Document Title	Specification of TCP/IP Stack
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	617

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R23-11

Document Change History

Date	Release	Changed by	Description
2023-11-23	R23-11	AUTOSAR Release Management	• Minor corrections and clarifications • Editorial changes • Wildcards descriptions improvement • Introduction of IND, TCP window scale option, TCP SACK and TLS updates for Charging Interface
2022-11-24	R22-11	AUTOSAR Release Management	• ARP defensive processing • Transition OFFLINE to ONLINE
2021-11-25	R21-11	AUTOSAR Release Management	• Add `TcpIp_IsConnectionReady`() • Minor corrections and clarifications • Editorial changes
2020-11-30	R20-11	AUTOSAR Release Management	• Introduction of IdsM (DRAFT) • Minor corrections and clarifications • Editorial changes
2019-11-28	R19-11	AUTOSAR Release Management	• Introduction of IPsec • Minor corrections and clarifications • Changed Document Status from Final to published

△

2018-10-31	4.4.0	AUTOSAR Release Management	• Introduction of Transport Layer Security - TLS (DRAFT) • ARP timing improvements • minor corrections / clarifications / editorial changes
2017-12-08	4.3.1	AUTOSAR Release Management	• Clarifications and corrections of requirements • Editorial changes
2016-11-30	4.3.0	AUTOSAR Release Management	• Improvements for robustness • Introduction of diagnostic features • Clarifications and corrections of requirements • Editorial changes
2015-07-31	4.2.2	AUTOSAR Release Management	• Support for transmission of fragmented IPv4/IPv6 frames • Clarifications and corrections of requirements • Editorial changes
2014-10-31	4.2.1	AUTOSAR Release Management	• Introduction of IPv6 for in-vehicle communication • Support for Switch Control/Configuration, Semi-Static Auto-Configuration • TcpIp generic upper layer support (CDD) • Clarifications and corrections of requirements and sequence charts
2014-03-31	4.1.3	AUTOSAR Release Management	• Clarifications and corrections of requirements • Editorial changes

▽

△

2013-10-31	4.1.2	AUTOSAR Release Management	• Added control functions for ARP • Clarifications and corrections of requirements • Editorial changes • Removed chapter(s) on change documentation
2013-03-15	4.1.1	AUTOSAR Administration	• Initial Release

Disclaimer

Contents

1 Introduction and functional overview

This specification describes the functionality, API and the configuration for the AUTOSAR Basic Software module TCP/IP.

The AUTOSAR TCP/IP module offers functionality to send and receive Internet Protocol data. The TCP/IP Stack (TcpIp) is located between the Socket Adaptor (SoAd) and the Ethernet Interface (EthIf) modules.

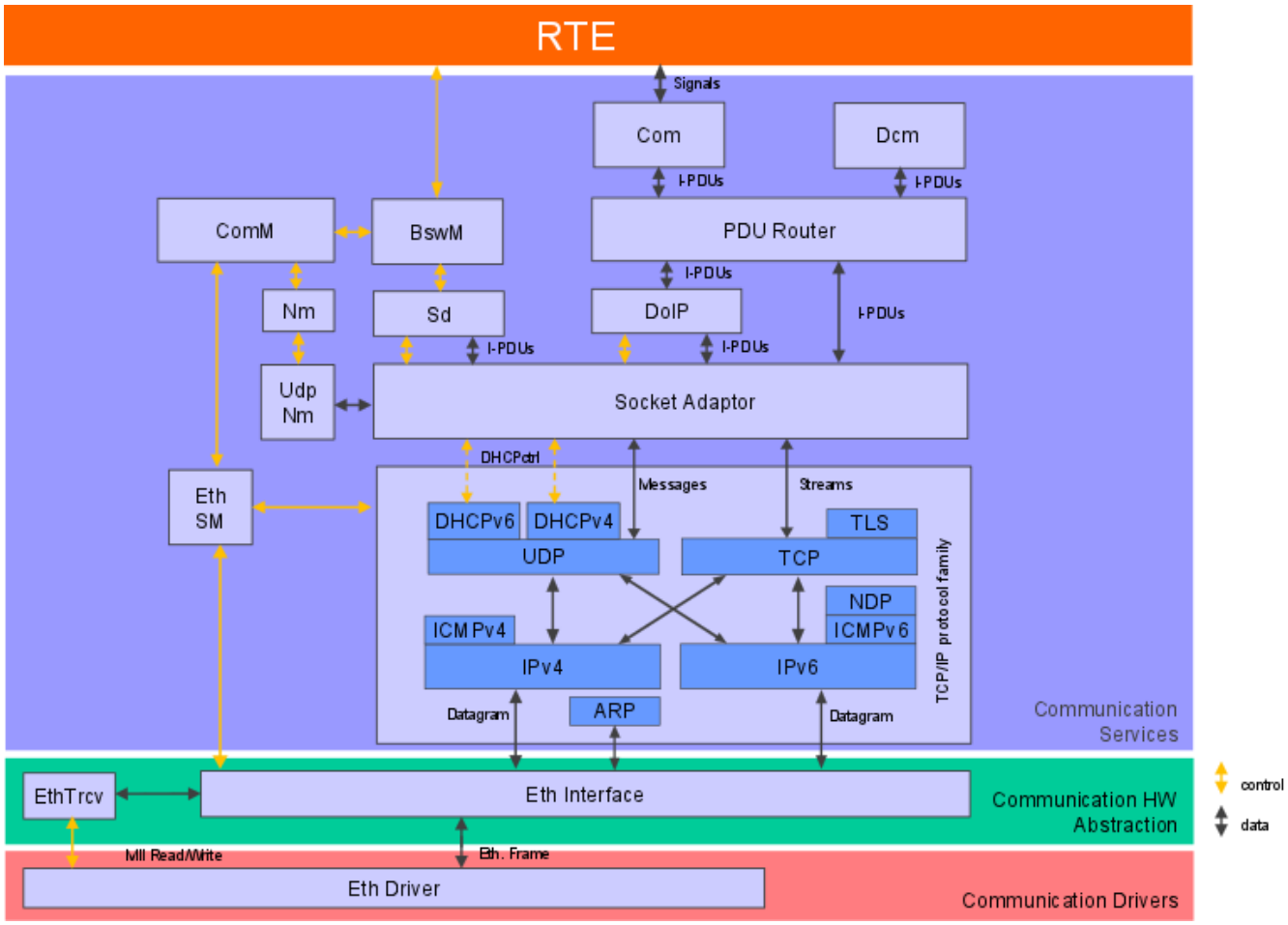

Figure 1.1: Extended AUTOSAR Communication Stack

2 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to the Tcp/Ip module that are not included in the [1, AUTOSAR glossary].

Abbreviation / Acronym:	Description:
ARP	Address Resolution Protocol
DAD	Duplicate Address Detection
DEM	Diagnostic Event Manager
DET	Default Error Tracer
DHCP	Dynamic Host Configuration Protocol
DHCPv4	Dynamic Host Configuration Protocol for Internet Protocol Version 4
DHCPv6	Dynamic Host Configuration Protocol for Internet Protocol Version 6
ECC	Elliptic Curve Cryptography
ECU	Electronic Control Unit
EthIf	Ethernet Interface
EthSM	Ethernet State Manager
HSM	Hardware Security Module
HTTP	HyperText Transfer Protocol
IANA	Internet Assigned Numbers Authority
ICMP	Internet Control Message Protocol
ICMPv4	Internet Control Message Protocol for Internet Protocol Version 4
ICMPv6	Internet Control Message Protocol for Internet Protocol Version 6
IETF	Internet Engineering Task Force
IP	Internet Protocol
IND	Inverse Neighbor Discovery
IPsec	Internet Protocol Security
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
MTU	Maximum Transmission Unit
NDP	Neighbor Discovery Protocol
PKI	Public Key Infrastructure
PRF	Pseudo Random Function
RNG	Random Number Generator
RSA	Rivest-Shamir-Adleman. A method using public and private key for data encryption and decryption.
SACK	Selective Acknowledgment
SNI	Server Name Identification
SoAd	Socket Adaptor
TCP	Transmission Control Protocol
TCP/IP	A family of communication protocols used in computer networks
TLS	Transport Layer Security
TP	Transport Protocol
UDP	User Datagram Protocol

Table 2.1: Acronyms and abbreviations used in the scope of this Document

3 Related documentation

3.1 Input documents & related standards and norms

[1] Glossary
AUTOSAR_FO_TR_Glossary

[2] Layered Software Architecture
AUTOSAR_CP_EXP_LayeredSoftwareArchitecture

[3] Specification of Basic Software Mode Manager
AUTOSAR_CP_SWS_BSWModeManager

[4] Specification of Socket Adaptor
AUTOSAR_CP_SWS_SocketAdaptor

[5] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[6] Requirements on Ethernet Support in AUTOSAR
AUTOSAR_CP_SRS_Ethernet

[7] Specification of ECU Configuration
AUTOSAR_CP_TPS_ECUConfiguration

[8] List of Basic Software Modules
AUTOSAR_CP_TR_BSWModuleList

[9] Specification of Crypto Service Manager
AUTOSAR_CP_SWS_CryptoServiceManager

[10] Specification of Key Manager
AUTOSAR_CP_SWS_KeyManager

[11] Requirements on IPsec Protocol
AUTOSAR_FO_RS_IPsecProtocol

[12] Dynamic Configuration of IPv4 Link-Local Addresses
https://rfc-editor.org/rfc/rfc3927.txt

[13] Requirements for Internet Hosts - Communication Layers
https://rfc-editor.org/rfc/rfc1122.txt

[14] An Ethernet Address Resolution Protocol:Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware
https://rfc-editor.org/rfc/rfc826.txt

[15] A Standard for the Transmission of IP Datagrams over Ethernet Networks
https://www.rfc-editor.org/info/rfc894

[16] Internet Protocol
https://rfc-editor.org/rfc/rfc791.txt

[17] IP DATAGRAM REASSEMBLY ALGORITHMS
https://www.rfc-editor.org/info/rfc815

[18] Classless Inter-domain Routing (CIDR):The Internet Address Assignment and Aggregation Plan
https://www.rfc-editor.org/info/rfc4632

[19] Host Extensions for IP Multicasting
https://www.rfc-editor.org/info/rfc1112

[20] Internet Control Message Protocol
https://rfc-editor.org/rfc/rfc792.txt

[21] Path MTU Discovery
https://www.rfc-editor.org/info/rfc1191

[22] Dynamic Host Configuration Protocol
https://rfc-editor.org/rfc/rfc2131.txt

[23] User Datagram Protocol
https://rfc-editor.org/rfc/rfc768.txt

[24] Transmission Control Protocol
https://rfc-editor.org/rfc/rfc793.txt

[25] TCP Congestion Control
https://rfc-editor.org/rfc/rfc5681.txt

[26] Internet Protocol, Version 6 (IPv6) Specification
https://www.rfc-editor.org/info/rfc8200

[27] IP Version 6 Addressing Architecture
https://rfc-editor.org/rfc/rfc4291.txt

[28] Transmission of IPv6 Packets over Ethernet Networks
https://rfc-editor.org/rfc/rfc2464.txt

[29] Default Address Selection for Internet Protocol Version 6 (IPv6)
https://rfc-editor.org/rfc/rfc6724.txt

[30] Handling of Overlapping IPv6 Fragments
https://rfc-editor.org/rfc/rfc5722.txt

[31] Deprecation of Type 0 Routing Headers in IPv6
https://rfc-editor.org/rfc/rfc5095.txt

[32] IPv6 Stateless Address Autoconfiguration
https://rfc-editor.org/rfc/rfc4862.txt

[33] Path MTU Discovery for IP version 6
https://rfc-editor.org/rfc/rfc1981.txt

[34] Optimistic Duplicate Address Detection (DAD) for IPv6
https://rfc-editor.org/rfc/rfc4429.txt

[35] Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification
https://rfc-editor.org/rfc/rfc4443.txt

[36] Neighbor Discovery for IP version 6 (IPv6)
https://rfc-editor.org/rfc/rfc4861.txt

[37] Dynamic Host Configuration Protocol for IPv6 (DHCPv6)
https://rfc-editor.org/rfc/rfc3315.txt

[38] The Dynamic Host Configuration Protocol (DHCP) Client Fully Qualified Domain Name (FQDN) Option
https://rfc-editor.org/rfc/rfc4702.txt

[39] The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Client Fully Qualified Domain Name (FQDN) Option
https://rfc-editor.org/rfc/rfc4704.txt

[40] The NewReno Modification to TCP's Fast Recovery Algorithm
https://rfc-editor.org/rfc/rfc6582.txt

[41] DHCP Options and BOOTP Vendor Extensions
https://rfc-editor.org/rfc/rfc2132.txt

[42] IPv6 Subnet Model:The Relationship between Links and Subnet Prefixes
https://rfc-editor.org/rfc/rfc5942.txt

[43] IPv6 Flow Label Specification
https://rfc-editor.org/rfc/rfc6437.txt

[44] Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers
https://rfc-editor.org/rfc/rfc2474.txt

[45] The Transport Layer Security (TLS) Protocol Version 1.2
https://rfc-editor.org/rfc/rfc5246.txt

[46] Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)
https://rfc-editor.org/rfc/rfc4492.txt

[47] Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)
https://www.rfc-editor.org/info/rfc7525

[48] Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)
https://rfc-editor.org/rfc/rfc4279.txt

[49] Encrypt-then-MAC for Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)
https://www.rfc-editor.org/info/rfc7366

[50] The Transport Layer Security (TLS) Protocol Version 1.3

https://tools.ietf.org/html/rfc8446

[51] Record Size Limit Extension for TLS
https://tools.ietf.org/html/rfc8449

[52] Dynamic Host Configuration Protocol (DHCP) and Bootstrap Protocol (BOOTP) Parameters
https://www.iana.org/assignments/bootp-dhcp-parameters/bootp-dhcp-parameters.xhtml

[53] Dynamic Host Configuration Protocol for IPv6 (DHCPv6)
https://www.iana.org/assignments/dhcpv6-parameters/dhcpv6-parameters.xhtml

[54] RFC 4301, Security Architecture for the Internet Protocol

[55] RFC 4302, IP Authentication Header

[56] RFC 4303, IP Encapsulating Security Payload (ESP)

[57] RFC 7296, Internet Key Exchange Protocol Version 2 (IKEv2)

[58] RFC 4304, Extended Sequence Number (ESN) Addendum to IPsec Domain of Interpretation (DOI) for Internet Security Association

[59] RFC 8221, Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)

[60] RFC 4478, Repeated Authentication in Internet Key Exchange (IKEv2) Protocol

[61] RFC 3706, A Traffic-Based Method of Detecting Dead Internet Key Exchange (IKE) Peers

[62] RFC 7427, Signature Authentication in the Internet Key Exchange Version 2 (IKEv2)

[63] RFC 4543, The Use of Galois Message Authentication Code (GMAC) in IPsec ESP and AH

[64] RFC 4494, The AES-CMAC-96 Algorithm and Its Use with IPsec

[65] RFC 4106, The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)

[66] RFC 4309, Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)

[67] RFC 6379, Suite B Cryptographic Suites for IPsec

[68] RFC 8247, Algorithm Implementation Requirements and Usage Guidance for the Internet Key Exchange Protocol Version 2 (IKEv2)

[69] Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation
https://www.rfc-editor.org/info/rfc7383

[70] Road vehicles – Diagnostic communication over Internet Protocol (DoIP) – Part 2: Network and transport layer requirements and services (Release 2019-12) https://www.iso.org

3.2 Related specification

AUTOSAR provides a Specification of Layered Software Architecture [2], which is also valid for TcpIp.

AUTOSAR provides a Specification of Basis Software Mode Manager [3].

AUTOSAR provides a Specification of Socket Adaptor [4].

AUTOSAR provides a General Specification for Basic Software modules [5], which is also valid for TcpIp.

Thus, the specification SWS BSW General shall be considered as additional and required specification for TcpIp.

AUTOSAR provides a Specification of Ethernet [6].

AUTOSAR provides a Specification of ECU Configuration [7], which is also valid for TcpIp.

AUTOSAR provides a List of Basic Software Modules [8].

AUTOSAR provides a Specification of Crypto Service Manager [9].

AUTOSAR provides a Specification of Key Manager [10].

AUTOSAR provides a Specification of IPsecProtocol [11].

IETF RFC 3927 [12].

IETF RFC 1122 [13].

IETF RFC 826 [14].

IETF RFC 894 [15].

IETF RFC 791 [16].

IETF RFC 815 [17].

IETF RFC 4632 [18].

IETF RFC 1112 [19].

IETF RFC 792 [20].

IETF RFC 1191 [21].

IETF RFC 2131 [22].

IETF RFC 768 [23].

IETF RFC 793 [24].

IETF RFC 5681 [25].

IETF RFC 8200 [26].

IETF RFC 4291 [27].

IETF RFC 2464 [28].

IETF RFC 6724 [29].

IETF RFC 5722 [30].

IETF RFC 5095 [31].

IETF RFC 4862 [32].

IETF RFC 1981 [33].

IETF RFC 4429 [34].

IETF RFC 4443 [35].

IETF RFC 4861 [36].

IETF RFC 3315 [37].

IETF RFC 4702 [38].

IETF RFC 4704 [39].

IETF RFC 6582 [40].

IETF RFC 2132 [41].

IETF RFC 5942 [42].

IETF RFC 6437 [43].

IETF RFC 2474 [44].

IETF RFC 5246 [45].

IETF RFC 4492 [46].

IETF RFC 7525 [47].

IETF RFC 4279 [48].

IETF RFC 7366 [49].

IETF RFC 8446 [50].

IETF RFC 8449 [51].

IANA DHCP Options [52].

IANA DHCPv6 Options [53].

IETF RFC 4301 [54].

IETF RFC 4302 [55].

IETF RFC 4303 [56].

IETF RFC 7296 [57].

IETF RFC 4304 [58].

IETF RFC 8221 [59].

IETF RFC 4478 [60].

IETF RFC 3706 [61].

IETF RFC 7427 [62].

IETF RFC 4543 [63].

IETF RFC 4494 [64].

IETF RFC 4106 [65].

IETF RFC 4309 [66].

IETF RFC 6379 [67].

IETF RFC 8247 [68].

IETF RFC 7383 [69].

IETF ISO13400-2 [70].

4 Constraints and assumptions

4.1 Limitations

This document does not cover the assignment of UDP or TCP port numbers. There is no reserved space within the IANA assigned number range. Each implementer is responsible for managing the used port numbers.

This document does not cover the management of IP addresses. This might be done dynamically, e.g. by using DHCP, or statically. It is the implementer's responsibility to prevent address conflicts and achieve compliance with IANA address assignments.

This specification does not prescribe a certain physical layer or data rate. Although a CDD interface is specified, allowing additional upper layer modules, a fan-out of one socket to multiple upper layer modules is not intended to be supported.

The AUTOSAR TLS implementation has the following limitations:

- A TLS implementation shall not support data compression or decompression.
- Session renegotiation shall not be supported.
- No support for secure connection over UDP (e.g. for DTLS)
- No support of FQDN
- No client Hello padding extension IETF RFC7685
- No session hash and extended master secret IETF RFC 7627
- No support for TLS versions lower than 1.2.
- No support for dynamic "downgrading" of a TCP connection with an established TLS connection to a plain TCP connection (without TLS)
- Static TLS connection assignment is bound to the port configuration of the server. Thus, using different TLS settings for different connections (possibly originating from different clients) to the same server port is not possible.

The AUTOSAR IPsec implementation has the following limitations:

- IPsec in "tunnel mode" is not supported right now. Transport mode only.
- IPv6 is not supported
- Multicast is not supported

4.2 Applicability to car domains

No restrictions.

5 Dependencies to other modules

5.1 EthIf

The Ethernet Interface is the lower layer module of the Tcp/Ip module.

5.2 EthSM

The Ethernet State Manager controls the communication mode of the Tcp/Ip module by requesting communication modes from the Tcp/Ip module. Tcp/Ip notifies the EthSM about communication mode changes.

5.3 Socket Adaptor

The Socket Adaptor is the upper layer module of the Tcp/Ip module.

5.4 KeyM

The Key Manager module provides operations for certificate handling for the TLS and IPsec sub module.

5.5 CSM

The crypto service manager allows to perform crypto job and key operations used by the TLS and IPsec sub module.

5.6 File structure

5.6.1 Code file structure

For details refer to the chapter 5.1.6 "Code file structure" in SWS_BSWGeneral.

5.7 Version check

For details refer to the chapter 5.1.8 "Version Check" in SWS_BSWGeneral.

6 Requirements Tracing

The following tables reference the requirements specified in chapter 3 and links to the fulfillment of these. Please note that if column “Satisfied by” is empty for a specific requirement this means that this requirement is not fulfilled by this document.

Requirement	Description	Satisfied by
[RS_IPSEC_00004]	The Internet Key Exchange (IKEv2) Protocol shall be supported according to IETF RFC 7296	[SWS_TcpIp_00353]
[RS_IPSEC_00010]	IKEv2 shall support periodic reauthentication and rekeying	[SWS_TcpIp_00355]
[RS_IPSEC_00011]	IKEv2 shall support a seamless handover of exchanged keys	[SWS_TcpIp_00355]
[RS_IPSEC_00013]	IKEv2 shall support dead peer detection	[SWS_TcpIp_00355]
[RS_IPSEC_00014]	IKEv2 shall support authentication based on X.509v3 certificates with digital signatures	[SWS_TcpIp_00356]
[RS_IPSEC_00021]	All algorithms which are classified as "MUST" in IETF RFC 8247 shall be supported by IKEv2	[SWS_TcpIp_00353]
[RS_IPSEC_00022]	IPsec’s Security Policy Database (SPD) shall be configurable for IPs, IP ranges, protocols, ports and port ranges	[SWS_TcpIp_00357]
[RS_IPSEC_00023]	IPsec’s Security Policy Database (SPD) default behavior shall be BYPASS	[SWS_TcpIp_00357]
[RS_IPSEC_00025]	IPsec’s Peer Authorization Database (PAD) shall be configurable for use with X.509v3	[SWS_TcpIp_00356]
[RS_IPSEC_00027]	It shall be possible to define the priority order of the algorithms used by IKEv2 during the IKE_INIT negotiations	[SWS_TcpIp_00358]
[RS_Ids_00810]	Basic SW security events	[SWS_TcpIp_00361] [SWS_TcpIp_00362]
[SRS_BSW_00323]	All AUTOSAR Basic Software Modules shall check passed API parameters for validity	[SWS_TcpIp_00147]
[SRS_BSW_00452]	Classification of runtime errors	[SWS_TcpIp_00282] [SWS_TcpIp_00283]
[SRS_Eth_00016]	ICMPv4 shall be implemented according to IETF RFC 792	[SWS_TcpIp_00277] [SWS_TcpIp_00297]
[SRS_Eth_00019]	TCP and UDP related requirement specified in IETF RFC 1122 shall be implemented	[SWS_TcpIp_00279] [SWS_TcpIp_00280]
[SRS_Eth_00045]	TCPIP automatic IP address assignment	[SWS_TcpIp_00254]
[SRS_Eth_00065]	An API shall be available to fill DHCP options field	[SWS_TCPIP_00020] [SWS_TCPIP_00190] [SWS_TcpIp_00243] [SWS_TcpIp_00244] [SWS_TcpIp_00245] [SWS_TcpIp_00246] [SWS_TcpIp_00247] [SWS_TcpIp_00248] [SWS_TcpIp_00249] [SWS_TcpIp_00250] [SWS_TcpIp_00251] [SWS_TcpIp_00252]

▽

△

Requirement	Description	Satisfied by
[SRS_Eth_00066]	An API shall be available to read any received DHCP options field	[SWS_TCPIP_00040] [SWS_TCPIP_00189] [SWS_TcpIp_00233] [SWS_TcpIp_00234] [SWS_TcpIp_00235] [SWS_TcpIp_00236] [SWS_TcpIp_00237] [SWS_TcpIp_00238] [SWS_TcpIp_00239] [SWS_TcpIp_00240] [SWS_TcpIp_00241] [SWS_TcpIp_00242]
[SRS_Eth_00087]	Semi-Static Auto-Configuration	[SWS_TcpIp_00058] [SWS_TcpIp_00201] [SWS_TcpIp_00216] [SWS_TcpIp_00217] [SWS_TcpIp_00218] [SWS_TcpIp_00219]
[SRS_Eth_00088]	DHCP Server	[SWS_TcpIp_00058] [SWS_TcpIp_00200]
[SRS_Eth_00090]	The Neighbor Discovery Protocol shall be implemented according to IETF RFC 4861	[SWS_TcpIp_00164] [SWS_TcpIp_00263] [SWS_TcpIp_00264] [SWS_TcpIp_00281]
[SRS_Eth_00091]	The Optimistic Duplicate Address Detection (DAD) for IPv6 shall be implemented according to IETF RFC 4429	[SWS_TcpIp_00282] [SWS_TcpIp_00283]
[SRS_Eth_00092]	The IPv6 Addressing Architecture shall be implemented according to IETF RFC 4291	[SWS_TcpIp_00162] [SWS_TcpIp_00269]
[SRS_Eth_00097]	The Path MTU Discovery for IPv6 shall be implemented according to IETF RFC 1981	[SWS_TcpIp_00267] [SWS_TcpIp_00268]
[SRS_Eth_00098]	ICMPv6 shall be implemented according to IETF RFC 4443	[SWS_TcpIp_00278] [SWS_TcpIp_00298]
[SRS_Eth_00103]	TcpIp shall support generic upper layers	[SWS_TCPIP_00018] [SWS_TCPIP_00223] [SWS_TCPIP_00224] [SWS_TCPIP_00225] [SWS_TCPIP_00226] [SWS_TCPIP_00227] [SWS_TCPIP_00228] [SWS_TCPIP_00229] [SWS_TcpIp_00220] [SWS_TcpIp_00221] [SWS_TcpIp_00222]
[SRS_Eth_00109]	TCP shall support the Nagle algorithm according to IETF RFC 896	[SWS_TcpIp_00063]
[SRS_Eth_00110]	The Relationship between Links and Subnet Prefixes shall be considered according to IETF RFC 5942	[SWS_TcpIp_00265]
[SRS_Eth_00111]	Robustness against unexpected communication patterns	[SWS_TcpIp_00260] [SWS_TcpIp_00261] [SWS_TcpIp_00262] [SWS_TcpIp_00266] [SWS_TcpIp_00370] [SWS_TcpIp_00371]
[SRS_Eth_00112]	Ethernet-related BSW modules shall report relevant runtime errors from the used protocols	[SWS_TCPIP_00255] [SWS_TcpIp_00256] [SWS_TcpIp_00257] [SWS_TcpIp_00258] [SWS_TcpIp_00259]
[SRS_Eth_00129]	The TCPIP shall support access to measurement counter values	[SWS_TcpIp_00284] [SWS_TcpIp_00285] [SWS_TcpIp_00286] [SWS_TcpIp_00287] [SWS_TcpIp_00288] [SWS_TcpIp_00289] [SWS_TcpIp_00290] [SWS_TcpIp_00291] [SWS_TcpIp_00292] [SWS_TcpIp_00293] [SWS_TcpIp_00294] [SWS_TcpIp_00295] [SWS_TcpIp_00296]
[SRS_Eth_00134]	Configuration of ciphersuites for TLS connections	[SWS_TcpIp_00311]
[SRS_Eth_00135]	The number of TLS connections that can be opened in parallel shall be configurable	[SWS_TcpIp_00326]
[SRS_Eth_00136]	The size of a TLS fragment length shall be configurable	[SWS_TcpIp_00327]

▽

△

Requirement	Description	Satisfied by
[SRS_Eth_00137]	PSK Identity to PSK mapping shall be possible using custom software.	[SWS_TCPIP_91013] [SWS_TCPIP_91014] [SWS_TCPIP_91015] [SWS_TcpIp_00325]
[SRS_Eth_00138]	TLS shall support at least basic requirements as defined in IETF RFC 5246 for version 1.2 or higher	[SWS_TcpIp_00300] [SWS_TcpIp_00302]
[SRS_Eth_00139]	TLS shall support elliptic curve cryptography as defined in IETF RFC 4492	[SWS_TcpIp_00304]
[SRS_Eth_00140]	TLS for diagnostic communication (DoIP) shall support at least one ciphersuite as defined in ISO13400-2.	[SWS_TcpIp_00300] [SWS_TcpIp_00329]
[SRS_Eth_00141]	TLS shall support the use of pre-shared keys as defined in IETF RFC 4279	[SWS_TcpIp_00325]
[SRS_Eth_00142]	The Security Architecture for the Internet Protocol shall be implemented according to IETF RFC 4301	[SWS_TcpIp_00352]
[SRS_Eth_00143]	The IP Authentication Header (AH) shall be implemented according to IETF RFC 4302	[SWS_TcpIp_00352]
[SRS_Eth_00144]	IP Encapsulating Security Payload (ESP) shall be implemented according to IETF RFC 4303	[SWS_TcpIp_00352]
[SRS_Eth_00145]	The Internet Key Exchange (IKEv2) Protocol shall be implemented according to IETF RFC 7296	[SWS_TcpIp_00352]

Table 6.1: RequirementsTracing

7 Functional specification

Figure 2 provides an architecture overview of the AUTOSAR TCP/IP stack. The TCP/IP stack consists of the sub modules within the red box. Furthermore the interaction with other AUTOSAR modules (beside Dem and Det) is shown.

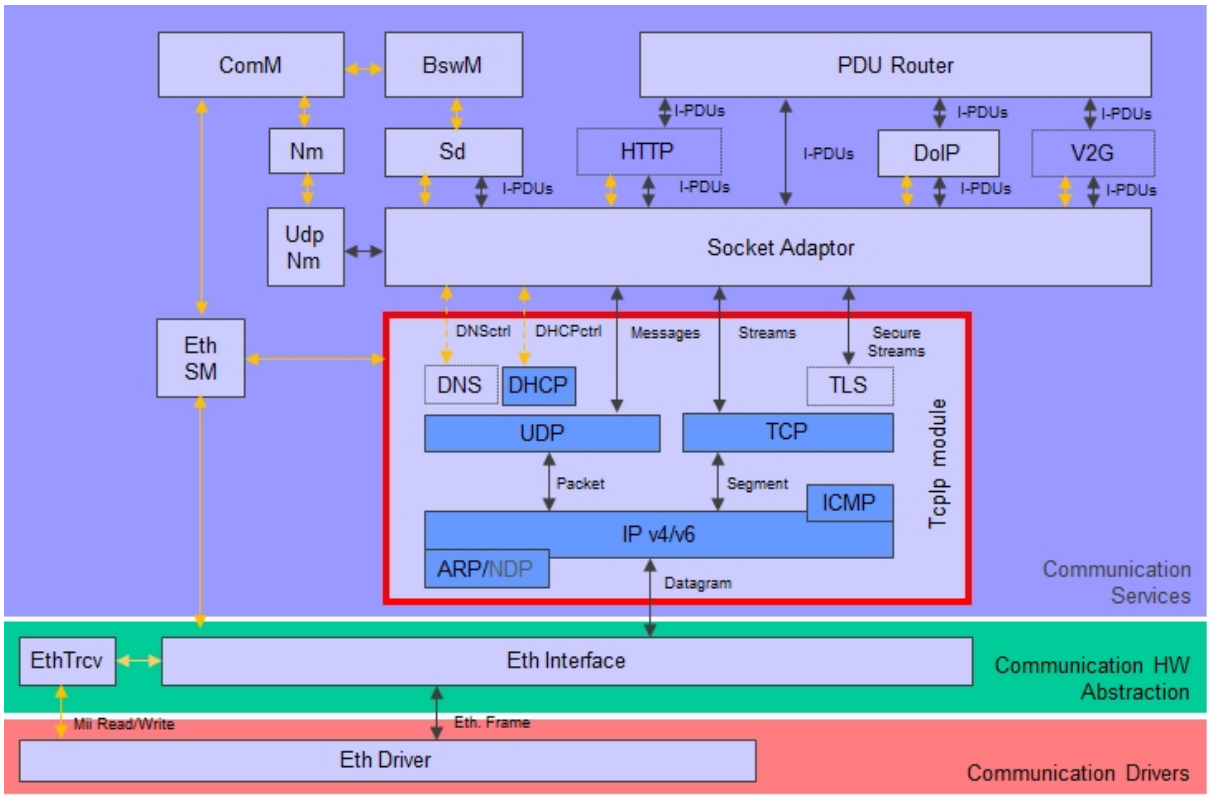

Figure 7.1: TCP/IP Architecture Overview

[SWS_TcpIp_00052] ⌈The TCP/IP stack shall consist of sub modules implementing specific functionalities defined in the subchapters below.⌋*()*

7.1 System Scalability

7.1.1 Background & Rationale

The TcpIp module supports a variety of different use case, not all of them are required by each user. In order to achieve a scalable TcpIp Stack the protocols shall be grouped according to the following scalability classes:

- **Scalability Class 1:**

  IPv4 - In-Vehicle and Diagnostic Communication

- **Scalability Class 2:**

IPv6 - In-Vehicle and Diagnostic Communication

- **Scalability Class 3:**

  IPv4 and IPv6 (Dual Stack) - In-Vehicle and Diagnostic Communication

The following protocols shall be available in the respective Scalability Class:

Feature	Scalability Class 1	Scalability Class 2	Scalability Class 3
IPv4	X		X
ARP	X		X
ICMPv4	X		X
DHCPv4	X		X
Auto-IP	X		X
UDP	X	X	X
TCP	X	X	X
IPv6		X	X
NDP		X	X
ICMPv6		X	X
DHCPv6		X	X

Table 7.1: TcpIp Scalability Classes

In addition to the scalability classes, the following Feature Groups allow a more fine-grained selection of optional features to address the specific needs of certain ECUs.

IPv4-Global Communication Feature Group:

The following features are available for Scalability Classes 1 and 3.

- Path MTU Discovery

IPv6-Global Communication Feature Group:

The following features are available for Scalability Classes 2 and 3.

- Path MTU Discovery
- IPv6 Anycasts Addresses
- NDP Redirect Messages

Special Features Group:

The following features are available for Scalability Classes 1, 2 and 3.

- DHCP Server

Security Features Group:

The following features are available for Scalability Classes 1, 2 and 3.

- TLS
- IPsec

7.1.2 Requirements

[SWS_TcpIp_00148] ⌈The TcpIp module for IPv4 - In-Vehicle and Diagnostic Communication (Scalability class 1) shall support the features listed in Figure 3: TcpIp Scalability Classes, column Scalability Class 1.⌋*()*

[SWS_TcpIp_00149] ⌈The TcpIp module for IPv6 - In-Vehicle and Diagnostic Communication (Scalability class 2) shall support the features listed in Figure 3: TcpIp Scalability Classes, column Scalability Class 2.⌋*()*

[SWS_TcpIp_00150] ⌈The TcpIp module for IPv4 and IPv6 (Dual Stack) - In-Vehicle and Diagnostic Communication (Scalability class 3) shall support the features listed in Figure 3: TcpIp Scalability Classes, column Scalability Class 3.⌋*()*

7.2 Internet Protocol Version 4

7.2.1 Internet Protocol (IPv4)

The Internet Protocol (IP) is the main protocol of the TCP/IP stack and is responsible for delivering datagrams from a source host identified by the source address to one or multiple destination hosts identified by the destination address. IP hides the underlying physical network interface, is an unreliable, best-effort, and connectionless packet delivery protocol.

[SWS_TcpIp_00053] ⌈The TcpIp shall implement the Internet Protocol as defined in IETF RFC 791 (Internet Protocol of version 4).⌋*()*

[SWS_TcpIp_00095] ⌈The TcpIp shall encapsulate IP packets in Ethernet frames according to IETF RFC 894.⌋*()*

[SWS_TcpIp_00096] ⌈The TcpIp shall support the identification of the network an IP address belongs to, by using a network mask (prefix) in addition to the IP address according to IETF RFC 4632, section 3.1.⌋*()*

[SWS_TcpIp_00102] ⌈The TcpIp shall fulfill the Internet Protocol related requirements specified by IETF RFC 1122, section 3.2.1.1 (Version number), 3.2.1.2 (Checksum), 3.2.1.3 (Addressing), 3.2.1.7 (TTL), and 3.3.2 (Reassembly).⌋*()*

[SWS_TcpIp_00097] ⌈The TcpIp shall be able to transmit IP datagrams to a group of hosts identified by a single IP destination address (multicast address) according to IETF RFC 1112, section 4, 6.2, and 6.4.⌋*()*

[SWS_TcpIp_00098] ⌈The TcpIp shall be able to receive multicast IP datagrams identified by a single IP destination address (multicast address) according to IETF RFC 1112, section 4 and 7.2 (excluding the requirement for IGMP).⌋*()*

[SWS_TcpIp_00054] ⌈The TcpIp shall be able to reassemble incoming datagrams that are fragmented according to IETF RFC 815 (IP Datagram Reassembly Algorithms).⌋*()*

[SWS_TcpIp_00231] ⌈The TcpIp shall fragment oversized IPv4 frames before transmission according to the description in IETF 791 Section Fragmentation and Reassembly.⌋*()*

[SWS_TcpIp_00055] ⌈The TcpIp shall discover the maximum transmission unit (MTU) for a path as defined in IETF RFC 1191 (Path MTU Discovery).⌋*()*

7.2.2 Address Resolution Protocol (ARP)

[SWS_TcpIp_00056] ⌈The TcpIp shall implement the Address Resolution Protocol (ARP) as defined in IETF RFC 826.⌋*()*

[SWS_TcpIp_00090] ⌈The TcpIp shall limit the number of ARP table (address resolution cache) entries to the number specified by the configuration parameter `TcpIpArpTableSizeMax`.⌋*()*

[SWS_TcpIp_00091] ⌈The TcpIp shall remove entries of the ARP table if they are not used for the timeout specified by the configuration parameter `TcpIpArpTableEntryTimeout`. If `TcpIpArpTableEntryTimeout` is set to INF, the TcpIp module shall never remove entries from the ARP table.⌋*()*

[SWS_TcpIp_00092] ⌈If `TcpIpArpDefensiveProcessing` is set to FALSE, the TcpIp shall use the information from each received IP packet to update the ARP table in addition to received ARP packets.⌋*()*

[SWS_TcpIp_00142] ⌈The TcpIp shall call `Up_PhysAddrTableChg`() directly after each ARP table change:

- If TcpIp adds a new entry or updates an existing one, the parameter valid shall be set to TRUE and the parameters IpAddrPtr and PhysAddrPtr shall be set according to the new or updated entry.
- In case TcpIp removes an entry, valid shall be set to FALSE and the parameters IpAddrPtr and PhysAddrPtr shall be set according to the removed entry.

