by mapping L3 traffic into aggregated labels of forwarding traffic for the purposes of, e.g., traffic engineering.

Even further, other works have emerged in recent years that have replaced the notion of packets with other concepts for the same purpose of improved traffic engineering and therefore efficiency gains. One such area is that of Software Defined Networks (SDN) [RFC7426], which has highlighted how a majority of Internet traffic is better identified by flows, rather than packets. Based on such observation, alternate forwarding architectures have been devised that are flow-based or path-based. With this approach, all data belonging to the same traffic stream is delivered over the same path, and traffic flows are identified by some connection or path identifier rather than by complete routing information, possibly enabling fast hardware based switching (e.g. [DETNET], [PANRG]).

On the one hand, such a communication model may be more suitable for real-time traffic like in the context of Deterministic Networks ([DETNET]), where indeed a lot of work has focused on how to "identify" packets belonging to the same DETNET flow in order to jointly manage the forwarding within the desired deterministic boundaries.

On the other hand, it may improve the communication efficiency in constrained wireless environments (cf., Section 2.1), by reducing the overhead, hence increasing the number of useful bits per second per Hz.

Also, the delivery of information across similar flows may be combined into a multipoint delivery of a single return flow, e.g., for scenarios of requests for a video chunk from many clients being responded to with a single (multi-destination) flow, as outlined in [BIER-MC] as an example. Another opportunity to improve communication efficiency is being pursued in ongoing IETF/IRTF work to deliver IP- or HTTP-level packets directly over path-based or flow-based transport network solutions, such as in [TROSSEN][BIER-MC][ICNIP][ICN5G] with the capability to bundle unicast forward communication streams flexibly together in return path multipoint relations. Such capability is particularly opportune in scenarios such as chunk-based video retrieval or distributed data storage. However, those solutions currently require gateways to "translate" the flow communication into the packet-level addressing semantic in the peering IP networks. Furthermore, the use of those alternative forwarding mechanisms often require the encapsulation of Internet addressing information, leading to wastage of bandwidth as well as processing resources.

Providing an alternative way of forwarding data has also been the motivation for the efforts created in the European Telecommunication Standards Institute (ETSI), which formed an Industry Specification Group (ISG) named Non-IP Networking (NIN) [ETSI-NIN]. This group sets out to develop and standardize a set of protocols leveraging an alternative forwarding architecture, such as provided by a flow-based switching paradigm. The deployment of such protocols may be seen to form limited domains, still leaving the need to interoperate with the (packet-based forwarding) Internet; a situation possibly enabled through a greater flexibility of the addressing used across Internet-based and alternative limited domains alike.

As an alternative to IP routing, EIBP (Extended Internet Bypass Protocol) [EIBP] offers a communications model that can work with IP in parallel and entirely transparent and independent to any operation at network layer. For this, EIBP proposes the use of physical and/or virtual structures in networks and among networks to auto assign

routable addresses that capture the relative position of routers in a network or networks in a connected set of networks, which can be used to route the packets between end domains. EIBP operates at Layer 2.5 and provides encapsulation (at source domain), routing, and de-encapsulation (at destination domain) for packets. EIBP can forward any type of packets between domains. A resolver to map the domain ID to EIBP's edge router addresses is required. When queried for a specific domain, the resolver will return the corresponding edge router structured addresses.

EIBP decouples routing operations from end domain operations, offering to serve any domain, without point solutions to specific domains. EIBP also decouples routing IDs or addresses from end device/domain addresses. This allows for accommodation of new and upcoming domains. A domain can extend EIBP's structured addresses into the domain, by joining as a nested domain under one or more edge routers, or by extending the edge router's structure addresses to its devices.

A greater flexibility in addressing semantics may reduce the aforementioned wastage by accommodating Internet addressing in the light of such alternative forwarding architectures, instead enabling the direct use of the alternative forwarding information.

### 3. Desired Network Features

From the previous subsection, we recognize that Internet technologies are used across a number of scenarios, each of which brings their own (vertical) view on needed capabilities in order to work in a satisfactory manner to those involved.

In the following, we complement those vertical-specific insights with answers to the question of network features that end users (in the form of individuals or organizations alike) desire from the networked system at large. Answers to this question look at the network more from a horizontal perspective, i.e. not with a specific usage in mind beyond communication within and across networks. The text here summarizes the discussion that took place on the INT Area mailing list after IETF112 on this issue. For some of those identified features, we can already identify how innovations on addressing may impact the realization of a particular feature.

We then combine the insights from both scenario-specific and wider horizontal views for the identification of issues when realizing the specific capability of addressing, presented in Section 4.

1. Always-On: The world is getting more and more connected, leading to being connected to the Internet, anywhere, by any technology (e.g., cable, fiber, or radio), even simultaneously, "all the time", and, most importantly, automatically (without any switch turning). However, when defining "all the time" there is a clear and important difference to be made between availability and reliability vs "desired usage". In other words, "always on" can be seen as a desirable perception at the end user level or as a characteristic of the underlying system. From an end user perspective, clearly the former is of importance, not necessarily leading to an "always on" system notion but instead "always-app-available", merely requiring the needed availability and reliability to realize the perception of being "always on" (e.g., for earthquake alerts), possibly complemented by app-specific methods to realize the "always on" perception (e.g., using local caching rather than communication over the network).
2. Transparency: Being agnostic with respect to local domains network protocols (Bluetooth, ZigBee, Thread, Airdrop, Airplay, or any others) is key to provide an easy and straightforward method for contacting people and devices without any knowledge of network issues, particularly those specific to network-specific solutions. While having a flexible addressing model that accommodates a wide range of use cases is important, the centrality of the IP protocol remains key as a mean to provide global connectivity.
3. Multi-homing: Seamless multi-homing capability for the host is key to best use the connectivity options that may be available to an end user, e.g., for increasing resilience in cases of failures of one available option. Protocols like LISP, SHIM6, QUIC, MPTCP, SCTP (to cite a few) have been successful at providing this capability in an incremental way, but too much of that capability is realized within the application, making it hard to leverage across all applications. While today each transport protocol has its own way to perform multi-address discovery, the network layer should provide the multi-homing feature (e.g., SHIM6 can be used to discover all addresses on both ends), and then leave the address selection to the transport. With that, multi-address discovery remains a network feature exposed to the upper layers. This may also mean to update the Socket API (which may be actually the first thing to do), which does not necessarily mean to expose more network details to the applications but instead be more address agnostic yet more expressive.

4. **Mobility:** A lot of work has been put in MobileIP ([RFC5944],[RFC6275]) to provide seamless and lossless communications for moving nodes (vehicle, satellites). However, it has never been widely deployed for several reasons, like complexity of the protocol and the fact that the problem has often been tackled at higher layers, with applications resilient to address changes. However, similar to multi-homing, solving the problem at higher layers means that each and every transport protocol and application have their own way to deal with mobility, leading to similar observations as those for the previous multi-homing aspect.
5. **Security and Privacy:** The COVID-19 pandemic has boosted end users' desire to be protected and protect their privacy. The balance among privacy, security, and accountability is not simple to achieve. There exist different views on what those properties should be, however the network should provide the means to provide what is felt as the best trade-off for the specific use case.
6. **Performance:** While certainly desirable, "performance" is a complex issue that depends on the objectives of those building for but also paying for performance. Examples are (i) speed (shorter paths/direct communications), (ii) bandwidth (10petabit/s for a link), (iii) efficiency (less overlays/encapsulations), (iv) high efficacy or sustainability (avoid waste). From an addressing perspective, length/format/semantics that may adapt to the specific use case (e.g. use short addresses for low power IoT, or, where needed, longer for addresses embedding certificates for strong authentication, authorization and accountability) may contribute to the performance aspects that end users desire, such as reducing waste through not needed encapsulation or needed conversion at network boundaries.
7. **Availability, Reliability, Predictability:** These three properties are important to enable wide-range of services and applications according to the desired usage (cf. point 1).
8. **Do not do harm:** Access to the Internet is considered a human right [RFC8280]. Access to and expression through it should align with this core principle. This issue transcends through a variety of previously discussed 'features' that are desired, such as privacy, security but also availability and reliability. However, lifting the feature of network access onto a basic rights level also brings in the aspect of "do not do harm" through the use of the Internet with respect to wider societal objectives. Similar to other industries, such as electricity or cars, preventing harm usually requires an interplay of

commercial, technological, and regulatory efforts, such as the enforcement of seat belt wearing to reduce accident death. As a first step, the potential harmfulness of a novel method must be recognized and weighted against the benefits of its introduction and use. One increasingly important consideration in the technology domain is "sustainability" of resource usage for an end user's consumption of and participation in Internet services. As an example, Distributed Ledger Technologies (DLT) are seen as an important tool for a variety of applications, including Internet decentralization ([DINRG]). However, the non-linear increase in energy consumption means that extending proof-of-work systems to the entire population of the planet would not only be impractical but also possibly highly wasteful, not just at the level of computational but also communication resource usage [DLT-draft]. This poses the question on how novel methods for addressing may improve on sustainability of such technologies, particularly if adopted more widely.

9. Maximum Transmission Unit (MTU): One long standing issue in the Internet is related to the MTU and how to discover the path MTU in order to avoid fragmentation ([I-D.ietf-6man-mtu-option], [I-D.templin-6man-aero]). While it makes sense to always leverage as much performance from local systems as possible, this should come without sacrificing the ability to communicate with all systems. Having a solid solution to solve the issue would make the overall interconnection of systems more robust.

#### 4. Issues in Addressing

The desired properties outlined in the previous section have implications that go beyond addressing and need to be tackled from a larger architectural point of view. Such a discussion is left as future action, limiting the present document at discussing only the addressing viewpoint and identifying shortcomings perceived from this perspective.

There are a number of issues that we can identify from the communication scenarios in Section 2 and the network features generally desire from the network, presented in Section 3. We do not claim to be exhaustive in our list:

1. Limiting Alternative Address Semantics: Several communication scenarios pursue the use of alternative semantics of what constitute an 'address' of a packet traversing the Internet, which may fall foul of the defined network interface semantic of IP addresses.

2. **Hampering Security:** Aligning with the semantic and length limitations of IP addressing may hamper the security objectives of any new semantic, possibly leading to detrimental effects and possible other workarounds (at the risk of introducing fragility rather than security).
1. **Hampering Privacy:**
  - \* Easy individual identification
  - \* Flow linkability
  - \* App/Activity profiling
2. **Complicating Traffic Engineering:** Utilizing a plethora of non-address inputs into the traffic steering decision in real networks complicates traffic engineering in that it makes the development of suitable policies more complex, while also leading to possible contention between methods being used.
3. **Hampering Efficiency:** Extending IP addressing through point-wise solutions also hampers efficiency, e.g., through needed re-encapsulation (therefore increasing the header processing overhead as well as header-to-payload ratio), through introducing path stretch, or through requiring compression techniques to reduce the header proportion of large addresses when operating in constrained environments.
4. **Fragility:** The introduction of point solutions, each of which comes with possibly own usages of address or packet fields, together with extension-specific operations, increases the overall fragility of the resulting system, caused, for instance, through contention between feature extensions that were neither foreseen in the design nor tested during the implementation phase.
5. **Extensibility:** Accommodating new requirements through ever new extensions as an extensibility approach to addressing compounds aspects discussed before, i.e., fragility, efficiency etc. It complicates engineering due to the clearly missing boundaries against which contentions with other extensions could be managed. It complicates standardization since extension-based extensibility requires independent, and often lengthy, standardization processes. And ultimately, deployments are complicated due to backward compatibility testing required for any new extension being integrated into the deployed system.



The table below shows how the above identified issues do arise somehow in our outlined communication scenarios in Section 2. This overview will be deepened in more details in the gap analysis document [I-D.jia-intarea-internet-addressing-gap-analysis].

	Issue 1	Issue 2	Issue 3	Issue 4	Issue 5	Issue 6
Constrained Environments				x	x	x
Dynamically Changing Topologies	x		x	x	x	x
Moving Endpoints	x		x	x	x	x
Across Services	x		x	x	x	x
Traffic Steering	x		x	x	x	x
Built-in Security	x	x		x	x	x
Alternative Forwarding Architectures	x			x		x

Table 1: Issues Involved in Challenging Scenarios

## 5. Problem Statement

This document identifies a number of scenarios as well as general features end users would want from the network, positioning the existing Internet addressing structure itself as a potential hindrance in solving key problems for Internet service provisioning. Such problems include supporting new, e.g., service-oriented, scenarios more efficiently, with improved security and efficient traffic engineering, as well as large scale mobility. We can observe that those new forms of communication are particularly driven by the conceptual framework of limited domains, realizing the requirements of stakeholders for an optimized communication in those limited domains, while still utilizing the Internet for interconnection as

well as for access to the wealth of existing Internet services.

This co-existence of optimized LD-level as well as Internet communication creates a tussle between those requirements on addressing stemming from those limited domains and those coming from the Internet in the form of agreed IPv6 semantics. This tussle directly refers back to our introductory question on flexibility in addressing (or leaving the problem to limited domain solutions to deal with). It is also captured in the discussion on where new features are being introduced, i.e. at the edge or core of the Internet.

But more importantly, the question on 'what is an address anyway' (derived from what features we may want from the network) should not be guided by the answers that the Internet can give us today, e.g., being a mere ephemeral token for accessing PoP-based services (as indicated in related arch-d mailing list discussions), but instead what features could be enabled by a particular view of what an address is. However, that is not to 'second guess' the market and its possible evolution, but to outline clear features from which to derive clear principles for a design.

For this, it is important to recognize that skewing the technical capabilities of any feature, let alone addressing, to the current economic situation of the Internet bears the danger of locking down innovation capabilities as an outcome of those technical limitations being introduced. Instead, addressing must align with enabling the model of permissionless but compatible innovation that the IETF has been promoting, ultimately enabling the serendipity of new applications that has led to many of those applications we can see in the Internet today.

At this stage, this document does not provide a definite answer nor does it propose or promote specific solutions to the problems here portrayed. Instead, this document aims at stimulating discussion on the emerging needs for addressing, with the possibility to fundamentally re-think the addressing in the Internet beyond the current objectives of IPv6, in order to provide the flexibility to suitably support the many new forms of communication that will emerge. Addressing can be rather flexible and can be of any form that applications may need. There is no limitation on the address to preclude any future applications.

To complement the problem statement in this document, the companion gap analysis document [I-D.jia-intarea-internet-addressing-gap-analysis] deepens the issues identified in Section 4 along key properties of today's Internet addressing.

## 6. Security Considerations

The present memo does not introduce any new technology and/or mechanism and as such does not introduce any security threat to the TCP/IP protocol suite.

Nevertheless, it is worth to observe whether or not greater flexibility of addressing (as suggested in previous sections) would allow to introduce fully featured security in endpoint identification, potentially able to eradicate the spoofing problem, as one example. Furthermore, it may be used to include application gateways' certificates in order to provide more efficiency, e.g., using web certificates also in the addressing of web services. While increasing security, privacy protection may also be improved.

## 7. IANA Considerations

This document does not include an IANA request.

## 8. References

### 8.1. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <<https://www.rfc-editor.org/info/rfc791>>.

### 8.2. Informative References

[ALOHA] Kuo, F., "The ALOHA System", ACM SIGCOMM Computer Communication Review Vol. 25, pp. 41-44, DOI 10.1145/205447.205451, January 1995, <<https://doi.org/10.1145/205447.205451>>.

[BACnet] "BACnet-A Data Communication Protocol for Building Automation and Control Networks", ANSI/ASHRAE Standard 135-2016, January 2016, <[https://www.techstreet.com/ashrae/standards/ashrae-135-2016?product\\_id=1918140](https://www.techstreet.com/ashrae/standards/ashrae-135-2016?product_id=1918140)>.

[BIER-MC] Trossen, D., Rahman, A., Wang, C., and T. Eckert, "Applicability of BIER Multicast Overlay for Adaptive Streaming Services", Work in Progress, Internet-Draft, draft-ietf-bier-multicast-http-response-06, 10 July 2021, <<https://www.ietf.org/archive/id/draft-ietf-bier-multicast-http-response-06.txt>>.

- [BLE] "Bluetooth Specification", Bluetooth SIG Working Groups, n.d., <<https://www.bluetooth.com/specifications>>.
- [CARTISEAN] Hughes, L., Shumon, K., and Y. Zhang, "Cartesian Ad Hoc Routing Protocols", Ad-Hoc, Mobile, and Wireless Networks pp. 287-292, DOI 10.1007/978-3-540-39611-6\_27, 2003, <[https://doi.org/10.1007/978-3-540-39611-6\\_27](https://doi.org/10.1007/978-3-540-39611-6_27)>.
- [CCN] Jacobson, V., Smetters, D., Thornton, J., Plass, M., Briggs, N., and R. Braynard, "Networking named content", Proceedings of the 5th international conference on Emerging networking experiments and technologies - CoNEXT '09, DOI 10.1145/1658939.1658941, 2009, <<https://doi.org/10.1145/1658939.1658941>>.
- [CHEN21] Chen, Y., Li, H., Liu, J., Wu, Q., and Z. Lai, "GAMS: An IP Address Management Mechanism in Satellite Mega-constellation Networks", 2021 International Wireless Communications and Mobile Computing (IWCMC), DOI 10.1109/iwcmc51323.2021.9498722, June 2021, <<https://doi.org/10.1109/iwcmc51323.2021.9498722>>.
- [CHRIKI19] Chriki, A., Touati, H., Snoussi, H., and F. Kamoun, "FANET: Communication, mobility models and security issues", Computer Networks Vol. 163, pp. 106877, DOI 10.1016/j.comnet.2019.106877, November 2019, <<https://doi.org/10.1016/j.comnet.2019.106877>>.
- [DDS] AL-Madani, B., Elkhider, S., and S. El-Ferik, "DDS-Based Containment Control of Multiple UAV Systems", Applied Sciences Vol. 10, pp. 4572, DOI 10.3390/app10134572, July 2020, <<https://doi.org/10.3390/app10134572>>.
- [DECT-ULE] "Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 1: Overview", ETSI European Standard, EN 300 175-1, V2.6.1, May 2009, <[https://www.etsi.org/deliver/etsi\\_en/300100\\_300199/30017501/02.06.01\\_60/en\\_30017501v020601p.pdf](https://www.etsi.org/deliver/etsi_en/300100_300199/30017501/02.06.01_60/en_30017501v020601p.pdf)>.
- [DETNET] "Deterministic Networking (DetNet)", n.d., <<https://datatracker.ietf.org/wg/detnet/about/>>.
- [DINRG] "Decentralized Internet Infrastructure - DINRG", n.d., <<https://datatracker.ietf.org/rg/dinrg/about/>>.

## [DLT-draft]

Trossen, D., Guzman, D., Bride, M. M., and X. Fan, "Impact of DLTs on Provider Networks", Work in Progress, Internet-Draft, draft-trossen-rtgwg-impact-of-dlts-01, 2 March 2022, <<https://www.ietf.org/archive/id/draft-trossen-rtgwg-impact-of-dlts-01.txt>>.

[ECMA-340] EECMA-340, "Near Field Communication - Interface and Protocol (NFCIP-1) 3rd Ed.", June 2013.

[EIBP] Shenoy, S Chandraiah, P Willis, N., "A Structured Approach to Routing in the Internet", June 2021, <First Intl Workshop on Semantic Addressing and Routing for Future Networks>.

[ETSI-NIN] ETSI - European Telecommunication Standards Institute, "Non-IP Networking - NIN", n.d., <<https://www.etsi.org/technologies/non-ip-networking>>.

[HANDLEY] Handley, M., "Delay is Not an Option: Low Latency Routing in Space", Proceedings of the 17th ACM Workshop on Hot Topics in Networks, DOI 10.1145/3286062.3286075, November 2018, <<https://doi.org/10.1145/3286062.3286075>>.

## [I-D.ietf-6man-mtu-option]

Hinden, R. M. and G. Fairhurst, "IPv6 Minimum Path MTU Hop-by-Hop Option", Work in Progress, Internet-Draft, draft-ietf-6man-mtu-option-13, 28 February 2022, <<https://www.ietf.org/archive/id/draft-ietf-6man-mtu-option-13.txt>>.

## [I-D.ietf-intarea-gue]

Herbert, T., Yong, L., and O. Zia, "Generic UDP Encapsulation", Work in Progress, Internet-Draft, draft-ietf-intarea-gue-09, 26 October 2019, <<https://www.ietf.org/archive/id/draft-ietf-intarea-gue-09.txt>>.

## [I-D.ietf-lisp-introduction]

Cabellos, A. and D. S. (Ed.), "An Architectural Introduction to the Locator/ID Separation Protocol (LISP)", Work in Progress, Internet-Draft, draft-ietf-lisp-introduction-15, 20 September 2021, <<https://www.ietf.org/archive/id/draft-ietf-lisp-introduction-15.txt>>.

[I-D.ietf-lisp-mn]

Farinacci, D., Lewis, D., Meyer, D., and C. White, "LISP Mobile Node", Work in Progress, Internet-Draft, draft-ietf-lisp-mn-11, 30 January 2022, <<https://www.ietf.org/archive/id/draft-ietf-lisp-mn-11.txt>>.

[I-D.ietf-lisp-nexagon]

Barkai, S., Fernandez-Ruiz, B., Tamir, R., Rodriguez-Natal, A., Maino, F., Cabellos-Aparicio, A., and D. Farinacci, "Network-Hexagons: H3-LISP GeoState & Mobility Network", Work in Progress, Internet-Draft, draft-ietf-lisp-nexagon-19, 14 September 2021, <<https://www.ietf.org/archive/id/draft-ietf-lisp-nexagon-19.txt>>.

[I-D.ietf-lisp-rfc6833bis]

Farinacci, D., Maino, F., Fuller, V., and A. Cabellos, "Locator/ID Separation Protocol (LISP) Control-Plane", Work in Progress, Internet-Draft, draft-ietf-lisp-rfc6833bis-30, 18 November 2020, <<https://www.ietf.org/archive/id/draft-ietf-lisp-rfc6833bis-30.txt>>.

[I-D.jia-intarea-internet-addressing-gap-analysis]

Jia, Y., Trossen, D., Iannone, L., Shenoy, N., and P. Mendes, "Gap Analysis in Internet Addressing", Work in Progress, Internet-Draft, draft-jia-intarea-internet-addressing-gap-analysis-01, 23 October 2021, <<https://www.ietf.org/archive/id/draft-jia-intarea-internet-addressing-gap-analysis-01.txt>>.

[I-D.templin-6man-aero]

Templin, F. L., "Automatic Extended Route Optimization (AERO)", Work in Progress, Internet-Draft, draft-templin-6man-aero-39, 22 February 2022, <<https://www.ietf.org/archive/id/draft-templin-6man-aero-39.txt>>.

[ICN5G]

Ravindran, R., Suthar, P., Trossen, D., Wang, C., and G. White, "Enabling ICN in 3GPP's 5G NextGen Core Architecture", Work in Progress, Internet-Draft, draft-irtf-icnrg-5gc-icn-04, 10 January 2021, <<https://www.ietf.org/archive/id/draft-irtf-icnrg-5gc-icn-04.txt>>.

- [ICNIP] Trossen, D., Robitzsch, S., Reed, M., Al-Naday, M., and J. Riihijarvi, "Internet Services over ICN in 5G LAN Environments", Work in Progress, Internet-Draft, draft-trossen-icnrg-internet-icn-5gln-04, 1 October 2020, <<https://www.ietf.org/archive/id/draft-trossen-icnrg-internet-icn-5gln-04.txt>>.
- [IEEE\_1901.1] "Standard for Medium Frequency (less than 15 MHz) Power Line Communications for Smart Grid Applications", IEEE 1901.1 IEEE-SA Standards Board, May 2018, <<https://ieeexplore.ieee.org/document/8360785>>.
- [LR-WPAN] "IEEE 802.15.4 - IEEE Standard for Low-Rate Wireless Networks", IEEE 802.15 WPAN Task Group 4, May 2020, <[https://standards.ieee.org/standard/802\\_15\\_4-2020.html](https://standards.ieee.org/standard/802_15_4-2020.html)>.
- [MANET1] Abdallah, A., Abdallah, E., Bsoul, M., and A. Ootom, "Randomized geographic-based routing with nearly guaranteed delivery for three-dimensional ad hoc network", International Journal of Distributed Sensor Networks Vol. 12, pp. 155014771667125, DOI 10.1177/1550147716671255, October 2016, <<https://doi.org/10.1177/1550147716671255>>.
- [MAROJEVIC20] Marojevic, V., Guvenc, I., Dutta, R., Sichitiu, M., and B. Floyd, "Advanced Wireless for Unmanned Aerial Systems: 5G Standardization, Research Challenges, and AERPAAW Architecture", IEEE Vehicular Technology Magazine Vol. 15, pp. 22-30, DOI 10.1109/mvt.2020.2979494, June 2020, <<https://doi.org/10.1109/mvt.2020.2979494>>.
- [OCADO] "Ocado Technologys robot warehouse a Hive of IoT innovation", n.d., <<https://techmonitor.ai/tech-leaders/ocado-technology-robot-hive-innovation>>.
- [PANRG] "Path Aware Networking Research Group - PANRG", n.d., <<https://datatracker.ietf.org/rg/panrg/about/>>.
- [PEARG] "Privacy Enhancements and Assessments Research Group - PEARG", n.d., <<https://irtf.org/pearg>>.
- [PILA] Krahenbuhl, C., Legner, M., Bitterli, S., and A. Perrig, "Pervasive Internet-Wide Low-Latency Authentication", 2021 International Conference on Computer Communications and Networks (ICCCN), DOI 10.1109/icccn52240.2021.9522235, July 2021, <<https://doi.org/10.1109/icccn52240.2021.9522235>>.

- [RFC2775] Carpenter, B., "Internet Transparency", RFC 2775, DOI 10.17487/RFC2775, February 2000, <<https://www.rfc-editor.org/info/rfc2775>>.
- [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <<https://www.rfc-editor.org/info/rfc3031>>.
- [RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, DOI 10.17487/RFC3972, March 2005, <<https://www.rfc-editor.org/info/rfc3972>>.
- [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>>.
- [RFC4919] Kushalnagar, N., Montenegro, G., and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals", RFC 4919, DOI 10.17487/RFC4919, August 2007, <<https://www.rfc-editor.org/info/rfc4919>>.
- [RFC5061] Stewart, R., Xie, Q., Tuexen, M., Maruyama, S., and M. Kozuka, "Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration", RFC 5061, DOI 10.17487/RFC5061, September 2007, <<https://www.rfc-editor.org/info/rfc5061>>.
- [RFC5177] Leung, K., Dommety, G., Narayanan, V., and A. Petrescu, "Network Mobility (NEMO) Extensions for Mobile IPv4", RFC 5177, DOI 10.17487/RFC5177, April 2008, <<https://www.rfc-editor.org/info/rfc5177>>.
- [RFC5275] Turner, S., "CMS Symmetric Key Management and Distribution", RFC 5275, DOI 10.17487/RFC5275, June 2008, <<https://www.rfc-editor.org/info/rfc5275>>.
- [RFC5517] HomChaudhuri, S. and M. Foschiano, "Cisco Systems' Private VLANs: Scalable Security in a Multi-Client Environment", RFC 5517, DOI 10.17487/RFC5517, February 2010, <<https://www.rfc-editor.org/info/rfc5517>>.
- [RFC5944] Perkins, C., Ed., "IP Mobility Support for IPv4, Revised", RFC 5944, DOI 10.17487/RFC5944, November 2010, <<https://www.rfc-editor.org/info/rfc5944>>.



- [RFC6158] DeKok, A., Ed. and G. Weber, "RADIUS Design Guidelines", BCP 158, RFC 6158, DOI 10.17487/RFC6158, March 2011, <<https://www.rfc-editor.org/info/rfc6158>>.
- [RFC6182] Ford, A., Raiciu, C., Handley, M., Barre, S., and J. Iyengar, "Architectural Guidelines for Multipath TCP Development", RFC 6182, DOI 10.17487/RFC6182, March 2011, <<https://www.rfc-editor.org/info/rfc6182>>.
- [RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <<https://www.rfc-editor.org/info/rfc6275>>.
- [RFC6626] Tsirtsis, G., Park, V., Narayanan, V., and K. Leung, "Dynamic Prefix Allocation for Network Mobility for Mobile IPv4 (NEMOv4)", RFC 6626, DOI 10.17487/RFC6626, May 2012, <<https://www.rfc-editor.org/info/rfc6626>>.
- [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>>.
- [RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <<https://www.rfc-editor.org/info/rfc7401>>.
- [RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S., Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, DOI 10.17487/RFC7426, January 2015, <<https://www.rfc-editor.org/info/rfc7426>>.
- [RFC7429] Liu, D., Ed., Zuniga, JC., Ed., Seite, P., Chan, H., and CJ. Bernardos, "Distributed Mobility Management: Current Practices and Gap Analysis", RFC 7429, DOI 10.17487/RFC7429, January 2015, <<https://www.rfc-editor.org/info/rfc7429>>.
- [RFC7476] Pentikousis, K., Ed., Ohlman, B., Corujo, D., Boggia, G., Tyson, G., Davies, E., Molinaro, A., and S. Eum, "Information-Centric Networking: Baseline Scenarios", RFC 7476, DOI 10.17487/RFC7476, March 2015, <<https://www.rfc-editor.org/info/rfc7476>>.

