Workgroup: Network Working Group
Internet-Draft:
draft-dekater-scion-dataplane-01
Published: 4 March 2024
Intended Status: Informational
Expires: 5 September 2024
Authors: C. de Kater
SCION Association
N. Rustignoli
SCION Association
S. Hitz
Anapaya Systems

SCION Data Plane

Abstract

This document describes the data plane of the path-aware, inter-domain network architecture SCION (Scalability, Control, and Isolation On Next-generation networks). One of the basic characteristics of SCION is that it gives path control to endpoints. The SCION control plane is responsible for discovering these paths and making them available as path segments to the endpoints. The responsibility of the **SCION data plane** is to combine the path segments into end-to-end paths, and forward data between endpoints according to the specified path.

The SCION data plane fundamentally differs from today's IP-based data plane in that it is *path-aware*: In SCION, interdomain forwarding directives are embedded in the packet header. This document first provides a detailed specification of the SCION data packet format as well as the structure of the SCION header. SCION also supports extension headers - these are described, too. The document then shows the life cycle of a SCION packet while traversing the SCION Internet. This is followed by a specification of the SCION path authorization mechanisms and the packet processing at routers. SCION also includes its own protocol to communicate failures to endpoints, the SCION Control Message Protocol (SCMP). This protocol will be described in a separate document, which will follow later.

About This Document

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

The latest revision of this draft can be found at https://scionassociation.github.io/scion-dp_I-D/draft-dekater-scion-dataplane.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-dekater-scion-dataplane/.

Source for this draft and an issue tracker can be found at https://github.com/scionassociation/scion-dp_I-D.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

SCION leverages source-based path selection, where path information is embedded in the packet header - this is called packet-carried forwarding state (PCFS). This section explains how data packets are forwarded through the network, how the SCION inter-domain routing differs from intra-domain routing, and how endpoints can construct end-to-end paths from path segments. It also briefly touches the concept of path authorization, which ensures that data packets always traverse the network along authorized paths.

Note: This is the very first version of the SCION Data Plane document. We are aware that the draft is far from perfect, and hope to improve the content in later versions of the document. To reach this goal, any feedback is welcome and much appreciated. Thanks!

Note: It is assumed that readers of this draft are familiar with the basic concepts of the SCION next-generation inter-domain network architecture. If not, please find more detailed information in the

IETF Internet Drafts [I-D.scion-overview], [I-D.scion-components], [I-D.scion-cppki], and [I-D.scion-cp], as well as in [CHUAT22], especially Chapter 2. A short description of the SCION basic terms and elements can be found in Section 1.1 below.

1.1. Terminology

Autonomous System (AS): An autonomous system is a network under a common administrative control. For example, the network of an Internet service provider, company, or university can constitute an AS.

Core AS: Each SCION isolation domain (ISD) is administered by a set of distinguished autonomous systems (ASes) called core ASes, which are responsible for initiating the path-discovery and -construction process (in SCION called "beaconing").

Data Plane: The data plane (sometimes also referred to as the forwarding plane) is responsible for forwarding data packets that endpoints have injected into the network. After routing information has been disseminated by the control plane, packets are forwarded according to such information by the data plane.

Endpoint: An endpoint is the start- or the endpoint of a SCION path. For example, an endpoint can be a host as defined in [RFC1122], or a gateway bridging a SCION and an IP domain. This definition is based on the definition in [RFC9473].

Forwarding Key: A forwarding key is a symmetric key that is shared between the control service (control plane) and the routers (data plane) of an AS. It is used to authenticate hop fields in the end-to-end SCION path. The forwarding key is an AS-local secret and is not shared with other ASes.

Forwarding Path: A forwarding path is a complete end-to-end path between two SCION hosts, which is used to transmit packets in the data plane and can be created with a combination of up to three path segments (an up-segment, a core-segment, and a down-segment).

Hop Field (HF): As they traverse the network, path-segment construction beacons (PCBs) accumulate cryptographically protected AS-level path information in the form of hop fields. In the data plane, hop fields are used for packet forwarding: they contain the incoming and outgoing interface IDs of the ASes on the forwarding path.

Info Field (INF): Each path-segment construction beacon (PCB) contains a single info field, which provides basic information about the PCB. Together with hop fields (HFs), info fields are used to create forwarding paths.

Isolation Domain (ISD): In SCION, autonomous systems (ASes) are organized into logical groups called isolation domains or ISDs. Each ISD consists of ASes that span an area with a uniform trust environment (e.g., a common jurisdiction). A possible model is for ISDs to be formed along national boundaries or federations of nations.

Leaf AS: An AS at the "edge" of an ISD, with no other downstream ASes.

MAC: Message Authentication Code. In the rest of this document, "MAC" always refers to "Message Authentication Code" and never to "Medium Access Control". When "Medium Access Control address" is implied, the phrase "Link Layer Address" is used.

Packet-Carried Forwarding State (PCFS): Rather than relying on inter-domain forwarding tables, SCION data packets contain all the necessary, cryptographically-protected path information. This property is referred to as packet-carried forwarding state.

Path Authorization: A requirement for the data plane is that endpoints can only use paths that were constructed and authorized by ASes in the control plane. This property is called path authorization. The goal of path authorization is to prevent endpoints from crafting hop fields (HFs) themselves, modifying HFs in authorized path segments, or combining HFs of different path segments.

Path Control: Path control is a property of a network architecture that gives endpoints the ability to select how their packets travel through the network. Path control is stronger than path transparency.

Path Segment: Path segments are derived from path-segment construction beacons (PCBs). A path segment can be (1) an up-segment (i.e., a path between a non-core AS and a core AS in the same ISD), (2) a down-segment (i.e., the same as an up-segment, but in the opposite direction), or (3) a core-segment (i.e., a path between core ASes). Up to three path segments can be used to create a forwarding path.

Path-Segment Construction Beacon (PCB): Core ASes generate PCBs to explore paths within their isolation domain (ISD) and among different ISDs. ASes further propagate selected PCBs to their neighboring ASes. As a PCB traverses the network, it carries path segments, which can subsequently be used for traffic forwarding.

Path Transparency: Path transparency is a property of a network architecture that gives endpoints full visibility over the network

paths their packets are taking. Path transparency is weaker than path control.

SCMP: SCION Control Message Protocol. SCMP is used for signaling connectivity problems, analogous to the Internet Control Message Protocol (ICMP). SCMP provides network diagnostic and error messages.

Valley Route: A valley route contains ASes that do not profit economically from traffic on this route. The name comes from the fact that such routes go "down" (following parent-child links) before going "up" (following child-parent links).

1.2. Conventions and Definitions

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. Overview

The SCION data plane forwards inter-domain packets between ASes. SCION routers are normally deployed at the edge of an AS, and peer with neighbor SCION routers. Inter-domain forwarding is based on end-to-end path information contained in the packet header. This path information consists of a sequence of hop fields (HFs). Each hop field corresponds to an AS on the path, and it includes an ingress- as well as an egress interface ID, which univocally identify the ingress and egress interfaces within the AS. The information is authenticated with a Message Authentication Code (MAC) to prevent forgery.

This concept allows SCION routers to forward packets to a neighbor AS without inspecting the destination address and also without consulting an inter-domain forwarding table. Intra-domain forwarding and routing are based on existing mechanisms (e.g., IP). A SCION border router reuses existing intra-domain infrastructure to communicate to other SCION routers or SCION endpoints within its AS. The last SCION router at the destination AS therefore uses the destination address to forward the packet to the appropriate local endpoint.

This SCION design choice has the following advantages:

* It provides control and transparency over forwarding paths to endpoints.

*It simplifies the packet-processing at routers: Instead of having to perform longest-prefix matching on IP addresses, which requires expensive hardware and substantial amounts of energy, a router can simply access the next hop from the packet header, after having verified the authenticity of the hop field's MAC.

1.3.1. Inter- and Intra-Domain Forwarding

As SCION is an inter-domain network architecture, it is not concerned with intra-domain forwarding. This corresponds to the general practice today where BGP and IP are used for inter-domain routing and forwarding, respectively, but ASes use an intra-domain protocol of their choice, for example OSPF or IS-IS for routing and IP, MPLS, and various layer-2 protocols for forwarding. In fact, even if ASes use IP-forwarding internally today, they typically encapsulate the original IP packet they receive at the edge of their network into another IP packet with the destination address set to the egress border router, to avoid full inter-domain forwarding tables at internal routers.

SCION emphasizes this separation, as SCION is used exclusively for inter-domain forwarding, and re-uses the intra-domain network fabric to provide connectivity among all SCION infrastructure services, border routers, and endpoints. As a consequence, minimal change to the infrastructure is required for ISPs when deploying SCION.

Although a complete SCION address is composed of the <ISD, AS, endpoint address> 3-tuple, the endpoint address is not used for inter-domain routing or forwarding. This implies that the endpoint addresses are not required to be globally unique or globally routable, they can be selected independently by the corresponding ASes. This means, for example, that an endpoint identified by a link-local IPv6 address in the source AS can directly communicate with an endpoint identified by a globally routable IPv4 address via SCION. Alternatively, it is possible for two SCION hosts with the same IPv4 address 10.0.0.42 but located in different ASes to communicate with each other via SCION ([RFC1918]).

1.3.2. Intra-Domain Forwarding Process

The full forwarding process for a packet transiting an intermediate AS consists of the following steps.

Note: In this context, a border router is called **ingress** border router when it refers to an entrance border router to an AS, as seen from the direction of travel of the SCION packet. A border router is

called **egress** border router when it refers to an exit border router of an AS, as seen from the direction of travel of the SCION packet.

1. The AS's SCION ingress router receives a SCION packet from the neighboring AS.

2. The SCION router parses, validates, and authenticates the SCION header.

3. The SCION router maps the egress interface ID in the current hop field of the SCION header to the destination "intra-protocol" address of the egress border router (where "intra-protocol" is the intra-domain forwarding protocol, e.g., MPLS or IP).

4. The packet is forwarded within the AS by routers and switches based on the "intra-protocol" header.

5. Upon receiving the packet, the SCION egress router strips off the "intra-protocol" header, again validates and updates the SCION header, and forwards the packet to the neighboring SCION router.

Note: The current SCION implementation uses the UDP/IP protocol as underlay. However, the use of other underlay protocols is possible and allowed.

1.4. Path Construction (Segment Combinations)

Paths are discovered by the SCION control plane, which makes them available to SCION endpoints in the form of path segments. As described in [I-D.scion-cp], there are three kinds of path segments: up, down, and core. In the data plane, a SCION endpoint creates end-to-end paths from the path segments, by combining multiple path segments. Depending on the network topology, a SCION forwarding path can consist of one, two, or three segments. Each path segment contains several hop fields representing the ASes on the segment as well as one info field with basic information about the segment, such as a timestamp.

Segments cannot be combined arbitrarily. To construct a valid forwarding path, the source endpoint **MUST** obey the following rules:

- There can be at most one of each type of segment (up-, core-, and down-segment). Allowing multiple up- or down-segments would decrease efficiency and the ability of ASes to enforce path policies.

- If an up-segment is present, it **MUST** be the first segment in the path.

- If a down-segment is present, it **MUST** be the last segment in the path.

- If there are two path segments (one up- and one down-segment) that both announce the same peering link, then a shortcut via this peering link is possible.

- If there are two path segments (one up- and one down-segment) that share a common ancestor AS (in the direction of beaconing), then a shortcut via this common ancestor AS is possible.

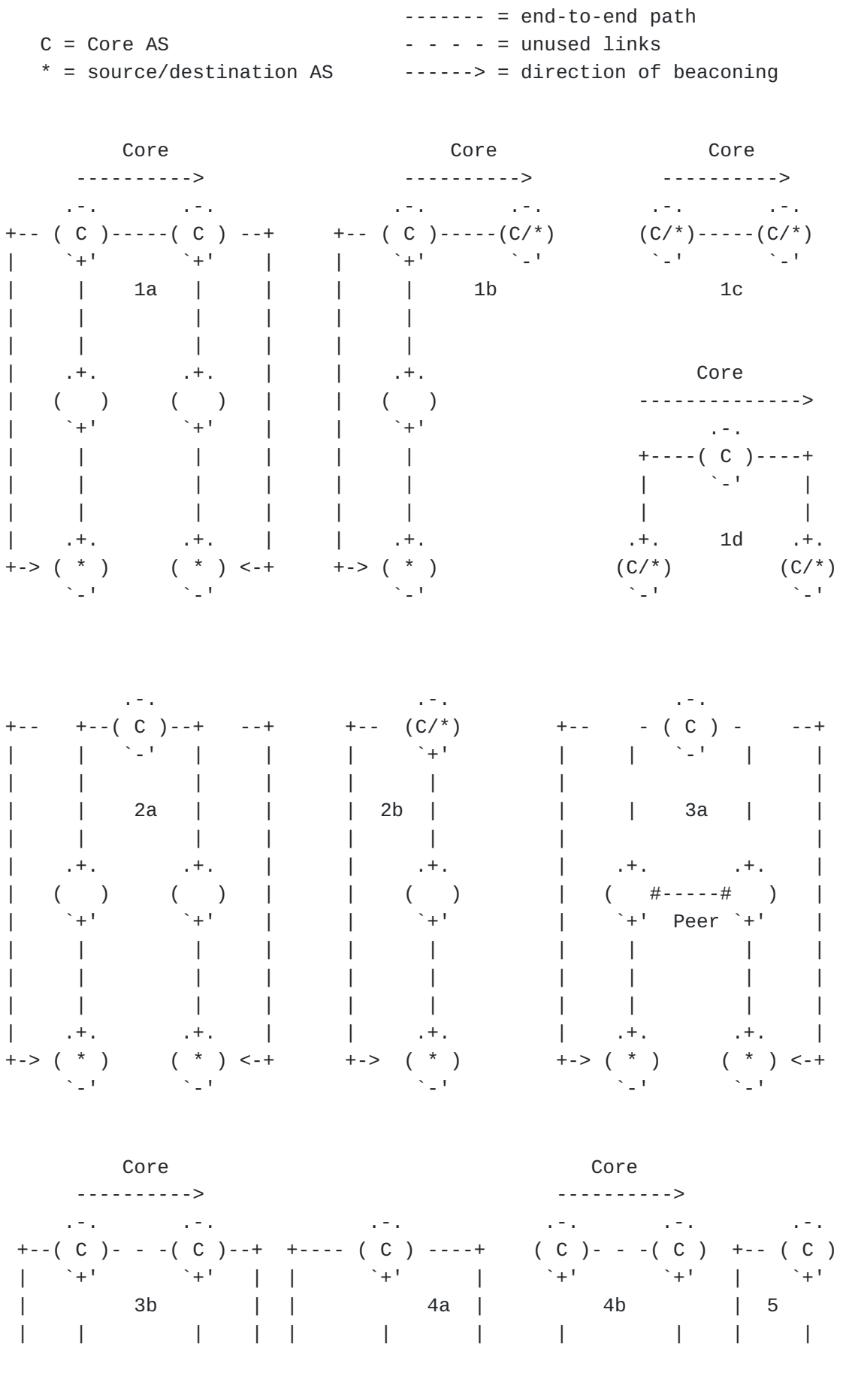
- Additionally, all segments without any peering possibility **MUST** consist of at least two hop fields.