⌋*()*

[SWS_TcpIp_00350] ⌈After the transmission of an ARP request the TcpIp shall skip the transmission of any further ARP requests to the same destination within a duration

of TcpIpArpRequestTimeout seconds, according to the mechanism to prevent ARP flooding described in IETF RFC 1122, section 2.3.2.1 ARP Cache Validation.⌋*()*

[SWS_TcpIp_00351] ⌈The TcpIp shall process received ARP packets either directly within the context of the TcpIp_RxIndication() or the first subsequent TcpIp_MainFunction().⌋*()*

[SWS_TcpIp_00093] ⌈On assignment of a new IP address the TcpIp shall send a configurable number (TcpIpArpNumGratuitousARPonStartup) of gratuitous ARP replies according to IETF RFC 2002, section 4.6, second indent. These announcements shall be timed according to IETF RFC 5227 section 2.3. Announcing an Address.⌋*()*

[SWS_TcpIp_00370] ⌈If TcpIpArpDefensiveProcessing is set to TRUE, the ARP shall silently discard all received ARP packets that have not been requested by a previously transmitted ARP request.⌋*(SRS_Eth_00111)*

[SWS_TcpIp_00371] ⌈If TcpIpArpDefensiveProcessing is set to TRUE, the ARP shall skip the update of the ARP table upon processing received Gratuitous ARP packets.⌋*(SRS_Eth_00111)*

7.2.3 Dynamic Configuration of IPv4 Link-Local Addresses (Auto-IP)

[SWS_TcpIp_00057] ⌈The TcpIp shall support the dynamic configuration of IPv4 Link Local addresses as defined in IETF RFC 3927 (Dynamic Configuration of IPv4 Link-Local Addresses).⌋*()*

7.2.4 Internet Control Message Protocol (ICMPv4)

[SWS_TcpIp_00059] ⌈The TcpIp shall support the transmission and reception of Internet Control Message Protocol (ICMPv4) messages as defined in IETF RFC 792 (Internet Control Message Protocol in version 4).⌋*()*

[SWS_TcpIp_00277] ⌈The TcpIp shall only reply to ICMPv4 Echo Request Messages if they are valid and TcpIpIcmpEchoReplyEnabled is set to TRUE.⌋*(SRS_Eth_00016)*

[SWS_TcpIp_00297] ⌈If a TcpIpIcmpMsgHandler is configured, the TcpIp shall call the respective Up_IcmpMsgHandler() if an ICMPv4 message is received and not handled by the TcpIp directly.⌋*(SRS_Eth_00016)*

Note: For example, if the TcpIp replies to an ICMP echo request Up_IcmpMsgHandler() is not called for this message.

7.3 Internet Protocol Version 6

[SWS_TcpIp_00376]{DRAFT} ⌈TcpIp shall process a received Inverse Neighbor Discovery (IND) Solicitation message and respond with an Advertisement message, as described in IETF RFC 3122. The neighbor cache shall be updated with the values provided in the Solicitation message.⌋*()*

[SWS_TcpIp_00377]{DRAFT} ⌈TLS shall be able to process X.509 v3 certificates and X.509 v2 certificate revocation list (CRL) profiles, as described in IETF RFC 5280.⌋*()*

[SWS_TcpIp_00378]{DRAFT} ⌈If `TcpIpTcpWindowScaleOptionEnabled` is set to TRUE, the TCP window scale option (WSopt) and mechanism shall be supported according to IETF RFC 7323, chapter 2. The value given by `TcpIpTcpWindowScale` shall be the value transmitted in SYN message and the limiting factor when replying with SYN-ACK message.⌋*()*

[SWS_TcpIp_00153] ⌈The TcpIp shall support the frame format for transmission of IPv6 packets and the method of forming IPv6 link-local addresses and statelessly autoconfigured addresses on Ethernet networks as defined in IETF RFC 2464 (Transmission of IPv6 Packets over Ethernet Networks).⌋*()*

[SWS_TcpIp_00154] ⌈The TcpIp shall support the source address selection algorithm as defined in IETF RFC 6724 (Default Address Selection for Internet Protocol Version 6 (IPv6)). Only section 5 Source Address Selection shall be supported.⌋*()*

[SWS_TcpIp_00156] ⌈The TcpIp shall support the IETF RFC 5095 (Deprecation of Type 0 Routing Headers in IPv6). The functionality provided by IPv6's Type 0 Routing Header can be exploited in order to achieve traffic amplification over a remote path for the purposes of generating denial-of-service traffic. This document updates the IPv6 specification to deprecate the use of IPv6 Type 0 Routing Headers, in light of this security concern.⌋*()*

[SWS_TcpIp_00157] ⌈The TcpIp shall support the section 5.1. Node Configuration Variables, section 5.3. Creation of Link-Local Addresses, section 5.4, Duplicate Address Detection, section 5.5 Creation of Global Addresses and section 5.6 Configuration Consistency of the IETF RFC 4862 (IPv6 Stateless Address Autoconfiguration).⌋*()*

[SWS_TcpIp_00158] ⌈The TcpIp shall support the Path MTU Discovery for IPv6 as defined in IETF RFC 1981 (Path MTU Discovery for IP version 6). If the max. MTU is used, the Path MTU Discovery shall not try to increase the value.⌋*()*

[SWS_TcpIp_00159] ⌈The TcpIp shall support the Duplicate Address Detection as defined in IETF RFC 4429 (Optimistic Duplicate Address Detection (DAD) for IPv6).⌋*()*

7.3.1 Internet Protocol (IPv6)

[SWS_TcpIp_00160] ⌈The TcpIp shall support the basic IPv6 header and the initially defined IPv6 extension headers and options as defined in IETF RFC 8200 (Internet Protocol, Version 6 (IPv6) Specification).⌋*()*

[SWS_TcpIp_00161] ⌈The TcpIp shall support the reception and reassembly of fragmented IPv6 frames according to IETF RFC 8200 Section 4.5 Fragment Header.⌋*()*

[SWS_TcpIp_00155] ⌈The TcpIp shall support the section 4, first paragraph of the IETF RFC 5722 (Handling of Overlapping IPv6 Fragments). The IETF RFC 5722 demonstrates the security issues associated with allowing overlapping fragments and updates the IPv6 specification to explicitly forbid overlapping fragments (transmission and reception).⌋*()*

[SWS_TcpIp_00232] ⌈The TcpIp shall fragment oversized IPv6 frames before transmission according to IETF RFC 8200 Section 4.5 Fragment Header.⌋*()*

[SWS_TcpIp_00162] ⌈The TcpIp shall support the section 2, IPv6 Addressing of IETF RFC 4291 (IP Version 6 Addressing Architecture) excluding Section 2.6. Anycast Addresses. Section 2.8 A Node's Required Addresses shall be limited to the node requirements for host only.⌋*(SRS_Eth_00092)*

[SWS_TcpIp_00269] ⌈The TcpIp shall support the Section 2.6. Anycast Addresses of IETF RFC 4291 (IP Version 6 Addressing Architecture).⌋*(SRS_Eth_00092)*

7.3.2 Internet Control Message Protocol (ICMPv6)

[SWS_TcpIp_00163] ⌈The TcpIp shall support the Internet Control Message Protocol Version 6 as defined in IETF RFC 4443 (Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification).⌋*()*

[SWS_TcpIp_00278] ⌈The TcpIp shall only reply to ICMPv6 Echo Request Messages if they are valid and `TcpIpIcmpV6EchoReplyEnabled` is set to TRUE.⌋*(SRS_Eth_00098)*

[SWS_TcpIp_00298] ⌈If a `TcpIpIcmpV6MsgHandler` is configured, the TcpIp shall call the respective `Up_IcmpMsgHandler`() if an ICMPv6 message is received and not handled by the TcpIp directly.⌋*(SRS_Eth_00098)*

Note: For example, if the TcpIp replies to an ICMPv6 echo request `Up_IcmpMsgHandler`() is not called for this message.

7.3.3 Neighbor Discovery Protocol (NDP)

[SWS_TcpIp_00164] ⌈The TcpIp shall support the Neighbor Discovery protocol for IP Version 6 as defined in IETF RFC 4861 (Neighbor Discovery for IP version 6 (IPv6))

except the sections 4.5 Redirect Message Format, 6.2. Router Specification, 7.2.8. Proxy Neighbor Advertisements and 8. Redirect Function.⌋*(SRS_Eth_00090)*

[SWS_TcpIp_00281] ⌈The TcpIp shall support the handling of redirect messages as defined in IETF RFC 4861 (Neighbor Discovery for IP version 6 (IPv6)) Section 8.3. Host Specification.⌋*(SRS_Eth_00090)*

[SWS_TcpIp_00261] ⌈If `TcpIpNdpDefensiveProcessing` is set to TRUE, the NDP shall silently discard all received Neighbor Advertisements that have not been requested by a previously transmitted Neighbor Solicitation.⌋*(SRS_Eth_00111)*

[SWS_TcpIp_00262] ⌈If `TcpIpNdpDefensiveProcessing` is set to TRUE, the NDP shall skip the update of the Neighbor Cache upon processing received Neighbor Solicitations.⌋*(SRS_Eth_00111)*

[SWS_TcpIp_00263] ⌈The TcpIp shall limit the number of neighbor cache entries to the number specified by the configuration parameter `TcpIpNdpMaxNeighborCacheSize` ([ECUC_TcpIp_00129])⌋*(SRS_Eth_00090)*

[SWS_TcpIp_00264] ⌈In case the neighbor cache is full and a new entry shall be added, the TcpIp shall drop the oldest entry to be able to add the new entry⌋*(SRS_Eth_00090)*

[SWS_TcpIp_00265] ⌈The TcpIp shall adhere to the rules defined in IETF RFC 5942 - Section 4 "Host Rules" and shall use the updated definition of "on-link" according to IETF RFC 5942 - Section 6 "Updates to RFC 4861".⌋*(SRS_Eth_00110)*

[SWS_TcpIp_00165] ⌈If a packet shall be transmitted to a remote host and the link layer address does not exist in the Neighbor Cache, the TcpIp shall queue this packet according to IETF RFC 4861, section 7.2.2. Sending Neighbor Solicitations, 5th paragraph and transmit the packet when the address has been resolved.⌋*()*

7.4 Internet Protocol Security (IPsec)

[SWS_TcpIp_00352] ⌈TcpIp shall support IPsec according to AUTOSAR foundation RS_IPsecProtocol [11].⌋*(SRS_Eth_00142, SRS_Eth_00143, SRS_Eth_00144, SRS_Eth_00145)*

[SWS_TcpIp_00353] ⌈IKEv2 shall be implemented according to IETF RFC 7296 and [RS_IPSEC_00021] with the limitations defined in [RS_IPSEC_00004]. IKEv1 shall not be supported.⌋*(RS_IPSEC_00004, RS_IPSEC_00021)*

Note: To ensure that IKEv2 is interoperable with the IETF IPsec standards in general and resolve any ambiguities, the open source IPsec implementation strongSwan (strongswan.org) is used as reference.

[SWS_TcpIp_00355] ⌈The general IKEv2 connection configuration, e.g. connection lifetime and re-keying / re-authentication timeouts, dead peer detection, may be config-

ured via the settings in the container "IKEConnections".⌋(*RS_IPSEC_00010, RS_IPSEC_00011, RS_IPSEC_00013*)

[SWS_TcpIp_00356] ⌈The IKEv2 certificates used for for authentication with other IKEv2 nodes may be configured via the settings in the container "IKECertificates" and "IKECertificate".⌋(*RS_IPSEC_00014, RS_IPSEC_00025*)

[SWS_TcpIp_00357] ⌈The security policy database, which defines which connections shall be protected by IPsec and by which protections, may be configured via the settings in the container "TcpIpSpdEntry" and "TcpIpIpSecPriority". The IpSecPriority is used to establish the order in which the SpdEntries are checked. The first successful rule match will be executed, disregarding all lower priority rules.⌋(*RS_IPSEC_00022, RS_IPSEC_00023*)

[SWS_TcpIp_00358] ⌈The priority of proposed algorithms for IKEv2 handshakes may be configured in the container "IKEIkeSaProposal".⌋(*RS_IPSEC_00027*)

7.5 IP Based Protocols

7.5.1 Local Address Table

[SWS_TcpIp_00099] ⌈The TcpIp shall maintain a table of local IP addresses, which can be assigned to an EthIf controller during runtime according to the configuration container TcpIpLocalAddr (including its subcontainers).⌋()

Note: Each entry of the local IP address table is uniquely identified by the configuration parameter TcpIpAddrId.

[SWS_TcpIp_00100] ⌈In case no TcpIpStaticIpAddressConfig is provided, the TcpIp shall enable to specify a multicast IP address during runtime via TcpIp_RequestIpAddrAssignment().⌋()

[SWS_TcpIp_00130] ⌈The Local IP address used for a socket is specified via TcpIp_Bind().⌋()

[SWS_TcpIp_00219] ⌈If a TcpIpAddrAssignment configured with TCPIP_STORE is started, TcpIp shall check the NvMBlock (see [ECUC_TcpIp_00184]) for a valid IP address. If a valid address is present, TcpIp shall assign this address as if it was a static address. If no valid address is present, TcpIp shall start the respective IP address assignment method related to the TcpIpAddrAssignment. Once the procedure is complete, TcpIp shall store the new address in the NvMBlock.⌋(*SRS_Eth_00087*)

7.5.2 User Datagram Protocol (UDP)

[SWS_TcpIp_00060] ⌈The TcpIp shall implement the User Datagram Protocol (UDP) as defined in IETF RFC 768 (User Datagram Protocol).⌋()

[SWS_TcpIp_00103] ⌈The TcpIp shall fulfill the UDP related requirements specified by IETF RFC 1122, section 4.1.3.1 (Ports), 4.1.3.4 (UDP Checksums), and 4.1.3.6 (Invalid Addresses).⌋*()*

7.5.3 Transmission Control Protocol (TCP)

[SWS_TcpIp_00373]{DRAFT} ⌈If `TcpIpTcpSackEnabled` is set to TRUE, the Selective Acknowledgement (SACK) mechanism shall be supported according to IETF RFC 2018. If enabled, the SACK option shall be sent in the TCP handshake⌋*()*

[SWS_TcpIp_00061] ⌈The TcpIp shall implement the Transmission Control Protocol (TCP) as defined in IETF RFC 793 (Transmission Control Protocol)⌋*()*

[SWS_TcpIp_00104] ⌈The TcpIp shall fulfill the TCP related requirements specified by IETF RFC 1122, section 4.2.2.3 (Window Size), 4.2.2.5 (TCP Options), 4.2.2.6 (MSS), 4.2.2.7 (Checksum), 4.2.2.9 (Initial sequence number selection), 4.2.2.10 (Simultaneous Open Attempts), 4.2.2.11 (Recovery from Old Duplicate SYN), 4.2.2.13 (Closing a Connection, excluding "half-duplex close"), 4.2.2.15 (Retransmission Timeout), 4.2.2.16 (Managing the Window), 4.2.2.17 (Probing Zero Windows), 4.2.2.18 (Passive OPEN Calls), 4.2.2.19 (TTL), 4.2.3.2 (delayed ACK), 4.2.3.6 (TCP Keep Alive), and 4.2.3.10 (Remote Address Validation).⌋*()*

[SWS_TcpIp_00062] ⌈The TcpIp shall support the Window and Acknowledgment Strategies in TCP as defined in ETF RFC 1122:

- 4.2.3.1 Retransmission Timeout Calculation
- 4.2.3.2 When to Send an ACK Segment
- 4.2.3.3 When to Send a Window Update
- 4.2.3.4 When to Send Data.

⌋*()*

[SWS_TcpIp_00063] ⌈The TcpIp shall implement the Nagle Algorithm as defined in IETF RFC 1122: 4.2.3.4 When to Send Data.⌋*(SRS_Eth_00109)*

[SWS_TcpIp_00064] ⌈The TcpIp shall implement the congestion control strategies slow-start, congestion avoidance, fast retransmit and fast recovery as defined in IETF RFC 5681.⌋*()*

[SWS_TcpIp_00168] ⌈The TcpIp shall support the specific algorithm for responding to partial acknowledgments as defined in IETF RFC 6582 (The NewReno Modification to TCP's Fast Recovery Algorithm).The modification shall only be used if the Fast Recovery strategy of IETF RFC 5681 is enabled.⌋*()*

7.5.4 Transport Layer Security (TLS)

[SWS_TcpIp_00374]{DRAFT} ⌈TcpIp shall be able to process the `BasicOCSPResponse` according to IETF RFC 6960, section 4.2.1, sent by a TLS server in the CertificateStatus handshake message.⌋*()*

[SWS_TcpIp_00375]{DRAFT} ⌈If the parameter `TcpIpTlsRootCertUpdateCalloutFunction` is enabled, the callout function, provided in it's value and defined in the header file given by `TcpIpTlsRootCertUpdateCalloutHeaderFile`, shall be called once a new, valid, root certificate is received during TLS handshake.⌋*()*

[SWS_TcpIp_00379]{DRAFT} ⌈If `TcpIpTlsUseExtensionMaxFragmentLength` is set to TRUE then the max_fragment_length extension shall be used to negotiate the max. fragment length between TLS server and client according to IETF RFC 6066, chapter 4. If `TcpIpTlsUseExtensionRecordSizeLimit` is set to TRUE, this extension must not be sent and must be ignored on reception.⌋*()*

[SWS_TcpIp_00380]{DRAFT} ⌈If `TcpIpTlsUseExtensionTrustedCAKeys` is set to TRUE then the TLS client shall transmit the elements of `TcpIpTlsTrustedCAList` in its ClientHello message using the trusted_ca_keys extension according to IETF RFC 6066, chapter 6. Each element of `TcpIpTlsTrustedCAList` represents a KeyM root certificate and is referenced by `TcpIpTlsTrustedCAListEntry`.⌋*()*

[SWS_TcpIp_00381]{DRAFT} ⌈If `TcpIpTlsUseExtensionCertificateStatusRequest` is set to TRUE the status_request_v2 extension shall be supported according to IETF RFC 6961. The TLS Client shall transmit a CertificateStatusRequest containing the elements of `TcpIpTlsUseExtensionTrustedCAKeys` in its ClientHello message. If `TcpIpTlsUseExtensionTrustedCAKeys` contains more than one element, the CertificateStatusRequest shall be of type ocsp_multi.⌋*()*

[SWS_TcpIp_00300] ⌈TcpIp shall support the Transport Layer Security for TCP communication according to IETF RFC5246, at least chapters 7 and 8.⌋*(SRS_Eth_00138, SRS_Eth_00140)*

At least those parts from IETF RFC5246 need to be implemented that are required for a basic and compatible interoperability with other nodes without any optional extensions.

[SWS_TcpIp_00301] ⌈Further recommendation according to IETF RFC 7525 for a secure TLS implementation shall be considered.⌋*()*

[SWS_TcpIp_00302] ⌈TLS connection requests with TLS version lower than 1.2 (IETF RFC5246) shall be disregarded respectively rejected with an alert. Thus, no backward compatibility handling to TLS versions lower than TLS 1.2 as described in IETF RFC5246, App. E shall be implemented or supported.⌋*(SRS_Eth_00138)*

[SWS_TcpIp_00346] ⌈If the TLS connection references TlsCiphersuiteDefinition of type `TLS_VERSION_V13`, then TLS V1.3 shall be the preferred protocol version. Only if this fails and ciphersuites for TLS V1.2 are also assigned to the TLS connection, then a downgrade operation to TLS V1.2 shall be allowed.⌋*()*

Info: If the TLS connection does not contain ciphersuites for TLS V1.3, then the handshake shall be initiated indicating TLS V1.2 protocol.

[SWS_TcpIp_00303] ⌈Session renegotiation shall be discarded by AUTOSAR TLS implementation.⌋*()*

The KeyExchange algorithms as described in section 7.4.7 and section 8 of IETF RFC5246 depend on the ciphersuites. The necessary CSM jobs for key exchange are therefore referenced in the ciphersuite configuration.

[SWS_TcpIp_00304] ⌈If ciphersuites for TLS include support for elliptic curves then mandatory parts of IETF RFC 4492 shall be supported accordingly.⌋*(SRS_Eth_00139)*

At least, the corresponding Key Exchange algorithms according to section 2 of IETF RFC 4492 have to be implemented such as ECDHE. Extensions according to section 5 only have to be supported if certificates with respective elliptic curve parameters are expected to be used.

[SWS_TcpIp_00329] ⌈The TLS implementation must support at least one ciphersuite that corresponds to the DoIP specification ISO13400-2 so that an upper layer is able to connect such a socket to a diagnostic communication.⌋*(SRS_Eth_00140)*

[SWS_TcpIp_00305] ⌈The TLS connection shall have a configuration parameter that defines if the socket is used for TLS client or TLS server communication from the node's perspective.⌋*()*

[SWS_TcpIp_00306] ⌈A TLS connection that is used for TLS server requires a reference to a local certificate with its private key.⌋*()*

In the configuration, TLS connections can be collected in `TcpIpTlsConnectionGroup`. If one TLS connection in a group is already active, another TLS connection of the same group shall not be activated. In other words, only one TLS connection of a group shall be active at the same time. This allows to define exclusive resources for a TLS connection group and resources for TLS connections in the same group can be shared.

[SWS_TcpIp_00315] ⌈A TLS Server shall request client authentication if the selected TLS connection is configured accordingly (i.e. the config parameter `TcpIpTlsUseClientAuthenticationRequest` is set to TRUE). In this case, a local certificate with its private key is also required for a TLS client and shall be provided to the server on demand during the TLS handshake.⌋*()*

[SWS_TcpIp_00349] ⌈If `TcpIpTlsUseExtensionRecordSizeLimit` is set to TRUE then the record_size_limit extension shall be used to negotiate the max. fragment length between TLS server and client according to IETF RFC 8449, chapter 4.1.⌋*()*

The assignment of TLS connections to TCP sockets is either based on static configuration (static TLS connection assignment) or done dynamically by means of an API call (dynamic TLS connection assignment).

[SWS_TcpIp_00307] ⌈In dynamic TLS connection assignment a TLS connection shall be assigned to a TCP socket through a function call to `TcpIp_ChangeParameter`() with the ParameterId `TCPIP_PARAMID_TLS_CONNECTION_ASSIGNMENT`. The ParameterValue of the function provides a reference to a TLS connection for this socket.⌋ *()*

Note: A typical approach to dynamically assign a TLS connection to a socket is during the channel set-up before a socket connection has been established. However, it shall also be possible to perform this operation after the socket connection has been established. This might be useful starting with plain text communication and later on switching to TLS encrypted communication to accomplish for e.g. a STARTTLS operation.

[SWS_TcpIp_00337] ⌈For dynamic TLS connection assignment via `TcpIp_ChangeParameter`(), the call to `TcpIp_ChangeParameter`() shall initiate the TLS handshake as follows:

- a TLS Server shall wait for a ClientHello as the next message on this socket.
- a TLS Client shall start sending a ClientHello message.
- after that TcpIp shall no longer pass on plain messages to upper or lower layer but pass it on to TLS.

⌋ *()*

The successful completion of the TLS handshake is signaled according to [SWS_TcpIp_00345].

[SWS_TcpIp_00308] ⌈For static TLS connection assignment a port and optionally an address is defined for at least one TLS connection, TCP shall check during TCP SYN (either reception or transmission of SYN) if a port assignment is available for any TLS connection and if this TLS connection is not in use. If so, the TCP shall check the ports and automatically assign this TLS connection to the socket if a port matches.⌋ *()*

[SWS_TcpIp_00343] ⌈For static TLS connection assignment the TCP client shall check its remote port configuration when the SYN frame will be transmitted. If the TLS port configuration matches it shall assign the corresponding TLS connection to the socket.⌋ *()*

Note: This approach rules out use cases where one client uses different TLS settings (including not using TLS at all) for different local sockets when connecting to the same remote listening socket. However, having one client connecting to the same remote listening socket via different local sockets using different TLS settings is deemed an exotic use case and is thus deliberately not supported.

[SWS_TcpIp_00344] ⌈For static TLS connection assignment the TCP server shall check its local port configuration when the SYN frame is received. If the TLS port configuration matches it shall assign the corresponding TLS connection to the socket.⌋ *()*

Note: This approach rules out use cases where one server uses different TLS settings (including not using TLS at all) for different remote sockets but the same local listening socket. However, having one server using different TLS settings for different clients with the same listening socket is deemed an exotic use case and is thus deliberately not supported.

[SWS_TcpIp_00336] ⌈For static TLS connection assignment the TCP client shall initiate the TLS handshake if a TLS connection is assigned to the socket after the SYN ACK has been transmitted successfully.⌋*()*

[SWS_TcpIp_00309] ⌈For static TLS connection assignment at the TCP client the interface `<Up_TcpConnected>` shall not be called after sending the ACK of the SYN to the server. Instead, this function shall be called after the TLS handshake has been finished successfully.⌋*()*

[SWS_TcpIp_00328] ⌈For static TLS connection the TCP server shall expect a TLS handshake after the ACK for the SYN has been received. All incoming messages for this socket shall further be passed on to TLS.⌋*()*

[SWS_TcpIp_00310] ⌈For static TLS connection assignment at the TCP server side the interface `<Up_TcpAccepted>` shall not be called after the ACK has been received. Instead, this function shall be called after the TLS handshake has been finished successfully.⌋*()*

[SWS_TcpIp_00345] ⌈For both dynamic and static TLS connection assignment, the socket owner shall be informed with `<Up_TcpIpEvent>`() and the event type `TCPIP_TLS_HANDSHAKE_SUCCEEDED` if an event callback is defined for a socket owner and the TLS handshake has been finished successfully. For static TLS connection assignment the call to `<Up_TcpIpEvent>` and the event type `TCPIP_TLS_HANDSHAKE_SUCCEEDED` shall take place after the call to `<Up_TcpAccepted>`/`<Up_TcpConnected>`.⌋*()*

[SWS_TcpIp_00311] ⌈A TLS server shall select the locally assigned ciphersuite with the highest priority that matches with one of the received ciphersuites. The local certificate that was assigned to this combination of TLS connection and TLS ciphersuite shall be provided during the handshake.⌋*(SRS_Eth_00134)*

[SWS_TcpIp_00316] ⌈The TLS SERVER shall provide the certificate referenced by `TcpIpTlsConnection`/ `TcpIpTlsCipherKeyMLocalCertificate` through the server_certificate message. The certificate shall be requested from the Key Manager with the function `KeyM_GetCertificate`().⌋*()*

[SWS_TcpIp_00338] ⌈If a certificate is received with the certificate or certificateVerify handshake message of TLS it shall be provided to the Key Manager using the function `KeyM_SetCertificate`() with the reference `TcpIpTlsCipherKeyMRemoteCertificate` of `TcpIpTlsConnection`. Afterwards, the certificate is verified using the function `KeyM_VerifyCertificate`() or, if more than one certificate has been received with the handshake message, with the function `KeyM_VerifyCertificateChain`(). This function also uses the `TcpIpTlsCipherKeyMRemoteCertificate` reference.⌋*()*

The TLS module uses CSM jobs that are assigned to the ciphersuite to perform the cryptographic operations. The key material will be negotiated and loaded during the handshake.

Note: CSM jobs can run synchronously or asynchronously. If a job shall run in asynchronous or synchronous mode depends on its configuration. For asynchronous jobs a callback is needed which are not defined in this document. They are vendor specific and shall be configured accordingly in the CSM as documented.

[SWS_TcpIp_00339] ⌈TLS shall use the CSM job referenced by `TcpIpTlsCsmRandomGenerateJobRef` referenced by `TcpIpTlsHandshake` and referenced in the `TcpIpTlsConnection` to generate random values. The system outside the TLS is responsible to collect entropy to seed the RNG if needed.⌋*()*

[SWS_TcpIp_00340] ⌈After selection of the ciphersuite the assigned `TcpIpTlsHandshake` of the TLS connection will provide all necessary references to CSM jobs and keys necessary to accomplish the key exchange algorithms.⌋*()*

Info: Not all CSM jobs referenced in the `TcpIpTlsHandshake` container are required. Which of the jobs and keys configured for a TLS handshake are needed for operation mainly depends on the ciphersuite and its associated certificate. They must be preconfigured and assigned accordingly. It also depends on the TLS type if it is a TLS Server or a TLS Client, which ciphersuites are assigned to the TLS connections and which public key type is contained in the certificate, i.e. if it is an ECC or RSA public key.

The following table provides an overview of jobs and keys for CSM that needs to be configured for the handshake operation:

Job type	RSA	ECC
TcpIpTlsCsmPrfMac[Job|Key]Ref	C/S	C/S
TcpIpTlsCsmHashVerifyJobRef	C/S	C/S
TcpIpTlsCsmMasterSecretKeyRef	C/S	C/S
TcpIpTlsCsmKeyExchangeCalcPubValJobRef	-	C/S[1]
TcpIpTlsCsmKeyExchangeKeyRef	-	C/S[2]
TcpIpTlsCsmKeyExchangeCalcSecretJobRef	-	C/S[1]
TcpIpTlsCsmKeyExchangeSignatureGenerate[Job|Key]Ref	-	S/B
TcpIpTlsCsmKeyExchangeSignatureVerify[Job|Key]Ref	-	C/B
TcpIpTlsCsmKeyExchangeEncrypt[Job|Key]Ref	C/B	-
TcpIpTlsCsmKeyExchangeDecrypt[Job|Key]Ref	S/B	-

C: TLS Client implementation

S: TLS Server implementation

B: Additionally required if client authentication is activated.

[1] Reference is used for asynchronous DH(E) operation.

[2] Reference is used for synchronous DH(E) operation.

The following examples can be used as a guideline.

Example #1:

A ciphersuite that references RSA provides TcpIpTlsCsmKeyExchangeEncryptJobRef for the TLS client to encrypt the pre-master secret. First, the TLS client verifies the received certificate, will take the public key and copy it into the CSM key location referenced by TcpIpTlsCsmKeyExchangeEncryptKeyRef. Then encrypts the pre-master secret and send it to the TLS server. The Server uses TcpIpTlsCsmKeyExchangeDecryptJobRef to decrypt the pre-master secret. The job either references statically the private key or, if TcpIpTlsConnection/ TcpIpTlsCipherKeyMLocalCertificate/ KeyMCertPrivateKeyStorageCryptoKeyRef/ KeyMCryptoKeyCsmKeyTargetRef is available, copy this key into TcpIpTlsCsmKeyExchangeDecryptKeyRef.

Example #2:

A ciphersuite references ECDHE_ECDSA and the used certificate contains appropriate ECC keys, ECDSA capable in this case. The server generates DH-parameter using the crypto job Csm_KeyExchangeCalcPubVal() using the reference to TcpIpTlsCsmKeyExchangeKeyRef and signs the result using TcpIpTlsHandshake/ TcpIpTlsCsmKeyExchangeSignatureGenerateKeyRef holding a reference to the certificate private key. If the key is not statically assigned to the job it must be copied accordingly (see Example #1). The resulting data is sent to the TLS client, who verifies the certificate and uses the key of the certificate to verify the provided ECDSA signature from the server using TcpIpTlsHandshake/ TcpIpTlsCsmKeyExchangeSignatureVerifyKeyRef. Afterwards, if successful, calculates its own DH parameter and provides this to the server. Both, TLS client and server will then calculate the pre-master secret using Csm_KeyExchangeCalcSecret().

Example #3:

The selected ciphersuite defines a pre-shared key according to IETF RFC 4279. The server provides the psk_identity_hint in the ServerKeyExchange message. This can either be derived from the TcpIpTlsPskIdentity/ TcpIpTlsPresharedKeyIdentityHint or, if not specified, it can be queried from the user callback TcpIpTlsPskGetKeyIdentityHintFunc. The TLS client uses the hint to select a pre-shared key that is known by both the TLS Client and this TLS Server. If one key can uniquely be identified with the identity hint, then the TcpIpTlsPskIdentity configuration can be used as an alternative to the callback functions. In this case, the selected key can be determined by TcpIpTlsPresharedKeyIdentityHint and the TcpIpTlsPresharedKeyIdentity with TcpIpTlsPresharedKeyCsmKeyRef can be used further. A more flexible solution provides the usage of the callback TcpIpTlsPskGet-

ClientKeyIdentityFunc that allows the selection of a key with its identity at runtime. After the key and its identity has been selected on the client side, the psk_identity will be provided back to the TLS server through the ClientKeyExchange message. On the TLS server side, the corresponding key can be identified in the same way, either through the static configuration of TcpIpTlsPskIdentity/ TcpIpTlsPresharedKeyIdentity or can be queried through a callback function determined by TcpIpTlsPskGetServerKeyIdentityFunc on server side. After the key has been selected, the master secret can be determined with the corresponding CSM jobs that are allocated in the TcpIpTlsHandshake container.

[SWS_TcpIp_00341] ⌈TLS shall use TcpIpTlsHandshake /TcpIpTlsCsmHashVerifyJobRef to calculate the hash over the handshake messages which is provided with the finish handshake message.⌋*()*

[SWS_TcpIp_00347] ⌈TLS shall use TcpIpTlsCsmPrfMacJobRef to calculate the master secret. The configuration item TcpIpTlsCsmPRFSupportType shall specify how the CSM job supports the generation of the master secret.⌋*()*

If TcpIpTlsCsmPRFSupportType is set to TLS_PRF_CSM_NO_SUPPORT then TcpIpTlsCsmPrfMacJobRef references a job for MAC generation. If it is set to TLS_PRF_CSM_INOUT_REDIRECT_SUPPORT, then the re-direction support mentioned below shall be used. If the configuration is set to TLS_PRF_CSM_FULL_SUPPORT then the CSM job will generate the master secret completely on its own. The TLS just need to call the job and the master secret will be available in the element ID #1 of TcpIpTlsCsmMasterSecretKeyRef. A key distribution to the worker jobs must be done in any case.

It is recommended to use input and output re-direction for the TcpIpTlsCsmPrfMacJobRef, that was introduced in CSM with AUTOSAR V4.4. This allows to leave the master secret and intermediate results of the calculation within the crypto driver (e.g. in HSM). The key elements of TcpIpTlsCsmPrfMacKeyRef is used for input and TcpIpTlsCsmMasterSecretKeyRef as output reference for this job. Csm_KeyElementSet() is used for initial value settings, Csm_KeyCopy() and Csm_KeyElementCopyPartial() are used to set-up the input values for the job operation. Csm_KeyElementCopyPartial() is finally used to distribute the master secret results to the TcpIpTlsCiphersuiteWorker key references that are used by the worker jobs during application data transmission.

[SWS_TcpIp_00312] ⌈If TcpIpTlsServerNameIdentification is configured for a TLS connection the configured name shall be added to the Client Hello message as the server name identification (SNI).⌋*()*

[SWS_TcpIp_00313] ⌈If a TLS server receives a ClientHello message that contains a server name identification with length greater than 0 the server shall search in TcpIpTlsCertificateIdentity for a matching identity reference and shall provide the certificate that is located in this container during the handshake.⌋*()*

[SWS_TcpIp_00314] ⌈The time stamp information that is contained in the ClientHello message shall be provided through the configured `TcpIpTlsConnectionGetTimeFunc` callout function.⌋*()*

[SWS_TcpIp_00325] ⌈If a ciphersuite is used for pre-shared keys and `TcpIpTlsUsePresharedKeys` is set to TRUE, callback functions shall provide the necessary information on the TLS client and the TLS server side to select the pre-shared keys according to IETF RFC 4279. The callbacks are used to provide the identity hint and eventually the key identification during the handshake. The callback functions are used to select the CSM key that is used for further processing. Alternatively, if callback functions are not configured, the static parameter configuration from `TcpIpTlsPskIdentity` can be used.⌋*(SRS_Eth_00141, SRS_Eth_00137)*

[SWS_TcpIp_00326] ⌈TLS shall be able to open and maintain a maximum number of connections as defined in `TcpIpTlsMaxConnections`.⌋*(SRS_Eth_00135)*

[SWS_TcpIp_00327] ⌈TCP data streams shall be segmented by TLS into fragments. The maximum size of a fragment shall be used as configured in `TcpIpTlsMaxFragmentLength`. A TCP socket must be able to transmit at least such a fragment within one segment.⌋*(SRS_Eth_00136)*

[SWS_TcpIp_00348] ⌈On reception of a TLS "close_notify" message the TLS connection shall be closed and all security related resources shall be destroyed. It shall not be possible to perform further plain text communication through TCP on this socket after the TLS connection was closed. Thus, it is recommended to close the TCP socket, too.⌋*()*

7.5.5 Dynamic Host Configuration Protocol

[SWS_TcpIp_00200] ⌈The server part of the Dynamic Host Configuration Protocol shall be pre compile time configurable ON/OFF by the configuration parameter `TcpIpDhcpServerEnabled` (see [ECUC_TcpIp_00183])⌋*(SRS_Eth_00088)*

[SWS_TcpIp_00201] ⌈The server part of the Dynamic Host Configuration Protocol shall respond to client requests by assigning an available IP address according to the DHCP server configuration for the related `TcpIpCtrl`.⌋*(SRS_Eth_00087)*

[SWS_TcpIp_00218] ⌈If the configuration contains `TcpIpDhcpAddressAssignment` that refer to specific ports of an Ethernet Switch, DHCP server shall identify the port the request was received from, by calling `EthIf_GetPortMacAddr`() with the MAC address of the DHCP client and choose an available IP address of the `TcpIpDhcpAddressAssignment` related to the same port.⌋*(SRS_Eth_00087)*

7.5.5.1 Dynamic Host Configuration Protocol (DHCPv4)

[SWS_TcpIp_00058] ⌈The TcpIp shall implement the client and the server part of the Dynamic Host Configuration Protocol (DHCPv4) for the dynamic configuration of

IPv4 addresses as defined in IETF RFC 2131 (Dynamic Host Configuration Protocol).⌋
(SRS_Eth_00087, SRS_Eth_00088)

[SWS_TcpIp_00152] ⌈The TcpIp shall support the Fully Qualified Domain Name Option for Dynamic Host Configuration Protocol for IPv4 Client requirements as defined in IETF RFC 4702 (The Dynamic Host Configuration Protocol for IPv4 (DHCPv4) Client Fully Qualified Domain Name (FQDN) Option). No DNS shall be supported. Only section 2 The Client FQDN Option and section 3 DHCP Client Behavior shall be supported. Sub-Section 3.2, 3.3, 3.5 shall not be supported.⌋*()*

7.5.5.2 Dynamic Host Configuration Protocol (DHCPv6)

[SWS_TcpIp_00166] ⌈The TcpIp shall support the client part of the Dynamic Host Configuration Protocol for IPv6 (DHCPv6) which enables DHCP servers to pass configuration parameters such as IPv6 network addresses to IPv6 nodes as defined in IETF RFC 3315 (Dynamic Host Configuration Protocol for IPv6 (DHCPv6)).Due to the fact that only the client functionality shall be supported, the following sections shall not be supported:

- Relay Agent Behavior
- Server Behavior
- Section 12. Management of Temporary Addresses
- Section 21. Authentication of DHCP Messages
- Section 22.5. Identity Association for Temporary Addresses Option
- Section 22.11. Authentication Option
- Section 22.14. Rapid Commit Option

⌋*()*

[SWS_TcpIp_00167] ⌈The TcpIp shall support the Fully Qualified Domain Name Option for Dynamic Host Configuration Protocol for IPv6 Client requirements as defined in IETF RFC 4704 (The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Client Fully Qualified Domain Name (FQDN) Option). No DNS shall be supported. Only section 4 DHCPv6 Client FQDN Option and section 5 DHCPv6 Client Behavior shall be supported. Sub-Section 5.1, 5.2, 5.4 shall not be supported.⌋*()*

7.6 Message Reception

[SWS_TcpIp_00169] ⌈The TcpIp IP-layer shall map received IP datagrams to an entry in the local address table (`TcpIpAddrId`).

The local address table mapping is successfully if ALL of the following conditions are fulfilled:

1. The receiving interface matches the interface assigned to the local address table entry (EthIfCtrl).
2. The destination IP address contained in the IP header matches the currently assigned IP address of the local address table entry.

All IP datagrams which cannot be mapped to an entry in the local address table shall be silently discarded. All successfully mapped IP datagrams shall be forwarded to the upper layer protocol.⌋()

[SWS_TcpIp_00359] ⌈If IPsec is has been configured, all received IP datagrams shall be mapped to a Security Policy entry and processed as below:

1. **TCPIP_IPSEC_POLICY_PROTECT:**

   The IP datagram is only forwarded to the upper layer if it contains a valid Authentication header as per IETF RFC 4302. Otherwise the IP Datagram shall be dropped and optional callback invoked.

