Network Working Group
Internet-Draft
Intended status: Informational
Expires: 17 September 2025

A. Martishina, Ed.
A. Urivskiy
A. Rybkin
L. Tychina
I. Parshin
InfoTeCS

16 March 2025

IPlir network layer security protocol
draft-iplir-protocol-07

Abstract

This document specifies the IPlir network layer security protocol. It describes how to provide a set of security services for traffic over public and corporate networks using the TCP/IP stack.

Status of This Memo

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

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

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

This Internet-Draft will expire on 17 September 2025.

Table of Contents

1. Introduction

1.1. Scope

The IPlir protocol may be used to protect IP packets during their transmission via communication channels. IP packet protection means ensuring data integrity and authenticity of the data source of the packets. For this purpose, when IPlir is applied, encapsulation of original IP packets, calculation of message authentication codes for the encapsulated packets and service information are used. IP packet protection also means option of ensuring their confidentiality and uses packet encryption for this purpose. In addition, the IPlir

protocol allows for protection against replay attacks based on the use of counter values and/or timestamps.

The IPlir protocol can be used to create Virtual Private Networks at the network layer of the basic ISO OSI reference model. Data is protected during transfer of IP packets between any two hosts supporting the IPlir protocol, including options of data exchange between two end hosts, an end host and a security gateway, and two security gateways. All protection mechanisms are implemented without establishing a connection (in terms of network) between the two interacting hosts.

This document is not a Security Architecture for the Internet; it addresses security only at the network layer, provided through the use of a combination of cryptographic and protocol security mechanisms.

This document does not have IETF consensus and does not imply IETF support for IPlir.

1.2. Audience

The target audience for this document is primarily individuals who implement this network layer security technology or who architect systems that will use this technology. Technically adept users of this technology (end users or system administrators) also are part of the target audience.

This document assumes that the reader is familiar with the Internet Protocol (IP), related networking technology, and general information system security terms and concepts.

2. Conventions Used in This Document

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

3. Notations

The following notations are used in this document:

LSB_s(x): truncation of binary string x with a length of m, m >= s, to a binary string with a length of s according to the following rule: LSB_s(x_{m-1} || ... || x_1 || x_0) = x_{s-1} || ... || x_1 || x_0, x_i \in V_1, i = 0,1,⋯,m-1.

IntToVec_s(x): representation of the ring element x \in Z_{2^s} as a binary string with a length of s: for x = x_0 + 2 * x_1 + ... + 2^{s-1} * x_{s-1}, where x_i \in V_1, i = 0,1,…,s-1, the following equation is true: IntToVec_s(x) = x_{s-1} || ... || x_1 || x_0.

CharToByte('c'): representation of the 'c' symbol as a binary string of length 8 calculated according to the following rule: CharToByte('c') = 0 || IntToVec_7(ASCII('c')), where ASCII('c') \in Z_{2^7} is the ASCII representation of the 'c' symbol.

StrToVec_s('string'): representation of the string of symbols 'string' = 'c_{m-1}c_{m-2}...c_0' consisting of m symbols as a binary string with a length of s, s >= 8*m according to the following rule: StrToVec_s('c_{m-1}c_{m-2}...c_0') = 0^{s-8*m} || CharToByte('c_{m-1}') || CharToByte('c_{m-2}') || ... || CharToByte('c_0').

4. System overview

4.1. IPlir packet format

An IPlir packet is an IP packet protected by IPlir. Its format is shown in Figure 1.

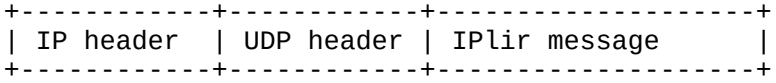


Figure 1: IPlir packet structure

The IP header is the header of a standard IP packet, where the Protocol field for IPv4 and the NextHeader field for IPv6 contain the value 99.

The UDP header is a standard UDP header used to multiplex by port number, i.e. to choose an appropriate application program at a target host to process an incoming IPlir message.

The IPlir message is the main part of the IPlir packet that includes protected data from the original IP packet and plaintext data required for the IPlir message processing.

4.2. IPlir message format

The IPlir message contains:

* IPlir header containing plaintext information related to encapsulation and protection of the original IP packet;

* IPlir body containing information the encryption of which is optional;

* IPlir trailer containing MACs, the transit host (the host implementing the function of routing IPlir packets) identifier, and the transit initialization value.

The IPlir message structure is shown in Figure 2.

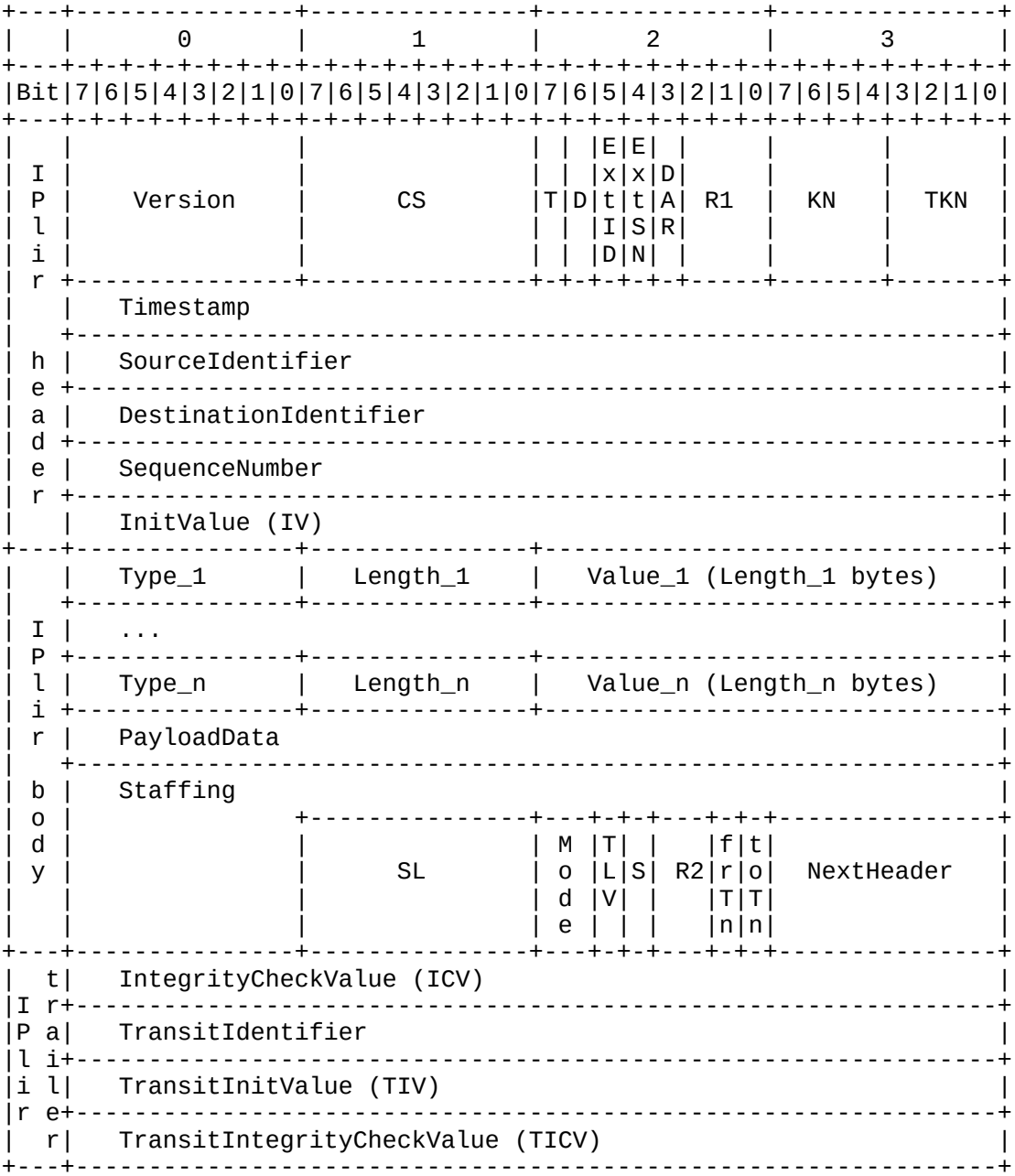


Figure 2: IPlir message structure

The IPlir message fields have a big-endian order of bytes. The numbering is left-to-right, the high bytes have lower numbers. Numbering of bits inside bytes is right-to-left, the high bits have higher numbers.

4.2.1. Version

IPlir header version. This document describes the IPlir header of Version = 1. The field length is 8 bit.

4.2.2. CS

Cryptographic suite identifier determining the contents of cryptographic mechanisms and their parameters used to create the IPlir packet. The field length is 8 bit.

4.2.3. T

The transit MAC flag. If T = 1, the IPlir trailer contains fields TransitIdentifier, TransitIntegrityCheckValue and TransitInitValue, otherwise the fields are absent. T field has to be set to 0 when calculating and checking the end-to-end MAC (ICV). The field length is 1 bit.

4.2.4. D

The DestinationIdentifier field flag. If D = 1, the header contains the DestinationIdentifier field, otherwise the field is absent. The destination host identifier is required for routing of IPlir packets. The field length is 1 bit.

4.2.5. ExtID

The extended network host identifiers flag. If ExtID = 0, the SourceIdentifier field and, if available, DestinationIdentifier and TransitIdentifier fields are 32 bits long. If ExtID = 1, all these network host identifiers are 64 bits long. The field length is 1 bit.

4.2.6. ExtSN

The packet extended sequence number flag. If ExtSN=0, the SequenceNumber field is 32 bits long. If ExtSN = 1, the SequenceNumber field is 64 bits long. The field length is 1 bit.

4.2.7. DAR

The flag of disabling the anti-replay mechanism. The use of the flag is regulated by security policies. The field length is 1 bit.

4.2.8. R1

The field reserved for future use. When IPlir header is generated, the field must contain all zeros. The receiving end must not analyze the field content. The field length is 3 bits.

4.2.9. KN

The number of the exchange key used to encrypt and calculate the end-to-end MAC. The field length is 4 bits.

4.2.10. TKN

The number of the exchange key used to calculate the transit MAC. If the transit MAC is not used, i.e., T = 0, the field value should be 0. When calculating and checking the end-to-end MAC (ICV), the TKN field must be filled with zeros. The field length is 4 bits.

4.2.11. Timestamp

Packet send time. The field contains the send time value based on the astronomical clock of the source host in the POSIX time format less 0x40000000 seconds. The estimated overflow time is the year 2140. The field length is 32 bits.

4.2.12. SourceIdentifier

The source host identifier is used by the destination host to identify the IPlir packet source and the related context of the source host for packet processing. The field length is 32 bits with ExtID = 0, or 64 bits with ExtID = 1.

4.2.13. DestinationIdentifier

The destination host identifier is required for routing of IPlir packets. The field is available, if D = 1. The field length is 32 bits with ExtID = 0, or 64 bits with ExtID = 1.

4.2.14. SequenceNumber

The packet sequence number; an unsigned integer. The field length is 32 bits with ExtSN = 0, or 64 bits with ExtSN = 1.

4.2.15. InitValue (IV)

An end-to-end initialization value that can be used to execute encryption operations and calculate an end-to-end MAC, as well as to derive keys for these operations. The field length is determined by the used cryptographic suite.

4.2.16. Tuples (Type, Length, Value)

Tuples (Type, Length, Value) enable transmission of additional information within the IPlir message. The Type field contains the type of the value in the Value field. The Type field length is 8 bit. The Length field contains the byte length of the Value field. The Length field length is 8 bit. The Value field contains a data of the Type field type. The Value field length of any tuple should be divisible by 8 bits and be less than 255 bytes.

Permissible Type field values for the tuples are provided in Table 1.

Type value	Description
0	the last tuple in the IPlir message; can be used by the vendor for its own needs
1	a pair of IPv4 addresses
2	a pair of IPv6 addresses
3-128	not in use
129-254	can be used by the vendor for its own needs
255	not in use

Table 1: Type Field Values for Tuples

The last tuple in the message always has type 0. The length of this tuple should be chosen so as to ensure effective IPlir message processing.

4.2.16.1. Pair of IPv4 addresses

The Value field of the tuple of this type contains a pair of IPv4 addresses. The source address comes first, followed by the destination address. The addresses are transmitted in the big-endian order of bytes.

The main function of the tuple of this type is to preserve the IPv4 addresses from the original IP packet in the light tunnel mode.

The tuple structure is shown in Figure 3.

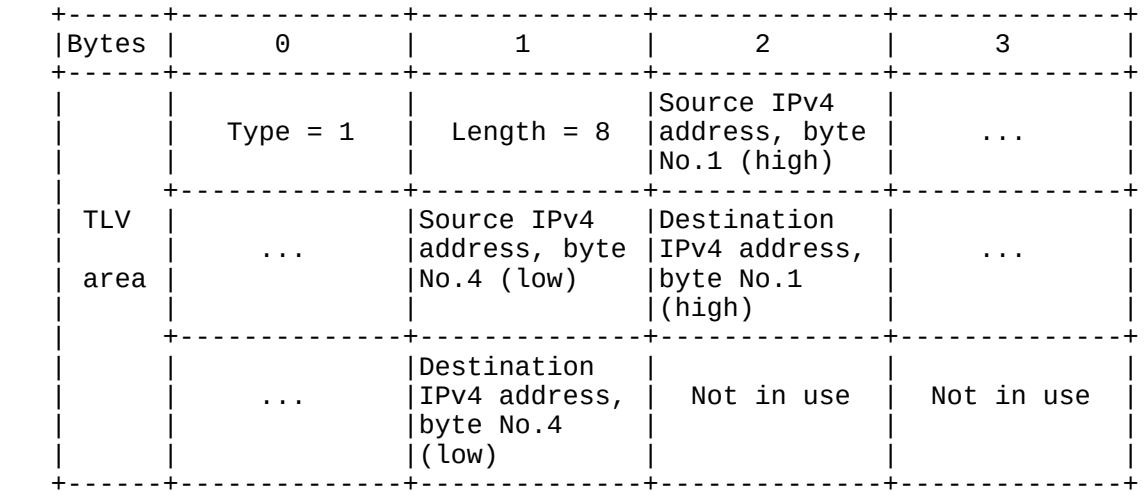


Figure 3: Type = 1 Tuple Structure

Note: Bytes labeled “not in use” contain information related to the following tuple.

4.2.16.2. Pair of IPv6 addresses

The Value field of the tuple of this type contains a pair of IPv6 addresses. The source address comes first, followed by the destination address. The addresses are transmitted in the big-endian order of bytes.

The main function of the tuple of this type is to preserve the IPv6 addresses from the original IP packet in the light tunnel mode.

The tuple structure is shown in Figure 4.

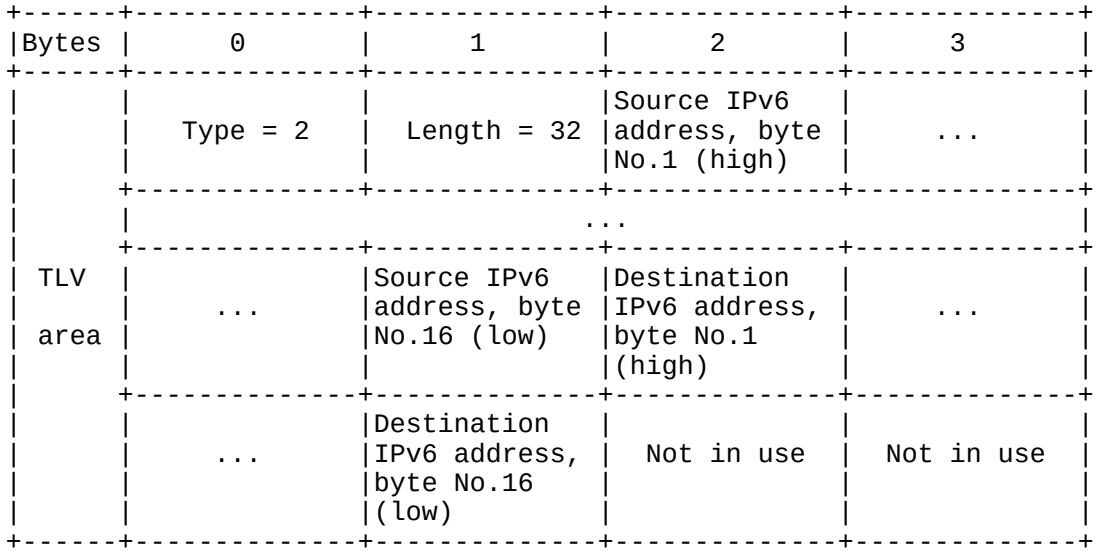


Figure 4: Type = 2 Tuple Structure

Note: Bytes labeled “not in use” contain information related to the following tuple.