Note: The type of segment is known to the endpoint but is not visible in the path header of data packets. Therefore, a SCION router needs to explicitly verify that these rules were followed correctly.

Besides enabling the enforcement of path policies, the above rules also protect the economic interest of ASes, as they prevent building "valley paths". A valley path contains ASes that do not profit economically from traffic on this route. The name comes from the fact that such paths go "down" (following parent-child links) before going "up" (following child-parent links).

Figure 1 below shows the different allowed segment combinations.

Note: It is assumed that the source and destination endpoints are in different ASes (as endpoints from the same AS use an empty forwarding path to communicate with each other).


------- = end-to-end path
C = Core AS
- - - - = unused links
* = source/destination AS
------> = direction of beaconing
Core
Core
Core
1a
1b
1c
Core
1d
2a
2b
3a
Peer
Core
Core
3b
4a
4b
5

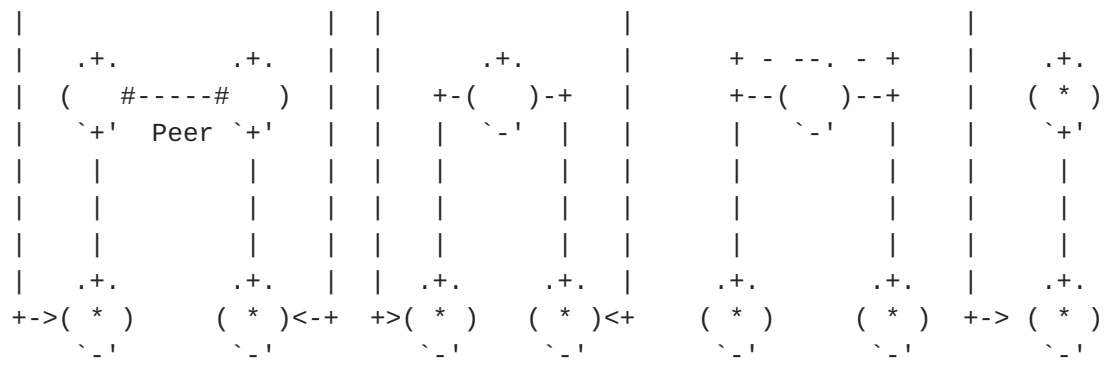


Figure 1: Illustration of possible path-segment combinations. Each node represents a SCION Autonomous System.

The following path-segment combinations are allowed:

* **Communication through core ASes**:
  - **Core-segment combination** (Cases 1a, 1b, 1c, 1d in Figure 1): The up- and down-segments of source and destination do not have an AS in common. In this case, a core-segment is required to connect the source's up- and the destination's down-segment (Case 1a). If either the source or the destination AS is a core AS (Case 1b) or both are core ASes (Cases 1c and 1d), then no up- or down-segment(s) are required to connect the respective AS(es) to the core-segment.
  - **Immediate combination** (Cases 2a, 2b in Figure 1): The last AS on the up-segment (which is necessarily a core AS) is the same as the first AS on the down-segment. In this case, a simple combination of up- and down-segments creates a valid forwarding path. In Case 2b, only one segment is required.
* **Peering shortcut** (Cases 3a and 3b): A peering link exists between the up- and down-segment. The extraneous path segments to the core are cut off. Note that the up- and down-segments do not need to originate from the same core AS and the peering link could also be traversing to a different ISD.
* **AS shortcut** (Cases 4a and 4b): The up- and down-segments intersect at a non-core AS below the ISD core, thus creating a shortcut. In this case, a shorter path is made possible by removing the extraneous part of the path to the core. Note that the up- and down-segments do not need to originate from the same core AS and can even be in different ISDs (if the AS at the intersection is part of multiple ISDs).
* **On-path** (Case 5): In the case where the source's up-segment contains the destination AS or the destination's down-segment contains the source AS, a single segment is sufficient to

construct a forwarding path. Again, no core AS is on the final path.

1.5. Path Authorization

The SCION data plane provides *path authorization*. This property ensures that data packets always traverse the network using path segments that were explicitly authorized by the respective ASes, and prevents endpoints from constructing unauthorized paths or paths containing loops. SCION uses symmetric cryptography in the form of Message Authentication Codes (MACs) to authenticate the information encoded in hop fields. Such MACs are verified by routers at forwarding. For a detailed specification, see Section 4.

2. SCION Header Specification

This section provides a detailed specification of the SCION packet header. SCION also supports extension headers - these are described, too.

2.1. SCION Packet Header Format

The SCION packet header is composed of a common header, an address header, a path header, and an **OPTIONAL** extension header, see Figure 2 below.

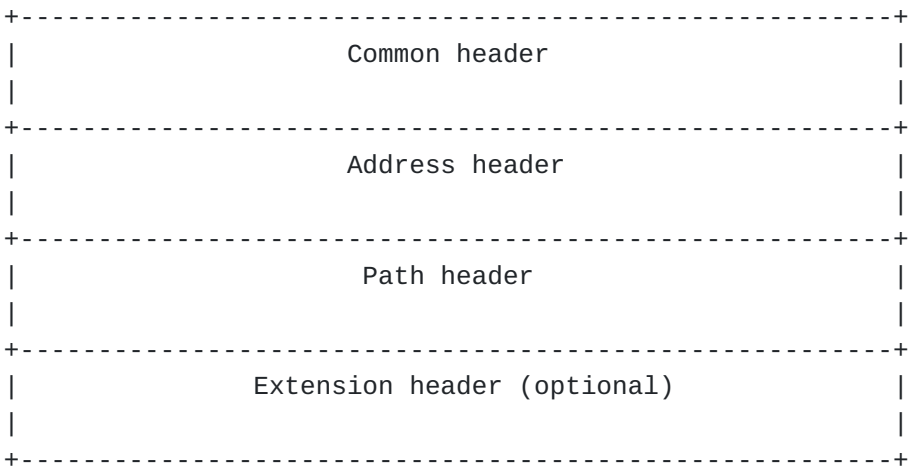


Figure 2: High-level SCION header structure

The *common header* contains important meta information like a version number and lengths of the header and payload. In particular, it contains flags that control the format of subsequent headers such as the address and path headers. For more details, see Section 2.2.1.

The *address header* contains the ISD-, AS-, and endpoint-addresses of source and destination. The type and length of endpoint addresses

are variable and can be set independently using flags in the common header. For more details, see Section 2.2.2.

The *path header* contains the full AS-level forwarding path of the packet. A path type field in the common header specifies the path format used in the path header. For more details, see Section 2.2.3.

Finally, the optional *extension* header contains a variable number of hop-by-hop and end-to-end options, similar to the extensions in the IPv6 header [RFC8200]. For more details, see Section 2.3.

2.1.1. Header Alignment

The SCION header is aligned to 4 bytes.

2.2. SCION Header Structure

2.2.1. Common Header

The SCION common header has the following packet format:

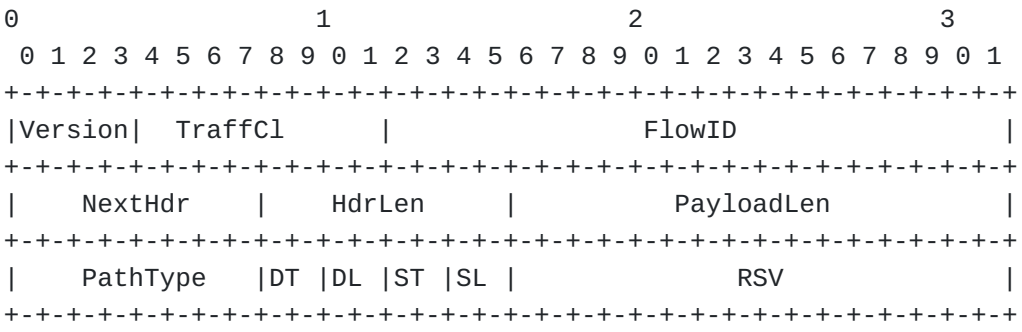


Figure 3: The SCION common header packet format

*Version: The version of the SCION common header. Currently, only version "0" is supported.

*TrafficClass (TraffCl in the image above): The 8-bit long identifier of the packet's class or priority. The value of the traffic class bits in a received packet or fragment might differ from the value sent by the packet's source. The current use of the TrafficClass field for Differentiated Services and Explicit Congestion Notification is specified in [RFC2474] and [RFC3168].

*FlowID: This 20-bit field labels sequences of packets to be treated in the network as a single flow. Sources **MUST** set this field.

*NextHdr: Encodes the type of the first header after the SCION header. This can be either a SCION extension or a layer-4

protocol such as TCP or UDP. Values of this field respect the Assigned SCION Protocol Numbers (see Appendix "Assigned SCION Protocol Numbers").

*HdrLen: Specifies the entire length of the SCION header in bytes, i.e., the sum of the lengths of the common header, the address header, and the path header. All SCION header fields are aligned to a multiple of 4 bytes. The SCION header length is computed as HdrLen * 4 bytes. The 8 bits of the HdrLen field limit the SCION header to a maximum of 255 * 4 = 1020 bytes.

*PayloadLen: Specifies the length of the payload in bytes. The payload includes (SCION) extension headers and the L4 payload. This field is 16 bits long, supporting a maximum payload size of 65'535 bytes.

*PathType: Specifies the type of the SCION path. It is possible to specify up to 256 different types. The format of one path type is independent of all other path types. The currently defined SCION path types are Empty (0), SCION (1), OneHopPath (2), EPIC (3) and COLIBRI (4). This document only specifies the Empty, SCION and OneHopPath path types. The other path types are currently experimental.

Value	Path Type
0	Empty path (EmptyPath)
1	SCION (SCION)
2	One-hop path (OneHopPath)
3	EPIC path (experimental)
4	COLIBRI path (experimental)

Table 1: SCION path types

*DT/DL/ST/SL: These fields define the endpoint-address type and endpoint-address length for the source and destination endpoint. DT and DL stand for Destination Type and Destination Length, whereas ST and SL stand for Source Type and Source Length. The possible endpoint address length values are 4 bytes, 8 bytes, 12 bytes, and 16 bytes. If some address has a length different from the supported values, the next larger size can be used and the address can be padded with zeros. Table 2 below lists the currently used values for address length. The "type" identifier is only defined in combination with a specific address length. For example, address type "0" is defined as IPv4 in combination with address length 4, but in combination with address length 16, it stands for IPv6. Per address length, several sub-types are possible. Table 3 shows the currently valid allocations of type values to length values.

DL/SL Value	Address Length
0	4 bytes
1	8 bytes
2	12 bytes
3	16 bytes

Table 2: Address length values

Length (bytes)	Type	Type/Length (binary)	Interpretation
4	0	0b0000	IPv4
4	1	0b0100	Service
16	0	0b1100	IPv6
other			Unassigned

Table 3: Allocations of type values to length values

*RSV: These bits are currently reserved for future use.