2. **TCPIP_IPSEC_POLICY_BYPASS:**

   The IP datagram is forwarded to the upper layer without any IPSec processing.

3. **TCPIP_IPSEC_POLICY_DISCARD:**

   The IP datagram shall be dropped without any IPSec processing.

⌋()

[SWS_TcpIp_00260] ⌈All IP datagrams mapped to an IPv6 entry in the local address table, configured with the optional TcpIpLocalAddrIPv6ExtHeaderFilterRef ([ECUC_TcpIp_00200]), that contains at least one IPv6 extension header not listed in the referenced TcpIpIpV6ConfigExtHeaderFilter ([ECUC_TcpIp_00198]) shall be silently discarded. If the Ipv6 entry in the local address table is not configured with the optional TcpIpLocalAddrIPv6ExtHeaderFilterRef, then this frame shall be processed.⌋(*SRS_Eth_00111*)

[SWS_TcpIp_00170] ⌈The TcpIp UDP-layer shall map received UDP datagrams to sockets based on the destination port as contained in the UDP protocol header and the local address (TcpIpAddrId). The local address (TcpIpAddrId) matches if ANY of the following conditions is fulfilled:

- The socket is bound to the local address (TcpIpAddrId)
- The socket local address uses the wildcard "ANY" AND the socket EthIfCtrl is identical to the EthIfCtrl used in the local address (TcpIpAddrId)
- The socket is bound to TCPIP_LOCALADDRID_ANY

The socket is bound to a local address and the EthIfCtrl is identical to the EthIfCtrl used in the local address (TcpIpAddrId) and the received local address (TcpIpAddrId) is a broadcast address.⌋()

[SWS_TcpIp_00171] ⌈For received UDP datagrams where the local address (TcpIpAddrId) is a broadcast or multicast address, all matching sockets shall receive the incoming message.⌋*()*

Note: A socket may either be explicitly bound to a local IP address by using TcpIp_Bind() or implicitly as part of TcpIp_UdpTransmit() (if it is called without a previous call of TcpIp_Bind()).

[SWS_TcpIp_00172] ⌈The TcpIp TCP-layer shall map received TCP datagrams to sockets based on the destination port as contained in the TCP protocol header and the local address (TcpIpAddrId). The local address (TcpIpAddrId) matches if ANY of the following conditions is fulfilled:

- The socket is bound to a unicast local address (TcpIpAddrId)
- The socket local address uses the wildcard "ANY" AND the socket EthIfCtrl is identical to the used in the local address (TcpIpAddrId)
- The socket is bound to TCPIP_LOCALADDRID_ANY

⌋*()*

[SWS_TcpIp_00173] ⌈Sockets with established TCP connections shall match source port, source IP address, destination port and destination IP address as contained in the protocol headers additionally to the generic TCP mapping criteria described in [SWS_TcpIp_00172].⌋*()*

[SWS_TcpIp_00174] ⌈Received TCP datagrams where the local address (TcpIpAddrId) is a broadcast or multicast address, shall be silently discarded.⌋*()*

[SWS_TcpIp_00266] ⌈If the filtering of TCP options has been enabled on a socket via TcpIp_ChangeParameter(), the TcpIp shall check received segments against the allowed list of options ([ECUC_TcpIp_00202] TcpIpTcpConfigOptionFilter) and if it contains at least one TCP option not listed the segment shall be silently discarded.⌋ *(SRS_Eth_00111)*

[SWS_TcpIp_00203] ⌈For receptions the TcpIp Module shall ignore the protocol checksum fields of frames with respect to the configuration of the Ethernet Controller according to the following list:

- for IPv4 frames if IPv4 checksum verification in hardware is enabled, i.e. EthCtrlEnableOffloadChecksumIPv4 is set to TRUE
- for ICMP frames if ICMP checksum verification in hardware is enabled, i.e. EthCtrlEnableOffloadChecksumICMP is set to TRUE
- for TCP frames if TCP checksum verification in hardware is enabled, i.e. EthCtrlEnableOffloadChecksumTCP is set to TRUE
- for UDP frames if UDP checksum verification in hardware is enabled, i.e. EthCtrlEnableOffloadChecksumUDP is set to TRUE

In all other cases, the TcpIp module shall treat frames with mismatching checksums according the related protocol specification.⌋*()*

[SWS_TcpIp_00279] ⌈For receptions the TcpIp Module shall accept UDP datagrams containing a zero checksum only on sockets that have been configured accordingly (i.e. `TcpIp_ChangeParameter`() has been called with `TCPIP_PARAMID_UDP_CHECKSUM` set to FALSE).⌋*(SRS_Eth_00019)*

[SWS_TcpIp_00296] ⌈If the measurement data is enabled (see `TcpIpGetAndResetMeasurementDataApi`), TcpIp shall increment the corresponding measurement data whenever a received datagram is discarded.⌋*(SRS_Eth_00129)*

The following guidelines are recommended for TLS data handling:

- If a TCP datagram is accepted and the socket is assigned to a TLS connection, TCP should pass the data to TLS for further processing.
- If a received TLS application message was successfully processed and verified, the data contents should be passed back to TCP to further provide it to the configured upper layer. This provides full transparency of data reception to the upper layer.
- If message reception is passed on to TLS but cannot be processed, because a TLS connection has not yet been established or the message cannot be authenticated and/or decrypted correctly, the message should be dropped.
- After TLS has processed a message and all data has been consumed completey, TCP should be notified to release all related resources for this message, regardless if the message was processed successfully or not.

7.7 Message Transmission

[SWS_TcpIp_00175] ⌈If data is transmitted using a socket which is bound to an IPv4 Unicast local address (`TcpIpAddrId`) the TcpIp shall use the IP address assigned to the local address (`TcpIpAddrId`) as source IP address in the IP datagram header. The IP datagram shall be transmitted using the `EthIfCtrl` the local address (`TcpIpAddrId`) is mapped to.⌋*()*

[SWS_TcpIp_00176] ⌈If data is transmitted using an IPv4 socket which is bound to a local address (`TcpIpAddrId`) using the wildcard "`ANY`", then the TcpIp shall use the IP address of the configured local address (`TcpIpAddrId`), which is of type IPv4 Unicast and assigned to the same `EthIfCtrl`, as the bound local address (`TcpIpAddrId`) as source IP address in the IP datagram header.⌋*()*

[SWS_TcpIp_00177] ⌈If data is transmitted using an IPv4 socket which is bound to `TCPIP_LOCALADDRID_ANY`, then the TcpIp shall use the IP address of the configured local address (`TcpIpAddrId`), which is of type IPv4 Unicast and assigned to the `EthIfCtrl` in the same subnet as the destination IPv4 address as source IP ad-

dress in the IP datagram header. If no matching subnet is found the IPv4 Unicast local address (TcpIpAddrId) of EthIfCtrl = 0 is selected.⌋()

[SWS_TcpIp_00178] ⌈If data is transmitted using an IPv4 UDP socket which is bound to a local address (TcpIpAddrId) of type Multicast, then the TcpIp shall use the IP address of the configured local address (TcpIpAddrId), which is of type IPv4 Unicast and assigned to the same EthIfCtrl, as the bound local address (TcpIpAddrId) as source IP address in the IP datagram header.⌋()

[SWS_TcpIp_00179] ⌈If data is transmitted using an IPv4 UDP socket which is bound to a local address (TcpIpAddrId) of type Broadcast, then the TcpIp shall use the IP address of the configured local address (TcpIpAddrId), which is of type IPv4 Unicast and assigned to the same EthIfCtrl, as the bound local address (TcpIpAddrId) as source IP address in the IP datagram header.⌋()

[SWS_TcpIp_00180] ⌈If data is transmitted using an IPv4 UDP socket which is not bound, then the TcpIp uses the IP address of the configured local address (TcpIpAddrId), which is of type IPv4 Unicast and assigned to the EthIfCtrl in the same subnet as the destination IPv4 address as source IP address in the IP datagram header. If no matching subnet is found the IPv4 Unicast local address (TcpIpAddrId) of EthIfCtrl = 0 is selected.⌋()

[SWS_TcpIp_00181] ⌈If data is transmitted using a socket which is bound to an IPv6 Unicast local address (TcpIpAddrId) the TcpIp shall use the IP address assigned to local address (TcpIpAddrId) as source IP address in the IP datagram header. The IP datagram shall be transmitted using the EthIfCtrl the local address (TcpIpAddrId) is mapped to.⌋()

[SWS_TcpIp_00182] ⌈If data is transmitted using an IPv6 socket which is bound to a local address (TcpIpAddrId) using the wildcard "ANY", the TcpIp shall select the source IP address of the IPv6 header according to the source address selection algorithm specified in section 5 of IETF RFC 6724 (Default Address Selection for IPv6). The selection shall be limited to the configured local addresses (TcpIpAddrId) on the same EthIfCtrl as the bound local address (TcpIpAddrId) only.⌋()

[SWS_TcpIp_00183] ⌈If data is transmitted using an IPv6 socket which is bound to TCPIP_LOCALADDRID_ANY, the TcpIp shall select the interface that has a local address (TcpIpAddrId) which uses the same network prefix as the destination address. If no matching interface is found EthIfCtrl = 0 is selected. The TcpIp shall select the source IP address of the IPv6 header according to the source address selection algorithm specified in section 5 of IETF RFC 6724 (Default Address Selection for IPv6).⌋()

[SWS_TcpIp_00184] ⌈If data is transmitted using an IPv6 UDP socket which is bound to a local address (TcpIpAddrId) of type Multicast, the TcpIp - shall select the source IP address of the IPv6 header according to the source address selection algorithm specified in section 5 of IETF RFC 6724 (Default Address Selection for IPv6). The selection shall be limited to the configured local addresses (TcpIpAddrId) on the same EthIfCtrl as the bound local address (TcpIpAddrId) only.⌋()

[SWS_TcpIp_00185] ⌈If data is transmitted using an IPv6 UDP socket which is not bound, the TcpIp shall select the interface that has a local address (TcpIpAddrId) which uses the same network prefix as the destination address. If no matching interface is found EthIfCtrl = 0 is selected. The TcpIp shall select the source IP address of the IPv6 header according to the source address selection algorithm specified in section 5 of IETF RFC 6724 (Default Address Selection for IPv6).⌋*()*

[SWS_TcpIp_00101] ⌈The TcpIp shall choose the correct next hop for each datagram it sends according to IETF RFC 1122, section 3.3.1.1. (IPv4) and IETF RFC4861 section 5.2. Conceptual Sending Algorithm (IPv6).⌋*()*

[SWS_TcpIp_00131] ⌈TcpIp shall always call EthIf_Transmit() with parameter TxConfirmation set to FALSE.⌋*()*

[SWS_TcpIp_00191] ⌈If the parameter TcpIpArpPacketQueueEnabled is set to TRUE and an IPv4 packet shall be transmitted to a remote host but the related link layer address does not exist in the ARP table, the TcpIp shall start the address resolution and queue this packet according to IETF RFC 1122, section 2.3.2.2 and accept the transmission request with E_OK.⌋*()*

[SWS_TcpIp_00192] ⌈If the parameter TcpIpArpPacketQueueEnabled is set to FALSE and an IPv4 packet shall be transmitted to a remote host but the related link layer address does not exist in the ARP table, the TcpIp shall start the address resolution but reject the transmission request with E_NOT_OK.⌋*()*

[SWS_TcpIp_00193] ⌈If the parameter TcpIpNdpPacketQueueEnabled is set to TRUE and an IPv6 packet shall be transmitted to a remote host but the related link layer address does not exist in the Neighbor Cache, the TcpIp shall start the address resolution and queue this packet according to IETF RFC 4861, section 7.2.2 and accept the transmission request with E_OK.⌋*()*

[SWS_TcpIp_00194] ⌈If the parameter TcpIpNdpPacketQueueEnabled is set to FALSE and an IPv6 packet shall be transmitted to a remote host but the related link layer address does not exist in the Neighbor Cache, the TcpIp shall start the address resolution but reject the transmission request with E_NOT_OK.⌋*()*

[SWS_TcpIp_00202] ⌈After the maximum retries configured via [ECUC_TcpIp_00069] are transmitted, the timer according to

- either TCPIP_PARAMID_TCP_RETRANSMIT_TIMEOUT if provided by TcpIp_ChangeParameter() or
- TcpIpTcpRetransmissionTimeout if TCPIP_PARAMID_TCP_RETRANSMIT_TIMEOUT was not provided by TcpIp_ChangeParameter()

shall be restarted the last time before the TCP connection is closed.⌋*()*

[SWS_TcpIp_00204] ⌈For transmissions the TcpIp Module shall skip the calculation of the protocol checksums and fill the field with the value 0 for frames with respect to the configuration of the Ethernet Controller according the following list:

- for IPv4 frames if IPv4 checksum calculation in hardware is enabled, i.e. `EthCtrlEnableOffloadChecksumIPv4` is set to TRUE
- for not fragmented ICMP frames if ICMP checksum calculation in hardware is enabled, `EthCtrlEnableOffloadChecksumICMP` is set to TRUE
- for TCP frames if TCP checksum calculation in hardware is enabled, `EthCtrlEnableOffloadChecksumTCP` is set to TRUE
- for not fragmented UDP frames if UDP checksum calculation in hardware is enabled, `EthCtrlEnableOffloadChecksumUDP` is set to TRUE

In all other cases, the TcpIp module shall calculate the checksum according the related protocol specification.⌋*()*

[SWS_TcpIp_00280] ⌈For transmissions the TcpIp Module shall skip the calculation of the UDP protocol checksum and use the value zero instead, on sockets that have been configured accordingly (i.e. `TcpIp_ChangeParameter`() has been called with `TCPIP_PARAMID_UDP_CHECKSUM` set to FALSE).⌋*(SRS_Eth_00019)*

[SWS_TcpIp_00267] ⌈Per default or if `TcpIp_ChangeParameter`() with ParameterId set to `TCPIP_PARAMID_PATHMTU_ENABLE` and the value set to TRUE has been called for a socket, the maximum size for outbound datagrams from this socket shall be determined by the Path MTU discovery.⌋*(SRS_Eth_00097)*

[SWS_TcpIp_00268] ⌈If `TcpIp_ChangeParameter`() with ParameterId set to `TCPIP_PARAMID_PATHMTU_ENABLE` and the value set to FALSE has been called for a socket, the maximum size for outbound datagrams from this socket is be determined by the static configuration.⌋*(SRS_Eth_00097)*

[SWS_TcpIp_00320] ⌈If transmission is requested from upper layer to TCP and the connection is configured for TLS but the handshake has not yet been started or completed, the message transmission request shall return `E_NOT_OK`.⌋*()*

[SWS_TcpIp_00360] ⌈If IPSec is has been configured, each IP datagram to be sent by TcpIp shall be mapped to a Security Policy entry and processed as following:

- **`TCPIP_IPSEC_POLICY_PROTECT:`**

  Authentication header as per IETF RFC 4302 shall be inserted after the IP header.

- **`TCPIP_IPSEC_POLICY_BYPASS:`**

  The IP datagram is transmitted without any IPSec processing.

- **`TCPIP_IPSEC_POLICY_DISCARD:`**

  The IP datagram shall be dropped.

⌋*()*

[SWS_TcpIp_00363] ⌈If `TcpIp_IsConnectionReady`() is called and a security association is configured, the module shall:

- check if socket exists and is bound to an assigned local address.
- check if the provided remote address has a corresponding physical address.
- check if a security association is established for this socket.

If all checks are successful, the function shall return TCPIP_E_OK.⌋()

[SWS_TcpIp_00365] ⌈If TcpIp_IsConnectionReady() is called and a security association is not configured, the module shall:

- check if socket exists and is bound to an assigned local address.
- check if the provided remote address has a corresponding physical address.

If all checks are successful, the function shall return TCPIP_E_OK.⌋()

[SWS_TcpIp_00366] ⌈If TcpIp_IsConnectionReady() is called and the socket is not bound to an assigned local address, the function shall return TCPIP_E_NOT_OK.⌋()

[SWS_TcpIp_00367] ⌈If TcpIp_IsConnectionReady() is called and the provided remote address has no corresponding physical address, TcpIp shall start the address resolution (if not already started) and return TCPIP_E_PENDING.⌋()

[SWS_TcpIp_00368] ⌈If TcpIp_IsConnectionReady() is called and for the socket a security association is configured but not established:

- If the security association establishment is in progress, TcpIp shall return TCPIP_E_PENDING.
- If the security association establishment is not started and the security association allows to initiate the secure connection, TcpIp shall start establishment and return TCPIP_E_PENDING.
- If the security association establishment is not started and the security association does not allow to initiate the secure connection, TcpIp shall return TCPIP_E_NOT_OK.

⌋()

7.8 TCP/IP Stack state handling

[SWS_TcpIp_00083] ⌈The TcpIp module shall maintain a separate state for each EthIf controller used by the TcpIp module, store the latest state request and distinguish at least the following states: TCPIP_STATE_OFFLINE, TCPIP_STATE_STARTUP, TCPIP_STATE_ONLINE, TCPIP_STATE_ONHOLD, and TCPIP_STATE_SHUTDOWN.⌋()

[SWS_TcpIp_00136] ⌈The TcpIp module shall initiate according actions to achieve the requested state if the stored state request is not the active state.⌋()

[SWS_TcpIp_00084] ⌈After each transition the TcpIp module shall report the new state to EthSM via `EthSM_TcpIpModeIndication()`.⌋*()*

[SWS_TcpIp_00075] ⌈If `TCPIP_STATE_ONLINE` is requested for an EthIf controller and the current state is `TCPIP_STATE_OFFLINE` for that EthIf controller, the TcpIp module shall

- enable all IP address assignments according to the configured assignment methods (`TcpIpAssignmentMethod`) and triggers (`TcpIpAssignmentTrigger`) for that EthIf controller. (Note: If the assignment trigger is configured to `TCPIP_MANUAL` no assignment is actually performed but initiation by the upper layer enabled) and
- enter the state `TCPIP_STATE_STARTUP` for the EthIf controller.

⌋*()*

[SWS_TcpIp_00127] ⌈In case multiple IP address assignment methods are configured and a new address from an assignment method with a higher priority (1 is highest) becomes available, TcpIp shall use the new IP address and release the IP address previously assigned by an assignment method with a lower priority.⌋*()*

[SWS_TcpIp_00088] ⌈If `TCPIP_STATE_OFFLINE` is requested for an EthIf controller and the current state is `TCPIP_STATE_STARTUP` for that EthIf controller, the TcpIp module shall

- abort all ongoing IP address assignment actions appropriate and
- enter the state `TCPIP_STATE_OFFLINE` for the EthIf controller.

⌋*()*

[SWS_TcpIp_00085] ⌈If at least one IP address has been successfully assigned to an EthIf controller and the current state is `TCPIP_STATE_STARTUP` for that EthIf controller, the TcpIp module shall enter the state `TCPIP_STATE_ONLINE` for the EthIf controller.⌋*()*

Note: After successfully assignment of an IP address to the EthIf controller the upper layer module will be notified via `<Up_LocalIpAddrAssignmentChg>`() with State `TCPIP_IPADDR_STATE_ASSIGNED`.

[SWS_TcpIp_00076] ⌈If `TCPIP_STATE_ONHOLD` is requested for an EthIf controller and the current state is `TCPIP_STATE_ONLINE` for that EthIf controller, the TcpIp module shall

- notify the upper layer via `<Up_LocalIpAddrAssignmentChg>`() with State `TCPIP_IPADDR_STATE_ONHOLD` for all assigned IP addresses of the related EthIf controller, and
- deactivate the communication within the TcpIp module for the related EthIf controller, and
- enter the state `TCPIP_STATE_ONHOLD` for the EthIf controller.

⌋()

[SWS_TcpIp_00086] ⌈If TCPIP_STATE_ONLINE is requested for an EthIf controller and the current state is TCPIP_STATE_ONHOLD for that EthIf controller, the TcpIp module shall

- reactivate the communication within the TcpIp module for the related EthIf controller,
- call <Up_LocalIpAddrAssignmentChg>() with State TCPIP_IPADDR_STATE_ASSIGNED for all assigned IP addresses of the related EthIf controller, and
- enter the state TCPIP_STATE_ONLINE for the EthIf controller.

⌋()

[SWS_TcpIp_00077] ⌈If TCPIP_STATE_OFFLINE is requested or all assigned IP address have been released for an EthIf controller and the current state is TCPIP_STATE_ONLINE or TCPIP_STATE_ONHOLD for that EthIf controller, the TcpIp module shall

- call <Up_LocalIpAddrAssignmentChg>() with State TCPIP_IPADDR_STATE_UNASSIGNED for all assigned IP addresses of the related EthIf controller,
- deactivate the communication within the TcpIp module for the related EthIf controller,
- release related resources, i.e. any socket using the EthIf controller shall be closed and thereafter any IP address assigned to the EthIf controller shall be unassigned,
- in case the no EthIf controller is assigned any more, all unbound sockets shall be released as well, and
- enter the state TCPIP_STATE_SHUTDOWN for the EthIf controller.

⌋()

[SWS_TcpIp_00372] ⌈If TCPIP_STATE_ONLINE is requested and the current state of an EthIf controller is TCPIP_STATE_SHUTDOWN, then TcpIp module shall

- immediately finish releasing all related resources stated in [SWS_TcpIp_00077],
- TCP connections shall be aborted and the ones that are still in one of the states (SYN-RECEIVED, CLOSE-WAIT, FIN-WAIT-1, FIN-WAIT-2) shall transmit a RST-segment to inform a remote host as soon as possible that the connection was closed,
- enter the state TCPIP_STATE_OFFLINE for the EthIf controller without indication this state to the EthSM,

- after all resources have been released and state TCPIP_STATE_OFFLINE was entered, start assigning the resources according to the requirement [SWS_TcpIp_00075].

⌋()

[SWS_TcpIp_00087] ⌈If the current state of an EthIf controller is TCPIP_STATE_SHUTDOWN and all related resources have been released, the TcpIp module shall enter the state TCPIP_STATE_OFFLINE for the EthIf controller.⌋()

[SWS_TcpIp_00094] ⌈The TcpIp module shall only accept new TCP connections if the related EthIf controller is in state TCPIP_STATE_ONLINE.⌋()

[SWS_TcpIp_00144] ⌈The TcpIp module shall indicate events related to sockets to the upper layer module by using the <Up_TcpIpEvent> API and the following events: TCPIP_TCP_RESET, TCPIP_TCP_CLOSED, TCPIP_TCP_FIN_RECEIVED and TCPIP_UDP_CLOSED.⌋()

7.9 Error Classification

Section "Error Handling" of the document [5] "General Specification of Basic Software Modules" describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

Based on this foundation, the following section specifies particular errors arranged in the respective subsections below.

7.9.1 Development Errors

The following table lists development error IDs the Tcp/Ip shall use for reporting of development errors to the Default Error Tracer:

[SWS_TCPIP_00042] Definiton of development errors in module TcpIp ⌈

Type of error	*Related error code*	*Error value*
API service called before initializing the module	TCPIP_E_UNINIT	0x01
API service called with NULL pointer	TCPIP_E_PARAM_POINTER	0x02
Invalid argument	TCPIP_E_INV_ARG	0x03
No buffer space available	TCPIP_E_NOBUFS	0x04
Message too long	TCPIP_E_MSGSIZE	0x07
Protocol wrong type for socket	TCPIP_E_PROTOTYPE	0x08
Address already in use	TCPIP_E_ADDRINUSE	0x09
Can't assign requested address	TCPIP_E_ADDRNOTAVAIL	0x0A
Socket is already connected	TCPIP_E_ISCONN	0x0B

▽

△

Type of error	Related error code	Error value
Socket is not connected	TCPIP_E_NOTCONN	0x0C
Protocol not available	TCPIP_E_NOPROTOOPT	0x0D
Address family not supported by protocol family	TCPIP_E_AFNOSUPPORT	0x0E
Invalid configuration set selection	TCPIP_E_INIT_FAILED	0x0F

⌋()

7.9.2 Runtime Errors

The following table lists runtime error IDs the Tcp/Ip shall use for reporting of runtime errors to the Default Error Tracer:

[SWS_TCPIP_00255] Definiton of runtime errors in module TcpIp ⌈

Type of error	Related error code	Error value
Operation timed out	TCPIP_E_TIMEDOUT	0x01
Connection refused	TCPIP_E_CONNREFUSED	0x02
No route to host	TCPIP_E_HOSTUNREACH	0x03
Path does not support frame size	TCPIP_E_PACKETTOBIG	0x04
Duplicate IP Address detected	TCPIP_E_DADCONFLICT	0x05

⌋(*SRS_Eth_00112*)

[SWS_TcpIp_00256] ⌈The TcpIp shall report the runtime error by calling `Det_ReportRuntimeError(TCPIP_E_TIMEDOUT)` if one of the following conditions applies:

- TcpIp module has sent a SYN to establish a connection but did not receive any response.
- An established idle TCP connection is closed because the peer is no longer present, i.e. keep-alive timer runs out and peer does not respond to keep-alive probes according to IETF RFC 1122 chapter 4.2.3.6 TCP Keep-Alives.
- An established TCP connection is closed because the peer does not respond, i.e. the maximum number of retransmissions has been sent without acknowledgement, according to [SWS_TcpIp_00202].

⌋(*SRS_Eth_00112*)

[SWS_TcpIp_00257] ⌈The TcpIp shall report the runtime error by calling `Det_ReportRuntimeError(TCPIP_E_CONNREFUSED)` if one of the following conditions applies:

- An ICMP message Destination Unreachable/Protocol Unreachable is received because the peer doesn't provide a service at the requested protocol.
- An ICMP message Destination Unreachable/Port Unreachable is received because the peer doesn't provide a service at the requested port.

⌋(*SRS_Eth_00112*)

[SWS_TcpIp_00258] ⌈The TcpIp shall report the runtime error by calling `Det_ReportRuntimeError(TCPIP_E_HOSTUNREACH)` if one of the following conditions applies:

- An ICMP message Destination Unreachable is received because the network or host is unreachable or there is no route to the destination.

⌋(*SRS_Eth_00112*)

[SWS_TcpIp_00259] ⌈The TcpIp shall report the runtime error by calling `Det_ReportRuntimeError(TCPIP_E_PACKETTOBIG)` if one of the following conditions applies:

- An ICMP message Destination Unreachable/ Fragmentation needed but DF bit set is received because the network can't forward an oversized frame since the DF (don't fragment) Flag is set.

⌋(*SRS_Eth_00112*)

[SWS_TcpIp_00282] ⌈The TcpIp shall report the runtime error by calling `Det_ReportRuntimeError(TCPIP_E_DADCONFLICT)` if one of the following conditions applies:

- A duplicate IP address was found by the Duplicate Address Detection (DAD) algorithm.

⌋(*SRS_Eth_00091*, *SRS_BSW_00452*)

7.9.3 Transient Faults

There are no transient faults.

7.9.4 Production Errors

There are no production errors.

7.9.5 Extended Production Errors

There are no extended production errors.

7.10 Version checking

For details refer to the chapter 5.1.8 "Version Check" in SWS_BSWGeneral.

7.11 Security Events

[SWS_TcpIp_00361] ⌈If security event reporting has been enabled for the TcpIp module (`TcpIpEnableSecurityEventReporting` = true) the respective security events shall be reported to the IdsM via the interfaces defined in AUTOSAR_SWS_BSWGeneral.⌋*(RS_Ids_00810)*

The following table lists the security events which are standardized for the TcpIp module together with their trigger conditions.

[SWS_TcpIp_00362] Security events for TcpIp ⌈

Name	*Description*	*ID*
SEV_ARP_IP_ADDR_CONFLICT	Received local IP address in ARP reply for different MAC.	10
SEV_TCP_DROP_INV_PORT	Dropped TCP packet because of invalid destination TCP-Port.	11
SEV_UDP_DROP_INV_PORT	Dropped UDP packet because of invalid destination UDP-Port.	12
SEV_IPV4_DROP_INV_ADDR	Dropped datagram because of invalid IPV4 address.	13
SEV_IPV6_DROP_INV_ADDR	Dropped datagram because of invalid IPV6 address.	14

⌋*(RS_Ids_00810)*

8 API specification

8.1 Imported types

In this chapter all types included from the following files are listed.

[SWS_TCPIP_00008] Definition of imported datatypes of module TcpIp ⌈

Module	*Header File*	*Imported Type*
ComStack_Types	ComStack_Types.h	BufReq_ReturnType
Csm	Rte_Csm_Type.h	Crypto_OperationModeType
	Rte_Csm_Type.h	Crypto_VerifyResultType
Dem	Rte_Dem_Type.h	Dem_EventIdType
	Rte_Dem_Type.h	Dem_EventStatusType
Eth	Eth_GeneralTypes.h	Eth_BufIdxType
	Eth_GeneralTypes.h	Eth_FilterActionType
	Eth_GeneralTypes.h	Eth_FrameType
IdsM	IdsM_Types.h	IdsM_SecurityEventIdType
KeyM	KeyM.h	KeyM_CertDataType
	Rte_KeyM_Type.h	KeyM_CertificateIdType
Std	Std_Types.h	Std_ReturnType
	Std_Types.h	Std_VersionInfoType

⌋()

8.2 Type definitions

[SWS_TCPIP_00067] Definition of datatype TcpIp_ConfigType ⌈

Name	TcpIp_ConfigType	
Kind	Structure	
Elements	implementation specific	
	Type	–
	Comment	The content of the configuration data structure is implementation specific.
Description	Configuration data structure of the TcpIp module.	
Available via	TcpIp.h	

⌋()

[SWS_TCPIP_00009] Definition of datatype TcpIp_DomainType ⌈

Name	TcpIp_DomainType
Kind	Type
Derived from	uint16

▽

△

Range	TCPIP_AF_INET	0x02	Use IPv4
	TCPIP_AF_INET6	0x1c	Use IPv6
Description	TcpIp address families.		
Available via	TcpIp.h		

⌋()

[SWS_TCPIP_00010] Definition of datatype TcpIp_ProtocolType ⌈

Name	TcpIp_ProtocolType		
Kind	Enumeration		
Range	TCPIP_IPPROTO_TCP	0x06	Use TCP
	TCPIP_IPPROTO_UDP	0x11	Use UDP
Description	Protocol type used by a socket.		
Available via	TcpIp.h		

⌋()

[SWS_TCPIP_00012] Definition of datatype TcpIp_SockAddrType ⌈

Name	TcpIp_SockAddrType	
Kind	Structure	
Elements	domain	
	Type	TcpIp_DomainType
	Comment	This is the code for the address format of this address
Description	Generic structure used by APIs to specify an IP address. (A specific address type can be derived from this structure via a cast to the specific struct type.)	
Available via	TcpIp.h	

⌋()

[SWS_TCPIP_00013] Definition of datatype TcpIp_SockAddrInetType ⌈

Name	TcpIp_SockAddrInetType	
Kind	Structure	
Elements	domain	
	Type	TcpIp_DomainType
	Comment	This is the code for the address format of this address
	port	
	Type	uint16
	Comment	port number
	addr	
	Type	Array of uint32
	Size	1
	Comment	IPv4 address in network byte order
Description	This structure defines an IPv4 address type which can be derived from the generic address structure via cast.	
Available via	TcpIp.h	

⌋()

[SWS_TCPIP_00014] Definition of datatype TcpIp_SockAddrInet6Type ⌈

Name	TcpIp_SockAddrInet6Type	
Kind	Structure	
Elements	domain	
	Type	TcpIp_DomainType
	Comment	This is the code for the address format of this address
	port	
	Type	uint16
	Comment	port number
	addr	
	Type	Array of uint32
	Size	4
	Comment	IPv6 address in network byte order
Description	This structure defines a IPv6 address type which can be derived from the generic address structure via cast.	
Available via	TcpIp.h	

⌋()

[SWS_TCPIP_00030] Definition of datatype TcpIp_LocalAddrIdType ⌈

Name	TcpIp_LocalAddrIdType
Kind	Type
Derived from	uint8
Description	Address identification type for unique identification of a local IP address and EthIf Controller configured in the TcpIp module.
Available via	TcpIp.h

⌋()

[SWS_TCPIP_00038] Definition of datatype TcpIp_SocketIdType ⌈

Name	TcpIp_SocketIdType	
Kind	Type	
Derived from	***Basetype***	***Variation***
	uint16	–
	uint8	–
Description	Socket identifier type for unique identification of a TcpIp stack socket. TCPIP_SOCKETID_ INVALID shall specify an invalid socket handle.	
Available via	TcpIp.h	

⌋()

[SWS_TCPIP_00073] Definition of datatype TcpIp_StateType ⌈

Name	TcpIp_StateType		
Kind	Enumeration		
Range	TCPIP_STATE_ONLINE	–	TCP/IP stack state for a specific EthIf controller is ONLINE, i.e. communication via at least one IP address is possible.
	TCPIP_STATE_ONHOLD	–	TCP/IP stack state for a specific EthIf controller is ONHOLD, i.e. no communication is currently possible (e.g. link down).
	TCPIP_STATE_OFFLINE	–	TCP/IP stack state for a specific EthIf controller is OFFLINE, i.e. no communication is possible.
	TCPIP_STATE_STARTUP	–	TCP/IP stack state for a specific EthIf controller is STARTUP, i.e. IP address assignment in progress or ready for manual start, communication is currently not possible.
	TCPIP_STATE_ SHUTDOWN	–	TCP/IP stack state for a specific EthIf controller is SHUTDOWN, i.e. release of resources using the EthIf controller, release of IP address assignment.
Description	Specifies the TcpIp state for a specific EthIf controller.		
Available via	TcpIp.h		

⌋()

[SWS_TCPIP_00082] Definition of datatype TcpIp_IpAddrStateType ⌈

Name	TcpIp_IpAddrStateType		
Kind	Enumeration		
Range	TCPIP_IPADDR_STATE_ ASSIGNED	–	local IP address is assigned
	TCPIP_IPADDR_STATE_ ONHOLD	–	local IP address is assigned, but cannot be used as the network is not active
	TCPIP_IPADDR_STATE_ UNASSIGNED	–	local IP address is unassigned
Description	Specifies the state of local IP address assignment		
Available via	TcpIp.h		

⌋()

[SWS_TCPIP_00031] Definition of datatype TcpIp_EventType ⌈

Name	TcpIp_EventType		
Kind	Enumeration		
Range	TCPIP_TCP_RESET	0x01	TCP connection was reset, TCP socket and all related resources have been released.
	TCPIP_TCP_CLOSED	0x02	TCP connection was closed successfully, TCP socket and all related resources have been released.
	TCPIP_TCP_FIN_ RECEIVED	0x03	A FIN signal was received on the TCP connection, TCP socket is still valid.
	TCPIP_UDP_CLOSED	0x04	UDP socket and all related resources have been released.

▽

△

	TCPIP_TLS_HANDSHAKE_SUCCEEDED	0x05	TLS handshake successfully established, TLS connection available.
Description	Events reported by TcpIp.		
Available via	TcpIp.h		

⌋()

[SWS_TCPIP_00065] Definition of datatype TcpIp_IpAddrAssignmentType ⌈

Name	TcpIp_IpAddrAssignmentType		
Kind	Enumeration		
Range	TCPIP_IPADDR_ASSIGNMENT_STATIC	–	Static configured IPv4/IPv6 address.
	TCPIP_IPADDR_ASSIGNMENT_LINKLOCAL_DOIP	–	Linklocal IPv4/IPv6 address assignment using DoIP parameters.
	TCPIP_IPADDR_ASSIGNMENT_DHCP	–	Dynamic configured IPv4/IPv6 address by DHCP.
	TCPIP_IPADDR_ASSIGNMENT_LINKLOCAL	–	Linklocal IPv4/IPv6 address assignment.
	TCPIP_IPADDR_ASSIGNMENT_IPV6_ROUTER	–	Dynamic configured IPv4/IPv6 address by Router Advertisement.
	TCPIP_IPADDR_ASSIGNMENT_ALL	–	All configured TcpIpAssignmentMethods with TcpIpAssignmentTrigger set to TCPIP_MANUAL
Description	Specification of IPv4/IPv6 address assignment policy.		
Available via	TcpIp.h		

⌋()

[SWS_TCPIP_00066] Definition of datatype TcpIp_ReturnType ⌈

Name	TcpIp_ReturnType		
Kind	Enumeration		
Range	TCPIP_E_OK	–	operation completed successfully.
	TCPIP_E_NOT_OK	–	operation failed.
	TCPIP_E_PHYS_ADDR_MISS	–	operation failed because of an ARP/NDP cache miss.
	TCPIP_E_PENDING	–	operation in progress
Description	TcpIp specific return type.		
Available via	TcpIp.h		

⌋()

[SWS_TCPIP_00126] Definition of datatype TcpIp_ParamIdType ⌈

Name	TcpIp_ParamIdType		
Kind	Type		
Derived from	uint8		
Range	TCPIP_PARAMID_TCP_RXWND_MAX	0x00	Specifies the maximum TCP receive window for the socket. [uint16]
	TCPIP_PARAMID_FRAMEPRIO	0x01	Specifies the frame priority for outgoing frames on the socket. [uint8]
	TCPIP_PARAMID_TCP_NAGLE	0x02	Specifies if the Nagle Algorithm according to IETF RFC 1122 (chapter 4.2.3.4 When to Send Data) is enabled or not. [boolean]
	TCPIP_PARAMID_TCP_KEEPALIVE	0x03	Specifies if TCP Keep Alive Probes are sent on the socket connection. [boolean]
	TCPIP_PARAMID_TTL	0x04	Specifies the time to live value for outgoing frames on the socket. For IPv6 this parameter specifies the value of the HopLimit field used in the IPv6 header. [uint8]
	TCPIP_PARAMID_TCP_KEEPALIVE_TIME	0x05	Specifies the time in [s] between the last data packet sent (simple ACKs are not considered data) and the first keepalive probe. [uint32]
	TCPIP_PARAMID_TCP_KEEPALIVE_PROBES_MAX	0x06	Specifies the maximum number of times that a keepalive probe is retransmitted. [uint16]
	TCPIP_PARAMID_TCP_KEEPALIVE_INTERVAL	0x07	Specifies the interval in [s] between subsequent keepalive probes. [uint32]
	TCPIP_PARAMID_TCP_OPTIONFILTER	0x08	Specifies which TCP option filter shall be applied on the related socket. [uint8]
	TCPIP_PARAMID_PATHMTU_ENABLE	0x09	Specifies if the Path MTU Discovery shall be performed on the related socket. [boolean]
	TCPIP_PARAMID_FLOWLABEL	0x0a	The 20-bit Flow Label according to IETF RFC 6437. [uint32]
	TCPIP_PARAMID_DSCP	0x0b	The 6-bit Differentiated Service Code Point according to IETF RFC 2474. [uint8]
	TCPIP_PARAMID_UDP_CHECKSUM	0x0c	0x0c Specifies if UDP checksum handling shall be enabled (TRUE) or skipped (FALSE) on the related socket. [boolean]
	TCPIP_PARAMID_TLS_CONNECTION_ASSIGNMENT	0x0d	0x0d is used to assign a TLS connection reference to a TCP socket.
	TCPIP_PARAMID_TCP_RETRANSMIT_TIMEOUT	0x0e	TCP Retransmission timeout before a unacknowledged segment is retransmitted (overrides TcpIpTcp RetransmissionTimeout (ECUC_TcpIp_00068))

▽

△

	TCPIP_PARAMID_VENDOR_SPECIFIC	0x80	Start of vendor specific range of parameter IDs. [vendor specific]
Description	Type for the specification of all supported Parameter IDs and their data types.		
Available via	TcpIp.h		

⌋()

[SWS_TCPIP_91004] Definition of datatype TcpIp_ArpCacheEntryType ⌈

Name	TcpIp_ArpCacheEntryType	
Kind	Structure	
Elements	InetAddr	
	Type	Array of uint32
	Size	1
	Comment	IPv4 address in network byte order
	PhysAddr	
	Type	Array of uint8
	Size	6
	Comment	physical address in network byte order
	State	
	Type	uint8
	Comment	state of the address entry (TCPIP_ARP_ENTRY_STATIC, TCPIP_ARP_ENTRY_VALID, TCPIP_ARP_ENTRY_STALE)
Description	TcpIp_ArpCacheEntries elements type	
Available via	TcpIp.h	

⌋()

[SWS_TCPIP_91003] Definition of datatype TcpIp_NdpCacheEntryType ⌈

Name	TcpIp_NdpCacheEntryType	
Kind	Structure	
Elements	Inet6Addr	
	Type	Array of uint32
	Size	4
	Comment	IPv6 address in network byte order
	PhysAddr	
	Type	Array of uint8
	Size	6
	Comment	physical address in network byte order
	State	
	Type	uint8
	Comment	state of the address entry (TCPIP_NDP_ENTRY_STATIC, TCPIP_NDP_ENTRY_VALID, TCPIP_NDP_ENTRY_STALE)
Description	TcpIp_NdpCacheEntries elements type	
Available via	TcpIp.h	

⌋()

[SWS_TCPIP_91010] Definition of datatype TcpIp_MeasurementIdxType ⌈

Name	TcpIp_MeasurementIdxType		
Kind	Type		
Derived from	uint8		
Range	TCPIP_MEAS_DROP_TCP	0x01	Measurement index of dropped PDUs caused by invalid destination TCP-Port
	TCPIP_MEAS_DROP_UDP	0x02	Measurement index of dropped PDUs caused by invalid destination UDP-Port
	TCPIP_MEAS_DROP_IPV4	0x03	Measurement index of dropped datagrams caused by invalid IPv4 address
	TCPIP_MEAS_DROP_IPV6	0x04	Measurement index of dropped datagrams caused by invalid IPv6 address
	TCPIP_MEAS_ RESERVED_1	0x05-0x7F	reserved by AUTOSAR
	TCPIP_MEAS_ RESERVED_2	0x80-0xEF	Vendor specific range
	TCPIP_MEAS_ RESERVED_3	0xF0-0xFE	reserved by AUTOSAR (future use)
	TCPIP_MEAS_ALL	0xFF	represents all measurement indexes
Description	Index to select specific measurement data		
Available via	TcpIp.h		

⌋()

[SWS_TCPIP_91011] Definition of datatype TcpIp_TlsConnectionIdType ⌈

Name	TcpIp_TlsConnectionIdType	
Kind	Type	
Derived from	***Basetype***	***Variation***
	uint16	–
	uint8	–
Description	TLS connection identifier type for unique identification of a TLS connection. TCPIP_ TLSCONNECTIONID_INVALID shall specify an invalid TLS connection handle.	
Available via	TcpIp.h	

⌋()

8.3 Symbol definitions

[SWS_TCPIP_00133] Definition of symbol TCPIP_IPADDR_ANY ⌈

Name	TCPIP_IPADDR_ANY
Kind	Symbol
Base Type	uint32

▽

△

Value	implementation specific, defines the value used as wildcard
Description	IP address wildcard.
Available via	TcpIp.