4.2.17. PayloadData

A variable-length field containing the original IP packet or its part, depending on the IPlir operation mode.

4.2.18. Staffing

A (network) filler to make the length of the IPlir message more suitable for efficient processing. The Staffing field contains a sequence of integers in a binary form: the first byte contains 1, the second one contains 2, etc. The field length is determined by the SL field value, if SL is absent (S = 0) or has the value 0, there is no Staffing field.

4.2.19. SL

The number of bytes in the Staffing field. The field is available, if S = 1. The field length is 8 bit.

4.2.20. Mode

The mode in which the IPlir packet was generated in. The field length is 2 bits.

4.2.21. TLV

Type, Length and Value fields flag. If TLV = 1, the IPlir body begins with tuples (Type, Length, Value), otherwise there are no tuples. The field length is 1 bit.

4.2.22. S

The SL field flag. If S = 1, the IPlir body contains the SL field, otherwise this field is absent. The field length is 1 bit.

4.2.23. R2

The field reserved for future use. When an IPlir message is generated, the field must contain all zeros. The receiving end must not analyze the field content. The field length is 2 bits.

4.2.24. frTN

The field reserved for future use. An originating security (VPN) gateway uses this flag to mark IPlir packets that were created from routed packets originally sent by other network devices. This flag cannot be set if the DAR flag is set. The field length is 1 bit.

4.2.25. toTN

The field reserved for future use. An originating host uses this flag to mark IPlir packets that are destined to other network devices but not to the destination host specified by the DestinationId field. This flag cannot be set if the DAR flag is set. The field length is 1 bit.

4.2.26. NextHeader

The original IP packet protocol number. The field length is 8 bit.

4.2.27. IntegrityCheckValue (ICV)

An end-to-end MAC calculated for the data from the IPlir message start to the NextHeader field inclusive. The field length is determined by the used cryptographic suite.

4.2.28. TransitIdentifier

The identifier of the transit host that routed the IPlir packet last. Each transit host updates the field value with its identifier. The field is available, if T = 1. The field length is 32 bits with ExtID = 0, or 64 bits with ExtID = 1.

4.2.29. TransitInitValue (TIV)

The transit initialization value used to calculate a transit MAC and derive keys for this operation. The field is available, if T = 1. The field length is determined by the used cryptographic suite.

4.2.30. TransitIntegrityCheckValue (TICV)

A transit MAC calculated for the data from the IPlir message start to the TransitInitValue field inclusive. The field is available, if T = 1. The field length is determined by the used cryptographic suite.

4.3. IPlir packet structure and IPlir header location

The IPlir protocol can operate in three modes: transport, tunnel, and light tunnel. The transport and light tunnel modes ensure protection of the data generated by protocols above the IP level in the basic ISO OSI reference model, in particular, by the transport layer. The tunnel mode ensures protection of the entire original IP packet.

The receiving end determines based on the value of the Mode field in what mode the packet was sent. Possible field values are shown in Table 2.

Mode field value	Mode description
0	transport mode
1	light tunnel mode
2	tunnel mode
3	reserved for future needs

Table 2: Mode Field Values

4.3.1. Transport mode

In the transport mode, the header for transport layer (UDP) and the IPlir header with (Type, Length, Value)-tuples follow the IP header and precede the header of the next layer (e.g., TCP, UDP, ICMP, etc.). For IPv4 it means that the IPlir header is located after the IP header, including all options in the original IP packet, but before the header of the next level protocol.

For IPv6 the IPlir header is addressed to the destination endpoint. Therefore, it should be placed after the Hop-by-hop, Routing, and Fragmentation extension headers. The Destination Options extension headers can be located before, after, or on both sides of the IPlir header, depending on the required semantics. However, since the IPlir protocol can ensure privacy of only the fields following the IPlir header, the destination options should follow the IPlir header.

Figure 5 shows an example of IP packet protection using the IPlir in the transport mode.

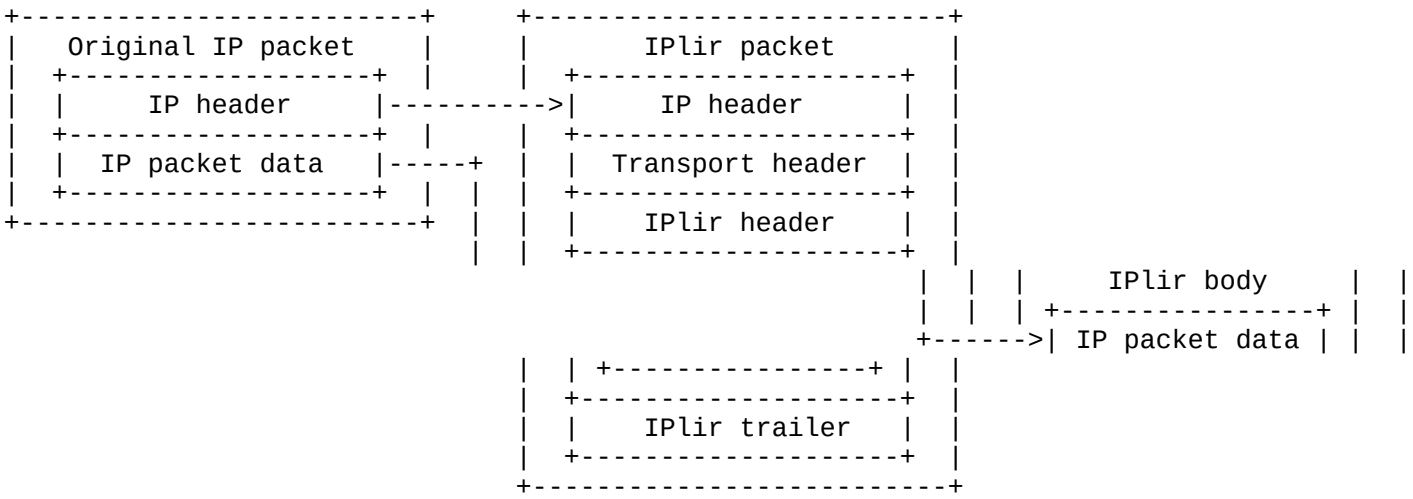


Figure 5: IP Packet Protection Using IPlir in the Transport Mode

4.3.2. Light tunnel mode

Location of the header for transport layer (UDP) and the IPlir header with (Type, Length, Value)-tuples in the light tunnel mode is the same as in the transport mode. An exception is that the set of tuples in the IPlir body must include a type 1 tuple (two IPv4 addresses) or a type 2 tuple (two IPv6 addresses), wherein the source and destination addresses are specified from the IP header of the original IP packet. The tuple type is defined by the version of the IP header of the original IP packet.

The destination host can restore the original IP addresses from the available tuple Value field.

Unlike the transport mode, the light tunnel mode makes it possible to change addresses in the IPlir packet IP header.

Figure 6 shows an example of IP packet protection using the IPlir in the light tunnel mode.

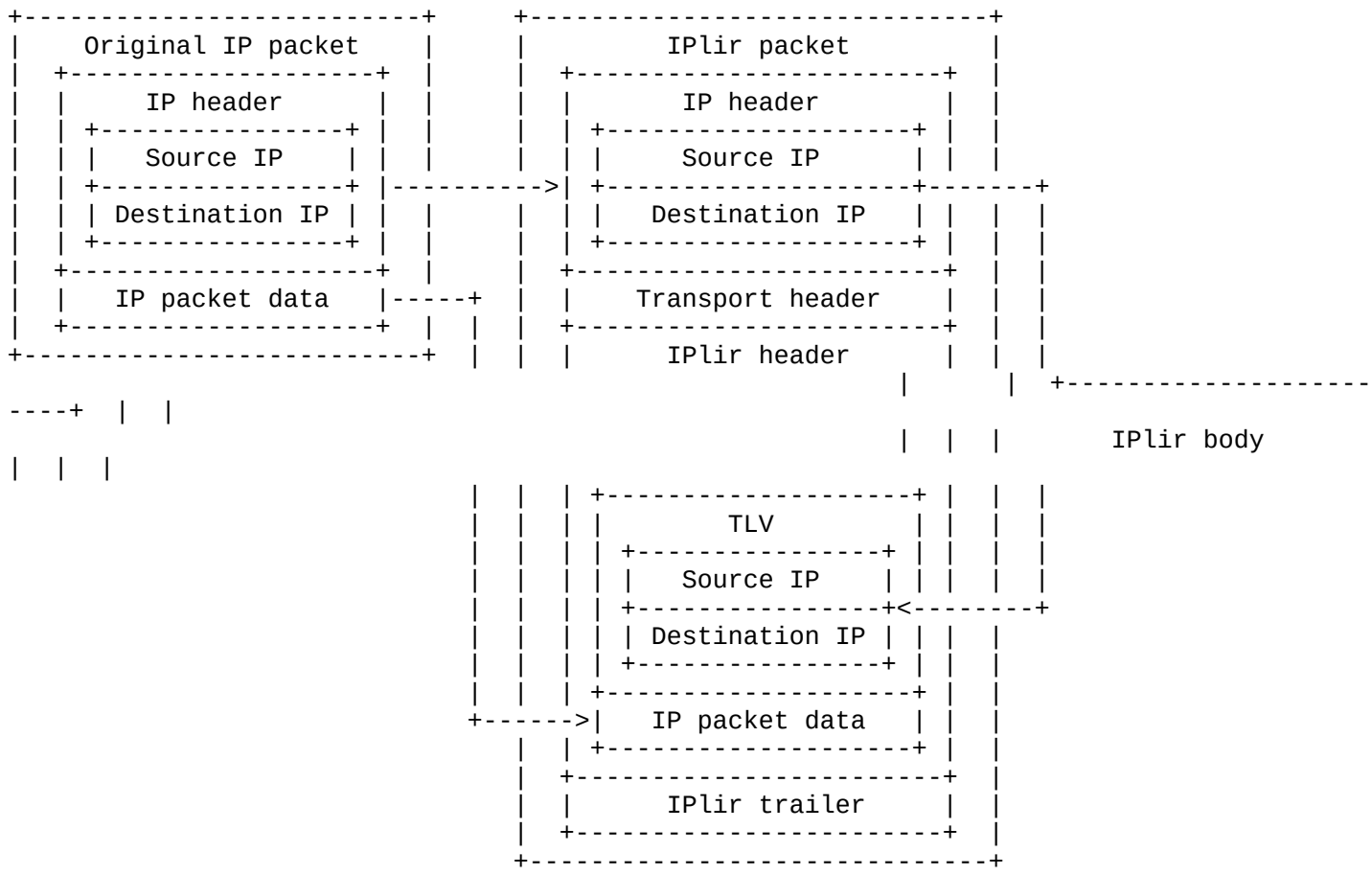


Figure 6: IP Packet Protection Using IPlir in the Light Tunnel Mode

4.3.3. Tunnel mode

Unlike the other modes, the tunnel mode protects the entire original IP packet, including its IP header.

In the tunnel mode, a new IP header is generated with its contents based on the destination host context and the source host IP routing table, followed by the header for transport layer (UDP) and the IPlir header with (Type, Length, Value)-tuples. This is followed by the original IP packet.

Versions of the original and new IP headers can be different. It means that IPv6 packets can be transmitted via the IPv4 protocol and vice versa.

Figure 7 shows an example of IP packet protection using the IPlir in the tunnel mode.

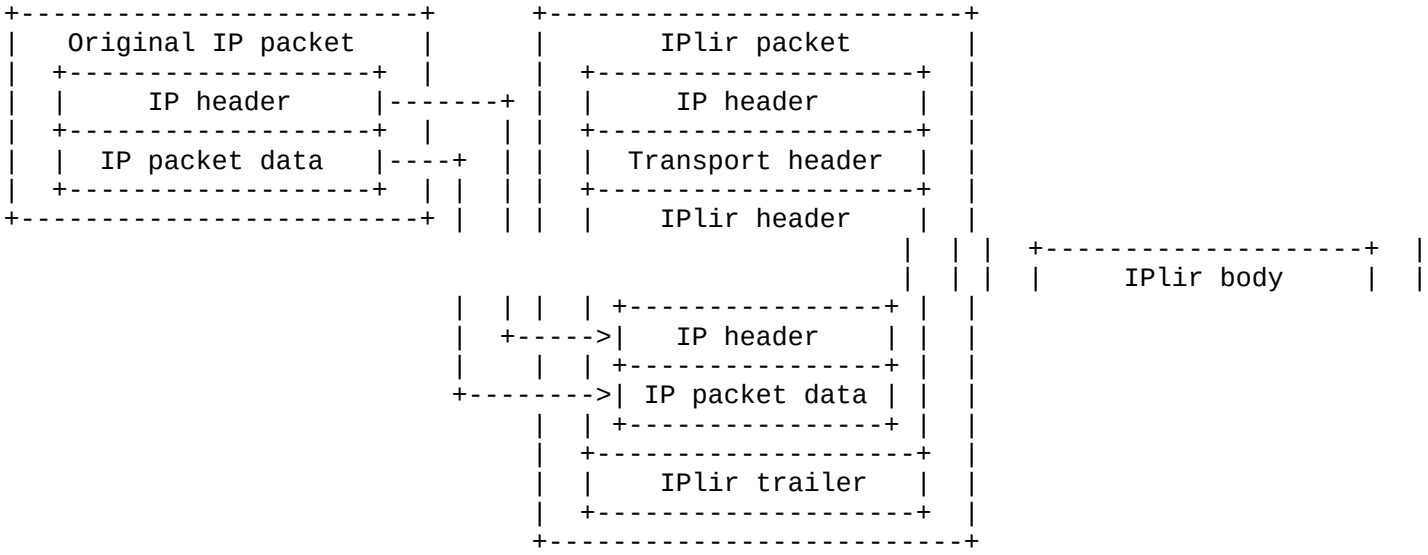


Figure 7: IP Packet Protection Using IPlir in the Tunnel Mode

5. Security Considerations

5.1. Encryption and MAC

To ensure the confidentiality of the packet, it is possible to encrypt it using a symmetric cryptographic algorithm. Packet encryption in IPlir is recommended, but not required. Encryption can be disabled by selecting a separate cryptographic suite clearly indicating that there is no encryption. In case of encryption, it is applied between the source and destination hosts regardless of the transfer network topology and existence of transit hosts.

To ensure packet integrity and authenticity of the data source, the IPlir protocol allows for end-to-end or transit MAC. End-to-end MAC is applied between the source and destination hosts. It is mandatory. Transit MAC is applied between two neighbor hosts in a packet transfer chain. It is optional.

To ensure confidentiality, packet integrity and authenticity of the data source, either separate encryption and MAC algorithms or AEAD algorithms to encrypt and calculate MAC simultaneously can be used.

5.2. Cryptographic keys

It is implied that there is a key system that provides the interacting hosts with necessary exchange keys and controls their synchronization. Exchange key is a key known only to the specified pair of hosts are used to derive keys. Keys can be managed manually or automatically.

The exchange keys required to process a specific packet with all their mandatory attributes (meta data) must be available when packet processing starts.