2.2.2. Address Header

The SCION address header has the following format:

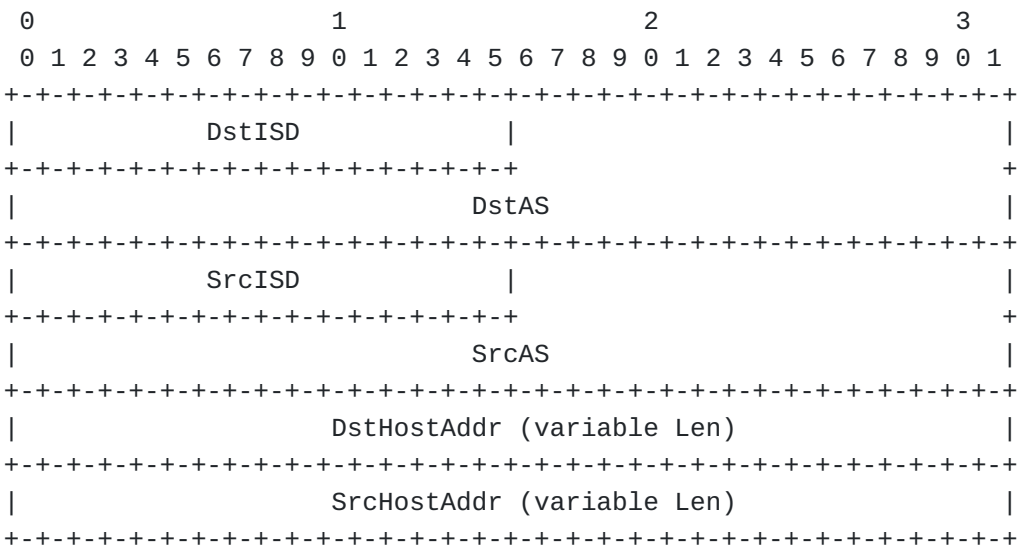


Figure 4: The SCION address header packet format

*DstISD, SrcISD: The 16-bit ISD identifier of the destination/ source.

*DstAS, SrcAS: The 48-bit AS identifier of the destination/source.

*DstHostAddr, SrcHostAddr: Specifies the variable length endpoint address of the destination/source. The accepted type and length are defined in the DT/DL/ST/SL fields of the common header.

Note: For more information on addressing in SCION, see the introduction of the SCION Control Plane Specification ([I-D.scion-cp]).

2.2.3. Path Header

The path header of a SCION packet differs for each SCION path type.

Note: The path type is set in the PathType field of the SCION common header.

Currently, SCION supports three path types:

- The Empty path type (PathType=0). For more information, see Section 2.2.3.1.
- The SCION path type (PathType=1). This is the standard path type in SCION. For a detailed description, see Section 2.2.3.2.
- The OneHopPath path type (PathType=2). For more information, see Section 2.2.3.3.

2.2.3.1. Empty Path Type

The Empty path type is used to send traffic within an AS. It has no additional fields, i.e., it consumes 0 bytes on the wire.

One use case of the Empty path type lies in the context of link-failure detection. To this end, SCION uses the Bidirectional Forwarding Detection (BFD) protocol ([RFC5880] and [RFC5881]). BFD is a protocol intended to detect faults in the bidirectional path between two forwarding engines, with typically very low latency. It operates independently of media, data protocols, and routing protocols. SCION uses the Empty path type, together with OneHopPath path type, to bootstrap BFD within SCION. (For more information on the OneHopPath path type, see Section 2.2.3.3.)

2.2.3.2. SCION Path Type

This section specifies the standard SCION path type. A SCION path has the following layout:

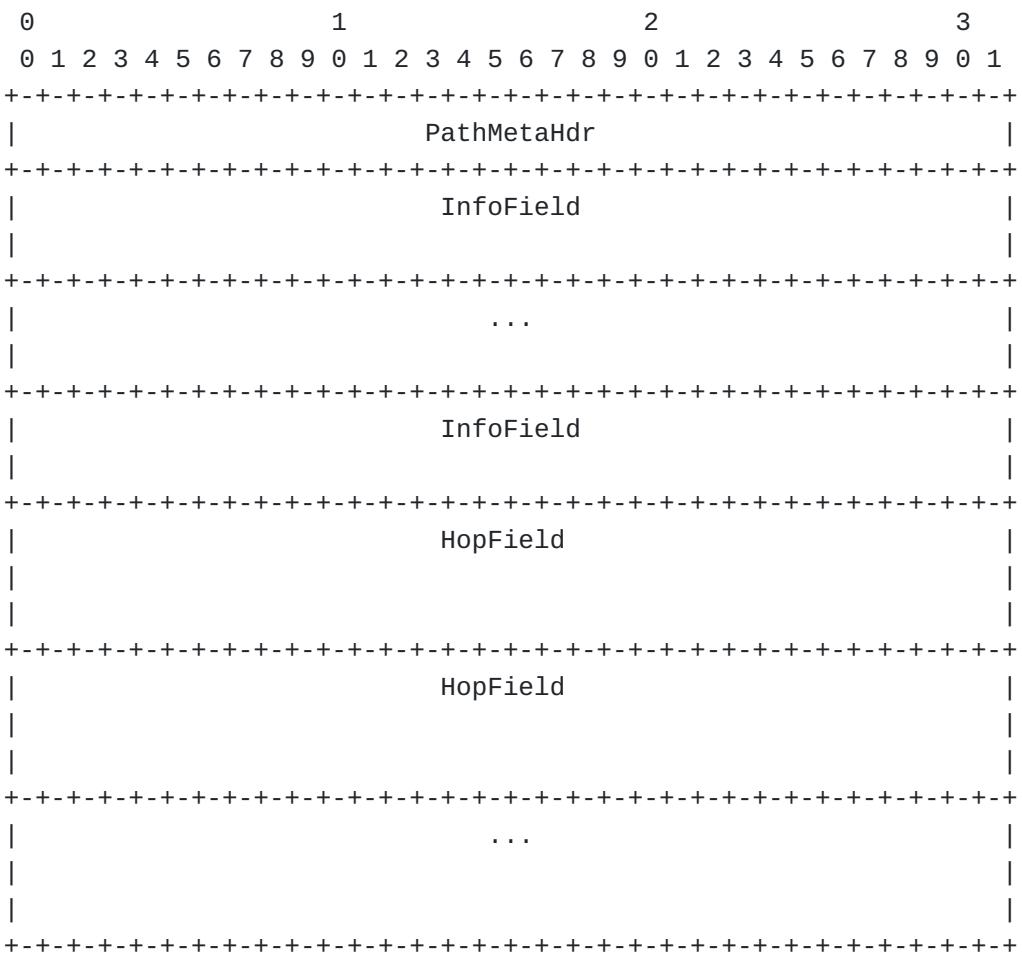


Figure 5: Layout of a standard SCION path

It consists of a path meta header, up to 3 info fields and up to 64 hop fields.

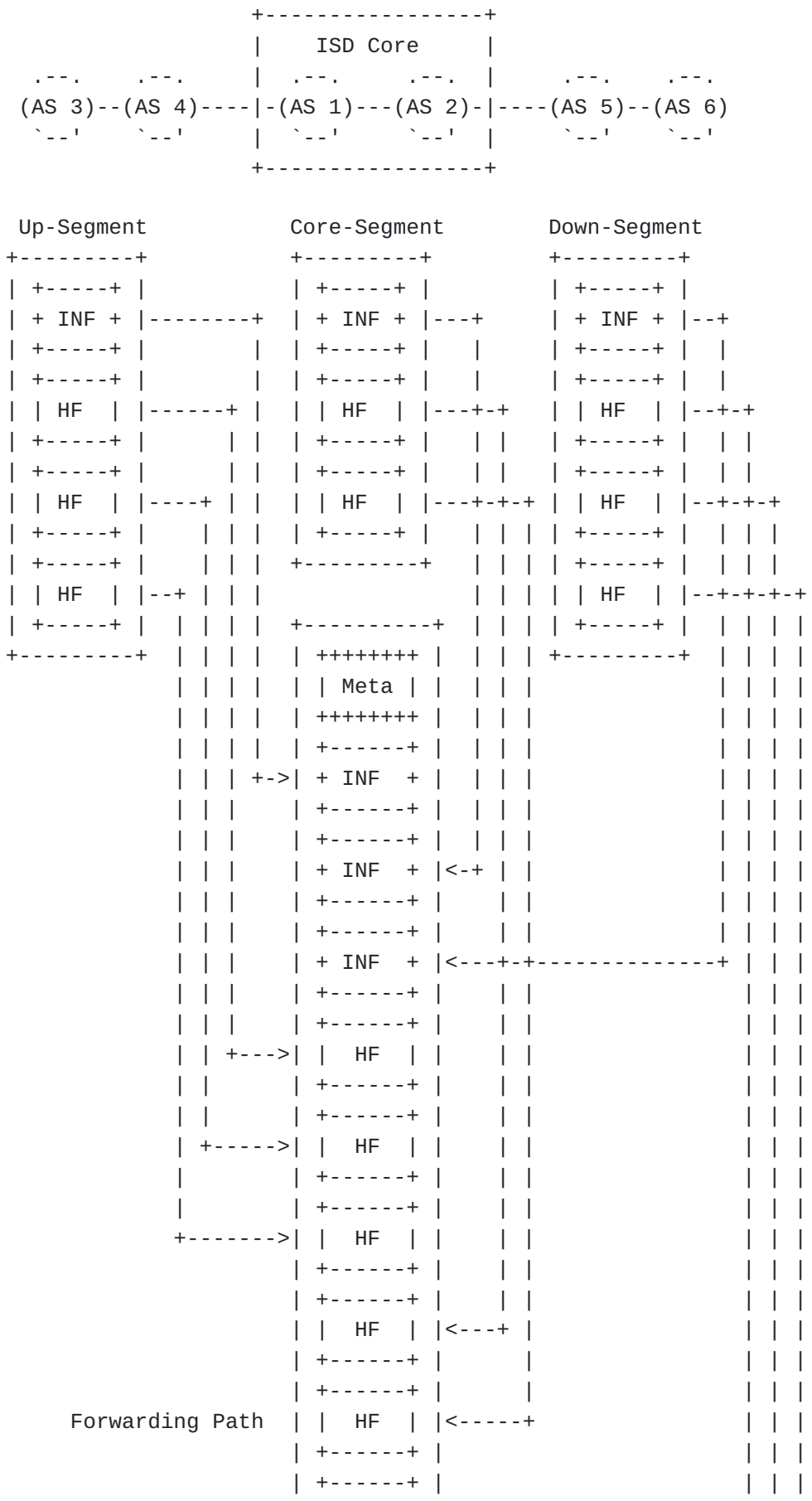
- The path meta header (PathMetaHdr) indicates the currently valid info field and hop field while the packet is traversing the network along the path, as well as the number of hop fields per segment.

- The number of info fields (InfoField) equals the number of path segments that the path contains - there is one info field per path segment. Each info field contains basic information about the corresponding segment, such as a timestamp indicating the creation time. There are also two flags. One specifies whether the segment is to be traversed in construction direction, the other whether the first or last hop field in the segment represents a peering hop field.

- Each hop field (HopField) represents a hop through an AS on the path, with the ingress and egress interface identifiers for this

AS. This information is authenticated with a Message Authentication Code (MAC) to prevent forgery.

The SCION header is created by extracting the required info fields and hop fields from the corresponding path segments. The process of extracting is illustrated in Figure 6 below. Note that ASes at the joints of multiple segments are represented by two hop fields. Be aware that these hop fields are not equal! In the hop field that represents the last hop in the first segment (seen in the direction of travel), only the ingress interface will be specified. However, in the hop field that represents the first hop in the second segment (also in the direction of travel), only the egress interface will be defined. Thus, the two hop fields for this one AS build a full hop through the AS, specifying both the ingress and egress interface. As such, they bring the two adjacent segments together.


ISD Core
(AS 1)
(AS 2)
(AS 3)
(AS 4)
(AS 5)
(AS 6)
Up-Segment
Core-Segment
Down-Segment
HF
INF
Meta
Forwarding Path

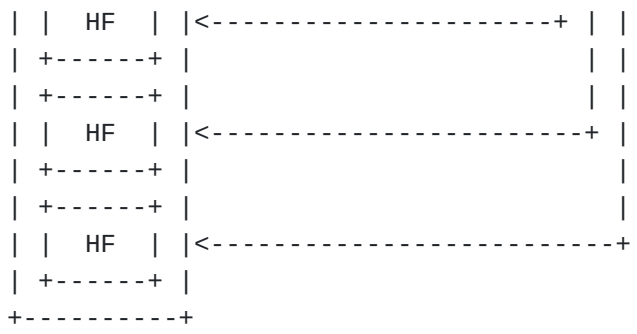


Figure 6: Path construction example

2.2.3.2.1. Path Meta Header Field

The 4-byte Path Meta Header field (PathMetaHdr) defines meta information about the SCION path that is contained in the path header. It has the following format:

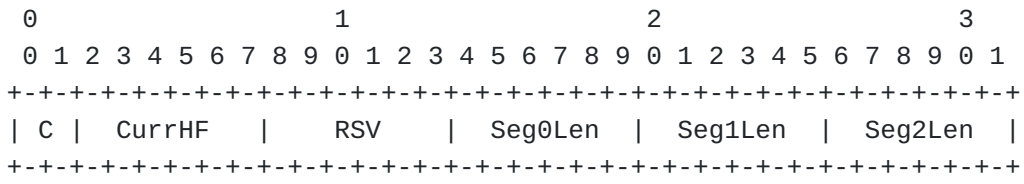


Figure 7: SCION path type - Format of the Path Meta Header field

- C (urrINF): Specifies a 2-bits index (0-based) pointing to the current info field for the packet on its way through the network. For details, see Section 2.2.3.2.2 below.

- CurrHF: Specifies a 6-bits index (0-based) pointing to the current hop field for the packet on its way through the network. For details, see Section 2.2.3.2.2 below. Note that the CurrHF index **MUST** point to a hop field that is part of the current path segment, as indicated by the CurrINF index.

Both indices are used by SCION routers when forwarding data traffic through the network. The SCION routers also increment the indexes if required. For more details, see Section 4.2.2.

- Seg{0,1,2}Len: The number of hop fields in a given segment. Seg{i}Len > 0 implies that segment *i* contains at least one hop field, which means that info field *i* exists. (If Seg{i}Len = 0 then segment *i* is empty, meaning that this path does not include segment *i*, and therefore there is no info field *i*.) The following rules apply:

  - The total number of hop fields in an end-to-end path **MUST** be equal to the sum of all Seg{0,1,2}Len contained in this end-to-end path.