- [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <<https://www.rfc-editor.org/info/rfc7665>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC8280] ten Oever, N. and C. Cath, "Research into Human Rights Protocol Considerations", RFC 8280, DOI 10.17487/RFC8280, October 2017, <<https://www.rfc-editor.org/info/rfc8280>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8595] Farrel, A., Bryant, S., and J. Drake, "An MPLS-Based Forwarding Plane for Service Function Chaining", RFC 8595, DOI 10.17487/RFC8595, June 2019, <<https://www.rfc-editor.org/info/rfc8595>>.
- [RFC8677] Trossen, D., Purkayastha, D., and A. Rahman, "Name-Based Service Function Forwarder (nSFF) Component within a Service Function Chaining (SFC) Framework", RFC 8677, DOI 10.17487/RFC8677, November 2019, <<https://www.rfc-editor.org/info/rfc8677>>.
- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.
- [RFC8763] Rahman, A., Trossen, D., Kutscher, D., and R. Ravindran, "Deployment Considerations for Information-Centric Networking (ICN)", RFC 8763, DOI 10.17487/RFC8763, April 2020, <<https://www.rfc-editor.org/info/rfc8763>>.
- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.
- [RFC8926] Gross, J., Ed., Ganga, I., Ed., and T. Sridhar, Ed., "Geneve: Generic Network Virtualization Encapsulation", RFC 8926, DOI 10.17487/RFC8926, November 2020, <<https://www.rfc-editor.org/info/rfc8926>>.

- [RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://www.rfc-editor.org/info/rfc9000>>.
- [SFCANYCAST] Wion, A., Bouet, M., Iannone, L., and V. Conan, "Distributed Function Chaining with Anycast Routing", Proceedings of the 2019 ACM Symposium on SDN Research, DOI 10.1145/3314148.3314355, April 2019, <<https://doi.org/10.1145/3314148.3314355>>.
- [TERASTREAM] "Deutsche Telekom tests TeraStream, the network of the future, in Croatia", n.d., <<https://www.telekom.com/en/media/media-information/archive/deutsche-telekom-tests-terastream-the-network-of-the-future-in-croatia-358444>>.
- [TROSSEN] Trossen, D., Sarela, M., and K. Sollins, "Arguments for an information-centric internetworking architecture", ACM SIGCOMM Computer Communication Review Vol. 40, pp. 26-33, DOI 10.1145/1764873.1764878, April 2010, <<https://doi.org/10.1145/1764873.1764878>>.
- [WANG19] Wang, P., Zhang, J., Zhang, X., Yan, Z., Evans, B., and W. Wang, "Convergence of Satellite and Terrestrial Networks: A Comprehensive Survey", IEEE Access Vol. 8, pp. 5550-5588, DOI 10.1109/access.2019.2963223, 2020, <<https://doi.org/10.1109/access.2019.2963223>>.

#### Acknowledgments

Thanks to all the people that shared insightful comments both privately to the authors as well as on various mailing list, especially on the INTArea Mailing List. Also thanks for the interesting discussions to Stewart Bryant, Ron Bonica, Toerless Eckert, Brian E. Carpenter, Kiran Makhijani, Fred Templin.

#### Authors' Addresses

Yihao Jia  
Huawei Technologies Co., Ltd  
156 Beiqing Rd.  
Beijing  
100095  
P.R. China  
Email: [jiayihao@huawei.com](mailto:jiayihao@huawei.com)

Dirk Trossen  
Huawei Technologies Duesseldorf GmbH  
Riesstr. 25C  
80992 Munich  
Germany  
Email: dirk.trossen@huawei.com

Luigi Iannone  
Huawei Technologies France S.A.S.U.  
18, Quai du Point du Jour  
92100 Boulogne-Billancourt  
France  
Email: luigi.iannone@huawei.com

Nirmala Shenoy  
Rochester Institute of Technology  
New-York, 14623  
United States of America  
Email: nxsvks@rit.edu

Paulo Mendes  
Airbus  
Willy-Messerschmitt Strasse 1  
81663 Munich  
Germany  
Email: paulo.mendes@airbus.com

Donald E. Eastlake 3rd  
Futurewei Technologies  
2386 Panoramic Circle  
Apopka, FL, 32703  
United States of America  
Email: d3e3e3@gmail.com

Peng Liu  
China Mobile  
32 Xuanwumen West Ave  
Xicheng, Beijing  
100053  
P.R. China  
Email: liupengygy@chinamobile.com

Dino Farinacci  
lispers.net  
United States of America  
Email: farinacci@gmail.com

IPPM  
Internet-Draft  
Intended status: Informational  
Expires: January 13, 2022

M. Cociglio  
Telecom Italia - TIM  
A. Ferrieux  
Orange Labs  
G. Fioccola  
Huawei Technologies  
I. Lubashev  
Akamai Technologies  
F. Bulgarella  
Telecom Italia - TIM  
I. Hamchaoui  
Orange Labs  
M. Nilo  
Telecom Italia - TIM  
R. Sisto  
Politecnico di Torino  
D. Tikhonov  
LiteSpeed Technologies  
July 12, 2021

Explicit Flow Measurements Techniques  
draft-mdt-ippm-explicit-flow-measurements-02

Abstract

This document describes protocol independent methods called Explicit Flow Measurement Techniques that employ few marking bits, inside the header of each packet, for loss and delay measurement. The endpoints, marking the traffic, signal these metrics to intermediate observers allowing them to measure connection performance, and to locate the network segment where impairments happen. Different alternatives are considered within this document. These signaling methods apply to all protocols but they are especially valuable when applied to protocols that encrypt transport header and do not allow traditional methods for delay and loss detection.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 13, 2022.

#### Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

#### Table of Contents

1. Introduction . . . . .	3
2. Notational Conventions . . . . .	4
3. Latency Bits . . . . .	5
3.1. Spin Bit . . . . .	5
3.2. Delay Bit . . . . .	6
3.2.1. Generation Phase . . . . .	8
3.2.2. Reflection Phase . . . . .	8
3.2.3. T_Max Selection . . . . .	9
3.2.4. Delay Measurement using Delay Bit . . . . .	10
3.2.5. Observer's Algorithm . . . . .	12
3.2.6. Two Bits Delay Measurement: Spin Bit + Delay Bit . .	13
3.2.7. Hidden Delay Bit - Delay Bit with Privacy Protection	13
4. Loss Bits . . . . .	13
4.1. T Bit - Round Trip Loss Bit . . . . .	14
4.1.1. Round Trip Packet Loss Measurement . . . . .	15
4.1.2. Setting the Round Trip Loss Bit on Outgoing Packets .	17
4.1.3. Observer's Logic for Round Trip Loss Signal . . . . .	18
4.1.4. Loss Coverage and Signal Timing . . . . .	19
4.2. Q Bit - Square Bit . . . . .	19
4.2.1. Q Block Length Selection . . . . .	19
4.2.2. Upstream Loss . . . . .	20
4.2.3. Identifying Q Block Boundaries . . . . .	21
4.3. L Bit - Loss Event Bit . . . . .	21
4.3.1. End-To-End Loss . . . . .	22

4.3.2. Loss Profile Characterization . . . . .	22
4.4. L+Q Bits - Upstream, Downstream, and End-to-End Loss Measurements . . . . .	22
4.4.1. Correlating End-to-End and Upstream Loss . . . . .	23
4.5. R Bit - Reflection Square Bit . . . . .	24
4.5.1. R+Q Bits - Using R and Q Bits for Passive Loss Measurement . . . . .	25
4.5.2. Enhancement of R Block Length Computation . . . . .	29
4.5.3. Improved Resilience to Packet Reordering . . . . .	29
4.6. Improved Q and R Bits Resilience to Burst Losses . . . . .	29
5. Summary of Delay and Loss Marking Methods . . . . .	30
6. ECN-Echo Event Bit . . . . .	32
6.1. Setting the ECN-Echo Event Bit on Outgoing Packets . . . . .	32
6.2. Using E Bit for Passive ECN-Reported Congestion Measurement . . . . .	32
7. Protocol Ossification Considerations . . . . .	33
8. Examples of Application . . . . .	33
8.1. QUIC . . . . .	33
8.2. TCP . . . . .	34
9. Security Considerations . . . . .	34
9.1. Optimistic ACK Attack . . . . .	35
10. Privacy Considerations . . . . .	35
11. IANA Considerations . . . . .	36
12. Change Log . . . . .	36
13. Contributors . . . . .	36
14. Acknowledgements . . . . .	36
15. References . . . . .	36
15.1. Normative References . . . . .	36
15.2. Informative References . . . . .	37
Authors' Addresses . . . . .	39

## 1. Introduction

Packet loss and delay are hard and pervasive problems of day-to-day network operation. Proactively detecting, measuring, and locating them is crucial to maintaining high QoS and timely resolution of crippling end-to-end throughput issues. To this effect, in a TCP-dominated world, network operators have been heavily relying on information present in the clear in TCP headers: sequence and acknowledgment numbers and SACKs when enabled (see [RFC8517]). These allow for quantitative estimation of packet loss and delay by passive on-path observation. Additionally, the problem can be quickly identified in the network path by moving the passive observer around.

With encrypted protocols, the equivalent transport headers are encrypted and passive packet loss and delay observations are not possible, as described in [TRANSPORT-ENCRYPT].



Measuring TCP loss and delay between similar endpoints cannot be relied upon to evaluate encrypted protocol loss and delay. Different protocols could be routed by the network differently, and the fraction of Internet traffic delivered using protocols other than TCP is increasing every year. It is imperative to measure packet loss and delay experienced by encrypted protocol users directly.

This document defines Explicit Flow Measurement Techniques. These hybrid measurement path signals (see [IPM-Methods]) are to be embedded into a transport layer protocol and are explicitly intended for exposing RTT and loss rate information to on-path measurement devices. These measurement mechanisms are applicable to any transport-layer protocol, and, as an example, the document describes QUIC and TCP bindings.

The Explicit Flow Measurement Techniques described in this document can be used alone or in combination with other Explicit Flow Measurement Techniques. Each technique uses a small number of bits and exposes a specific measurement.

Following the recommendation in [RFC8558] of making path signals explicit, this document proposes adding a small number of dedicated measurement bits to the clear portion of the protocol headers. These bits can be added to an encrypted portion of a header belonging to any protocol layer, e.g. IP (see [IP]) and IPv6 (see [IPv6]) headers or extensions, such as [IPv6AltMark], UDP surplus space (see [UDP-OPTIONS] and [UDP-SURPLUS]), reserved bits in a QUIC v1 header (see [QUIC-TRANSPORT]).

The measurements are not designed for use in automated control of the network in environments where signal bits are set by untrusted hosts. Instead, the signal is to be used for troubleshooting individual flows as well as for monitoring the network by aggregating information from multiple flows and raising operator alarms if aggregate statistics indicate a potential problem.

The spin bit, delay bit and loss bits explained in this document are inspired by [AltMark], [SPIN-BIT], [I-D.trammell-tsvwg-spin] and [I-D.trammell-ippm-spin].

Additional details about the Performance Measurements for QUIC are described in the paper [ANRW19-PM-QUIC].

## 2. Notational Conventions

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. Latency Bits

This section introduces bits that can be used for round trip latency measurements. Whenever this section of the specification refers to packets, it is referring only to packets with protocol headers that include the latency bits.

[QUIC-TRANSPORT] introduces an explicit per-flow transport-layer signal for hybrid measurement of RTT. This signal consists of a spin bit that toggles once per RTT. [SPIN-BIT] discusses an additional two-bit Valid Edge Counter (VEC) to compensate for loss and reordering of the spin bit and increase fidelity of the signal in less than ideal network conditions.

This document introduces a stand-alone single-bit delay signal that can be used by passive observers to measure the RTT of a network flow, avoiding the spin bit ambiguities that arise as soon as network conditions deteriorate.

#### 3.1. Spin Bit

This section is a small recap of the spin bit working mechanism. For a comprehensive explanation of the algorithm, please see [SPIN-BIT].

The spin bit is an alternate marking [AltMark] generated signal, where the size of the alternation changes with the flight size each RTT.

The latency spin bit is a single bit signal that toggles once per RTT, enabling latency monitoring of a connection-oriented communication from intermediate observation points.

A "spin period" is a set of packets with the same spin bit value sent during one RTT time interval. A "spin period value" is the value of the spin bit shared by all packets in a spin period.

The client and server maintain an internal per-connection spin value (i.e. 0 or 1) used to set the spin bit on outgoing packets. Both endpoints initialize the spin value to 0 when a new connection starts. Then:

- when the client receives a packet with the packet number larger than any number seen so far, it sets the connection spin value to the opposite value contained in the received packet;
- when the server receives a packet with the packet number larger than any number seen so far, it sets the connection spin value to the same value contained in the received packet.

The computed spin value is used by the endpoints for setting the spin bit on outgoing packets. This mechanism allows the endpoints to generate a square wave such that, by measuring the distance in time between pairs of consecutive edges observed in the same direction, a passive on-path observer can compute the round trip delay of that network flow.

Spin bit enables round trip latency measurement by observing a single direction of the traffic flow.

Note that packet reordering can cause spurious edges that require heuristics to correct. The spin bit performance deteriorates as soon as network impairments arise as explained in Section 3.2.

### 3.2. Delay Bit

The delay bit has been designed to overcome accuracy limitations experienced by the spin bit under difficult network conditions:

- packet reordering leads to generation of spurious edges and errors in delay estimation;
- loss of edges causes wrong estimation of spin periods and therefore wrong RTT measurements;
- application-limited senders cause the spin bit to measure the application delays instead of network delays.

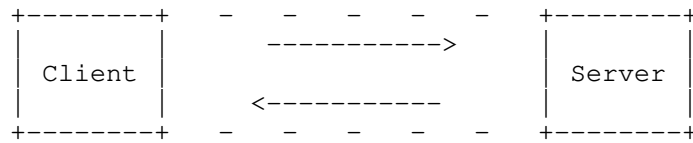
Unlike the spin bit, which is set in every packet transmitted on the network, the delay bit is set only once per round trip.

When the delay bit is used, a single packet with a marked bit (the delay bit) bounces between a client and a server during the entire connection lifetime. This single packet is called "delay sample".

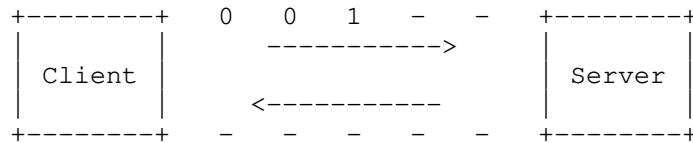
An observer placed at an intermediate point, observing a single direction of traffic, tracking the delay sample and the relative timestamp, can measure the round trip delay of the connection.

The delay sample lifetime is comprised of two phases: initialization and reflection. The initialization is the generation of the delay sample, while the reflection realizes the bounce behavior of this single packet between the two endpoints.

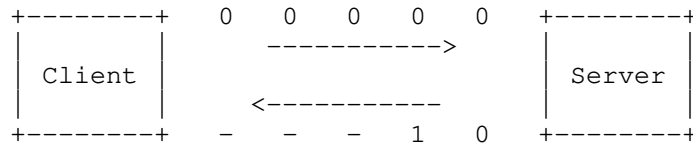
The next figure describes the elementary Delay bit mechanism.



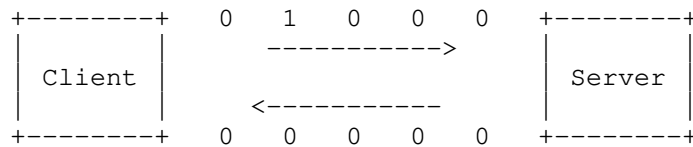
(a) No traffic at beginning.



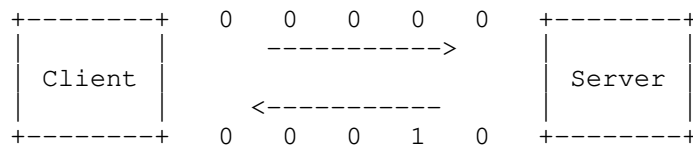
(b) The Client starts sending data and sets the first packet as Delay Sample.



(c) The Server starts sending data and reflects the Delay Sample.



(d) The Client reflects the Delay Sample.



(e) The Server reflects the Delay Sample and so on.

Delay bit mechanism

### 3.2.1. Generation Phase

Only client is actively involved in the generation phase. It maintains an internal per-flow timestamp variable ("ds\_time") updated every time a delay sample is transmitted.

When connection starts, the client generates a new delay sample initializing the delay bit of the first outgoing packet to 1. Then it updates the "ds\_time" variable with the timestamp of its transmission.

The server initializes the delay bit to 0 at the beginning of the connection, and its only task during the connection is described in Section 3.2.2.

In absence of network impairments, the delay sample should bounce between client and server continuously, for the entire duration of the connection. That is highly unlikely for two reasons:

1. the packet carrying the delay bit might be lost;
2. an endpoint could stop or delay sending packets because the application is limiting the amount of traffic transmitted;

To deal with these problems, the client generates a new delay sample if more than a predetermined time ("T\_Max") has elapsed since the last delay sample transmission (including reflections). Note that "T\_Max" should be greater than the max measurable RTT on the network. See Section 3.2.3 for details.

### 3.2.2. Reflection Phase

Reflection is the process that enables the bouncing of the delay sample between a client and a server. The behavior of the two endpoints is almost the same.

- Server side reflection: when a delay sample arrives, the server marks the first packet in the opposite direction as the delay sample.
- Client side reflection: when a delay sample arrives, the client marks the first packet in the opposite direction as the delay sample. It also updates the "ds\_time" variable when the outgoing delay sample is actually forwarded.

In both cases, if the outgoing delay sample is being transmitted with a delay greater than a predetermined threshold after the reception of

the incoming delay sample (1ms by default), the delay sample is not reflected, and the outgoing delay bit is kept at 0.

By doing so, the algorithm can reject measurements that would overestimate the delay due to lack of traffic on the endpoints. Hence, the maximum estimation error would amount to twice the threshold (e.g. 2ms) per measurement.

### 3.2.3. T\_Max Selection

The internal "ds\_time" variable allows a client to identify delay sample losses. Considering that a lost delay sample is regenerated at the end of an explicit time ("T\_Max") since the last generation, this same value can be used by an observer to reject a measure and start a new one.

In other words, if the difference in time between two delay samples is greater or equal than "T\_Max", then these cannot be used to produce a delay measure. Therefore the value of "T\_Max" must also be known to the on-path network probes.

There are two alternatives to select the "T\_Max" value so that both client and observers know it. The first one requires that "T\_Max" is known a priori ("T\_Max\_p") and therefore set within the protocol specifications that implements the marking mechanism (e.g. 1 second which usually is greater than the max expectable RTT). The second alternative requires a dynamic mechanism able to adapt the duration of the "T\_Max" to the delay of the connection ("T\_Max\_c").

For instance, client and observers could use the connection RTT as a basis for calculating an effective "T\_Max". They should use a predetermined initial value so that "T\_Max = T\_Max\_p" (e.g. 1 second) and then, when a valid RTT is measured, change "T\_Max" accordingly so that "T\_Max = T\_Max\_c". In any case, the selected "T\_Max" should be large enough to absorb any possible variations in the connection delay.

"T\_Max\_c" could be computed as two times the measured "RTT" plus a fixed amount of time ("100ms") to prevent low "T\_Max" values in case of very small RTTs. The resulting formula is: "T\_Max\_c = 2RTT + 100ms". If "T\_Max\_c" is greater than "T\_Max\_p" then "T\_Max\_c" is forced to "T\_Max\_p" value.

Note that the observer's "T\_Max" should always be less than or equal to the client's "T\_Max" to avoid considering as a valid measurement what is actually the client's "T\_Max". To obtain this result, the client waits for two consecutive incoming samples and computes the two related RTTs. Then it takes the largest of them as the basis of

the "T\_Max\_c" formula. At this point, observers have already measured a valid RTT and then computed their "T\_Max\_c".

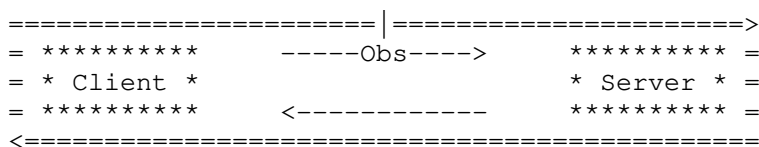
### 3.2.4. Delay Measurement using Delay Bit

When the Delay Bit is used, a passive observer can use delay samples directly and avoid inherent ambiguities in the calculation of the RTT as can be seen in spin bit analysis.

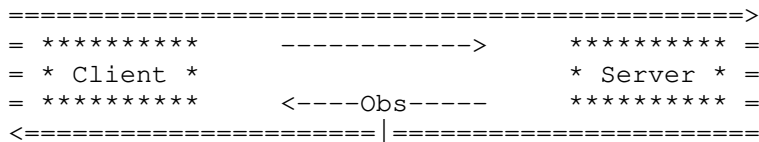
#### 3.2.4.1. RTT Measurement

The delay sample generation process ensures that only one packet marked with the delay bit set to 1 runs back and forth between two endpoints per round trip time. To determine the RTT measurement of a flow, an on-path passive observer computes the time difference between two delay samples observed in a single direction.

To ensure a valid measurement, the observer must verify that the distance in time between the two samples taken into account is less than "T\_Max".



(a) client-server RTT



(b) server-client RTT

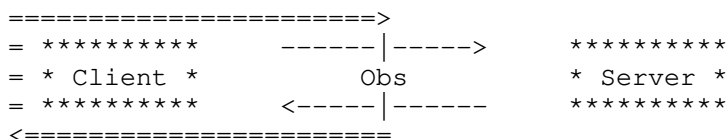
Round-trip time (both direction)

#### 3.2.4.2. Half-RTT Measurement

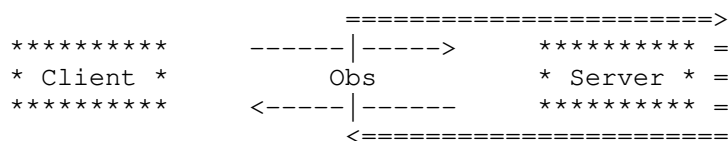
An observer that is able to observe both forward and return traffic directions can use the delay samples to measure "upstream" and "downstream" RTT components, also known as the half-RTT measurements. It does this by measuring the time between a delay sample observed in one direction and the delay sample previously observed in the opposite direction.

As with RTT measurement, the observer must verify that the distance in time between the two samples taken into account is less than "T\_Max".

Note that upstream and downstream sections of paths between the endpoints and the observer, i.e. observer-to-client vs client-to-observer and observer-to-server vs server-to-observer, may have different delay characteristics due to the difference in network congestion and other factors.



(a) client-observer half-RTT



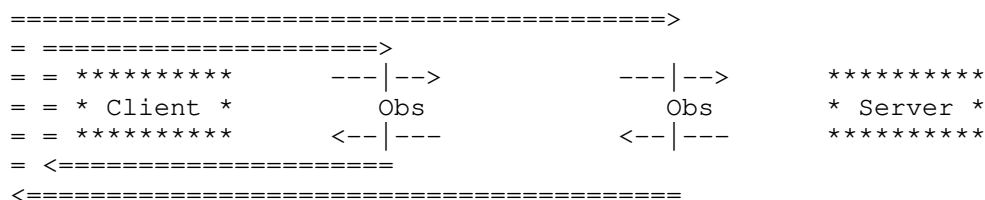
(b) observer-server half-RTT

Half Round-trip time (both direction)

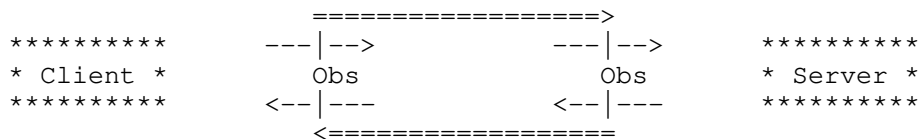
#### 3.2.4.3. Intra-Domain RTT Measurement

Intra-domain RTT is the portion of the entire RTT used by a flow to traverse the network of a provider. To measure intra-domain RTT, two observers capable of observing traffic in both directions must be employed simultaneously at ingress and egress of the network to be measured. Intra-domain RTT is difference between the two computed upstream (or downstream) RTT components.





(a) client-observer RTT components (half-RTTs)



(b) the intra-domain RTT resulting from the subtraction of the above RTT components

Intra-domain Round-trip time (client-observer: upstream)

### 3.2.5. Observer's Algorithm

An on-path observer maintains an internal per-flow variable to keep track of time at which the last delay sample has been observed.

A unidirectional observer, upon detecting a delay sample:

- if a delay sample was also detected previously in the same direction and the distance in time between them is less than " $T_{Max} - K$ ", then the two delay samples can be used to calculate RTT measurement. " $K$ " is a protection threshold to absorb differences in " $T_{Max}$ " computation and delay variations between two consecutive delay samples (e.g. " $K = 10\% T_{Max}$ ").

If the observer can observe both forward and return traffic flows, and it is able to determine which direction contains the client and the server (e.g. by observing the connection handshake), upon detecting a delay sample:

- if a delay sample was also detected in the opposite direction and the distance in time between them is less than " $T_{Max} - K$ ", then the two delay samples can be used to measure the observer-client half-RTT or the observer-server half-RTT, according to the direction of the last delay sample observed.

### 3.2.6. Two Bits Delay Measurement: Spin Bit + Delay Bit

Spin and Delay bit algorithms work independently. If both marking methods are used in the same connection, observers can choose the best measurement between the two available:

- when a precise measurement can be produced using the delay bit, observers choose it;
- when a delay bit measurement is not available, observers choose the approximate spin bit one.

### 3.2.7. Hidden Delay Bit - Delay Bit with Privacy Protection

Theoretically, delay measurements can be used to roughly evaluate the distance of the client from the server (using the RTT) or from any intermediate observer (using the client-observer half-RTT). To protect users privacy, the algorithm of the delay bit can be slightly modified to mask the RTT of the connection to an intermediate observer. This result can be achieved using a simple expedient which consists in delaying the client-side reflection of the delay sample by a predetermined time value. This would lead an intermediate observer to inevitably measure a delay greater than the real one.

The Additional Delay should be randomly selected by the client and kept constant for a certain amount of time across multiple connections. This ensures that the client-server jitter remains the same as if no Additional Delay had been inserted. For instance, a new Additional Delay value could be generated whenever the client's IP address changes.

Using this technique, despite the Additional Delay introduced, it is still possible to correctly measure the right component of RTT (observer-server) and all the intra-domain measurements used to distribute the delay in the network. Furthermore, differently from the Delay Bit, the hidden Delay Bit makes the use of the client reflection threshold (1ms) redundant. Removing this threshold leads to the further advantage of increasing the number of valid measurements produced by the algorithm.

## 4. Loss Bits

This section introduces bits that can be used for loss measurements. Whenever this section of the specification refers to packets, it is referring only to packets with protocol headers that include the loss bits - the only packets whose loss can be measured.

- T: the "round Trip loss" bit is used in combination with the Spin bit to measure round-trip loss. See Section 4.1.
- Q: the "Square signal" bit is used to measure upstream loss. See Section 4.2.
- L: the "Loss event" bit is used to measure end-to-end loss. See Section 4.3.
- R: the "Reflection square signal" bit is used in combination with Q bit to measure end-to-end loss. See Section 4.1.

Loss measurements enabled by T, Q, and L bits can be implemented by those loss bits alone (T bit requires a working Spin Bit). Two-bit combinations Q+L and Q+R enable additional measurement opportunities discussed below.

Each endpoint maintains appropriate counters independently and separately for each separately identifiable flow (each sub-flow for multipath connections).

Since loss is reported independently for each flow, all bits (except for L bit) require a certain minimum number of packets to be exchanged per flow before any signal can be measured. Therefore, loss measurements work best for flows that transfer more than a minimal amount of data.

#### 4.1. T Bit - Round Trip Loss Bit

The round Trip loss bit is used to mark a variable number of packets exchanged twice between the endpoints realizing a two round-trip reflection. A passive on-path observer, observing either direction, can count and compare the number of marked packets seen during the two reflections, estimating the loss rate experienced by the connection. The overall exchange comprises:

- The client selects, generates and consequently transmits a first train of packets, by setting the T bit to 1;
- The server, upon receiving each packet included in the first train, reflects to the client a respective second train of packets of the same size as the first train received, by setting the T bit to 1;
- The client, upon receiving each packet included in the second train, reflects to the server a respective third train of packets of the same size as the second train received, by setting the T bit to 1;

- The server, upon receiving each packet included in the third train, finally reflects to the client a respective fourth train of packets of the same size as the third train received, by setting the T bit to 1.

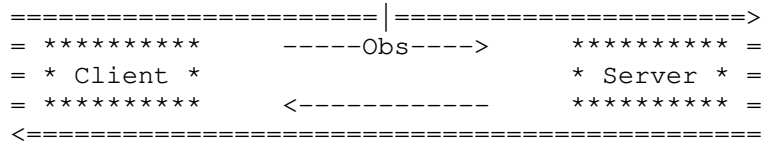
Packets belonging to the first round trip (first and second train) represent the Generation Phase, while those belonging to the second round trip (third and fourth train) represent the Reflection Phase.

A passive on-path observer can count and compare the number of marked packets seen during the two round trips (i.e. the first and third or the second and the fourth trains of packets, depending on which direction is observed) and estimate the loss rate experienced by the connection. This process is repeated continuously to obtain more measurements as long as the endpoints exchange traffic. These measurements can be called Round Trip losses.

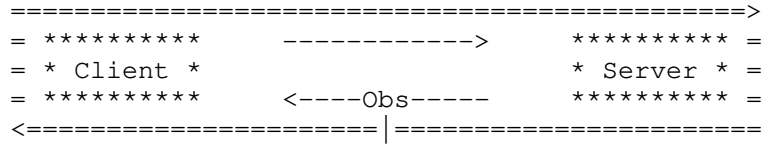
Since packet rates in two directions may be different, the number of marked packets in the train is determined by the direction with the lowest packet rate. See Section 4.1.2 for details on packet generation and for a mechanism to allow an observer to distinguish between trains belonging to different phases (Generation and Reflection).

#### 4.1.1. Round Trip Packet Loss Measurement

Since the measurements are performed on a portion of the traffic exchanged between the client and the server, the observer calculates the end-to-end Round Trip Packet Loss (RTPL) that, statistically, will correspond to the loss rate experienced by the connection along the entire network path.