The packet encryption keys, end-to-end MAC keys and transit MAC keys are derived from the exchange key to protect each IP packet. Each exchange key related to the specific pair of hosts is indexed with the corresponding pair of their identifiers. It is possible to use key systems in which several exchange keys exist (up to 16) simultaneously for two hosts. To make it possible, each exchange key in the IPlir protocol is additionally indexed with an integer value between 0 and 15 (inclusive) located in the KN or TKN field of the IPlir message. This allows to unambiguously determine the exchange key for the two hosts.

The peculiarity of IPlir is that unique packet encryption, end-to-end MAC and transit MAC keys used in the corresponding cryptographic algorithms are derived for each IP packet based on the exchange keys. The packet encryption, end-to-end MAC and transit MAC keys derived for the same IP packet should be different, except when AEAD algorithms are used, where one packet encryption and end-to-end MAC key is used for encryption and end-to-end MAC of the packet.

The exchange key used to derive packet encryption and end-to-end MAC keys (or packet encryption and end-to-end MAC key) is determined by the KN field value of the IPlir message and identifiers of the source and destination hosts. The exchange key used to derive the packet transit MAC key is determined by the TKN field value of the IPlir message and identifiers of the interacting (transit) hosts.

Exchange key types and methods to derive packet protection keys from them are determined by the cryptographic suite.

For any unique key used in the IPlir protocol at any time it should be impossible to calculate it from the other keys, except for calculation of derived keys for packet protection from a specific exchange key.

The maximum quantity of material that can be processed using the same key should be determined considering theoretical limits arising from the use of particular cryptographic algorithms and practical limits arising during IPlir implementation.

The maximum number of keys (packet encryption, end-to-end MAC, transit MAC keys or packet encryption and end-to-end MAC keys) derived from one exchange key should be determined considering theoretical limits arising from the use of particular cryptographic algorithms and practical limitations arising during IPlir implementation.

When the allowed limit for a specific key is reached, the interacting parties should stop using it. For protection of further interactions, the parties should use a key for which the allowed limit has not been achieved, e.g., a new key.

5.3. Cryptographic suites

The cryptographic algorithms and parameters used in the IPlir protocol make up a cryptographic suite designated by its CS number in the CS field of each IPlir message. There can be up to 256 different cryptographic suites in total.

Permissible CS field values are provided in Table 3:

CS value	Description
0	not in use
1	MAGMA-MGM cryptographic suite
2	KUZN-CTR-CMAC cryptographic suite
3-128	not in use
129	AES-128-GCM cryptographic suite
130-131	can be used by the vendor for its own needs
132	AES-256-CTR-CMAC cryptographic suite
133	can be used by the vendor for its own needs
134	AES-256-CFB-CMAC cryptographic suite
135-254	can be used by the vendor for its own needs
255	Not in use

Table 3: Permissible CS Field Values

The list of main mechanisms and parameters specified in the cryptographic suite is shown in Table 4:

Parameter	Description	Purpose
EncAlg	encryption algorithm	the algorithm is used to encrypt the packet
MACAlg	end-to-end MAC algorithm	the algorithm is used to calculate packet end-to-end MAC
TMACAlg	transit MAC algorithm	the algorithm is used to calculate packet transit MAC
MACLen	end-to-end MAC length	
TMACLen	transit MAC length	
IVLen	end-to-end initialization value length	the initialization value can be used for packet encryption, end-to-end MAC, and derivation of packet encryption keys and packet end-to-end MAC keys (or packet encryption and end-to-end MAC keys)
TIVLen	transit initialization value length	the transit initialization value can be used for packet transit MAC and derivation of packet transit MAC keys
KDAlg	algorithms of deriving packet protection keys from exchange keys	the algorithms are used to derive packet encryption keys and packet end-to-end MAC keys (or packet encryption and end-to-end MAC keys), and to derive packet transit MAC keys

Table 4: Main Mechanisms And Parameters In The Cryptographic Suite

5.3.1. MAGMA-MGM cryptographic suite: CS = 1

MAGMA-MGM Cryptographic Suite Description is shown in Table 5:

Parameter	Value
EncAlg	GOST R 34.122015 (Magma) [RFC8891] in the MGM mode [RFC9058]
MACAlg	GOST R 34.122015 (Magma) [RFC8891] in the MGM mode [RFC9058]
TMACAlg	GOST R 34.122015 (Magma) [RFC8891] in the MGM mode [RFC9058]
MACLen	32 bits
TMACLen	32 bits
IVLen	64 bits
TIVLen	64 bits
KDAlg	see the description below

Table 5: MAGMA-MGM cryptographic suite: CS = 1

5.3.1.1. Exchange keys

For each pair of interacting hosts, there is a single exchange key with a length of 256 bits used for deriving of packet encryption and end-to-end MAC keys, as well as packet transit MAC keys.

5.3.1.2. Requirements for initialization values

The end-to-end initialization value InitValue in the InitValue field of the IPlir message should have a length of 64 bits and be unique for each IPlir packet the encryption and end-to-end MAC of which are implemented by the same source host using the same exchange key.

The transit initialization value TransitInitValue in the TransitInitValue field of the IPlir message should have a length of 64 bits and be unique for each IPlir packet the transit MAC of which is implemented by the same (transit) host using the same exchange key.

5.3.1.3. Key derivation algorithms

The packet encryption and end-to-end MAC key K_AEAD of 256 bit length is calculated as follows:

K_AEAD = K_1 || K_2 || K_3 || K_4,

where each value of K_i \in V_64, i = 1,2,3,4 is calculated as per GOST R 34.122015 (Magma) [RFC8891] in the CMAC mode as per ISO/IEC 9797-1:2011 [ISO9797-1], wherein

* the exchange key is used as the key. The exchange key is specified by the source and destination hosts and the KN field value of the IPlir message,

* a binary string as shown below is used as the data: IntToVec_8(i)||Label||aL||IV_KDF||SN||Node||cL||oL, where

  - Label = StrToVec_48('AEAD'),

  - aL = IntToVec_8(LabelByteLength), where LabelByteLength = 6,

  - IV_KDF = InitValue, where InitValue is initialized by the InitValue field value of the IPlir message,

  - SN = SequenceNumber, where SequenceNumber is initialized by the SequenceNumber field value of the IPlir message,

  - Node = SourceIdentifier, where SourceIdentifier is initialized by the SourceIdentifier field value of the IPlir message,

  - cL = IntToVec_16(ContextByteLength), where ContextByteLength is the sum of byte lengths of the InitValue, SequenceNumber and SourceIdentifier fields of the IPlir message,

  - oL = IntToVec_16(OutputBitLength), where OutputBitLength = 256,

* the MAC length is 64 bits.

The packet transit MAC key K_TMAC of 256 bit length is calculated as follows:

K_TMAC = K_1 || K_2 || K_3 || K_4,

where each value of K_i \in V_64, i = 1,2,3,4 is calculated as per GOST R 34.122015 (Magma) [RFC8891] in the CMAC mode, as per ISO/IEC 9797-1:2011 [ISO9797-1], wherein

* the exchange key is used as the key. The exchange key is specified by the (transit) hosts the IPlir packet passes through and the TKN field value of the IPlir message,

* a binary string as shown below is used as the data: IntToVec_8(i)||Label||aL||TIV_KDF||SN||Node||cL||oL, where

  - Label = StrToVec_48('TMAC'),

  - aL = IntToVec_8(LabelByteLength), where LabelByteLength = 6,

  - TIV_KDF = TransitInitValue, where TransitInitValue is initialized by the TransitInitValue field value of the IPlir message,

  - SN = SequenceNumber, where SequenceNumber is initialized by the SequenceNumber field value of the IPlir message,

  - Node = TransitIdentifier, where TransitIdentifier is initialized by the TransitIdentifier field value of the IPlir message,

  - cL = IntToVec_16(ContextByteLength), where ContextByteLength is the sum of byte lengths of the TransitInitValue, SequenceNumber and TransitIdentifier fields of the IPlir message,

  - oL = IntToVec_16(OutputBitLength), where OutputBitLength = 256,

* the MAC length is 64 bits.

5.3.1.4. Encryption and MAC algorithms

Encryption of the IPlir body and calculation of the end-to-end MAC ICV in the IntegrityCheckValue field of the IPlir message are implemented as per GOST R 34.122015 (Magma) [RFC8891] in the MGM mode [RFC9058], wherein

* the packet encryption and end-to-end MAC key K_AEAD is used as the encryption key,

* data in the IPlir header fields in the order of their appearance in the IPlir message are used as associated authenticated data,

* data in the IPlir body fields in the order of their appearance in the IPlir message are used as plaintext,

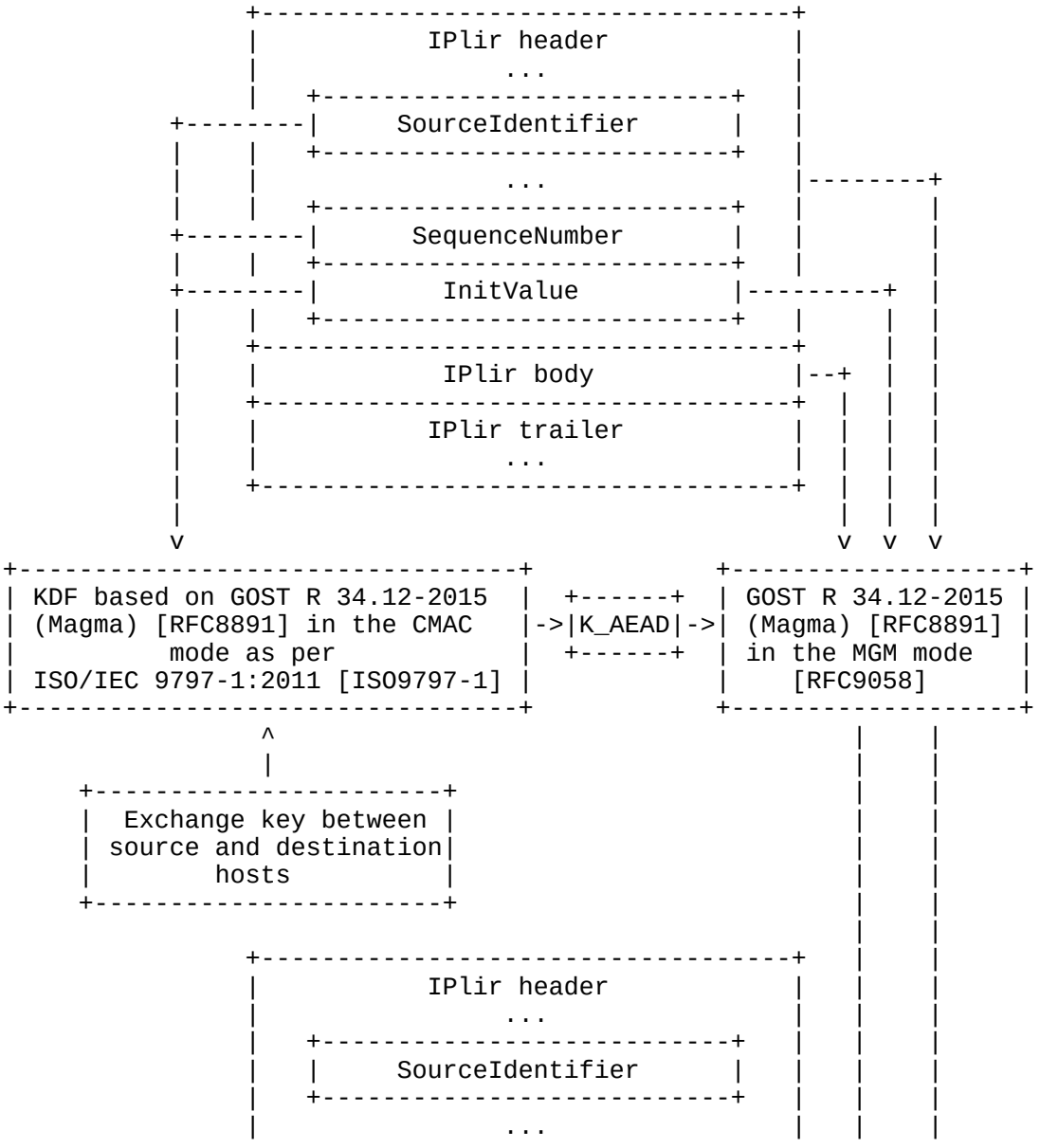
* the value of IV_AEAD \in V_63 is used as the initial counter nonce:

  IV_AEAD = LSB_63(InitValue),

where InitValue is initialized by the InitValue field value of the IPlir message,

* the MAC length is 32 bits.

The diagram of encryption and end-to-end MAC is shown in Figure 8.



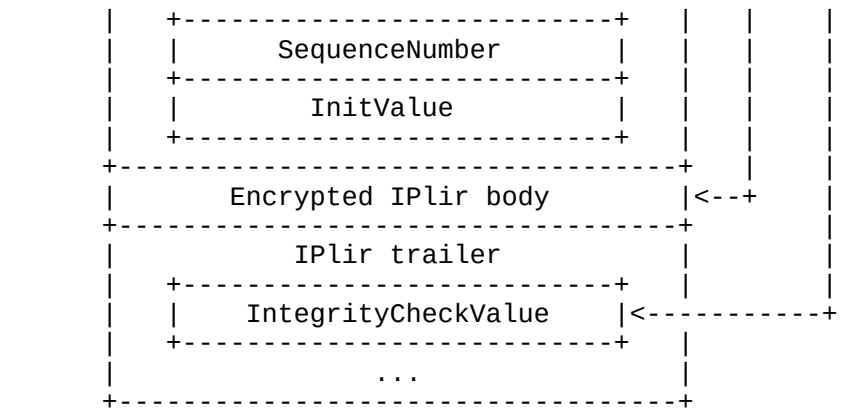


Figure 8: Diagram of Encryption and End-to-End MAC Using the MAGMA-MGM Cryptographic Suite

Calculation of the transit MAC TICV in the TransitIntegrityCheckValue field of the IPlir message is implemented as per the GOST R 34.122015 (Magma) [RFC8891] in the MGM mode [RFC9058], wherein

* the packet transit MAC key K_TMAC is used as the encryption key,

* data in the IPlir header fields, the encrypted IPlir body and IntegrityCheckValue, TransitIdentifier, TransitInitValue fields data in the order of their appearance in the IPlir message are used as the associated authenticated data,