-It is an error to have Seg{X}Len > 0 AND Seg{Y}Len == 0, where 2 >= *X* > *Y* >= 0. That is, if path segment Y is empty, the following path segment X **MUST** also be empty.

*RSV: Unused and reserved for future use.

2.2.3.2.2. Path Offset Calculations

The path offset calculations are used by the SCION border routers to determine the info field and hop field that are currently valid for the packet on its way through the network.

The following rules apply when calculating the path offsets:

```
if Seg2Len > 0: NumINF = 3
else if Seg1Len > 0: NumINF = 2
else if Seg0Len > 0: NumINF = 1
else: invalid
```

The offsets of the current info field and current hop field (relative to the end of the address header) are now calculated as:

```
B = byte
InfoFieldOffset = 4B + 8B.CurrINF
HopFieldOffset = 4B + 8B.NumINF + 12B.CurrHF
```

To check that the current hop field is in the segment of the current info field, the CurrHF needs to be compared to the SegLen fields of the current and preceding info fields.

2.2.3.2.3. Info Field

The 8-byte Info Field (InfoField) has the following format:

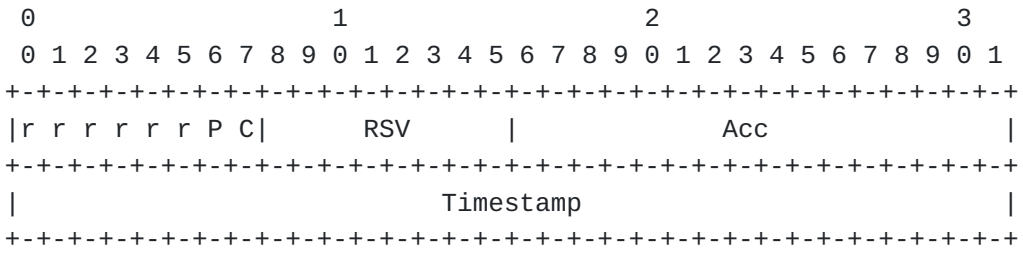


Figure 8: SCION path type - Format of the Info Field

*r: The r bits are unused and reserved for future use.

*P: Peering flag. If the flag has value "1", the segment represented by this info field contains a peering hop field,

which requires special processing in the data plane. For more details, see Section 4.1.2 and Section 4.2.

*C: Construction direction flag. If the flag has value "1", the hop fields in the segment represented by this info field are arranged in the direction they have been constructed during beaconing.

*RSV: Unused and reserved for future use.

*Acc: This updatable field/counter is required for calculating the MAC in the data plane. Acc stands for "Accumulator". For more details, see Section 4.1.

*Timestamp: Timestamp created by the initiator of the corresponding beacon. The timestamp is defined as the number of seconds elapsed since the POSIX Epoch (1970-01-01 00:00:00 UTC), encoded as a 32-bit unsigned integer. This timestamp enables the validation of a hop field in the segment represented by this info field, by verifying the expiration time and MAC set in the hop field - the expiration time of a hop field is calculated relative to the timestamp.

2.2.3.2.4. Hop Field

The 12-byte Hop Field (HopField) has the following format:

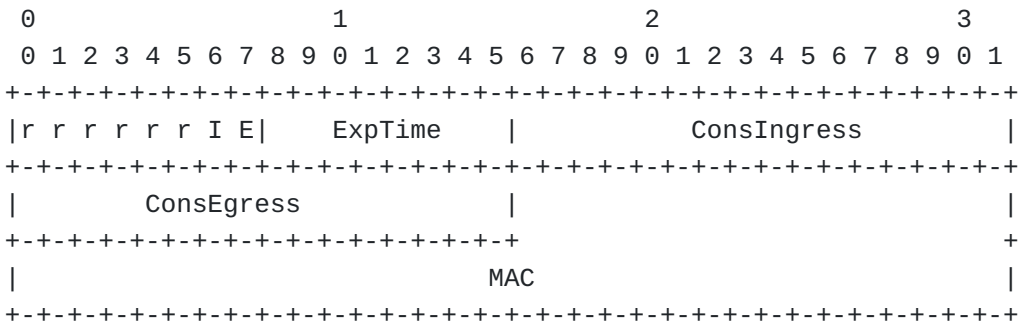


Figure 9: SCION path type - Format of the Hop Field

*r: The r bits are unused and reserved for future use.

*I: The ConsIngress Router Alert flag. If the ConsIngress Router Alert flag has value "1", the ingress router (in construction direction) will process the L4 payload in the packet. The construction direction is the direction of beaconing.

*E: The ConsEgress Router Alert flag. If the ConsEgress Router Alert flag has value "1", the egress router (in construction direction) will process the L4 payload in the packet.

Note: A sender cannot rely on multiple routers retrieving and processing the payload even if it sets multiple router alert flags. This is use-case dependent: In the case of Traceroute informational messages, for example, the router for which the traceroute request is intended will process the request (if the corresponding Router Alert flag is set to "1") and reply to it without further forwarding the request along the path. Use cases that require multiple routers/hops on the path to process a packet have to instead rely on a hop-by-hop extension (see Section 2.3). For general information on router alerts, see [RFC2711].

*ExpTime: Expiry time of a hop field. The field is 1-byte long, thus there are 256 different values available to express an expiration time. The expiration time specified in this field is relative. An absolute expiration time in seconds is computed in combination with the Timestamp field (from the corresponding info field), as follows:

  -Timestamp + (1 + ExpTime) * (24 hours/256)

*ConsIngress, ConsEgress: The 16-bits ingress/egress interface IDs in construction direction, that is, the direction of beaconing.

*MAC: The 6-byte Message Authentication Code to authenticate the hop field. For details on how this MAC is calculated, see Section 4.1.1.

2.2.3.3. One-Hop Path Type

The one-hop path type OneHopPath is currently used to bootstrap beaconing between neighboring ASes. This is necessary, as neighbor ASes do not have a forwarding path yet before beaconing is started.

A one-hop path has exactly one info field and two hop fields with the specialty that the second hop field is not known a priori, but is instead created by the ingress SCION border router of the neighboring AS while processing the one-hop path. Any entity with access to the forwarding key of the source endpoint AS can create a valid info and hop field as described in Section 2.2.3.2.3 and Section 2.2.3.2.4, respectively.

Upon receiving a packet containing a one-hop path, the ingress border router of the destination AS fills in the ConsIngress field in the second hop field of the one-hop path with the ingress interface ID. It sets the ConsEgress field to "0", indicating that the path cannot be used beyond the destination AS. Then it calculates and appends the appropriate MAC for the hop field.

Figure 10 below shows the layout of a SCION one-hop path type. There is only a single info field; the appropriate hop field can be

processed by a border router based on the source and destination address. In this context, the following rules apply:

- *At the source endpoint AS, *CurrHF := 0*.
- *At the destination endpoint AS, *CurrHF := 1*.

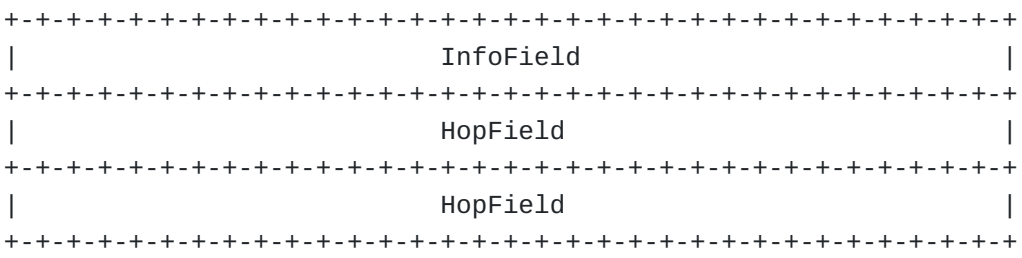


Figure 10: Layout of the SCION one-hop path type

2.2.3.4. Path Reversal

When a destination endpoint receives a SCION packet, it can use the path information in the SCION header for sending the reply packets. For this, the destination endpoint **MUST** perform the following steps:

1. Reverse the order of the info fields;
2. Reverse the order of the hop fields;
3. For each info field, negate the construction direction flag C; do not change the accumulator field Acc.
4. In the PathMetaHdr field:
   - *Set the CurrINF and CurrHF to "0".
   - *Reverse the order of the non-zero SegLen fields.

   Note: For more information on the PathMetaHdr field, see Section 2.2.3.2.1.

A similar mechanism is possible for on-path routers, for example to send SCION Control Message Protocol (SCMP) messages to the sender of the original packet.

2.3. Extension Headers

This section specifies the SCION extension headers. SCION currently provides two types of extension headers: the Hop-by-Hop (HBH) Options header and the End-to-End (E2E) Options header.

- The Hop-by-Hop Options header is used to carry optional information that may be examined and processed by every SCION router along a packet's delivery path. The Hop-by-Hop Options header is identified by value "200" in the NextHdr field of the SCION common header (see Section 2.2.1).

- The End-to-End Options header is used to carry optional information that may be examined and processed by the sender and/or the receiver of the packet. The End-to-End Options header is identified by value "201" in the NextHdr field of the SCION common header (see Section 2.2.1).

If both headers are present, the HBH Options header **MUST** come before the E2E Options header.

Note: The SCION extension headers are defined and used based on and similar to the IPv6 extensions as specified in section 4 of [RFC8200]. The SCION Hop-by-Hop Options header and End-to-End Options header resemble the IPv6 Hop-by-Hop Options Header (section 4.3 in the RFC) and Destination Options Header (section 4.6), respectively.

2.3.1. Format of the SCION Options Headers

The SCION Hop-by-Hop Options and End-to-End Options headers have the following format:

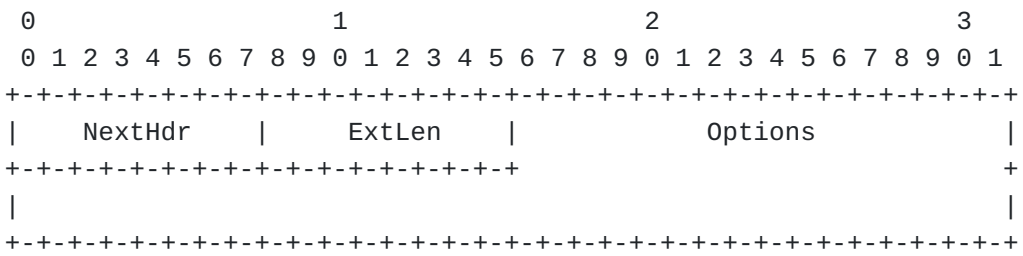


Figure 11: Extension headers: Options header

- NextHdr: Unsigned 8-bit integer. Identifies the type of header immediately following the Hop-by-Hop/End-to-End Options header. Values of this field respect the Assigned SCION Protocol Numbers (see also Appendix "Assigned SCION Protocol Numbers").

*ExtLen: Unsigned 8-bit integer. The value of this field is computed as the length of the complete Hop-by-Hop/End-to-End Options header in multiples of 4-bytes minus 1.

*Options: This is a variable-length field. The length of this field **MUST** be such that the complete length of the Hop-by-Hop/End-to-End Options header is an integer multiple of 4 bytes. The Options field contains one or more Type-Length-Value (TLV) encoded options. For details, see Section 2.3.1.1.

The Hop-by-Hop/End-to-End Options header is aligned to 4 bytes.

2.3.1.1. Options Field

The Options field of the Hop-by-Hop Options and the End-to-End Options headers carries a variable number of options that are type-length-value (TLV) encoded. Each TLV-encoded option has the following format:

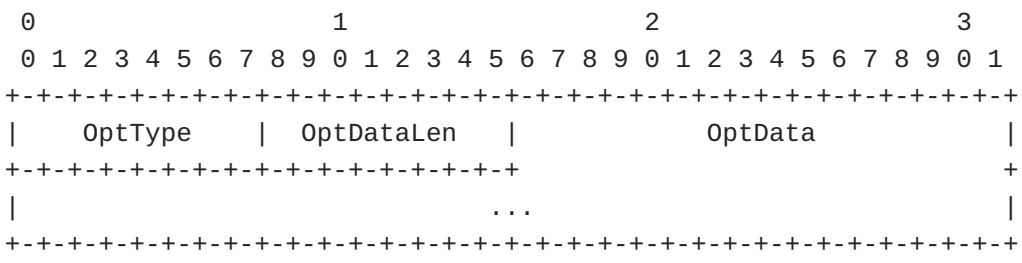


Figure 12: Options field: TLV-encoded options

*OptType: 8-bit identifier of the type of option. The following option types are assigned to the SCION HBH/E2E Options header:

Decimal	Option Type
0	Pad1 (see Section 2.3.1.1.1)
1	PadN (see Section 2.3.1.1.2)
2	SCION Packet Authenticator Option. Only used by the End-to-End Options header (experimental).
253	Used for experimentation and testing
254	Used for experimentation and testing
255	Reserved

Table 4: Option types of SCION Options header

*OptDataLen: Unsigned 8-bit integer denoting the length of the OptData field of this option in bytes.

*OptData: Variable-length field. Option-type specific data.

The options within a header **MUST** be processed strictly in the order they appear in the header. This is to prevent a receiver from, for example, scanning through the header looking for a specific option and processing this option prior to all preceding ones.