⌋()

[SWS_TCPIP_00132] Definition of symbol TCPIP_IP6ADDR_ANY ⌈

Name	TCPIP_IP6ADDR_ANY
Kind	Symbol
Base Type	uint32
Value	implementation specific, defines the value used as wildcard for all IP6 address parts
Description	IP6 address wildcard.
Available via	TcpIp.h

⌋()

[SWS_TCPIP_00134] Definition of symbol TCPIP_PORT_ANY ⌈

Name	TCPIP_PORT_ANY
Kind	Symbol
Base Type	uint16
Value	Zero (0) is used as wildcard
Description	Port wildcard.
Available via	TcpIp.h

⌋()

[SWS_TCPIP_00135] Definition of symbol TCPIP_LOCALADDRID_ANY ⌈

Name	TCPIP_LOCALADDRID_ANY
Kind	Symbol
Base Type	TcpIp_LocalAddrIdType
Value	implementation specific, defines the value used as wildcard
Description	LocalAddrId wildcard.
Available via	TcpIp.h

⌋()

8.4 Function definitions

This is a list of functions provided for upper layer modules.

8.4.1 General

8.4.1.1 TcpIp_Init

[SWS_TCPIP_00002] Definition of API function TcpIp_Init ⌈

Service Name	TcpIp_Init	

Syntax	`void TcpIp_Init (`
	`  const TcpIp_ConfigType* ConfigPtr`
	`)`	

Service ID [hex]	0x01	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	ConfigPtr	Pointer to the configuration data of the TcpIp module
Parameters (inout)	None	
Parameters (out)	None	
Return value	void	None
Description	This service initializes the TCP/IP Stack. TcpIp_Init may not block the start-up process for an indefinite amount of time. Caveats: The call of this service is mandatory before using the TcpIp instance for further processing.	
Available via	TcpIp.h	

⌋()

8.4.1.2 TcpIp_GetVersionInfo

[SWS_TCPIP_00004] Definition of API function TcpIp_GetVersionInfo ⌈

Service Name	TcpIp_GetVersionInfo	

Syntax	`void TcpIp_GetVersionInfo (`
	`  Std_VersionInfoType* versioninfo`
	`)`	

Service ID [hex]	0x02	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	versioninfo	Pointer to where to store the version information of this module.
Return value	None	
Description	Returns the version information.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00005] ⌈The function `TcpIp_GetVersionInfo`() shall return the version information of this module. The version information includes:

- Module Id
- Vendor Id

- Vendor specific version numbers (BSW00407).

⌋()

[SWS_TcpIp_00006] ⌈The function TcpIp_GetVersionInfo() shall be pre compile time configurable On/Off by the configuration parameter: TCPIP_VERSION_INFO_API⌋()

8.4.2 Core Communication Control

8.4.2.1 TcpIp_Close

[SWS_TCPIP_00017] Definition of API function TcpIp_Close ⌈

Service Name	TcpIp_Close	
Syntax	`Std_ReturnType TcpIp_Close ( TcpIp_SocketIdType SocketId, boolean Abort )`	
Service ID [hex]	0x04	
Sync/Async	Asynchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket handle identifying the local socket resource.
	Abort	TRUE: connection will immediately be terminated by sending a RST-Segment and releasing all related resources. FALSE: connection will be terminated after performing a regular connection termination handshake and releasing all related resources.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted.
Description	By this API service the TCP/IP stack is requested to close the socket and release all related resources.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00109] ⌈The service TcpIp_Close() shall perform the following actions for the socket specified by SocketId in case it is a TCP socket:

1. if the connection is active and
   (a) **abort = FALSE:** the connection shall be terminated after performing a regular connection termination handshake and releasing all related resources.
   (b) **abort = TRUE:** connection shall immediately be terminated by sending a RST-Segment and releasing all related resources.
2. if the socket is in the Listen state, the Listen state shall be left immediately and related resources shall be released.

⌋()

[SWS_TcpIp_00110] ⌈The service TcpIp_Close() shall release all related resources immediately for the socket specified by SocketId in case it is a UDP socket .⌋*()*

Note: The upper layer will be notified via <Up_TcpIpEvent>(TCPIP_TCP_CLOSED, TCPIP_TCP_RESET or TCPIP_UDP_CLOSED) after the socket and all related resources have been released. After this call the SocketId is invalid until allocated again with TcpIp_GetSocket().

8.4.2.2 TcpIp_Bind

[SWS_TCPIP_00015] Definition of API function TcpIp_Bind ⌈

Service Name	TcpIp_Bind	

Syntax	`Std_ReturnType TcpIp_Bind (`
	`  TcpIp_SocketIdType SocketId,`
	`  TcpIp_LocalAddrIdType LocalAddrId,`
	`  uint16* PortPtr`
	`)`	

Service ID [hex]	0x05	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	LocalAddrId	IP address identifier representing the local IP address and EthIf controller to bind the socket to. Note: to listen to all EthIf controller, TCPIP_LOCALADDRID_ANY has to be specified as LocalAddrId. Note: to listen on any IP addresss of a EthIf controller, the configuration parameter TcpIpStaticIpAddress referenced by LocalAddrId must be set to "ANY". The remote IP address of an incoming packet has no effect then. In case the socket shall be used as client socket, the IP address and EthIf controller represented by LocalAddrId is used for transmission. Note: for an automatic selection of the Local IP address and EthIf Controller, TCPIP_LOCALADDRID_ANY has to be specified as LocalAddrId.
Parameters (inout)	PortPtr	Pointer to memory where the local port to which the socket shall be bound is specified. In case the parameter is specified as TCPIP_PORT_ANY, the TCP/IP stack shall choose the local port automatically from the range 49152 to 65535 and shall update the parameter to the chosen value.
Parameters (out)	None	
Return value	Std_ReturnType	Result of operation E_OK The request has been accepted E_NOT_OK The request has not been accepted (e.g. address in use)
Description	By this API service the TCP/IP stack is requested to bind a UDP or TCP socket to a local resource.	
Available via	TcpIp.h	

⌋*()*

[SWS_TcpIp_00111] ⌈The service TcpIp_Bind() shall bind the socket specified by parameter SocketId to the local resource specified by parameters LocalAddrId and PortPtr.⌋*()*

Note: Sockets that shall be switched in a listening state later on must be bound to a local resource. Optionally this API can be used to specify the local IP address and port used by later calls of TcpIp_TcpConnect() or TcpIp_UdpTransmit().

[SWS_TcpIp_00146] ⌈TcpIp_Bind() shall check if there is another socket already bound to the same port, protocol and local address and if that is the case refuse the request and return E_NOT_OK. If development error detection is enabled, the service TcpIp_Bind() shall also raise the development error code TCPIP_E_ADDRINUSE.⌋*()*

[SWS_TcpIp_00147] ⌈If development error detection is enabled: TcpIp_Bind() shall check if the parameter LocalAddrId is valid. If the check fails, TcpIp_Bind() shall refuse the request and raise the development error code TCPIP_E_ADDRNOTAVAIL instead.⌋*(SRS_BSW_00323)*

[SWS_TcpIp_00254] ⌈TcpIp_Bind() shall check if the local address specified by LocalAddrId is assigned and if that is not the case refuse the request and return E_NOT_OK⌋*(SRS_Eth_00045)*

8.4.2.3 TcpIp_TcpConnect

[SWS_TCPIP_00022] Definition of API function TcpIp_TcpConnect ⌈

Service Name	TcpIp_TcpConnect	
Syntax	`Std_ReturnType TcpIp_TcpConnect ( TcpIp_SocketIdType SocketId, const TcpIp_SockAddrType* RemoteAddrPtr )`	
Service ID [hex]	0x06	
Sync/Async	Asynchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	RemoteAddrPtr	IP address and port of the remote host to connect to.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted, e.g. connection is already established or no route to destination specified by remoteAddrPtr found.
Description	By this API service the TCP/IP stack is requested to establish a TCP connection to the configured peer.	
Available via	TcpIp.h	

⌋*()*

[SWS_TcpIp_00112] ⌈The service TcpIp_TcpConnect() shall establish a TCP connection between the local socket specified by parameter SocketId and the remote socket specified with parameter RemoteAddrPtr.⌋*()*

[SWS_TcpIp_00129] ⌈If development error detection is enabled and the parameter RemoteAddrPtr equals NULL_PTR, the TcpIp_TcpConnect() function shall raise the development error code TCPIP_E_PARAM_POINTER.⌋*()*

8.4.2.4 TcpIp_TcpListen

[SWS_TCPIP_00023] Definition of API function TcpIp_TcpListen ⌈

Service Name	TcpIp_TcpListen	
Syntax	`Std_ReturnType TcpIp_TcpListen ( TcpIp_SocketIdType SocketId, uint16 MaxChannels )`	
Service ID [hex]	0x07	
Sync/Async	Asynchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	MaxChannels	Maximum number of new parallel connections established on this listen connection.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted, the socket is not configured to be a server socket.
Description	By this API service the TCP/IP stack is requested to listen on the TCP socket specified by the socket identifier.	
Available via	TcpIp.h	

⌋*()*

[SWS_TcpIp_00113] ⌈The service TcpIp_TcpListen() shall put the socket specified by SocketId to the listen state (i.e. local socket is listening for incoming connections).⌋ *()*

[SWS_TcpIp_00114] ⌈TcpIp shall derive a separate socket from the listen socket to establish a new connection from an incoming connection request on the listen socket and limit the number of new parallel connections to the value specified by MaxChannels.⌋*()*

8.4.2.5 TcpIp_TcpReceived

[SWS_TCPIP_00024] Definition of API function TcpIp_TcpReceived ⌈

Service Name	TcpIp_TcpReceived	

Syntax	`Std_ReturnType TcpIp_TcpReceived (`
	`  TcpIp_SocketIdType SocketId,`
	`  uint32 Length`
	`)`	

Service ID [hex]	0x08	
Sync/Async	Asynchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	Length	Number of bytes finally consumed by the upper layer.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: The request has been accepted `E_NOT_OK`: The request has not been accepted
Description	By this API service the reception of socket data is confirmed to the TCP/IP stack.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00115] ⌈The service `TcpIp_TcpReceived`() shall increase the TCP receive window of the socket specified by `SocketId` considering the number of finally consumed bytes specified by `Length`.⌋()

8.4.2.6 TcpIp_RequestComMode

[SWS_TCPIP_00070] Definition of API function TcpIp_RequestComMode ⌈

Service Name	TcpIp_RequestComMode	

Syntax	`Std_ReturnType TcpIp_RequestComMode (`
	`  uint8 CtrlIdx,`
	`  TcpIp_StateType State`
	`)`	

Service ID [hex]	0x09	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	EthIf controller index to identify the communication network where the TcpIp state is requested.
	State	Requested TcpIp state.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: Service accepted `E_NOT_OK`: Service denied
Description	By this API service the TCP/IP stack is requested to change the TcpIp state of the communication network identified by EthIf controller index.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00071] ⌈If TCPIP_STATE_ONLINE is requested, the TcpIp module shall initiate activation of the TcpIp communication on the related EthIf controller (e.g. start IP-Address assignment according to the configured IP address assignment policy for the EthIf controller).⌋*()*

[SWS_TcpIp_00072] ⌈If TCPIP_STATE_OFFLINE is requested, the TcpIp module shall initiate deactivation of the TcpIp communication on the related EthIf controller (e.g. close all sockets using the specified EthIf controller).⌋*()*

[SWS_TcpIp_00074] ⌈If TCPIP_STATE_ONHOLD is requested, the TcpIp module shall set the TcpIp communication to on hold, i.e. new transmit requests shall not be accepted, but sockets and assigned IP addresses shall be kept.⌋*()*

[SWS_TcpIp_00089] ⌈If TCPIP_STATE_STARTUP or TCPIP_STATE_SHUTDOWN is requested as state the function TcpIp_RequestComMode() shall abort with E_NOT_OK and report TCPIP_E_INV_ARG if development error detection is enabled.⌋*()*

Note: According to [SWS_TcpIp_00075] and [SWS_TcpIp_00077] TCPIP_STATE_STARTUP or TCPIP_STATE_SHUTDOWN are intermediate states arising from requesting TCPIP_STATE_OFFLINE or TCPIP_STATE_ONLINE. Requesting these intermediate states is not useful.

8.4.3 Extended Communication Control and Information

8.4.3.1 TcpIp_RequestIpAddrAssignment

[SWS_TCPIP_00037] Definition of API function TcpIp_RequestIpAddrAssignment ⌈

Service Name	TcpIp_RequestIpAddrAssignment	

Syntax	`Std_ReturnType TcpIp_RequestIpAddrAssignment (`
	`  TcpIp_LocalAddrIdType LocalAddrId,`
	`  TcpIp_IpAddrAssignmentType Type,`
	`  const TcpIp_SockAddrType* LocalIpAddrPtr,`
	`  uint8 Netmask,`
	`  const TcpIp_SockAddrType* DefaultRouterPtr`
	`)`	

Service ID [hex]	0x0A	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	LocalAddrId	IP address index specifying the IP address for which an assignment shall be initiated.
	Type	Type of IP address assignment which shall be initiated
	LocalIpAddrPtr	Pointer to structure containing the IP address which shall be assigned to the EthIf controller indirectly specified via LocalAddr Id. Note: This parameter is only used in case the parameter Type is set to TCPIP_IPADDR_ASSIGNMENT_STATIC, can be set to NULL_PTR otherwise.

▽

△

	Netmask	Network mask of IPv4 address or address prefix of IPv6 address in CIDR Notation. Note: This parameter is only used in case the parameter Type is set to TCPIP_IPADDR_ASSIGNMENT_STATIC.
	DefaultRouterPtr	Pointer to structure containing the IP address of the default router (gateway) which shall be assigned. Note: This parameter is only used in case the parameter Type is set to TCPIP_IPADDR_ASSIGNMENT_STATIC, can be set to NULL_PTR otherwise.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted
Description	By this API service the local IP address assignment for the IP address specified by LocalAddrId shall be initiated.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00116] ⌈The service TcpIp_RequestIpAddrAssignment() shall initiate the local IP address assignment according to the IP address table entry specified by LocalAddrId using the method specified by Type.⌋()

[SWS_TcpIp_00079] ⌈In case TcpIp_RequestIpAddrAssignment() is called with parameter Type set to TCPIP_IPADDR_ASSIGNMENT_STATIC and no TcpIpStaticIpAddressConfig container is configured for the TcpIpLocalAddr specified by parameter LocalAddrId, TcpIp shall assign the IP address, netmask and default router specified by parameter LocalIpAddrPtr, Netmask and DefaultRouterPtr as soon as TCPIP_STATE_ONLINE is requested or immediately if already requested.⌋()

[SWS_TcpIp_00080] ⌈In case a multicast address is assigned, TcpIp shall derive the related physical address from the multicast IP address and add the derived address to the Eth MAC address filter by calling EthIf_UpdatePhysAddrFilter() with action set to ETH_ADD_TO_FILTER.⌋()

[SWS_TcpIp_00299] ⌈In case TcpIp_RequestIpAddrAssignment() is called with parameter Type set to TCPIP_IPADDR_ASSIGNMENT_ALL, the IP address assignment for the IP address table entry specified by LocalAddrId shall be initiated for all configured TcpIpAssignmentMethod with TcpIpAssignmentTrigger set to TCPIP_MANUAL.⌋()

[SWS_TcpIp_00195] ⌈If TcpIp_RequestIpAddrAssignment() is called for a LocalAddrId configured with TcpIpAssignmentTrigger set to TCPIP_MANUAL, TcpIp shall consider the related assignment as available.⌋()

[SWS_TcpIp_00196] ⌈If TcpIp_ReleaseIpAddrAssignment is called for a LocalAddrId configured with TcpIpAssignmentTrigger set to TCPIP_MANUAL, TcpIp shall consider the related assignment as unavailable.⌋()

[SWS_TcpIp_00197] ⌈TcpIpAddrAssignment configured with TcpIpAssignmentTrigger set to TCPIP_AUTOMATIC shall always be available.⌋()

[SWS_TcpIp_00198] ⌈If TcpIp_RequestIpAddrAssignment() is called for a LocalAddrId configured with TcpIpAssignmentTrigger set to TCPIP_AUTOMATIC, TcpIp shall reject the request and return E_NOT_OK.⌋*()*

[SWS_TcpIp_00199] ⌈If TcpIp_ReleaseIpAddrAssignment() is called for a LocalAddrId configured with TcpIpAssignmentTrigger set to TCPIP_AUTOMATIC, TcpIp shall reject the request and return E_NOT_OK.⌋*()*

8.4.3.2 TcpIp_ReleaseIpAddrAssignment

[SWS_TCPIP_00078] Definition of API function TcpIp_ReleaseIpAddrAssignment ⌈

Service Name	TcpIp_ReleaseIpAddrAssignment	
Syntax	`Std_ReturnType TcpIp_ReleaseIpAddrAssignment ( TcpIp_LocalAddrIdType LocalAddrId )`	
Service ID [hex]	0x0B	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	LocalAddrId	IP address index specifying the IP address for which an assignment shall be released.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted
Description	By this API service the local IP address assignment for the IP address specified by LocalAddrId shall be released.	
Available via	TcpIp.h	

⌋*()*

[SWS_TcpIp_00117] ⌈The service TcpIp_ReleaseIpAddrAssignment() shall release the local IP address assignment related to the IP address table entry specified by LocalAddrId.⌋*()*

8.4.3.3 TcpIp_ResetIpAssignment

[SWS_TCPIP_00215] Definition of API function TcpIp_ResetIpAssignment ⌈

Service Name	TcpIp_ResetIpAssignment
Syntax	`Std_ReturnType TcpIp_ResetIpAssignment ( void )`
Service ID [hex]	0x1b
Sync/Async	Synchronous

▽

△

Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: success `E_NOT_OK`: switch port could not be initialized
Description	Resets all learned IP-addresses to invalid values.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00216] ⌈The service `TcpIp_ResetIpAssignment`() shall reset all persistently stored IP addresses in the NvMBlock (see [[ECUC_TcpIp_00184]]) to invalid values (e.g. to `0.0.0.0` for IPv4 addresses).⌋*(SRS_Eth_00087)*

Note: The next time the `TcpIpAddrAssignment` configured with `TCPIP_STORE` are started, the related address assignment method are started to obtain new IP addresses.

[SWS_TcpIp_00217] ⌈The service `TcpIp_ResetIpAssignment`() shall be pre compile time configurable On/Off by the configuration parameter: `TcpIpResetIpAssignmentApi` (see [ECUC_TcpIp_00182]).⌋*(SRS_Eth_00087)*

8.4.3.4 TcpIp_IcmpTransmit

[SWS_TCPIP_00039] Definition of API function TcpIp_IcmpTransmit ⌈

Service Name	TcpIp_IcmpTransmit	

Syntax	`Std_ReturnType TcpIp_IcmpTransmit (`
	`  TcpIp_LocalAddrIdType LocalIpAddrId,`
	`  const TcpIp_SockAddrType* RemoteAddrPtr,`
	`  uint8 Ttl,`
	`  uint8 Type,`
	`  uint8 Code,`
	`  uint16 DataLength,`
	`  const uint8* DataPtr`
	`)`	

Service ID [hex]	0x0C	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	LocalIpAddrId	IP address identifier representing the local IP address and EthIf controller which shall be used for transmission of the ICMP message.
	RemoteAddrPtr	pointer to struct representing the remote address
	Ttl	Time to live value to be used for the ICMP message. If 0 is specified the default value shall be used.
	Type	type field value to be used in the ICMP message (Note: the value of the type field determines the format of the remaining ICMP message data)

▽

△

	Code	code field value to be used in the ICMP message
	DataLength	length of ICMP message
	DataPtr	Pointer to data which shall be sent as ICMP message data
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	Result of operation E_OK The ICMP message has been sent successfully E_NOT_OK The ICMP message was not sent.
Description	By this API service the TCP/IP stack sends an ICMP message according to the specified parameters.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00118] ⌈The service TcpIp_IcmpTransmit() shall (a) construct an ICMP message according to the parameters Type, Code, DataLength and DataPtr and (b) transmit the ICMP message using the local IP address and EthIf controller specified by LocalIpAddrId to the destination specified by RemoteAddrPtr using a time to live value according to the parameter Ttl.⌋()

8.4.3.5 TcpIp_IcmpV6Transmit

[SWS_TCPIP_00187] Definition of API function TcpIp_IcmpV6Transmit ⌈

Service Name	TcpIp_IcmpV6Transmit	

Syntax	`Std_ReturnType TcpIp_IcmpV6Transmit (`
	`  TcpIp_LocalAddrIdType LocalIpAddrId,`
	`  const TcpIp_SockAddrType* RemoteAddrPtr,`
	`  uint8 HopLimit,`
	`  uint8 Type,`
	`  uint8 Code,`
	`  uint16 DataLength,`
	`  const uint8* DataPtr`
	`)`	

Service ID [hex]	0x18	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	LocalIpAddrId	IP address identifier representing the local IP address and EthIf controller which shall be used for transmission of the ICMPv6 message.
	RemoteAddrPtr	pointer to struct representing the remote address
	HopLimit	Hop Limit value to be used for the ICMPv6 message. If 0 is specified the default value shall be used.
	Type	type field value to be used in the ICMPv6 message. (Note: the value of the type field determines the format of the remaining ICMPv6 message data)
	Code	code field value to be used in the ICMPv6 message
	DataLength	length of ICMPv6 message
	DataPtr	Pointer to data which shall be sent as ICMPv6 message data

▽

△

Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	Result of operation E_OK: The ICMPv6 message has been sent successfully E_NOT_OK: The ICMPv6 message was not sent.
Description	By this API service the TCP/IP stack sends an ICMPv6 message according to the specified parameters.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00230] ⌈The service TcpIp_IcmpV6Transmit() shall (a) construct an ICMPv6 message according to the parameters Type, Code, DataLength and DataPtr and (b) transmit the ICMPv6 message using the local IP address and EthIf controller specified by LocalIpAddrId to the destination specified by RemoteAddrPtr using a Hop Limit value according to the parameter HopLimit.⌋()

8.4.3.6 TcpIp_DhcpReadOption

[SWS_TCPIP_00040] Definition of API function TcpIp_DhcpReadOption ⌈

Service Name	TcpIp_DhcpReadOption	

Syntax	`Std_ReturnType TcpIp_DhcpReadOption (`
	`  TcpIp_LocalAddrIdType LocalIpAddrId,`
	`  uint8 Option,`
	`  uint8* DataLength,`
	`  uint8* DataPtr`
	`)`	

Service ID [hex]	0x0D	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	LocalIpAddrId	IP address identifier representing the local IP address and EthIf controller for which the DHCP option shall be read.
	Option	DHCP option (note: according to IANA DHCP Options)
Parameters (inout)	DataLength	As input parameter, contains the length of the provided data buffer. Will be overwritten with the length of the actual data.
Parameters (out)	DataPtr	Pointer to memory containing DHCP option data
Return value	Std_ReturnType	Result of operation E_OK requested data retrieved successfully. E_NOT_OK requested data could not be retrieved.
Description	By this API service the TCP/IP stack retrieves DHCP option data identified by parameter option for already received DHCP options.	
Available via	TcpIp.h	

⌋(*SRS_Eth_00066*)

[SWS_TcpIp_00233] ⌈If development error detection is enabled: TcpIp_DhcpReadOption() shall check if the parameter LocalIpAddrId is valid. If the check fails, TcpIp_DhcpReadOption() shall raise the development error TCPIP_E_INV_ARG.⌋ (*SRS_Eth_00066*)

[SWS_TcpIp_00234] ⌈If development error detection is enabled: TcpIp_DhcpReadOption() shall check if the parameter Option is valid. If the check fails, TcpIp_DhcpReadOption() shall raise the development error TCPIP_E_INV_ARG.⌋*(SRS_Eth_00066)*

[SWS_TcpIp_00235] ⌈If development error detection is enabled: TcpIp_DhcpReadOption() shall check if the parameter DataLength is valid (i.e. the buffer is large enough for the requested option). If the check fails, TcpIp_DhcpReadOption() shall raise the development error TCPIP_E_INV_ARG.⌋*(SRS_Eth_00066)*

[SWS_TcpIp_00236] ⌈If the requested option has been set for the address specified by LocalIpAddrId, TcpIp_DhcpReadOption() shall copy this option into the buffer provided by DataPtr, set the parameter DataLength to the length of the option and return E_OK.⌋*(SRS_Eth_00066)*

[SWS_TcpIp_00237] ⌈If the requested option has not been set for the address specified by LocalIpAddrId, TcpIp_DhcpReadOption() shall set the parameter DataLength to zero, leave the buffer provided by DataPtr unchanged and return E_OK.⌋*(SRS_Eth_00066)*

8.4.3.7 TcpIp_DhcpV6ReadOption

[SWS_TCPIP_00189] Definition of API function TcpIp_DhcpV6ReadOption ⌈

Service Name	TcpIp_DhcpV6ReadOption	
Syntax	`Std_ReturnType TcpIp_DhcpV6ReadOption ( TcpIp_LocalAddrIdType LocalIpAddrId, uint16 Option, uint16* DataLength, uint8* DataPtr )`	
Service ID [hex]	0x19	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	LocalIpAddrId	IP address identifier representing the local IP address and EthIf controller for which the DHCPv6 option shall be read.
	Option	DHCP option (note: according to IANA DHCP[v6] Options)
Parameters (inout)	DataLength	As input parameter, contains the length of the provided data buffer. Will be overwritten with the length of the actual data.
Parameters (out)	DataPtr	Pointer to memory containing DHCPv6 option data
Return value	Std_ReturnType	Result of operation E_OK: requested data retrieved successfully. E_NOT_OK: requested data could not be retrieved.
Description	By this API service the TCP/IP stack retrieves DHCPv6 option data identified by parameter option for already received DHCPv6 options.	
Available via	TcpIp.h	

⌋*(SRS_Eth_00066)*

[SWS_TcpIp_00238] ⌈If development error detection is enabled: TcpIp_DhcpV6ReadOption() shall check if the parameter LocalIpAddrId is valid. If

the check fails, TcpIp_DhcpV6ReadOption() shall raise the development error TCPIP_E_INV_ARG.⌋(*SRS_Eth_00066*)

[SWS_TcpIp_00239] ⌈If development error detection is enabled: TcpIp_DhcpV6ReadOption() shall check if the parameter Option is valid. If the check fails, TcpIp_DhcpV6ReadOption() shall raise the development error TCPIP_E_INV_ARG.⌋(*SRS_Eth_00066*)

[SWS_TcpIp_00240] ⌈If development error detection is enabled: TcpIp_DhcpV6ReadOption() shall check if the parameter DataLength is valid (i.e. the buffer is large enough for the requested option). If the check fails, TcpIp_DhcpV6ReadOption() shall raise the development error TCPIP_E_INV_ARG.⌋(*SRS_Eth_00066*)

[SWS_TcpIp_00241] ⌈If the requested option has been set for the address specified by LocalIpAddrId, TcpIp_DhcpV6ReadOption() shall copy this option into the buffer provided by DataPtr, set the parameter DataLength to the length of the option and return E_OK.⌋(*SRS_Eth_00066*)

[SWS_TcpIp_00242] ⌈If the requested option has not been set for the address specified by LocalIpAddrId, TcpIp_DhcpV6ReadOption() shall set the parameter DataLength to zero, leave the buffer provided by DataPtr unchanged and return E_OK.⌋(*SRS_Eth_00066*)

8.4.3.8 TcpIp_DhcpWriteOption

[SWS_TCPIP_00020] Definition of API function TcpIp_DhcpWriteOption ⌈

Service Name	TcpIp_DhcpWriteOption	
Syntax	`Std_ReturnType TcpIp_DhcpWriteOption ( TcpIp_LocalAddrIdType LocalIpAddrId, uint8 Option, uint8 DataLength, const uint8* DataPtr )`	
Service ID [hex]	0x0E	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	LocalIpAddrId	IP address identifier representing the local IP address and EthIf controller for which the DHCP option shall be written.
	Option	DHCP option (note: according to IANA DHCP Options)
	DataLength	length of DHCP option data
	DataPtr	Pointer to memory containing DHCP option data
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	Result of operation E_OK no error occured. E_NOT_OK DHCP option data could not be written.

▽

△

Description	By this API service the TCP/IP stack writes the DHCP option data identified by parameter option.
Available via	TcpIp.h

⌋*(SRS_Eth_00065)*

[SWS_TcpIp_00243] ⌈If development error detection is enabled: `TcpIp_DhcpWriteOption`() shall check if the parameter `LocalIpAddrId` is valid. If the check fails, `TcpIp_DhcpWriteOption`() shall raise the development error `TCPIP_E_INV_ARG`.⌋*(SRS_Eth_00065)*

[SWS_TcpIp_00244] ⌈If development error detection is enabled: `TcpIp_DhcpWriteOption`() shall check if the parameter `Option` is valid. If the check fails, `TcpIp_DhcpWriteOption`() shall raise the development error `TCPIP_E_INV_ARG`.⌋*(SRS_Eth_00065)*

[SWS_TcpIp_00245] ⌈If development error detection is enabled: `TcpIp_DhcpWriteOption`() shall check if the parameter `DataLength` is valid (i.e. the length of the provided option is not larger than supported by the protocol). If the check fails, `TcpIp_DhcpWriteOption`() shall raise the development error `TCPIP_E_INV_ARG`.⌋*(SRS_Eth_00065)*

[SWS_TcpIp_00246] ⌈If the length indicated by `DataLength` is larger than zero `TcpIp_DhcpWriteOption`() shall set the option identified by `Option` to the value provided by `DataPtr` internally for the address specified by `LocalIpAddrId` and return `E_OK`.⌋*(SRS_Eth_00065)*

[SWS_TcpIp_00247] ⌈If the length indicated by `DataLength` is equal to zero `TcpIp_DhcpWriteOption`() shall unset the option identified by `Option` for the address specified by `LocalIpAddrId` and return `E_OK`.⌋*(SRS_Eth_00065)*

8.4.3.9 TcpIp_DhcpV6WriteOption

[SWS_TCPIP_00190] Definition of API function TcpIp_DhcpV6WriteOption ⌈

Service Name	TcpIp_DhcpV6WriteOption	
Syntax	`Std_ReturnType TcpIp_DhcpV6WriteOption ( TcpIp_LocalAddrIdType LocalIpAddrId, uint16 Option, uint16 DataLength, const uint8* DataPtr )`	
Service ID [hex]	0x1a	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	LocalIpAddrId	IP address identifier representing the local IP address and EthIf controller for which the DHCPv6 option shall be written.

▽

△

	Option	DHCP option (note: according to IANA DHCP[v6] Options
	DataLength	length of DHCPv6 option data
	DataPtr	Pointer to memory containing DHCPv6 option data
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	Result of operation E_OK: no error occured. E_NOT_OK: DHCPv6 option data could not be written.
Description	By this API service the TCP/IP stack writes the DHCPv6 option data identified by parameter option.	
Available via	TcpIp.h	

⌋(*SRS_Eth_00065*)

[SWS_TcpIp_00248] ⌈If development error detection is enabled: TcpIp_DhcpV6WriteOption() shall check if the parameter LocalIpAddrId is valid. If the check fails, TcpIp_DhcpV6WriteOption() shall raise the development error TCPIP_E_INV_ARG.⌋(*SRS_Eth_00065*)

[SWS_TcpIp_00249] ⌈If development error detection is enabled: TcpIp_DhcpV6WriteOption() shall check if the parameter Option is valid. If the check fails, TcpIp_DhcpV6WriteOption() shall raise the development error TCPIP_E_INV_ARG.⌋(*SRS_Eth_00065*)

[SWS_TcpIp_00250] ⌈If development error detection is enabled: TcpIp_DhcpV6WriteOption() shall check if the parameter DataLength is valid (i.e. the length of the provided option is not larger than supported by the protocol). If the check fails, TcpIp_DhcpV6WriteOption() shall raise the development error TCPIP_E_INV_ARG.⌋(*SRS_Eth_00065*)

[SWS_TcpIp_00251] ⌈If the length indicated by DataLength is larger than zero TcpIp_DhcpV6WriteOption() shall set the option identified by Option to the value provided by DataPtr internally for the address specified by LocalIpAddrId and return E_OK.⌋(*SRS_Eth_00065*)

[SWS_TcpIp_00252] ⌈If the length indicated by DataLength is equal to zero TcpIp_DhcpV6WriteOption() shall unset the option identified by Option for the address specified by LocalIpAddrId and return E_OK.⌋(*SRS_Eth_00065*)

8.4.3.10 TcpIp_ChangeParameter

[SWS_TCPIP_00016] Definition of API function TcpIp_ChangeParameter ⌈

Service Name	TcpIp_ChangeParameter	
Syntax	`Std_ReturnType TcpIp_ChangeParameter ( TcpIp_SocketIdType SocketId, TcpIp_ParamIdType ParameterId, const uint8* ParameterValue )`	
Service ID [hex]	0x0F	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	ParameterId	Identifier of the parameter to be changed
	ParameterValue	Pointer to memory containing the new parameter value
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: The parameter has been changed successfully. E_NOT_OK: The parameter could not be changed.
Description	By this API service the TCP/IP stack is requested to change a parameter of a socket. E.g. the Nagle algorithm may be controlled by this API.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00119] ⌈The service `TcpIp_ChangeParameter`() shall change the parameter specified by `ParameterId` with the value (casted to the respective data type) specified by `ParameterValue` for the `SocketId`.⌋()

8.4.3.11 TcpIp_GetIpAddr

[SWS_TCPIP_00032] Definition of API function TcpIp_GetIpAddr ⌈

Service Name	TcpIp_GetIpAddr	
Syntax	`Std_ReturnType TcpIp_GetIpAddr ( TcpIp_LocalAddrIdType LocalAddrId, TcpIp_SockAddrType* IpAddrPtr, uint8* NetmaskPtr, TcpIp_SockAddrType* DefaultRouterPtr )`	
Service ID [hex]	0x10	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	LocalAddrId	Local address identifier referring to the local IP address which shall be obtained.

▽

△

Parameters (inout)	IpAddrPtr	Pointer to a struct where the IP address shall be stored. The struct member domain shall be set to the desired TcpIp_Domain Type and it shall be ensured that the struct is large enough to store an address of the selected type (INET or INET6). Struct members not related to the IP address are of arbitrary value and shall not be used.
	DefaultRouterPtr	Pointer to struct where the IP address of the default router (gateway) is stored (struct member "port" is not used and of arbitrary value). The struct must be of the same type and size as IpAddrPtr.
Parameters (out)	NetmaskPtr	Pointer to memory where Network mask of IPv4 address or address prefix of IPv6 address in CIDR Notation is stored
Return value	Std_ReturnType	Result of operation E_OK: The request was successful E_NOT_OK: The request was not successful, e.g. domain in Ip AddrPtr and the local domain type do not match
Description	Obtains the local IP address actually used by LocalAddrId, the netmask and default router	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00205] ⌈TcpIp_GetIpAddr() shall refuse the request if the domain set in IpAddrPtr does not match the TcpIp_DomainType of the selected local address and return E_NOT_OK. If development error detection is enabled, the service TcpIp_GetIpAddr() shall also raise the development error TCPIP_E_INV_ARG.⌋()

[SWS_TcpIp_00206] ⌈TcpIp_GetIpAddr() shall refuse the request if the domain set in IpAddrPtr does not match the domain set in DefaultRouterPtr and return E_NOT_OK. If development error detection is enabled, the service TcpIp_GetIpAddr() shall also raise the development error TCPIP_E_INV_ARG.⌋()

8.4.3.12 TcpIp_GetPhysAddr

[SWS_TCPIP_00033] Definition of API function TcpIp_GetPhysAddr ⌈

Service Name	TcpIp_GetPhysAddr	
Syntax	`Std_ReturnType TcpIp_GetPhysAddr ( TcpIp_LocalAddrIdType LocalAddrId, uint8* PhysAddrPtr )`	
Service ID [hex]	0x11	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	LocalAddrId	Local address identifier implicitely specifing the EthIf controller for which the physical address shall be obtained.
Parameters (inout)	None	
Parameters (out)	PhysAddrPtr	Pointer to the memory where the physical source address (MAC address) in network byte order is stored

▽

△

Return value	Std_ReturnType	Result of operation E_OK The request was successful E_NOT_OK The request was not successful, e.g. no unique Ctrl specified via IpAddrId.
Description	Obtains the physical source address used by the EthIf controller implicitly specified via Local AddrId.	
Available via	TcpIp.h	

⌋()

8.4.3.13 TcpIp_GetRemotePhysAddr

[SWS_TCPIP_00137] Definition of API function TcpIp_GetRemotePhysAddr ⌈

Service Name	TcpIp_GetRemotePhysAddr	
Syntax	`TcpIp_ReturnType TcpIp_GetRemotePhysAddr ( uint8 CtrlIdx, const TcpIp_SockAddrType* IpAddrPtr, uint8* PhysAddrPtr, boolean initRes )`	
Service ID [hex]	0x16	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	EthIf controller index to identify the related ARP/NDP table.
	IpAddrPtr	specifies the IP address for which the physical address shall be retrieved
	initRes	specifies if the address resolution shall be initiated (TRUE) or not (FALSE) in case the physical address related to the specified IP address is currently unknown.
Parameters (inout)	None	
Parameters (out)	PhysAddrPtr	Pointer to the memory where the physical address (MAC address) related to the specified IP address is stored in network byte order.
Return value	TcpIp_ReturnType	TCPIP_E_OK: specified IP address resolved, physical address provided via PhysAddrPtr TCPIP_E_PHYS_ADDR_MISS: physical address currently unknown (address resolution initiated if initRes set to TRUE)
Description	TcpIp_GetRemotePhysAddr queries the IP/physical address translation table specified by Ctrl Idx and returns the physical address related to the IP address specified by IpAddrPtr. In case no physical address can be retrieved and parameter initRes is TRUE, address resolution for the specified IP address is initiated on the local network.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00138] ⌈TcpIp_GetRemotePhysAddr() shall lookup the physical address for the IP address specified by IpAddrPtr at the IP/physical address translation table related to the controller identified by CtrlIdx.