* plaintext is an empty string,

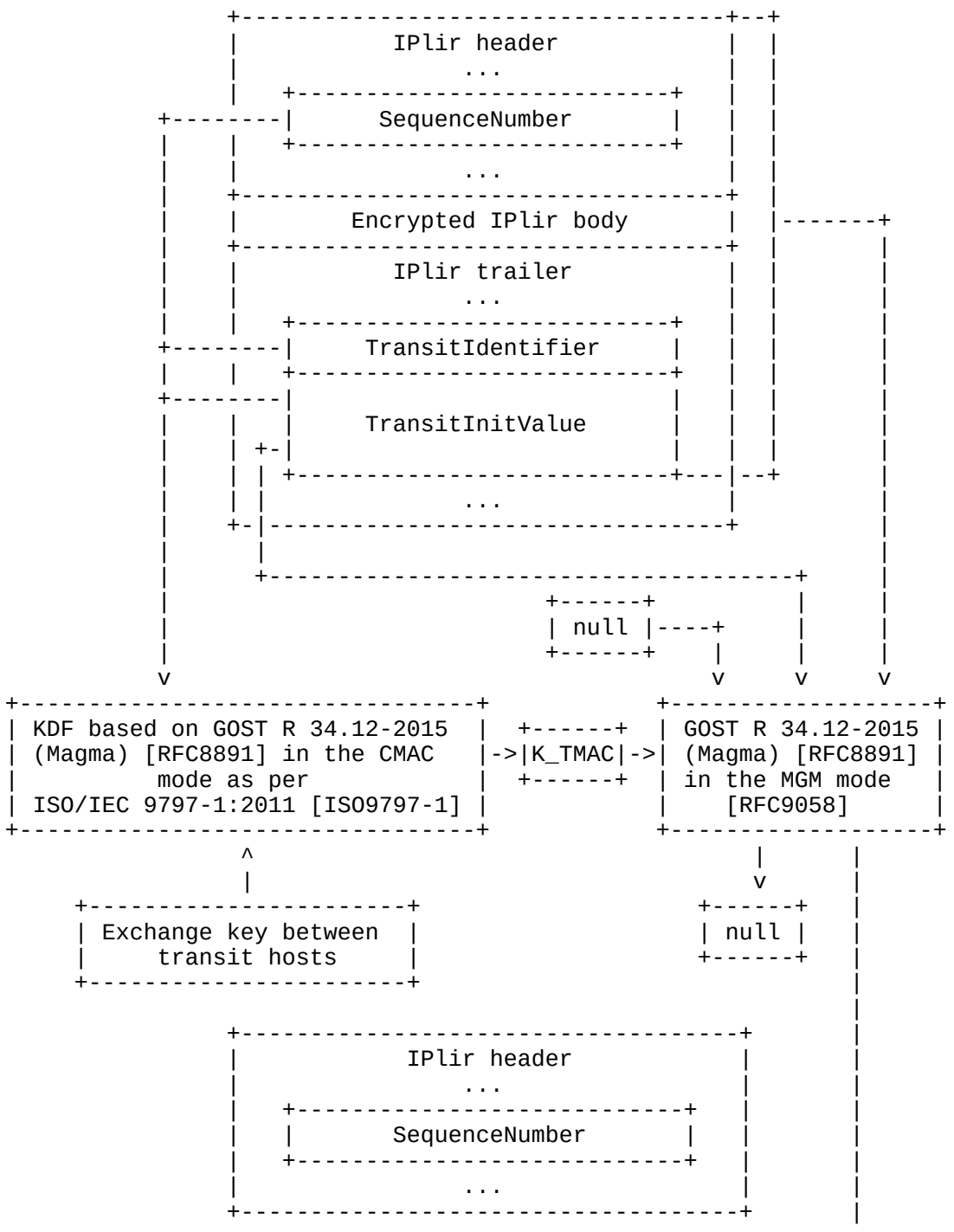
* the value of TIV_AEAD \in V_63 is used as the initial counter nonce:

  TIV_AEAD = LSB_63(TransitInitValue),

  where TransitInitValue is initialized by the TransitInitValue field value of the IPlir message,

* the MAC length is 32 bits.

The diagram of transit MAC is shown in Figure 9. The “null” value means an empty binary string.


IPlir header
SequenceNumber
Encrypted IPlir body
IPlir trailer
TransitIdentifier
TransitInitValue
null
KDF based on GOST R 34.12-2015 (Magma) [RFC8891] in the CMAC mode as per ISO/IEC 9797-1:2011 [ISO9797-1]
K_TMAC
GOST R 34.12-2015 (Magma) [RFC8891] in the MGM mode [RFC9058]
Exchange key between transit hosts
null
IPlir header
SequenceNumber

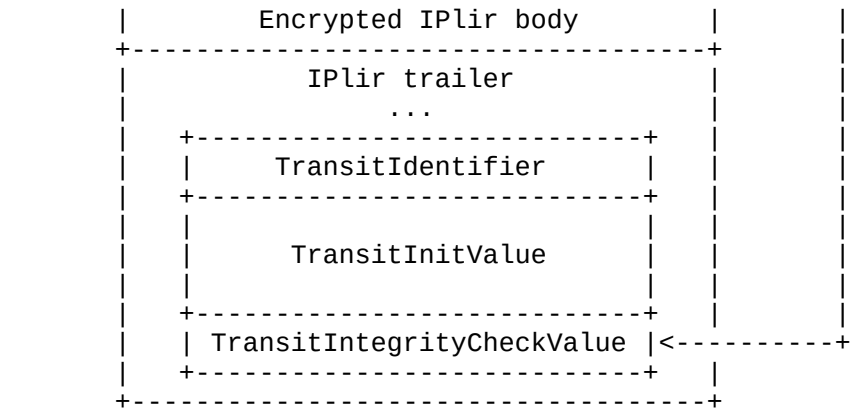


Figure 9: Diagram of Transit MAC Using the MAGMA-MGM Cryptographic Suite

5.3.2. KUZN-CTR-CMAC cryptographic suite: CS=2

KUZN-CTR-CMAC Cryptographic Suite Description is shown in Table 6:

Parameter	Value
EncAlg	GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CTR mode [ISO10116]
MACAlg	GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CMAC mode [ISO9797-1]
TMACAlg	GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CMAC mode [ISO9797-1]
MACLen	64 bits
TMACLen	64 bits
IVLen	64 bits
TIVLen	64 bits
KDAlg	see the description below

Table 6: KUZN-CTR-CMAC cryptographic suite: CS=2

5.3.2.1. Exchange keys

For each pair of interacting hosts, there is a single exchange key with a length of 256 bits designed for derivation of packet encryption keys, end-to-end MAC keys, and transit MAC keys.

5.3.2.2. Requirements for initialization values

The end-to-end initialization value InitValue in the InitValue field of the IPlir message should have a length of 64 bits and be unique for each IPlir packet the encryption and end-to-end MAC of which are implemented by the same source host using the same exchange key.

The transit initialization value TransitInitValue in the TransitInitValue field of the IPlir message should have a length of 64 bits and be unique for each IPlir packet the transit MAC of which is implemented by the same (transit) host using the same exchange key.

5.3.2.3. Key derivation algorithms

The packet encryption key K_ENC of 256 bit length and the packet end-to-end MAC key K_MAC of 256 bit length are calculated as follows:

K_ENC = K_1 || K_2,

K_MAC = K_3 || K_4,

where each value of K_i \in V_128, i = 1,2,3,4 is calculated as per GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CMAC mode as per ISO/IEC 9797-1:2011 [ISO9797-1], wherein

* the exchange key is used as the key. The exchange key is specified by the source and destination hosts and the KN field value of the IPlir message,

* a binary string as shown below is used as the data: IntToVec_8(i)||Label||aL||IV_KDF||SN||Node||cL||oL, where

  - Label = StrToVec_48(’ENCMAC’),

  - aL = IntToVec_8(LabelByteLength), where LabelByteLength = 6,

  - IV_KDF = InitValue, where InitValue is initialized by the InitValue field value of the IPlir message,

  - SN = SequenceNumber, where SequenceNumber is initialized by the SequenceNumber field value of the IPlir message,

- Node = SourceIdentifier, where SourceIdentifier is initialized by the SourceIdentifier field value of the IPlir message,

- cL = IntToVec_16(ContextByteLength), where ContextByteLength is the sum of byte lengths of the InitValue, SequenceNumber and SourceIdentifier fields of the IPlir message,

- oL = IntToVec_16(OutputBitLength), where OutputBitLength = 512,

* the MAC length is 128 bits.

The packet transit MAC key K_TMAC of 256 bit length is calculated as follows:

K_TMAC = K_1 || K_2,

where each value of K_i \in V_128, i = 1,2 is calculated as per GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CMAC mode as per ISO/IEC 9797-1:2011 [ISO9797-1], wherein

* the exchange key is used as the key. The exchange key is specified by the (transit) hosts the IPlir packet passes through and the TKN field value of the IPlir message,

* a binary string as shown below is used as the data: IntToVec_8(i)||Label||aL||TIV_KDF||SN||Node||cL||oL, where

- Label = StrToVec_48(’TMAC’),

- aL = IntToVec_8(LabelByteLength), where LabelByteLength = 6,

- TIV_KDF = TransitInitValue, where TransitInitValue is initialized by the TransitInitValue field value of the IPlir message,

- SN = SequenceNumber, where SequenceNumber is initialized by the SequenceNumber field value of the IPlir message,

- Node = TransitIdentifier, where TransitIdentifier is initialized by the TransitIdentifier field value of the IPlir message,

- cL = IntToVec_16(ContextByteLength), where ContextByteLength is the sum of byte lengths of the TransitInitValue, SequenceNumber and TransitIdentifier fields of the IPlir message,

- oL = IntToVec_16(OutputBitLength), where OutputBitLength = 256,

* the MAC length is 128 bits.

5.3.2.4. Encryption and MAC algorithms

The IPlir body is encrypted as per the GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CTR mode as per ISO/IEC 10116:2017 [ISO10116] without padding, wherein

* the packet encryption key K_ENC is used as the key,

* data in the IPlir body fields in the order of their appearance in the IPlir message are used as plaintext,

* the value of SV \in V_128 is used as the initialization value:

  SV = InitValue||0^64,

  where InitValue is initialized by the InitValue field value of the IPlir message,

* the key stream block length is 128 bits.

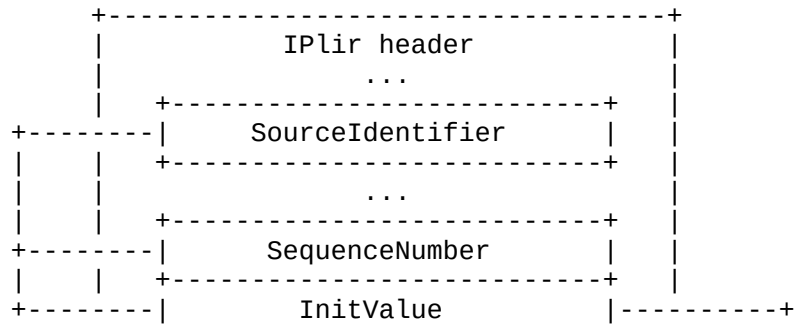
Calculation of the end-to-end MAC ICV in the IntegrityCheckValue field of the IPlir message is implemented as per the GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CMAC mode as per ISO/IEC 9797-1:2011 [ISO9797-1], wherein

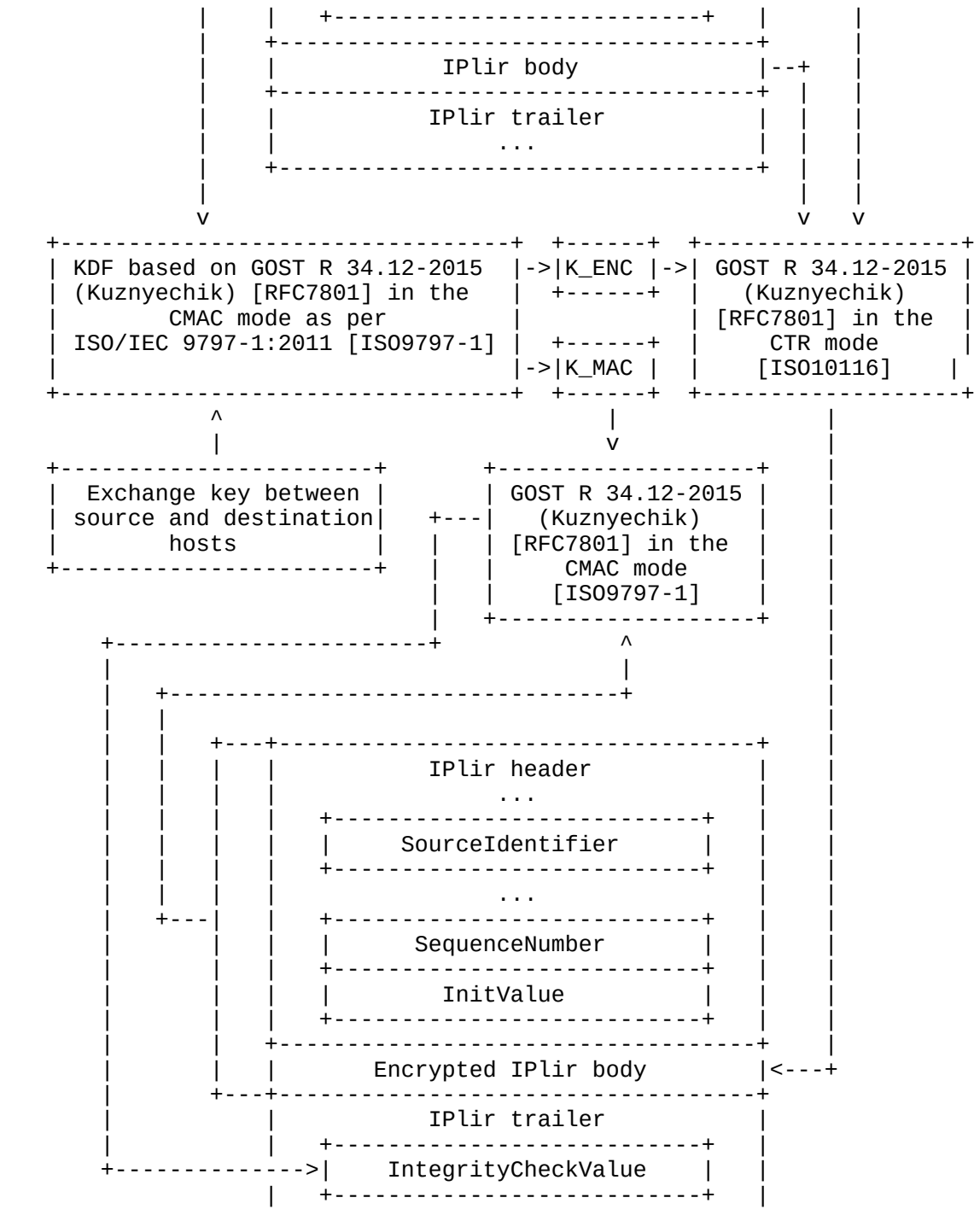
* the packet end-to-end MAC key K_MAC is used as the key,

* data in the IPlir header fields and the encrypted IPlir body in the order of their appearance in the IPlir message are used as the data,

* the MAC length is 64 bits.

The diagram of encryption and end-to-end MAC is shown in Figure 10.




IPlir body
IPlir trailer
...
KDF based on GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CMAC mode as per ISO/IEC 9797-1:2011 [ISO9797-1]
K_ENC
K_MAC
GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CTR mode [ISO10116]
Exchange key between source and destination hosts
GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CMAC mode [ISO9797-1]
IPlir header
...
SourceIdentifier
...
SequenceNumber
InitValue
Encrypted IPlir body
IPlir trailer
IntegrityCheckValue

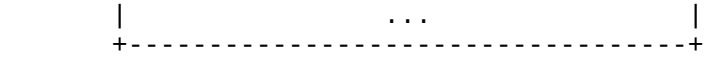


Figure 10: Diagram of Encryption and End-to-End MAC Using the KUZN-CTR-CMAC Cryptographic Suite

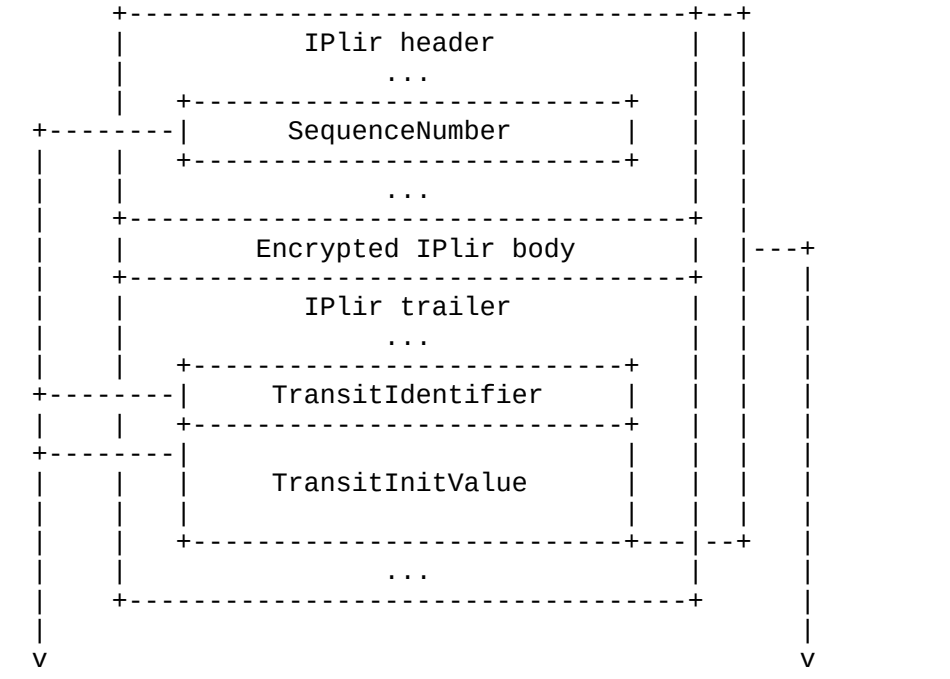
Calculation of the transit MAC TICV in the TransitIntegrityCheckValue field of the IPlir message is implemented as per the GOST R 34.12-2015 (Kuznyechik) [RFC7801] in the CMAC mode as per ISO/IEC 9797-1:2011 [ISO9797-1], wherein

* the packet transit MAC key K_TMAC is used as the key,

* data in the IPlir header fields, the encrypted IPlir body and IntegrityCheckValue, TransitIdentifier, TransitInitValue fields data in the order of their appearance in the IPlir message are used as the data protected by MAC,

* the MAC length is 64 bits.