Individual options may have specific alignment requirements, to ensure that multibyte values within the OptData fields have natural boundaries. The alignment requirement of an option is specified using the notation "xn+y". This means that the OptType **MUST** appear at an integer multiple of x bytes from the start of the header, plus y bytes. For example:

- *2n: means any 2-bytes offset from the start of the header.
- *4n+2: means any 4-bytes offset from the start of the header, plus 2 bytes.

There are two padding options to align subsequent options and to pad out the containing header to a multiple of 4 bytes in length - for details, see below. All SCION implementations **MUST** recognize these padding options.

2.3.1.1.1. Pad1 Option

Alignment requirement: none.

Figure 13: TLV-encoded options - Pad1 option

Note: The format of the Pad1 option is a special case - it does not have length and value fields.

The Pad1 option is used to insert 1 byte of padding into the Options field of an extension header. If more than one byte of padding is required, the PadN option **SHOULD** be used, rather than multiple Pad1 options. See the next section for more information on the PadN option.

2.3.1.1.2. PadN Option

Alignment requirement: none.

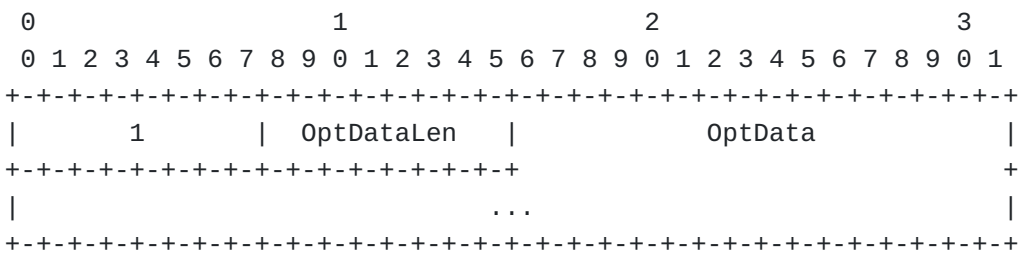


Figure 14: TLV-encoded options - PadN option

The PadN option is used to insert two or more bytes of padding into the Options field of an extension header. For N bytes of padding, the OptDataLen field contains the value N-2, and the OptData consists of N-2 zero-valued bytes.

2.4. Upper-Layer Protocol Issues

2.4.1. Pseudo Header for Upper-Layer Checksum

Any transport or other upper-layer protocol that includes addresses from the SCION header in the checksum computation **MUST** use the following pseudo header:

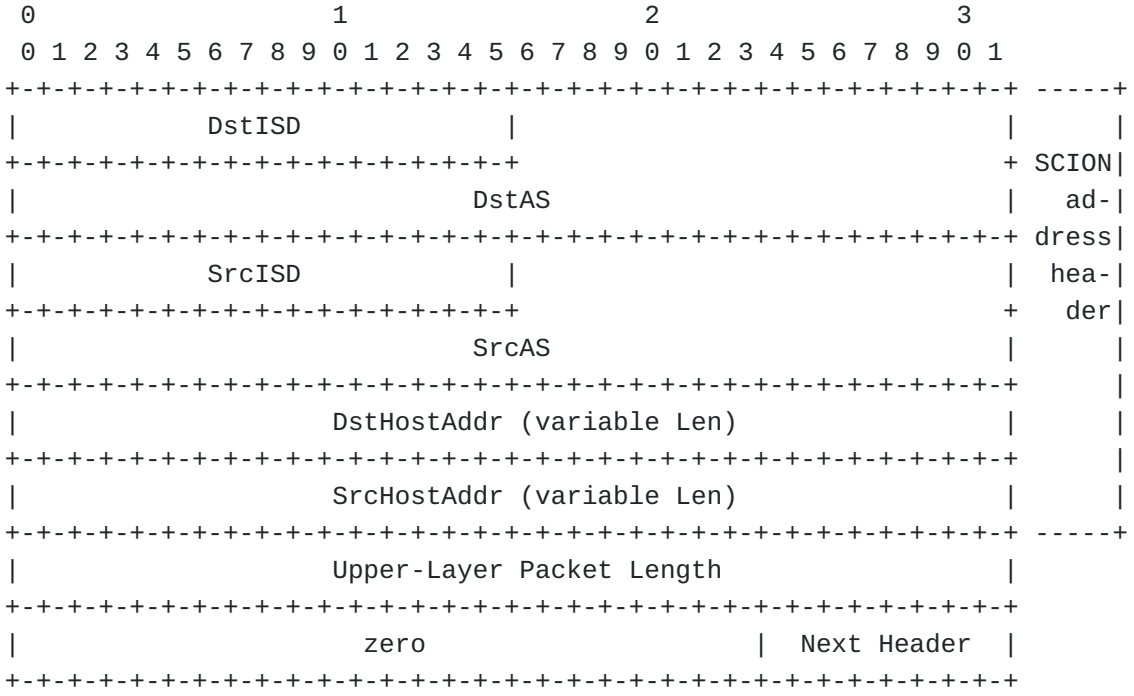


Figure 15: Layout of the pseudo header for the upper-layer checksum

*DstISD, SrcISD, DstAS, SrcAS, DstHostAddr, SrcHostAddr: These values are taken from the SCION address header.

*Upper-Layer Packet Length: The length of the upper-layer header and data. Some upper-layer protocols define headers that carry the length information explicitly (e.g., UDP). This information is used as the upper-layer packet length in the pseudo header for these protocols. The remaining protocols, which do not carry the length information directly (e.g., the SCION Control Message Protocol SCMP), use the value from the PayloadLen field in the SCION common header, minus the sum of the extension header lengths.

*Next Header: The protocol identifier associated with the upper-layer protocol (e.g., 17 for UDP - see also Appendix "Assigned SCION Protocol Numbers"). This field can differ from the NextHdr field in the SCION common header, if extensions are present.

3. Life of a SCION Data Packet

This section gives a high-level description of the life cycle of a SCION packet: How it is crafted at its source endpoint, passes through a number of routers, and finally reaches its destination endpoint. It is assumed that both source and destination are native SCION endpoints (i.e., they both run a native SCION network stack).

To keep it brief, the example illustrates an intra-ISD case, i.e., all communication happens within a single ISD. As the sample ISD only consists of one core AS, the end-to-end path only includes an up-path and down-path segment. In the case of inter-ISD forwarding, the complete end-to-end path from source endpoint to destination endpoint would always require a core-path segment, too. But this makes no difference for the forwarding process, which works the same in an intra- and inter-ISD context. We therefore abstain from describing the inter-ISD forwarding.

3.1. Description

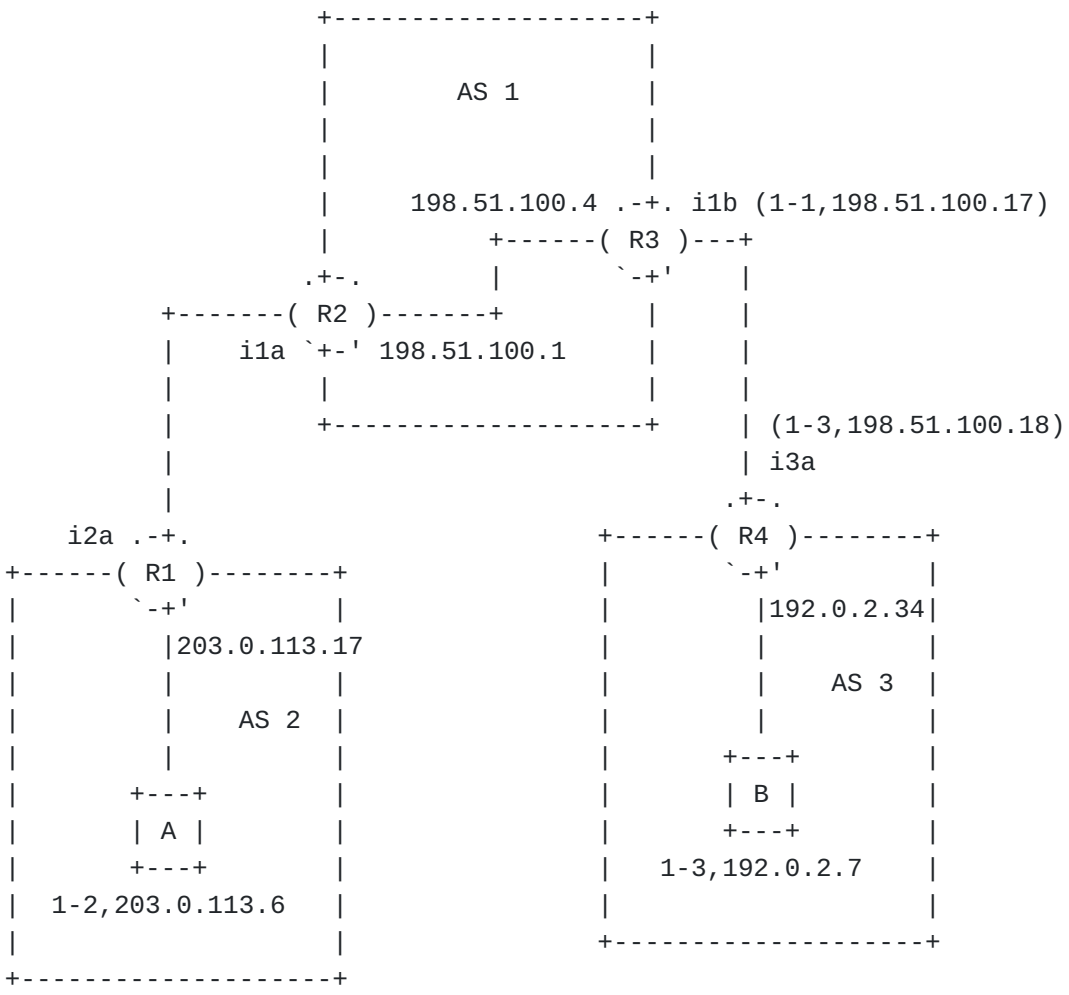


Figure 16: Sample topology to illustrate the life cycle of a SCION packet. AS 1 is the core AS of ISD 1, and AS 2 and AS 3 are non-core ASes of ISD 1.

Based on the network topology in Figure 16 above, this example shows the path of a SCION packet sent from source endpoint A to destination endpoint B, and how it will be processed by each router on the path. This is done by means of simplified snapshots of the packet header after each such processing step. These snapshots, which are depicted in tables, show the most relevant information of the header, i.e., the SCION path and IP encapsulation for local communication.

3.2. Creating an End-to-End SCION Forwarding Path

In this example, source endpoint A in AS 2 wants to send a data packet to destination endpoint B in AS 3. Both AS 2 and AS 3 are part of ISD 1. To create an end-to-end SCION forwarding path, source endpoint A first requests its own AS-2 control service for up-segments to the core AS in its ISD. The AS-2 control service will return up-segments from AS 2 to the ISD core. Endpoint A will also

query its AS-2 control service for a down-segment from its ISD core AS to AS 3, in which destination endpoint B is located. The AS-2 control service (possibly after connecting to the core control service) will return down-segments from the ISD core down to AS 3.

Note: For more details on the lookup of path segments, see the section "Path Lookup" in the Control Plane specification ([I-D.scion-cp]).

Based on its own selection criteria, source endpoint A selects the up-segment (0,i2a)(i1a,0) and the down-segment (0,i1b)(i3a,0) from the path segments returned by its own AS-2 control service. The path segments consist of hop fields that carry the ingress and egress interfaces of each AS (e.g., i2a, i1a, ...), as described in detail in Section 2.1 - (x,y) represents one hop field.

To obtain an end-to-end forwarding path from the source AS to the destination AS, source endpoint A combines the two path segments into the resulting SCION forwarding path, which contains the two info fields *IF1* and *IF2* and the hop fields (0,i2a), (i1a,0), (0,i1b), and (i3a,0).

Note: As this brief sample path does not contain a core-segment, the end-to-end path only consists of two path segments.

Source endpoint A now adds this end-to-end forwarding path to the header of the packet that A wants to send to destination endpoint B, and starts transferring the packet. The following section describes what happens with the SCION packet header on the packet's way from A to B.

3.3. Step-by-Step Explanation

This section explains the packet header modifications at each router, based on the network topology in Figure 16 above. Each step includes a table that represents a simplified snapshot of the packet header at the end of this specific step. Regarding the notation used in the figure/tables, each SRC and DST entry should be read as router (or endpoint) followed by its address. The current info field (with metadata on the current path segment) in the SCION header is depicted italic/cursive in the tables. The current hop field, representing the current AS, is shown bold. The snapshot tables also include references to IP/UDP addresses.

Note: In this context, a border router is called **ingress** border router when it refers to an entrance border router to an AS, as seen from the direction of travel of the SCION packet. So in the context here, the ingress border router is the *(packet) incoming* border router. A border router is called **egress** border router when it refers to an exit border router of an AS, as seen from the direction

of travel of the SCION packet. So in this context, the egress border router is the *(packet) leaving* border router.

*Step 1

A->R1: The SCION-enabled source endpoint A in AS 2 creates a new SCION packet destined for destination endpoint B in AS 3, with payload P. Endpoint A sends the packet (for the chosen forwarding path) to the next SCION router as provided by its control service, which is in this case R1. A encapsulates the SCION packet into an underlay UDP/IPv4 header for the local delivery to R1, utilizing AS 2's internal routing protocol. The current info field is *IF1*. Upon receiving the packet, R1 will forward the packet on the egress interface that endpoint A has included into the first hop field of the SCION header.