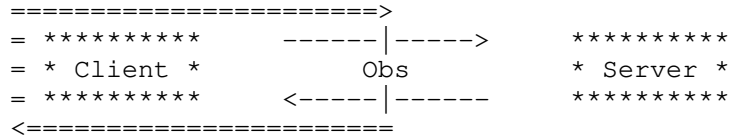
(a) client-server RTPL



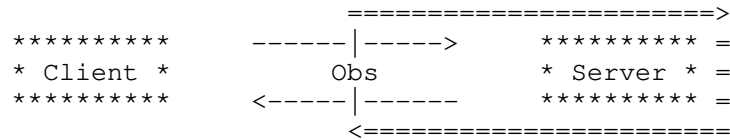
(b) server-client RTPL

Round-trip packet loss (both direction)

This methodology also allows the Half-RTPL measurement and the Intra-domain RTPL measurement in a way similar to RTT measurement.

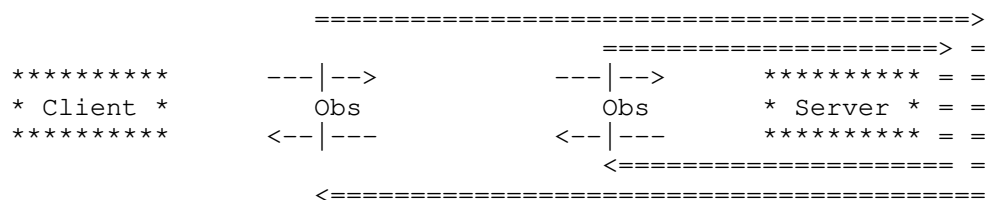


(a) client-observer half-RTPL

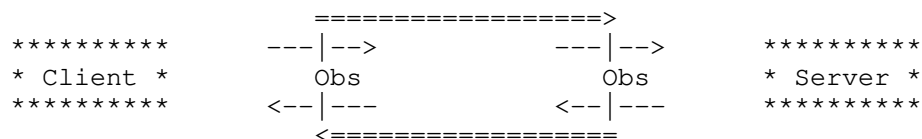


(b) observer-server half-RTPL

Half Round-trip packet loss (both direction)



(a) observer-server RTPL components (half-RTPLs)



(b) the intra-domain RTPL resulting from the subtraction of the above RTPL components

Intra-domain Round-trip packet loss (observer-server)

#### 4.1.2. Setting the Round Trip Loss Bit on Outgoing Packets

The round Trip loss signal requires a working Spin-bit signal to separate trains of marked packets (packets with T bit set to 1). A "pause" of at least one empty spin-bit period between each phase of the algorithm serves as such separator for the on-path observer.

The client is in charge of launching trains of marked packets and does so according to the algorithm:

1. Generation Phase. The client starts generating marked packets for two consecutive spin-bit periods; it maintains a "generation token" count that is reset to zero at the beginning of the algorithm phase and is incremented every time a packet arrives. When the client transmits a packet and a "generation token" is available, the client marks the packet and retires a "generation token". If no token is available, the outgoing packet is transmitted unmarked. At the end of the first spin-bit period spent in generation, the reflection counter is unlocked to start counting incoming marked packets that will be reflected later;
2. Pause Phase. When the generation is completed, the client pauses till it has observed one entire spin bit period with no marked packets. That spin bit period is used by the observer as a separator between generated and reflected packets. During this marking pause, all the outgoing packets are transmitted with T

bit set to 0. The reflection counter is still incremented every time a marked packet arrives;

3. Reflection Phase. The client starts transmitting marked packets, decrementing the reflection counter for each transmitted marked packet until the reflection counter reached zero. The "generation token" method from the generation phase is used during this phase as well. At the end of the first spin-period spent in reflection, the reflection counter is locked to avoid incoming reflected packets incrementing it;
4. Pause Phase 2. The pause phase is repeated after the reflection phase and serves as a separator between the reflected packet train and a new packet train.

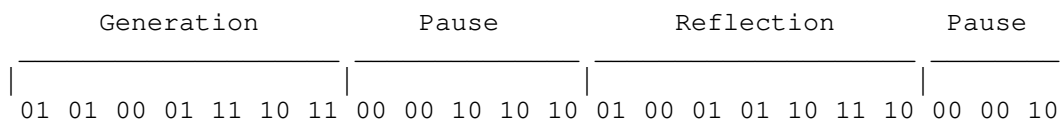
The generation token counter should be capped to limit the effects of a subsequent sudden reduction in the other endpoint's packet rate that could prevent that endpoint from reflecting collected packets. The most conservative cap value is "1".

A server maintains a "marking counter" that starts at zero and is incremented every time a marked packet arrives. When the server transmits a packet and the "marking counter" is positive, the server marks the packet and decrements the "marking counter". If the "marking counter" is zero, the outgoing packet is transmitted unmarked.

#### 4.1.3. Observer's Logic for Round Trip Loss Signal

The on-path observer counts marked packets and separates different trains by detecting spin-bit periods (at least one) with no marked packets. The Round Trip Packet Loss (RTPL) is the difference between the size of the Generation train and the Reflection train.

In the following example, packets are represented by two bits (first one is the spin bit, second one is the loss bit):



Round Trip Loss signal example

Note that 5 marked packets have been generated of which 4 have been reflected.

#### 4.1.4. Loss Coverage and Signal Timing

A cycle of the round Trip loss signaling algorithm contains 2 RTTs of Generation phase, 2 RTTs of Reflection phase, and two Pause phases at least 1 RTT in duration each. Hence, the loss signal is delayed by about 6 RTTs since the loss events.

The observer can only detect loss of marked packets that occurs after its initial observation of the Generation phase and before its subsequent observation of the Reflection phase. Hence, if the loss occurs on the path that sends packets at a lower rate (typically ACKs in such asymmetric scenarios), "2/6" ("1/3") of the packets will be sampled for loss detection.

If the loss occurs on the path that sends packets at a higher rate, " $\text{lowPacketRate}/(3*\text{highPacketRate})$ " of the packets will be sampled for loss detection. For protocols that use ACKs, the portion of packets sampled for loss in the higher rate direction during unidirectional data transfer is " $1/(3*\text{packetsPerAck})$ ", where the value of "packetsPerAck" can vary by protocol, by implementation, and by network conditions.

#### 4.2. Q Bit - Square Bit

The sSquare bit (Q bit) takes its name from the square wave generated by its signal. Every outgoing packet contains the Q bit value, which is initialized to the 0 and inverted after sending N packets (a sSquare Block or simply Q Block). Hence, Q Period is  $2*N$ . The Q bit represents "packet color" as defined by [AltMark].

Observation points can estimate upstream losses by watching a single direction of the traffic flow and counting the number of packets in each observed Q Block, as described in Section 4.2.2.

##### 4.2.1. Q Block Length Selection

The length of the block must be known to the on-path network probes. There are two alternatives to selecting the Q Block length. The first one requires that the length is known a priori and therefore set within the protocol specifications that implements the marking mechanism. The second requires the sender to select it.

In this latter scenario, the sender is expected to choose N (Q Block length) based on the expected amount of loss and reordering on the path. The choice of N strikes a compromise - the observation could become too unreliable in case of packet reordering and/or severe loss if N is too small, while short flows may not yield a useful upstream loss measurement if N is too large (see Section 4.2.2).



The value of  $N$  should be at least 64 and be a power of 2. This requirement allows an Observer to infer the Q Block length by observing one period of the square signal. It also allows the Observer to identify flows that set the loss bits to arbitrary values (see Section 7).

If the sender does not have sufficient information to make an informed decision about Q Block length, the sender should use  $N=64$ , since this value has been extensively tried in large-scale field tests and yielded good results. Alternatively, the sender may also choose a random power-of-2  $N$  for each flow, increasing the chances of using a Q Block length that gives the best signal for some flows.

The sender must keep the value of  $N$  constant for a given flow.

#### 4.2.2. Upstream Loss

Blocks of  $N$  (Q Block length) consecutive packets are sent with the same value of the Q bit, followed by another block of  $N$  packets with an inverted value of the Q bit. Hence, knowing the value of  $N$ , an on-path observer can estimate the amount of upstream loss after observing at least  $N$  packets. The upstream loss rate ("uloss") is one minus the average number of packets in a block of packets with the same Q value ("p") divided by  $N$  ("uloss= $1-\text{avg}(p)/N$ ").

The observer needs to be able to tolerate packet reordering that can blur the edges of the square signal, as explained in Section 4.2.3.

```

=====>
*****      -----Obs----->      *****
* Client *                               * Server *
*****      <-----              *****

```

(a) in client-server channel (uloss\_up)

```

*****      ----->      *****
* Client *                               * Server *
*****      <----Obs----- *****
<=====

```

(b) in server-client channel (uloss\_down)

Upstream loss

#### 4.2.3. Identifying Q Block Boundaries

Packet reordering can produce spurious edges in the square signal. To address this, the observer should look for packets with the current Q bit value up to X packets past the first packet with a reverse Q bit value. The value of X, a "Marking Block Threshold", should be less than "N/2".

The choice of X represents a trade-off between resiliency to reordering and resiliency to loss. A very large Marking Block Threshold will be able to reconstruct Q Blocks despite a significant amount of reordering, but it may erroneously coalesce packets from multiple Q Blocks into fewer Q Blocks, if loss exceeds 50% for some Q Blocks.

#### 4.3. L Bit - Loss Event Bit

The Loss Event bit uses an Unreported Loss counter maintained by the protocol that implements the marking mechanism. To use the Loss Event bit, the protocol must allow the sender to identify lost packets. This is true of protocols such as QUIC, partially true for TCP and SCTP (losses of pure ACKs are not detected) and is not true of protocols such as UDP and IP/IPv6.

The Unreported Loss counter is initialized to 0, and L bit of every outgoing packet indicates whether the Unreported Loss counter is positive (L=1 if the counter is positive, and L=0 otherwise).

The value of the Unreported Loss counter is decremented every time a packet with L=1 is sent.

The value of the Unreported Loss counter is incremented for every packet that the protocol declares lost, using whatever loss detection machinery the protocol employs. If the protocol is able to rescind the loss determination later, a positive Unreported Loss counter may be decremented due to the rescission, but it should NOT become negative due to the rescission.

This loss signaling is similar to loss signaling in [ConEx], except the Loss Event bit is reporting the exact number of lost packets, whereas Echo Loss bit in [ConEx] is reporting an approximate number of lost bytes.

For protocols, such as TCP ([TCP]), that allow network devices to change data segmentation, it is possible that only a part of the packet is lost. In these cases, the sender must increment Unreported Loss counter by the fraction of the packet data lost (so Unreported

Loss counter may become negative when a packet with L=1 is sent after a partial packet has been lost).

Observation points can estimate the end-to-end loss, as determined by the upstream endpoint, by counting packets in this direction with the L bit equal to 1, as described in Section 4.3.1.

#### 4.3.1. End-To-End Loss

The Loss Event bit allows an observer to estimate the end-to-end loss rate by counting packets with L bit value of 0 and 1 for a given flow. The end-to-end loss rate is the fraction of packets with L=1.

The assumption here is that upstream loss affects packets with L=0 and L=1 equally. If some loss is caused by tail-drop in a network device, this may be a simplification. If the sender's congestion controller reduces the packet send rate after loss, there may be a sufficient delay before sending packets with L=1 that they have a greater chance of arriving at the observer.

#### 4.3.2. Loss Profile Characterization

In addition to measuring the end-to-end loss rate, the Loss Event bit allows an observer to characterize loss profile, since the distribution of observed packets with L bit set to 1 roughly corresponds to the distribution of packets lost between 1 RTT and 1 RTO before (see Section 4.4.1). Hence, observing random single instances of L bit set to 1 indicates random single packet loss, while observing blocks of packets with L bit set to 1 indicates loss affecting entire blocks of packets.

#### 4.4. L+Q Bits - Upstream, Downstream, and End-to-End Loss Measurements

Combining L and Q bits allows a passive observer watching a single direction of traffic to accurately measure:

- upstream loss: sender-to-observer loss (see Section 4.2.2)
- downstream loss: observer-to-receiver loss (see Section 4.4.1.1)
- end-to-end loss: sender-to-receiver loss on the observed path (see Section 4.3.1) with loss profile characterization (see Section 4.3.2)

#### 4.4.1. Correlating End-to-End and Upstream Loss

Upstream loss is calculated by observing packets that did not suffer the upstream loss (Section 4.2.2). End-to-end loss, however, is calculated by observing subsequent packets after the sender's protocol detected the loss. Hence, end-to-end loss is generally observed with a delay of between 1 RTT (loss declared due to multiple duplicate acknowledgments) and 1 RTO (loss declared due to a timeout) relative to the upstream loss.

The flow RTT can sometimes be estimated by timing protocol handshake messages. This RTT estimate can be greatly improved by observing a dedicated protocol mechanism for conveying RTT information, such as the Spin bit (see Section 3.1) or Delay bit (see Section 3.2).

Whenever the observer needs to perform a computation that uses both upstream and end-to-end loss rate measurements, it should use upstream loss rate leading the end-to-end loss rate by approximately 1 RTT. If the observer is unable to estimate RTT of the flow, it should accumulate loss measurements over time periods of at least 4 times the typical RTT for the observed flows.

If the calculated upstream loss rate exceeds the end-to-end loss rate calculated in Section 4.3.1, then either the Q Period is too short for the amount of packet reordering or there is observer loss, described in Section 4.4.1.2. If this happens, the observer should adjust the calculated upstream loss rate to match end-to-end loss rate, unless the following applies.

In case of a protocol like TCP and SCTP that does not track losses of pure ACK packets, observing a direction of traffic dominated by pure ACK packets could result in measured upstream loss that is higher than measured end-to-end loss, if said pure ACK packets are lost upstream. Hence, if the measurement is applied to such protocols, and the observer can confirm that pure ACK packets dominate the observed traffic direction, the observer should adjust the calculated end-to-end loss rate to match upstream loss rate.

##### 4.4.1.1. Downstream Loss

Because downstream loss affects only those packets that did not suffer upstream loss, the end-to-end loss rate ("eloss") relates to the upstream loss rate ("uloss") and downstream loss rate ("dloss") as  $(1-uloss)(1-dloss)=1-eloss$ . Hence,  $dloss=(eloss-uloss)/(1-uloss)$ .

#### 4.4.1.2. Observer Loss

A typical deployment of a passive observation system includes a network tap device that mirrors network packets of interest to a device that performs analysis and measurement on the mirrored packets. The observer loss is the loss that occurs on the mirror path.

Observer loss affects upstream loss rate measurement, since it causes the observer to account for fewer packets in a block of identical Q bit values (see Section 4.2.2). The end-to-end loss rate measurement, however, is unaffected by the observer loss, since it is a measurement of the fraction of packets with the L bit value of 1, and the observer loss would affect all packets equally (see Section 4.3.1).

The need to adjust the upstream loss rate down to match end-to-end loss rate as described in Section 4.4.1 is an indication of the observer loss, whose magnitude is between the amount of such adjustment and the entirety of the upstream loss measured in Section 4.2.2. Alternatively, a high apparent upstream loss rate could be an indication of significant packet reordering, possibly due to packets belonging to a single flow being multiplexed over several upstream paths with different latency characteristics.

#### 4.5. R Bit - Reflection Square Bit

R bit requires a deployment alongside Q bit. Unlike the square signal for which packets are transmitted into blocks of fixed size, the Reflection square signal (being an alternate marking signal too) produces blocks of packets whose size varies according to these rules:

- when the transmission of a new block starts, its size is set equal to the size of the last Q Block whose reception has been completed;
- if, before transmission of the block is terminated, the reception of at least one further Q Block is completed, the size of the block is updated to the average size of the further received Q Blocks. Implementation details follow.

The Reflection square value is initialized to 0 and is applied to the R-bit of every outgoing packet. The Reflection square value is toggled for the first time when the completion of a Q Block is detected in the incoming square signal (produced by the opposite node using the Q-bit). When this happens, the number of packets ("p"), detected within this first Q Block, is used to generate a reflection

square signal which toggles every "M=p" packets (at first). This new signal produces blocks of M packets (marked using the R-bit) and each of them is called "Reflection Block" (R Block).

The M value is then updated every time a completed Q Block in the incoming square signal is received, following this formula:  
"M=round(avg(p))".

The parameter "avg(p)" is the average number of packets in a marking period computed considering all the Q Blocks received since the beginning of the current R Block.

To ensure a proper computation of the M value, endpoints implementing the R bit must identify the boundaries of incoming Q Blocks. The same approach described in {#endmarkingblock} should be used.

Looking at the R-bit, unidirectional observation points have an indication of losses experienced by the entire unobserved channel plus those occurred in the path from the sender up to them.

Since the Q Block is sent in one direction, and the corresponding reflected R Block is sent in the opposite direction, the reflected R signal is transmitted with the packet rate of the slowest direction. Namely, if the observed direction is the slowest, there can be multiple Q Blocks transmitted in the unobserved direction before a complete R Block is transmitted in the observed direction. If the unobserved direction is the slowest, the observed direction can be sending R Blocks of the same size repeatedly before it can update the signal to account for a newly-completed Q Block.

#### 4.5.1. R+Q Bits - Using R and Q Bits for Passive Loss Measurement

Since both sSquare and Reflection square bits are toggled at most every N packets (except for the first transition of the R-bit as explained before), an on-path observer can count the number of packets of each marking block and, knowing the value of N, can estimate the amount of loss experienced by the connection. An observer can calculate different measurements depending on whether it is able to observe a single direction of the traffic or both directions.

Single directional observer:

- upstream loss in the observed direction: the loss between the sender and the observation point (see Section 4.2.2)

- "three-quarters" connection loss: the loss between the receiver and the sender in the unobserved direction plus the loss between the sender and the observation point in the observed direction
- end-to-end loss in the unobserved direction: the loss between the receiver and the sender in the opposite direction

Two directions observer (same metrics seen previously applied to both direction, plus):

- client-observer half round-trip loss: the loss between the client and the observation point in both directions
- observer-server half round-trip loss: the loss between the observation point and the server in both directions
- downstream loss: the loss between the observation point and the receiver (applicable to both directions)

#### 4.5.1.1. Three-Quarters Connection Loss

Except for the very first block in which there is nothing to reflect (a complete Q Block has not been yet received), packets are continuously R-bit marked into alternate blocks of size lower or equal than N. Knowing the value of N, an on-path observer can estimate the amount of loss occurred in the whole opposite channel plus the loss from the sender up to it in the observation channel. As for the previous metric, the "three-quarters" connection loss rate ("tqloss") is one minus the average number of packets in a block of packets with the same R value ("t") divided by "N" ("tqloss=1-avg(t)/N").

```

=====>
= *****      -----Obs----->      *****
= * Client *                               * Server *
= *****      <-----              *****
<=====

```

(a) in client-server channel (tqloss\_up)

```

=====>
*****      ----->      ***** =
* Client *                               * Server * =
*****      <-----Obs-----      ***** =
<=====

```

(b) in server-client channel (tqloss\_down)

#### Three-quarters connection loss

The following metrics derive from this last metric and the upstream loss produced by the Q Bit.

#### 4.5.1.2. End-To-End Loss in the Opposite Direction

End-to-end loss in the unobserved direction ("eloss\_unobserved") relates to the "three-quarters" connection loss ("tqloss") and upstream loss in the observed direction ("uloss") as  $(1 - \text{eloss\_unobserved})(1 - \text{uloss}) = 1 - \text{tqloss}$ . Hence,  $\text{eloss\_unobserved} = (\text{tqloss} - \text{uloss}) / (1 - \text{uloss})$ .

```

*****      -----Obs----->      *****
* Client *                               * Server *
*****      <-----              *****
<=====

```

(a) in client-server channel (eloss\_down)

```

=====>
*****      ----->      *****
* Client *                               * Server *
*****      <-----Obs-----      *****

```

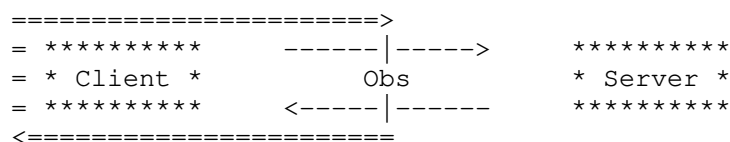
(b) in server-client channel (eloss\_up)

#### End-To-End loss in the opposite direction

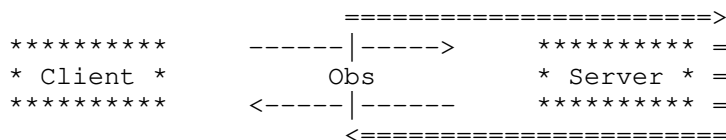


## 4.5.1.3. Half Round-Trip Loss

If the observer is able to observe both directions of traffic, it is able to calculate two "half round-trip" loss measurements - loss from the observer to the receiver (in a given direction) and then back to the observer in the opposite direction. For both directions, "half round-trip" loss ("hrtloss") relates to "three-quarters" connection loss ("tqloss\_opposite") measured in the opposite direction and the upstream loss ("uloss") measured in the given direction as  $(1-uloss)(1-hrtloss)=1-tqloss\_opposite$ . Hence,  $hrtloss=(tqloss\_opposite-uloss)/(1-uloss)$ .



(a) client-observer half round-trip loss (hrtloss\_co)



(b) observer-server half round-trip loss (hrtloss\_os)

Half Round-trip loss (both direction)

## 4.5.1.4. Downstream Loss

If the observer is able to observe both directions of traffic, it is able to calculate two downstream loss measurements using either end-to-end loss and upstream loss, similar to the calculation in Section 4.4.1.1 or using "half round-trip" loss and upstream loss in the opposite direction.

For the latter,  $dloss=(hrtloss-uloss\_opposite)/(1-uloss\_opposite)$ .

```

          =====>
*****  -----|----->  *****
* Client *      Obs      * Server *
*****  <-----|-----  *****

```

(a) in client-server channel (dloss\_up)

```

*****  -----|----->  *****
* Client *      Obs      * Server *
*****  <-----|-----  *****
<=====

```

(b) in server-client channel (dloss\_down)

Downstream loss

#### 4.5.2. Enhancement of R Block Length Computation

The use of the rounding function used in the M computation introduces errors that can be minimized by storing the rounding applied each time M is computed, and using it during the computation of the M value in the following R Block.

This can be achieved introducing the new "r\_avg" parameter in the computation of M. The new formula is "Mr=avg(p)+r\_avg; M=round(Mr); r\_avg=Mr-M" where the initial value of "r\_avg" is equal to 0.

#### 4.5.3. Improved Resilience to Packet Reordering

When a protocol implementing the marking mechanism is able to detect when packets are received out of order, it can improve resilience to packet reordering beyond what is possible using methods described in Section 4.2.3.

This can be achieved by updating the size of the current R Block while this is being transmitted. The reflection block size is then updated every time an incoming reordered packet of the previous Q Block is detected. This can be done if and only if the transmission of the current reflection block is in progress and no packets of the following Q Block have been received.

#### 4.6. Improved Q and R Bits Resilience to Burst Losses

Burst losses can affect Q and R measurements accuracy. Generally, burst losses can be absorbed and correctly measured if smaller than the established Q Block length. On the other hand, entire periods might be wiped out if the burst sizes become too large thus making the observer completely unaware of their loss.

To improve burst loss resilience, an observer might consider a received Q or R Block larger than the selected Q Block length as a burst loss event. Then compute the loss as three times Q Block length minus the measured block length. By doing so, an observer can detect burst losses of less than two blocks (e.g., less than 128 packets for Q Block length of 64 packets). A burst loss equal or greater than two consecutive periods would still remain unnoticed by the observer (or underestimated if a period longer than Q Block length were formed).

## 5. Summary of Delay and Loss Marking Methods

This section summarizes the marking methods described in this draft.

For the Delay measurement, it is possible to use the spin bit and/or the delay bit. A unidirectional or bidirectional observer can be used.

Method	# of bits	Available Delay Metrics		Impairments Resiliency	# of meas.
		UNIDIR Observer	BIDIR Observer		
S: Spin Bit	1	RTT	x2 Half RTT	low	very high
D: Delay Bit	1	RTT	x2 Half RTT	high	medium
D <sup>^</sup> : Hidden Delay Bit	1	RTT <sup>^</sup>	x2 Left Half <sup>^</sup> Right Half	high	high
SD: Spin Bit & Delay Bit *	2	RTT	x2 Half RTT	high	very high

x2 Same metric for both directions

\* Both algorithms work independtly; an observer could use approximate spin bit measures when delay bit ones aren't available

<sup>^</sup> Masked metric (real value can be calculated only by those who know the Additional Delay)

Figure 1: Delay Comparison

For the Loss measurement, each row in the table of Figure 2 represents a loss marking method. For each method the table specifies the number of bits required in the header, the available metrics using an unidirectional or bidirectional observer, applicable protocols, measurement fidelity and delay.

Method	Bits	Available Loss Metrics		Protocols	Measurement Aspects	
		UNIDIR Observer	BIDIR Observer		Fidelity	Delay
T: Round Trip Loss Bit	\$ 1	RT	x2 Half RT	*	Rate by sampling 1/3 to 1/(3*ppa) of pkts over 2 RTT	~6 RTT
Q: Square Bit	1	Upstream	x2	*	Rate over N pkts (e.g. 64)	N pkts (e.g. 64)
L: Loss Event Bit	1	E2E	x2	#	Loss shape (and rate)	Min: RTT Max: RTO
QL: Square + Loss Ev. Bits	2	Upstream Downstream E2E	x2 x2 x2	#	-> see Q -> see Q L -> see L	Up: see Q Others: see L
QR: Square + Ref. Sq. Bits	2	Upstream 3/4 RT !E2E	x2 x2 E2E Downstream Half RT	*	Rate over N*ppa pkts (see Q bit for N)	Up: see Q Others: N*ppa pk (see Q for N)

\* All protocols

# Protocols employing loss detection (w/ or w/o pure ACK loss detection)

\$ Require a working spin bit

! Metric relative to the opposite channel

x2 Same metric for both directions

ppa Packets-Per-Ack

Q|L See Q if Upstream loss is significant; L otherwise

Figure 2: Loss Comparison

## 6. ECN-Echo Event Bit

While the primary focus of the draft is on exposing packet loss and delay, modern networks can report congestion before they are forced to drop packets, as described in [ECN]. When transport protocols keep ECN-Echo feedback under encryption, this signal cannot be observed by the network operators. When tasked with diagnosing network performance problems, knowledge of a congestion downstream of an observation point can be instrumental.

If downstream congestion information is desired, this information can be signaled with an additional bit.

- E: The "ECN-Echo Event" bit is set to 0 or 1 according to the Unreported ECN Echo counter, as explained below in Section 6.1.

### 6.1. Setting the ECN-Echo Event Bit on Outgoing Packets

The Unreported ECN-Echo counter operates identically to Unreported Loss counter (Section 4.3), except it counts packets delivered by the network with CE markings, according to the ECN-Echo feedback from the receiver.

This ECN-Echo signaling is similar to ECN signaling in [ConEx]. ECN-Echo mechanism in QUIC provides the number of packets received with CE marks. For protocols like TCP, the method described in [ConEx-TCP] can be employed. As stated in [ConEx-TCP], such feedback can be further improved using a method described in [ACCURATE].

### 6.2. Using E Bit for Passive ECN-Reported Congestion Measurement

A network observer can count packets with CE codepoint and determine the upstream CE-marking rate directly.

Observation points can also estimate ECN-reported end-to-end congestion by counting packets in this direction with a E bit equal to 1.

The upstream CE-marking rate and end-to-end ECN-reported congestion can provide information about downstream CE-marking rate. Presence of E bits along with L bits, however, can somewhat confound precise estimates of upstream and downstream CE-markings in case the flow contains packets that are not ECN-capable.

## 7. Protocol Ossification Considerations

Accurate loss and delay information is not critical to the operation of any protocol, though its presence for a sufficient number of flows is important for the operation of networks.

The delay and loss bits are amenable to "greasing" described in [RFC8701], if the protocol designers are not ready to dedicate (and ossify) bits used for loss reporting to this function. The greasing could be accomplished similarly to the Latency Spin bit greasing in [QUIC-TRANSPORT]. Namely, implementations could decide that a fraction of flows should not encode loss and delay information and, instead, the bits would be set to arbitrary values. The observers would need to be ready to ignore flows with delay and loss information more resembling noise than the expected signal.

## 8. Examples of Application

### 8.1. QUIC

The binding of a delay signal to QUIC is partially described in [QUIC-TRANSPORT], which adds the spin bit to the first byte of the short packet header, leaving two reserved bits for future experiments.

To implement the additional signals discussed in this document, the first byte of the short packet header can be modified as follows:

- the delay bit (D) can be placed in the first reserved bit (i.e. the fourth most significant bit `_0x10_`) while the round trip loss bit (T) in the second reserved bit (i.e. the fifth most significant bit `_0x08_`); the proposed scheme is:

```

  0 1 2 3 4 5 6 7
+--+--+--+--+--+--+
|0|1|S|D|T|K|P|P|
+--+--+--+--+--+--+

```

Scheme 1

- alternatively, a two bits loss signal (QL or QR) can be placed in both reserved bits; the proposed schemes, in this case, are:

```

  0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|0|1|S|Q|L|K|P|P|
+---+---+---+---+---+---+

```

Scheme 2A

```

  0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|0|1|S|Q|R|K|P|P|
+---+---+---+---+---+---+

```

Scheme 2B

A further option would be to substitute the spin bit with the delay bit (or hidden delay bit) leaving the two reserved bits for loss detection. The proposed schemes are:

```

  0 1 2 3 4 5 6 7          0 1 2 3 4 5 6 7
+---+---+---+---+---+---+  +---+---+---+---+---+---+
|0|1|D|Q|L|K|P|P|  OR  |0|1|D^|Q|L|K|P|P|
+---+---+---+---+---+---+  +---+---+---+---+---+---+

```

Scheme 3A

```

  0 1 2 3 4 5 6 7          0 1 2 3 4 5 6 7
+---+---+---+---+---+---+  +---+---+---+---+---+---+
|0|1|D|Q|R|K|P|P|  OR  |0|1|D^|Q|R|K|P|P|
+---+---+---+---+---+---+  +---+---+---+---+---+---+

```

Scheme 3B

## 8.2. TCP

The signals can be added to TCP by defining bit 4 of byte 13 of the TCP header to carry the spin bit or the delay bit, and possibly bits 5 and 6 to carry additional information, like the delay bit and the round-trip loss bit (DT), or a two bits loss signal (QL or QR).