The diagram of transit MAC is shown in Figure 11.



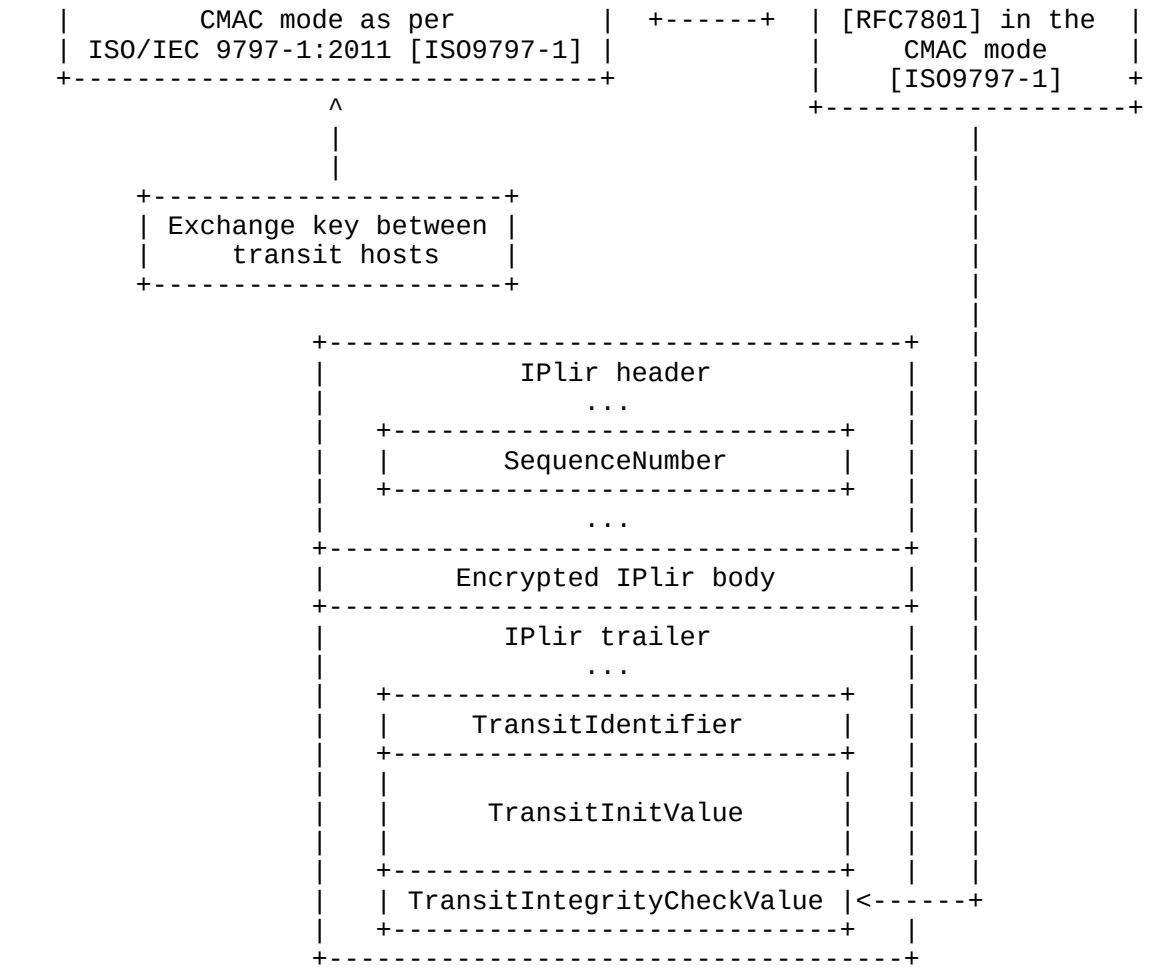


Figure 11: Diagram of Transit MAC Using the KUZN-CTR-CMAC Cryptographic Suite

5.3.3. AES-128-GCM cryptographic suite: CS = 129

AES-128-GCM Cryptographic Suite Description is shown in Table 7:

Parameter	Value
EncAlg	AES-128 [ISO18033-3] in the GCM mode [ISO19772]
MACAlg	AES-128 [ISO18033-3] in the GCM mode [ISO19772]
TMACAlg	AES-128 [ISO18033-3] in the GCM mode [ISO19772]
MACLen	64 bits
TMACLen	64 bits
IVLen	96 bits
TIVLen	96 bits
KDAlg	see the description below

Table 7: AES-128-GCM cryptographic suite: CS = 129

5.3.3.1. Exchange keys

For each pair of interacting hosts, there is a single exchange key with a length of 128 bits used for deriving of packet encryption and end-to-end MAC keys, as well as packet transit MAC keys.

5.3.3.2. Requirements for initialization values

The end-to-end initialization value InitValue in the InitValue field of the IPlir message should have a length of 96 bits and be unique for each IPlir packet the encryption and end-to-end MAC of which are implemented by the same source host using the same exchange key.

The transit initialization value TransitInitValue in the TransitInitValue field of the IPlir message should have a length of 96 bits and be unique for each IPlir packet the transit MAC of which is implemented by the same (transit) host using the same exchange key.

5.3.3.3. Key derivation algorithms

The packet encryption and end-to-end MAC key K_AEAD of 128 bit length is calculated as follows:

K_AEAD = K_1,

where value of K_1 \in V_128 is calculated as per KDF in Counter Mode using AES-128 [ISO18033-3] in the CMAC mode [RFC4493] as the PRF, wherein

* the exchange key is used as the key for the PRF. The exchange key is specified by the source and destination hosts and the KN field value of the IPlir message,

* a binary string as shown below is used as the input data for the PRF: IntToVec_8(1)||Label||aL||IV_KDF||SN||Node||cL||oL, where

  - Label = StrToVec_48(’AEAD’),

  - aL = IntToVec_8(LabelByteLength), where LabelByteLength = 6,

  - IV_KDF = InitValue, where InitValue is initialized by the InitValue field value of the IPlir message,

  - SN = SequenceNumber, where SequenceNumber is initialized by the SequenceNumber field value of the IPlir message,

  - Node = SourceIdentifier, where SourceIdentifier is initialized by the SourceIdentifier field value of the IPlir message,

  - cL = IntToVec_16(ContextByteLength), where ContextByteLength is the sum of byte lengths of the InitValue, SequenceNumber and SourceIdentifier fields of the IPlir message,

  - oL = IntToVec_8(OutputBitLength), where OutputBitLength = 128,

* the PRF output length is 128 bits.

The packet transit MAC key K_TMAC of 128 bit length is calculated as follows:

K_TMAC = K_1,

where value of K_1 \in V_128 is calculated as per KDF in Counter Mode using AES-128 [ISO18033-3] in the CMAC mode [RFC4493] as the PRF, wherein

* the exchange key is used as the key for the PRF. The exchange key is specified by the (transit) hosts the IPlir packet passes through and the TKN field value of the IPlir message,

* a binary string as shown below is used as the input data for the PRF: IntToVec_8(1)||Label||aL||TIV_KDF||SN||Node||cL||oL, where

- Label = StrToVec_48('TMAC'),

- aL = IntToVec_8(LabelByteLength), where LabelByteLength = 6,

- TIV_KDF = TransitInitValue, where TransitInitValue is initialized by the TransitInitValue field value of the IPlir message,

- SN = SequenceNumber, where SequenceNumber is initialized by the SequenceNumber field value of the IPlir message,

- Node = TransitIdentifier, where TransitIdentifier is initialized by the TransitIdentifier field value of the IPlir message,

- cL = IntToVec_16(ContextByteLength), where ContextByteLength is the sum of byte lengths of the TransitInitValue, SequenceNumber and TransitIdentifier fields of the IPlir message,

- oL = IntToVec_8(OutputBitLength), where OutputBitLength = 128,

* the PRF output length is 128 bits.

5.3.3.4. Encryption and MAC algorithms

Encryption of the IPlir body and calculation of the end-to-end MAC ICV in the IntegrityCheckValue field of the IPlir message are implemented as per the AES-128 [ISO18033-3] in the GCM mode [ISO19772], wherein

* the packet encryption and end-to-end MAC key K_AEAD is used as the key,

* data in the IPlir header fields in the order of their appearance in the IPlir message are used as additional authenticated data,

* data in the IPlir body fields in the order of their appearance in the IPlir message are used as plaintext,

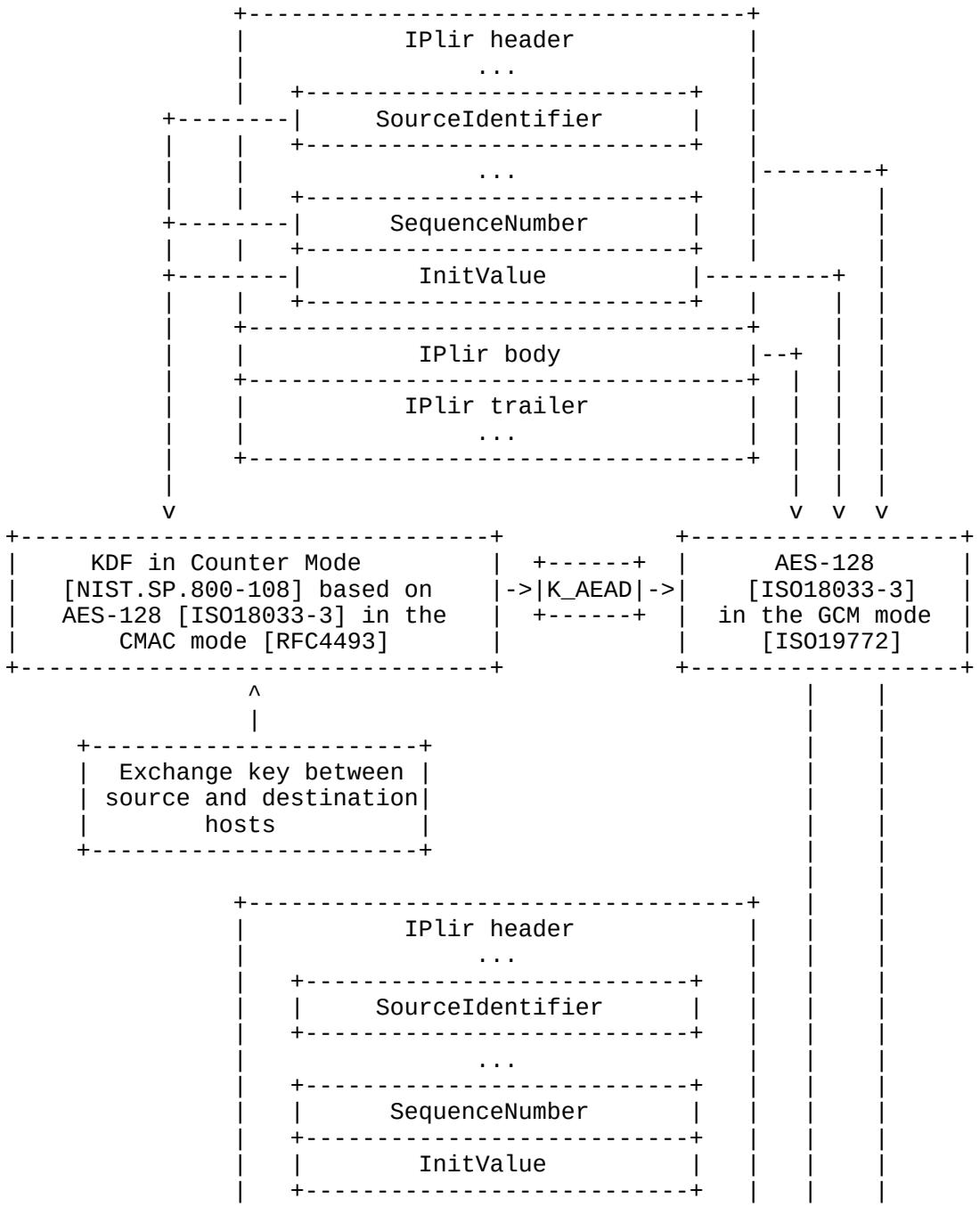
* the value of IV_AEAD \in V_96 is used as initialization vector:

  IV_AEAD = InitValue,

  where InitValue is initialized by the InitValue field value of the IPlir message,

* the MAC length is 64 bits.

The diagram of encryption and end-to-end MAC is shown in Figure 12.