1. If the physical address is already known, PhysAddrPtr shall be set to the related physical address and the function shall return with TCPIP_E_OK.
2. Otherwise it shall

(a) initiate an address resolution if parameter initRes is set to TRUE and

(b) return with TCPIP_E_PHYS_ADDR_MISS.

PhysAddrPtr is not updated in this case.⌋()

[SWS_TcpIp_00139] ⌈TcpIp_GetRemotePhysAddr() shall immediately return with TCPIP_E_NOT_OK if it is called with an IP address that is not part of the same sub network as the local address currently assigned to the controller identified by CtrlIdx.⌋ ()

8.4.3.14 TcpIp_GetCtrlIdx

[SWS_TCPIP_00140] Definition of API function TcpIp_GetCtrlIdx ⌈

Service Name	TcpIp_GetCtrlIdx	
Syntax	`Std_ReturnType TcpIp_GetCtrlIdx ( TcpIp_LocalAddrIdType LocalAddrId, uint8* CtrlIdxPtr )`	
Service ID [hex]	0x17	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	LocalAddrId	Local address identifier implicitely specifing the EthIf controller that shall be returned.
Parameters (inout)	None	
Parameters (out)	CtrlIdxPtr	Pointer to the memory where the index of the controller related to LocalAddrId is stored
Return value	Std_ReturnType	Result of operation E_OK the request was successful E_NOT_OK the request was not successful.
Description	TcpIp_GetCtrlIdx returns the index of the controller related to LocalAddrId.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00141] ⌈TcpIp_GetCtrlIdx() shall return the index of the controller related to LocalAddrId.⌋ ()

8.4.3.15 TcpIp_GetArpCacheEntries

[SWS_TCPIP_91002] Definition of API function TcpIp_GetArpCacheEntries ⌈

Service Name	TcpIp_GetArpCacheEntries	
Syntax	`Std_ReturnType TcpIp_GetArpCacheEntries ( uint8 ctrlIdx, uint32* numberOfElements, TcpIp_ArpCacheEntryType* entryListPtr )`	
Service ID [hex]	0x1d	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	ctrlIdx	EthIf controller index to identify the related ARP table.
Parameters (inout)	numberOfElements	In: Maximum number of entries that can be stored in output entry ListPtr. Out: Number of entries written to output entryListPtr (Number of all entries in the cache if input value is 0).
Parameters (out)	entryListPtr	Pointer to memory where the list of cache entries shall be stored.
Return value	Std_ReturnType	E_OK: physical address cache could be read. E_NOT_OK: physical address cache could not be read (i.e. no IPv4 instance active on this controller)
Description	Copies entries from the physical address cache of the IPv4 instance that is active on the EthIf controller specified by ctrlIdx into a user provided buffer. The function will copy all or numberOf Elements into the output list. If input value of numberOfElements is 0 the function will not copy any data but only return the number of valid entries in the cache. EntryListPtr may be NULL_ PTR in this case.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00271] ⌈TcpIp_GetArpCacheEntries() shall only consider entryListPtr set to NULL_PTR as valid if numberOfElements is set to zero.⌋()

[SWS_TcpIp_00272] ⌈If TcpIp_GetArpCacheEntries() is called with numberOfElements set to zero, TcpIp shall set the parameter numberOfElements to the number of valid entries in the physical address cache related to ctrlIdx, leave the buffer provided by entryListPtr unchanged and return E_OK.⌋()

[SWS_TcpIp_00273] ⌈If the numberOfElements is greater zero, TcpIp_GetArpCacheEntries() shall copy up to that number of valid entries from the physical address cache related to ctrlIdx into the buffer provided by entryListPtr, set the parameter numberOfElements to the number of copied elements and return E_OK.⌋()

8.4.3.16 TcpIp_GetNdpCacheEntries

[SWS_TCPIP_91001] Definition of API function TcpIp_GetNdpCacheEntries ⌈

Service Name	TcpIp_GetNdpCacheEntries	
Syntax	`Std_ReturnType TcpIp_GetNdpCacheEntries ( uint8 ctrlIdx, uint32* numberOfElements, TcpIp_NdpCacheEntryType* entryListPtr )`	
Service ID [hex]	0x1c	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	ctrlIdx	EthIf controller index to identify the related NDP table.
Parameters (inout)	numberOfElements	In: Maximum number of entries that can be stored in output entry ListPtr. Out: Number of entries written to output entryListPtr (Number of all entries in the cache if input value is 0).
Parameters (out)	entryListPtr	Pointer to memory where the list of cache entries shall be stored.
Return value	Std_ReturnType	E_OK: physical address cache could be read. E_NOT_OK: physical address cache could not be read (i.e. no IPv6 instance active on this controller)
Description	Copies entries from the physical address cache of the IPv6 instance that is active on the EthIf controller specified by ctrlIdx into a user provided buffer. The function will copy all or numberOf Elements into the output list. If input value of numberOfElements is 0 the function will not copy any data but only return the number of valid entries in the cache. EntryListPtr may be NULL_ PTR in this case.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00274] ⌈TcpIp_GetNdpCacheEntries() shall only consider entryListPtr set to NULL_PTR as valid if numberOfElements is set to zero.⌋()

[SWS_TcpIp_00275] ⌈If TcpIp_GetNdpCacheEntries() is called with numberOfElements set to zero, TcpIp shall set the parameter numberOfElements to the number of valid entries in the physical address cache related to ctrlIdx, leave the buffer provided by entryListPtr unchanged and return E_OK.⌋()

[SWS_TcpIp_00276] ⌈If the numberOfElements is greater zero, TcpIp_GetNdpCacheEntries() shall copy up to that number of valid entries from the physical address cache related to ctrlIdx into the buffer provided by entryListPtr, set the parameter numberOfElements to the number of copied elements and return E_OK.⌋()

8.4.3.17 TcpIp_GetAndResetMeasurementData

[SWS_TCPIP_91006] Definition of API function TcpIp_GetAndResetMeasurement Data ⌈

Service Name	TcpIp_GetAndResetMeasurementData	
Syntax	`Std_ReturnType TcpIp_GetAndResetMeasurementData ( TcpIp_MeasurementIdxType MeasurementIdx, boolean MeasurementResetNeeded, uint32* MeasurementDataPtr )`	
Service ID [hex]	0x45	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	MeasurementIdx	Data index of measurement data
	MeasurementReset Needed	Flag to trigger a reset of the measurement data
Parameters (inout)	None	
Parameters (out)	MeasurementDataPtr	Reference to data buffer, where to copy measurement data
Return value	Std_ReturnType	`E_OK`: successful `E_NOT_OK`: failed
Description	Allows to read and reset detailed measurement data for diagnostic purposes. Get all MeasurementIdx's at once is not supported. TCPIP_MEAS_ALL shall only be used to reset all MeasurementIdx's at once. A NULL_PTR shall be provided for MeasurementDataPtr in this case.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00284] ⌈The function `TcpIp_GetAndResetMeasurementData`() shall be pre compile time configurable On/Off by the configuration parameter: `TcpIpGetAndResetMeasurementDataApi`.⌋(*SRS_Eth_00129*)

[SWS_TcpIp_00285] ⌈If development error detection is enabled: `TcpIp_GetAndResetMeasurementData`() shall check that the service `TcpIp_Init`() was previously called. If the check fails, `TcpIp_GetAndResetMeasurementData`() shall raise the development error `TCPIP_E_UNINIT`.⌋(*SRS_Eth_00129*)

[SWS_TcpIp_00295] ⌈`TcpIp_GetAndResetMeasurementData`() shall accept `MeasurementDataPtr` set to `NULL_PTR`. In this case the measurement data shall not be copied.⌋(*SRS_Eth_00129*)

[SWS_TcpIp_00286] ⌈`TcpIp_GetAndResetMeasurementData`() shall return measurement data for selected measurement index.⌋(*SRS_Eth_00129*)

[SWS_TcpIp_00287] ⌈For measurement index `TCPIP_MEAS_DROP_TCP` `TcpIp_GetAndResetMeasurementData`() shall return the number of all TCP datagrams which cannot be mapped to a valid local IP/Port.⌋(*SRS_Eth_00129*)

[SWS_TcpIp_00288] ⌈For measurement index `TCPIP_MEAS_DROP_UDP` `TcpIp_GetAndResetMeasurementData`() shall return the number of all UDP datagrams which cannot be mapped to a valid local IP/Port.⌋(*SRS_Eth_00129*)

[SWS_TcpIp_00289] ⌈For measurement index TCPIP_MEAS_DROP_IPV4 TcpIp_GetAndResetMeasurementData() shall return the number of all dropped IPv4 datagrams, caused by invalid IP address.⌋*(SRS_Eth_00129)*

[SWS_TcpIp_00290] ⌈For measurement index TCPIP_MEAS_DROP_IPV6 TcpIp_GetAndResetMeasurementData() shall return the number of all dropped IPv6 datagrams, caused by invalid IP address.⌋*(SRS_Eth_00129)*

[SWS_TcpIp_00291] ⌈TcpIp_GetAndResetMeasurementData() shall return E_NOT_OK if the requested measurement index is not supported.⌋*(SRS_Eth_00129)*

[SWS_TcpIp_00292] ⌈TcpIp_GetAndResetMeasurementData() shall additionally reset the measurement data to 0 if the MeasurementResetNeeded is true. The reset shall be applied after measurement data has been read.⌋*(SRS_Eth_00129)*

[SWS_TcpIp_00293] ⌈TcpIp_GetAndResetMeasurementData() shall reset all existing measurement data to 0, if MeasurementResetNeeded is true and measurement index is set to TCPIP_MEAS_ALL.⌋*(SRS_Eth_00129)*

[SWS_TcpIp_00294] ⌈All measurement data which counts data shall not overrun.⌋*(SRS_Eth_00129)*

8.4.3.18 TcpIp_IsConnectionReady

[SWS_TCPIP_91016] Definition of API function TcpIp_IsConnectionReady ⌈

Service Name	TcpIp_IsConnectionReady	
Syntax	`TcpIp_ReturnType TcpIp_IsConnectionReady ( TcpIp_SocketIdType SocketId, const TcpIp_SockAddrType* RemoteAddrPtr )`	
Service ID [hex]	0x46	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket handle identifying the local socket resource.
	RemoteAddrPtr	Pointer to the structure containing the requested remote IP address and port.
Parameters (inout)	None	
Parameters (out)	None	
Return value	TcpIp_ReturnType	TCPIP_E_OK - SocketId is ready for communication. TCPIP_E_NOT_OK - Request was rejected. TCPIP_E_PENDING - Connection establishment in progress.
Description	API allows to check if a communication over this socket is possible for a dedicated remote address. It includes that the socket is bound, a physical address is available for the requested remote address and if a security association is configured that a secured connection is already established.	
Available via	TcpIp.h	

⌋()

[SWS_TcpIp_00369] ⌈If development error detection is enabled and the parameter RemoteAddrPtr equals NULL_PTR, the TcpIp_IsConnectionReady() function shall raise the development error code TCPIP_E_PARAM_POINTER.⌋*()*

8.4.4 Transmission

8.4.4.1 TcpIp_UdpTransmit

[SWS_TCPIP_00025] Definition of API function TcpIp_UdpTransmit ⌈

Service Name	TcpIp_UdpTransmit	
Syntax	`Std_ReturnType TcpIp_UdpTransmit ( TcpIp_SocketIdType SocketId, const uint8* DataPtr, const TcpIp_SockAddrType* RemoteAddrPtr, uint16 TotalLength )`	
Service ID [hex]	0x12	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	DataPtr	Pointer to a linear buffer of TotalLength bytes containing the data to be transmitted. In case DataPtr is a NULL_PTR, TcpIp shall retrieve data from upper layer via callback <Up>_CopyTxData().
	RemoteAddrPtr	IP address and port of the remote host to transmit to.
	TotalLength	indicates the payload size of the UDP datagram.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request to transmit the UDP message has been accepted. E_NOT_OK: UDP message could not be sent because of a permanent error, e.g. message is too long.
Description	This service transmits data via UDP to a remote node. The transmission of the data is immediately performed with this function call by forwarding it to EthIf.	
Available via	TcpIp.h	

⌋*()*

[SWS_TcpIp_00120] ⌈With respect to [SWS_TcpIp_00191] and [SWS_TcpIp_00193], the service TcpIp_UdpTransmit() shall immediately transmit TotalLength data bytes via UDP and the socket specified by SocketId to a remote socket specified by RemoteAddrPtr.⌋*()*

Note: Transmission stated in [SWS_TcpIp_00120] is done according to the sequence diagram specified in section section 9.5.

[SWS_TcpIp_00121] ⌈DataPtr shall either point to a linear buffer of TotalLength bytes containing the data for transmission or be a NULL_PTR. For data transmission the service TcpIp_UdpTransmit() shall either use all data from the linear buffer if DataPtr is not a NULL_PTR, or retrieve TotalLength data bytes from the upper layer by calling <Up_CopyTxData>() one or multiple times in the context of this service otherwise.⌋*()*

[SWS_TcpIp_00122] ⌈The service TcpIp_UdpTransmit() shall select the local IP address and port for transmission if the socket specified by SocketId has not been bound to a local resource via a previous call to TcpIp_Bind().⌋*()*

8.4.4.2 TcpIp_TcpTransmit

[SWS_TCPIP_00050] Definition of API function TcpIp_TcpTransmit ⌈

Service Name	TcpIp_TcpTransmit	

Syntax	`Std_ReturnType TcpIp_TcpTransmit (`
	`  TcpIp_SocketIdType SocketId,`
	`  const uint8* DataPtr,`
	`  uint32 AvailableLength,`
	`  boolean ForceRetrieve`
	`)`	

Service ID [hex]	0x13	
Sync/Async	Asynchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	DataPtr	Pointer to a linear buffer of AvailableLength bytes containing the data to be transmitted. In case DataPtr is a NULL_PTR, TcpIp shall retrieve data from upper layer via callback <Up>_CopyTx Data().
	AvailableLength	Available data for transmission in bytes.
	ForceRetrieve	This parameter is only valid if DataPtr is a NULL_PTR. Indicates how the TCP/IP stack retrieves data from upper layer if DataPtr is a NULL_PTR. TRUE: the whole data indicated by availableLength shall be retrieved from the upper layer via one or multiple <Up>_ CopyTxData() calls within the context of this transmit function. FALSE: The TCP/IP stack may retrieve up to availableLength data from the upper layer. It is allowed to retrieve less than availableLength bytes. Note: Not retrieved data will be provided by upper layer with the next call to TcpIp_TcpTransmit (along with new data if available).
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted, e.g. due to a lack of buffer space or the socket is not connected.
Description	This service requests transmission of data via TCP to a remote node. The transmission of the data is decoupled. Note: The TCP segment(s) are sent dependent on runtime factors (e.g. receive window) and configuration parameter (e.g. Nagle algorithm) .	
Available via	TcpIp.h	

⌋*()*

[SWS_TcpIp_00123] ⌈The service TcpIp_TcpTransmit() shall transmit data via TCP and the socket specified by SocketId to the connected remote socket.⌋*()*

Note: Transmission stated in [SWS_TcpIp_00123] is done according to the sequence diagram specified in section section 9.4.

[SWS_TcpIp_00124] ⌈DataPtr shall either point to a linear buffer of AvailableLength bytes containing the data for transmission or be a NULL_PTR. For data trans-

mission the service TcpIp_TcpTransmit() shall either use all data from the linear buffer if DataPtr is not a NULL_PTR, or retrieve up to AvailableLength data bytes from the upper layer by calling <Up_CopyTxData>() one or multiple times in the context of this service otherwise.⌋()

[SWS_TcpIp_00125] ⌈The service TcpIp_TcpTransmit() shall retrieve exactly AvailableLength bytes from the upper layer if the parameter DataPtr is a NULL_PTR and ForceRetrieve is TRUE. (If DataPtr is a NULL_PTR and ForceRetrieve is FALSE, TcpIp may retrieve less data then available).⌋()

Note: The TCP segment(s) are sent dependent on runtime factors (e.g. receive window) and configuration parameter (e.g. Nagle algorithm).

8.5 Callback notifications

This is a list of functions provided for other modules.

8.5.1 TcpIp_RxIndication

[SWS_TCPIP_00029] Definition of API function TcpIp_RxIndication ⌈

Service Name	TcpIp_RxIndication	

Syntax	`void TcpIp_RxIndication (`
	`  uint8 CtrlIdx,`
	`  Eth_FrameType FrameType,`
	`  boolean IsBroadcast,`
	`  const uint8* PhysAddrPtr,`
	`  const uint8* DataPtr,`
	`  uint16 LenByte`
	`)`	

Service ID [hex]	0x14	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the EthIf controller.
	FrameType	frame type of received Ethernet frame
	IsBroadcast	parameter to indicate a broadcast frame
	PhysAddrPtr	pointer to Physical source address (MAC address in network byte order) of received Ethernet frame
	DataPtr	Pointer to payload of the received Ethernet frame (i.e. Ethernet header is not provided).
	LenByte	Length of received data.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	By this API service the TCP/IP stack gets an indication and the data of a received frame.	
Available via	TcpIp.h	

⌋()

8.6 Scheduled functions

These functions are directly called by Basic Software Scheduler. The following functions shall have no return value and no parameter. All functions shall be non reentrant.

8.6.1 Terms and definitions

For details refer to the chapter 8.5 "Scheduled functions" in SWS_BSWGeneral.

8.6.2 TcpIp_MainFunction

[SWS_TCPIP_00026] Definition of scheduled function TcpIp_MainFunction ⌈

Service Name	TcpIp_MainFunction

Syntax	`void TcpIp_MainFunction (`
	`  void`
	`)`

Service ID [hex]	0x15
Description	Schedules the TCP/IP stack. (Entry point for scheduling)
Available via	SchM_TcpIp.h

⌋()

8.7 Expected interfaces

In this chapter all interfaces required from other modules are listed.

8.7.1 Mandatory interfaces

Note: This section defines all interfaces, which are required to fulfill the core functionality of the module.

[SWS_TCPIP_00027] Definition of mandatory interfaces in module TcpIp ⌈

API Function	*Header File*	*Description*
Dem_SetEventStatus	Dem.h	Called by SW-Cs or BSW modules to report monitor status information to the Dem. BSW modules calling Dem_SetEventStatus can safely ignore the return value. This API will be available only if ({Dem/Dem ConfigSet/DemEventParameter/DemEvent ReportingType} == STANDARD_REPORTING)
Det_ReportRuntimeError	Det.h	Service to report runtime errors. If a callout has been configured then this callout shall be called.

▽

△

API Function	Header File	Description
EthIf_GetPhysAddr	EthIf.h	Obtains the physical source address used by the indexed controller
EthIf_ProvideTxBuffer	EthIf.h	Provides access to a transmit buffer of the specified Ethernet controller.
EthIf_SetPhysAddr	EthIf.h	Sets the physical source address used by the indexed controller.
EthIf_Transmit	EthIf.h	Triggers transmission of a previously filled transmit buffer
EthSM_TcpIpModeIndication	EthSM_TcpIp.h	This service is called by the TcpIp to report the actual TcpIp state (e.g. online, offline).

⌋()

8.7.2 Optional interfaces

This section defines all interfaces, which are required to fulfill an optional functionality of the module.

[SWS_TCPIP_00028] Definition of optional interfaces in module TcpIp ⌈

API Function	Header File	Description
Csm_AEADDecrypt	Csm.h	Uses the given data to perform an AEAD encryption and stores the ciphertext and the MAC in the memory locations pointed by the ciphertext pointer and Tag pointer.
Csm_AEADEncrypt	Csm.h	Uses the given input data to perform a AEAD encryption and stores the ciphertext and the MAC in the memory locations pointed by the ciphertext pointer and Tag pointer.
Csm_Decrypt	Csm.h	Decrypts the given encrypted data and store the decrypted plaintext in the memory location pointed by the result pointer.
Csm_Encrypt	Csm.h	Encrypts the given data and store the ciphertext in the memory location pointed by the result pointer.
Csm_Hash	Csm.h	Uses the given data to perform the hash calculation and stores the hash.
Csm_KeyElementCopy	Csm.h	This function shall copy a key elements from one key to a target key.
Csm_KeyElementCopyPartial	Csm.h	Copies a key element to another key element in the same crypto driver. The keyElementSourceOffset and keyElementCopyLength allows to copy just a part of the source key element into the destination. The offset into the target key is also specified with this function.
Csm_KeyExchangeCalcPubVal	Csm.h	Calculates the public value of the current user for the key exchange and stores the public key in the memory location pointed by the public value pointer.
Csm_KeyExchangeCalcSecret	Csm.h	Calculates the shared secret key for the key exchange with the key material of the key identified by the keyId and the partner public key. The shared secret key is stored as a key element in the same key.

▽

△

API Function	Header File	Description
Csm_MacGenerate	Csm.h	Uses the given data to perform a MAC generation and stores the MAC in the memory location pointed to by the MAC pointer.
Csm_MacVerify	Csm.h	Verifies the given MAC by comparing if the MAC is generated with the given data.
Csm_RandomGenerate	Csm.h	Generate a random number and stores it in the memory location pointed by the result pointer.
Csm_SignatureGenerate	Csm.h	Uses the given data to perform the signature calculation and stores the signature in the memory location pointed by the result pointer.
Csm_SignatureVerify	Csm.h	Verifies the given MAC by comparing if the signature is generated with the given data.
Det_ReportError	Det.h	Service to report development errors.
EthIf_UpdatePhysAddrFilter	EthIf.h	Update the physical source address to/from the indexed controller filter. If the Ethernet Controller is not capable to do the filtering, the software has to do this.
IdsM_SetSecurityEvent	IdsM.h	This API is the application interface to report security events to the IdsM.
IdsM_SetSecurityEventWithContext Data	IdsM.h	This API is the application interface to report security events with context data to the IdsM.
KeyM_GetCertificate	KeyM.h	This function provides the DER encoded certificate data
KeyM_SetCertificate	KeyM.h	This function provides the certificate data to the key management module to temporarily store the certificate.
KeyM_VerifyCertificate	KeyM.h	This function verifies a certificate that was previously provided with KeyM_SetCertificate() against already stored and provided certificates stored with other certificate IDs.
KeyM_VerifyCertificateChain	KeyM.h	This function performs a certificate verification against a list of certificates. It is a pre-requisite that the certificate that shall be checked has already been written with KeyM_SetCertificate() and that the root certificate is either in the list or is already assigned to one of the other certificates.

⌋()

8.7.3 Configurable interfaces

In this section, all interfaces are listed where the target function could be configured. The target function is usually a callback function. The names of this kind of interfaces are not fixed because they are configurable.

The `ServiceID` of the functions defined in this chapter are specified at the upper layer module implementing the functions.

8.7.3.1 TcpIp_<Up>GetSocket

[SWS_TCPIP_00018] Definition of API function TcpIp_<Up>GetSocket ⌈

Service Name	TcpIp_<Up>GetSocket	
Syntax	`Std_ReturnType TcpIp_<Up>GetSocket ( TcpIp_DomainType Domain, TcpIp_ProtocolType Protocol, TcpIp_SocketIdType* SocketIdPtr )`	
Service ID [hex]	0x03	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Domain	IP address family.
	Protocol	Socket protocol as sub-family of parameter type.
Parameters (inout)	None	
Parameters (out)	SocketIdPtr	Pointer to socket identifier representing the requested socket. This socket identifier must be provided for all further API calls which requires a SocketId. Note: SocketIdPtr is only valid if return value is E_OK.
Return value	Std_ReturnType	Result of operation E_OK The request has been accepted E_NOT_OK The request has not been accepted: no free socket
Description	By this API service the TCP/IP stack is requested to allocate a new socket. Note: Each accepted incoming TCP connection also allocates a socket resource.	
Available via	TcpIp.h	

⌋*(SRS_Eth_00103)*

[SWS_TcpIp_00128] ⌈If development error detection is enabled, the service `TcpIp_GetSocket`() shall check the parameter `Domain` for being valid and raise the development error `TCPIP_E_AFNOSUPPORT` if it is invalid.⌋*()*

[SWS_TcpIp_00222] ⌈For each configured `TcpIpSocketOwner` TcpIp shall provide a separate `TcpIp_GetSocket`() API by replacing the tag <Up> with the short name of the `TcpIpSocketOwner` container. Sockets allocated by a dedicated `TcpIp_GetSocket`() API shall be assigned exclusively to the respective upper layer.⌋*(SRS_Eth_00103)*

8.7.3.2 <Up_PhysAddrTableChg>

[SWS_TCPIP_00143] Definition of configurable interface <Up_PhysAddrTable Chg> ⌈

Service Name	<Up_PhysAddrTableChg>	

Syntax	`void <Up_PhysAddrTableChg> (`
	`  uint8 CtrlIdx,`
	`  const TcpIp_SockAddrType* IpAddrPtr,`
	`  const uint8* PhysAddrPtr,`
	`  boolean valid`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	EthIf controller index of the related ARP/NDP table.
	IpAddrPtr	specifies the IP address of the changed ARP/NDP table entry
	PhysAddrPtr	specifies the physical address of the changed ARP/NDP table entry
	valid	specifies if the ARP/NDP table entry is added or changed (TRUE) or has been removed (FALSE)
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This API is called by TcpIp in case of a change in the ARP/NDP table related to the controller specified by CtrlIdx.	
Available via	TcpIp_Externals.h	

⌋*()*

8.7.3.3 SocketOwner functions

[SWS_TcpIp_00220] ⌈For sockets related to a `TcpIpSocketOwner` with `TcpIpSocketOwnerUpperLayerType` set to 'SOAD', TcpIp shall replace the tag <Up> with 'SoAd' for each of the following configurable interfaces.⌋*(SRS_Eth_00103)*

[SWS_TcpIp_00221] ⌈For sockets related to a `TcpIpSocketOwner` with `TcpIpSocketOwnerUpperLayerType` set to 'CDD', TcpIp shall use the configured API names for each of the following configurable interfaces.⌋*(SRS_Eth_00103)*

8.7.3.3.1 <Up_RxIndication>

[SWS_TCPIP_00223] Definition of configurable interface <Up_RxIndication> ⌈

Service Name	<Up_RxIndication>	

Syntax	`void <Up_RxIndication> (`
	`  TcpIp_SocketIdType SocketId,`
	`  const TcpIp_SockAddrType* RemoteAddrPtr,`
	`  const uint8* BufPtr,`
	`  uint16 Length`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	RemoteAddrPtr	Pointer to memory containing IP address and port of the remote host which sent the data.
	BufPtr	Pointer to the received data.
	Length	Data length of the received TCP segment or UDP datagram.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	The TCP/IP stack calls this primitive after the reception of data on a socket. The socket identifier along with configuration information determines which module is to be called.	
Available via	SoAd.h if the respective TcpIpSocketOwnerUpperLayerType is SOAD, otherwise TcpIpSocket OwnerHeaderFileName	

⌋(*SRS_Eth_00103*)

8.7.3.3.2 <Up_TcpIpEvent>

[SWS_TCPIP_00224] Definition of configurable interface <Up_TcpIpEvent> ⌈

Service Name	<Up_TcpIpEvent>	

Syntax	`void <Up_TcpIpEvent> (`
	`  TcpIp_SocketIdType SocketId,`
	`  TcpIp_EventType Event`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	Event	This parameter contains a description of the event just encountered.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This service gets called if the stack encounters a condition described by the values in Event.	
Available via	SoAd.h if the respective TcpIpSocketOwnerUpperLayerType is SOAD, otherwise TcpIpSocket OwnerHeaderFileName	

⌋(*SRS_Eth_00103*)

8.7.3.3.3 <Up_TxConfirmation>

[SWS_TCPIP_00225] Definition of configurable interface <Up_TxConfirmation> ⌈

Service Name	<Up_TxConfirmation>	

Syntax	`void <Up_TxConfirmation> (`
	`  TcpIp_SocketIdType SocketId,`
	`  uint16 Length`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	Length	Number of transmitted data bytes.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	The TCP/IP stack calls this function after the data has been acknowledged by the peer for TCP. Caveats: The upper layer might not be able to determine exactly which data bytes have been confirmed.	
Available via	SoAd.h if the respective TcpIpSocketOwnerUpperLayerType is SOAD, otherwise TcpIpSocketOwnerHeaderFileName	

⌋(*SRS_Eth_00103*)

8.7.3.3.4 <Up_TcpAccepted>

[SWS_TCPIP_00226] Definition of configurable interface <Up_TcpAccepted> ⌈

Service Name	<Up_TcpAccepted>	

Syntax	`Std_ReturnType <Up_TcpAccepted> (`
	`  TcpIp_SocketIdType SocketId,`
	`  TcpIp_SocketIdType SocketIdConnected,`
	`  const TcpIp_SockAddrType* RemoteAddrPtr`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SocketId	Socket identifier of the related local socket resource which has been used at TcpIp_Bind()
	SocketIdConnected	Socket identifier of the local socket resource used for the established connection.
	RemoteAddrPtr	IP address and port of the remote host.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	Result of operation `E_OK` upper layer accepts the established connection `E_NOT_OK` upper layer refuses the established connection, TcpIp stack shall close the connection.
Description	This service gets called if the stack put a socket into the listen mode before (as server) and a peer connected to it (as client). In detail: The TCP/IP stack calls this function after a socket was set into the listen state with TcpIp_TcpListen() and a TCP connection is requested by the peer.	
Available via	SoAd.h if the respective TcpIpSocketOwnerUpperLayerType is SOAD, otherwise TcpIpSocketOwnerHeaderFileName	

⌋(*SRS_Eth_00103*)

8.7.3.3.5 <Up_TcpConnected>

[SWS_TCPIP_00227] Definition of configurable interface <Up_TcpConnected> ⌈

Service Name	<Up_TcpConnected>	

Syntax	`void <Up_TcpConnected> (`
	`  TcpIp_SocketIdType SocketId`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This service gets called if the stack initiated a TCP connection before (as client) and the peer (the server) acknowledged the connection set up. In detail: The TCP/IP stack calls this function after a socket was requested to connect with TcpIp_TcpConnect() and a TCP connection is confirmed by the peer. The parameter value of SocketId equals the SocketId value of the preceeding TcpIp_TcpConnect() call.	
Available via	SoAd.h if the respective TcpIpSocketOwnerUpperLayerType is SOAD, otherwise TcpIpSocket OwnerHeaderFileName	

⌋(*SRS_Eth_00103*)

8.7.3.3.6 <Up_CopyTxData>

[SWS_TCPIP_00228] Definition of configurable interface <Up_CopyTxData> ⌈

Service Name	<Up_CopyTxData>	

Syntax	`BufReq_ReturnType <Up_CopyTxData> (`
	`  TcpIp_SocketIdType SocketId,`
	`  uint8* BufPtr,`
	`  uint16 BufLength`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	BufLength	Length of provided data buffer.
Parameters (inout)	None	
Parameters (out)	BufPtr	Pointer to buffer for transmission data.
Return value	BufReq_ReturnType	`BUFREQ_OK`: Data has been copied to the transmit buffer completely as requested. `BUFREQ_E_NOT_OK`: Data has not been copied. Request failed. (No further action for TcpIp required. Later the upper layer might either close the socket or retry the transmit request)

▽

△

Description	This service requests to copy data for transmission to the buffer indicated. This call is triggered by TcpIp_Transmit(). Note: The call to <Up>_CopyTxData() may happen in the context of TcpIp_Transmit().
Available via	SoAd.h if the respective TcpIpSocketOwnerUpperLayerType is SOAD, otherwise TcpIpSocketOwnerHeaderFileName

⌋(*SRS_Eth_00103*)

8.7.3.3.7 <Up_LocalIpAddrAssignmentChg>

[SWS_TCPIP_00229] Definition of configurable interface <Up_LocalIpAddrAssignmentChg> ⌈

Service Name	<Up_LocalIpAddrAssignmentChg>	

Syntax	`void <Up_LocalIpAddrAssignmentChg> (`
	`  TcpIp_LocalAddrIdType IpAddrId,`
	`  TcpIp_IpAddrStateType State`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	IpAddrId	IP address Identifier, representing an IP address specified in the TcpIp module configuraiton (e.g. static IPv4 address on EthIf controller 0).
	State	state of IP address assignment
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This service gets called by the TCP/IP stack if an IP address assignment changes (i.e. new address assigned or assigned address becomes invalid).	
Available via	SoAd.h if the respective TcpIpSocketOwnerUpperLayerType is SOAD, otherwise TcpIpSocketOwnerHeaderFileName	

⌋(*SRS_Eth_00103*)

8.7.3.4 <Up_IcmpMsgHandler>

[SWS_TCPIP_00270] Definition of configurable interface <Up_IcmpMsgHandler> ⌈

Service Name	<Up_IcmpMsgHandler>	

Syntax	`void <Up_IcmpMsgHandler> (`
	`  TcpIp_LocalAddrIdType LocalAddrId,`
	`  const TcpIp_SockAddrType* RemoteAddrPtr,`
	`  uint8 Ttl,`
	`  uint8 Type,`
	`  uint8 Code,`
	`  uint16 DataLength,`
	`  uint8* DataPtr`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	LocalAddrId	Local address identifier representing the local IP address and Eth If controller where the ICMP message has been received.
	RemoteAddrPtr	pointer to struct representing the address of the ICMP sender
	Ttl	Time to live value of the received ICMPv4 message or Hop Limit value of the received ICMPv6 message.
	Type	type field value of the reveived ICMP message (Note: the value of the type field determines the format of the remaining ICMP message data)
	Code	code field value of the received ICMP message
	DataLength	length of ICMP message
	DataPtr	Pointer to the received ICMP message
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	By this API service the configured ICMP message handler function is called by the TCP/IP stack on reception of a ICMP message which is not handled by the TCP/IP stack.	
Available via	TcpIp_Externals.h	

⌋()

8.7.3.5 <Up_DADAddressConflict>

[SWS_TCPIP_91005] Definition of configurable interface <Up_DADAddressConflict> ⌈

Service Name	<Up_DADAddressConflict>

Syntax	`void <Up_DADAddressConflict> (`
	`  TcpIp_LocalAddrIdType IpAddrId,`
	`  const TcpIp_SockAddrType* IpAddrPtr,`
	`  const uint8* LocalPhysAddrPtr,`
	`  const uint8* RemotePhysAddrPtr`
	`)`

Service ID [hex]	0x1e

▽

△

Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	IpAddrId	IP address Identifier, representing an IP address specified in the TcpIp module configuration.
	IpAddrPtr	Pointer to a struct where the conflicted IP address is stored.
	LocalPhysAddrPtr	Pointer to the memory where the local physical address (MAC address) related to the specified IP address is stored in network byte order.
	RemotePhysAddrPtr	Pointer to the memory where the remote physical address (MAC address) related to the specified IP address is stored in network byte order.
Parameters (inout)	None	
Parameters (out)	None	
Return value	void	–
Description	This API is called by TcpIp in case the Duplicate Address Detection (DAD) is enabled and detecting a duplicate IP Address.	
Available via	TcpIp_Externals.h	

⌋()

[SWS_TcpIp_00283] ⌈If the optional `TcpIpDuplicateAddressDetectionConfig` is defined and a duplicate IP address was found by the Duplicate Address Detection (DAD) algorithm, the TcpIp shall call the callout function specified by `TcpIpDuplicateAddressDetectionCalloutName`.⌋(*SRS_Eth_00091*, *SRS_BSW_00452*)

8.7.3.6 <Up_TlsGetCurrentTime>

[SWS_TCPIP_91012] Definition of configurable interface <Up_TlsGetCurrentTime> ⌈

Service Name	<Up_TlsGetCurrentTime>	

Syntax	`Std_ReturnType <Up_TlsGetCurrentTime> (`
	`  uint32* CurrentTimeUtc`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	CurrentTimeUtc	Pointer to uint32 to provide the GMT Unix time value.
Return value	Std_ReturnType	`E_OK`: Time stamp successfully provided. `E_NOT_OK`: Time stamp can currently not be provided. Data in CurrentTimeUtc not valid.
Description	This function queries the current time. This information will be requested when assembling the client hello message.	
Available via	TcpIp_Externals.h	

⌋()

[SWS_TcpIp_00330] ⌈If the optional parameter `TcpIpTlsConnectionGetTimeFunc` is defined the `TLS_CLIENT` shall call the configured function to query the current

time. The value 0 indicates that no time is available. The value 0 is also transmitted if the function returns E_NOT_OK.⌋()

[SWS_TcpIp_00332] ⌈The function <Up_TlsGetCurrentTime>() shall provide the current UTC time. It is used to assemble the ClientHello handshake message. The time is provided in big endian format and follows either the GMT Unix time format or can be 0 (See IETF RFC 5246, section 7.4.1.2, gmt_unix_time for details).⌋()

8.7.3.7 <Up_TlsServerGetPskIdentityHint>

[SWS_TCPIP_91013] Definition of configurable interface <Up_TlsServerGetPskIdentityHint> ⌈

Service Name	<Up_TlsServerGetPskIdentityHint>	

Syntax	`Std_ReturnType <Up_TlsServerGetPskIdentityHint> (`
	`  TcpIp_SocketIdType SocketId,`
	`  TcpIp_TlsConnectionIdType TlsConnectionId,`
	`  uint16* IdentityHintLengthPtr,`
	`  uint8* IdentityHintPtr`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	TlsConnectionId	Provides the TLS connection identifier.
Parameters (inout)	IdentityHintLengthPtr	In: Provides the number of bytes available where identityHintPtr links to. Out: Provides the number of bytes that has been overwritten in identityHintPtr.
Parameters (out)	IdentityHintPtr	Ptr to buffer that is used to store the IdentityHint information.
Return value	Std_ReturnType	E_OK: IdentityHint successfully provided E_NOT_OK: IdentityHint could not be provided. Data in the pointer is invalid and shall not be used.
Description	Queries the Identity hint for a pre-shared key ciphersuite. This information is transmitted by the TLS Server to provide its identification to the TLS client.	
Available via	TcpIp_Externals.h	

⌋(*SRS_Eth_00137*)

[SWS_TcpIp_00333] ⌈If the TLS_SERVER selects a PSK ciphersuite from the offered ciphersuite list and TcpIpTlsPresharedKeyIdentityHint is not defined but <Up_TlsServerGetPskIdentityHint>() is defined, then this function shall be called when the TLS_SERVER assembles the ServerKeyExchange message (according to RFC4279, Sect. 2) during the handshake to query the psk_identity_hint.⌋()

8.7.3.8 <Up_TlsClientGetPskIdentity >

[SWS_TCPIP_91014] Definition of configurable interface <Up_TlsClientGetPskIdentity> ⌈

Service Name	<Up_TlsClientGetPskIdentity>	

Syntax	`Std_ReturnType <Up_TlsClientGetPskIdentity> (`
	`  TcpIp_SocketIdType SocketId,`
	`  TcpIp_TlsConnectionIdType TlsConnectionId,`
	`  uint16 PskIdentityHintLength,`
	`  const uint8* PskIdentityHintPtr,`
	`  uint16* PskKeyIdentityLengthPtr,`
	`  uint8* PskKeyIdentityPtr,`
	`  uint32* CsmKeyId`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	TlsConnectionId	Provides the TLS connection identifier.
	PskIdentityHintLength	Provides the number of bytes available in identityHintPtr.
	PskIdentityHintPtr	Pointer to the identity hint information from the server.
Parameters (inout)	PskKeyIdentityLengthPtr	In: Provides the number of bytes available in PskKeyIdentityPtr. Out: Provides the actual number of bytes that has been written to PskKeyIdentityPtr.
Parameters (out)	PskKeyIdentityPtr	Buffer that is used to store the pre-shared key identification.
	CsmKeyId	Provides the identifier of a CSM key.
Return value	Std_ReturnType	`E_OK`: Pre-Shared key selected properly. All output values are valid. `E_NOT_OK`: Pre-Shared key could not be selected. Key selection failed.
Description	This function is called on the TLS client side. It provides the key identification based on the identity hint provided by the TLS server. The TLS client selects the pre-shared key and returns the key identification name and the CSM key reference.	
Available via	TcpIp_Externals.h	

⌋*(SRS_Eth_00137)*

[SWS_TcpIp_00334] ⌈If the `TLS_CLIENT` receives a selected PSK ciphersuite and `TcpIpTlsPresharedKeyIdentityHint` or `TcpIpTlsPresharedKeyIdentity` or `TcpIpTlsPresharedKeyCsmKeyRef` is not defined but `<Up_TlsClientGetPskIdentity>`() is defined, then this function shall be called when the `TLS_CLIENT` assembles the ClientKeyExchange message (according to RFC4279, Sect. 2). The function provides the pre-shared key and the psk_identity which is provided in the ClientKeyExchange message.⌋*()*

8.7.3.9 <Up_TlsServerGetPskIdentity>

[SWS_TCPIP_91015] Definition of configurable interface <Up_TlsServerGetPskIdentity> ⌈

Service Name	<Up_TlsServerGetPskIdentity>	

Syntax	`Std_ReturnType <Up_TlsServerGetPskIdentity> (`
	`  TcpIp_SocketIdType SocketId,`
	`  TcpIp_TlsConnectionIdType TlsConnectionId,`
	`  uint16 PskKeyIdentityLength,`
	`  const uint8* PskKeyIdentityPtr,`
	`  uint32* CsmKeyId`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SocketId	Socket identifier of the related local socket resource.
	TlsConnectionId	Provides the TLS connection identifier.
	PskKeyIdentityLength	Provides the number of bytes available in PskKeyIdentityPtr.
	PskKeyIdentityPtr	Pointer to a buffer that provides the PSK key identification information.
Parameters (inout)	None	
Parameters (out)	CsmKeyId	Provides the identifier of a CSM key.
Return value	Std_ReturnType	`E_OK`: PSK key was identified and CsmKey reference provided properly. `E_NOT_OK`: Key identification or PSK key could not be identified.
Description	This callback is used for the TLS server to provide the CSM key name according to the key identification that was selected by the TLS client. The TLS server must provide a CsmKey reference to a key that matches this key identification name.	
Available via	TcpIp_Externals.h	

⌋*(SRS_Eth_00137)*

[SWS_TcpIp_00335] ⌈If the `TLS_SERVER` receives the ClientKeyExchange message during the handshake and `TcpIpTlsPresharedKeyIdentity` or `TcpIpTlsPresharedKeyCsmKeyRef` is not defined but `<Up_TlsServerGetPskIdentity>`() is defined, then this function shall be called when the `TLS_CLIENT` assembles the ClientKeyExchange message (according to RFC4279, Sect. 2). The function provides the pre-shared key and the psk_identity which is provided in the ClientKeyExchange message.⌋*()*

8.8 Service Interfaces

No service interfaces provided.