A -> R1	
SCION	SRC = 1-2,203.0.113.6 (source endpoint A)
	DST = 1-3,192.0.2.7 (dest. endpoint B)
	PATH =
	- *IF1* **(0,i2a)** (i1a,0)
	- IF2 (0,i1b) (i3a,0)
UDP	P_S = 30041, P_D = 30041
IP	SRC = 203.0.113.6 (endpoint A)
	DST = 203.0.113.17 (router R1)
Link layer	SRC=A, DST=R1

Table 5: Snapshot header - step 1

*Step 2

R1->R2: Router R1 inspects the SCION header and considers the relevant info field of the specified SCION path, which is the info field indicated by the current info field pointer. In this case, it is the first info field *IF1*. The current hop field is the first hop field (0,i2a), which instructs router R1 to forward the packet on its interface i2a. After reading the current hop field, R1 moves the pointer forward by one position to the second hop field (i1a,0). Note that, at this point, no underlay IP header is necessary, since the routers R1 and R2 are directly connected over layer 2.

Note: Although technically there is no need for a UDP/IP underlay if two routers are directly connected, the SCION implementation always uses a UDP/IP underlay in practice. This is to enable a common interface for all routers.

R1 -> R2	
SCION	SRC = 1-2,203.0.113.6 (source endpoint A)
	DST = 1-3,192.0.2.7 (dest. endpoint B)

R1 -> R2	
	PATH =
	- *IF1* (0,i2a) **(i1a,0)**
	- IF2 (0,i1b) (i3a,0)
Link layer	SRC=R1, DST=R2

Table 6: Snapshot header - step 2

**Step 3*

R2->R3: When receiving the packet, router R2 of core AS 1 checks whether the packet has been received through the ingress interface i1a as specified by the current hop field. Otherwise, the packet is dropped by router R2. The router notices that it has consumed the last hop field of the current path segment, and hence moves the pointer from the current info field to the next info field *IF2*. The corresponding current hop field is (0,i1b), which contains egress interface i1b. R2 maps the i1b interface ID to egress router R3, it therefore encapsulates the SCION packet inside an intra-AS underlay IP packet with the address of R3 as the underlay destination.

R2 -> R3	
SCION	SRC = 1-2,203.0.113.6 (source endpoint A)
	DST = 1-3,192.0.2.7 (dest. endpoint B)
	PATH =
	- IF1 (0,i2a) (i1a,0)
	- *IF2* **(0,i1b)** (i3a,0)
UDP	P_S = 30041, P_D = 30041
IP	SRC = 198.51.100.1 (router R2)
	DST = 198.51.100.4 (router R3)
Link layer	SRC=R2, DST=R3

Table 7: Snapshot header - step 3

**Step 4*

R3->R4: Router R3 inspects the current hop field in the SCION header, uses interface i1b to forward the packet to its neighbor SCION router R4 of AS 3, and moves the current hop-field pointer forward. It adds an IP header to reach R4.

R3 -> R4	
SCION	SRC = 1-2,203.0.113.6 (source endpoint A)
	DST = 1-3,192.0.2.7 (dest. endpoint B)
	PATH =
	- IF1 (0,i2a) (i1a,0)
	- *IF2* (0,i1b) **(i3a,0)**
UDP	P_S = 30041, P_D = 30041
IP	SRC = 1-1,198.51.100.17 (router R3)

R3 -> R4	
	DST = 1-3,198.51.100.18 (router R4)
Link layer	SRC=R3, DST=R4

Table 8: Snapshot header - step 4

*Step 5

R4->B: SCION router R4 first checks whether the packet has been received through the ingress interface i3a as specified by the current hop field. R4 will then also realize, based on the fields CurrHF and SegLen in the SCION header, that the packet has reached the last hop in its SCION path. Therefore, instead of stepping up the pointers to the next info or hop field, router R4 inspects the SCION destination address and extracts the endpoint address 192.0.2.7. It creates a fresh underlay UDP/IP header with this address as destination and with itself as source. The intra-domain forwarding can now deliver the packet to destination endpoint B.

R4 -> B	
SCION	SRC = 1-2,203.0.113.6 (source endpoint A)
	DST = 1-3,192.0.2.7 (dest. endpoint B)
	PATH =
	- IF1 (0,i2a) (i1a,0)
	- *IF2* (0,i1b) **(i3a,0)**
UDP	P_S = 30041, P_D = 30041
IP	SRC = 192.0.2.34 (router R4)
	DST = 192.0.2.7 (endpoint B)
Link layer	SRC=R4, DST=B

Table 9: Snapshot header - step 5

When destination endpoint B wants to respond to source endpoint A, it can just swap the source and destination addresses in the SCION header, reverse the SCION path, and set the pointers to the info and hop fields at the beginning of the reversed path (see also Section 2.2.3.4).

4. Path Authorization

Path authorization guarantees that data packets always traverse the network along paths segments authorized by all on-path ASes in the control plane. In contrast to the IP-based Internet, where forwarding decisions are made by routers based on locally stored information, SCION routers base their forwarding decisions purely on the forwarding information carried in the packet header and set by endpoints.

SCION uses cryptographic mechanisms to efficiently provide path authorization. The mechanisms are based on *symmetric* cryptography in

the form of Message Authentication Codes (MACs) in the data plane to secure forwarding information encoded in hop fields. This chapter first explains how hop field MACs are computed, then how they are validated as they traverse the network.

4.1. Authorizing Segments through Chained MACs

When authorizing SCION PCBs and path segments in the control plane and forwarding information in the data plane, an AS authenticates not only its own hop information but also an aggregation of all upstream hops. This section describes how this works.

4.1.1. Hop Field MAC Computation

The MAC in the hop fields of a SCION path has two purposes:

- Preventing malicious endpoints from illegally adding, removing, or reordering hops within a path segment created during beaconing in the control plane.

- Authentication of the information contained in the hop field itself, in particular the ExpTime, ConsIngress, and ConsEgress.

To fulfill the above purposes, the MAC for the hop field of AS_i includes both the components of the current hop field HF_i and an aggregation of the path segment identifier and all preceding hop fields/entries in the path segment. The aggregation is a 16-bit XOR-sum of the path segment identifier and the hop field MACs.

When originating a path-segment construction beacon PCB in the **control plane**, a core AS chooses a random 16-bit value as segment identifier SegID for the path segment and includes it in the PCB's Segment Info component. In the control plane, each AS_i on the path segment computes the MAC for the current hop HF_i, based on the value of SegID and the MACs of the preceding hop entries. Here, the full XOR-sum is computed explicitly.

For high-speed packet processing in the **data plane**, computing even cheap operations such as the XOR-sum over a variable number of inputs is complicated, in particular for hardware router implementations. To avoid this overhead for the MAC-chaining in path authorization in the data plane, the XOR-sum is tracked incrementally for each (of the up to three) path segments in a path, as a separate, updatable accumulator field Acc. The routers update the accumulator field Acc by adding/subtracting only a single 16-bit value each.

When combining path segments to create a path to the destination endpoint, the source endpoint **MUST** also initialize the value of accumulator field Acc for each path segment. The Acc field **MUST**

contain the correct XOR-sum of the path segment identifier and preceding hop field MACs expected by the first router that is traversed.

In the following, the computation of the hop field MAC as well as the accumulator field Acc is explained.

Note: The algorithm used by SCION to compute the hop field MAC is based on the AES-CMAC algorithm, truncated to 48-bits - see also [RFC4493]. In principle, the computation of the MAC is an AS-specific choice; only the control service and routers of the AS need to agree on keys, algorithm, and input for the MAC. However, note that we do not provide nor specify any mechanism to coordinate AS-specific choices between the routers and the control services of the AS.

4.1.1.1. MAC - Definition

- Consider a path segment with "n" hops, containing ASes AS_0, ... , AS_{n-1}, with forwarding keys K_0, ... , K_{n-1} in this order.

- AS_0 is the core AS that created the PCB representing the path segment and that added a random initial 16-bit segment identifier SegID to the Segment Info field of the PCB.

The MAC_i of the hop field of AS_i is now calculated based on the following definition:

MAC_i =
Ck_i (SegID XOR MAC_0 [:2] ... XOR MAC_{i-1} [:2], Timestamp, $ExpTime_i$, $ConsIngress_i$, $ConsEgress_i$)

where

- k_i = The forwarding key k of the current AS_i

- Ck_i (...) = Checksum C over (...) computed with forwarding key k_i

- SegID = The random initial 16-bit segment identifier set by the core AS when creating the corresponding PCB

- XOR = The bitwise "exclusive or" operation

- MAC_i [:2] = The hop field MAC for AS_i, truncated to 2 bytes

- Timestamp = The timestamp set by the core AS when creating the corresponding PCB

- $ExpTime_i$, $ConsIngress_i$, $ConsEgress_i$ = The content of the hop field HF_i

The above definition implies that the current MAC is based on the XOR-sum of the truncated MACs of all preceding hop fields in the path segment as well as the path segment's SegID. In other words, the current MAC is *chained* to all preceding MACs.

4.1.1.2. Layout of the Input Data for the MAC Calculation

Figure 17 below shows the layout of the input data to calculate the hop field MAC in the data plane. The input layout represents the 8 bytes of the info field and the first 8 bytes of the hop field, with some fields zeroed out.

Note that the MAC field itself is only 6 bytes long - see also Section 2.2.3.2.4.

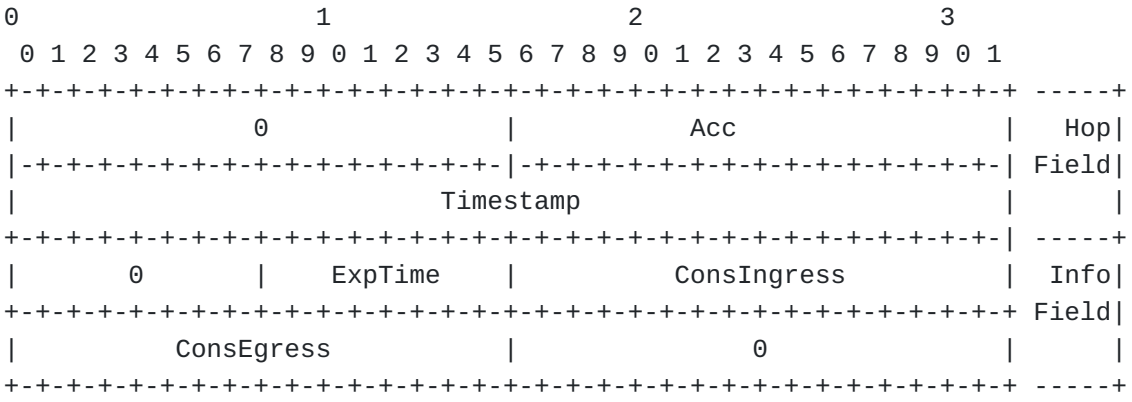


Figure 17: Input data to calculate the hop field MAC

4.1.1.3. Accumulator Acc - Definition

The accumulator Acc_i is an updatable counter introduced in the data plane to reduce overhead caused by MAC-chaining for path authorization. This is achieved by incrementally tracking the XOR-sum of the previous MACs as a separate, updatable accumulator field Acc, which is part of the path segment's info field InfoField in the packet header (see also Section 2.2.3.2.3). Routers update this field by adding/subtracting only a single 16-bit value each.

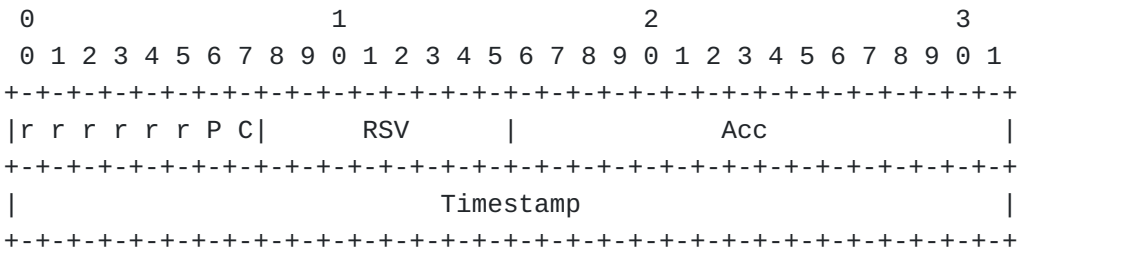


Figure 18: The Info Field of a specific path segment in the packet header, with the updatable accumulator field `Acc`.

This is how it works:

Section 4.1.1.1 defines MAC_i as follows:

MAC_i =
Ck_i (SegID XOR MAC_0 [:2] ... XOR MAC_{i-1} [:2], Timestamp, $ExpTime_i$, $ConsIngress_i$, $ConsEgress_i$)

In the data plane, the expression SegID XOR MAC_0 [:2] ... XOR MAC_i-1 [:2] is replaced by Acc_i. This results in the following alternative procedure for the computation of MAC_i used in the data plane:

$MAC_i = Ck_i$ (Acc_i, Timestamp, $ExpTime_i$, $ConsIngress_i$, $ConsEgress_i$)

During forwarding in the data plane, each AS_i updates the Acc field in the packet header, such, that it contains the correct input value of the Accumulator Acc for the next AS in the path segment to be able to calculate the MAC over its hop field. Note that the correct input value of the Acc field depends on the direction of travel.

The value of the accumulator Acc_{i+1} is calculated based on the following definition (in the direction of beaconing):

$Acc_{i+1} = Acc_i$ XOR MAC_i [:2]

*XOR = The bitwise "exclusive or" operation

*MAC_i [:2] = The hop field MAC for the current AS_i, truncated to 2 bytes

4.1.2. Peering Links

The above described computation of a hop field MAC does not apply to a peering hop field, i.e., to a hop field that allows transiting from a child interface/link to a peering interface/link.