## 9. Security Considerations

Passive loss and delay observations have been a part of the network operations for a long time, so exposing loss and delay information to the network does not add new security concerns for protocols that are currently observable.

In the absence of packet loss, Q and R bits signals do not provide any information that cannot be observed by simply counting packets

transiting a network path. In the presence of packet loss, Q and R bits will disclose the loss, but this is information about the environment and not the endpoint state. The L bit signal discloses internal state of the protocol's loss detection machinery, but this state can often be gleamed by timing packets and observing congestion controller response.

Hence, loss bits do not provide a viable new mechanism to attack data integrity and secrecy.

### 9.1. Optimistic ACK Attack

A defense against an Optimistic ACK Attack, described in [QUIC-TRANSPORT], involves a sender randomly skipping packet numbers to detect a receiver acknowledging packet numbers that have never been received. The Q bit signal may inform the attacker which packet numbers were skipped on purpose and which had been actually lost (and are, therefore, safe for the attacker to acknowledge). To use the Q bit for this purpose, the attacker must first receive at least an entire Q Block of packets, which renders the attack ineffective against a delay-sensitive congestion controller.

A protocol that is more susceptible to an Optimistic ACK Attack with the loss signal provided by Q bit and uses a loss-based congestion controller, should shorten the current Q Block by the number of skipped packets numbers. For example, skipping a single packet number will invert the square signal one outgoing packet sooner.

Similar considerations apply to the R Bit, although a shortened R Block along with a matching skip in packet numbers does not necessarily imply a lost packet, since it could be due to a lost packet on the reverse path along with a deliberately skipped packet by the sender.

## 10. Privacy Considerations

To minimize unintentional exposure of information, loss bits provide an explicit loss signal - a preferred way to share information per [RFC8558].

New protocols commonly have specific privacy goals, and loss reporting must ensure that loss information does not compromise those privacy goals. For example, [QUIC-TRANSPORT] allows changing Connection IDs in the middle of a connection to reduce the likelihood of a passive observer linking old and new sub-flows to the same device. A QUIC implementation would need to reset all counters when it changes the destination (IP address or UDP port) or the Connection ID used for outgoing packets. It would also need to avoid



incrementing Unreported Loss counter for loss of packets sent to a different destination or with a different Connection ID.

## 11. IANA Considerations

This document makes no request of IANA.

## 12. Change Log

TBD

## 13. Contributors

The following people provided valuable contributions to this document:

- Marcus Ihlar, Ericsson, [marcus.ihlar@ericsson.com](mailto:marcus.ihlar@ericsson.com)
- Jari Arkko, Ericsson, [jari.arkko@ericsson.com](mailto:jari.arkko@ericsson.com)
- Emile Stephan, Orange, [emile.stephan@orange.com](mailto:emile.stephan@orange.com)

## 14. Acknowledgements

TBD

## 15. References

### 15.1. Normative References

- [ConEx] Mathis, M. and B. Briscoe, "Congestion Exposure (ConEx) Concepts, Abstract Mechanism, and Requirements", RFC 7713, DOI 10.17487/RFC7713, December 2015, <<https://www.rfc-editor.org/info/rfc7713>>.
- [ConEx-TCP] Kuehlewind, M., Ed. and R. Scheffenegger, "TCP Modifications for Congestion Exposure (ConEx)", RFC 7786, DOI 10.17487/RFC7786, May 2016, <<https://www.rfc-editor.org/info/rfc7786>>.
- [ECN] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <<https://www.rfc-editor.org/info/rfc3168>>.

- [IP] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <<https://www.rfc-editor.org/info/rfc791>>.
- [IPM-Methods] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [IPv6] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [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>>.
- [RFC8558] Hardie, T., Ed., "Transport Protocol Path Signals", RFC 8558, DOI 10.17487/RFC8558, April 2019, <<https://www.rfc-editor.org/info/rfc8558>>.
- [TCP] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <<https://www.rfc-editor.org/info/rfc793>>.

## 15.2. Informative References

- [ACCURATE] Briscoe, B., Kuehlewind, M., and R. Scheffenegger, "More Accurate ECN Feedback in TCP", draft-ietf-tcpm-accurate-ecn-14 (work in progress), February 2021.
- [AltMark] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.
- [ANRW19-PM-QUIC] Bulgarella, F., Cociglio, M., Fioccola, G., Marchetto, G., and R. Sisto, "Performance measurements of QUIC communications", Proceedings of the Applied Networking Research Workshop, DOI 10.1145/3340301.3341127, July 2019.

- [I-D.trammell-ippm-spin]  
Trammell, B., "An Explicit Transport-Layer Signal for Hybrid RTT Measurement", draft-trammell-ippm-spin-00 (work in progress), January 2019.
- [I-D.trammell-tsvwg-spin]  
Trammell, B., "A Transport-Independent Explicit Signal for Hybrid RTT Measurement", draft-trammell-tsvwg-spin-00 (work in progress), July 2018.
- [IPv6AltMark]  
Fioccola, G., Zhou, T., Cociglio, M., Qin, F., and R. Pang, "IPv6 Application of the Alternate Marking Method", draft-ietf-6man-ipv6-alt-mark-04 (work in progress), March 2021.
- [QUIC-TRANSPORT]  
Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-34 (work in progress), January 2021.
- [RFC8517] Dolson, D., Ed., Snellman, J., Boucadair, M., Ed., and C. Jacquenet, "An Inventory of Transport-Centric Functions Provided by Middleboxes: An Operator Perspective", RFC 8517, DOI 10.17487/RFC8517, February 2019, <<https://www.rfc-editor.org/info/rfc8517>>.
- [RFC8701] Benjamin, D., "Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility", RFC 8701, DOI 10.17487/RFC8701, January 2020, <<https://www.rfc-editor.org/info/rfc8701>>.
- [SPIN-BIT]  
Trammell, B., Vaere, P. D., Even, R., Fioccola, G., Fossati, T., Ihlar, M., Morton, A., and E. Stephan, "Adding Explicit Passive Measurability of Two-Way Latency to the QUIC Transport Protocol", draft-trammell-quic-spin-03 (work in progress), May 2018.
- [TRANSPORT-ENCRYPT]  
Fairhurst, G. and C. Perkins, "Considerations around Transport Header Confidentiality, Network Operations, and the Evolution of Internet Transport Protocols", draft-ietf-tsvwg-transport-encrypt-21 (work in progress), April 2021.

## [UDP-OPTIONS]

Touch, J., "Transport Options for UDP", draft-ietf-tsvwg-udp-options-12 (work in progress), May 2021.

## [UDP-SURPLUS]

Herbert, T., "UDP Surplus Header", draft-herbert-udp-space-hdr-01 (work in progress), July 2019.

## Authors' Addresses

Mauro Cociglio  
Telecom Italia - TIM  
Via Reiss Romoli, 274  
Torino 10148  
Italy

EMail: mauro.cociglio@telecomitalia.it

Alexandre Ferrieux  
Orange Labs

EMail: alexandre.ferrieux@orange.com

Giuseppe Fioccola  
Huawei Technologies  
Riesstrasse, 25  
Munich 80992  
Germany

EMail: giuseppe.fioccola@huawei.com

Igor Lubashev  
Akamai Technologies

EMail: ilubashe@akamai.com

Fabio Bulgarella  
Telecom Italia - TIM  
Via Reiss Romoli, 274  
Torino 10148  
Italy

EMail: fabio.bulgarella@guest.telecomitalia.it

Isabelle Hamchaoui  
Orange Labs

EMail: [isabelle.hamchaoui@orange.com](mailto:isabelle.hamchaoui@orange.com)

Massimo Nilo  
Telecom Italia - TIM  
Via Reiss Romoli, 274  
Torino 10148  
Italy

EMail: [massimo.nilo@telecomitalia.it](mailto:massimo.nilo@telecomitalia.it)

Riccardo Sisto  
Politecnico di Torino

EMail: [riccardo.sisto@polito.it](mailto:riccardo.sisto@polito.it)

Dmitri Tikhonov  
LiteSpeed Technologies

EMail: [dtikhonov@litespeedtech.com](mailto:dtikhonov@litespeedtech.com)

Internet Draft  
Intended status: Experimental  
Expires: April 2022

P. Urien  
Telecom Paris  
October 4 2021

Internet of Secure Elements  
draft-urien-coinrg-iose-04.txt

## Abstract

This draft defines an infrastructure for secure elements over internet, and features needed for their secure remote use. It describes a network architecture based on the TLS 1.3 protocol, which enables remote calls of cryptographic procedures, identified by Unified Resource Identifier (URI) such as `schemeS://sen@server.com:443/?query`. The Internet of Secure Element (IoSE) is a set of secure elements providing TLS servers, communication interfaces, and identified by their name (Secure Element Name, sen).

## 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.

## Status of this Memo

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

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

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

This Internet-Draft will expire on April 2022

## Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

Abstract.....	1
Requirements Language.....	1
Status of this Memo.....	1
Copyright Notice.....	2
1 Overview.....	4
2. About Secure Elements.....	5
3. Network Architecture.....	6
4 Unified Resource Identifier (URI).....	7
5 URI Example.....	7
6 Overview of Internet Of Secure Elements Framework.....	8
7 Functional Entities.....	9
8 Attestation Procedure.....	9
8.1 Service Request.....	11
8.2 SE-App Downloading.....	11
8.3 SE-App Certificate.....	11
8.4 User Notification.....	12
8.5 User Enrollment.....	12
9 IANA Considerations.....	12
10 Security Considerations.....	12
11 References.....	12
11.1 Normative References.....	12
11.2 Informative References.....	13
12 Authors' Addresses.....	13



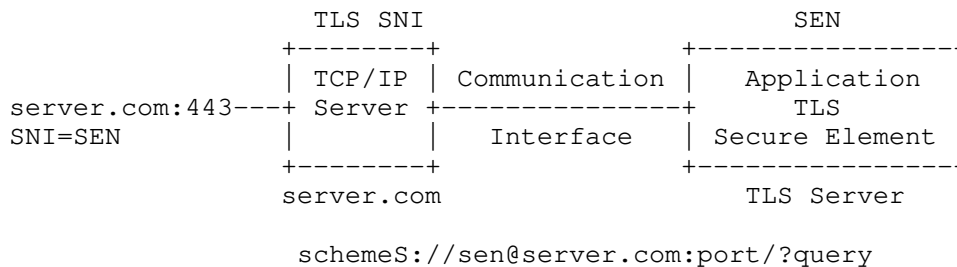
## 1 Overview

This draft defines an infrastructure for the deployment of secure elements over internet, and features needed for their secure remote use.

Secure elements [ISO7816] are tamper resistant micro-controllers, whose security Evaluation Assurance Levels (EAL) are in the range EAL5+/EAL6+ according to Common Criteria standards [CC], which define up to 7 levels.

This draft describes a network architecture based on the TLS 1.3 [RFC8446] protocol, which enables remote calls of cryptographic procedures, identified by Unified Resource Identifier (URI) [RFC3986].

We believe that internet should provide to its users open computing resources, with high security and trust levels. Many applications, such as blockchain, require on-line trusted computing resources, running cryptographic algorithms.



The network architecture comprises the following elements:

- Secure elements, identified by their name (Secure Element Name, SEN) running embedded TLS servers and applications.
- TCP/IP servers, able to parse TLS ClientHello message, in order to extract SNI (Server Name Indication) extension [RFC6066]. If the SNI value matches the SEN value, the TLS packets are routed toward the selected secure element.

The secure element URI [RFC3986] is schemeS://sen@server.com:443/?query, in which:

- scheme indicates the application data interchange format,
- S means secured by TLS,
- sen is the secure element name included in the TLS SNI extension,
- server.com:port is a TCP/IP node and associated port
- query is the command to be executed by the secure element

TLS sessions MUST use mutual authentication between client and server, either based either on pre-shared-key (PSK) or X509 certificates.

The TCP/IP server MAY manage multiple secure elements. As an illustration, according to the IETF draft [RACS] a grid of Secure Elements (GoSE) is a server hosting a set of secure elements.

In summary the Internet of Secure Element (IoSE) is a set of secure elements providing TLS servers, communication interfaces, and identified by their SEN name.

## 2. About Secure Elements

Secure elements are defined according to [ISO7816] standards. Most of them use 8 bits Micro Controller Unit (MCU) and embedded cryptographic accelerator. Non volatile memory size is up to 100KB, and RAM size is up to 10KB. Open software can be written thanks to the JavaCard (JC) programming language, and associated API frameworks such as JC3.04, JC3.05, JC3.1.

Secure elements are dedicated to cryptographic procedures; they are available under multiples physical form factors, such as smartcard, NFC chip, embedded SIM (eSIM), or surface-mount devices.

Secure elements have no network resources. They exchange small messages (up to 256 bytes) over communication interfaces such as ISO7816 (5 wires) [ISO7816], I2C (Inter-Integrated Circuit), or SPI (Serial Peripheral Interface) [GP-SPI-I2C].

Nevertheless they are able to process the TLS 1.3 protocol. For example the IETF draft [TLS-SE] defines segmentation/reassembly mechanisms over ISO7816, which enable exchange of TLS packets with secure elements. The open project [TLS-SE-CODE] is an implementation of [TLS-SE] for javacards. The open project [KEYSTORE-CODE] is an implementation of secure element server. The open project [IOSE-CODE] is a demonstrator for Internet of Secure Elements.

Therefore secure element can be used as host, providing TLS server, and communication interface.

They are several ways to provide a host name for a secure element (i.e. a server name), which is referred as secure element name (SEN) by this draft,:

- The [TLS-SE] draft uses historical bytes (up to 15 bytes) inserted in the ISO7816 ATR (Answer To Reset), which is a response triggered by a physical reset. A javacard application may define the value of historical bytes.

- The [RACS] IETF draft describes Grid of Secure Elements (GoSE), and introduces Secure Element Identifier (SEID) as unique identifier indicating that a given SE is hosted by a GoSE. SEID also implicitly refers the physical slot (SlotID) to which the secure element is plugged. SEID MAY be used as SEN.

### 3. Network Architecture

The network architecture is based on TLS1.3 servers and future versions.

A TCP/IP node manages a server. According to [ESNI] TLS has two working modes, shared and split.

- In Shared Mode, the provider is the origin server for all the domains whose DNS records point to it. In this mode, the TLS connection is terminated by the provider
- In Split Mode, the provider is not the origin server for private domains. Rather, the DNS records for private domains point to the provider, and the provider's server relays the connection back to the origin server, who terminates the TLS connection with the client.

According to this terminology the secure element is the backend server, identified by a server name (referred as SEN).

The client-facing server finds in the ClientHello message required secure element name. Thereafter it performs segmentation/reassembly operations in order to shuttle TLS packet over the communication interface.

The client-facing server MAY also use encrypted server name indication (ESNI) features in order to protect secure elements name.

The application-layer protocol negotiation extension (ALPN) [RFC7301] MAY be used by secure element to select an internal application.

TLS protocol MUST be used with mutual authentication between client and secure element. PSK is a symmetric cryptographic scheme for one client-to-one-secure-element, while PKI is an asymmetric cryptographic scheme adapted to multiple-clients-to-one-secure-element.

Nevertheless it should be noticed that secure elements have not clock and therefore are not able to check validity date or certificate revocation.

#### 4 Unified Resource Identifier (URI)

According to [RFC3986] the URI comprises a scheme name ended by the 'S' character, the secure element name, the client-facing name and port (server.com:port), and a query.

```
URI= schemeS://sen@server.com:port/?query
```

A client software entity able to process this URI, MUST retrieves the PSK or the certificate chain to be used within the TLS protocol.

The secure element name MUST be included in the SNI extension. The used scheme used by the query, MAY be included in the ALPN extension.

For PSK it is possible, but not recommended for security reasons, to include the PSK value in the URI:

```
schemeS://sen:psk@server.com:port/?query
```

#### 5 URI Example

A secure element implements a keystore, of which keys are identified by an index. The secure element name is mykeystore

The secure element name is found in the historical bytes of the ISO7816 ATR.

The client-facing server is server.com:443

The scheme used by the secure element is a shell, i.e. ASCII command lines ended by line feed and carriage return characters.

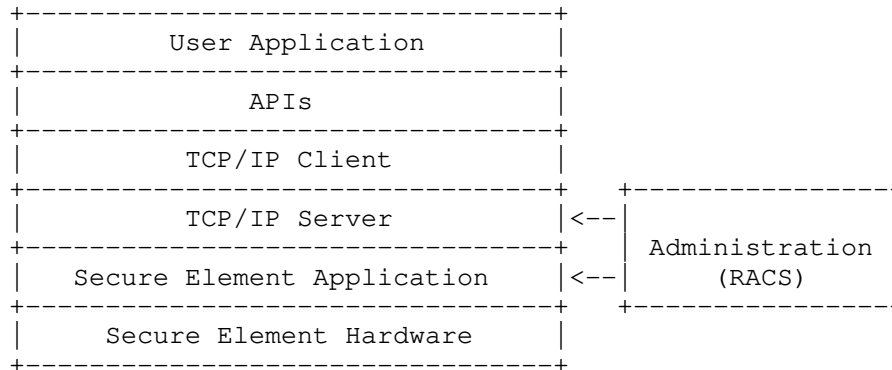
The query  
s010102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F20%0D%0A computes a signature command ('s' prefix) with key of index 01, over the 32 bytes value 0102...1920

The URI is:

```
shellS://mykeystore@server.com:443/?s010102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F20%0D%0A
```

The software client opens a TLS session with the server server.com:443, with the name "mykeystore" inserted the SNI extension. Upon success a TLS secure channel is established with the secure element. The client sends the query, the secure element computes the signature and returns its value encoded in hexadecimal text.

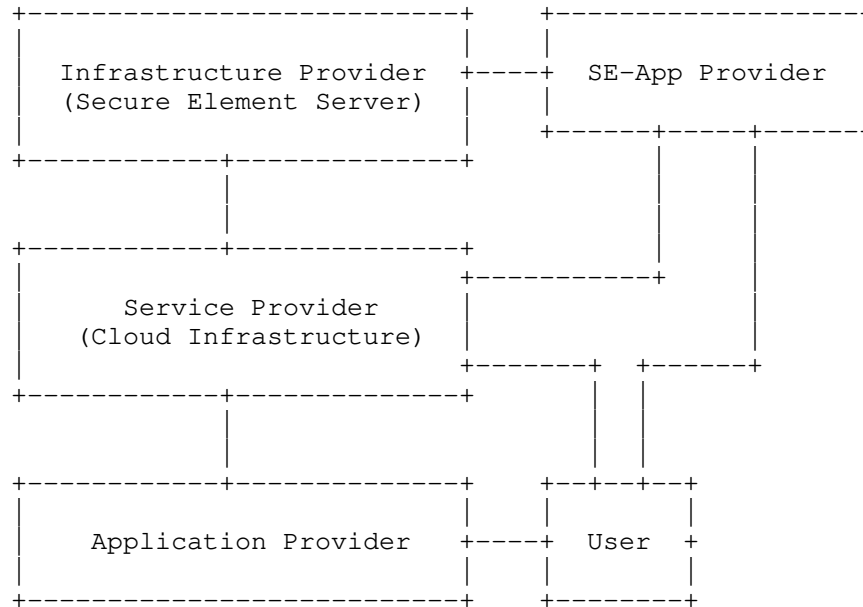
## 6 Overview of Internet Of Secure Elements Framework



The goal of IOSE is to provide to internet users open computing resources, with high security and trust levels. In order to reach this objective, the IOSE framework comprises seven layers.

- The User Application layer uses secure resources hosted in the internet
- The APIs layer provides software interface to virtual resources. It SHOULD provide secure storage of credentials required by TLS sessions.
- The TCP/IP client layer manages TLS session, according to profiles compatible with secure element computing capacities.
- The TCP/IP server layer manages one or several secure elements. It MAY provide privacy features such as server name encryption.
- The secure element application layer defines data interchange format and available procedures
- The secure element hardware layer defines security profile (according to Common Criteria standards) and communication interfaces
- The administration layer is in charge of secure elements application deployment and lifetime. These operations are performed locally or remotely (through the internet).

## 7 Functional Entities

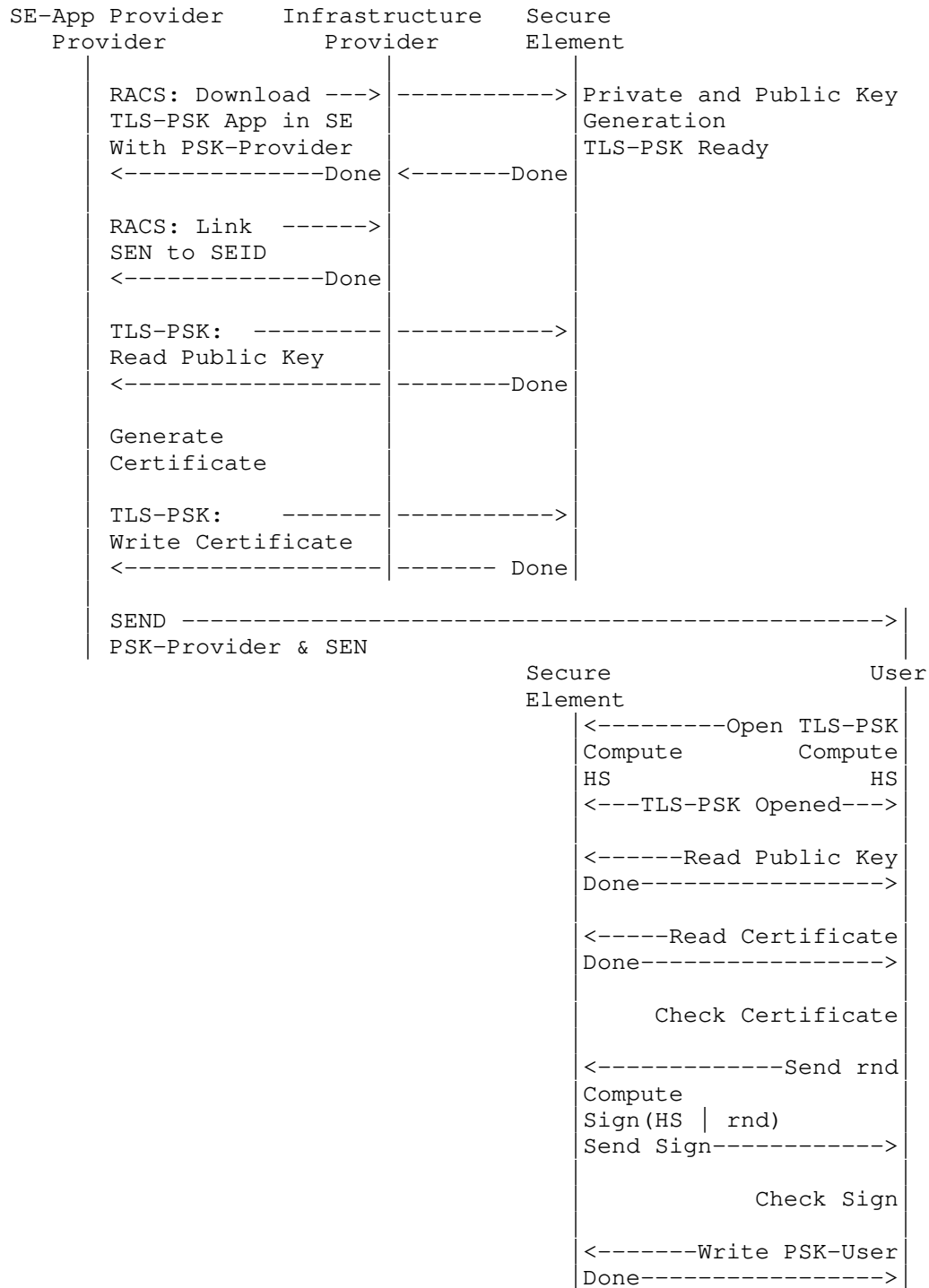


The functional entities COULD involve five elements.

- The User is equipped with a connected device, executing an application using IOSE services.
- The Application Provider (AP) designs software, using IOSE infrastructure.
- The Service Provider (SP) manages a cloud infrastructure, and all facilities needed to setup secure element applications. An attestation mechanism MUST be available in order to prove SE application authenticity.
- The Infrastructure Provider (IP) provides secure element servers.
- The SE-App Provider (SE-AppP) designs secure and trusted software (SE-App) for secure elements.

## 8 Attestation Procedure

The goal of the attestation procedure is to allocate a secure element, and to prove to its user the exclusive access to a genuine secure element.



### 8.1 Service Request

The User requests to an on-line Service Provider a secure element for a specific application.

### 8.2 SE-App Downloading

The Service Provider requests the SE-App Provider to download the user's application in a secure element hosted by an Infrastructure Provider.

The SE-App Provider downloads the selected application thanks to protocols such as [RACS].

There is a mutual authentication between SE-App Provider and Infrastructure Provider. The SE-App Provider owns a set of secure elements identified by their SEID (Secure Element Identifier). He may erase their content and write binary image, but he can't read binary images.

The SE-App includes a TLS 1.3 server with a pre shared key (PSK-App-Provider) and a server name (SEN). Upon instantiation, the downloaded SE-App generates a pair of private key (SE-App-Priv-Key) and public key (SE-App-Pub-key).

Thanks to a dedicated [RACS] command, the SE-APP provider notifies to the Infrastructure Provider the SEN associated to the secure element SEID.

At this step the secure element is on-line, and may process TLS sessions, with the right server name (SEN), and authenticated by PSK-App-Provider.

The PSK-App provider has the exclusive knowledge of the pre-shared-key, and consequently is the only entity able to communicate with the secure element.

### 8.3 SE-App Certificate

The SE-App Provider opens a TLS (with PSK= PSK-App-Provider) session with the SEN secure element, reads its SE-App-Pub-key, and computes a certificate (SE-Cert) for this public key.

The SE-Cert is remotely written in the secure element

The SE-App provider forwards the secure element URI and PSK-App-Provider to the Service Provider or to the User, according to pre-defined agreements.



## 8.4 User Notification

The User receives the secure element URI and pre-shared-key (i.e. PSK-App-Provider).

## 8.5 User Enrollment

A secure element only manages a unique TLS session at a given time.

The User opens a TLS session with the secure element (with PSK-App-Provider and SEN). According to [RFC8446] a TLS handshake secret (HS) is computed from the Diffie-Hellman exchange.

He reads the secure element public key (SE-App-Pub-key) and its certificate (SE-Cert)

He checks the certificate SE-Cert with the Infrastructure Provider public key.

He performs the attestation procedure that sends a random value (RND) to the secure element. The secure element returns a signature of the concatenation of HS and RND values, computed with the private key SE-App-Priv-key.

He checks the signature with the public key SE-App-Pub-key.

At this point the User has the proof that it shares an exclusive TLS session with the secure element.

The user sets the pre-share-key value to new one: PSK-User-Provider.

At this step the User has the exclusive access to a genuine secure element.

## 9 IANA Considerations

This draft does not require any action from IANA.

## 10 Security Considerations

This entire document is about security.

## 11 References

### 11.1 Normative References

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, July 2014

[RFC3986] Berners-Lee, Tim; Fielding, Roy T.; Masinter, Larry. Uniform Resource Identifiers (URI): Generic Syntax. Internet Engineering Task Force. doi:10.17487/RFC3986, January 2005

[ISO7816] ISO 7816, "Cards Identification - Integrated Circuit Cards with Contacts", The International Organization for Standardization (ISO).

[CC] ISO/IEC 15408, "Common Criteria for Information Technology Security Evaluation", The International Organization for Standardization (ISO)

[GP-SPI-I2C] GlobalPlatform Technology, APDU Transport over SPI/I2C Version 0.0.0.39", July 2019

## 11.2 Informative References

[ESNI] "TLS Encrypted Client Hello", draft-ietf-tls-esni-13, 2021

[RACS] "Remote APDU Call Secure (RACS)", draft-urien-core-racs-15.txt, 2021

[TLS-SE] IETF Draft, "Secure Element for TLS Version 1.3", draft-urien-tls-se-03.txt, 2021

[TLS-SE-CODE] "tls-se.java", <https://github.com/purien/TLS-SE>

[KEYSTORE-CODE] <https://github.com/purien/keystore>

[IOSE-CODE] <https://github.com/purien/iose>

## 12 Authors' Addresses

Pascal Urien  
Telecom Paris  
19 place Marguerite Perey  
91120 Palaiseau Phone: NA  
France Email: [Pascal.Urien@telecom-paris.fr](mailto:Pascal.Urien@telecom-paris.fr)

CORE Working Group  
Internet Draft  
Intended status: Experimental

P. Urien  
Telecom Paris

October 2 2021

Expires: April 2022

Remote APDU Call Secure (RACS)  
draft-urien-core-racs-15.txt

## Abstract

This document describes the Remote APDU Call Protocol Secure (RACS) protocol, dedicated to Grid of Secure Elements (GoSE). These servers host Secure Elements (SE), i.e. tamper resistant chips offering secure storage and cryptographic resources.

Secure Elements are microcontrollers whose chip area is about 25mm<sup>2</sup>; they deliver trusted computing services in constrained environments.

RACS supports commands for GoSE inventory and data exchange with secure elements. It is designed according to the representational State Transfer (REST) architecture. RACS resources are identified by dedicated URIs. An HTTP interface is also supported.

An open implementation [OPENRACS] is available (<https://github.com/purien>) for various OS.

## 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.

## Status of this Memo

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

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

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

This Internet-Draft will expire on April 2022.

.

## Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

Abstract.....	1
Requirements Language.....	1
Status of this Memo.....	1
Copyright Notice.....	2
1 Overview.....	5
1.1 What is a Secure Element.....	5
1.2 Grid Of Secure Elements (GoSE).....	6
1.3 Secure Element Identifier (SEID).....	7
1.3.1 SlotID example .....	7
1.3.2 SEID for Secure Elements .....	8
1.4 APDUs.....	9
1.4.1 ISO7816 APDU request .....	9
1.4.2 ISO7816 APDU response .....	9
2 The RACS protocol.....	10
2.1 Structure of RACS request.....	10
2.2 Structure of a RACS response.....	11
2.2.1 BEGIN Header .....	11
2.2.2 END Header .....	11
2.2.3 Status line .....	11
2.2.4 Examples of RACS responses: .....	12
2.3 RACS request commands.....	12
2.3.1 BEGIN .....	12
2.3.2 END .....	12
2.3.3 The APPEND parameter .....	13
2.3.4 GET-VERSION .....	14
2.3.5 SET-VERSION .....	14
2.3.6 LIST .....	15
2.3.7 RESET .....	15
2.3.8 APDU .....	16
2.3.9 SHUTDOWN .....	19
2.3.10 POWERON .....	20
2.3.11 ECHO .....	21
2.3.12 SEN .....	21
2.3.13 GET-SEN .....	23
2.4 Status header encoding.....	24
2.4.1 Event class .....	24
2.4.2 Command class .....	24
3 URI for the GoSE.....	25
4 HTTP interface.....	25
4.1 HTTPS Request.....	25
4.2 HTTPS response.....	26
5 Security Considerations.....	26
5.1 Authorization.....	26
5.2 Secure Element access.....	26
5.3 Applications security policy.....	27
5.3.1 Users-Table .....	27
5.3.2 SEID-Table .....	27
5.3.3 APDU-Table .....	27
5.4 Overview of the security policy.....	28
6 IANA Considerations.....	28

7	References.....	28
	7.1 Normative References.....	28
	7.2 Informative References.....	28
8	Authors' Addresses.....	29

## 1 Overview

This document describes the Remote APDU Call Protocol Secure (RACS) protocol, dedicated to Grids of Secure Elements (GoSE). These servers host Secure Elements (SE), i.e. tamper resistant chips offering secure storage and cryptographic resources.

Secure Elements are microcontrollers whose chip area is about 25mm<sup>2</sup>; they deliver trusted computing services in constrained environments.

RACS supports commands for GoSE inventory and data exchange with secure elements.

RACS is designed according to the representational State Transfer (REST) architecture [REST], which encompasses the following features:

- Client-Server architecture.
- Stateless interaction.
- Cache operation on the client side.
- Uniform interface.
- Layered system.
- Code On Demand.

### 1.1 What is a Secure Element

A Secure Element (SE) is a tamper resistant microcontroller equipped with host interfaces such as [ISO7816], SPI (Serial Peripheral Interface) or I2C (Inter Integrated Circuit).

The typical area size of these electronic chips is about 25mm<sup>2</sup>. They comprise CPU (8, 16, 32 bits), ROM (a few hundred KB), nonvolatile memory (EEPROM, FLASH, a few hundred KB) and RAM (a few ten KB). Security is enforced by multiple hardware and logical countermeasures.

According to the [EUROSMART] association height billion of such secure devices were shipped in 2013. Secure elements are widely deployed for electronic payment (EMV cards), telecommunication (SIM modules), identity (electronic passports), ticketing, and access control.

Most of secure elements include a Java Virtual Machine and therefore are able to execute embedded program written in the JAVACARD language. Because these devices are dedicated to security purposes they support numerous cryptographic resources such as digest functions (MD5, SHA1, SHA2...), symmetric cipher (3xDES, AES) or asymmetric procedures (RSA, ECC).

A set of Global Platform [GP] standards control the lifecycle of embedded software, i.e. application downloading, activation and deletion.

As an illustration a typical Secure Element has the following characteristics:

- JAVACARD operating system;
- Compliant with the GP (Global Platform) standards;
- 160 KB of ROM;
- 72 KB of EEPROM;
- 4KB of RAM;
- Embedded crypto-processor;
- 3xDES, AES, RSA, ECC;
- Certification according to Common Criteria (CC) EAL5+ level;
- Security Certificates from payment operators.

### 1.2 Grid Of Secure Elements (GoSE)

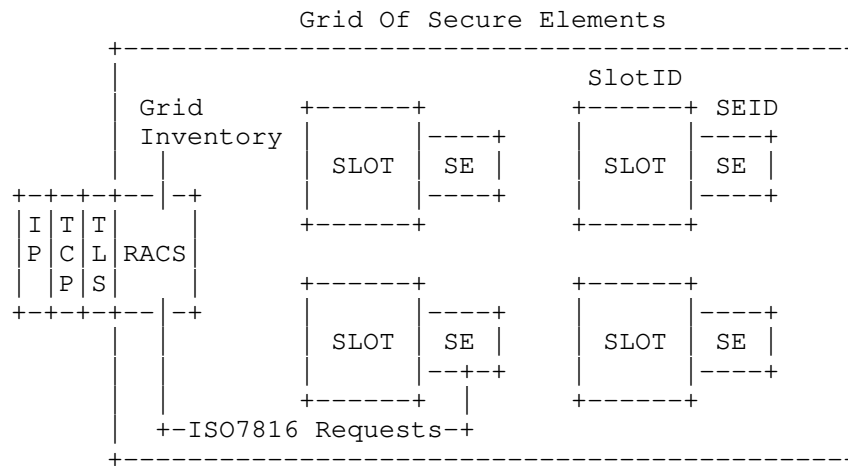


Figure 1. Architecture of a Grid of Secure Elements

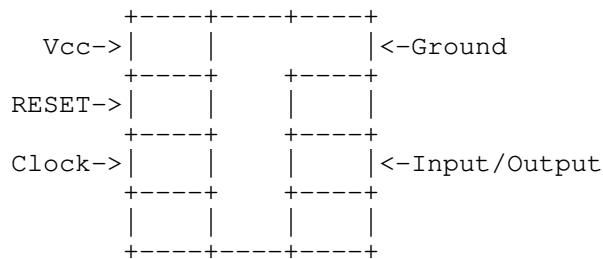


Figure 2. Illustration of an ISO7816 Secure Element

A grid of Secure Elements (GoSE) is a server hosting a set of secure elements.



The goal of these platforms is to deliver trusted services over the Internet. These services are available in two functional planes,

- The user plane, which provides trusted computing and secure storage.
- The management plane, which manages the lifecycle (downloading, activation, deletion) of applications hosted by the Secure Element.

A grid of Secure Elements offers services similar to HSM (Hardware Secure Module), but may be managed by a plurality of administrators, dealing with specific secure microcontrollers.

According to this draft all accesses to a GoSE require the TCP transport and are secured by the TLS [TLS 1.0] [TLS 1.1] [TLS 2.0] protocol.

The RACS protocol provides all the features needed for the remote use of secure elements, i.e.

- Inventory of secure elements
- Information exchange with the secure elements

### 1.3 Secure Element Identifier (SEID)

Every secure element needs a physical slot that provides electrical feeding and communication resources. This electrical interface is for example realized by a socket soldered on an electronic board, or a CAD (Card Acceptance Device, i.e. a reader) supporting host buses such as USB.

Within the GoSE each slot is identified by a SlotID (slot identifier) attribute, which may be a socket number or a CAD name.

The SEID (Secure Element Identifier) is a unique identifier indicating that a given SE is hosted by a GoSE. It also implicitly refers the physical slot (SlotID) to which the SE is plugged.

The GoSE manages an internal table that establishes the relationship between SlotIDs and SEIDs.

Therefore three parameters are needed for remote communication with secure element, the IP address of the GoSE, the associated TCP port, and the SEID.

#### 1.3.1 SlotID example

According to the PC/SC (Personal Computer/Smart Card) standard [PS/SC], a smart card reader MAY include a serial number. This attribute (VENDOR-IFD-SERIAL) is associated to the tag 0x0103 in the class VENDOR-INFO.

### 1.3.2 SEID for Secure Elements

According to the Global Platform standard [GP] the Issuer Security Domain (ISD) manages applications lifecycle (downloading, activation, deletion). The command 'initialize update' is used to start a mutual authentication between the administration entity and the secure element; it collects a set of data whose first ten bytes are called the 'key diversification data'. This information is used to compute symmetric keys, and according for example to [EMV] MAY comprise a serial number.

## 1.4 APDUs

According to the [ISO7816] standards secure element process ISO7816 request messages and return ISO7816 response messages, named APDUs (application protocol data unit).

### 1.4.1 ISO7816 APDU request

An APDU request comprises two parts: a header and an optional body.

The header is a set of four or five bytes noted CLA INS P1 P2 P3

- CLA indicates the class of the request, and is usually bound to standardization committee (00 for example means ISO request).
- INS indicates the type of request, for example B0 for reading or D0 for writing.
- P1 P2 gives additional information for the request (such index in a file or identifier of cryptographic procedures)
- P3 indicates the length of the request body (from P3=01 to P3=FF), or the size of the expected response body (a null value meaning 256 bytes). Short ISO7816 requests may comprise only 4 bytes
- The body may be empty. Its maximum size is 255 bytes

### 1.4.2 ISO7816 APDU response

An APDU response comprises two parts an optional body and a mandatory status word.

- The optional body is made of 256 bytes at the most.
- The response ends by a two byte status noted SW. SW1 refers the most significant byte and SW2 the less significant byte.

An error free operation is usually associated to the 9000 status word. Following are some interpretations of the tuple SW1, SW2 according to various standards:

- '61' 'xx', indicates that xx bytes (modulus 256) are ready for reading. Operation result MUST be fetched by the ISO Get Response APDU (CLA=00, INS=C0, P1=P2=00, P3=XX)
- '9F' 'xx', indicates that xx bytes (modulus 256) are ready for reading. Operation result MUST be fetched by the ISO Get Response APDU (CLA=00, INS=C0, P1=P2=00, P3=XX)
- '6C' 'XX', the P3 value is wrong, request must be performed again with the LE parameter value sets to 'XX'
- '6E' 'XX', wrong instruction class (CLA) given in the request
- '6D' 'XX', unknown instruction code (INS) given in the request
- '6B' 'XX', incorrect parameter P1 or P2
- '67' 'XX', incorrect parameter P3
- '6F' 'XX', technical problem, not implemented...

## 2 The RACS protocol

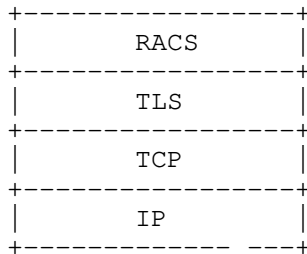


Figure 2. The RACS stack

The RACS protocol works over the TCP transport layer and is secured by the TLS protocol. The TLS client (i.e. the RACS client) MUST be authenticated by a certificate.

One of the main targets of the RACS protocol is to efficiently push a set of ISO7816 requests towards a secure element in order to perform cryptographic operations in the user's plane. In that case a RACS request typically comprises a prefix made with multiple ISO7816 requests and a suffix that collects the result of a cryptographic procedure.

The mandatory use of TLS with mutual authentication based on certificate provides a simple and elegant way to establish the credentials of a RACS client over the GoSE. It also enables an easy splitting between users' and administrators' privileges.

### 2.1 Structure of RACS request

A RACS request is a set of command lines, encoded according to the ASCII format. Each line ends by the Cr (carriage return) and line feed (Lf) characters. The RACS protocol is case sensitive.

Each command is a set of tokens (i.e. words) separated by space (0x20) character(s).

The first token of each line is the command to be executed.

A command line MAY comprise other tokens, which are called the command parameters.

A RACS request MUST start by a BEGIN command and MUST end by an END command.

Each command line is associated to an implicit line number. The BEGIN line is associated to the zero line number.

The processing of a RACS request is stopped after the first error. In that case the returned response contained the error status induced by the last executed command.

## 2.2 Structure of a RACS response

A RACS response is a set of lines, encoded according to the ASCII format. Each line ends by the Cr (carriage return) and line feed (Lf) characters. The RACS protocol is case sensitive.

Each line is a set of tokens (i.e. words) separated by space (0x20) character(s).

The first token of each line is the header.

The second token of response each line is associated command line number

A response line MAY comprise other tokens, which are called the response parameters.

Three classes of headers are defined BEGIN, END and Status.

A RACS response MUST start by a BEGIN header and MUST end by an END header. It comprises one or several status lines.

### 2.2.1 BEGIN Header

This header starts a response message.

It comprises an optional parameter, an identifier associated to a previous request message.

### 2.2.2 END Header

This header ends a response message.

### 2.2.3 Status line

A status header indicates a status line.

It begins by the character '+' in case of success or '-' if an error occurred during the RACS request execution. It is followed by an ASCII encoded integer, which is the value of the status.

The second mandatory token of a status line is the command line number (starting from zero)

A status line MAY comprise other tokens, which are called the response parameters.

#### 2.2.4 Examples of RACS responses:

```
BEGIN CrLf
+001 000 Success CrLf
END CrLf
```

```
BEGIN moon1969 CrLf
-301 007 Illegal command, BEGIN condition not satisfied at line 7
END CrLf
```

```
BEGIN Asterix237 CrLf
+006 001 [ISO7816-Response] CrLf
END CrLf
```

```
BEGIN CrLf
-100 002 Unknown command at line 2 CrLf
END CrLf
```

```
BEGIN CrLf
-606 001 Unauthorized command APDU command at line 1
END CrLf
```

```
BEGIN CrLf
-706 001 SEID Already in use, APDU command at line 1
END CrLf
```

### 2.3 RACS request commands

#### 2.3.1 BEGIN

This command starts a request message. A response message is returned if an error is detected.

An optional parameter is the request identifier, which MUST be echoed in the parameter of the first response line (i.e. starting by the BEGIN header).

#### 2.3.2 END

This command ends a request message. It returns the response message triggered by the last command.

## Example1

=====

## Request:

BEGIN CrLf

END CrLf

## Response:

BEGIN CrLf

+001 000 Success CrLf

END CrLf

## Example2

=====

## Request:

BEGIN Marignan1515 CrLf

APDU ASTERIX-CRYPTO-MODULE [ISO7816-Request] CrLf

END CrLf

## Response:

BEGIN Marignan1515 CrLf

+006 001 [ISO7816-Response] CrLf

END CrLf

## 2.3.3 The APPEND parameter

The APPEND parameter MAY be used in all command lines, excepted BEGIN and END. The APPEND parameter MUST be the last parameter of a command line.

By default a response message returns only the last status line. When APPEND is inserted, the command line, if executed, MUST produce a status line.

## Example

## Request:

BEGIN SanchoPanza CrLf

APDU 100 [ISO7816-Request-1] CrLf

APDU 100 [ISO7816-Request-2] CrLf

END CrLf

## Response:

BEGIN SanchoPanza CrLf

+006 002 [ISO7816-Response-2] CrLf

END CrLf

## Request:

BEGIN DonQuichotte CrLf

APDU 100 [ISO7816-Request-1] APPEND CrLf

APDU 100 [ISO7816-Request-2] APPEND CrLf

END CrLf

## Response:

```
BEGIN DonQuichotte CrLf
+006 001 [ISO7816-Response-1] CrLf
+006 002 [ISO7816-Response-2] CrLf
END CrLf
```

## 2.3.4 GET-VERSION

This command requests the current version of the RACS protocol. The returned response is the current version encoded by two integer separated by the '.' character. The first integer indicates the major version and the second integer gives the minor version.

This draft version is 0.2

## Example

=====

## Request:

```
BEGIN CrLf
GET-VERSION CrLf
END CrLf
```

## Response:

```
BEGIN CrLf
+002 001 1.0 CrLf
END CrLf
```

## 2.3.5 SET-VERSION

This command sets the version to be used for the RACS request. An error status is returned by the response if an error occurred.

## Example 1

=====

## Request:

```
BEGIN CrLf
SET-VERSION 2.0 CrLf
END CrLf
```

## Response:

```
BEGIN CrLf
-403 001 Error line 1 RACS 2.0 is not supported CrLf
END CrLf
```

## Example 2

=====

## Request:

```
BEGIN CrLf
SET-VERSION 1.0 CrLf
END CrLf
```



```
Response:
BEGIN CrLf
+003 001 RACS 1.0 has been activated CrLf
END CrLf
```

### 2.3.6 LIST

This command requests the list of SEID plugged in the GoSE.

It returns a list of SEIDs separated by space (0x20) character(s).

Some SEID attributes MAY be built from a prefix and an integer suffix (such as SE#100 in which SE# is the suffix and 100 is the integer suffix. A list of non-consecutive SEID MAY be encoded as prefix[i1;i2;...;ip] where i1,i2,ip indicates the integer suffix. A list of consecutive SEID could be encoded as prefix[i1-ip] where i1,i2,ip indicates the integer suffix.

#### Example 1

=====

```
Request:
BEGIN CrLf
LIST CrLf
END CrLf
```

```
Response:
BEGIN CrLf
+004 001 SEID1 SEID2 CR LF
END CrLf
```

#### Example 2

=====

```
Request:
BEGIN CrLf
LIST CrLf
END CrLf
```

```
Response:
BEGIN CrLf
+004 001 Device[1000-2000] SerialNumber[567;789;243] CrLf
END CrLf
```

### 2.3.7 RESET

This command resets a secure element. The first parameter gives the secure element identifier (SEID). An optional second parameter specifies a warm reset. The default behavior is a cold reset. The response status indicates the success or the failure of this operation.

Syntax: RESET SEID [WARM] CrLf

Example 1

=====

Request:

BEGIN CrLf

RESET device#45 CrLf

END CrLf

Response:

BEGIN CrLf

+005 001 device#45 Reset Done

END CrLf

Example 2

=====

Request:

BEGIN CrLf

RESET device#45 CrLf

END CrLf

Response:

BEGIN CrLf

-705 001 error device#45 is already in use

END CrLf

Example 3

=====

Request:

BEGIN CrLf

RESET device#45 WARM CrLf

END CrLf

Response:

BEGIN CrLf

+005 001 device#45 Warm Reset Done CrLf

END CrLf

### 2.3.8 APDU

This command sends an ISO7816 request to a secure element or a set of ISO7816 commands.

The first parameter specifies the SEID.

The second parameter is an ISO7816 request.

Three optional parameters are available; they MUST be located after the second parameter.

- CONTINUE=value, indicates that the next RACS command will be executed only if the ISO7816 status word (SW) is equal to a given value. Otherwise an error status is returned.
- MORE=value, indicates that a FETCH request will be performed (i.e. a new ISO7816 request will be sent) if the first byte of the ISO7816 status word (SW1) is equal to a given value.
- FETCH=value fixes the four bytes of the ISO7816 FETCH request (i.e. CLA INS P1 P2). The default value (when FETCH is omitted) is 00C00000 (CLA=00, INS=C0, P1=00, P2=00)

When the options CONTINUE and MORE are simultaneously set the SW1 byte is first checked. If there is no match then the SW word is afterwards checked.

The ISO7816 6Cxx status MUST be autonomously processed by the GoSE.

#### SYNTAX

APDU SEID ISO7816-REQUEST [CONTINUE=SW] [MORE=SW1] [FETCH=CMD] CrLf

The returned response is the ISO7816 response. If multiple ISO7816 requests are executed (due to the MORE option), the bodies are concatenated in the response, which ends by the last ISO7816 status word.

The pseudo code of the APDU command is the following :

```

1. BODY = empty;
2. SW   = empty;
3. DoIt = true;
3. Do
4. { iso7816-response = send(iso7816-request);
5.   body || sw1 || sw2 = iso7816-response;
6.   If ( (first request) && (iso7816-request.size==5) &&
        (body==empty) && (sw1==6C) )
7.   { iso7816-request.P3 = sw2 ; }
6.   Else
7.   { SW = sw1 || sw2
8.     BODY = BODY || body;
9.     If (sw1 == MORE)
10.    { iso7816-request = FETCH || sw2 ; }
11.    Else
12.    { DoIt=false; }
13.  }
14. }
15. While (DoIt == true)

16. iso7816-response = BODY || SW ;
17. If (SW != CONTINUE) Error ;
18. Else
    No Error;
```

## Example 1

=====

## Request:

BEGIN CrLf

APDU SEID ISO7816-REQUEST CrLf

END CrLf

## Response:

BEGIN CrLf

+006 001 ISO7816-RESPONSE CrLf

END CrLf

## Example 2

=====

## Request:

BEGIN CrLf

APDU SEID ISO7816-REQUEST CrLf

END CrLf

## Response:

BEGIN CrLf

-706 001 error SEID is already used CrLf

END CrLf

## Example 3

=====

## Request:

BEGIN CrLf

APDU SEID ISO7816-REQUEST CrLf

END CrLf

## Response:

BEGIN CrLf

-606 001 error access unauthorized access CrLf

END CrLf

## Example 4

=====

BEGIN CrLf

APDU SEID ISO7816-REQUEST-1 CONTINUE=9000 CrLf

APDU SEID ISO7816-REQUEST-2 CrLf

END CrLf

## Response:

BEGIN CrLf

+006 002 ISO7816-RESPONSE-2 CrLf

END CrLf

## Example 5

=====

```
BEGIN CrLf
APDU SEID ISO7816-REQUEST-1 CONTINUE=9000 CrLf
APDU SEID ISO7816-REQUEST-2 CrLf
END CrLf
```

## Response:

```
BEGIN CrLf
-006 001 Request Error line 1 wrong SW CrLf
END CrLf
```

## Example 6

=====

```
BEGIN CrLf
APDU SEID ISO7816-REQ-1 CONTINUE=9000 CrLf
APDU SEID ISO7816-REQ-2 CONTINUE=9000 CrLf
APDU SEID ISO7816-REQ-3 CONTINUE=9000 MORE=61 FETCH=00C00000 CrLf
END CrLf
```

## Response:

```
BEGIN CrLf
+006 003 ISO7816-RESP-3 CrLf
END CrLf
```

Multiple ISO7816 requests have been performed by the third APDU command according to the following scenario :

- the ISO7816-REQ-3 request has been forwarded to the secure element (SEID)
- the ISO 7816 response comprises a body (body-0) and a status word (SW-0) whose first byte is 0x61, and the second byte is SW2-0
- the FETCH command CLA=00, INS=00, P1=00, P2=00, P3=SW2-0 is sent to the secure element
- the ISO 7816 response comprises a body (body-1) and a status word (SW-1) set to 9000

The RACS response is set to  
 +006 003 body-0 || body-1 || SW-1 CrLf  
 where || indicates a concatenation operation.

## 2.3.9 SHUTDOWN

This command powers down a secure element. The first parameter gives the secure element identifier (SEID).

Syntax: SHUTDOWN SEID CrLf

## Example

=====

## Request:

```
BEGIN Goodbye CrLf
SHUTDOWN device#45 CrLf
END CrLf
```

## Response:

```
BEGIN Goodbye CrLf
+007 001 device#45 has been powered down CrLf
END CrLf
```

## 2.3.10 POWERON

This command powers up a secure element. The first parameter gives the secure element identifier (SEID).

Syntax: POWERON SEID CrLf

## Example 1

=====

## Request:

```
BEGIN CrLf
POWERON device#45 CrLf
END CrLf
```

## Response:

```
BEGIN CrLf
+008 001 device#45 Has been powered up CrLf
END CrLf
```

## Example 2

=====

## Request:

```
BEGIN CrLf
POWERON device#45 CrLf
END CrLf
```

## Response:

```
BEGIN CrLf
-708 001 error device#45 is already in use CrLf
END CrLf
```

## Example 3

=====

## Request:

```
BEGIN CrLf
POWERON device#45 CrLf
END CrLf
```

Response:  
BEGIN CrLf  
-608 001 error unauthorized access CrLf  
END CrLf

### 2.3.11 ECHO

This command echoes a token. The first parameter is the token (word) to be echoed by the response.

Syntax: ECHO SEID CrLf

Example 1  
=====

Request:  
BEGIN TestEcho CrLf  
ECHO Hello CrLf  
END CrLf

Response:  
BEGIN TestEcho CrLf  
+009 001 Hello CrLf  
END CrLf

Example 2  
=====

Request:  
BEGIN ResetSEID CrLf  
POWERON device#45 CrLf  
ECHO Done CrLf  
END CrLf

Response:  
BEGIN ResetSEID CrLf  
+009 001 Done CrLf  
END CrLf

### 2.3.12 SEN

This command associates Secure Element Name (SEN) to SEID. Secure Element Name are defined in [IOSE]

The first parameter (mandatory) is the SEID. By default the SEN is found in the ISO7816 ATR, and the TLS-SE application is the secure element default application.

The second parameter (optional) is the SEN. This option sets the SEN, and discards the ATR content.

The third parameter (optional) is the TLS-SE Application Identifier (AID).

Syntax: SEN SEID [SEN] [AID] CrLf

Example 1  
=====

Request:  
BEGIN CrLf  
SEN mySEID CrLf  
END CrLf

Response:  
BEGIN CrLf  
+010 001 SEN= key1.com AID= default  
END CrLf

Example 2  
=====

Request:  
BEGIN CrLf  
SEN mySEID key1.com CrLf  
END CrLf

Response:  
BEGIN CrLf  
+010 001 SEN= key1.com AID= default CrLf  
END CrLf

Example 3  
=====

Request:  
BEGIN CrLf  
SEN mySEID key1.com 010203040500 CrLf  
END CrLf

Response:  
BEGIN CrLf  
+010 001 SEN= key1.com AID= 010203040500 CrLf  
END CrLf

Example 4  
=====



Request:  
BEGIN CrLf  
SEN wrongSEID key1.com CrLf  
END CrLf

Response:  
BEGIN CrLf  
-410 001 SEN invalid SEID (wrongSEID) CrLf  
END CrLf

### 2.3.13 GET-SEN

This command gets Secure Element Name (SEN) associated to SEID.  
Secure Element Name are defined in [IOSE]

Syntax: GET-SEN SEID CrLf

#### Example 1 =====

Request:  
BEGIN CrLf  
GET-SEN mySEID CrLf  
END CrLf

Response:  
BEGIN CrLf  
+011 001 key1.com [AID= default]  
END CrLf

#### Example 2 =====

Request:  
BEGIN CrLf  
GET-SEN mySEID CrLf  
END CrLf

Response:  
BEGIN CrLf  
+011 001 key1.com [AID= 010203040500]  
END CrLf

#### Example 3 =====

Request:  
BEGIN CrLf

```
GET-SEN wrongSEID CrLf
END CrLf
```

```
Response:
BEGIN CrLf
-511 001 GET-SEN invalid SEID (wrongSEID)
END CrLf
```

## 2.4 Status header encoding

The first token of a response line is the status header. It begins by a '+' or a '-' character, and comprises three decimal digits (xyz).

The first digit (x) MUST indicate an event class.  
The second and third digits (yz) MAY indicate a command class.

### 2.4.1 Event class

This draft only defines the meaning of the first digit located at the left most side.

```
+0yz: No error
-0yz: Command execution error
-1yz: Unknown command, the command is not defined by this draft
-2yz: Not implemented command
-3yz: Illegal command, the command can't be executed
-4yz: Not supported parameter or parameter illegal value
-5yz: Parameter syntax error or parameter missing
-6yz: Unauthorized command
-7yz: Already in use, a session with this SE is already opened
-8yz: Hardware error
-9yz: System error
```

### 2.4.2 Command class

The second and third digits (yz) MAY indicate the command that triggered the current line status

```
01 BEGIN
02 GET-VERSION
03 SET-VERSION
04 LIST
05 RESET
06 APDU
07 SHUTDOWN
08 POWERON
09 ECHO
10 SEN
11 GET-SEN
```

### 3 URI for the GoSE

The URI addressing the resources hosted by the GoSE is represented by the string:

`RACS://GoSE-Name:port/?request`

where request is the RACS request to be forwarded to a the GoSE.

RACS command lines are encoded in a way similar to the INPUT field of an HTML form. Each command is associated to an INPUT name, the remaining of the command line i.e. a set of ASCII characters, is written according to the URL encoding rules. End of line characters, i.e. carriage return (Cr) and line feed (Lf) are omitted.

As a consequence a request is written to the following syntax  
`cmd1=cmd1-parameters&cmd2=cmd2-parameters`

Example:

`RACS://GoSE-Name:port/?BEGIN=&APDU=SEID%20[ISO7816-REQUEST]&END=`

### 4 HTTP interface

A GoSE SHOULD support an HTTP interface. RACS requests/responses are transported by HTTP messages. The use of TLS is mandatory.

#### 4.1 HTTPS Request

`https://GoSE-Name:port/RACS?request`

where request is the RACS request to be forwarded to a secure element (SEID)

The RACS request is associated to an HTML form whose name is "RACS". The request command lines are encoded as the INPUT field of an HTML form. Each command is associated to an INPUT name, the remaining of the command line i.e. a set of ASCII characters is written according to the URL encoding rules. End of line characters, i.e. carriage return (Cr) and line feed (Lf) are omitted.

As a consequence a RACS request is written as  
`https://GoSE-Name/RACS?cmd1=cmd1-parameters&cmd2=cmd2-parameters`

Example:

`https://GoSE-Name/RACS?BEGIN=&APDU=SEID%20[ISO7816-REQUEST]&END=`

## 4.2 HTTPS response

The RACS response is returned in an XML document.

The root element of the document is <RACS-Response>

The optional parameter of the BEGIN header, is the content of the <begin> element.

Each status line is the content of the <Cmd-Response> element, which includes the following information :

- The status header is the content of the <status> element.
- The line number is the content of the <line> element.
- The other parameters of the status line are the content of the <parameters> element.

The END header is associated to the element <end>

End of line, i.e. carriage return (Cr) and line feed (Lf) characters are omitted.

As a consequence a RACS response is written as :

```
<RACS-Response>
<begin>Optionnal-ID</begin>
<Cmd-Response
<status>+000</status>
<line>001</line>
<parameters>other parameters of the RACS response</parameters>
</Cmd-Response>
<end></end>
</RACS-Response>
```

## 5 Security Considerations

### 5.1 Authorization

A RACS client MUST be authenticated by an X509 certificate.

The GoSE software MUST provide a mean to establish a list of SEIDs that can be accessed from a client whose identity is the CommonName (CN) attribute of its certificate. It MAY allocate a UserID (UID), i.e. an integer index from the certificate common name.

### 5.2 Secure Element access

The GoSE MUST manage a unique session identifier (SID) for each TLS session. The SID is bound to the client's certificate CommonName (SID(CN))

A secure element has two states, unlocked and locked. In the locked state the secure element may be only used by the SID that previously locked it.