9 Sequence diagrams

Note: The following sequence charts showcase SoAd as upper layer of TcpIp. They shall be understood as example for any other configurable upper layer module.

9.1 TCP Connection Setup - Client

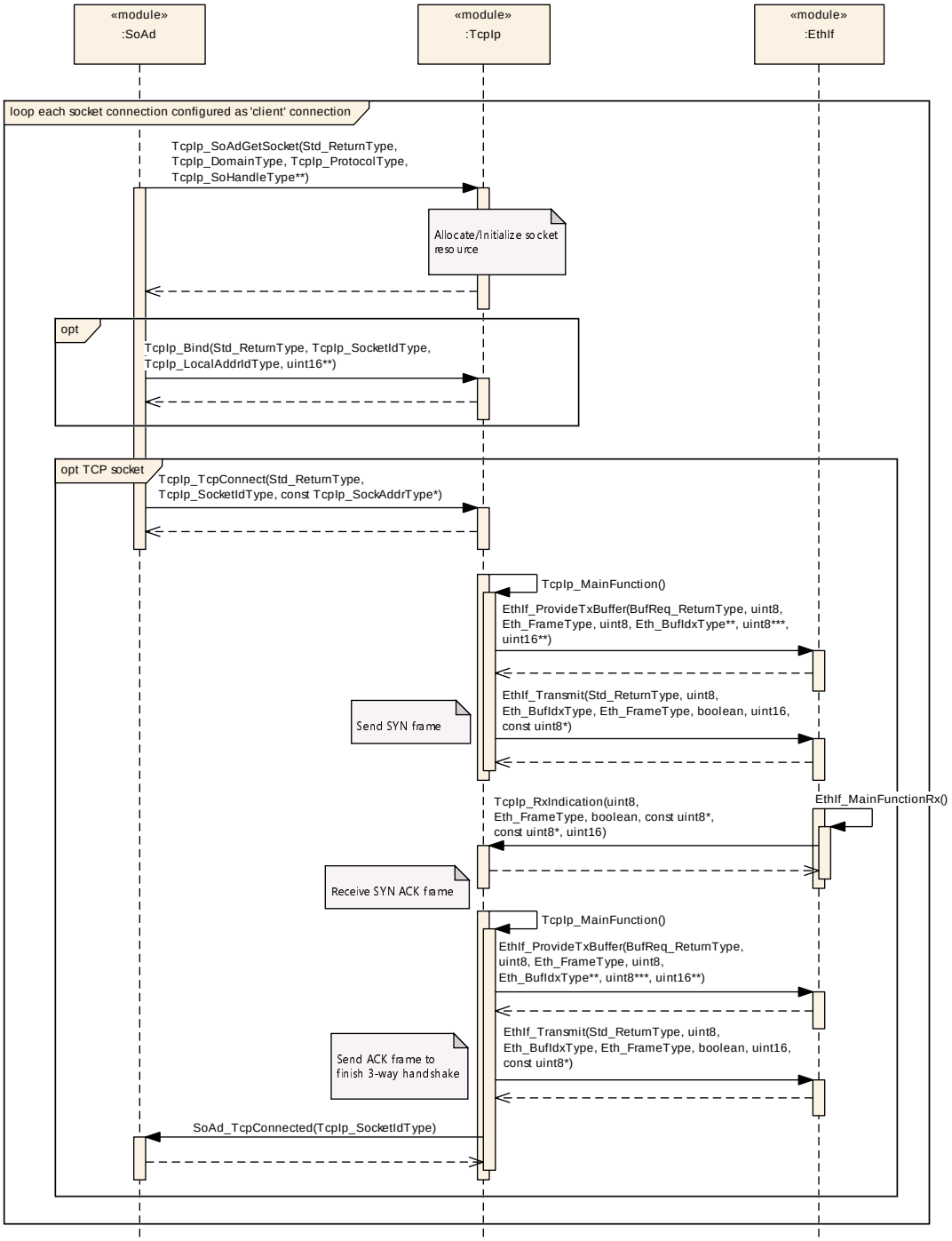

Figure 9.1: TcpIp TCP connection Setup Client

9.2 TCP Connection Setup - Server

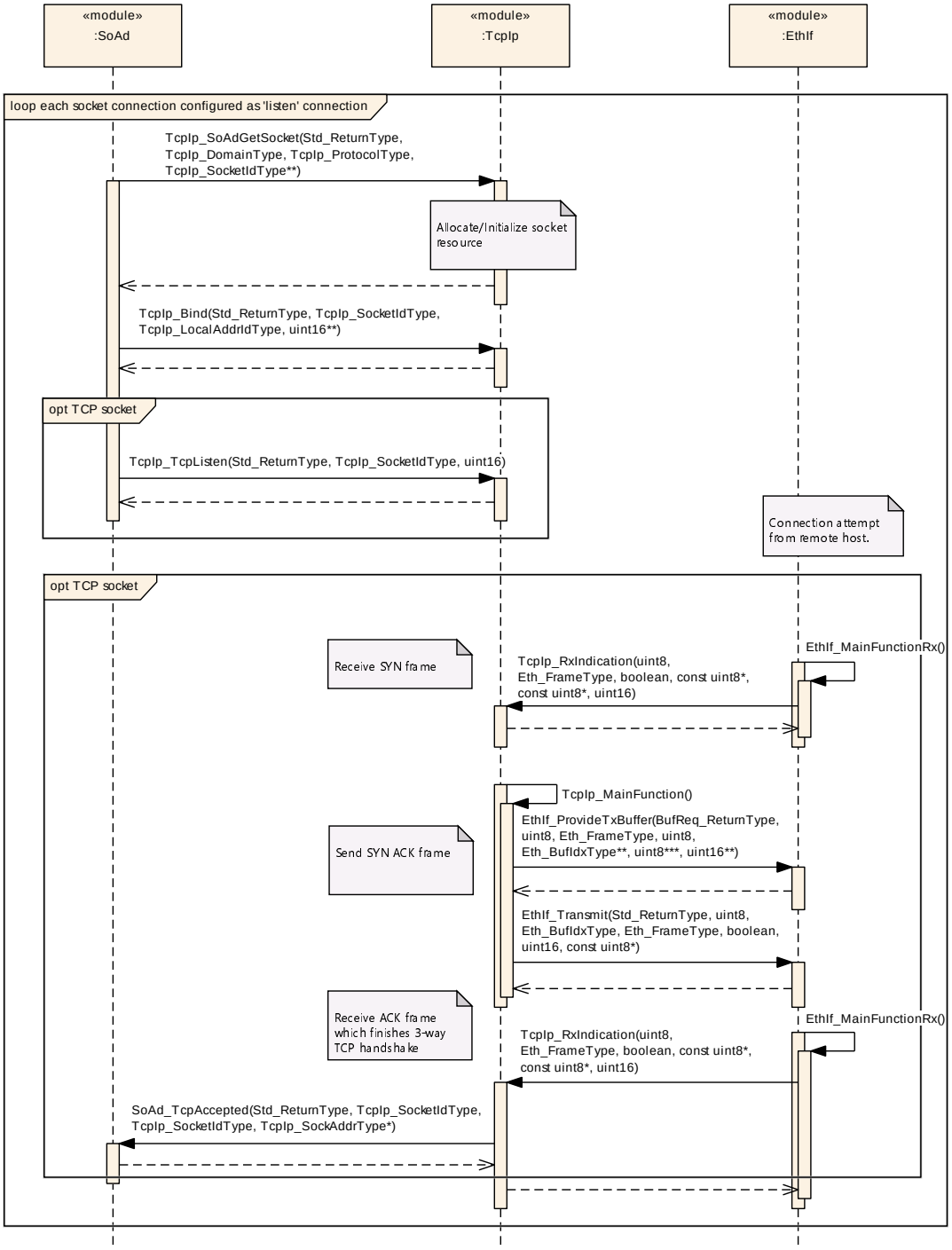

Figure 9.2: TcpIp TCP connection Setup Server

9.3 Reception

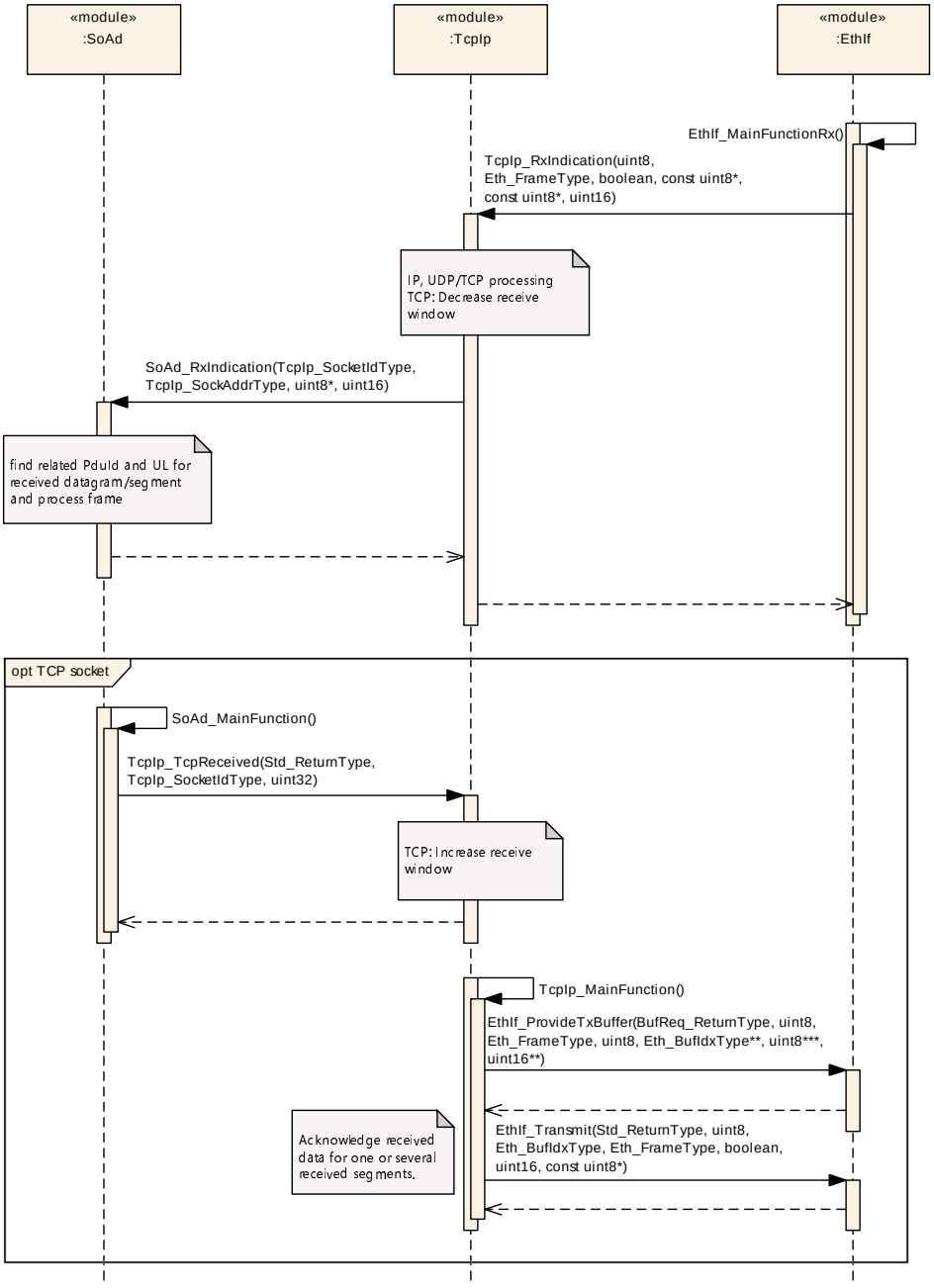

Figure 9.3: TcpIp Rx

Note: Even it is not shown in the sequence diagram of section section 9.3, TcpIp may decouple the data reception if required. E.g. for reassembling of incoming IP datagrams that are fragmented, TcpIp shall copy the received data to a TcpIp buffer and decouple `TcpIp_RxIndication`() from `SoAd_RxIndication`().

9.4 Transmission TCP

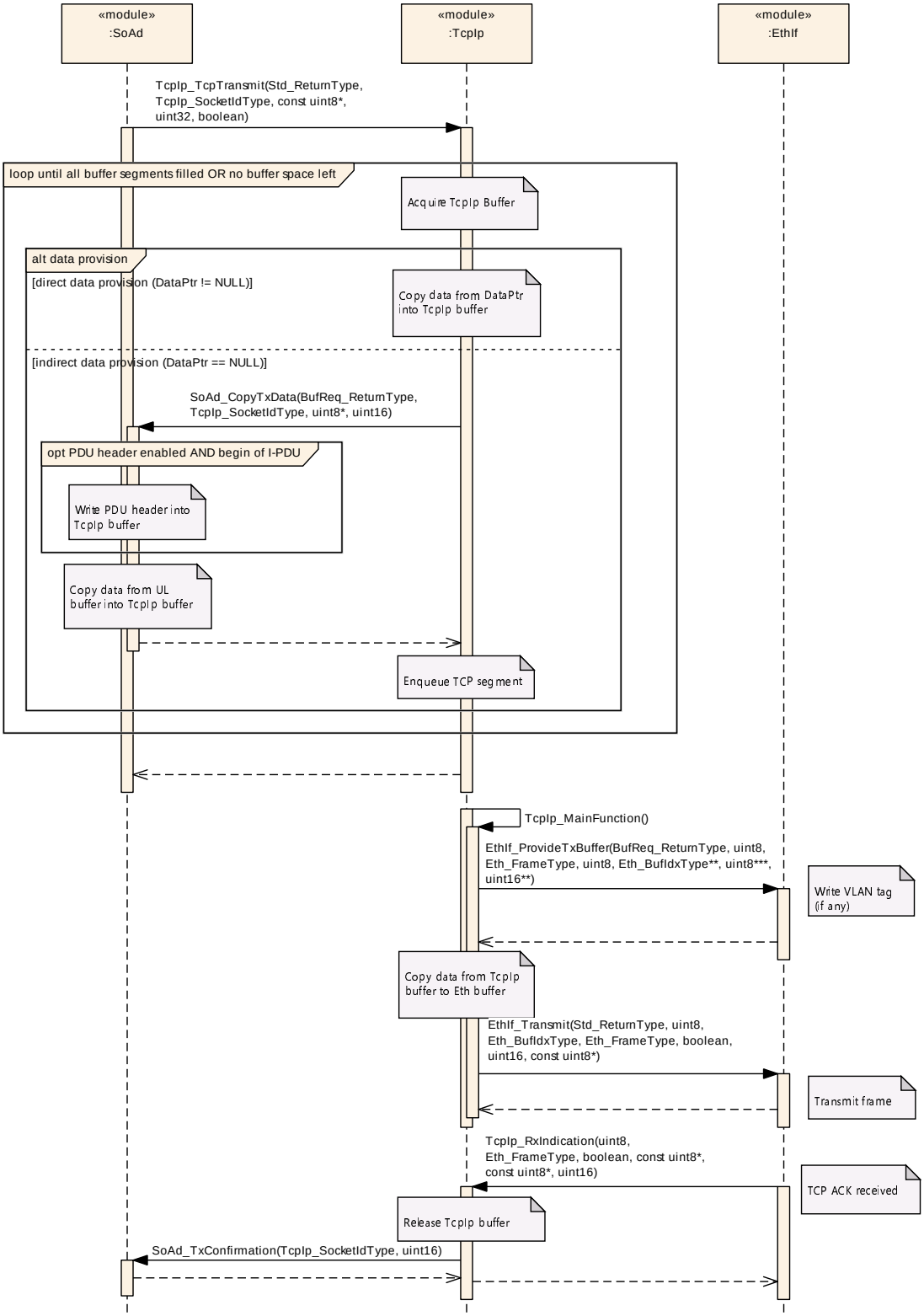

Figure 9.4: TcpIp TCP Tx

9.5 Transmission UDP

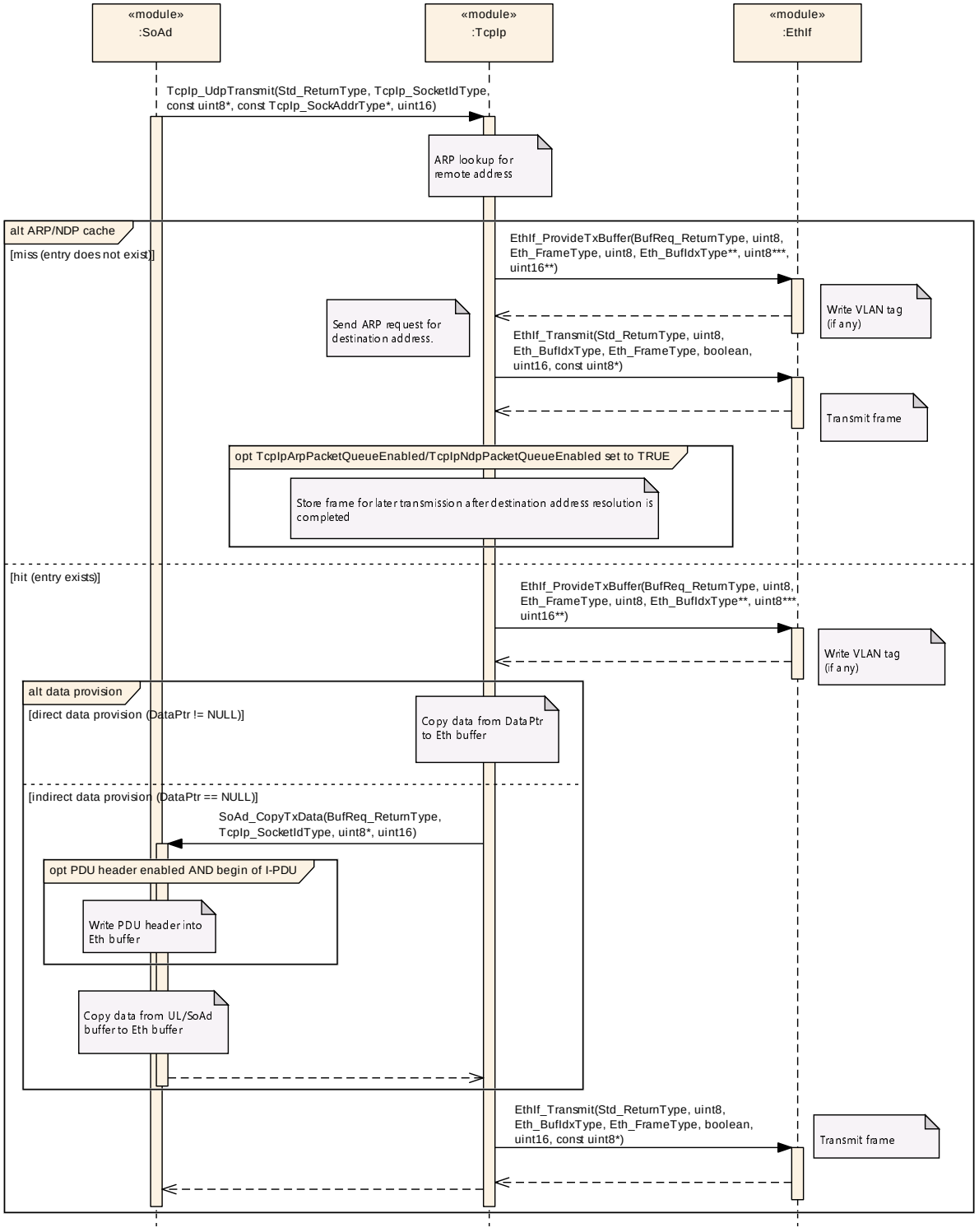

Figure 9.5: TcpIp UDP Tx

9.6 Connection setup for a TLS server

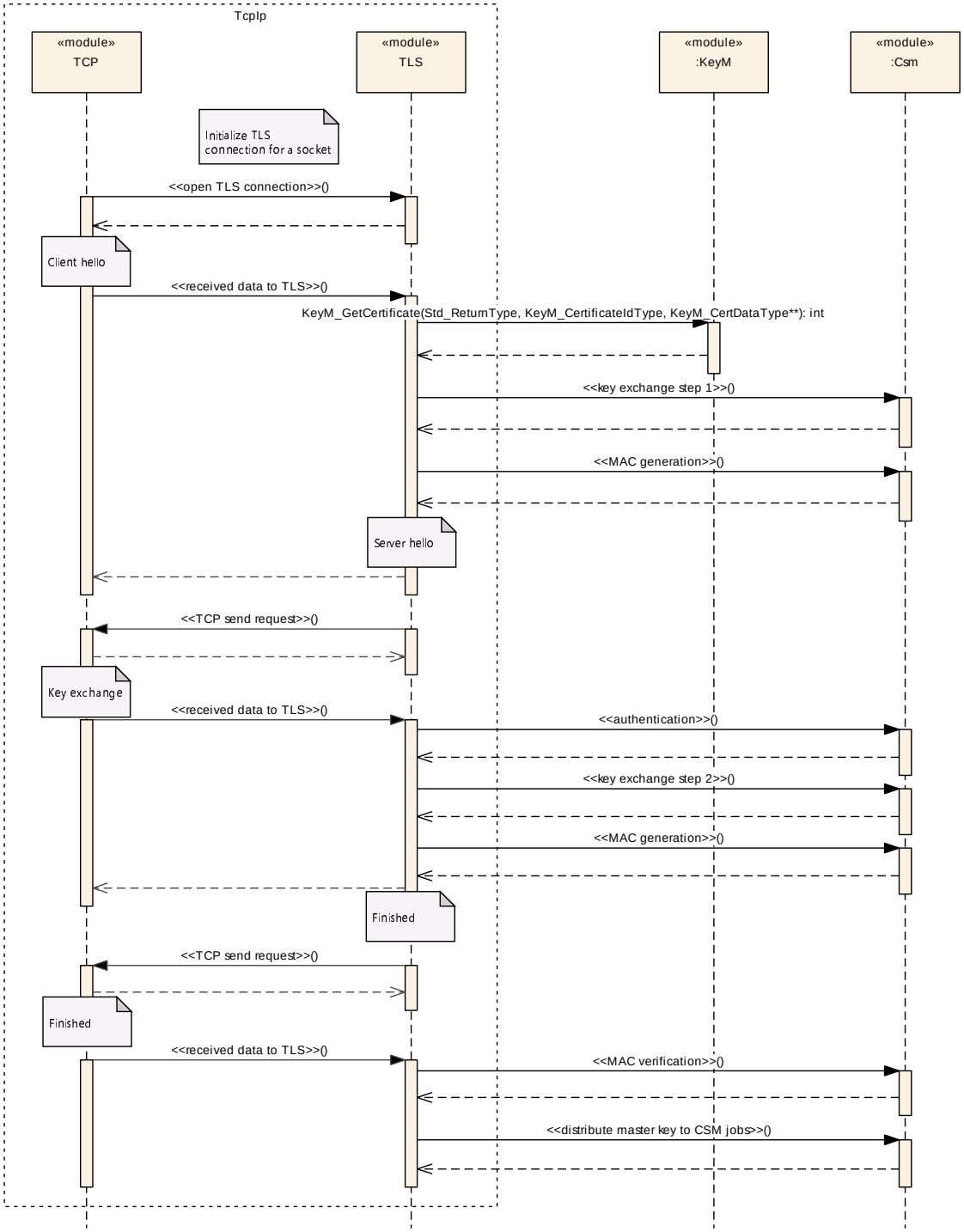

Figure 9.6: TcpIp TLS server communication

9.7 TLS connection assignment to socket

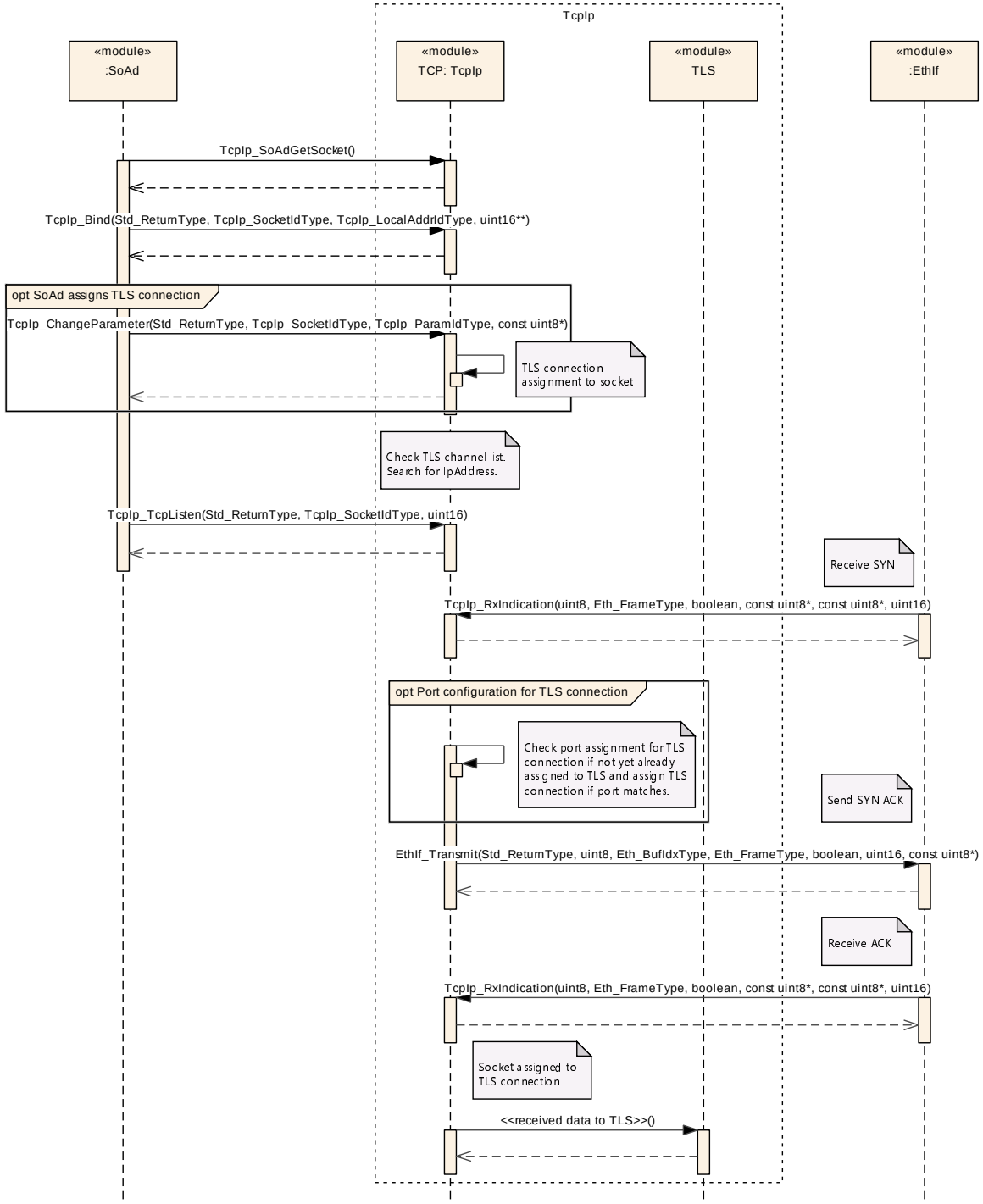

Figure 9.7: TcpIp TLS connection setup server

10 Configuration specification

In general, this chapter defines configuration parameters and their clustering into containers. In order to support the specification Chapter 10.1 describes fundamentals. It also specifies a template (table) you shall use for the parameter specification. We intend to leave Chapter 10.1 in the specification to guarantee comprehension.

Chapter 10.2 specifies the structure (containers) and the parameters of the module Tcp/Ip.

Chapter 10.3 specifies published information of the module Tcp/Ip.

10.1 How to read this chapter

For details refer to the chapter 10.1 “Introduction to configuration specification” in SWS_BSWGeneral.

10.2 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters describe Chapter 7 and Chapter 8.

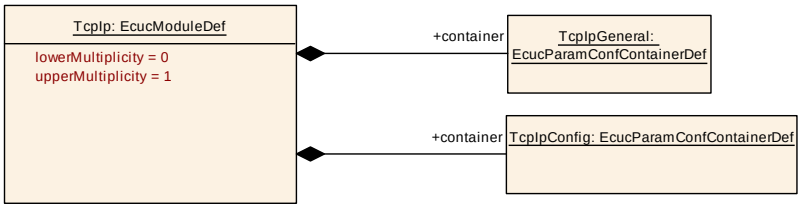

Figure 10.1: TcpIp

10.2.1 TcpIp

SWS Item	[ECUC_TcpIp_00001]
Module Name	TcpIp
Description	Configuration of the TcpIp (TCP/IP stack) module.
Post-Build Variant Support	true
Supported Config Variants	VARIANT-LINK-TIME, VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpConfig	1	This container contains the configuration parameters and sub containers of the AUTOSAR TcpIp module.
TcpIpGeneral	1	This container is a subcontainer of TcpIp and specifies the general configuration parameters of the TCP/IP stack.