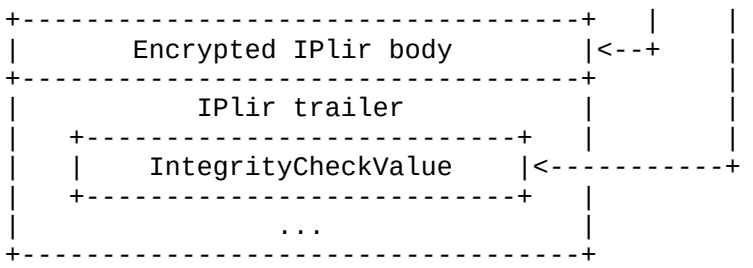


Figure 12: Diagram of Encryption and End-to-End MAC Using the AES-128-GCM Cryptographic Suite

Calculation of the transit MAC TICV in the TransitIntegrityCheckValue field of the IPlir message is implemented as per the AES-128 [ISO18033-3] in the GCM mode [ISO19772], wherein

* the packet transit MAC key K_TMAC is used as the key,

* data in the IPlir header fields, the encrypted IPlir body and IntegrityCheckValue, TransitIdentifier, TransitInitValue fields data in the order of their appearance in the IPlir message are used as additional authenticated data,

* plaintext is an empty string,

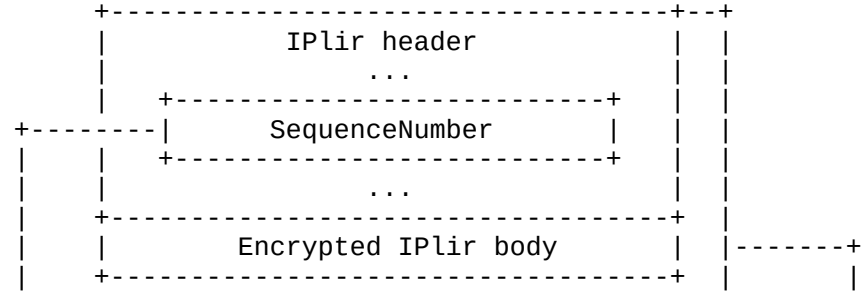
* the value of TIV_AEAD \in V_96 is used as initialization vector:

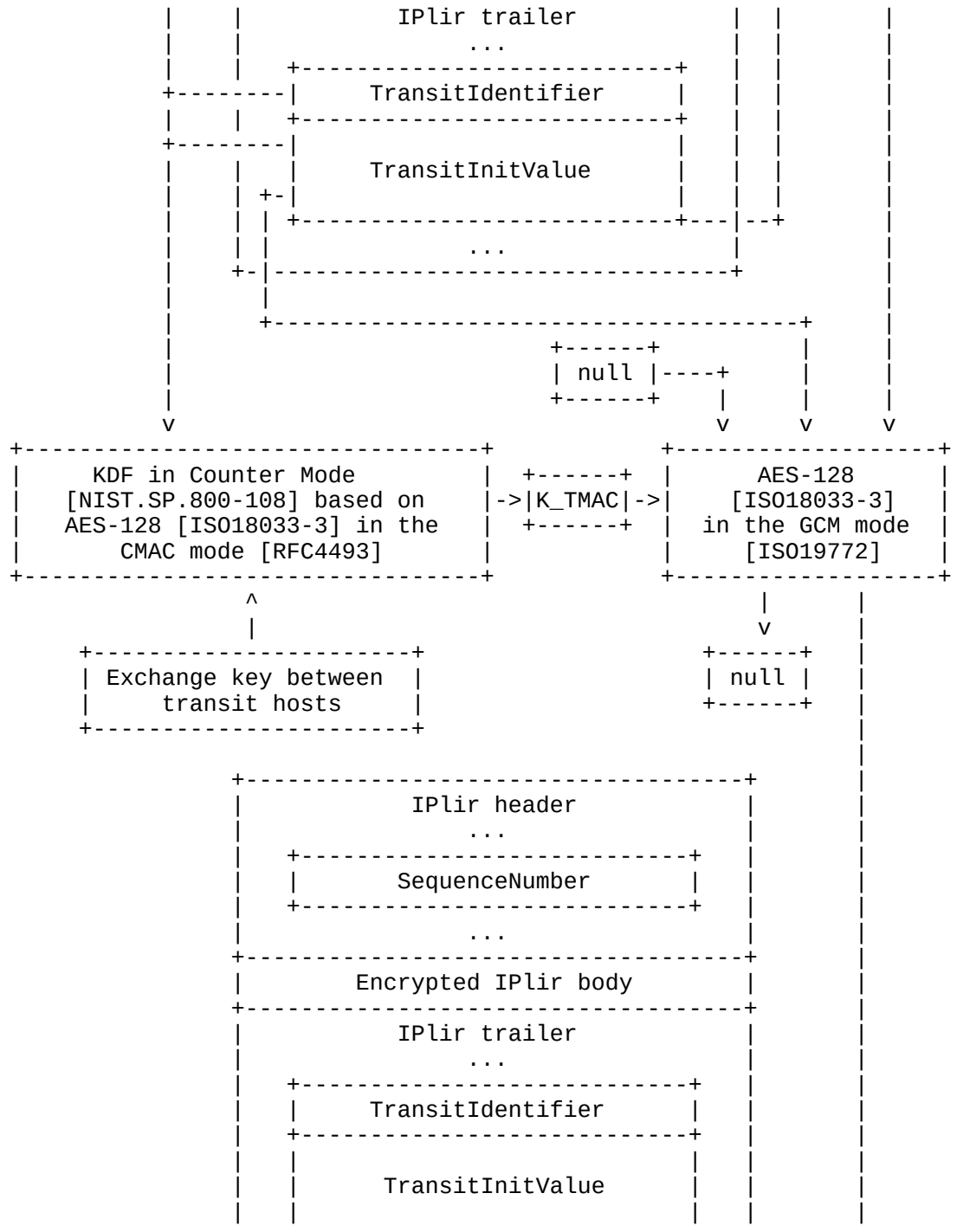
  TIV_AEAD = TransitInitValue,

  where TransitInitValue is initialized by the TransitInitValue field value of the IPlir message,

* the MAC length is 64 bits.

The diagram of transit MAC is shown in Figure 13. The “null” value means an empty binary string.




IPlir trailer
...
TransitIdentifier
TransitInitValue
...
null
KDF in Counter Mode
[NIST.SP.800-108] based on
AES-128 [ISO18033-3] in the
CMAC mode [RFC4493]
K_TMAC
AES-128
[ISO18033-3]
in the GCM mode
[ISO19772]
Exchange key between
transit hosts
null
IPlir header
...
SequenceNumber
...
Encrypted IPlir body
IPlir trailer
...
TransitIdentifier
TransitInitValue

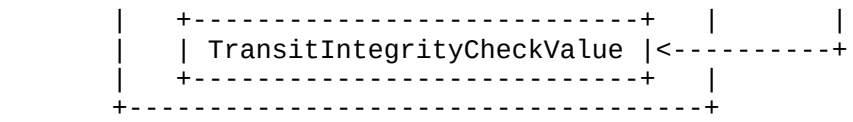


Figure 13: Diagram of Transit MAC Using the AES-128-GCM Cryptographic Suite

5.3.4. AES-256-CTR-CMAC cryptographic suite: CS = 132

AES-256-CTR-CMAC Cryptographic Suite Description is shown in Table 8:

Parameter	Value
EncAlg	AES-256 [ISO18033-3] in the CTR mode [ISO10116]
MACAlg	AES-256 [ISO18033-3] in the CMAC mode [RFC4493]
TMACAlg	AES-256 [ISO18033-3] in the CMAC mode [RFC4493]
MACLen	64 bits
TMACLen	64 bits
IVLen	64 bits
TIVLen	64 bits
KDAlg	see the description below

Table 8: AES-256-CTR-CMAC cryptographic suite: CS = 132

5.3.4.1. Exchange keys

For each pair of interacting hosts, there is a single exchange key with a length of 256 bits designed for derivation of packet encryption keys, end-to-end MAC keys, and transit MAC keys.

5.3.4.2. Requirements for initialization values

The end-to-end initialization value InitValue in the InitValue field of the IPlir message should have a length of 64 bits and be unique for each IPlir packet the encryption and end-to-end MAC of which are implemented by the same source host using the same exchange key.

The transit initialization value TransitInitValue in the TransitInitValue field of the IPlir message should have a length of 64 bits and be unique for each IPlir packet the transit MAC of which is implemented by the same (transit) host using the same exchange key.

5.3.4.3. Key derivation algorithms

The packet encryption key K_ENC of 256 bit length and the packet end-to-end MAC key K_MAC of 256 bit length are calculated as follows:

K_ENC = K_1 || K_2,

K_MAC = K_3 || K_4,

where each value of K_i \in V_128, i = 1,2,3,4 is calculated as per KDF in Counter Mode using AES-256 [ISO18033-3] in the CMAC mode [RFC4493] as the PRF, wherein

* the exchange key is used as the key for the PRF. The exchange key is specified by the source and destination hosts and the KN field value of the IPlir message,

* a binary string as shown below is used as the input data for the PRF: IntToVec_8(i)||Label||aL||IV_KDF||SN||Node||cL||oL, where

  - Label = StrToVec_48(’ENCMAC’),

  - aL = IntToVec_8(LabelByteLength), where LabelByteLength = 6,

  - IV_KDF = InitValue, where InitValue is initialized by the InitValue field value of the IPlir message,

  - SN = SequenceNumber, where SequenceNumber is initialized by the SequenceNumber field value of the IPlir message,

  - Node = SourceIdentifier, where SourceIdentifier is initialized by the SourceIdentifier field value of the IPlir message,

  - cL = IntToVec_16(ContextByteLength), where ContextByteLength is the sum of byte lengths of the InitValue, SequenceNumber and SourceIdentifier fields of the IPlir message,

  - oL = IntToVec_16(OutputBitLength), where OutputBitLength = 512,

* the PRF output length is 128 bits.

The packet transit MAC key K_TMAC of 256 bit length is calculated as follows:

K_TMAC = K_1 || K_2,

where each value of K_i \in V_128, i = 1,2 is calculated as per KDF in Counter Mode using AES-256 [ISO18033-3] in the CMAC mode [RFC4493] as the PRF, wherein

* the exchange key is used as the key for the PRF. The exchange key is specified by the (transit) hosts the IPlir packet passes through and the TKN field value of the IPlir message,

* a binary string as shown below is used as the input data for the PRF: IntToVec_8(i)||Label||aL||TIV_KDF||SN||Node||cL||oL, where

  - Label = StrToVec_48('TMAC'),

  - aL = IntToVec_8(LabelByteLength), where LabelByteLength = 6,

  - TIV_KDF = TransitInitValue, where TransitInitValue is initialized by the TransitInitValue field value of the IPlir message,

  - SN = SequenceNumber, where SequenceNumber is initialized by the SequenceNumber field value of the IPlir message,

  - Node = TransitIdentifier, where TransitIdentifier is initialized by the TransitIdentifier field value of the IPlir message,

  - cL = IntToVec_16(ContextByteLength), where ContextByteLength is the sum of byte lengths of the TransitInitValue, SequenceNumber and TransitIdentifier fields of the IPlir message,

  - oL = IntToVec_16(OutputBitLength), where OutputBitLength = 256,

* the PRF output length is 128 bits.

5.3.4.4. Encryption and MAC algorithms

The IPlir body is encrypted as per the AES-256 [ISO18033-3] in the CTR mode as per ISO/IEC 10116:2017[ISO10116] without padding, wherein

* the packet encryption key K_ENC is used as the key,

* data in the IPlir header fields and the encrypted IPlir body in the order of their appearance in the IPlir message are used as the data,

* the values of CTR_i \in V_128 are used as counters in CTR mode:

  CTR_i = InitValue || IntToVec_64(i), i = 1, 2, ... , n,

  where InitValue is initialized by the InitValue field value of the IPlir message.

* the key stream block length is 128 bits.

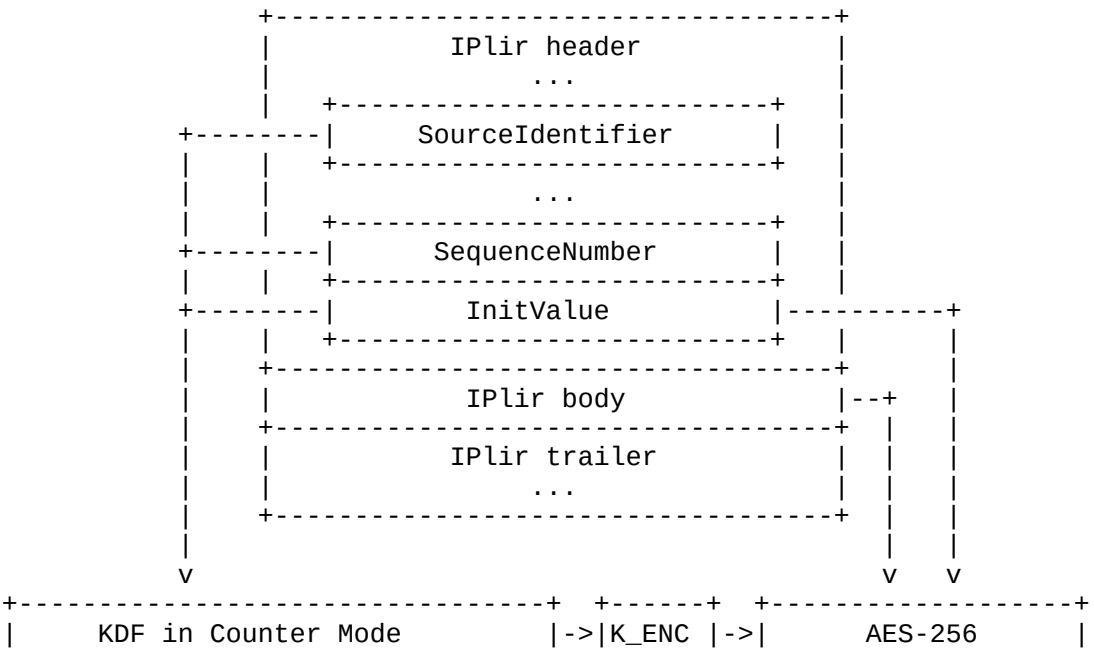
Calculation of the end-to-end MAC ICV in the IntegrityCheckValue field of the IPlir message is implemented as per the AES-256 [ISO18033-3] in the CMAC mode [RFC4493], wherein

* the packet end-to-end MAC key K_MAC is used as the key,

* data in the IPlir body fields in the order of their appearance in the IPlir message are used as plaintext,

* the MAC length is 64 bits.

The diagram of encryption and end-to-end MAC is shown in Figure 14.



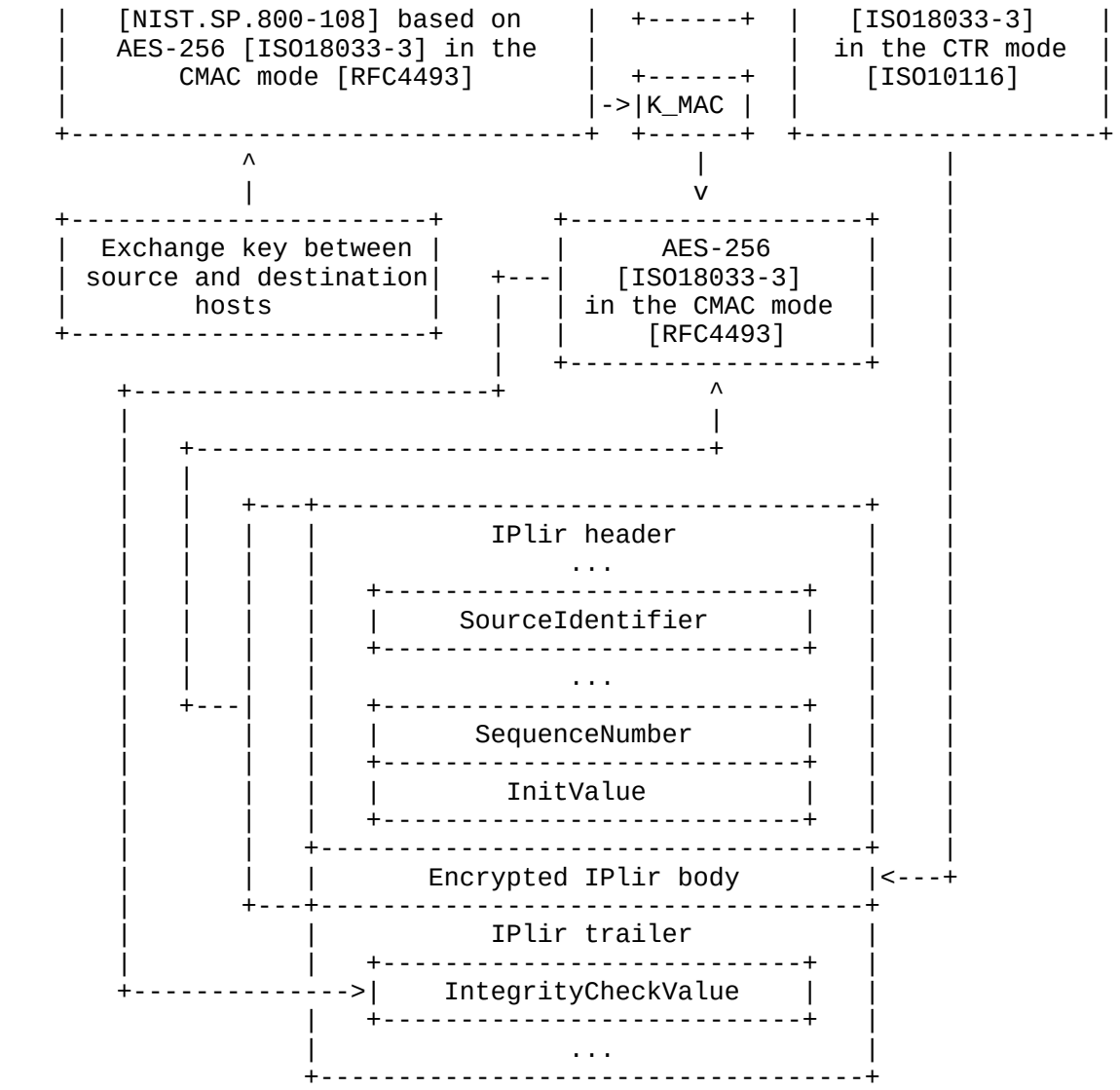


Figure 14: Diagram of Encryption and End-to-End MAC Using the AES-256-CTR-CMAC Cryptographic Suite