The first authorized command that successfully accesses to a SEID (either POWERON ,RESET, APDU) locks a secure element (SEID) with the current session (SID).

The SHUTDOWN command MUST unlock a secure element (SEID).

The end of a TLS session MUST unlock all the secure elements locked by the session.

### 5.3 Applications security policy

According to the [ISO7816] standards each Application embedded within a secure element (associated to a SEID) is identified by an AID parameter (16 bytes at the most)

The RACS server SHOULD support the following facilities

#### 5.3.1 Users-Table

Each CN (the Users-Table primary key) is associated to a list of SEIDs whose access is authorized.

#### 5.3.2 SEID-Table

Each AID (the SEID-Table primary key) is associated to a list of CNs whose access is authorized.

#### 5.3.3 APDU-Table

For a given AID and an authorized CN, an APDU-Table MAY be available. This table acts as a firewall, which defined a set of forbidden ISO7816 commands.

For example this filter could be expressed as a set of the four first bytes of an APDU-Prefix (CLA INS P1 P2) and a four bytes Mask  
An ISO7816-Request is firewall if:

ISO7816-Request AND Mask IsEQUAL to APDU-Prefix

## 5.4 Overview of the security policy

The summary of the security policy is illustrated by the figure 3.

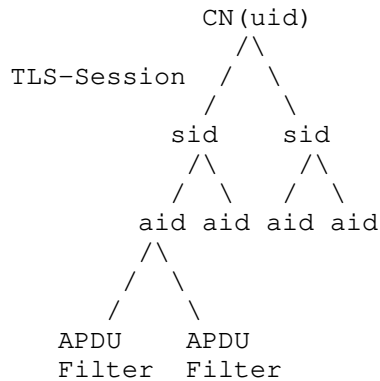


Figure 3. Summary of the security policy

## 6 IANA Considerations

This draft does not require any action from IANA.

## 7 References

### 7.1 Normative References

[TLS 1.0] Dierks, T., C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999

[TLS 1.1] Dierks, T., Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006

[TLS 1.2] Dierks, T., Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5746, August 2008

[TLS 1.3] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, August 2018

[ISO7816] ISO 7816, "Cards Identification - Integrated Circuit Cards with Contacts", The International Organization for Standardization (ISO)

### 7.2 Informative References

[REST] Fielding, R., "Architectural Styles and the Design of Network-based Software Architectures", 2000, <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>

[GP] Global Platform Standards, <http://www.globalplatform.org>

[EUROSMART] The EUROSMART association, <http://www.eurosmart.com>

[PC/SC] The PC/SC workgroup, <http://www.pcscworkgroup.com>

[EMV] EMV Card Personalization Specification, Version 1.1, July 2007

[OPENRACS] <https://github.com/purien>, open RACS implementation for Win32, Ubuntu, Raspberrypi

[IOSE] Internet of Secure Elements, draft-urien-coinrg-iose-03.txt, September 2021

## 8 Authors' Addresses

Pascal Urien  
Telecom Paris  
19 place Marguerite Perey  
91120 Palaiseau                      Phone: NA  
France                                      Email: [Pascal.Urien@telecom-paris.fr](mailto:Pascal.Urien@telecom-paris.fr)

TLS Working Group  
Internet Draft  
Intended status: Experimental

P. Urien  
Telecom Paris

July 26 2021

Expires: January 2022

Identity Module for TLS Version 1.3  
draft-urien-tls-im-05.txt

## Abstract

TLS 1.3 will be deployed in the Internet of Things ecosystem. In many IoT frameworks, TLS or DTLS protocols, based on pre-shared key (PSK), are used for device authentication. So PSK tamper resistance, is a critical market request, in order to prevent hijacking issues. If DH exchange is used with certificate bound to DH ephemeral public key, there is also a benefit to protect its signature procedure. The TLS identity module (im) MAY be based on secure element; it realizes some HKDF operations bound to PSK, and cryptographic signature if certificates are used. Secure Element form factor could be standalone chip, or embedded in SoC like eSIM.

## 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.

## Status of this Memo

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

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

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

This Internet-Draft will expire on January 2022.

.



## Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

Abstract.....	1
Requirements Language.....	1
Status of this Memo.....	1
Copyright Notice.....	2
1 Overview.....	5
2 Protecting the Key Schedule for PSK.....	5
2.1 Context.....	5
2.2 Identity Module Procedures.....	6
2.3 KSGS: Keys Secure Generation and Storage.....	6
2.4 Identity Module Key Procedures (IMKP).....	6
2.4.1 CETS: Client Early Traffic Secret .....	6
2.4.2 EEMS: Early Exporter Master Secret .....	7
2.4.3 HEDSK: HKDF-Extract from Derived Secret Key .....	7
2.4.4 HBSK: HMAC from Binder Key Secret .....	7
3. Asymmetric Signature.....	7
3.1 GENKEY.....	8
3.2 GETPUB.....	8
3.3 SIGN.....	8
4 Optional Procedures.....	8
4.1 GENDHE.....	8
4.2 GETEPK.....	8
4.3 RAND.....	8
5 Identity Module Procedures Summary.....	9
6. Secure Element as Identity Module.....	10
6.1 Administrator mode.....	10
6.2 User Mode.....	10
6.3 KSGS: Keys Secure Generation and Storage.....	10
6.3.1 Example .....	11
6.4 CETS: Client Early Traffic Secret.....	11
6.4.1 Example .....	11
6.5 EEMS: Early Exporter Master Secret.....	11
6.5.1 Example .....	12
6.6 HEDSK: HKDF-Extract from Derived Secret Key.....	12
6.6.1 Example .....	12
6.7 HBSK: HMAC from Binder Key Secret.....	12
6.7.1 Example .....	12
6.8 Signature Procedures.....	12
6.8.1 Keys Generation .....	12
6.8.2 Keys Setting .....	13
6.8.3 Signature .....	14
6.9 GENDHE.....	14
6.9.1 Example .....	14
6.10 GETEPK.....	14
6.10.1 Example .....	14
6.11 RAND.....	14
6.11.1 Example .....	15
7. A simple Identity Module code for Javacard 3.04.....	15
8 IANA Considerations.....	32
9 Security Considerations.....	32
10 References.....	32

10.1 Normative References.....	32
10.2 Informative References.....	32
11 Authors' Addresses.....	32

## 1 Overview

TLS 1.3 [RFC8446] will be deployed in the Internet of Things ecosystem. In many IoT frameworks, TLS or DTLS protocols, based on pre-shared key (PSK), are used for device authentication. So PSK tamper resistance, is a critical market request, in order to prevent hijacking issues. If DH exchange is used with certificate bound to DH ephemeral public key, there is also a benefit to protect its signature procedure. The TLS identity module (im) MAY be based on secure element [ISO7816]; it realizes some HKDF [RFC5869] operations bound to PSK, and cryptographic signature if certificates are used. Secure Element form factor could be standalone chip or embedded in SOC like eSIM.

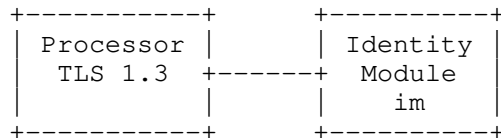


Figure 1. TLS 1.3 Identity Module (im)

The ISO7816 standards specify the binary encoding for ISO7816-4 commands and responses, refereed as Application Protocol Data Unit (APDU). APDUs can be exchanged with secure elements according to various transport protocols [GP-SPI-I2C] such as ISO7816-3 T=0, ISO7816-3 T=1, Inter Integrated Circuit (I2C) or Serial Peripheral Interface (SPI)

## 2 Protecting the Key Schedule for PSK

### 2.1 Context

According to [RFC8446] external PSKs MAY be provisioned outside of TLS.

The Early Secret (ESK) is computed according to relation:  

$$\text{ESK} = \text{HKDF-Extract}(\text{salt}=0s, \text{PSK}) = \text{HMAC}(\text{salt}=0s, \text{PSK})$$

The Binder Key (BSK) for outside provisioning is computed according to the relation:  

$$\text{BSK} = \text{Derive-Secret}(\text{ESK}, \text{"ext binder"}, \text{""})$$

The Derived Secret (DSK) is computed according to the relation:  

$$\text{DSK} = \text{Derive-Secret}(\text{ESK}, \text{"derived"}, \text{""})$$

The Finished External Key (FEK) is computed according to the relation:  

$$\text{FEK} = \text{KDF-Expand-Label}(\text{BSK}, \text{"finished"}, \text{""}, \text{Hash.length})$$

For Derive-Secret procedures, "" is equivalent to the value hash(empty), whose size is hash-length.

## 2.2 Identity Module Procedures

The identity module MUST provide a "Keys Secure Generation and Storage" (KSGS) procedure, which computes and securely stores ESK, BSK and FEK keys.

The KSGS procedure MUST require administrative rights.

A set "Identity Module Key Procedures" (IMKP) of four procedures is required, in order to protect from public exposure ESK, BSK, and FEK:

- CETS: Client Early Traffic Secret
- EEMS: Early Exporter Master Secret
- HEDSK: HKDF-Extract from Derived Secret Key
- HBSK: HMAC from Binder Key Secret

These procedures MAY require user rights.

## 2.3 KSGS: Keys Secure Generation and Storage

The Identity module MUST provide a KSGS procedure, requiring administrative rights, which computes and securely stores ESK, BSK, DSK, and FEK. The KSGS procedure uses with a hash function (for example SHA256) identifies by an AlgoId attribute.

Input: AlgoId, salt, PSK  
Output: Success or Failure

ESK, DSK, and BSK secret values are stored in the identity module.

HL16 : hash Length, 16 bits  
HL8 : hash length, 8 bits  
H0 : hash(empty)

ESK= HMAC(salt=0s,PSK)  
DSK= HMAC(ESK,HL16||0d746c7331332064657269766564||HL8||H0||01)  
BSK= HMAC(ESK,HL16||10746c733133206578742062696e646572||HL8||H0||01)  
FEK= HMAC(BSK,HL16||0E746C7331332066696E69736865640001)

## 2.4 Identity Module Key Procedures (IMKP)

### 2.4.1 CETS: Client Early Traffic Secret

Input: Length, Message  
Output: Client Early Traffic Secret or Failure

```
CETS(ClientHello) = Derive-Secret(ESK, "c e traffic", Message)
= HMAC(ESK, Length || 11746c7331332063206520747261666666963 ||
Message || 01)
```

Message is a hash value.

#### 2.4.2 EEMS: Early Exporter Master Secret

Input: Length, Message

Output: Early Exporter Master Secret or Failure

```
EEMS(ClientHello) = Derive-Secret(ESK, "e exp master", Message)
= HMAC(ESK, Length || 12746c733133206520657870206d6173746572 ||
Message || 01)
```

Message is a hash value

#### 2.4.3 HEDSK: HKDF-Extract from Derived Secret Key

Input: DHE value

Output: Handshake Secret or Failure

```
HEDSK(DHE) = HKDF-Extract(salt=DSK, DHE) = HMAC(salt=DSK, DHE)
```

#### 2.4.4 HBSK: HMAC from Binder Key Secret

Input: data

Output: HMAC(BSK, data) or Failure

```
HBSK(data) = HMAC(FEK, data)
```

Data is a hash value

### 3. Asymmetric Signature

The identity module MUST provide a "Generate Key" (GENKEY) procedure, in order to store or generate private asymmetric key and associated public key. This procedure MUST require administrative rights.

The procedure "Get Public Key" (GETPUB:) is required in order to read the public key value. This procedure MAY require user rights.

The procedure "Signature" (SIGN) is required in order to perform a raw signature for a digest value, computed from certificate. This procedure MAY require user rights.

The symmetric algorithm is identified by the AlgoId attribute. The key is identified by the KeyId attribute.

### 3.1 GENKEY

Input: AlgoId, KeyId

Output: Success or Failure

A private key is generated and stored in the identity module. A public key is computed from the private key.

### 3.2 GETPUB

Input: KeyId

Output: Public Key Value or Failure

### 3.3 SIGN

Input: KeyId, DigestValue

Output: Signature Value or Failure

## 4 Optional Procedures

In IoT context, the computing resources needed for supporting cryptographic procedures such as elliptic curves or true random number generators can be an issue. Optional procedures facilitate TLS1.3 support in such devices.

### 4.1 GENDHE

Input: AlgoId, PublicKey, KeyId

Output: The DHE value

A DHE is computed according to an input public key, and an algorithm identifier.

An ephemeral public key EPK is generated, which is identified by the KeyId attribute and can be retrieved by the GETEPK procedure.

### 4.2 GETEPK

Input: KeyId

Output: Error or ephemeral public key

This procedure returns the ephemeral public key (EPK), identified by the KeyId attribute, previously computed by GENDHE.

### 4.3 RAND

Input: Number of random bytes, Nr

Output: Nr bytes

This procedure generates Nr random bytes

## 5 Identity Module Procedures Summary

First column: The procedure name

Second column: The procedure status Mandatory (M) or Optional (O) for PSK or PKI (e.g. signature generation)

Third column: Input parameters

Fourth column: Output value

Fifth column: The mode, ADMINISTRATOR or USER

Name	Status	Comment	Input	Output	Mode
KSGS	M PSK	Compute Secrets from PSK	AlgoId Salt PSK	none	ADM
GENKEY	M PKI	Generate Private and Public Key	AlgoId KeyId	none	ADM
CETS	M PSK	Compute Early Traffic Secret	Length Message	Client Early Traffic Secret	USER
EEMS	M PSK	Compute Early Exporter Master Secret	Length Message	Early Exporter Master Secret	USER
HEDSK	M PSK	Compute Handshake Secret	DHE	Handshake Secret	USER
HBSK	M PSK	Compute HMAC For Identity Binder	Data	HMAC For Identity Binder	USER
GETPUB	M PKI	Read Public Key	KeyId	Public Key	USER
SIGN	M PKI	Compute Signature	KeyId Data	Public Key	USER
GENDHE	O	Generate Pub.Key Compute DH	AlgoID PUB.Key KeyId	DH Value	USER
GETEPK	O	Read Ephemeris Public Key	KeyId	Public Key	USER
RAND	O	Generate Random Bytes	Number of Bytes	Random Bytes	USER



## 6. Secure Element as Identity Module

Secure elements are defined according to [ISO7816] standards. They support hash functions (sha256, sha384, sha512) and associated HMAC procedures. They also provide DH procedures in  $Z/pZ^*$  groups, and elliptic curves. Open software can be released thanks to the Javacard standards, such as JC3.04, JC3.05, JC3.1.

This section is an illustration of binary encoding rules for secure element according to the T=1 ISO7816 protocol.

An ISO7816 command (TAPDU) is a set of bytes comprising a five bytes header and an optional payload (up to 255 bytes)

The header comprises the following five bytes

- CLA, Class
- INS, Instruction code
- P1, P1 byte
- P2, P2 byte
- P3, length of the payload, or number of expected bytes

The response comprises a payload (up to 255 bytes) and a two bytes status word SW=(SW1, SW2), 9000 meaning successful operation.

### 6.1 Administrator mode

The [ISO7816] command VERIFY (INS=0x20) SHOULD be used to enter the administrative mode.

Tx: CLA=00 INS=20 P1=00 P2=Adm P3=PIN-Length [PIN-Value]  
Rx: 9000

### 6.2 User Mode

The [ISO7816] command VERIFY SHOULD be used to enter the user mode

Tx: CLA=00 INS=20 P1=00 P2=User P3=PIN-Length [PIN-Value]  
Rx: 9000

### 6.3 KSGS: Keys Secure Generation and Storage

Length= 2 + Salt-Length + PSK-Length

Tx: CLA=00 INS=TLS13 P1=AlgoId P2=KSGS P3=Length Salt-Length [Salt-Value] PSK-Length [PSK-Value]  
Rx: 9000

This procedure computes and stores ESK, BSK DSK and FEK.

## 6.3.1 Example

PSK=0102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F20

Tx: CLA=00 INS=85 P1=00 P2=0A P3=23 01 00 20  
 0102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F20  
 Rx: 9000

Sha256(empty) =  
 E3B0C44298FC1C149AFBF4C8996FB92427AE41E4649B934CA495991B7852B855

ESK= HMAC-SHA256(0, PSK)  
 ESK= 23499E7EDF0FBE6BAA137DF0F23BECAEF722AD19FC262855409DE8CD8B3C897

DSK= HMAC-SHA256(ESK, 0020 0d746c7331332064657269766564 20  
 E3B0C44298FC1C149AFBF4C8996FB92427AE41E4649B934CA495991B7852B855 01)  
 DSK=E8E7AC087158FC8440E41A12989F9194783764CD5FC36564028037F2C8206E96

BSK = HMAC-SHA256(ESK, 0020 10746c733133206578742062696e646572 20  
 E3B0C44298FC1C149AFBF4C8996FB92427AE41E4649B934CA495991B7852B855 01)  
 BSK=4351F8A53AA85AC394AB04C516464CAB96E9340C269632D09899537887EE651F

FEK= HMAC-256(BSK, 0020 0E746C7331332066696E6973686564 00 01)  
 FEK=FCA24690D17DDE3F727D29D2186A5F83E1AEBD4889A4841793139168A65BFCB0

## 6.4 CETS: Client Early Traffic Secret

Length = 2 + Messages-Length  
 Hash-Length: the hash length (2 bytes)

Tx: CLA INS=TLS13 P1=CETS P2=ESK P3=Length Hash-Length Messages-  
 Length [Messages]  
 Rx:[Client Early Traffic Secret] SW

## 6.4.1 Example

Tx: CLA=00 INS=85 P1=00 P2=0B P3=03 0020 00  
 Rx: 0738A2B6F6FAA2AF5CDD9B6F0F2B232F19B3256A5926EAC600B911F91E98D2D4  
 9000

Message= NULL = 0s  
 [Client Early Traffic Secret] =  
 HMAC-SHA256(ESK, 0020 11746c733133206320652074726166666963 00 01)

## 6.5 EEMS: Early Exporter Master Secret

Length = 2 + Messages-Length  
 Hash-Length: the hash length (2 bytes)

Tx: CLA INS=TLS13 P1=EEMS P2=ESK P3=Length Hash-Length Messages-  
Length [Messages]

Rx: [Early Exporter Master Secret] SW

#### 6.5.1 Example

Tx: CLA=00 INS=85 P1=01 P2=0B P3=03 0020 00

Rx: 9B7FC6A8F854C16A301DFC566859931DB5EE9A22793142A0C67159C445E7BEAB  
9000

Message= NULL = 0s

[Early Exporter Master Secret] =

HMAC-SHA256(ESK, 0020 12746c733133206520657870206d6173746572 00 01)

#### 6.6 HEDSK: HKDF-Extract from Derived Secret Key

Tx: CLA INS=TLS13 P1=0 P2=HEDSK P3=Data-Length [Data]

Rx: [HMAC(Data,DSK)] SW

##### 6.6.1 Example

Tx: CLA=00 INS=85 P1=00 P2=0E P3=01 00

Rx: 7092C2117D67E6AEB5C5FDF5E6D9C70FBDC69B374E914C26AB08A122483D0E73

DHE=NULL=0s

HMAC-256(DSK,DHE) = HMAC-256(DSK,0s)

#### 6.7 HBSK: HMAC from Binder Key Secret

Tx: CLA INS=TLS13 P1=0 P2=HBSK P3=Data-Length [Data]

Rx: [HMAC(FEK,data)] SW

##### 6.7.1 Example

Tx: CLA=00 INS=85 P1=00 P2=0C P3=01 00

Rx: 3E015D850B89C2470D4C49D4BD8E7C76F2B74175DDD85F393569315DA15480A4

Data=NULL=0s

HMAC-256(FEK,Data) = HMAC-256(DSK,0s)

#### 6.8 Signature Procedures

##### 6.8.1 Keys Generation

Select Identity Module Application (AID= 010203040500)

Tx: CLA=00 INS=A4 P1=04 P2=00 P3=06 01 02 03 04 05 00

Rx: 9000

Verify Administrator PIN (PIN= "00000000")

Tx: CLA=00 INS=20 P1=00 P2=01 P3=08 30 30 30 30 30 30 30 30  
Rx: 9000

Clear Key (P2=KeyId=0)

Tx: CLA=00 INS=81 P1=00 P2=00 P3=00  
Rx: 9000

Init Curve secp256r1 (P1 = idCurve, P2=KeyId)

Tx: CLA=00 INS=89 P1=00 P2=00 P3=00  
Rx: 9000

GenKey (P2=KeyId)

Tx: CLA=00 INS=82 P1=00 P2=00 P3=00  
Rx: 9000

Read PublicKey (P2=KeyId)

Tx: CLA=00 INS=84 P1=06 P2=00 P3=00  
Rx: 0041049E92726E24A548BB69ADA51103F265AA9B9F304E25971427D79EFAF471  
889CCC52FD8B05A729A400105C06AF99592535A4EDF338B5A37BB6089D3B11E7  
1B847B 9000

Read PrivateKey (P2= KeyId)

Tx: CLA=00 INS=84 P1=07 P2=00 P3=00  
Rx: 00208E8793D5C399659D8A35B585534B5D9D0FAB37AD3FC7E8B43373C4BAD81E  
9000

#### 6.8.2 Keys Setting

Select Identity Module Application (AID= 010203040500)

Tx: CLA=00 INS=A4 P1=04 P2=00 P3=06 01 02 03 04 05 00  
Rx: 9000

Verify Administrator PIN (PIN= "00000000")

Tx: CLA=00 INS=20 P1=00 P2=01 P3=08 30 30 30 30 30 30 30 30  
Rx: 9000

Clear Key (P2=KeyId=0)

Tx: CLA=00 INS=81 P1=00 P2=00 P3=00  
Rx: 9000

Init Curve secp256r1 (P1 = idCurve, P2=KeyId)

Tx: CLA=00 INS=89 P1=00 P2=00 P3=00  
Rx: 9000

Set PrivateKey (P2=KeyId)

Tx: CLA=00 INS=88 P1=07 P2=00 P3=20  
2e86bdd6d3b241ddbd00999f6a0ac1cb546d2bfb55744dca40f0268ac2bf7338  
Rx: 9000

Set PublicKey (P2=KeyId)

Tx: CLA=00 INS=88 P1=06 P2=00 P3=41  
045c8c90d0859dd96c722a589c4b62047ff01323cc74383e0e8eb80bea4ea45e55b8  
5499abd39d719885e874ed3f6327960d519ba25423c3fbdc14e6fd0cd5edee  
Rx: 9000

### 6.8.3 Signature

Select Identity Module Application (AID= 010203040500)  
Tx: CLA=00 INS=A4 P1=04 P2=00 P3=06 01 02 03 04 05 00  
Rx: 9000  
Verify User PIN (PIN= "0000")  
CLA=00 INS=20 P1=00 P2=00 P3=04 30 30 30 30

ECDSA secp256r1 Signature (P2=KeyId)  
Tx: CLA=00 INS=80 P1=00 P2=00 P3=20  
0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF  
Rx: 0047304502206BB1B02742C90B5FEAD3EF34F87B49D2A87F846F0368D0DBB3A  
0E9D9F3ABC450022100A0178CDE84FB9ACA4662ECC68638437D46EC27B69657  
8F8080E43ACCA4B35586

### 6.9 GENDHE

Tx: CLA INS=GENDHE P1=AlgoId P2=KeyId P3=Key-Length [Public Key]  
Rx: [DHE] SW

#### 6.9.1 Example

Tx: CLA=00 INS=8A P1=00=Secp256r1 P2=FF P3=41  
4104C4B5F7682C374AAD1C9125C2F225D343A8986C8E0A475E4003F6C98DA13F99  
9E80A55E66F0644E84F7F6503615B9EC4CB7C2844AF6BE7F9091BF319B0291A2D8  
Rx: 1BEE561B95F9EC99EE0E49F28E415D4F74580ACB8D9E0019BD3A7974FF3148E5

### 6.10 GETEPK

Tx: CLA INS=GETEPK P1=06 P2=KeyId P3=2+Key-length  
Rx: [Key-Length] [Public Key] SW

#### 6.10.1 Example

Tx: CLA=00 INS=84 P1=06 P2=FF P3=43  
Rx: 004104CD8EF9695FAC953D89B9C91B994DC77F3140BDEDF54AABF63521548AB9  
8031942C829FC5D958F143AA09E622E6CB190D7A91773E1794F792E1D4D7E1B84603  
FF9000

### 6.11 RAND

Tx: CLA INS=RAND P1=00 P2=00 P3=[Number-Of-bytes]  
Rx: [Number-Of-Bytes] SW

## 6.11.1 Example

Tx: CLA=00 INS=8B P1=00 P2=00 P3=20

Rx: 85DEE2DD24BF79D8CCB6D21C1F515CE040A2E13B8C98177822BD3B66876CD9A1  
9000

## 7. A simple Identity Module code for Javacard 3.04

An example of TLS-IM code is available at [IM-JC].

```
package im;

import javacard.framework.*;
import javacard.security.* ;
import javacardx.crypto.* ;

public class im extends Applet
{
    final static byte    INS_SIGN                = (byte) 0x80 ;
    final static byte    INS_CLEAR_KEYPAIR       = (byte) 0x81 ;
    final static byte    INS_GEN_KEYPAIR        = (byte) 0x82 ;
    final static byte    INS_GET_KEY_PARAM      = (byte) 0x84 ;
    final static byte    INS_HMAC               = (byte) 0x85 ;
    final static byte    INS_GET_STATUS         = (byte) 0x87 ;
    final static byte    INS_SET_KEY_PARAM      = (byte) 0x88 ;
    final static byte    INS_INIT_CURVE         = (byte) 0x89 ;
    final static byte    INS_SELECT             = (byte) 0xA4 ;
    public final static byte INS_VERIFY         = (byte) 0x20 ;
    public final static byte INS_CHANGE_PIN     = (byte) 0x24 ;

    public final static short N_KEYS           = (short) 16;
    public final static byte[] VERSION= {(byte)1, (byte)0};

    KeyPair[] ECCkp          = null ;
    Signature ECCsig         = null ;
    MessageDigest sha256     = null ;

    short status=0 ;
    byte [] DB = null ;
    public final static short DBSIZE = (short)320 ;

    private static OwnerPIN UserPin=null;

    private static final byte[] MyPin =
    {(byte)0x30, (byte)0x30, (byte)0x30, (byte)0x30,
     (byte)0xFF, (byte)0xFF, (byte)0xFF, (byte)0xFF};

    private static OwnerPIN AdminPin=null;

    private static final byte[] OpPin =
    {(byte)0x30, (byte)0x30, (byte)0x30, (byte)0x30,
     (byte)0x30, (byte)0x30, (byte)0x30, (byte)0x30};

    private final static short SW_VERIFICATION_FAILED = (short)0x6300;
    private final static short SW_PIN_VERIFICATION_REQUIRED =
    (short)0x6380;
    final static short SW_KPUB_DEFINED = (short)0x6401;
    final static short SW_KPRIV_DEFINED = (short)0x6402;
    final static short SW_KPRIV_UNDEFINED = (short)0x6403;
```

```
final static short SW_GENKEY_ERROR      = (short)0x6D10;
final static short SW_SIGN_ERROR        = (short)0x6D20;
final static short SW_DUMP_KEYS_PAIR    = (short)0x6D30;
final static short SW_SET_KEY_PARAM     = (short)0x6D40;

private final static byte [] ParamA1 =
{ (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0x00, (byte)0x00,
  (byte)0x00, (byte)0x01, (byte)0x00, (byte)0x00, (byte)0x00, (byte)0x00,
  (byte)0x00, (byte)0x00, (byte)0x00, (byte)0x00, (byte)0x00, (byte)0x00,
  (byte)0x00, (byte)0x00, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff,
  (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff,
  (byte)0xff, (byte)0xfc};

private final static byte [] ParamB1 =
{ (byte)0x5a, (byte)0xc6, (byte)0x35, (byte)0xd8, (byte)0xaa, (byte)0x3a,
  (byte)0x93, (byte)0xe7, (byte)0xb3, (byte)0xeb, (byte)0xbd, (byte)0x55,
  (byte)0x76, (byte)0x98, (byte)0x86, (byte)0xbc, (byte)0x65, (byte)0x1d,
  (byte)0x06, (byte)0xb0, (byte)0xcc, (byte)0x53, (byte)0xb0, (byte)0xf6,
  (byte)0x3b, (byte)0xce, (byte)0x3c, (byte)0x3e, (byte)0x27, (byte)0xd2,
  (byte)0x60, (byte)0x4b};

private final static byte [] ParamField1=
{ (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0x00, (byte)0x00,
  (byte)0x00, (byte)0x01, (byte)0x00, (byte)0x00, (byte)0x00, (byte)0x00,
  (byte)0x00, (byte)0x00, (byte)0x00, (byte)0x00, (byte)0x00, (byte)0x00,
  (byte)0x00, (byte)0x00, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff,
  (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff,
  (byte)0xff, (byte)0xff};

private final static byte [] ParamG1=
{ (byte)0x04, (byte)0x6b, (byte)0x17, (byte)0xd1, (byte)0xf2, (byte)0xe1,
  (byte)0x2c, (byte)0x42, (byte)0x47, (byte)0xf8, (byte)0xbc, (byte)0xe6,
  (byte)0xe5, (byte)0x63, (byte)0xa4, (byte)0x40, (byte)0xf2, (byte)0x77,
  (byte)0x03, (byte)0x7d, (byte)0x81, (byte)0x2d, (byte)0xeb, (byte)0x33,
  (byte)0xa0, (byte)0xf4, (byte)0xa1, (byte)0x39, (byte)0x45, (byte)0xd8,
  (byte)0x98, (byte)0xc2, (byte)0x96, (byte)0x4f, (byte)0xe3, (byte)0x42,
  (byte)0xe2, (byte)0xfe, (byte)0x1a, (byte)0x7f, (byte)0x9b, (byte)0x8e,
  (byte)0xe7, (byte)0xeb, (byte)0x4a, (byte)0x7c, (byte)0x0f, (byte)0x9e,
  (byte)0x16, (byte)0x2b, (byte)0xce, (byte)0x33, (byte)0x57, (byte)0x6b,
  (byte)0x31, (byte)0x5e, (byte)0xce, (byte)0xcb, (byte)0xb6, (byte)0x40,
  (byte)0x68, (byte)0x37, (byte)0xbf, (byte)0x51, (byte)0xf5};

private final static short ParamK1 = (short) 0x0001;

private final static byte [] ParamR1=
{ (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0x00, (byte)0x00,
  (byte)0x00, (byte)0x00, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff,
  (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xff, (byte)0xbc, (byte)0xe6,
  (byte)0xfa, (byte)0xad, (byte)0xa7, (byte)0x17, (byte)0x9e, (byte)0x84,
  (byte)0xf3, (byte)0xb9, (byte)0xca, (byte)0xc2, (byte)0xfc, (byte)0x63,
  (byte)0x25, (byte)0x51};
```



```

private byte []    ESK = new byte[32]; // Early Secret Key
private byte []    HSK = new byte[32]; // Handshake Secret Key
private byte []    eBSK = new byte[32]; // Binder Secret Key
private byte []    rBSK = new byte[32]; // Binder Secret Key
private byte []    feBSK = new byte[32]; // Finished Binder Secret Key
private byte []    frBSK = new byte[32]; // Finished Binder Secret Key

private final static byte  EXTRACT_EARLY    = (byte)0x0A;
private final static byte  EXPAND_EARLY     = (byte)0x0B;
private final static byte  HMAC_EBSK       = (byte)0x0C;
private final static byte  HMAC_RBSK       = (byte)0x0D;
private final static byte  EXTRACT_HANDSHAKE = (byte)0x0E;

private byte [] derived =
{ (byte)0x00, (byte)32, (byte)13, (byte)'t', (byte)'l', (byte)'s',
  (byte)'l', (byte)'3', (byte)' ', (byte)'d', (byte)'e', (byte)'r',
  (byte)'i', (byte)'v', (byte)'e', (byte)'d',
  (byte)0x20, (byte)0xE3, (byte)0xB0, (byte)0xC4, (byte)0x42, (byte)0x98,
  (byte)0xFC, (byte)0x1C, (byte)0x14, (byte)0x9A, (byte)0xFB, (byte)0xF4,
  (byte)0xC8, (byte)0x99, (byte)0x6F, (byte)0xB9, (byte)0x24, (byte)0x27,
  (byte)0xAE, (byte)0x41, (byte)0xE4, (byte)0x64, (byte)0x9B, (byte)0x93,
  (byte)0x4C, (byte)0xA4, (byte)0x95, (byte)0x99, (byte)0x1B, (byte)0x78,
  (byte)0x52, (byte)0xB8, (byte)0x55, (byte)1};

private byte [] ext_binder =
{ (byte)0x00, (byte)32, (byte)16, (byte)'t', (byte)'l', (byte)'s',
  (byte)'l', (byte)'3', (byte)' ', (byte)'e', (byte)'x', (byte)'t',
  (byte)' ', (byte)'b', (byte)'i', (byte)'n', (byte)'d', (byte)'e',
  (byte)'r', (byte)0x20, (byte)0xE3, (byte)0xB0, (byte)0xC4, (byte)0x42,
  (byte)0x98, (byte)0xFC, (byte)0x1C, (byte)0x14, (byte)0x9A, (byte)0xFB,
  (byte)0xF4, (byte)0xC8, (byte)0x99, (byte)0x6F, (byte)0xB9, (byte)0x24,
  (byte)0x27, (byte)0xAE, (byte)0x41, (byte)0xE4, (byte)0x64, (byte)0x9B,
  (byte)0x93, (byte)0x4C, (byte)0xA4, (byte)0x95, (byte)0x99, (byte)0x1B,
  (byte)0x78, (byte)0x52, (byte)0xB8, (byte)0x55, (byte)0x01};

private byte [] res_binder =
{ (byte)0x00, (byte)32, (byte)16, (byte)'t', (byte)'l', (byte)'s',
  (byte)'l', (byte)'3', (byte)' ', (byte)'r', (byte)'e', (byte)'s',
  (byte)' ', (byte)'b', (byte)'i', (byte)'n', (byte)'d', (byte)'e',
  (byte)'r', (byte)0x00, (byte)0x01};

private byte [] c_e_traffic =
{ (byte)17, (byte)'t', (byte)'l', (byte)'s', (byte)'l', (byte)'3',
  (byte)' ', (byte)'c', (byte)' ', (byte)'e', (byte)' ', (byte)'t',
  (byte)'r', (byte)'a', (byte)'f', (byte)'f', (byte)'i', (byte)'c'};