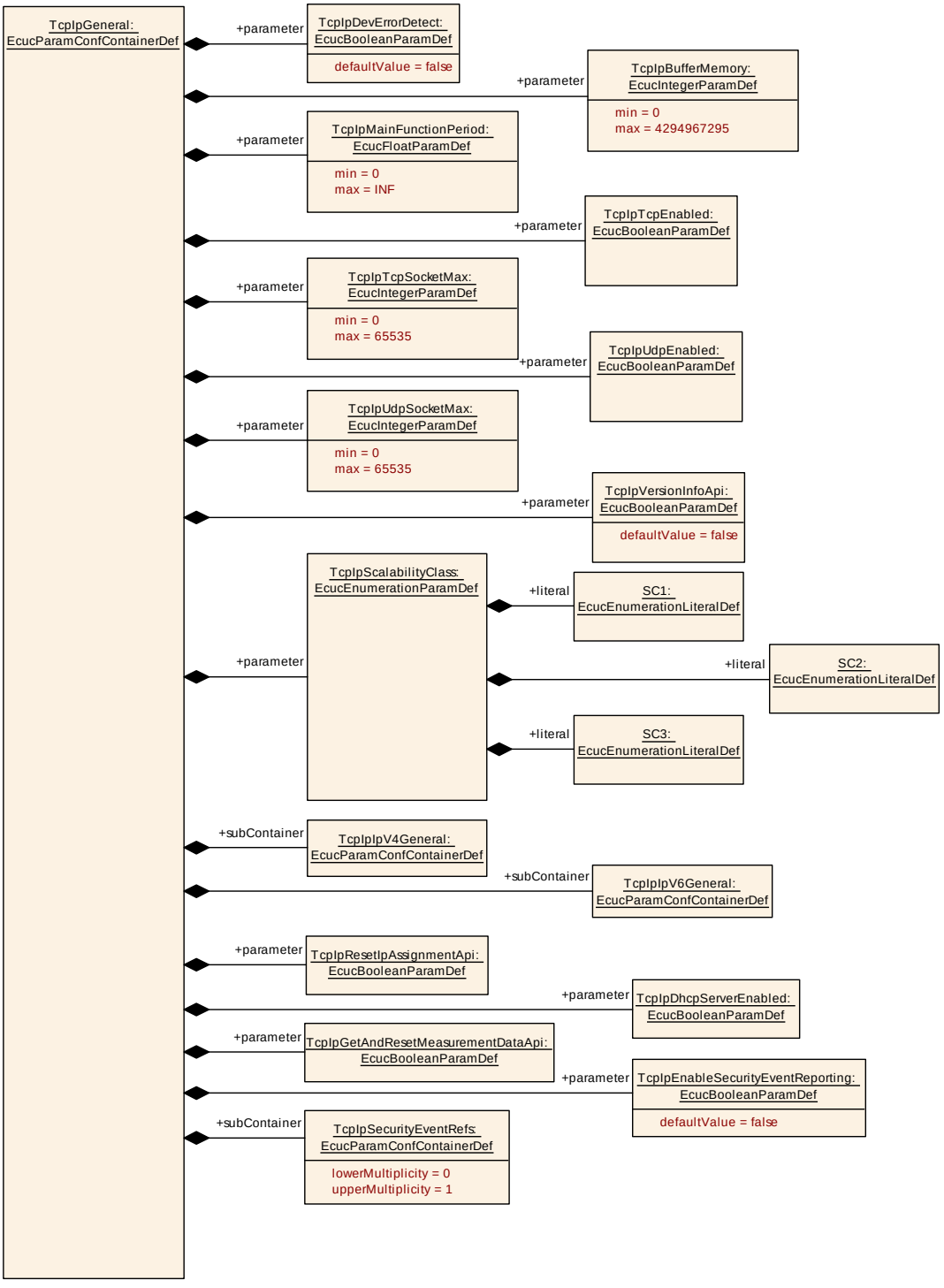

Figure 10.2: TcpIpGeneral

10.2.2 TcpIpGeneral

SWS Item	[ECUC_TcpIp_00002]		
Container Name	TcpIpGeneral		
Parent Container	TcpIp		
Description	This container is a subcontainer of TcpIp and specifies the general configuration parameters of the TCP/IP stack.		
Configuration Parameters			

SWS Item	[ECUC_TcpIp_00016]		
Parameter Name	TcpIpBufferMemory		
Parent Container	TcpIpGeneral		
Description	Memory size in bytes reserved for TCP/IP buffers.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00004]		
Parameter Name	TcpIpDevErrorDetect		
Parent Container	TcpIpGeneral		
Description	Switches the development error detection and notification on or off. • true: detection and notification is enabled. • false: detection and notification is disabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00183]		
Parameter Name	TcpIpDhcpServerEnabled		
Parent Container	TcpIpGeneral		
Description	Enables (TRUE) or disables (FALSE) the DHCP (Dynamic Host Configuration Protocol) Server.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Value Configuration Class	**Pre-compile time**	X	All Variants

△

	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00319]**		
Parameter Name	TcpIpEnableSecurityEventReporting		
Parent Container	TcpIpGeneral		
Description	Switches the reporting of security events to the IdsM: - true: reporting is enabled. - false: reporting is disabled. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

SWS Item	**[ECUC_TcpIp_00217]**		
Parameter Name	TcpIpGetAndResetMeasurementDataApi		
Parent Container	TcpIpGeneral		
Description	Enables / Disables the Get and Reset Measurement Data API		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00013]**		
Parameter Name	TcpIpMainFunctionPeriod		
Parent Container	TcpIpGeneral		
Description	Period of TcpIp_MainFunction in [s].		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	]0 .. INF[		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00182]		
Parameter Name	TcpIpResetIpAssignmentApi		
Parent Container	TcpIpGeneral		
Description	Enables/disables the API TcpIp_ResetIpAssignment of a DHCP-client.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00169]	
Parameter Name	TcpIpScalabilityClass	
Parent Container	TcpIpGeneral	
Description	In order to customize the TcpIp Stack to the specific needs of the user it can be scaled according to the scalability classes.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	SC1	IPv4 - In-Vehicle and Diagnostic Communication
	SC2	IPv6 - In-Vehicle and Diagnostic Communication
	SC3	IPv4 and IPv6 (Dual Stack) - In-Vehicle and Diagnostic Communication
Post-Build Variant Value	false	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local	

SWS Item	[ECUC_TcpIp_00008]		
Parameter Name	TcpIpTcpEnabled		
Parent Container	TcpIpGeneral		
Description	Enables (TRUE) or disabled (FALSE) support of TCP (Transmission Control Protocol).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00014]
Parameter Name	TcpIpTcpSocketMax
Parent Container	TcpIpGeneral
Description	Maximum number of TCP sockets
Multiplicity	1

▽

△

Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00009]**		
Parameter Name	TcpIpUdpEnabled		
Parent Container	TcpIpGeneral		
Description	Enables (TRUE) or disabled (FALSE) support of UDP (User Datagram Protocol)		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00015]**		
Parameter Name	TcpIpUdpSocketMax		
Parent Container	TcpIpGeneral		
Description	Maximum number of UDP sockets.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00005]**		
Parameter Name	TcpIpVersionInfoApi		
Parent Container	TcpIpGeneral		
Description	If true the TcpIp_GetVersionInfo API is available.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants

▽

△

	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpIpV4General	1	This container is a subcontainer of TcpIp and specifies the general configuration parameters of the TCP/IP stack for IPv4
TcpIpIpV6General	1	This container is a subcontainer of TcpIp and specifies the general configuration parameters of the TCP/IP stack for IPv6.
TcpIpSecurityEventRefs	0..1	Container for the references to IdsMEvent elements representing the security events that the TcpIp module shall report to the IdsM in case the coresponding security related event occurs (and if TcpIpEnableSecurityEventReporting is set to "true"). The standardized security events in this container can be extended by vendor-specific security events. **Tags:** atp.Status=draft

10.2.3 TcpIpIpV4General

SWS Item	**[ECUC_TcpIp_00163]**
Container Name	TcpIpIpV4General
Parent Container	TcpIpGeneral
Description	This container is a subcontainer of TcpIp and specifies the general configuration parameters of the TCP/IP stack for IPv4
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00006]**		
Parameter Name	TcpIpArpEnabled		
Parent Container	TcpIpIpV4General		
Description	Enables (TRUE) or disables (FALSE) support of ARP (Address Resolution Protocol).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00011]**
Parameter Name	TcpIpAutoIpEnabled
Parent Container	TcpIpIpV4General
Description	Enables (TRUE) or disables (FALSE) the Auto-IP (automatic private IP addressing) sub-module.
Multiplicity	1

▽

△

Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00010]**		
Parameter Name	TcpIpDhcpClientEnabled		
Parent Container	TcpIpIpV4General		
Description	Enables (TRUE) or disables (FALSE) the DHCP (Dynamic Host Configuration Protocol) Client.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00007]**		
Parameter Name	TcpIpIcmpEnabled		
Parent Container	TcpIpIpV4General		
Description	Enables (TRUE) or disabled (FALSE) support of ICMP (Internet Control Message Protocol).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00088]**		
Parameter Name	TcpIpIpV4Enabled		
Parent Container	TcpIpIpV4General		
Description	Enables (TRUE) or disables (FALSE) support of IPv4 (Internet Protocol version 4).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

▽

△

Scope / Dependency	scope: local

SWS Item	**[ECUC_TcpIp_00018]**		
Parameter Name	TcpIpLocalAddrIpv4EntriesMax		
Parent Container	TcpIpIpV4General		
Description	Maximum number of LocalAddr table entries for IPv4.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00012]**		
Parameter Name	TcpIpPathMtuDiscoveryEnabled		
Parent Container	TcpIpIpV4General		
Description	Enables (TRUE) or disables (FALSE) the discovery of the maximum transmission unit on a path according to IETF RfC 1191.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.2.4 TcpIpIpV6General

SWS Item	**[ECUC_TcpIp_00164]**
Container Name	TcpIpIpV6General
Parent Container	TcpIpGeneral
Description	This container is a subcontainer of TcpIp and specifies the general configuration parameters of the TCP/IP stack for IPv6.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00093]		
Parameter Name	TcpIpDhcpV6ClientEnabled		
Parent Container	TcpIpIpV6General		
Description	Enables (TRUE) or disables (FALSE) the DHCPv6 (Dynamic Host Configuration Protocol for IPv6) Client.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00089]		
Parameter Name	TcpIpIpV6Enabled		
Parent Container	TcpIpIpV6General		
Description	Enables (TRUE) or disables (FALSE) support of IPv6 (Internet Protocol version 6).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00090]		
Parameter Name	TcpIpIpV6PathMtuDiscoveryEnabled		
Parent Container	TcpIpIpV6General		
Description	Enables (TRUE) or disables (FALSE) Path MTU Discovery support for IPv6 according to IETF RFC 1981.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00017]
Parameter Name	TcpIpLocalAddrIpv6EntriesMax
Parent Container	TcpIpIpV6General
Description	Maximum number of LocalAddr table entries for IPv6.
Multiplicity	1
Type	EcucIntegerParamDef

▽

△

Range	0 .. 255		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00091]**		
Parameter Name	TcpIpNdpAddressResolutionUnrechabilityDetectionEnabled		
Parent Container	TcpIpIpV6General		
Description	Enables (TRUE) or disables (FALSE) support of Address Resoultion and Neighbor Unreachability Detetion via NDP.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00092]**		
Parameter Name	TcpIpNdpPrefixAndRouterDiscoveryEnabled		
Parent Container	TcpIpIpV6General		
Description	Enables (TRUE) or disables (FALSE) support of Prefix and Router Discovery via NDP.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

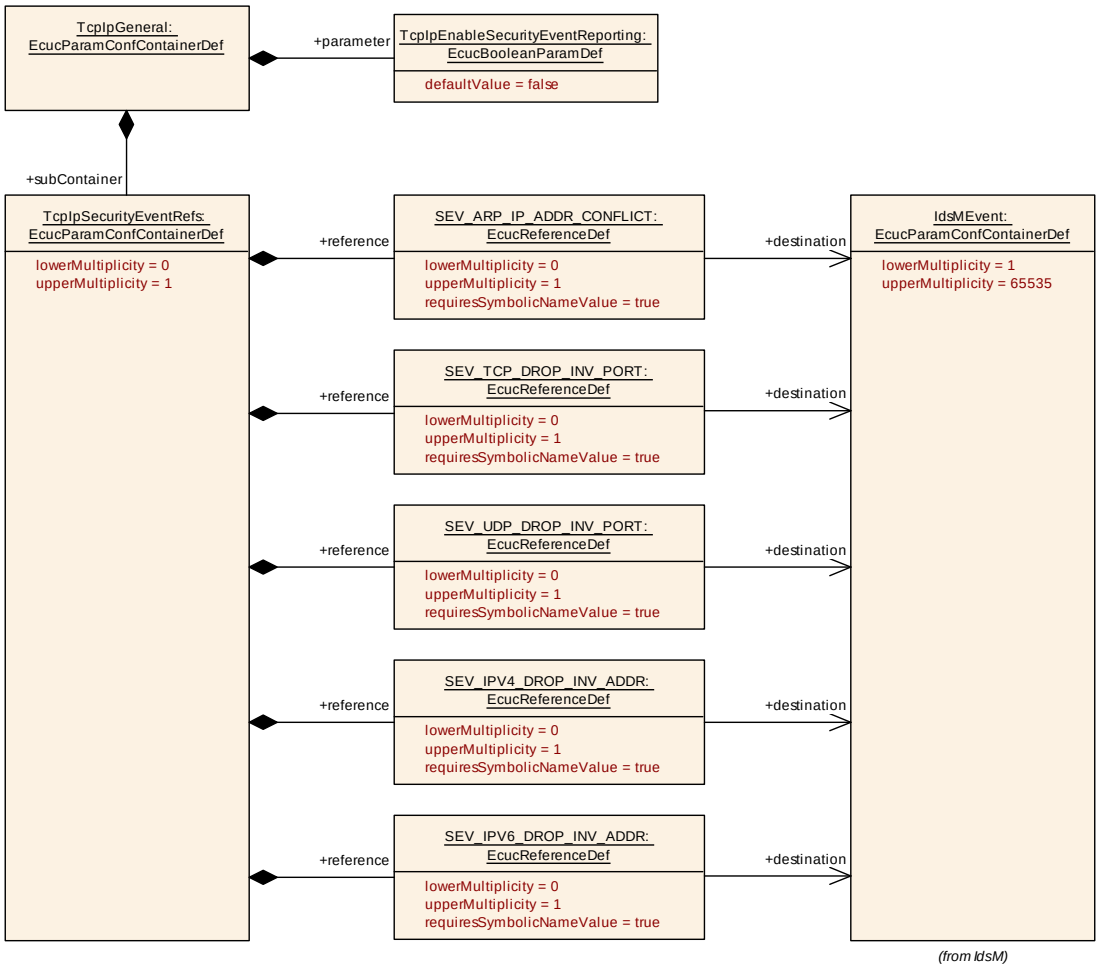

Figure 10.3: TcpIpSecurityEventRefs

10.2.5 TcpIpSecurityEventRefs

SWS Item	**[ECUC_TcpIp_00320]**		
Container Name	TcpIpSecurityEventRefs		
Parent Container	TcpIpGeneral		
Description	Container for the references to IdsMEvent elements representing the security events that the TcpIp module shall report to the IdsM in case the coresponding security related event occurs (and if TcpIpEnableSecurityEventReporting is set to "true"). The standardized security events in this container can be extended by vendor-specific security events. **Tags:** atp.Status=draft		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	[ECUC_TcpIp_00321]		
Parameter Name	SEV_ARP_IP_ADDR_CONFLICT		
Parent Container	TcpIpSecurityEventRefs		
Description	Received local IP address in ARP reply for different MAC. **Tags:** atp.Status=draft		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00324]		
Parameter Name	SEV_IPV4_DROP_INV_ADDR		
Parent Container	TcpIpSecurityEventRefs		
Description	Dropped datagram because of invalid IPV4 address. **Tags:** atp.Status=draft		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00325]		
Parameter Name	SEV_IPV6_DROP_INV_ADDR		
Parent Container	TcpIpSecurityEventRefs		
Description	Dropped datagram because of invalid IPV6 address. **Tags:** atp.Status=draft		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

▽

△

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00322]**		
Parameter Name	SEV_TCP_DROP_INV_PORT		
Parent Container	TcpIpSecurityEventRefs		
Description	Dropped TCP packet because of invalid destination TCP-Port. **Tags:** atp.Status=draft		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00323]**		
Parameter Name	SEV_UDP_DROP_INV_PORT		
Parent Container	TcpIpSecurityEventRefs		
Description	Dropped UDP packet because of invalid destination UDP-Port. **Tags:** atp.Status=draft		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

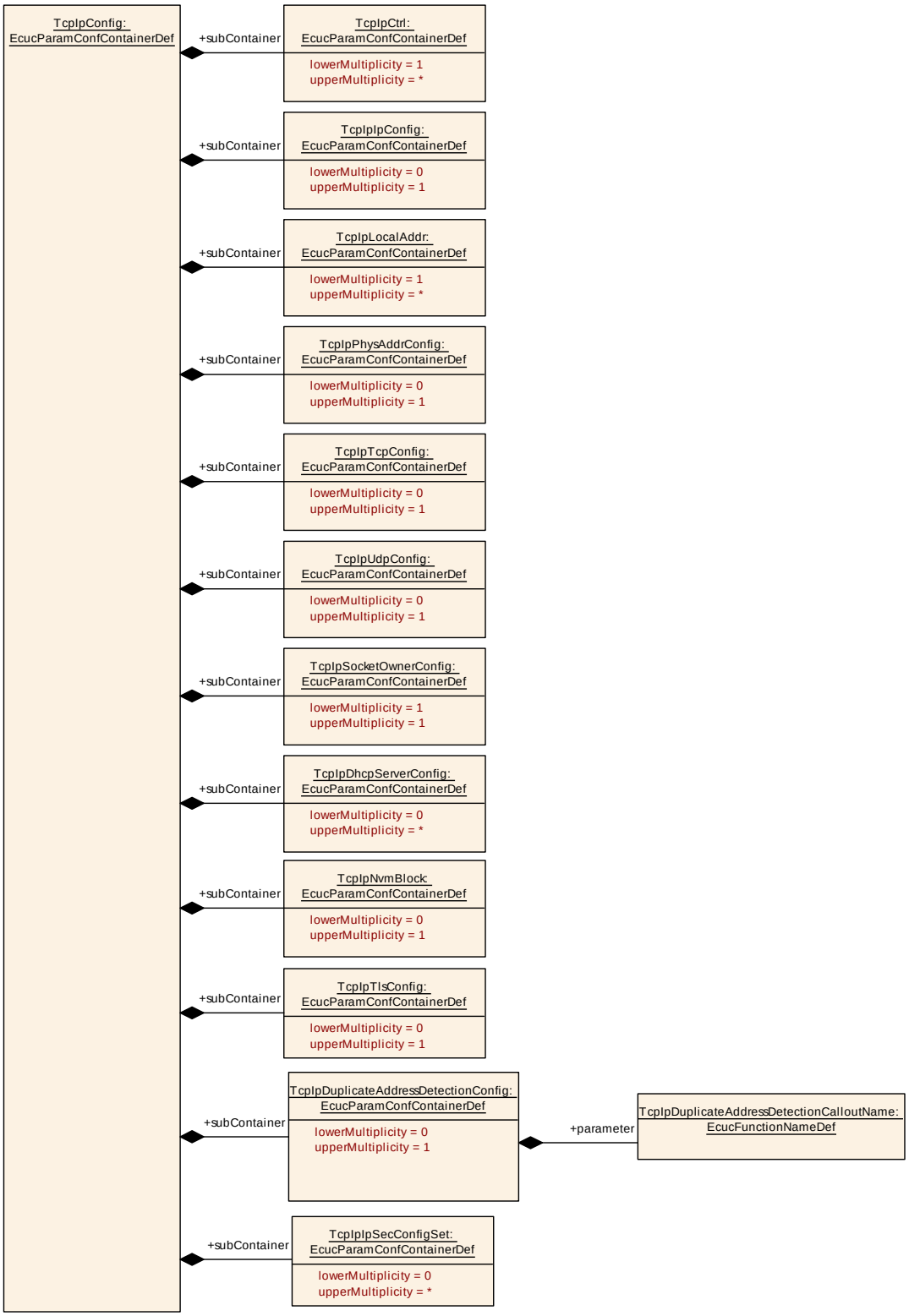

Figure 10.4: TcpIpConfig

10.2.6 TcpIpConfig

SWS Item	[ECUC_TcpIp_00003]
Container Name	TcpIpConfig
Parent Container	TcpIp
Description	This container contains the configuration parameters and sub containers of the AUTOSAR TcpIp module.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpCtrl	1..*	Specifies the EthIf controller used for IP communication.
TcpIpDhcpServerConfig	0..*	Specifies the configuration parameters of the DHCP Server sub-module.
TcpIpDuplicateAddressDetection Config	0..1	Specifies the DAD callout function.
TcpIpIpConfig	0..1	Specifies the configuration parameters of the IP (Internet Protocol) sub-module
TcpIpIpSecConfigSet	0..*	Specifies the IPsec configuration.
TcpIpLocalAddr	1..*	Specifies the local IP (Internet Protocol) addresses used for IP communication.
TcpIpNvmBlock	0..1	Configuration of optional usage of Nvm in case the TcpIp module requires non volatile memory in the Ecu to store information (e.g. IP Address received via DHCP and shall be stored).
TcpIpPhysAddrConfig	0..1	Specifies the physical address configuration.
TcpIpSocketOwnerConfig	1	Specifies the upper layer modules of TcpIp using the socket API.
TcpIpTcpConfig	0..1	Specifies the configuration parameters of the TCP (Transmission Control Protocol) sub-module.
TcpIpTlsConfig	0..1	Specifies the configuration parameters of the TLS (Transport Layer Security) sub module.
TcpIpUdpConfig	0..1	Specifies the configuration parameters of the UDP (User Datagram Protocol) sub-module

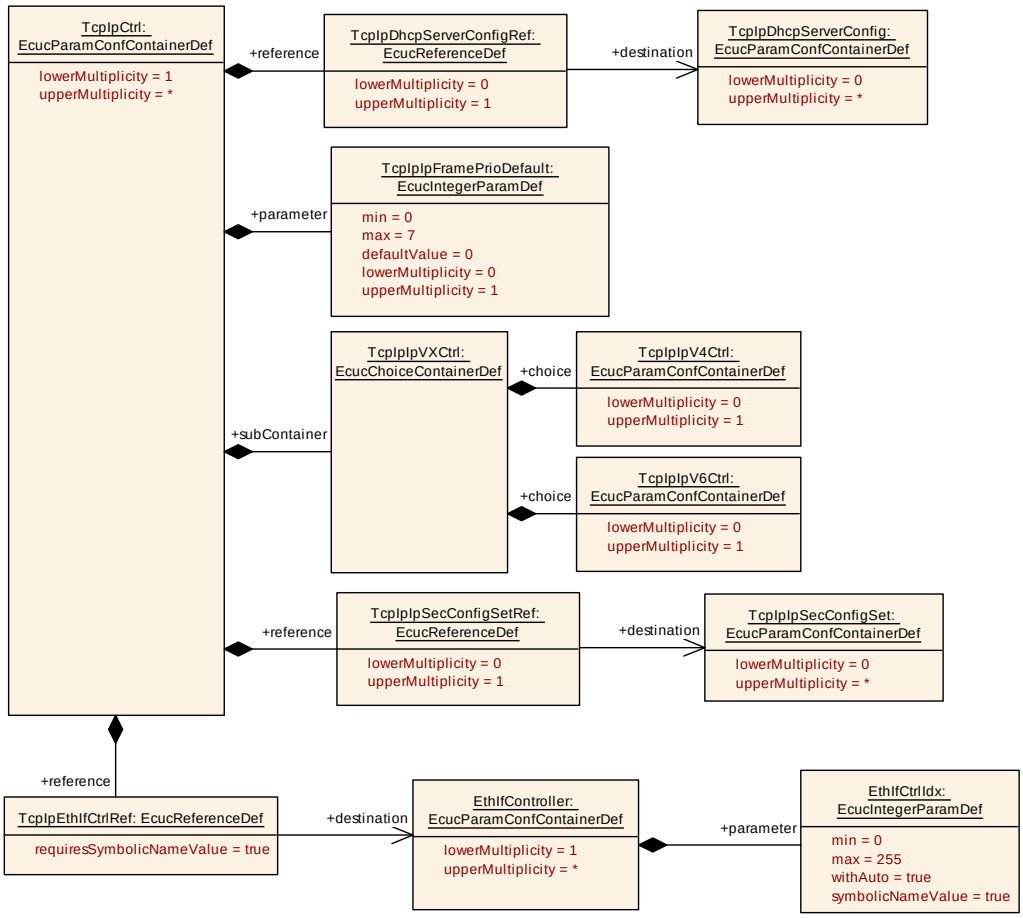

Figure 10.5: TcpIpCtrl

10.2.7 TcpIpCtrl

SWS Item	[ECUC_TcpIp_00021]
Container Name	TcpIpCtrl
Parent Container	TcpIpConfig
Description	Specifies the EthIf controller used for IP communication.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00081]
Parameter Name	TcpIpIpFramePrioDefault
Parent Container	TcpIpCtrl
Description	Specifies the default value for the priority for all outgoing frames. Note: the value can be changed for each socket individually via TcpIp_ChangeParameter() service. If this optional parameter is not available, 0 is used as default priority.
Multiplicity	0..1
Type	EcucIntegerParamDef
Range	0 .. 7

▽

△

Default value	0		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00195]**		
Parameter Name	TcpIpDhcpServerConfigRef		
Parent Container	TcpIpCtrl		
Description	Reference to a TcpIpDhcpServerConfig which shall be used for this controller setting (VLAN).		
Multiplicity	0..1		
Type	Reference to TcpIpDhcpServerConfig		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00041]**		
Parameter Name	TcpIpEthIfCtrlRef		
Parent Container	TcpIpCtrl		
Description	Reference to EthIf controller where the IP address shall be assigned.		
Multiplicity	1		
Type	Symbolic name reference to EthIfController		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00315]**		
Parameter Name	TcpIpIpSecConfigSetRef		
Parent Container	TcpIpCtrl		
Description	Reference to set of SDP entries which shall be used for IPsec.		
Multiplicity	0..1		
Type	Reference to TcpIpIpSecConfigSet		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

▽

△

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpIpVXCtrl	1	Specifies whether this controller is an Internet Protocol version 4 (IPv4) or Internet Protocol version 6 (IPv4) instance.

10.2.8 TcpIpIpVXCtrl

SWS Item	**[ECUC_TcpIp_00094]**
Choice Container Name	TcpIpIpVXCtrl
Parent Container	TcpIpCtrl
Description	Specifies whether this controller is an Internet Protocol version 4 (IPv4) or Internet Protocol version 6 (IPv4) instance.

Container Choices		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpIpV4Ctrl	0..1	Specifies an Internet Protocol version 4 (IPv4) instance.
TcpIpIpV6Ctrl	0..1	Specifies an Internet Protocol version 6 (IPv6) instance.

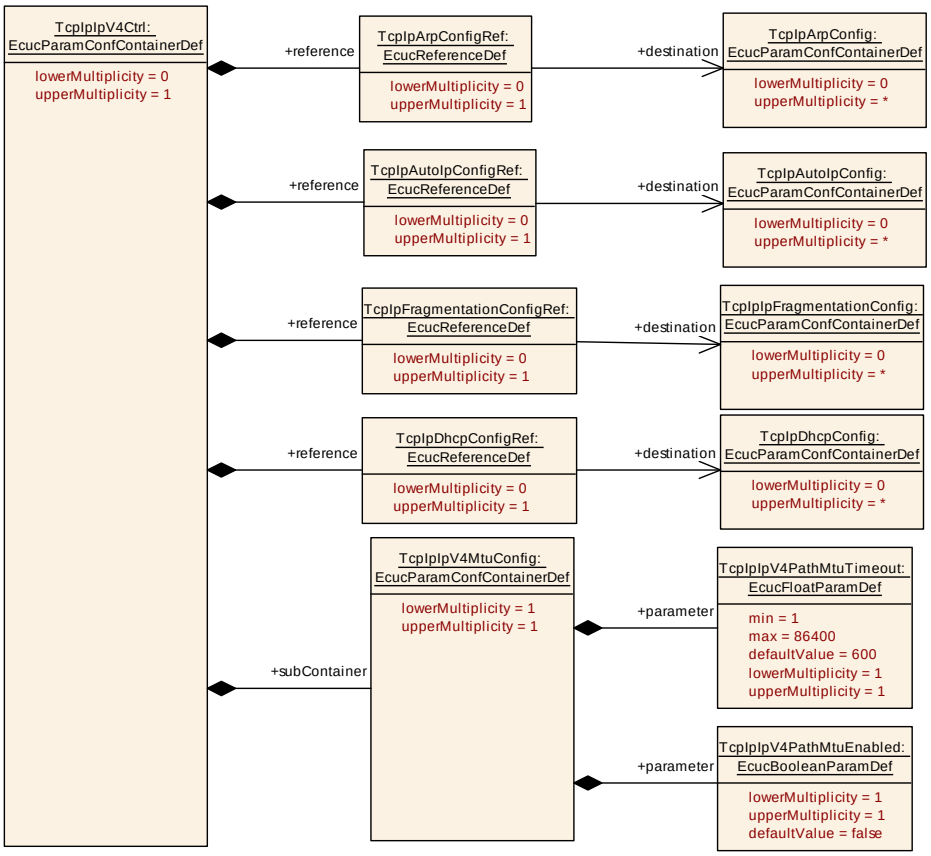

Figure 10.6: TcpIpIpV4Ctrl

10.2.9 TcpIpIpV4Ctrl

SWS Item	**[ECUC_TcpIp_00166]**
Container Name	TcpIpIpV4Ctrl
Parent Container	TcpIpIpVXCtrl
Description	Specifies an Internet Protocol version 4 (IPv4) instance.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00097]**		
Parameter Name	TcpIpArpConfigRef		
Parent Container	TcpIpIpV4Ctrl		
Description	Reference to ARP configuration for this IPv4 instance. (Multiple IPv4 instances may use the same configuration container but will operate independently)		
Multiplicity	0..1		
Type	Reference to TcpIpArpConfig		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	

△

	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00098]**		
Parameter Name	TcpIpAutoIpConfigRef		
Parent Container	TcpIpIpV4Ctrl		
Description	Reference to AutoIp configuration for this IPv4 instance. (Multiple IPv4 instances may use the same configuration container but will operate independently)		
Multiplicity	0..1		
Type	Reference to TcpIpAutoIpConfig		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00100]**		
Parameter Name	TcpIpDhcpConfigRef		
Parent Container	TcpIpIpV4Ctrl		
Description	Reference to DHCP configuration for this IPv4 instance. (Multiple IPv4 instances may use the same configuration container but will operate independently)		
Multiplicity	0..1		
Type	Reference to TcpIpDhcpConfig		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00099]**
Parameter Name	TcpIpFragmentationConfigRef
Parent Container	TcpIpIpV4Ctrl
Description	Reference to Fragmentation configuration for this IPv4 instance. (Multiple IPv4 instances may use the same configuration container but will operate independently)
Multiplicity	0..1
Type	Reference to TcpIpIpFragmentationConfig

▽

△

Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpIpV4MtuConfig	1	This container specifies the Maximum Transmission Unit parameters for this IPv4 instance.

10.2.10 TcpIpIpV4MtuConfig

SWS Item	**[ECUC_TcpIp_00209]**
Container Name	TcpIpIpV4MtuConfig
Parent Container	TcpIpIpV4Ctrl
Description	This container specifies the Maximum Transmission Unit parameters for this IPv4 instance.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00211]**		
Parameter Name	TcpIpIpV4PathMtuEnabled		
Parent Container	TcpIpIpV4MtuConfig		
Description	If enabled the IPv4 processes incoming ICMPv4 "Packet Too Big" messages and stores a MTU value for each destination address.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00210]**
Parameter Name	TcpIpIpV4PathMtuTimeout
Parent Container	TcpIpIpV4MtuConfig
Description	If this value is >0 the IpV4 will reset the MTU value stored for each destination after n seconds. see [RFC1191 6.3. Purging stale PMTU information] Default: 600 seconds (10 minutes)
Multiplicity	1

▽

△

Type	EcucFloatParamDef		
Range	[1 .. 86400]		
Default value	600		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

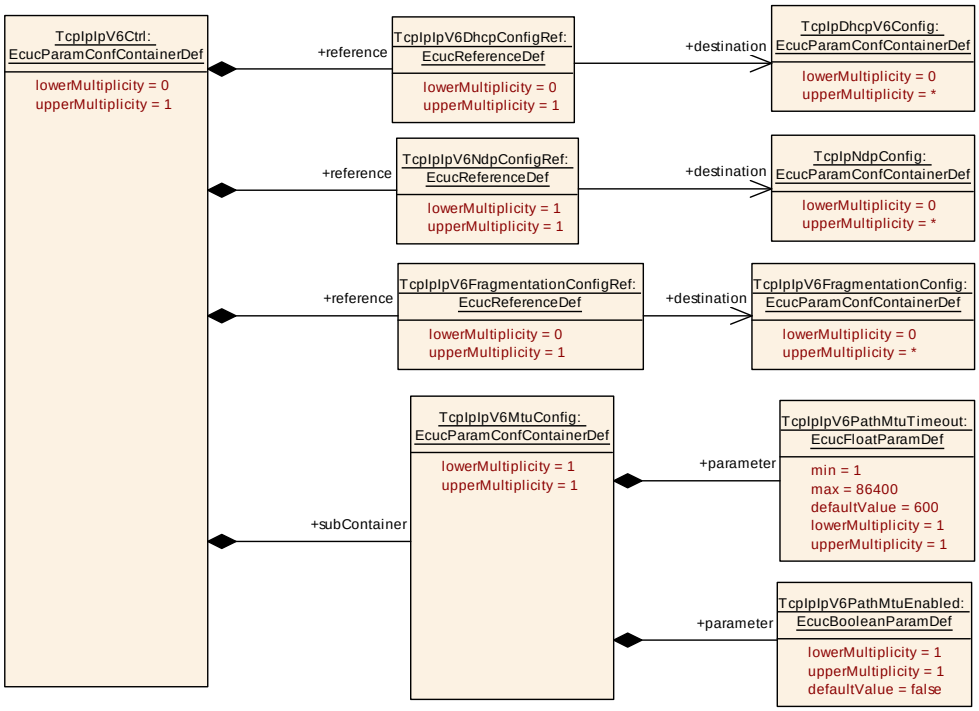

Figure 10.7: TcpIpIpV6Ctrl

10.2.11 TcpIpIpV6Ctrl

SWS Item	**[ECUC_TcpIp_00096]**
Container Name	TcpIpIpV6Ctrl
Parent Container	TcpIpIpVXCtrl
Description	Specifies an Internet Protocol version 6 (IPv6) instance.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00101]		
Parameter Name	TcpIpIpV6DhcpConfigRef		
Parent Container	TcpIpIpV6Ctrl		
Description	Reference to DHCPv6 configuration. (Multiple IPv6 instances may use the same configuration container but will operate independently)		
Multiplicity	0..1		
Type	Reference to TcpIpDhcpV6Config		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00103]		
Parameter Name	TcpIpIpV6FragmentationConfigRef		
Parent Container	TcpIpIpV6Ctrl		
Description	Reference to IPv6 Fragmentation Configuration. (Multiple IPv6 instances may use the same configuration container but will operate independently)		
Multiplicity	0..1		
Type	Reference to TcpIpIpV6FragmentationConfig		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00102]		
Parameter Name	TcpIpIpV6NdpConfigRef		
Parent Container	TcpIpIpV6Ctrl		
Description	Reference to Neighbor Discovery Protocol Configuration. (Multiple IPv6 instances may use the same configuration container but will operate independently)		
Multiplicity	1		
Type	Reference to TcpIpNdpConfig		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpIpV6MtuConfig	1	This container specifies the Maximum Transmission Unit parameters for this IPv6 instance.

10.2.12 TcpIpIpV6MtuConfig

SWS Item	**[ECUC_TcpIp_00104]**
Container Name	TcpIpIpV6MtuConfig
Parent Container	TcpIpIpV6Ctrl
Description	This container specifies the Maximum Transmission Unit parameters for this IPv6 instance.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00107]**		
Parameter Name	TcpIpIpV6PathMtuEnabled		
Parent Container	TcpIpIpV6MtuConfig		
Description	If enabled the IPv6 processes incoming ICMPv6 "Packet Too Big" messages and stores a MTU value for each destination address. See RFC1981 "Path MTU Discovery for IP version 6" for details about PathMTU.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00105]**		
Parameter Name	TcpIpIpV6PathMtuTimeout		
Parent Container	TcpIpIpV6MtuConfig		
Description	If this value is >0 the IpV6 will reset the MTU value stored for each destination after n seconds. see [RFC1981 5.3. Purging stale PMTU information] Default: 600 seconds (10 minutes)		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[1 .. 86400]		
Default value	600		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

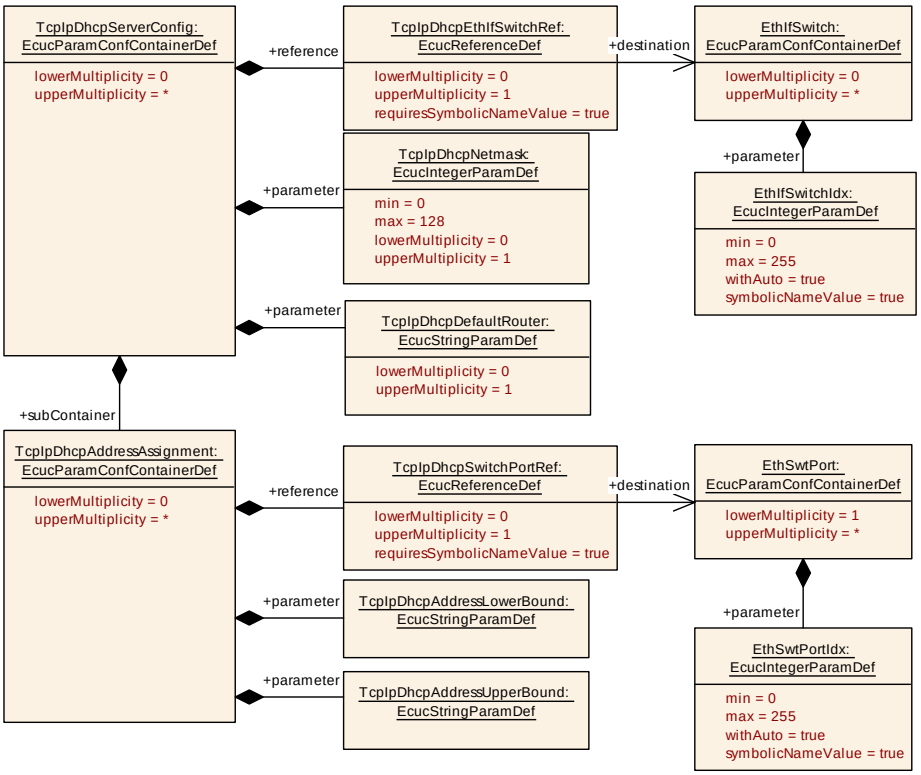

Figure 10.8: TcpIpDhcpServer

10.2.13 TcpIpDhcpServerConfig

SWS Item	**[ECUC_TcpIp_00187]**		
Container Name	TcpIpDhcpServerConfig		
Parent Container	TcpIpConfig		
Description	Specifies the configuration parameters of the DHCP Server sub-module.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Configuration Parameters			

SWS Item	**[ECUC_TcpIp_00190]**
Parameter Name	TcpIpDhcpDefaultRouter
Parent Container	TcpIpDhcpServerConfig
Description	IP address of default router (gateway).
Multiplicity	0..1
Type	EcucStringParamDef
Default value	–
Regular Expression	–
Post-Build Variant Multiplicity	true

▽

△

Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00189]**		
Parameter Name	TcpIpDhcpNetmask		
Parent Container	TcpIpDhcpServerConfig		
Description	Network mask of IPv4 address or address prefix of IPv6 address in CIDR Notation, i.e. decimal value between 0 and 32 (IPv4) or 0 and 128 (IPv6) that describes the number of significant bits defining the network number or prefix of an IP address.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 128		
Default value	–		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00188]**		
Parameter Name	TcpIpDhcpEthIfSwitchRef		
Parent Container	TcpIpDhcpServerConfig		
Description	Reference to EthIfSwitch representation. Optional in case the Dhcp server is operating without an Ethernet switch.		
Multiplicity	0..1		
Type	Symbolic name reference to EthIfSwitch		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpDhcpAddressAssignment	0..*	Defines a Ethernet Switch port based IP address assignment.

10.2.14 TcpIpDhcpAddressAssignment

SWS Item	[ECUC_TcpIp_00191]		
Container Name	TcpIpDhcpAddressAssignment		
Parent Container	TcpIpDhcpServerConfig		
Description	Defines a Ethernet Switch port based IP address assignment.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Configuration Parameters			

SWS Item	[ECUC_TcpIp_00193]		
Parameter Name	TcpIpDhcpAddressLowerBound		
Parent Container	TcpIpDhcpAddressAssignment		
Description	The lower bound IP address which shall be assigned. If lower bound and upper bound are identical exactly this IP address shall be assigned.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00194]		
Parameter Name	TcpIpDhcpAddressUpperBound		
Parent Container	TcpIpDhcpAddressAssignment		
Description	The upper bound IP address which shall be assigned. If lower bound and upper bound are identical exactly this IP address shall be assigned.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00192]
Parameter Name	TcpIpDhcpSwitchPortRef
Parent Container	TcpIpDhcpAddressAssignment
Description	Reference to Ethernet Switch port. Optional in case the Dhcp server is operating without an Ethernet switch.

▽

△

Multiplicity	0..1		
Type	Symbolic name reference to EthSwtPort		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

No Included Containers

10.2.15 TcpIpDuplicateAddressDetectionConfig

SWS Item	**[ECUC_TcpIp_00214]**
Container Name	TcpIpDuplicateAddressDetectionConfig
Parent Container	TcpIpConfig
Description	Specifies the DAD callout function.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00216]**		
Parameter Name	TcpIpDuplicateAddressDetectionCalloutName		
Parent Container	TcpIpDuplicateAddressDetectionConfig		
Description	This parameter defines the name of the DAD callout function <Up_DADAddress Conflict>.		
Multiplicity	1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

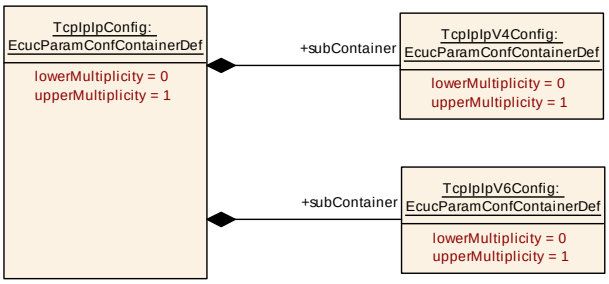

Figure 10.9: TcpIpIpConfig

10.2.16 TcpIpIpConfig

SWS Item	[ECUC_TcpIp_00022]
Container Name	TcpIpIpConfig
Parent Container	TcpIpConfig
Description	Specifies the configuration parameters of the IP (Internet Protocol) sub-module
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpIpV4Config	0..1	Specifies the configuration parameters of the IPv4 (Internet Protocol version 4) sub-module.
TcpIpIpV6Config	0..1	Specifies the configuration parameters of the IPv6 (Internet Protocol version 6) sub-module.

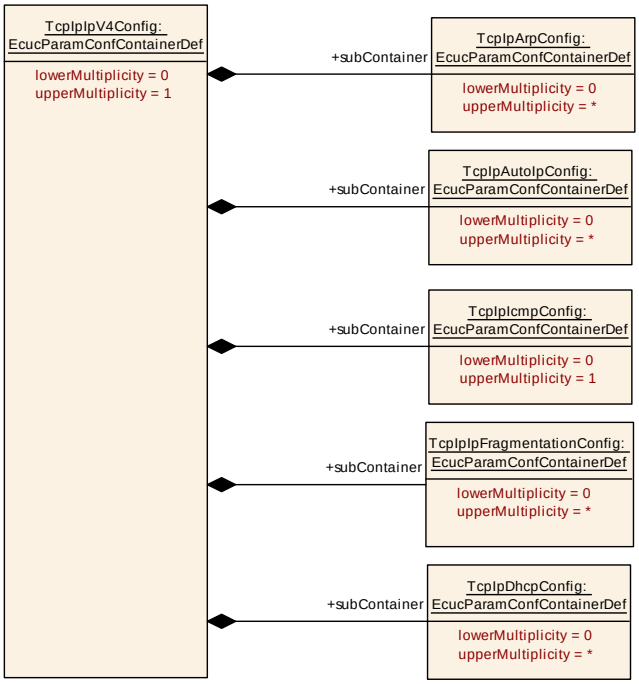

Figure 10.10: TcpIpV4Config

10.2.17 TcpIpIpV4Config

SWS Item	[ECUC_TcpIp_00095]
Container Name	TcpIpIpV4Config
Parent Container	TcpIpIpConfig
Description	Specifies the configuration parameters of the IPv4 (Internet Protocol version 4) sub-module.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpArpConfig	0..*	Specifies the configuration parameters of the ARP (Address Resolution Protocol) sub-module.
TcpIpAutoIpConfig	0..*	Specifies the configuration parameters of the Auto-IP (automatic private IP addressing) sub-module.
TcpIpDhcpConfig	0..*	Specifies the configuration parameters of the DHCPv4. This container may be referenced by multiple IPv4 instances if they shall use the same configuration. This container may have multiple instances if different configurations are required for different IPv4 instances.
TcpIpIcmpConfig	0..1	Specifies the configuration parameters of the ICMP (Internet Control Message Protocol) sub-module.
TcpIpIpFragmentationConfig	0..*	Specifies the configuration parameters of IPv4 packet fragmentation/reassembly. This container may be referenced by multiple IPv4 instances if they shall use the same configuration. This container may have multiple instances if different configurations are required for different IPv4 instances.