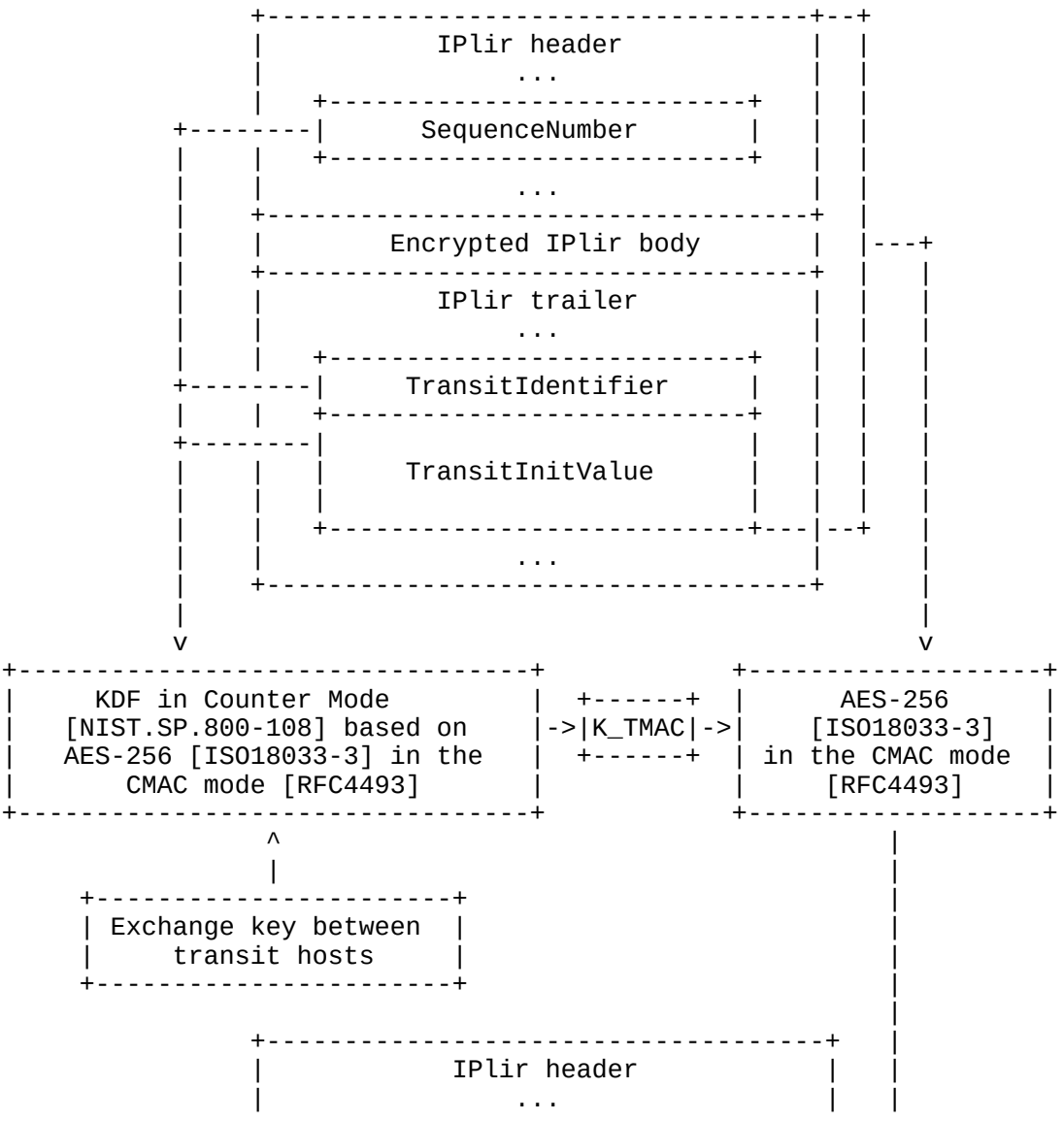
Calculation of the transit MAC TICV in the TransitIntegrityCheckValue field of the IPlir message is implemented as per the AES-256 [ISO18033-3] in the CMAC mode [RFC4493], wherein

* the packet transit MAC key K_TMAC is used as the key,

* data in the IPlir header fields, the encrypted IPlir body and IntegrityCheckValue, TransitIdentifier, TransitInitValue fields data in the order of their appearance in the IPlir message are used as the data protected by MAC,

* the MAC length is 64 bits.

The diagram of transit MAC is shown in Figure 15.



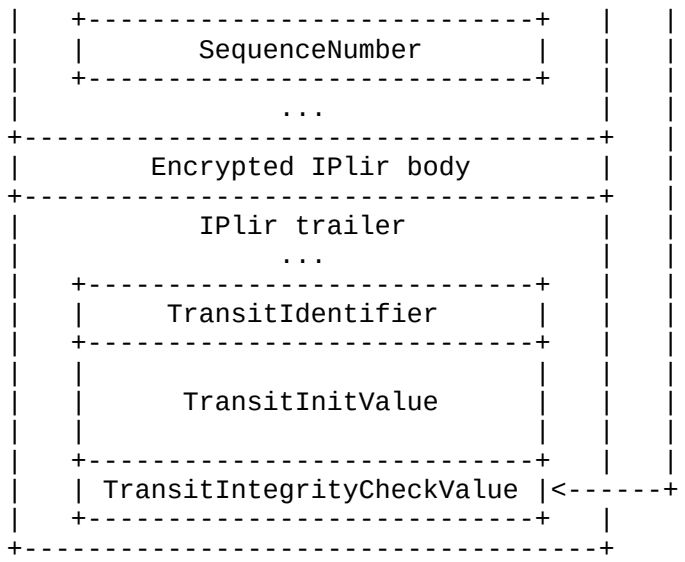


Figure 15: Diagram of Transit MAC Using the AES-256-CTR-CMAC Cryptographic Suite

5.3.5. AES-256-CFB-CMAC cryptographic suite: CS = 134

AES-256-CTR-CMAC Cryptographic Suite Description is shown in Table 9:

Parameter	Value
EncAlg	AES-256 [ISO18033-3] in the CFB mode [ISO10116]
MACAlg	AES-256 [ISO18033-3] in the CMAC mode [RFC4493]
TMACAlg	AES-256 [ISO18033-3] in the CMAC mode [RFC4493]
MACLen	64 bits
TMACLen	64 bits
IVLen	128 bits
TIVLen	64 bits
KDAlg	see the description below

Table 9: AES-256-CFB-CMAC cryptographic suite: CS = 134

5.3.5.1. Exchange keys

For each pair of interacting hosts, there is a single exchange key with a length of 256 bits designed for derivation of packet encryption keys, end-to-end MAC keys, and transit MAC keys.

5.3.5.2. Requirements for initialization values

The end-to-end initialization value InitValue in the InitValue field of the IPlir message should have a length of 128 bits and be random.

The transit initialization value TransitInitValue in the TransitInitValue field of the IPlir message should have a length of 64 bits and be unique for each IPlir packet the transit MAC of which is implemented by the same (transit) host using the same exchange key.

5.3.5.3. Key derivation algorithms

The packet encryption key K_ENC of 256 bit length and the packet end-to-end MAC key K_MAC of 256 bit length are calculated as follows:

K_ENC = K_1 || K_2,

K_MAC = K_3 || K_4,

where each value of K_i \in V_128, i = 1,2,3,4 is calculated as per KDF in Counter Mode using AES-256 [ISO18033-3] in the CMAC mode [RFC4493] as the PRF, wherein

* the exchange key is used as the key for the PRF. The exchange key is specified by the source and destination hosts and the KN field value of the IPlir message,

* a binary string as shown below is used as the input data for the PRF: IntToVec_8(i)||Label||aL||IV_KDF||SN||Node||cL||oL, where

  - Label = StrToVec_48(’ENCMAC’),

  - aL = IntToVec_8(LabelByteLength), where LabelByteLength = 6,

  - IV_KDF = InitValue, where InitValue is initialized by the InitValue field value of the IPlir message,

  - SN = SequenceNumber, where SequenceNumber is initialized by the SequenceNumber field value of the IPlir message,

- Node = SourceIdentifier, where SourceIdentifier is initialized by the SourceIdentifier field value of the IPlir message,

- cL = IntToVec_16(ContextByteLength), where ContextByteLength is the sum of byte lengths of the InitValue, SequenceNumber and SourceIdentifier fields of the IPlir message,

- oL = IntToVec_16(OutputBitLength), where OutputBitLength = 512,

* the PRF output length is 128 bits.

The packet transit MAC key K_TMAC of 256 bit length is calculated as follows:

K_TMAC = K_1 || K_2,

where each value of K_i \in V_128, i = 1,2 is calculated as per KDF in Counter Mode using AES-256 [ISO18033-3] in the CMAC mode [RFC4493] as the PRF, wherein

* the exchange key is used as the key for the PRF. The exchange key is specified by the (transit) hosts the IPlir packet passes through and the TKN field value of the IPlir message,

* a binary string as shown below is used as the input data for the PRF: IntToVec_8(i)||Label||aL||TIV_KDF||SN||Node||cL||oL, where

  - Label = StrToVec_48('TMAC'),

  - aL = IntToVec_8(LabelByteLength), where LabelByteLength = 6,

  - TIV_KDF = TransitInitValue, where TransitInitValue is initialized by the TransitInitValue field value of the IPlir message,

  - SN = SequenceNumber, where SequenceNumber is initialized by the SequenceNumber field value of the IPlir message,

  - Node = TransitIdentifier, where TransitIdentifier is initialized by the TransitIdentifier field value of the IPlir message,

  - cL = IntToVec_16(ContextByteLength), where ContextByteLength is the sum of byte lengths of the TransitInitValue, SequenceNumber and TransitIdentifier fields of the IPlir message,

  - oL = IntToVec_16(OutputBitLength), where OutputBitLength = 256,

* the PRF output length is 128 bits.

5.3.5.4. Encryption and MAC algorithms

The IPlir body is encrypted as per the AES-256 [ISO18033-3] in the CFB mode as per ISO/IEC 10116:2017 [ISO10116], wherein

* the packet encryption key K_ENC is used as the key,

* data in the IPlir body fields in the order of their appearance in the IPlir message are used as plaintext,

* the value of SV \in V_128 is used as the initialization value:

  SV = InitValue,

  where InitValue is initialized by the InitValue field value of the IPlir message.

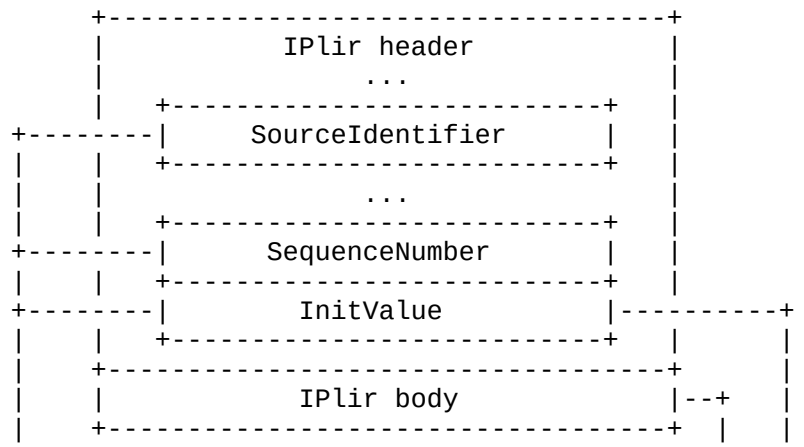
Calculation of the end-to-end MAC ICV in the IntegrityCheckValue field of the IPlir message is implemented as per the AES-256 [ISO18033-3] in the CMAC mode [RFC4493], wherein

* the packet end-to-end MAC key K_MAC is used as the key,

* data in the IPlir header fields and the encrypted IPlir body in the order of their appearance in the IPlir message are used as the data,

* the MAC length is 64 bits.

The diagram of encryption and end-to-end MAC is shown in Figure 16.



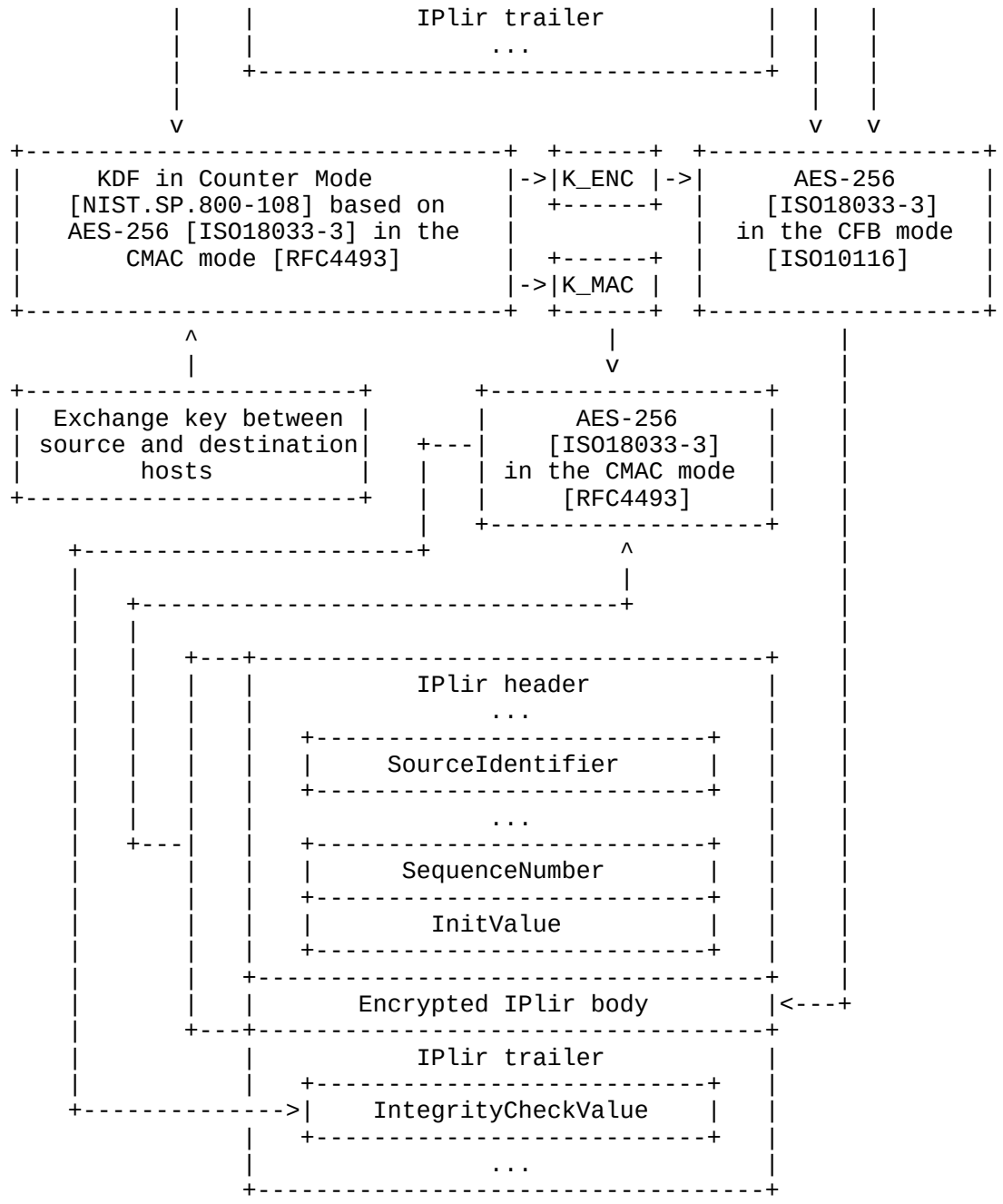


Figure 16: Diagram of Encryption and End-to-End MAC Using the AES-256-CFB-CMAC Cryptographic Suite