The reason for this is that the MACs of the hop fields "after" the peering hop field (in beaconing direction) are not chained to the MAC of the peering hop field, but to the MAC of the main hop field in the corresponding AS entry. To make this work, the MAC of the peering hop field is also chained to the MAC of the main hop field - this allows to validate the chained MAC for both the peering hop field and the following hop fields, by using the same Acc field value.

This results in the following definition for the calculation of the MAC for a peering hop field.

The Control Plane Internet-Draft defines a peering hop field as follows:

$\text{hop field}^{\text{Peer}}_i = (\text{ExpTime}^{\text{Peer}}_i, \text{ConsIngress}^{\text{Peer}}_i, \text{ConsEgress}^{\text{Peer}}_i, \text{MAC}^{\text{Peer}}_i)$

This definition, the general definition of a hop field MAC and the explanation above leads to the following definition of the MAC for a peering hop $\text{field}^{\text{Peer}}_i$:

$\text{MAC}^{\text{Peer}}_i =$
$\text{Ck}^{\text{Peer}}_i (\text{SegID XOR MAC}_0 [:2] \text{ ... XOR MAC}_i [:2], \text{Timestamp}, \text{ExpTime}^{\text{Peer}}_i, \text{ConsIngress}^{\text{Peer}}_i, \text{ConsEgress}^{\text{Peer}}_i)$

Note: The XOR-sum of the MACs in the formula of the peering hop field **also includes** the MAC of the main hop field (whereas for the calculation of the MAC for the main hop field itself only the XOR-sum of the *previous* MACs is used).

Note: The Control-Plane Internet-Draft is available here: [I-D.scion-cp].

4.2. Path Initialization and Packet Processing

As is described in Section 2.1, the path header of the data-plane packets only contains a sequence of info fields and hop fields without any additional data from the corresponding PCBs. Also, the SCION path does not contain any AS numbers (except for the source and destination ASes), and there is no field explicitly defining the type of each segment (up, core, or down). This chapter describes the required steps for the source endpoint and each SCION router to ensure that a data packet only traverses authorized segments. The chapter first specifies the initialization of a path at the source endpoint, followed by the steps that the SCION routers needs to perform when a data-plane packet traverses an AS on its way to the destination.

4.2.1. Initialization at Source Endpoint

The source endpoint needs to initialize a path correctly for the SCION routers to be able to verify the hop fields in the data plane. To this end, the source endpoint **MUST** perform the following steps:

1. Combine the preferred end-to-end path from the path segments obtained during path lookup.

2. Extract the info fields and hop fields from the different path segments that together build the end-to-end path to the destination endpoint. Then insert the relevant information from the path segments' info and hop fields into the corresponding InfoFields and Hopfields, respectively, in the data packet header.

3. Each 8-byte info field InfoField in the packet header contains the updatable Acc field as well as a Peering flag P and a Construction Direction flag C (see also Section 2.2.3.2.3). As a next step in the path initialization process, the source **MUST** correctly set the flags and the Acc field of all InfoFields included in the path, according to the following rules:

   Note: As already stated above, the type of segment is not visible directly in the forwarding path but can be inferred from flags and other information. See also Section 4.2.2.

   * The Construction Direction flag C **MUST** be set to "1" whenever the corresponding segment is traversed in construction direction, i.e., for down-path segments and potentially for core-segments. It **MUST** be set to "0" for up-path segments and "reversed" core-segments.

   * The Peering flag P **MUST** be set to "1" for up- and down-segments if, and only if, the path contains a peering hop field.

   * The field Acc field is an updatable field. It is used to compute the MAC over the current hop field. The value of the field Acc field corresponds to the value of the Accumulator Acc_i for AS_i. It is initialized based on the location of the sender in relation to path construction.

   The following InfoField settings are possible, based on the following possible use cases:

   * **Use case 1**
     The path segment is traversed in construction direction and includes no peering hop field. It starts at the *i*-th AS of the full segment discovered in beaconing. In this case:

     - The Peering flag P = "0"

     - The Construction Direction flag C = "1"

     - The value of the Acc = Acc_i. For more details, see Section 4.1.1.3.

* **Use case 2**
  The path segment is traversed in construction direction and includes a peering hop field (which is the first hop field of the segment). It starts at the *i*-th AS of the full segment discovered in beaconing. In this case:

  - The Peering flag P = "1"

  - The Construction Direction flag C = "1"

  - The value of the Acc = Acc_{i+1}. For more details, see Section 4.1.1.3.

* **Use case 3**
  The path segment is traversed against construction direction. The full segment has "n" hops. In this case:

  - The Peering flag P = "0" or "1" (depending on whether the last hop field in the up-segment is a peering hop field)

  - The Construction Direction flag C = "0"

  - The value of the Acc = Acc_{n-1}. This is because seen from the direction of beaconing, the source endpoint is the last AS in the path segment. For more details, see Section 4.1.1.1 and Section 4.1.1.3.

4. Besides setting the flags and the Acc field, the source endpoint **MUST** also set the pointers in the CurrInf and CurrHF fields of the Path Meta Header PathMetaHdr (see Section 2.2.3.2.1). As the source endpoint builds the starting point of the forwarding, both pointers **MUST** be set to "0".

4.2.2. Processing at Routers

During forwarding, each AS_i verifies the path contained in the packet header with the help of the current value of the MAC in the current hop field, and the current value of the Accumulator in the Acc field of the current info field. Additionally, each AS has to correctly set the value of the Accumulator in the Acc field for the next AS to be able to verify its hop field. The exact operations differ based on the location of the AS on the path.

The processing of SCION packets for ASes where a peering link is crossed between path segments is special cased. A path containing a peering link contains exactly two path segments, one against construction direction (up) and one in construction direction (down). On the path segment against construction direction (up), the peering hop field is the last hop of the segment. In construction

direction (down), the peering hop field is the first hop of the segment.

The following sections describe the tasks to be performed by the ingress and egress border router of each on-path AS. Each operation is described from the perspective of AS_i, where i belongs to [0 ... n-1], and n == the number of ASes in the path segment (counted from the first AS in the beaconing direction).

Note: In this context, a border router is called **ingress** border router when it refers to an entrance border router to an AS, as seen from the direction of travel of the SCION packet. So in the context here, the ingress border router is the *(packet) incoming* border router. A border router is called **egress** border router when it refers to an exit border router of an AS, as seen from the direction of travel of the SCION packet. So in this context, the egress border router is the *(packet) leaving* border router.

The following figure provides a simplified representation of the processing at routers both in construction direction and against construction direction.

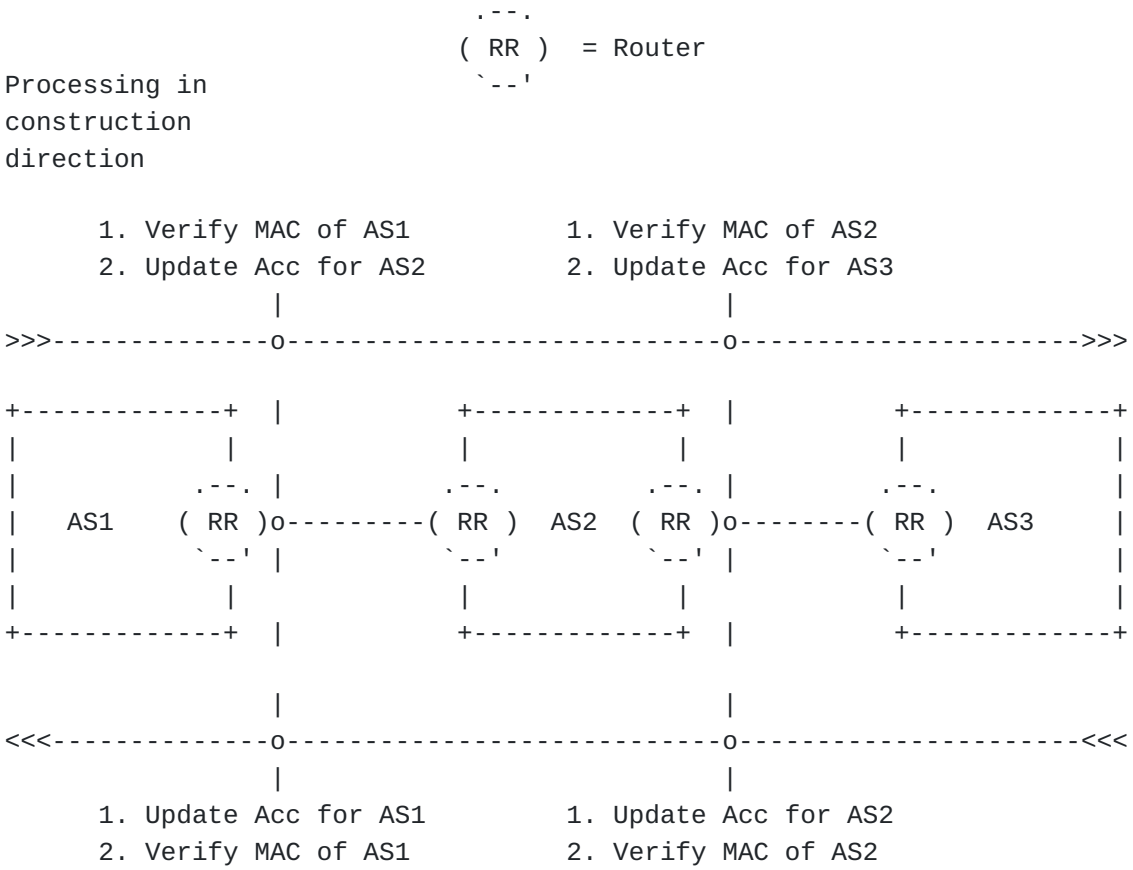


Figure 19: A simplified representation of the processing at routers in both directions.

4.2.2.1. Steps Ingress Border Router

This section describes the steps that a SCION ingress border router **MUST** perform when it receives a SCION packet.

1. Check that the interface through which the packet was received is equal to the ingress interface in the current hop field.

2. Check that the current hop field is not expired and within its validity period.

3. The next steps depend on the direction of travel and whether this segment includes a peering hop field. Both features are indicated by the settings of the Construction Direction flag C and the Peering flag P in the current info field. Therefore,

check the settings of both flags. The following combinations are possible:

* The packet traverses the path segment in **construction direction** (C = "1" and P = "0" or "1"). In this case, proceed with step 4.

* The packet traverses the path segment **against construction direction** (C = "0"). The following use cases are possible:

  - **Use case 1**
    The path segment includes **no peering hop field** (P = "0"). In this case, the ingress border router **MUST** take the following step(s):

    o Compute the value of the Accumulator Acc as follows:

      $Acc = Acc_{i+1}$ XOR MAC_i
      where
      Acc_{i+1} = the current value of the field Acc in the current info field
      MAC_i = the value of MAC_i in the current hop field representing AS_i

      Note: In the case described here the packet travels against direction of beaconing. That is, the packet comes from AS_{i+1} and is going to enter AS_i. This means that the Acc field of this incoming packet represents the value of Accumulator Acc_{i+1}. However, to compute the MAC_i for the current AS_i, we need the value of Accumulator Acc_i (see Section 4.1.1.3). Because the border router knows that the formula for Acc_{i+1} = Acc_i XOR MAC_i [:2] (see also Section 4.1.1.3), and because the values of Acc_{i+1} and MAC_i are known, the router will be able to recover the value Acc_i based on the just-mentioned formula for Acc.

    o Replace the current value of the field Acc in the current info field with the newly calculated value of Acc.

    o Compute the MAC^{Verify}_i over the hop field of the current AS_i. For this, use the formula in Section 4.1.1.1, but replace SegID XOR MAC_0[:2] ... XOR MAC_i-1 [:2] in the formula with the value of the accumulator Acc as just set in the Acc field in the current info field.

    o Check that the MAC_i in the current hop field matches the just-calculated MAC^{Verify}_i. If yes, it is fine.

Otherwise, drop the packet, and reply with a "parameter problem" type of SCMP message.

- Check whether the current hop field is the last hop field in the path segment. For this, look at the value of the current SegLen and other metadata in the path meta header. If yes, increment both CurrInf and CurrHF in the path meta header. Proceed with step 4.

- **Use case 2**
  The path segment includes a **peering hop field** (P = "1"), but the current hop is **not** the peering hop, that is, the current hop field is **not** the *last* hop field of the segment, seen from the direction of travel - this can be determined by looking at the value of the current SegLen and other metadata in the path meta header. In this case, the ingress border router needs to perform the steps previously described for the path segment without peering hop field. However, the border router **MUST NOT** increment CurrInf and **MUST NOT** increment CurrHF in the path meta header. Proceed with step 4.

- **Use case 3**
  The path segment includes a **peering hop field** (P = "1"), and the current hop field *is* the peering hop field. This would be the case if the current hop field is the *last* hop field of the segment, seen from the direction of travel - to find out whether this is true, check the value of the current SegLen and other metadata in the path meta header. In this case, the ingress border router **MUST** take the following step(s):

  - Compute MAC^{Peer}_{i}. For this, use the formula in Section 4.1.2, but replace SegID XOR MAC_0[:2] ... XOR MAC_i [:2] in the formula with the value of the accumulator Acc as set in the Acc field in the current info field (this is the value of the accumulator Acc as it comes with the packet).

  - Check that the MAC_i in the current hop field matches the just-calculated MAC^{Peer}_{i}. If yes, it is fine. Otherwise, drop the packet, and reply with a "parameter problem" type of SCMP message.