```

```
private byte [] c_exp_master =
{ (byte)18, (byte)'t', (byte)'l', (byte)'s', (byte)'l', (byte)'3',
  (byte)' ', (byte)'e', (byte)' ', (byte)'e', (byte)'x', (byte)'p',
  (byte)' ', (byte)'m', (byte)'a', (byte)'s', (byte)'t', (byte)'e',
  (byte)'r' };

private byte [] finished =
{ (byte)0x00, (byte)32, (byte)14, (byte)'t', (byte)'l', (byte)'s',
  (byte)'l', (byte)'3', (byte)' ', (byte)'f', (byte)'i', (byte)'n',
  (byte)'i', (byte)'s', (byte)'h', (byte)'e', (byte)'d', (byte)0,
  (byte)1 };

public void process(APDU apdu) throws ISOException
{ short adr=0, len=0, index=0, readCount=0;

  byte[] buffer = apdu.getBuffer() ;

  byte cla = buffer[ISO7816.OFFSET_CLA];
  byte ins = buffer[ISO7816.OFFSET_INS];
  byte P1 = buffer[ISO7816.OFFSET_P1] ;
  byte P2 = buffer[ISO7816.OFFSET_P2] ;
  byte P3 = buffer[ISO7816.OFFSET_PC] ;

  adr = Util.makeShort(P1,P2)      ;
  len = Util.makeShort((byte)0,P3) ;

  switch (ins)
  {

  case INS_SELECT:
    readCount = apdu.setIncomingAndReceive();
    return;

  case INS_GET_STATUS:
    Util.arrayCopyNonAtomic(VERSION, (short)0, buffer, (short)0, (short)VERSION.length);
    Util.setShort(buffer, (short)VERSION.length, status);
    apdu.setOutgoingAndSend((short)0, (short)(2+VERSION.length));
    break;

  case INS_VERIFY:
    readCount = apdu.setIncomingAndReceive();
    if (P2 == (byte)1)
    { if (readCount != (short)8)
      { ISOException.throwIt(ISO7816.SW_WRONG_LENGTH);
        verify(AdminPin,buffer) ;
        if(AdminPin.isValidated()) UserPin.resetAndUnblock();
      }
    }
  }
}
```

```
else if (P2 == (byte)0xFF)
{
    if (readCount != (short)8)
        ISOException.throwIt(ISO7816.SW_WRONG_LENGTH);
    verify(AdminPin,buffer) ;
    if(AdminPin.isValidated())
    {
        UserPin.resetAndUnblock();
        UserPin.update(MyPin, (short)0, (byte)8) ;
    }
}
}
else if (P2 == (byte)0)
{
    if (readCount > (short)8)
        ISOException.throwIt(ISO7816.SW_WRONG_LENGTH);
    verify(UserPin,buffer);
}
else
    ISOException.throwIt(ISO7816.SW_WRONG_P1P2);
break;
```

```
case INS_CHANGE_PIN:
    readCount = apdu.setIncomingAndReceive() ;
    if (readCount != (short)16)
        ISOException.throwIt(ISO7816.SW_WRONG_LENGTH);
    buffer[4]=(byte)8;
    if (P2 == (byte)1)
    {
        verify(AdminPin,buffer) ;
        AdminPin.update(buffer, (short)13, (byte)8);
    }
    else if (P2 == (byte)0)
    {
        verify(UserPin,buffer) ;
        UserPin.update(buffer, (short)13, (byte)8);
    }
    else
        ISOException.throwIt(ISO7816.SW_WRONG_P1P2);
    break;
```

```
case INS_HMAC:
```

```
    readCount = apdu.setIncomingAndReceive();
    len = Util.makeShort((byte)0,buffer[(short)4]);
```

```
    if (len != readCount)
        ISOException.throwIt(ISO7816.SW_CORRECT_LENGTH_00);
```

```
    else if ( (!AdminPin.isValidated()) && (!UserPin.isValidated()) )
        ISOException.throwIt(SW_PIN_VERIFICATION_REQUIRED);
```

```
if (P2 == (byte)2) // Compute HMAC
{
    len = Util.makeShort((byte)0,buffer[(short)5]) ;
    hmac(buffer, (short)6, len, buffer, (short)(7+len),
        Util.makeShort((byte)0, buffer[(short)(6+len)]),
        sha256, buffer, (short)0,true);
    apdu.setOutgoingAndSend((short)0,(short)sha256.getLength());
}

else if (P2 == EXTRACT_EARLY)
{
    len = Util.makeShort((byte)0,buffer[(short)5]); //HMAC: key-length
    hmac(buffer, (short)6, len, buffer, (short)(7+len),
        Util.makeShort((byte)0,buffer[(short)(6+len)]),
        sha256, buffer, (short)0,true);
    Util.arrayCopyNonAtomic(buffer, (short)0,ESK, (short)0,
        (short)ESK.length);
    Util.arrayCopyNonAtomic(buffer, (short)0,buffer, (short)32,
        (short)32);

    hmac(ESK, (short)0, (short)ESK.length,
        derived, (short)0, (short)derived.length,
        sha256,
        buffer, (short)0,true);
    Util.arrayCopyNonAtomic(buffer, (short)0,HSK, (short)0,
        (short)HSK.length);
    Util.arrayCopyNonAtomic(buffer, (short)0,buffer, (short)64,
        (short)32);

    hmac(ESK, (short)0, (short)ESK.length,
        ext_binder, (short)0, (short)ext_binder.length,
        sha256,
        buffer, (short)0,true);
    Util.arrayCopyNonAtomic(buffer, (short)0,eBSK, (short)0,
        (short)eBSK.length);
    Util.arrayCopyNonAtomic(buffer, (short)0,buffer, (short)96,
        (short)32);

    hmac(ESK, (short)0, (short)ESK.length,
        res_binder, (short)0, (short)res_binder.length,
        sha256,
        buffer, (short)0,true);
    Util.arrayCopyNonAtomic(buffer, (short)0,rBSK, (short)0,
        (short)rBSK.length);
    Util.arrayCopyNonAtomic(buffer, (short)0,buffer, (short)128,
        (short)32);
}
```

```

    hmac(eBSK, (short)0, (short)eBSK.length,
        finished, (short)0, (short)finished.length,
        sha256,
        buffer, (short)0, true);
    Util.arrayCopyNonAtomic(buffer, (short)0, feBSK, (short)0,
        (short)feBSK.length);
    Util.arrayCopyNonAtomic(buffer, (short)0, buffer, (short)160,
        (short)32);

    hmac(rBSK, (short)0, (short)rBSK.length,
        finished, (short)0, (short)finished.length,
        sha256,
        buffer, (short)0, true);
    Util.arrayCopyNonAtomic(buffer, (short)0, frBSK, (short)0,
        (short)frBSK.length);
    Util.arrayCopyNonAtomic(buffer, (short)0, buffer, (short)192,
        (short)32);

    If (P1==(byte)0xFF)
        apdu.setOutgoingAndSend((short)32, (short)192);
    return ;
}

else if (P2 == EXPAND_EARLY)
{
    len = Util.makeShort((byte)0, buffer[(short)7]); // data length
    if (P1 == (byte)0)
    {
        Util.arrayCopyNonAtomic(buffer, (short)5, buffer, (short)0,
            (short)2);
        Util.arrayCopyNonAtomic(buffer, (short)7,
            buffer, (short)(2+ c_e_traffic.length),
            (short)(readCount-2));
        Util.arrayCopyNonAtomic(c_e_traffic, (short)0, buffer, (short)2,
            (short)c_e_traffic.length);
        buffer[(short)(readCount + c_e_traffic.length)] = (byte)0x01;
        hmac(ESK, (short)0, (short)ESK.length,
            buffer, (short)0, (short)(readCount+c_e_traffic.length+1),
            sha256,
            buffer, (short)0, true);
        apdu.setOutgoingAndSend((short)0, (short)32);
        return;
    }
    else if (P1 == (byte)1)
    {
        Util.arrayCopyNonAtomic(buffer, (short)5, buffer, (short)0,
            (short)2);
        Util.arrayCopyNonAtomic(buffer, (short)7, buffer,
            (short)(2+ c_exp_master.length),
            (short)(readCount-2));
    }
}

```

```

        Util.arrayCopyNonAtomic(c_exp_master, (short)0, buffer, (short)2,
                                (short)c_exp_master.length);
        buffer[(short)(readCount + c_exp_master.length)] = (byte)0x01;
        hmac(ESK, (short)0, (short)ESK.length,
            buffer, (short)0, (short)(readCount+c_exp_master.length+1),
            sha256,
            buffer, (short)0, true);
        apdu.setOutgoingAndSend((short)0, (short)32);
        return;
    }
    else
        ISOException.throwIt(ISO7816.SW_INCORRECT_P1P2);

else if ( P2 == HMAC_RBSK)
{
    hmac(frBSK, (short)0, (short)rBSK.length,
        buffer, (short)5, readCount,
        sha256,
        buffer, (short)0, true);
    apdu.setOutgoingAndSend((short)0, (short)sha256.getLength());
}

else if (P2 == HMAC_EBSK)
{
    hmac(feBSK, (short)0, (short)eBSK.length,
        buffer, (short)5, readCount,
        sha256,
        buffer, (short)0, true);
    apdu.setOutgoingAndSend((short)0, (short)sha256.getLength());
}

else if (P2 == EXTRACT_HANDSHAKE )
{
    hmac(HSK, (short)0, (short)HSK.length,
        buffer, (short)5, readCount,
        sha256,
        buffer, (short)0, true);
    apdu.setOutgoingAndSend((short)0, (short)32);
}

else
    ISOException.throwIt(ISO7816.SW_INCORRECT_P1P2);
break;

case INS_SIGN:
    readCount = apdu.setIncomingAndReceive();
    if ( (!AdminPin.isValidated()) && (!UserPin.isValidated()) )
        ISOException.throwIt(SW_PIN_VERIFICATION_REQUIRED);

    index= Util.makeShort((byte)0, P2);
    if ( (index <0) || (index >= N_KEYS))
        ISOException.throwIt(ISO7816.SW_CONDITIONS_NOT_SATISFIED);
    if (!ECCKp[index].getPublic().isInitialized())

```

```

ISOException.throwIt(SW_KPUB_DEFINED);
if (!ECCKp[index].getPrivate().isInitialized())
ISOException.throwIt(SW_KPRIV_DEFINED);

switch (P1)
{
    case (byte)0: // RAW 256 bits
    case (byte)33:// ALG_ECDSA_SHA_256
        len= EccSign(ECCKp[index],buffer,P1) ;
        apdu.setOutgoingAndSend((short)0,len);
        break;
    default:
        ISOException.throwIt(ISO7816.SW_INCORRECT_P1P2);
        break;
}
break;

case INS_CLEAR_KEYPAIR:

if ( !AdminPin.isValidated())
ISOException.throwIt(SW_PIN_VERIFICATION_REQUIRED);
index= Util.makeShort((byte)0,P2);
if ( (index <0) || (index >= N_KEYS))
ISOException.throwIt(ISO7816.SW_CONDITIONS_NOT_SATISFIED);
if (ECCKp[index].getPublic().isInitialized())
ECCKp[index].getPublic().clearKey();
if (ECCKp[index].getPrivate().isInitialized())
ECCKp[index].getPrivate().clearKey();
break;

case INS_GEN_KEYPAIR: // Generate KeyPair

if ( !AdminPin.isValidated())
ISOException.throwIt(SW_PIN_VERIFICATION_REQUIRED);
index= Util.makeShort((byte)0,P2);
if ( (index <0) || (index >= N_KEYS))
ISOException.throwIt(ISO7816.SW_CONDITIONS_NOT_SATISFIED);
if (ECCKp[index].getPublic().isInitialized())
ISOException.throwIt(SW_KPUB_DEFINED);
if (ECCKp[index].getPrivate().isInitialized())
ISOException.throwIt(SW_KPRIV_DEFINED);
len=this.GenECCKp(ECCKp[index]);
break;

case INS_GET_KEY_PARAM:

if ( (!AdminPin.isValidated()) && (!UserPin.isValidated()) )
ISOException.throwIt(SW_PIN_VERIFICATION_REQUIRED);
index= Util.makeShort((byte)0,P2);
if ( (index <0) || (index >= N_KEYS))

```

```
ISOException.throwIt(ISO7816.SW_CONDITIONS_NOT_SATISFIED);
if ( (P1 == (byte)7) && !AdminPin.isValidated())
ISOException.throwIt(SW_PIN_VERIFICATION_REQUIRED);
if ( (P1 == (byte)6) && !ECCKp[index].getPublic().isInitialized())
ISOException.throwIt(SW_KPUB_DEFINED);
if ( (P1 == (byte)7) && !ECCKp[index].getPrivate().isInitialized())
ISOException.throwIt(SW_KPRIV_DEFINED)
try
{
    switch (P1)
    {
        case 0:
            len= ((ECPublicKey) ECCKp[index].getPublic())
                .getA(buffer, (short) 2));
            Util.setShort(buffer, (short) 0, len);
            apdu.setOutgoingAndSend((short) 0, (short) (len+2));
            break;

            case 1:
            len= ((ECPublicKey) ECCKp[index].getPublic())
                .getB(buffer, (short) 2));
            Util.setShort(buffer, (short) 0, len);
            apdu.setOutgoingAndSend((short) 0, (short) (len+2));
            break;

            case 2:
            len= ((ECPublicKey) ECCKp[index].getPublic())
                .getField(buffer, (short) 2));
            Util.setShort(buffer, (short) 0, len);
            apdu.setOutgoingAndSend((short) 0, (short) (len+2));
            break;

            case 3:
            len= ((ECPublicKey) ECCKp[index].getPublic())
                .getG(buffer, (short) 2));
            Util.setShort(buffer, (short) 0, len);
            apdu.setOutgoingAndSend((short) 0, (short) (len+2));
            break;

            case 4:
            len= ((ECPublicKey) ECCKp[index].getPublic()).getK();
            Util.setShort(buffer, (short) 2, len);
            Util.setShort(buffer, (short) 0, (short) 2);
            apdu.setOutgoingAndSend((short) 0, (short) 4);
            break;

            case 5:
            len= ((ECPublicKey) ECCKp[index].getPublic())
                .getR(buffer, (short) 2));
            Util.setShort(buffer, (short) 0, len);
            apdu.setOutgoingAndSend((short) 0, (short) (len+2));
            break;
```



```

        case (byte)6:
            len= ((ECPublicKey) ECCKp[index].getPublic())
                .getW(buffer, (short) (2));
            Util.setShort(buffer, (short) 0, len);
            apdu.setOutgoingAndSend((short) 0, (short) (len+2));
            break;

        case (byte)7:
            len= ((ECPrivateKey) ECCKp[index].getPrivate())
                .getS(buffer, (short) (2));
            Util.setShort(buffer, (short) 0, len);
            apdu.setOutgoingAndSend((short) 0, (short) (len+2));
            break;

        default:
            ISOException.throwIt(ISO7816.SW_INCORRECT_P1P2);
            break;
    }
}
catch (CryptoException e)
{ISOException.throwIt(SW_DUMP_KEYS_PAIR);
 break;
}

break;

case INS_SET_KEY_PARAM:

readCount = apdu.setIncomingAndReceive();

if ( !AdminPin.isValidated())
ISOException.throwIt(SW_PIN_VERIFICATION_REQUIRED);
index= Util.makeShort((byte) 0, P2);
if ( (index < 0) || (index >= N_KEYS))
ISOException.throwIt(ISO7816.SW_CONDITIONS_NOT_SATISFIED);
if ( (P1 == (byte)6) && ECCKp[index].getPublic().isInitialized())
ISOException.throwIt(SW_KPUB_DEFINED);
if ( (P1 == (byte)7) && ECCKp[index].getPrivate().isInitialized())
ISOException.throwIt(SW_KPRIV_DEFINED);

try
{
    switch (P1)
    {
        case (byte)0:
            ((ECPublicKey) ECCKp[index].getPublic())
                .setA(buffer, (short) 5, len);
            ((ECPrivateKey) ECCKp[index].getPrivate())
                .setA(buffer, (short) 5, len);
            break;
    }
}

```

```
case (byte)1:
    ((ECPublicKey) ECCKp[index].getPublic())
        .setB(buffer, (short) 5, len);
    ((ECPrivateKey) ECCKp[index].getPrivate())
        .setB(buffer, (short) 5, len);
    break;

case (byte)2:
    ((ECPublicKey) ECCKp[index].getPublic())
        .setFieldFP(buffer, (short) 5, len) ;
    ((ECPrivateKey) ECCKp[index].getPrivate())
        .setFieldFP(buffer, (short) 5, len);
    break;

case (byte)3:
    ((ECPublicKey) ECCKp[index].getPublic())
        .setG(buffer, (short) 5, len) ;
    ((ECPrivateKey) ECCKp[index].getPrivate())
        .setG(buffer, (short) 5, len);
    break;

case (byte)4:
    ((ECPublicKey) ECCKp[index].getPublic())
        .setK(Util.makeShort(buffer[5],buffer[6])) ;
    ((ECPrivateKey) ECCKp[index].getPrivate())
        .setK(Util.makeShort(buffer[5],buffer[6]));
    break;

case (byte)5:
    ((ECPublicKey) ECCKp[index].getPublic())
        .setR(buffer, (short) 5, len);
    ((ECPrivateKey) ECCKp[index].getPrivate())
        .setR(buffer, (short) 5, len);
    break;

case (byte)6:
    ((ECPublicKey) ECCKp[index].getPublic())
        .setW(buffer, (short) 5, len) ;
    break;

case (byte)7:
    ((ECPrivateKey) ECCKp[index].getPrivate())
        .setS(buffer, (short) 5, len);
    break;

default:
    ISOException.throwIt(ISO7816.SW_INCORRECT_P1P2);
    break;
}
```

```

catch (CryptoException e)
{ISOException.throwIt(SW_SET_KEY_PARAM);
  break;
}

break;

case INS_INIT_CURVE:

if ( !AdminPin.isValidated())
ISOException.throwIt(SW_PIN_VERIFICATION_REQUIRED);
index= Util.makeShort((byte)0,P2);
if ( (index <0) || (index >= N_KEYS))
ISOException.throwIt(ISO7816.SW_CONDITIONS_NOT_SATISFIED);
if ( (P1 == (byte)6) && ECCKp[index].getPublic().isInitialized() )
ISOException.throwIt(SW_KPUB_DEFINED);
if ((P1 == (byte)7) && ECCKp[index].getPrivate().isInitialized())
ISOException.throwIt(SW_KPRIV_DEFINED);

switch((byte)P1)
{  case (byte)0:
    case (byte)1:
      ((ECPublicKey)ECCKp[index].getPublic())
        .setA(ParamA1, (short)0, (short)ParamA1.length) ;
      ((ECPrivateKey)ECCKp[index].getPrivate())
        .setA(ParamA1, (short)0, (short)ParamA1.length);
      ((ECPublicKey)ECCKp[index].getPublic())
        .setB(ParamB1, (short)0, (short)ParamB1.length) ;
      ((ECPrivateKey)ECCKp[index].getPrivate())
        .setB(ParamB1, (short)0, (short)ParamB1.length);
      ((ECPublicKey)ECCKp[index].getPublic())
        .setFieldFP(ParamField1, (short)0, (short)ParamField1.length);
      ((ECPrivateKey)ECCKp[index].getPrivate())
        .setFieldFP(ParamField1, (short)0, (short)ParamField1.length);
      ((ECPublicKey)ECCKp[index].getPublic())
        .setG(ParamG1, (short)0, (short)ParamG1.length) ;
      ((ECPrivateKey)ECCKp[index].getPrivate())
        .setG(ParamG1, (short)0, (short)ParamG1.length);
      ((ECPublicKey)ECCKp[index].getPublic())
        .setK(ParamK1) ;
      ((ECPrivateKey)ECCKp[index].getPrivate())
        .setK(ParamK1);
      ((ECPublicKey)ECCKp[index].getPublic())
        .setR(ParamR1, (short)0, (short)ParamR1.length) ;
      ((ECPrivateKey)ECCKp[index].getPrivate())
        .setR(ParamR1, (short)0, (short)ParamR1.length);
      break;

```

```

        default:
            ISOException.throwIt(ISO7816.SW_INCORRECT_P1P2);
            break;
    }
break;

default:
    ISOException.throwIt(ISO7816.SW_INS_NOT_SUPPORTED);
}

}

public short EccSign(KeyPair ECCkeyPair, byte [] buf, byte mode)
{ short len, sLen=(short)0;
  len= Util.makeShort((byte)0, buf[4]);
  Util.arrayCopy(buf, (short)5, buf, (short)2, len) // Sign
  try
  { if (mode == (byte)0) // default
    { ECCsig.init(ECCkeyPair.getPrivate(), Signature.MODE_SIGN);
      sLen = ECCsig.signPreComputedHash(buf, (short)2, len buf,
        (short)(2+len));
    }
    else
    { ECCsig.init(ECCkeyPair.getPrivate(), Signature.MODE_SIGN);
      sLen = ECCsig.sign(buf, (short)2, len, buf, (short)(2+len));
    }
  }
  catch (CryptoException e)
  { ISOException.throwIt(SW_SIGN_ERROR);
    return (short)0;
  }

  Util.arrayCopy(buf, (short)(2+len), buf, (short)2, sLen);
  Util.setShort(buf, (short)0, sLen);
  return(short)(sLen+2);
}

public short GenECCkp(KeyPair ECCkeyPair)
{ short len;
  try
  { ECCkeyPair.genKeyPair(); }
  catch (CryptoException e)
  { ISOException.throwIt(SW_GENKEY_ERROR);
    return (short)0;
  }
  return 0;
}

```

```

public void verify(OwnerPIN pin,byte [] buffer) throws ISOException
{
    short i,x;
    x = Util.makeShort((byte)0,buffer[4]);
    for(i=x;i<(short)8;i=(short)(i+1))
        buffer[(short)(5+i)]=(byte)0xFF;
    if ( pin.check(buffer, (short)5,(byte)8) == false )
        ISOException.throwIt((short)((short)SW_VERIFICATION_FAILED |
                                   (short)pin.getTriesRemaining()));
}

public static final short DB_off = (short)0 ;

public void hmac
( byte [] k,short k_off, short lk, // Secret key
  byte [] d,short d_off,short ld, // data
  MessageDigest md,
  byte out[], short out_off, boolean init)
{
    short i,DIGESTSIZE, DIGESTSIZE2=(short)64,BLOCKSIZE=(short)128;
    DIGESTSIZE=(short)md.getLength();
    if (md.getAlgorithm() == md.ALG_SHA_512)
    { DIGESTSIZE2= (short)64; BLOCKSIZE = (short)128; }
    else if (md.getAlgorithm() == md.ALG_SHA_256)
    { DIGESTSIZE2= (short)32; BLOCKSIZE = (short)64; }

    if (init)
    { if (lk > (short)BLOCKSIZE )
      { md.reset();
        md.doFinal(k,k_off,lk,k,k_off);
        lk = DIGESTSIZE ;
      }
      for (i = 0 ; i < lk ; i=(short)(i+1))
        DB[(short)(i+DB_off+BLOCKSIZE+DIGESTSIZE2)] =
          (byte)(k[(short)(i+k_off)] ^ (byte)0x36) ;
      Util.arrayFillNonAtomic (
        DB,(short)(BLOCKSIZE+DIGESTSIZE2+lk+DB_off),
        (short)(BLOCKSIZE-lk), (byte)0x36);
      for (i = 0 ; i < lk ; i=(short)(i+1))
        DB[(short)(i+DB_off)] = (byte)(k[(short)(i+k_off)] ^ (byte)0x5C);
      Util.arrayFillNonAtomic(DB,(short)(lk+DB_off),
                              (short)(BLOCKSIZE-lk), (byte)0x5C);
    }

    md.reset();
    md.update(DB,(short)(DB_off+BLOCKSIZE+DIGESTSIZE2),BLOCKSIZE);
    md.doFinal(d, d_off,ld,DB,(short)(DB_off+BLOCKSIZE));
    md.reset();
    md.doFinal(DB,DB_off,(short)(DIGESTSIZE+BLOCKSIZE),out,out_off);
}

```

```

protected im(byte[] bArray,short bOffset,byte bLength)
{ init();
  register();
}

public void init()
{ short i=0;
  status = (short)0;
  ECCkp = new KeyPair[N_KEYS];
  UserPin = new OwnerPIN((byte)3,(byte)8); // 3 tries, 4=Max Size
  AdminPin = new OwnerPIN((byte)10,(byte)8); // 10 tries 8=Max Size
  UserPin.update(MyPin,(short)0,(byte)8) ;
  AdminPin.update(OpPin,(short)0,(byte)8);
  for(i=0;i<N_KEYS;i++)
  {
    try{
      ECCkp[i] = new
        KeyPair(KeyPair.ALG_EC_FP,KeyBuilder.LENGTH_EC_FP_256);
      status =(short)(status + (short)1);
    }
    catch (CryptoException e){}
  }
  try {
    ECCsig =
      Signature.getInstance(Signature.ALG_ECDSA_SHA_256, false);
    status =(short)(status | (short)0x0100);
  }
  catch (CryptoException e){}
  try {
    sha256 =
      MessageDigest.getInstance(MessageDigest.ALG_SHA_256, false);
    status =(short)(status | (short)0x2000);
  }
  catch (CryptoException e){}
  DB = JCSysSystem.makeTransientByteArray(DBSIZE,
                                           JCSysSystem.CLEAR_ON_DESELECT);
}

public static void install(byte[] bArray, short bOffset,
                           byte bLength )
{ new im(bArray,bOffset,bLength);}

public boolean select()
{ if (UserPin.isValidated()) UserPin.reset();
  if (AdminPin.isValidated()) AdminPin.reset();
  return true;
}

```

## 8 IANA Considerations

This draft does not require any action from IANA.