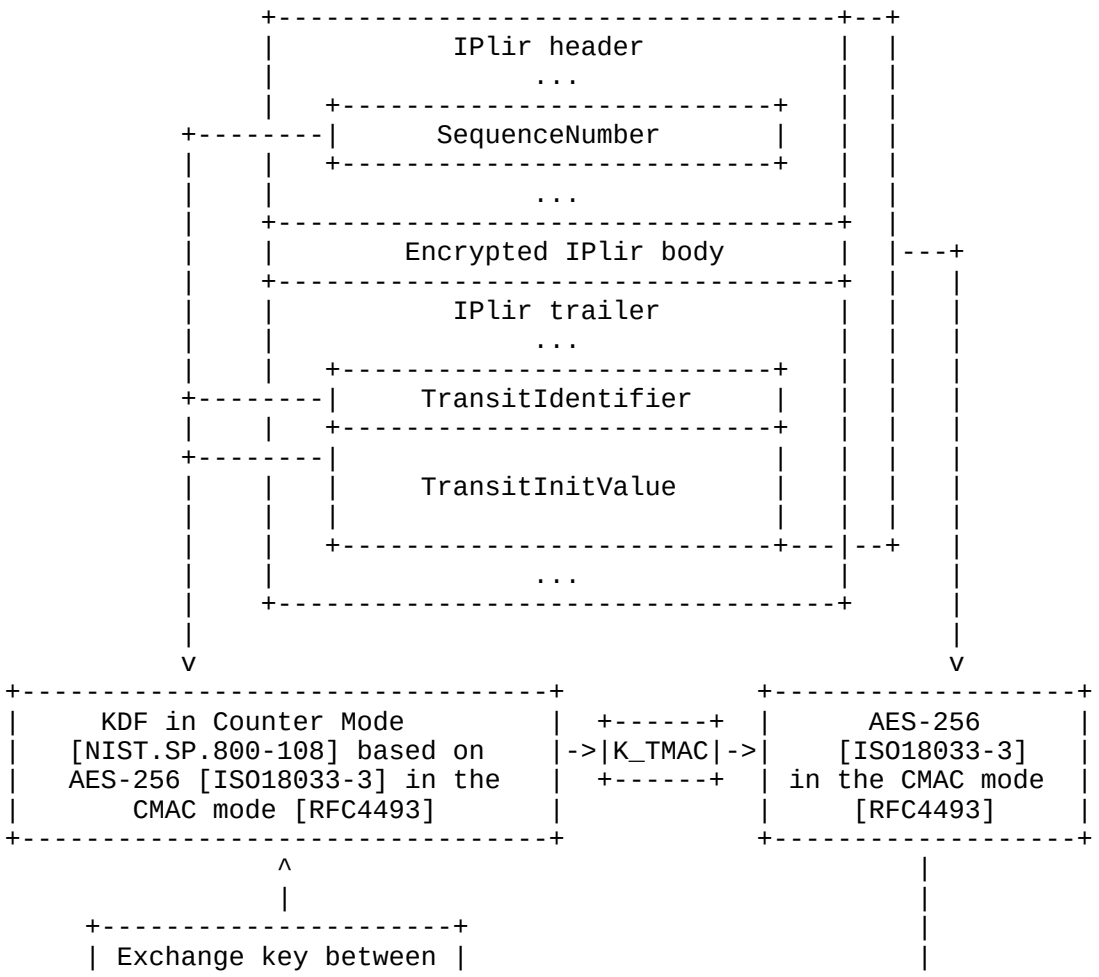
Calculation of the transit MAC TICV in the TransitIntegrityCheckValue field of the IPlir message is implemented as per the AES-256 [ISO18033-3] in the CMAC mode [RFC4493], wherein

* the packet transit MAC key K_TMAC is used as the key,

* data in the IPlir header fields, the encrypted IPlir body and IntegrityCheckValue, TransitIdentifier, TransitInitValue fields data in the order of their appearance in the IPlir message are used as the data protected by MAC,

* the MAC length is 64 bits.

The diagram of transit MAC is shown in Figure 17.



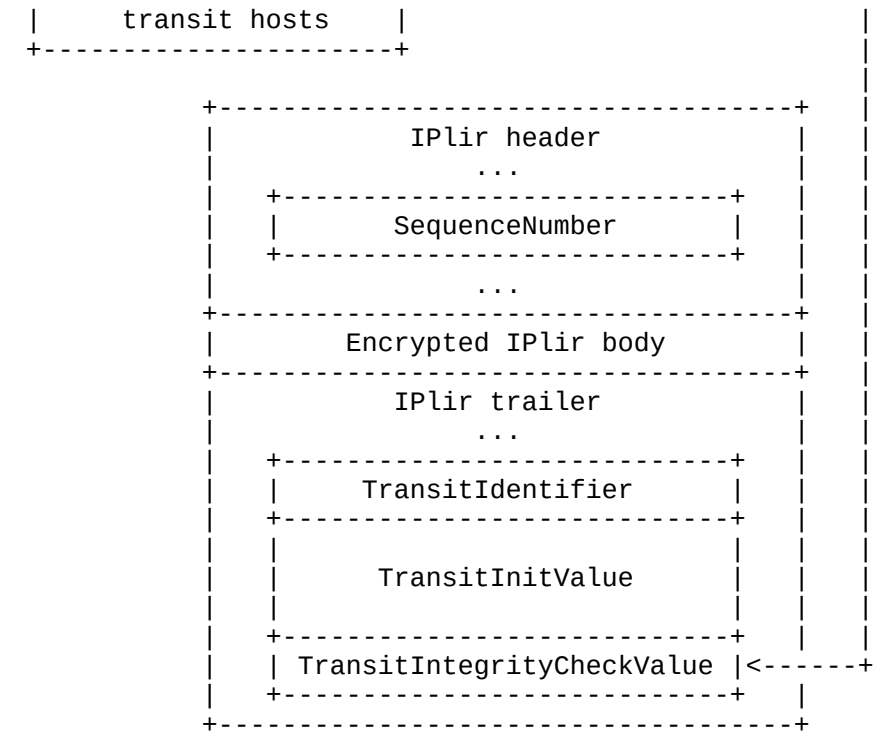


Figure 17: Diagram of Transit MAC Using the AES-256-CFB-CMAC Cryptographic Suite

6. IPlir packet processing

The algorithms used for cryptographic processing of network packets are determined by the cryptographic suite.

The cryptographic suite for protection of the original IP packet is chosen depending on the corresponding security policy of the source host and the destination host context on the source host. The logic and procedure of processing IPlir packets protected using a certain cryptographic suite depend on the IPlir packet reception policy and the source host context on the destination host. The necessity and procedure of using transit MAC are determined based on the source host security policy and IPlir packet reception policies of transit hosts and the destination host.

Depending on security policies and other requirements, protection of the destination host or transit host against replay of previously transmitted IPlir packets may be required. The IPlir protocol makes it possible to arrange such protection by using counter values and/or

timestamps, as well as by tracking the history of their change on transit hosts and the destination host. As an example, SequenceNumber, InitValue, TransitInitValue field values can be used as counter values, Timestamp field values can be used as timestamps. Description of specific mechanisms designed for protection against replay of previously transmitted IPlir packets is beyond the scope of this document.

6.1. IP and IPlir packet fragmentation

When packing data in IP packets, the IP protocol can fragment (break down into fragments) messages of the higher transport layer protocols UDP, TCP, etc. This results in several (linked) IP packets, each called an IP fragment.

The IPlir protocol in transport and light tunnel modes should only be applied to whole (non-fragmented) IP packets, but not IP fragments. In the tunnel mode, the IPlir protocol can be applied to both whole IP packets and IP fragments.

In case of encapsulation in IPv4, the IPlir packet, just like any other IPv4 packet, can be fragmented by routers during transmission. Before the IPlir packet is processed on the end of the destination or transit host, the IPlir packet must be defragmented.

6.2. Original IP packet protection by the source host

If the source host decides to protect a specific IP packet, an IPlir packet is created as follows:

1. The destination and transit hosts contexts along with the applied security policy determine:
    * IPlir operation mode;
    * cryptographic suite;
    * transit MAC necessity;
2. Based on the destination and transit hosts contexts along with the cryptographic suite:
    * an end-to-end initialization value and, if transit MAC is needed, a transit initialization value are generated;
    * the packet number and timestamp are generated;

* the packet encryption key, end-to-end MAC key and, if necessary, transit MAC key are derived;

3. The IPlir packet fields are filled in considering the data from the original IP packet and the data generated previously.

4. The IPlir body is encrypted and the end-to-end MAC value is calculated as prescribed by the cryptographic suite. The end-to-end MAC value is placed in the IntegrityCheckValue field of the IPlir trailer.

5. If transit integrity control is required, the corresponding fields and flags of the IPlir header and IPlir trailer are filled in, the transit MAC value is calculated. The transit MAC value is placed in the TransitIntegrityCheckValue field of the IPlir trailer.

6. An IPlir packet is generated, wherein parts of the original IP packet are located according to the rules specified in Section 4.4.

6.3. IPlir packet processing on the transit host

After receiving an IPlir packet, the transit host processes the IPlir packet as follows:

1. It checks whether the received IPlir packet corresponds to the IPlir packet reception policy. If the packet does not comply with the policy, it is blocked.

2. If the IPlir version in the IPlir header is not supported by the transit host, the packet is blocked.

3. The IPlir packet is matched to the context of the source host or the previous transit host. If the context is not found or contains cryptographic suites not matching the set from the IPlir header, the packet is blocked.

4. If the suite from the IPlir header does not imply transit MAC, the packet is blocked.

5. Based on the context of the previous transit host (or source host) and the IPlir header, the packet transit MAC key is derived. The IPlir packet integrity is verified by checking the transit MAC. If the transit MAC is not correct, the packet is blocked.

6. The next transit host is determined based on the destination host context on the transit host (or it is determined that the IPlir packet can be delivered to the destination host directly). If the context of the next transit host (or destination host) is not found, the packet is blocked.

7. If the found context contains cryptographic suites not matching the set from the IPlir header, the packet is blocked.

8. The transit initialization value is generated and the packet transit MAC key is derived based on the context of the next transit host, IPlir header and IPlir trailer. The number of the packet transit MAC key and the transit host identifier are established in the IPlir message. The transit initialization value is located in the TransitInitValue field.

9. The transit MAC value is calculated and placed in the TransitIntegrityCheckValue field of the IPlir trailer.

10. An IPlir packet is generated, wherein parts of the original IP packet are located according to the rules specified in Section 4.4.

There may be cases when the security policies require a transit MAC to be added to the routed packet without checking the previous value or, vice versa, the received IPlir packet integrity to be checked without calculating a new transit MAC value, as well as cases when no transit protection is required. In this case:

* If no integrity verification of the received transit IPlir packet is required, steps 3, 4, 5 of the above algorithm are skipped,

* If no transit MAC calculation is required, steps 7, 8, 9 of the above algorithm are skipped,

* If transit protection is not required, steps 3, 4, 5, 7, 8, 9 of the above algorithm are skipped.

6.4. Original IP packet recovery by the destination host

After receiving the IPlir packet, the destination host recovers the original IP packet as follows:

1. It checks whether the received IPlir packet corresponds to the IPlir packet reception policy. If the packet does not comply with the policy, it is blocked.

2. If the IPlir version in the IPlir header is not supported by the destination host, the packet is blocked.

3. The IPlir packet is matched to the context of the previous transit host (or source host). If the context is not found or contains cryptographic suites not matching the set from the IPlir header, the packet is blocked.

4. Based on the context of the previous transit host (or source host) and the IPlir header, the packet transit MAC key is derived. The IPlir packet integrity is verified by checking the transit MAC. If the transit MAC is not correct, the packet is blocked.

5. The IPlir packet is matched to the context of the source host. If the context is not found or contains cryptographic suites not matching the suite from the IPlir header, the packet is blocked.

6. Based on the context of the source host and the IPlir header, the packet end-to-end MAC key and, if necessary optional packet encryption key are derived.

7. The end-to-end MAC is checked and, if the IPlir body was encrypted, the packet IPlir body is decrypted as defined by the cryptographic suite. If the end-to-end MAC is not correct, the packet is blocked.

8. The IP packet is restored according to the rules of Section 4.4.

There may be cases when the security policies do not require transit MAC checking by the destination host. Then steps 3, 4 of the algorithm are skipped.

7. IANA Considerations

This document has no IANA actions.

8. Normative References

[ISO10116] International Organization for Standartization, "Information technology - Security techniques - Modes of operation for an n-bit block cipher", ISO/IEC 10116:2017, 2017.

[ISO18033-3]
International Organization for Standartization, "Information technology - Security techniques - Encryption algorithms - Part 3: Block ciphers", ISO/IEC 18033-3:2020, 2020.

[ISO19772] International Organization for Standartization, "Information technology - Authenticated encryption", ISO/IEC 19772:2020, 2020.

[ISO9797-1]
International Organization for Standartization, "Information technology - Security techniques - Message Authentication Codes (MACs) - Part 1: Mechanisms using a block cipher", ISO/IEC 9797-1:2011, 2011.

[NIST_SP_800_108]
Chen, L. and NIST, "Recommendation for key derivation using pseudorandom functions (revised)", NIST Special Publications (General) 800-108, DOI 10.6028/NIST.SP.800-108, 2009, <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-108.pdf>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[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/info/rfc4493>.

[RFC7801] Dolmatov, V., Ed., "GOST R 34.12-2015: Block Cipher "Kuznyechik"", RFC 7801, DOI 10.17487/RFC7801, March 2016, <https://www.rfc-editor.org/info/rfc7801>.

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

[RFC8891] Dolmatov, V., Ed. and D. Baryshkov, "GOST R 34.12-2015: Block Cipher "Magma"", RFC 8891, DOI 10.17487/RFC8891, September 2020, <https://www.rfc-editor.org/info/rfc8891>.

[RFC9058] Smyshlyaev, S., Ed., Nozdrunov, V., Shishkin, V., and E. Griboedova, "Multilinear Galois Mode (MGM)", RFC 9058, DOI 10.17487/RFC9058, June 2021, <https://www.rfc-editor.org/info/rfc9058>.

Authors' Addresses

Martishina Alexandra (editor)
InfoTeCS
2B stroenie 1, ul. Otradnaya
Moscow
127273
Russian Federation
Phone: +7 (495) 737-61-92
Email: Aleksandra.Martishina@infotecs.ru

Urivskiy Alexey
InfoTeCS
2B stroenie 1, ul. Otradnaya
Moscow
127273
Russian Federation
Phone: +7 (495) 737-61-92
Email: urivskiy@infotecs.ru

Rybkin Andrey
InfoTeCS
2B stroenie 1, ul. Otradnaya
Moscow
127273
Russian Federation
Phone: +7 (495) 737-61-92
Email: Andrey.Rybkin@infotecs.ru

Tychina Leonid
InfoTeCS
2B stroenie 1, ul. Otradnaya
Moscow
127273
Russian Federation
Phone: +7 (495) 737-61-92
Email: tychina@infotecs.ru

Parshin Ilia
InfoTeCS
2B stroenie 1, ul. Otradnaya
Moscow
127273
Russian Federation
Phone: +7 (495) 737-61-92
Email: Parshin.Ilia@infotecs.ru