  - Increment both CurrInf and CurrHF in the path meta header. Proceed with step 4.

4. Forward the packet to the egress border router (based on the egress interface ID in the current hop field) or to the destination endpoint, if this is the destination AS.

Note: For more information on the path meta header, see Section 2.2.3.2.1.

4.2.2.2. Steps Egress Border Router

This section describes the steps that a SCION egress border router **MUST** perform when it receives a SCION packet.

1. Parse the SCION packet.

2. The next steps depend on the direction of travel and whether this segment includes a peering link. Both features are indicated by the settings of the Construction Direction flag C and the Peering flag P in the currently valid info field. Therefore, first check the settings of both flags. The following use cases are possible:

   * **Use case 1**
     The packet traverses the path segment in **construction direction** (C = "1"). The path segment either includes **no peering hop field** (P = "0"), or the path segment does include a **peering hop field** (P = "1"), but the current hop is **not** the peering hop, that is, the current hop field is **not** the *first* hop field of the segment, seen from the direction of travel. To check whether this is true, look at the value of the current SegLen and other metadata in the path meta header. In this case, the egress border router **MUST** take the following step(s):

     - Compute MAC^{Verify}_i over the hop field of the current AS_i. For this, use the formula in Section 4.1.1.1, but replace SegID XOR MAC_0[:2] ... XOR MAC_i-1 [:2] in the formula with the value of the accumulator Acc as set in the Acc field in the current info field.

     - Check that the just-calculated MAC^{Verify}_i matches MAC_i in the hop field of the current AS_i. If yes, it is fine. Otherwise, drop the packet, and reply with a "parameter problem" type of SCMP message.

     - Compute the value of Acc_{i+1}. For this, use the formula in Section 4.1.1.3. Replace Acc_i in the formula with the current value of the accumulator Acc as set in the Acc field of the current info field.

-Replace the value of the Acc field in the current info field with the just-calculated value of Acc_{i+1}.

-Proceed with step 3.

*__Use case 2__
The packet traverses the path segment in **construction direction** (C = "1"). The path segment includes a **peering hop field** (P = "1"), and the current hop field *is* the peering hop field. This would be the case if the current hop field is the *first* hop field of the segment, seen from the direction of travel - to find out whether this is true, check the value of the current SegLen and other metadata in the path meta header. In this case, the egress border router **MUST** take the following steps:

-Compute MAC^{Peer}_{i}. For this, use the formula in Section 4.1.2, but replace SegID XOR MAC_0 [:2] ... XOR MAC_i [:2] with the value in the Acc field of the current info field.

-Check that the MAC_i in the hop field of the current AS_i matches the just-calculated MAC^{Peer}_{i}. If yes, it is fine - proceed with step 3. Otherwise, drop the packet, and reply with a "parameter problem" type of SCMP message.

*__Use case 3__
The packet traverses the path segment **against construction direction** (C = "0" and P = "0" or "1"). In this case, proceed with the next step, step 3.

3. Increment CurrHF in the path meta header.

4. Forward the packet to the neighbor AS.

Note: For more information on the path meta header, see Section 2.2.3.2.1.

5. Security Considerations

This section describes the possible security risks and attacks that SCION's data plane may be prone to, and how these attacks may be mitigated. It first discusses security risks that pertain to path authorization, followed by a section on other forwarding-related security considerations.

5.1. Path Authorization

A central property of the SCION path-aware data plane is path authorization. Path authorization guarantees that data packets

always traverse the network along path segments authorized in the control plane by all on-path ASes. This section discusses how an adversary may attempt to violate the path-authorization property, as well as SCION's prevention mechanisms to these attacks; either an attacker can attempt to create unauthorized hop fields, or they can attempt to create illegitimate paths assembled from authentic individual hop fields.

The main protection mechanism here is the hop field MAC (see Section 4.1), authenticating the hop field content, consisting of ingress/egress interface identifiers, creation and expiration timestamp and, virtually, the preceding hop field MACs in the path segment. Recall that each hop field MAC is computed using the respective AS's secret forwarding key, which is shared across the SCION routers and control plane services within each AS.

5.1.1. Forwarding key compromise

For the current default MAC algorithm, AES-CMAC truncated to 48 bits, key recovery attacks from (any number of) known plaintext/MAC combinations is computationally infeasible, as far as publicly known. In addition, the MAC algorithm can be freely chosen by each AS, enabling algorithmic agility for MAC computations. Should a MAC algorithm be discovered to be weak or insecure, each AS can quickly switch to a secure algorithm without the need for coordination with other ASes.

A more realistic risk to the secrecy of the forwarding key is exfiltration from a compromised router or control plane service. An AS can optionally rotate its forwarding key at regular intervals to limit the exposure after a temporary device compromise. However, as is perhaps self-evident, such a key rotation scheme cannot mitigate the impact of an undiscovered, permanent compromise of a device.

5.1.2. Forging hop field MAC

As a second method to break path authorization is to directly forge a hop field in an online attack, using the router as an oracle to determine the validity of the hop field MAC. The adversary needs to send one packet per guess for verification. For a 6-byte MAC, the adversary would need an expected 2^{47} (~140 trillion) tries to successfully forge the MAC of a single hop field. As the router only checks MACs during the encoded validity period of the hop field, which is limited by the packet header format to at most 24 hours, these tries need to occur in a limited time period. This results in a seemingly infeasible number of ~1.6e9 guesses per second. In the unlikely case that an online brute-force attack succeeds, the obtained hop field can be used until its inevitable expiration after the just mentioned 24 hour limit.

5.1.3. Path Splicing

In a path-splicing attack, an adversary source endpoint takes valid hop fields of multiple path segments and splices them together to obtain a new unauthorized path. However, SCION's MAC-chaining mechanism prevents from this kind of attacks. MAC validation for spliced segments would fail at SCION routers and the corresponding packets would be dropped. For details, see Section 4.1.

5.2. On-Path Attacks

When an adversary sits on the path between the source and destination endpoint, it is able to intercept the data packets that are being forwarded. This would allow the adversary to hijack traffic onto a path that is different from the intended one selected by the source endpoint. Possible on-path attacks in the data plane are modifications of the SCION path header and SCION address header. In addition, an on-path adversary can always simply drop packets. This kind of attack is fundamental and generally cannot be prevented. However, in this case, the endpoint can use SCION's path-awareness to immediately select an alternate path if available.

5.2.1. Modification of the Path Header

An on-path adversary could modify the SCION path header, and replace the remaining part of path segments to the destination with different segments. Such replaced segments must include authorized segments, as otherwise the packet would be simply dropped on its way to the destination. The already traversed portion of the current segment and past segments can also be modified by the adversary (for instance, deleting and adding valid and invalid hop fields). On reply packets from the destination, the adversary can transparently revert the changes to the path header again. For instance, if an adversary M is an intermediate AS on the path of a packet from A to B, then M can replace the packet's past path (leading up to, but not including M). The new path may not be a valid end-to-end path. However, when B reverses the path and sends a reply packet, that packet would go via M, which can then transparently change the invalid path back to the valid path to A. In addition, the endpoint address header can also be modified.

Modifications of the SCION path and address header can be discovered by the destination endpoint by a data integrity protection system. Such a data integrity protection system, loosely analogous to the IPSec Authentication Header, exists for SCION but is out of scope for this document. This is described as the SCION Packet Authentication Option (SPAO) in [CHUAT22].

Moreover, packet integrity protection is not enough if there are two colluding adversaries on the path. These colluding adversaries can forward the packet between them using a different path than selected by the source endpoint: The first on-path attacker remodels the packet header arbitrarily, and the second on-path attacker changes the path back to the original source-selected path, such that the integrity check by the destination endpoint succeeds. To prevent this attack and to defend against multiple on-path adversaries in general, proof of transit is required, which is not in scope for this document.

5.3. Off-Path Attacks

SCION's path-awareness limits the abilities of an off-path adversary to influence forwarding in the data plane. Once a packet is "in flight", it will follow its set route, no matter what an adversary may do. An adversary can attempt to disrupt the connectivity of said path by flooding a link with excessive traffic (see Section 5.4 below). After detecting congestion, the endpoint can switch to another, non-congested path for subsequent packets.

5.4. Volumetric Denial of Service Attacks

An adversary can attempt to disrupt the connectivity of a network path by flooding a link with excessive traffic. In this case, the endpoint can switch to another, non-congested path for subsequent packets.

SCION provides protection against certain reflection-based DoS attacks. Here, the adversary sends requests to a server with the source address set to the address of the victim. The server will send a reply, typically larger than the request, to the victim. As long as the attacker and the victim are located in different ASes, this can be prevented in SCION. The reply packets are simply returned along reversed path to the actual sender, regardless of the source address information. Thus, the reflected will be forwarded to the attackers AS (where it will be discarded because the destination AS does not match).

On the flip side, the path choice of the endpoint may possibly be exploited by an attacker to create intermittent congestion with relatively low send rate; the attacker can abuse the latency differences of the available paths, sending at precisely timed intervals to cause short, synchronized bursts of packets near the victim.

Note that SCION does not protect against two other types of DoS attacks, namely transport protocol attacks and application layer attacks. Such attacks are out of SCION's scope. However, the

additional information contained in the SCION header enables more targeted filtering, e.g., by ISD, AS or path length.

6. IANA Considerations

This document has no IANA actions.

The SCION AS and ISD number are SCION-specific numbers. They are currently allocated by Anapaya Systems, a provider of SCION-based networking software and solutions (see Anapaya ISD AS assignments). This task is currently being transitioned from Anapaya to the SCION Association.

7. References

7.1. Normative References

[I-D.scion-cp] de Kater, C., Rustignoli, N., and S. Hitz, "SCION Control Plane", 2024, <https://datatracker.ietf.org/doc/draft-dekater-scion-controlplane/>.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <https://www.rfc-editor.org/rfc/rfc1122>.

[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/rfc/rfc1918>.

[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/rfc/rfc2119>.

[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/rfc/rfc2474>.

[RFC2711] Partridge, C. and A. Jackson, "IPv6 Router Alert Option", RFC 2711, DOI 10.17487/RFC2711, October 1999, <https://www.rfc-editor.org/rfc/rfc2711>.

[RFC3168] 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/rfc/rfc3168>.

[RFC4493] Song, JH., Poovendran, R., Lee, J., and T. Iwata, "The AES-CMAC Algorithm", RFC 4493, DOI 10.17487/RFC4493, June 2006, <https://www.rfc-editor.org/rfc/rfc4493>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/rfc/rfc5880>.

[RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, DOI 10.17487/RFC5881, June 2010, <https://www.rfc-editor.org/rfc/rfc5881>.

[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/rfc/rfc8174>.

[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/rfc/rfc8200>.

[RFC9473] Enghardt, R. and C. Krähenbühl, "A Vocabulary of Path Properties", RFC 9473, DOI 10.17487/RFC9473, September 2023, <https://www.rfc-editor.org/rfc/rfc9473>.

7.2. Informative References

[CHUAT22] Chuat, L., Legner, M., Basin, D., Hausheer, D., Hitz, S., Mueller, P., and A. Perrig, "The Complete Guide to SCION", ISBN 978-3-031-05287-3, 2022, <https://doi.org/10.1007/978-3-031-05288-0>.

[I-D.scion-components] Rustignoli, N. and C. de Kater, "SCION Components Analysis", 2023, <https://datatracker.ietf.org/doc/draft-rustignoli-panrg-scion-components/>.

[I-D.scion-cppki] de Kater, C., Rustignoli, N., and S. Hitz, "SCION Control-Plane PKI", 2024, <https://datatracker.ietf.org/doc/draft-dekater-scion-pki/>.

[I-D.scion-overview] de Kater, C., Rustignoli, N., and A. Perrig, "SCION Overview", 2023, <https://datatracker.ietf.org/doc/draft-dekater-panrg-scion-overview/>.

Acknowledgments

Many thanks go to Matthias Frei (SCION Association), Juan A. Garcia Prado (ETH Zurich), Kevin Meynell (SCION Association) and Jean-Christophe Hugly (SCION Association) for reviewing this document. We are also very grateful to Adrian Perrig (ETH Zurich), for providing guidance and feedback about each aspect of SCION. Finally, we are indebted to the SCION development teams of Anapaya and ETH Zurich, for their practical knowledge and for the documentation about the SCION Data Plane, as well as to the authors of [CHUAT22] - the book is an important source of input and inspiration for this draft.

Assigned SCION Protocol Numbers

This appendix lists the assigned SCION protocol numbers.

Considerations

SCION attempts to take the IANA's assigned Internet protocol numbers into consideration. Widely employed protocols have the same protocol number as the one assigned by IANA. SCION specific protocol numbers start at 200.

The protocol numbers are used in the SCION header to identify the next level protocol.

Assignment

Decimal	Keyword	Protocol
0-5		Unassigned
6	TCP/SCION	Transmission Control Protocol over SCION
7-16		Unassigned
17	UDP/SCION	User Datagram Protocol over SCION
18-199		Unassigned
200	HBH	SCION Hop-by-Hop Options
201	E2E	SCION End-to-End Options
202	SCMP	SCION Control Message Protocol
203	BFD/SCION	BFD over SCION
204-252		Unassigned
253		Use for experimentation and testing
254		Use for experimentation and testing
255		Reserved

Table 10: The assigned SCION protocol numbers

Authors' Addresses

Corine de Kater
SCION Association

Email: c_de_kater@gmx.ch

Nicola Rustignoli
SCION Association

Email: nic@scion.org

Samuel Hitz
Anapaya Systems

Email: hitz@anapaya.net