## 9 Security Considerations

This entire document is about security.

## 10 References

### 10.1 Normative References

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <https://www.rfc-editor.org/info/rfc5869>.

[ISO7816] ISO 7816, "Cards Identification - Integrated Circuit Cards with Contacts", The International Organization for Standardization (ISO).

[GP-SPI-I2C] GlobalPlatform Technology, APDU Transport over SPI/I2C Version 0.0.0.39, July 2019

### 10.2 Informative References

[IM-JC] <https://github.com/purien/TLS-SE/blob/master/im/im.java>

## 11 Authors' Addresses

Pascal Urien  
Telecom Paris  
19 place Marguerite Perey  
91120 Palaiseau Phone: NA  
France Email: [Pascal.Urien@telecom-paris.fr](mailto:Pascal.Urien@telecom-paris.fr)

TLS Working Group  
Internet Draft  
Intended status: Experimental

P. Urien  
Telecom Paris

September 24 2021

Expires: March 2022

Secure Element for TLS Version 1.3  
draft-urien-tls-se-03.txt

## Abstract

This draft presents ISO7816 interface for TLS1.3 stack running in secure element. It presents supported cipher suites and key exchange modes, and describes embedded software architecture. TLS 1.3 is the de facto security stack for emerging Internet of Things (IoT) devices. Some of them are constraint nodes, with limited computing resources. Furthermore cheap System on Chip (SoC) components usually provide tamper resistant features, so private or pre shared keys are exposed to hacking. According to the technology state of art, some ISO7816 secure elements are able to process TLS 1.3, but with a limited set of cipher suites. There are two benefits for TLS-SE; first fully tamper resistant processing of TLS protocol, which increases the security level insurance; second embedded software component ready for use, which relieves the software of the burden of cryptographic libraries and associated attacks. TLS-SE devices may also embed standalone applications, which are accessed via internet node, using a routing procedure based on SNI extension.

## 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.

## Status of this Memo

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

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

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

This Internet-Draft will expire on March 2022.



## Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

Abstract.....	1
Requirements Language.....	1
Status of this Memo.....	1
Copyright Notice.....	2
1 Overview.....	4
2 About Secure Elements.....	5
3 Software components for TLS-SE.....	5
3.1 Cryptographic resources.....	6
3.2 Data exchange.....	6
3.2.1 Receiving Record Packet .....	6
3.2.2 Sending Record Packet .....	7
3.2.4 RECV and SEND procedure for open application AEAD .....	9
3.3 TLS state machine.....	9
3.4 TLS library.....	10
4 ISO7816 interface.....	11
5 ISO 7816 Use Case.....	12
5 TLS-SE Name.....	14
6 Server Name Indication.....	14
7 IANA Considerations.....	14
8 Security Considerations.....	14
9 References.....	14
9.1 Normative References.....	14
9.2 Informative References.....	15
10 Authors' Addresses.....	15

## 1 Overview

This draft presents ISO7816 interface for TLS1.3 stack running in secure element (see Figure 1), it presents supported cipher suites and key exchange modes, and describes embedded software architecture. TLS 1.3 [RFC8446] is the de facto security stack for emerging Internet of Things (IoT) devices. Some of them are constraint nodes, with limited computing resources. Furthermore cheap System on Chip (SOC) components don't usually provide tamper resistant features, so private or pre shared keys are exposed to hacking. The identity module (im) detailed in [IM] protects identity credentials. The TLS identity module [IM] MAY be based on secure element [ISO7816]. According to the technology state of art, some secure elements are able to process TLS 1.3, but with a limited set of cipher suites. There are two benefits for TLS-SE; first fully tamper resistant processing of TLS protocol, which increases the security level insurance; second embedded software component ready for use, which relieves the software of the burden of cryptographic libraries and associated attacks. Multiple TLS-SE devices, embedding standalone applications, can be hosted by an internet node. In this case SNI extension [RFC6066] MAY be used in order to select the right secure element (see Figure 2).

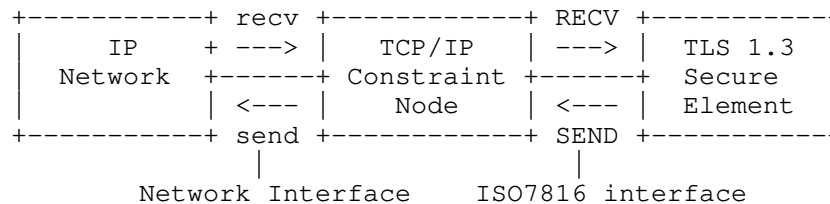


Figure 1. TLS 1.3 Secure Element (TLS-SE)

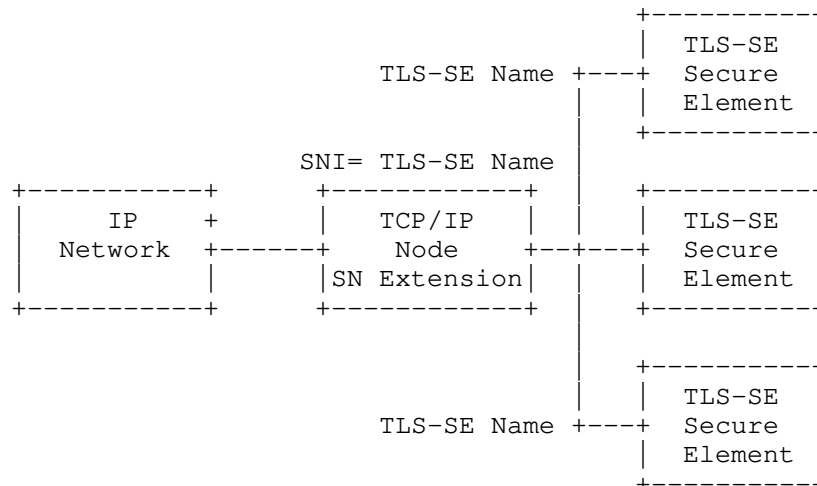


Figure 2. Routing procedure based on SNI for TLS-SE devices

## 2 About Secure Elements

Secure elements are defined according to [ISO7816] standards. They support hash functions (sha256, sha384, sha512) and associated HMAC procedures. They also provide signatures and DH procedures in  $Z/pZ^*$  groups, or elliptic curves (for example secp256r1). Open software can be released thanks to JavaCard (JC) standards, such as JC3.04, or JC3.05. Most of secure elements use 8 bits Micro Controller Unit (MCU) and embedded cryptographic accelerator. Non volatile memory size is up to 100KB, and RAM size is up to 10KB.

Below is an illustration of binary encoding rules for secure element according to the T=0 ISO7816 protocol.

An ISO7816 request is a set of bytes comprising a five byte header and an optional payload (up to 255 bytes)

The header comprises the following five bytes:

- CLA, Class
- INS, Instruction code
- P1, P1 byte
- P2, P2 byte
- P3, length of the optional payload, or number of expected bytes

The response comprises an optional payload (up to 256 bytes) and a two bytes status word (SW1, SW2), SW1=90, SW2=00 (SW=9000) meaning successful operation.

The ISO7816 defines two main classes for data exchange (called transport protocol), T=0, and T=1.

The T=0 transport protocol is a byte stream; a payload can be included in request or response, but not in both.

The T=1 transport protocol is a frame stream; payload can be included both in request and response.

## 3 Software components for TLS-SE

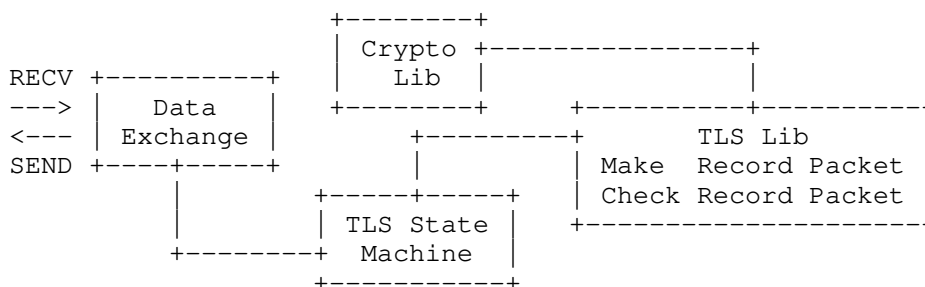


Figure 3. Software Components for TLS-SE

### 3.1 Cryptographic resources

Many secure elements support hash functions sha256, sha384 and sha512 used by TLS1.3. Associated HMAC, HKDF-Extract and Derive-Secret, MUST be implemented by a dedicated cryptographic library.

Many secure elements support the secp256r1 elliptic curve. Diffie-Hellman (DH) calculation are performed according to [IEEE1363] using the ECKAS-DH1 scheme with the identity map as the key derivation function, (KDF), so that the shared secret is the x-coordinate of the ECDH shared secret elliptic curve point represented as an octet string. ECDSA signature is also available for 256,384 and 512 hash size.

AES-128 is usually implemented, by not AES-CCM. So this AEAD algorithm SHOULD be implemented by a dedicated cryptographic library.

In summary, according to the state of art TLS-SE supports the secp256r1 EC group, associated ECDSA signature computing and checking, and EC-DHE key establishment. It also implements the AES-128-CCM-SHA256 cipher suite.

### 3.2 Data exchange

TLS record layer packets are received and sent from/to TCP/IP network thanks to well known socket procedures. TLS-SE processes these packets according to a dedicated state machine.

#### 3.2.1 Receiving Record Packet

Dedicated ISO7816 requests (named RECV) push incoming record messages in secure element. A fragmentation mechanism splits the record packet in one a several ISO7816 requests, whose payload size is less than 255 bytes. A 2 bits fragmentation-flag field indicates the fragment status; bit F-First notifies the first fragment, and bit F-Last notifies the last fragment.

The ISO7816 RECV request COULD be encoded as  
CLA=00, INS=D8, P1=0, P2=fragmentation-flag, P3=fragment-length  
F-First=b01, F-Last=b10, F-More=b00

When application AEAD is opened a two bits flag (F-Encrypt, F-Decrypt) indicates the cryptographic operation:

- P2=b01= F-Decrypt, decryption
- P2=b10= F-Encrypt, encryption
- P2=b00= Standalone embedded application.

If F-Last is not set, the ISO7816 response is always 9000 when no error occurs. For the last fragment five cases may occur:

- sw-ok: no error, no record message returned, response = 9000.
- sw-open, no error, no record message returned, TLS application AEAD is opened, for example response =9001.
- sw-close: no error, , no record message returned, TLS application AEAD is closed, for example response =9002
- sw-error: error, no record message returned.
- sw-more(size): no error, a message or message fragment is ready. For example sw-more(size)= 6lxy, in which xy is the size of the first fragment.

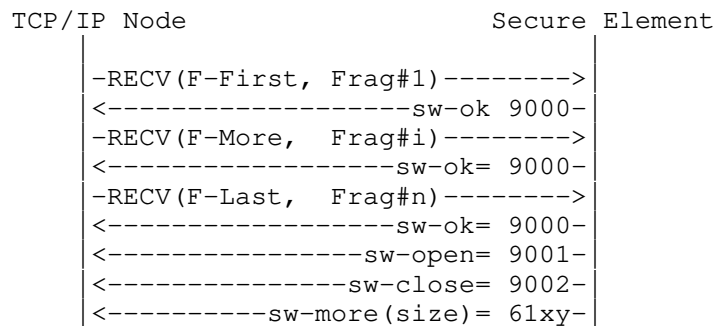


Figure 4. Receiving record packet, segmentation mechanism.

### 3.2.2 Sending Record Packet

A sending procedure starts by the reception of a sw-more(size) status, ending a response. This event may occur at the end of RECV procedure (see figure 6) or after TLS state machine reset (see figure 5).

A RECV(F-First, No-Frag) request resets the TLS state machine. For TLS client a sw-more(size) status is returned. For TLS server the sw-ok status is returned.

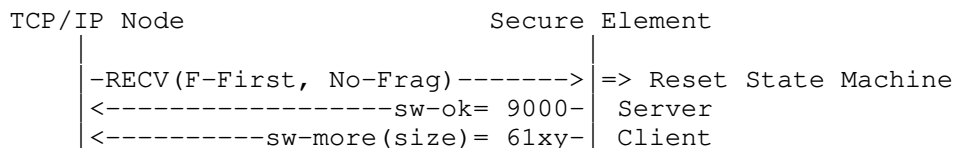


Figure 5. Starting the SEND procedure after RESET request.

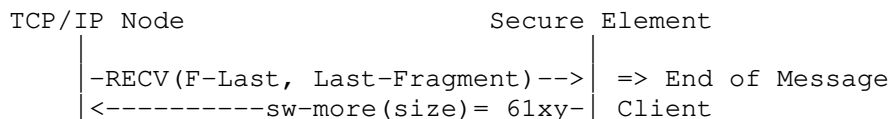


Figure 6. Starting SEND procedure after the end of RECV procedure.

The SEND(size) reads a record fragment, whose length is equal to size. It MAY be necessary to adjust the SEND size (see figure 7). Typically at the end of RECV procedure, the size indicated by the sw-more(size) status is an expected fragment length. In that case the status sw-retry status (for example 6Cxy) indicates the fragment size.

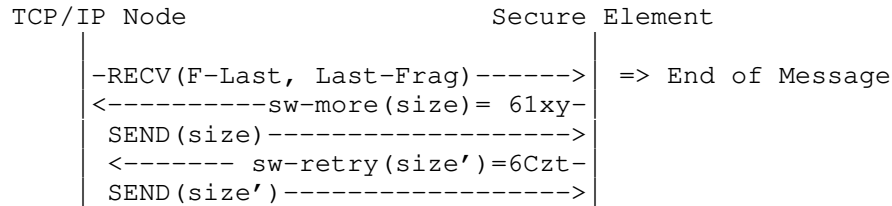


Figure 7. Adjusting SEND size.

The SEND(size) request is encoded as :  
CLA=0, INS=C0, P1=0, P2=0, P3=size

The SEND procedure (see Figure 8) is a set of SEND requests, which read record packet fragments.

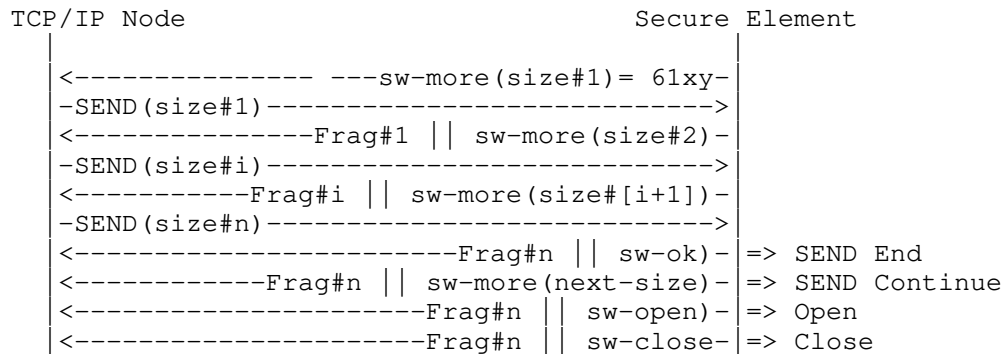


Figure 8. The SEND procedure

At the end of SEND procedure four events MAY occur:

- End of SEND procedure (status = sw-ok). No more record packets are available.
- SEND procedure to be continued (status = sw-more(size)). Another record packet is available.
- End of SEND procedure, application AEAD is ready for use (status = sw-open)
- End of SEND procedure, application AEAD is closed (status = sw-close)

### 3.2.4 RECV and SEND procedure for open application AEAD

When the application AEAD is opened RECV performs decryption and encryption operations (see figure 9).

For decryption operation (RECV(F-Decrypt)) the RECV procedure pushes the incoming record packet. The returned payload by the SEND procedure is the decrypted message ended by the protocol byte.

For encryption operation (RECV(F-Encrypt)) the RECV procedure pushes the content to encrypt ended by the associated protocol byte. The returned payload by the SEND procedure is a record packet, including the encrypted content.

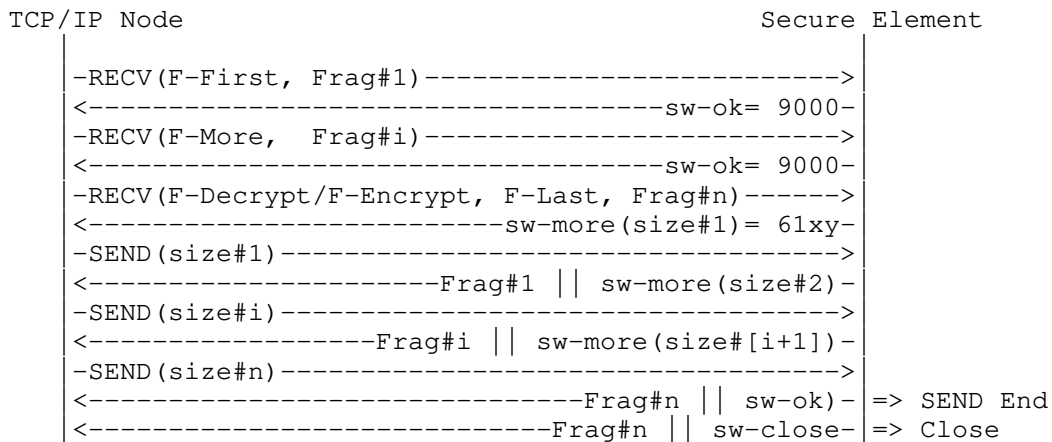


Figure 9. Decryption/Encryption operations.

### 3.3 TLS state machine

The state machine manages TLS flights, it determines the next record packet to be received and checked, and the next record packet to be built and sent. The number of states and their order is dependent on the TLS-SE role (client or server), and on the supported working mode (pre shared key, server with certificate, server and client with certificate). Figure 10 details an example of state machine for TLS-SE server, using pre-shared key. The ordered list of states comprises: S-Ready, S-Extensions, S-SFinished, S-ClientCCS, S-CFinished and S-Open.



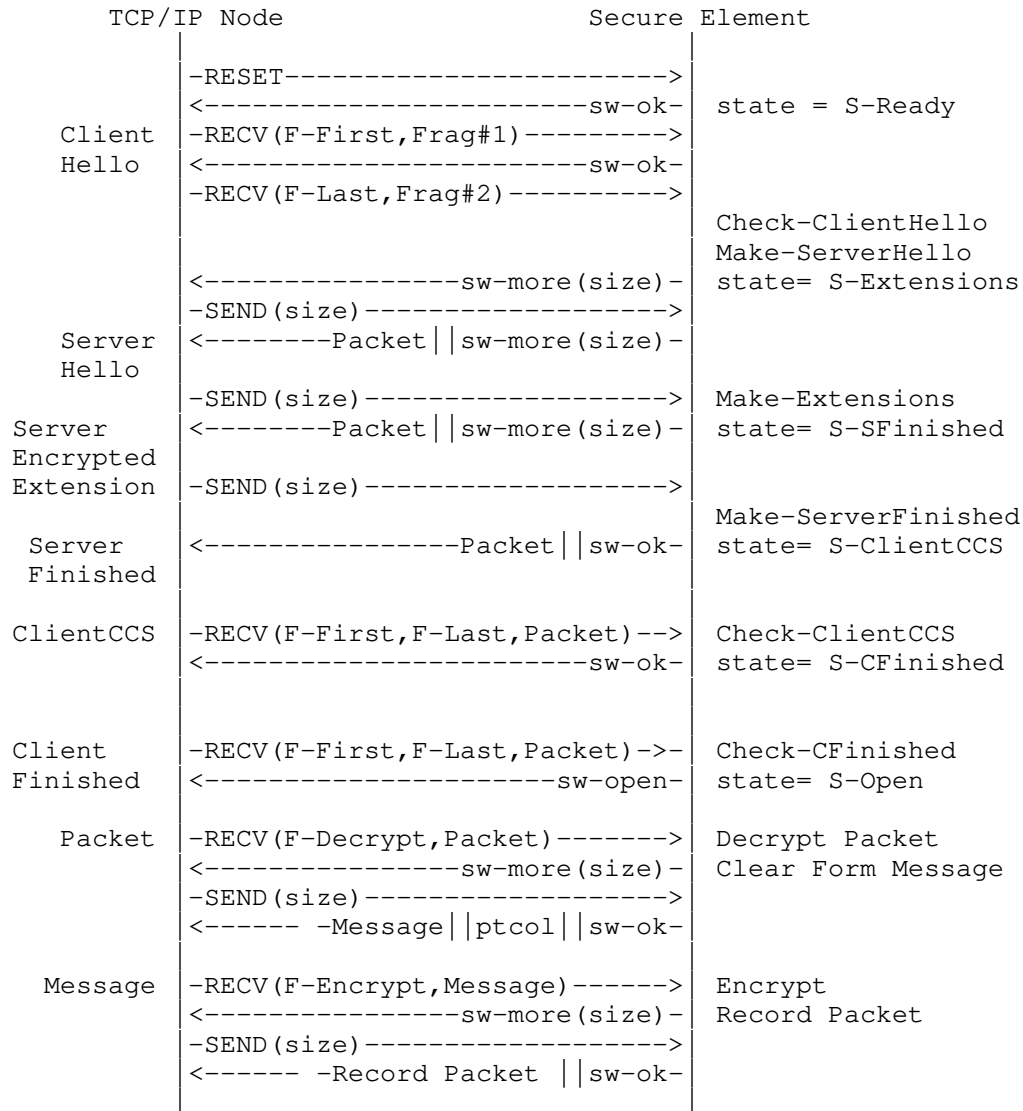


Figure 10. TLS-SE server with pre-share key state machine

### 3.4 TLS library

The TLS-SE library is a set of procedures that check, according to the state machine, incoming record packets and build outgoing record packets. In figure 10 the TLS library comprises the following elements: Check-ClientHello, Check-ClientCCS, Check-ClientFinished, Make-ServerHello, Make-EncryptedExtensions, and MakeServerFinished.

## 4 ISO7816 interface

The RECV and SEND binary encoding is shown by figure 11

name	CLA	INS	P1	P2	P3	Payload
RECV	00	D8	01= Decrypt 02= Encrypt	01= First 02= Last	Fragment Length 0= RESET	Yes
SEND	00	C0	00	00	Incoming Length	No

Figure 11. RECV and SEND ISO7816 requests binary encoding

The status word binary encoding is shown by figure 12. Two binary encoding of sw-more MUST be supported. In the T=0 context, SE operating system returns the 61xy status when a request including a payload, induces a response with a payload. The status 9Fxy is managed by the application in order to notify response size to be returned. The TLS-SE application MAY use 61xy status, but this could induce interoperability issues.

name	SW1	SW2
sw-ok	90	00
sw-more(size)	61 9F	size size
sw-retry(size)	6C +	size
sw-open	90	01
sw-close	90	02
sw-error	6D 6F	error number

Figure 12. ISO7816 status word binary encoding

## 5 ISO 7816 Use Case

An open implementation is available at [TLS-SE].

Below is an illustration of TLS-SE server, using a pre-shared key (PSK) with DHE over the secp256r1 curve, and the cipher suite AES-128-CCM-SHA256. The time consumed by handshake is about 1.4s.

PSK=

0102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F20

DHE=

037E6E633541EC03DB700A28E7DABB74F8E84D4A28E5F024B46F468A7821305D

RECV(Client Hello)

```
Tx: 00 D8 00 01 F0 16 03 03 00 F2 01 00 00 EE 03 03
    4E 65 53 05 52 AB 3E 83 14 0B 2F 9C 2F D7 BC 16
    F9 F5 C4 A9 86 CA 3F C8 8C 6E 8C D1 10 BB B1 57
    00 00 02 13 04 01 00 00 C3 00 2D 00 03 02 00 01
    00 2B 00 03 02 03 04 00 0D 00 1E 00 1C 06 03 05
    03 04 03 02 03 08 06 08 0B 08 05 08 0A 08 04 08
    09 06 01 05 01 04 01 02 01 00 33 00 47 00 45 00
    17 00 41 04 9A 1E 0A D8 40 88 D4 21 D1 55 D7 F2
    8F 78 4C 28 75 F5 19 CA 12 71 96 92 C4 07 8F B4
    35 42 57 E7 64 24 C1 BC 5D 89 0E F4 08 FD 25 8D
    24 F4 64 BB C3 F4 80 D3 BF 2C 23 A0 F9 2D A7 88
    0C 5B 44 53 00 0A 00 06 00 04 00 18 00 17 00 29
    00 3A 00 15 00 0F 43 6C 69 65 6E 74 5F 69 64 65
    6E 74 69 74 79 00 00 00 00 00 21 20 CC 05 4A 9F
    DE 70 E9 96 D6 01 69 61 F5 9A 78 20 D9 FC 6D ED
    4C C6 0A 7B 0D
```

Rx: 90 00 [47 ms]

Tx: 00 D8 00 02 07 4B 68 8F 4E B9 B2 CA

Rx: 61 86 [879 ms]

SEND(Server Hello)

Tx: 00 C0 00 00 86

```
Rx: 16 03 03 00 81 02 00 00 7D 03 03 5C 78 A4 E1 93
    34 D7 D9 64 B2 85 64 1B E4 76 63 94 39 1F 4A 15
    27 0A A4 C6 A0 C6 93 D9 E2 16 4D 00 13 04 00 00
    55 00 29 00 02 00 00 00 33 00 45 00 17 00 41 04
    25 C9 16 94 8B 39 51 D2 8E 88 70 F7 F5 4E 6C 31
    62 93 B1 65 55 2C 30 B2 5E 75 6C D8 FE AF DA A7
    67 D8 AD A7 BE 68 54 EA 3E A0 0B 4D CC 62 93 96
    38 07 68 29 3E D5 E6 0C 25 4A EA 12 C9 F8 99 7F
    00 2B 00 02 03 04 9F 1C [32 ms]
```

SEND(Server Encrypted Extensions)

Tx: 00 C0 00 00 1C

Rx: 17 03 03 00 17 E6 04 4A 52 1A 50 B5 54 D8 73 5E  
00 F4 FD 66 BB B3 74 50 99 36 C8 08 9F 3A [78 ms]

SEND(Server Encrypted Finished)

Tx: 00 C0 00 00 3A

Rx: 17 03 03 00 35 CB CA 03 3E E4 34 7E D2 0C 7C 24  
C1 8F 39 A2 74 39 24 47 78 BE 94 95 7A 31 EC 03  
D5 0C A8 1C 46 04 05 F2 83 3E 99 0D AD D6 66 63  
60 23 F8 5D 7B 77 0F 95 18 35 90 00 [185 ms]

RECV(Client Encrypted Finished)

Tx: 00 D8 00 03 3A 17 03 03 00 35 BC 29 18 D1 B8 4B  
C0 3F 6F 81 79 D9 7E FD 58 E3 76 EA 61 13 9C 3E  
40 0F 34 CD 94 CE C1 44 CB 76 70 7D DA 8A 54 69  
41 D9 80 CD 5D 52 8F E5 38 D8 52 92 20 54 5E  
Rx: 90 01 [389 ms]

TLS13 session is open

Decryption of incoming Record Packet

RECV(Decrypt, Packet)

Tx: 00 D8 01 03 24 17 03 03 00 1F 56 E2 D5 B5 C4 A6  
E2 3E 54 56 5A C4 2D E9 99 F3 58 22 34 15 15 A7  
96 FD 0E B0 61 60 4C 52 87  
Rx: 61 0F [78 ms]

SEND(Message)

Tx: 00 C0 00 00 0F

Rx: 68 65 6C 6C 6F 20 77 6F 72 6C 64 21 0D 0A 17 90  
00 [15 ms]  
Rx: hello world! ptcol=17

Encryption of message

RECV(Encrypt, Message)

Tx: 00 D8 02 03 0F 68 65 6C 6C 6F 20 77 6F 72 6C 64  
21 0D 0A 17  
Rx: 61 24 [79 ms]

SEND(Record Packet)

Tx: 00 C0 00 00 24

Rx: 17 03 03 00 1F 6F 78 FF 68 0F CA 9E 31 53 2C 96  
B3 FA D7 B0 51 1B 92 81 35 3D DB FE E9 18 A7 DF  
36 2F A5 27 90 00 [16 ms]

## 5 TLS-SE Name

According to ISO7816 standards, secure elements return upon reset a set of bytes called Answer to Reset (ATR). ATR comprises at least two bytes (TS, T0). The LSB nibble of T0 indicates the number of historical bytes (ranging from 0 to 15). Historical bytes (HB) are located at the end of ATR. Historical bytes can be programmed by standardized API, and therefore MAY be used for secure element naming.

## 6 Server Name Indication

According to [RFC6066] Server Name Indication extension is used to route TLS packets toward a virtual host. Multiple TLS-SE devices, embedding standalone applications, can be hosted by an internet node. In this case SNI extension MAY be used in order to select the right secure element, whose name, typically stored in historical bytes, is determined from SNI.

## 7 IANA Considerations

This draft does not require any action from IANA.

## 8 Security Considerations

This entire document is about security.

## 9 References

### 9.1 Normative References

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC6066] Eastlake 3rd, D., "Transport Layer [RFC6066] Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011.

[ISO7816] ISO 7816, "Cards Identification - Integrated Circuit Cards with Contacts", The International Organization for Standardization (ISO).

[IEEE1363] IEEE, "IEEE Standard Specifications for Public Key Cryptography", IEEE Std. 1363-2000, DOI 10.1109/IEEE STD.2000.92292.

## 9.2 Informative References

[IM] Urien, P., "Identity Module for TLS Version 1.3", draft-urien-tls-im-04.txt, July 2021.

[TLS-SE] "tls-se.java", <https://github.com/purien/TLS-SE>

## 10 Authors' Addresses

Pascal Urien  
Telecom Paris  
19 place Marguerite Perey  
91120 Palaiseau Phone: NA  
France Email: Pascal.Urien@telecom-paris.fr