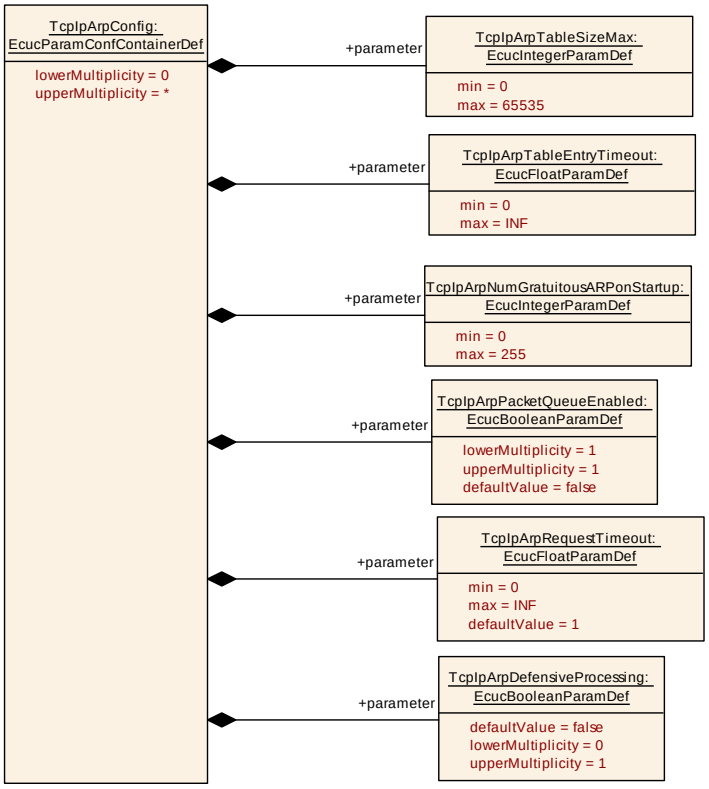

Figure 10.11: TcpIpArpConfig

10.2.18 TcpIpArpConfig

SWS Item	[ECUC_TcpIp_00023]
Container Name	TcpIpArpConfig
Parent Container	TcpIpIpV4Config
Description	Specifies the configuration parameters of the ARP (Address Resolution Protocol) sub-module.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00326]		
Parameter Name	TcpIpArpDefensiveProcessing		
Parent Container	TcpIpArpConfig		
Description	If enabled the ARP shall only process ARP replies which are received in reaction to a previously transmitted ARP request as well as skipping updates to the ARP table based on received Gratuitous ARP packets. If disabled all ARP packets shall be processed as specified in IETF RFC 826.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00054]		
Parameter Name	TcpIpArpNumGratuitousARPonStartup		
Parent Container	TcpIpArpConfig		
Description	Specifies the number of gratuitous ARP replies which shall be sent on assignment of a new IP address.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00170]
Parameter Name	TcpIpArpPacketQueueEnabled
Parent Container	TcpIpArpConfig
Description	Enables (TRUE) or disables (FALSE) support of the ARP Packet Queue according to IETF RFC 1122, section 2.3.2.2.

△

Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00218]**		
Parameter Name	TcpIpArpRequestTimeout		
Parent Container	TcpIpArpConfig		
Description	Specifies a timeout in seconds for the validity of ARP requests. After the transmission of an ARP request the TcpIp shall skip the transmission of any further ARP requests to the same destination within a duration of TcpIpArpRequestTimeout seconds. (IETF RFC 1122, section 2.3.2.1) The value for this parameter shall be an integral multiple of TcpIpMainFunctionPeriod or 0. If this parameter set to 0 this features is disabled and no delay between ARP requests is enforced.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. INF[		
Default value	1		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00053]**		
Parameter Name	TcpIpArpTableEntryTimeout		
Parent Container	TcpIpArpConfig		
Description	Timeout in seconds after which an unused ARP entry is removed.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	]0 .. INF]		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00052]**		
Parameter Name	TcpIpArpTableSizeMax		
Parent Container	TcpIpArpConfig		
Description	Maximum number of entries in the ARP table.		
Multiplicity	1		
Type	EcucIntegerParamDef		

▽

△

Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

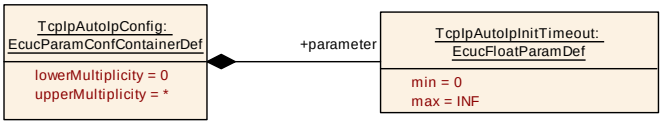

Figure 10.12: TcpIpAutoIpConfig

10.2.19 TcpIpAutoIpConfig

SWS Item	**[ECUC_TcpIp_00028]**
Container Name	TcpIpAutoIpConfig
Parent Container	TcpIpIpV4Config
Description	Specifies the configuration parameters of the Auto-IP (automatic private IP addressing) sub-module.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00074]**		
Parameter Name	TcpIpAutoIpInitTimeout		
Parent Container	TcpIpAutoIpConfig		
Description	The time in seconds Auto-IP waits at startup, before beginning with ARP probing. This delay is used to give DHCP time to acquire a lease in case a DHCP server is present.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. INF]		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

No Included Containers

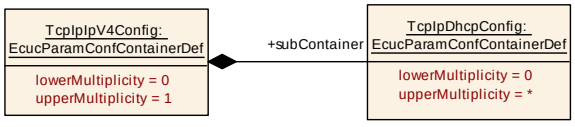

Figure 10.13: TcpIpDhcpConfig

10.2.20 TcpIpDhcpConfig

SWS Item	[ECUC_TcpIp_00167]
Container Name	TcpIpDhcpConfig
Parent Container	TcpIpIpV4Config
Description	Specifies the configuration parameters of the DHCPv4. This container may be referenced by multiple IPv4 instances if they shall use the same configuration. This container may have multiple instances if different configurations are required for different IPv4 instances.
Configuration Parameters	

No Included Containers

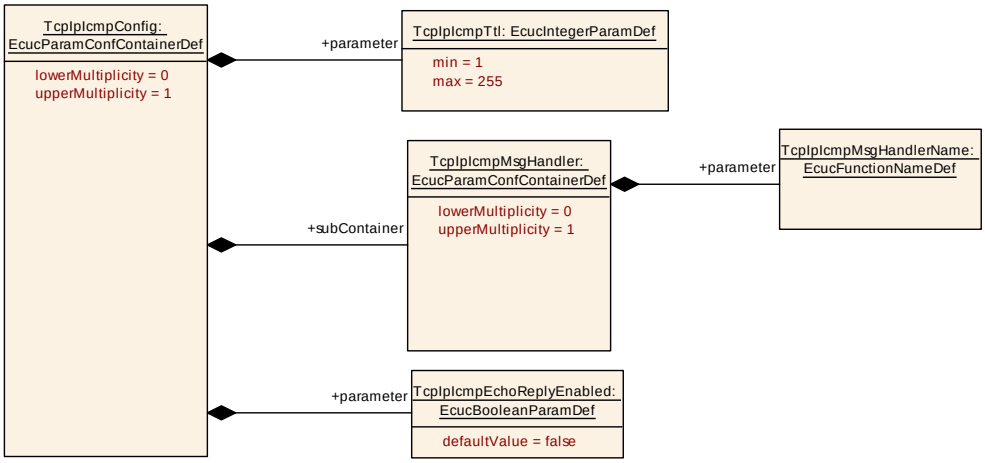

Figure 10.14: TcpIpIcmpConfig

10.2.21 TcpIpIcmpConfig

SWS Item	[ECUC_TcpIp_00024]
Container Name	TcpIpIcmpConfig
Parent Container	TcpIpIpV4Config
Description	Specifies the configuration parameters of the ICMP (Internet Control Message Protocol) sub-module.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00213]
Parameter Name	TcpIpIcmpEchoReplyEnabled
Parent Container	TcpIpIcmpConfig
Description	Enables or disables transmission of ICMP echo reply message in case of a ICMP echo reception.
Multiplicity	1
Type	EcucBooleanParamDef
Default value	false

▽

△

Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00055]**		
Parameter Name	TcpIpIcmpTtl		
Parent Container	TcpIpIcmpConfig		
Description	Default Time-to-live value of outgoing ICMP packets.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpIcmpMsgHandler	0..1	This container is a subcontainer of TcpIpIcmpConfig and specifies the configuration parameters for the ICMP message handler.

10.2.22 TcpIpIcmpMsgHandler

SWS Item	**[ECUC_TcpIp_00056]**
Container Name	TcpIpIcmpMsgHandler
Parent Container	TcpIpIcmpConfig
Description	This container is a subcontainer of TcpIpIcmpConfig and specifies the configuration parameters for the ICMP message handler.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00057]**
Parameter Name	TcpIpIcmpMsgHandlerName
Parent Container	TcpIpIcmpMsgHandler
Description	This parameter defines the name of the ICMP message handler function <Up_Icmp MsgHandler>.
Multiplicity	1
Type	EcucFunctionNameDef
Default value	–
Regular Expression	–
Post-Build Variant Value	false

▽

△

Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

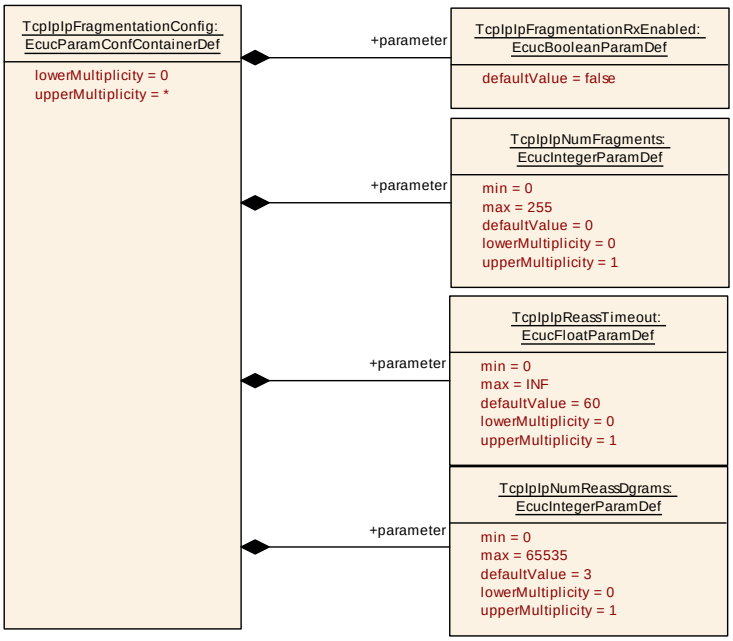

Figure 10.15: TcpIpIpFragmentationConfig

10.2.23 TcpIpIpFragmentationConfig

SWS Item	**[ECUC_TcpIp_00108]**
Container Name	TcpIpIpFragmentationConfig
Parent Container	TcpIpIpV4Config
Description	Specifies the configuration parameters of IPv4 packet fragmentation/reassembly. This container may be referenced by multiple IPv4 instances if they shall use the same configuration. This container may have multiple instances if different configurations are required for different IPv4 instances.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00077]**
Parameter Name	TcpIpIpFragmentationRxEnabled
Parent Container	TcpIpIpFragmentationConfig
Description	Enables (TRUE) or disables (FALSE) support for reassembling of incoming datagrams that are fragmented according to IETF RFC 815 (IP Datagram Reassembly Algorithms).

▽

△

Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00078]**		
Parameter Name	TcpIpIpNumFragments		
Parent Container	TcpIpIpFragmentationConfig		
Description	Specifies the maximum number of IP fragments per datagram. Note: this parameter is only relevant if TcpIpIpFragmentationRxEnabled is TRUE.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	0		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: TcpIpIpFragmentationRxEnabled		

SWS Item	**[ECUC_TcpIp_00080]**		
Parameter Name	TcpIpIpNumReassDgrams		
Parent Container	TcpIpIpFragmentationConfig		
Description	Specifies the maximum number of fragmented IP datagrams that can be reassembled in parallel. Note: this parameter is only relevant if TcpIpIpFragmentationRxEnabled is TRUE.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	3		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD

▽

△

Scope / Dependency	scope: local dependency: TcpIpIpFragmentationRxEnabled

SWS Item	**[ECUC_TcpIp_00079]**		
Parameter Name	TcpIpIpReassTimeout		
Parent Container	TcpIpIpFragmentationConfig		
Description	Specifies the timeout in [s] after which an incomplete datagram gets discarded. Note: this parameter is only relevant if TcpIpIpFragmentationRxEnabled is TRUE.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[0 .. INF]		
Default value	60		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: TcpIpIpFragmentationRxEnabled		

No Included Containers

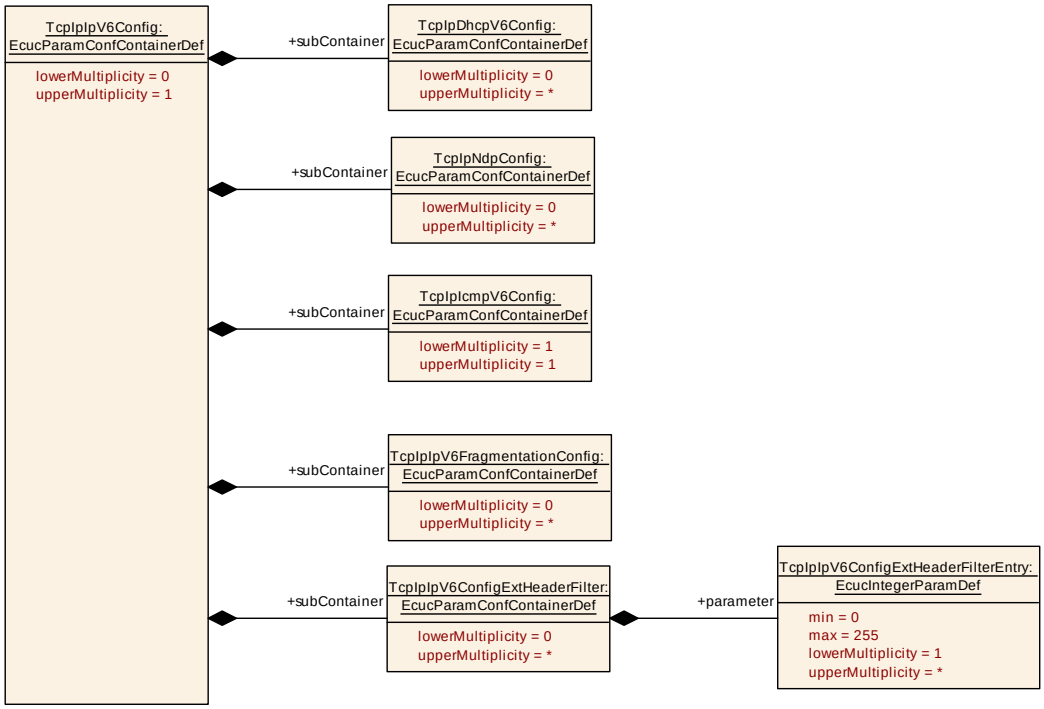

Figure 10.16: TcpIpV6Config

10.2.24 TcpIpIpV6Config

SWS Item	[ECUC_TcpIp_00168]
Container Name	TcpIpIpV6Config
Parent Container	TcpIpIpConfig
Description	Specifies the configuration parameters of the IPv6 (Internet Protocol version 6) sub-module.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpDhcpV6Config	0..*	Specifies the configuration parameters of the DHCPv6. This container may be referenced by multiple IPv6 instances if they shall use the same configuration. This container may have multiple instances if different configurations are required for different IPv6 instances.
TcpIpIcmpV6Config	1	Specifies the configuration parameters of the ICMPv6 (Internet Control Message Protocol for IPv6) sub-module.
TcpIpIpV6ConfigExtHeaderFilter	0..*	This container describes the white list for the filtering of IPv6 extension headers, i.e. frames containing IPv6 extension headers not listed here shall be silently dropped.
TcpIpIpV6FragmentationConfig	0..*	Specifies the configuration parameters of IPv6 packet fragmentation/reassembly. This container may be referenced by multiple IPv6 instances if they shall use the same configuration. This container may have multiple instances if different configurations are required for different IPv6 instances.
TcpIpNdpConfig	0..*	Specifies the configuration parameters of the Neighbor Discovery Protocol for IPv6 This container may be referenced by multiple IPv6 instances if they shall use the same configuration. This container may have multiple instances if different configurations are required for different IPv6 instances.

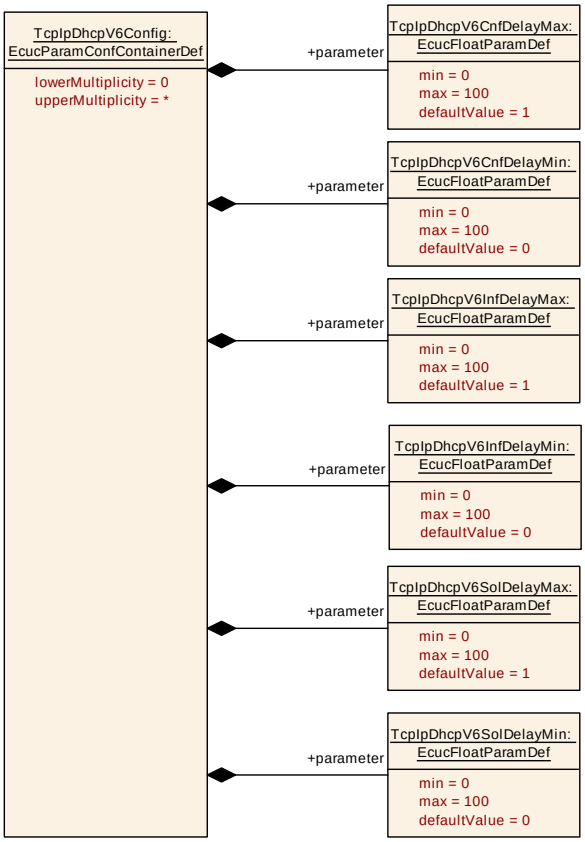

Figure 10.17: TcpIpDhcpV6Config

10.2.25 TcpIpDhcpV6Config

SWS Item	**[ECUC_TcpIp_00110]**
Container Name	TcpIpDhcpV6Config
Parent Container	TcpIpIpV6Config
Description	Specifies the configuration parameters of the DHCPv6. This container may be referenced by multiple IPv6 instances if they shall use the same configuration. This container may have multiple instances if different configurations are required for different IPv6 instances.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00116]**	
Parameter Name	TcpIpDhcpV6CnfDelayMax	
Parent Container	TcpIpDhcpV6Config	
Description	Maximum delay (s) before sending the first Confirm message. If this value is bigger than the previous minimum delay value a random delay will be chosen from the interval.	
Multiplicity	1	
Type	EcucFloatParamDef	
Range	[0 .. 100]	
Default value	1	

▽

△

Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00117]**		
Parameter Name	TcpIpDhcpV6CnfDelayMin		
Parent Container	TcpIpDhcpV6Config		
Description	Minimum delay (s) before the first Confirm message will be sent.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 100]		
Default value	0		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00118]**		
Parameter Name	TcpIpDhcpV6InfDelayMax		
Parent Container	TcpIpDhcpV6Config		
Description	Maximum delay (s) before sending the first Information Request message. If this value is bigger than the previous minimum delay value a random delay will be chosen from the interval.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 100]		
Default value	1		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00119]**		
Parameter Name	TcpIpDhcpV6InfDelayMin		
Parent Container	TcpIpDhcpV6Config		
Description	Minimum delay (s) before the first Information Request message will be sent.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 100]		
Default value	0		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	

▽

△

	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00120]**		
Parameter Name	TcpIpDhcpV6SolDelayMax		
Parent Container	TcpIpDhcpV6Config		
Description	Maximum delay (s) before sending the first Solicit message. If this value is bigger than the previous minimum delay value a random delay will be chosen from the interval.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 100]		
Default value	1		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00121]**		
Parameter Name	TcpIpDhcpV6SolDelayMin		
Parent Container	TcpIpDhcpV6Config		
Description	Minimum delay (s) before the first Solicit message will be sent.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 100]		
Default value	0		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

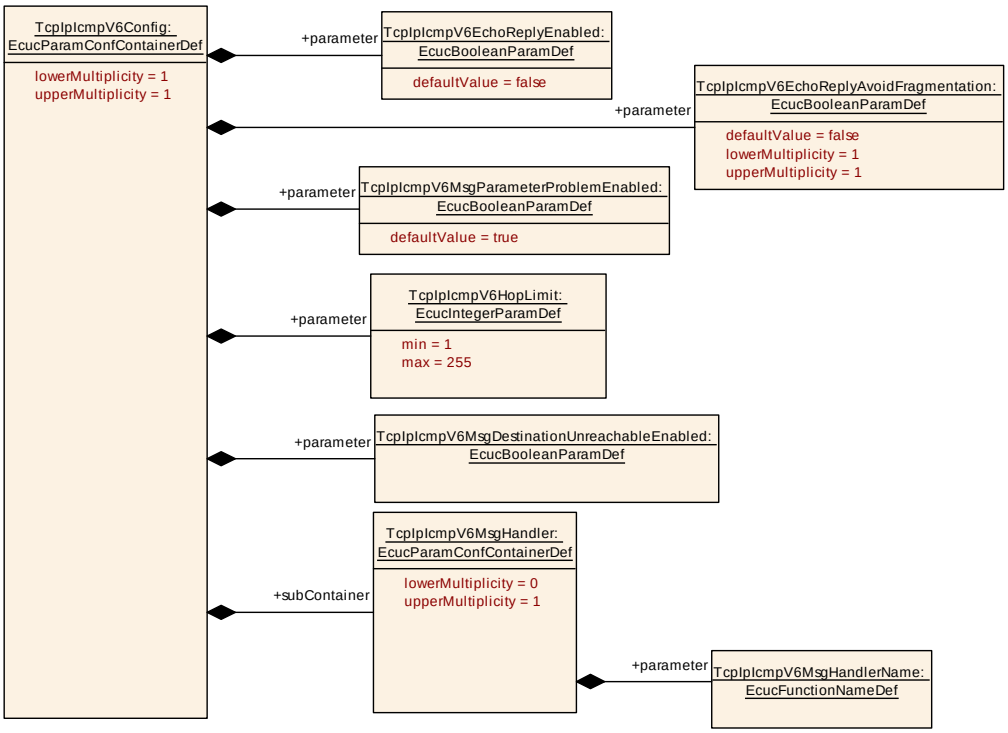

Figure 10.18: TcpIpIcmpV6Config

10.2.26 TcpIpIcmpV6Config

SWS Item	**[ECUC_TcpIp_00113]**
Container Name	TcpIpIcmpV6Config
Parent Container	TcpIpIpV6Config
Description	Specifies the configuration parameters of the ICMPv6 (Internet Control Message Protocol for IPv6) sub-module.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00212]**		
Parameter Name	TcpIpIcmpV6EchoReplyAvoidFragmentation		
Parent Container	TcpIpIcmpV6Config		
Description	If enabled, the stack will respond only to incoming ICMPv6 Echo Requests (Pings) that fit the MTU of the respective interface, i.e. can be transmitted without IPv6 fragmentation. Only relevant if TcpIpIcmpV6EchoReplyEnabled is enabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

△

Scope / Dependency	scope: local dependency: TcpIpIcmpV6EchoReplyEnabled

SWS Item	**[ECUC_TcpIp_00149]**		
Parameter Name	TcpIpIcmpV6EchoReplyEnabled		
Parent Container	TcpIpIcmpV6Config		
Description	If enabled, the stack will respond to incoming ICMPv6 Echo Requests (Pings).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00152]**		
Parameter Name	TcpIpIcmpV6HopLimit		
Parent Container	TcpIpIcmpV6Config		
Description	Default Hop-Limit value of outgoing ICMPv6 packets.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00153]**		
Parameter Name	TcpIpIcmpV6MsgDestinationUnreachableEnabled		
Parent Container	TcpIpIcmpV6Config		
Description	Dis/Enables transmission of Destination Unreachable Messages		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00151]**
Parameter Name	TcpIpIcmpV6MsgParameterProblemEnabled
Parent Container	TcpIpIcmpV6Config

▽

△

Description	If enabled an ICMPv6 parameter problem message will be sent if a received packet has been dropped due to unknown options or headers that are found in the packet. [RFC8200 4. IPv6 Extension Headers]		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpIcmpV6MsgHandler	0..1	This container is a subcontainer of TcpIpIcmpConfig and specifies the configuration parameters for the ICMPv6 message handler.

10.2.27 TcpIpIcmpV6MsgHandler

SWS Item	**[ECUC_TcpIp_00154]**
Container Name	TcpIpIcmpV6MsgHandler
Parent Container	TcpIpIcmpV6Config
Description	This container is a subcontainer of TcpIpIcmpConfig and specifies the configuration parameters for the ICMPv6 message handler.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00156]**		
Parameter Name	TcpIpIcmpV6MsgHandlerName		
Parent Container	TcpIpIcmpV6MsgHandler		
Description	This parameter defines the name of the ICMP message handler function <Up_Icmp MsgHandler>.		
Multiplicity	1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.2.28 TcpIpIpV6ConfigExtHeaderFilter

SWS Item	**[ECUC_TcpIp_00198]**
Container Name	TcpIpIpV6ConfigExtHeaderFilter
Parent Container	TcpIpIpV6Config
Description	This container describes the white list for the filtering of IPv6 extension headers, i.e. frames containing IPv6 extension headers not listed here shall be silently dropped.
Post-Build Variant Multiplicity	false
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00199]**		
Parameter Name	TcpIpIpV6ConfigExtHeaderFilterEntry		
Parent Container	TcpIpIpV6ConfigExtHeaderFilter		
Description	IPv6 Extension Header type allowed by this filter.		
Multiplicity	1..*		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

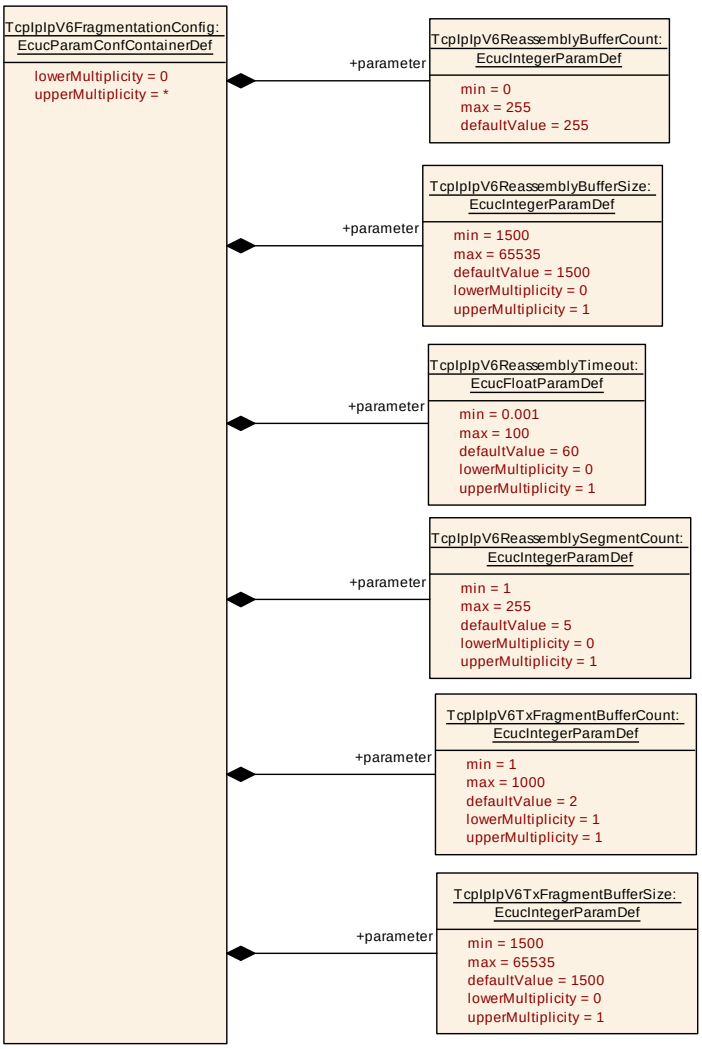

Figure 10.19: TcpIpIpV6FragmentationConfig

10.2.29 TcpIpIpV6FragmentationConfig

SWS Item	**[ECUC_TcpIp_00114]**
Container Name	TcpIpIpV6FragmentationConfig
Parent Container	TcpIpIpV6Config
Description	Specifies the configuration parameters of IPv6 packet fragmentation/reassembly. This container may be referenced by multiple IPv6 instances if they shall use the same configuration. This container may have multiple instances if different configurations are required for different IPv6 instances.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00157]**
Parameter Name	TcpIpIpV6ReassemblyBufferCount
Parent Container	TcpIpIpV6FragmentationConfig

▽

△

Description	Number of buffers that can be used for fragment reassembly. In case of a reassembly error or if not all fragments are received in time this buffer will be blocked until the specified "Fragment Reassembly Timeout" has been exceeded. A value of 0 disables fragment reassembly. [RFC8200 5. Packet Size Issues] "In order to send a packet larger than a path's MTU, a node may use the IPv6 Fragment header to fragment the packet at the source and have it reassembled at the destination(s). However, the use of such fragmentation is discouraged in any application that is able to adjust its packets to fit the measured path MTU (i.e., down to 1280 octets)."		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	255		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

SWS Item	**[ECUC_TcpIp_00158]**		
Parameter Name	TcpIpIpV6ReassemblyBufferSize		
Parent Container	TcpIpIpV6FragmentationConfig		
Description	[RFC8200 5. Packet Size Issues] "A node must be able to accept a fragmented packet that, after reassembly, is as large as 1500 octets. A node is permitted to accept fragmented packets that reassemble to more than 1500 octets."the measured path MTU (i.e., down to 1280 octets)."		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1500 .. 65535		
Default value	1500		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00160]**		
Parameter Name	TcpIpIpV6ReassemblySegmentCount		
Parent Container	TcpIpIpV6FragmentationConfig		
Description	Specifies the maximum number of consecutive data segments that can be managed in each reassembly buffer. If all fragments are received in order, only one segment will be needed. To deal with fragments received out of order this value should be configured bigger than 1.		
Multiplicity	0..1		

▽

△

Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	5		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00159]**		
Parameter Name	TcpIpIpV6ReassemblyTimeout		
Parent Container	TcpIpIpV6FragmentationConfig		
Description	[RFC8200 4.5 Fragment Header] Default: 60 seconds		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[0.001 .. 100]		
Default value	60		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00161]**		
Parameter Name	TcpIpIpV6TxFragmentBufferCount		
Parent Container	TcpIpIpV6FragmentationConfig		
Description	These buffers will be used if the IpV6 receives packets from the upper layer that do not fit into the MTU and thus must be fragmented. A value of 0 disables tx fragmentation. If the upper layer transmits packets that do not fit into the link or path MTU, the IpV6 will split-up the packet into fragments. see "Enable Fragment Reassembly"		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 1000		
Default value	2		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	

▽

△

	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00162]**		
Parameter Name	TcpIpIpV6TxFragmentBufferSize		
Parent Container	TcpIpIpV6FragmentationConfig		
Description	Size of each fragment tx buffer in bytes		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1500 .. 65535		
Default value	1500		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

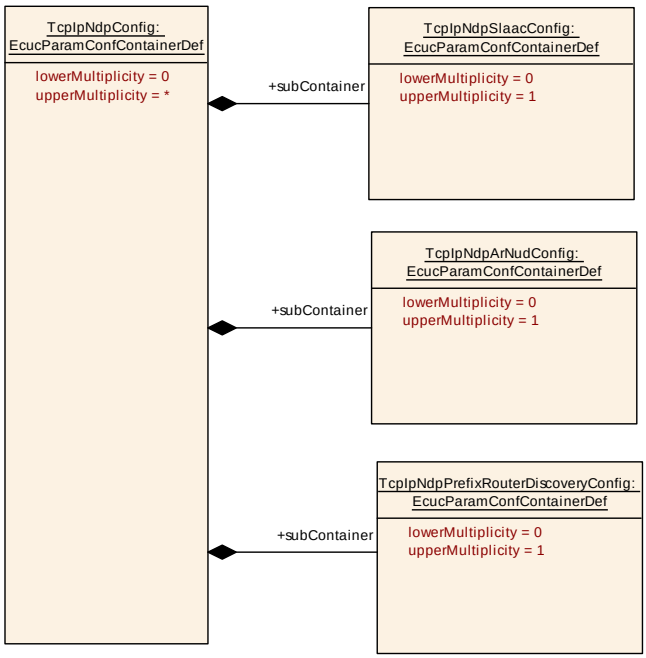

Figure 10.20: TcpIpNdpConfig

10.2.30 TcpIpNdpConfig

SWS Item	[ECUC_TcpIp_00112]
Container Name	TcpIpNdpConfig
Parent Container	TcpIpIpV6Config
Description	Specifies the configuration parameters of the Neighbor Discovery Protocol for IPv6 This container may be referenced by multiple IPv6 instances if they shall use the same configuration. This container may have multiple instances if different configurations are required for different IPv6 instances.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpNdpArNudConfig	0..1	Specifies the configuration parameters for NDP Address Resolution and Neighbor Unreachability Detection.
TcpIpNdpPrefixRouterDiscoveryConfig	0..1	Specifies the configuration parameters for NDP Prefix and Router Discovery.
TcpIpNdpSlaacConfig	0..1	Specifies the configuration parameters for StateLess Address AutoConfiguration.

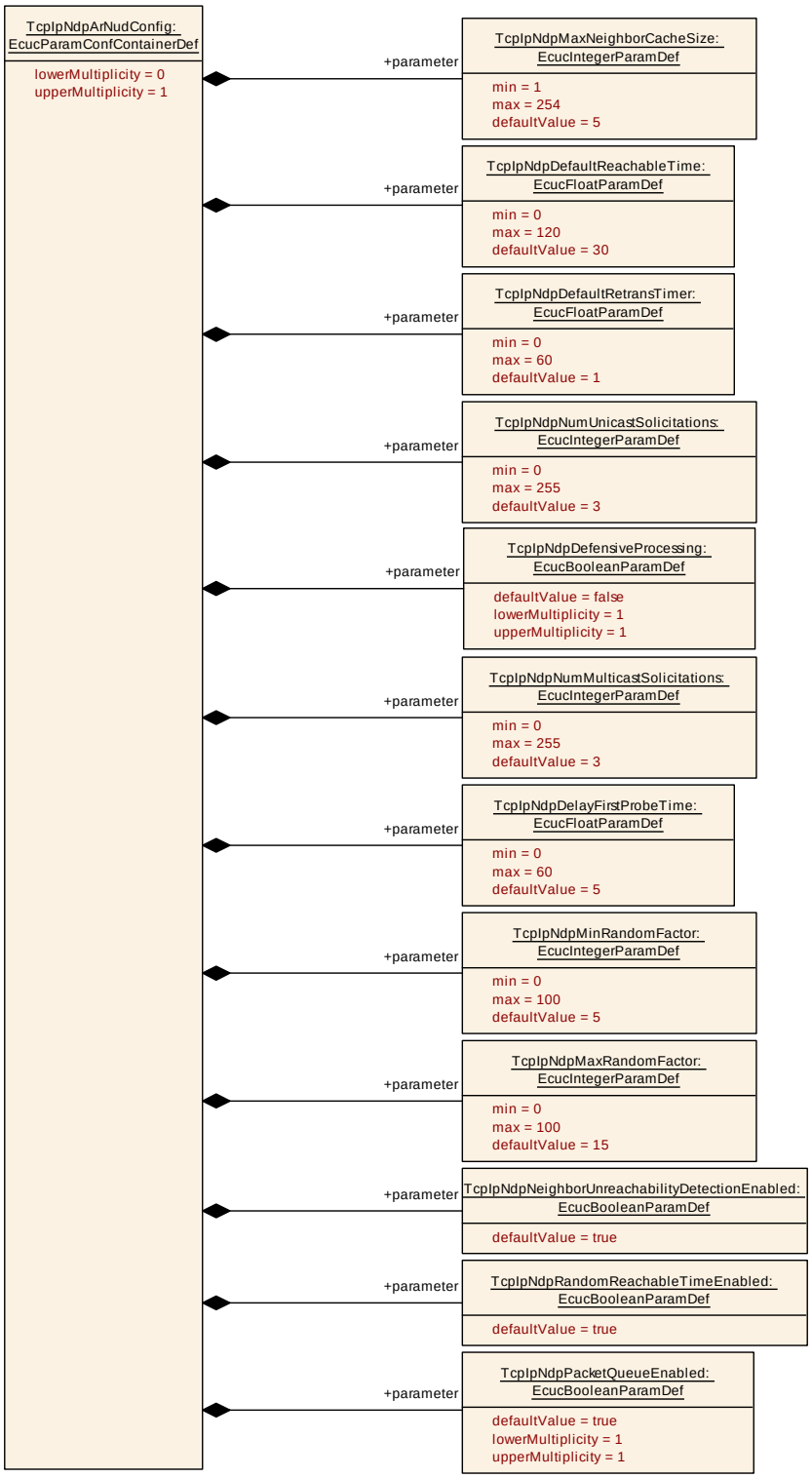

Figure 10.21: TcpIpNdpArNudConfig

10.2.31 TcpIpNdpArNudConfig

SWS Item	[ECUC_TcpIp_00123]
Container Name	TcpIpNdpArNudConfig
Parent Container	TcpIpNdpConfig
Description	Specifies the configuration parameters for NDP Address Resolution and Neighbor Unreachability Detection.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00130]		
Parameter Name	TcpIpNdpDefaultReachableTime		
Parent Container	TcpIpNdpArNudConfig		
Description	Configuration of the ReachableTime (s) specified in [RFC4861 6.3.2. Host Variables]. "The time a neighbor is considered reachable after receiving a reachability confirmation." If "TcpIpNdpDynamicReachableTimeEnabled" is checked, this value may be reconfigured based on received Router Advertisements. Default: REACHABLE_TIME = 30 seconds		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 120]		
Default value	30		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00165]		
Parameter Name	TcpIpNdpDefaultRetransTimer		
Parent Container	TcpIpNdpArNudConfig		
Description	Configures the default value (s) for the RetransTimer variable specified in [RFC4861 6.3.2. Host Variables]. "The time between retransmissions of Neighbor Solicitation messages to a neighbor when resolving the address or when probing the reachability of a neighbor." If "TcpIpNdpDynamicRetransTimeEnabled" is checked, this value may be reconfigured based on received Router Advertisements. Default: RETRANS_TIMER = 1 second		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 60]		
Default value	1		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00201]		
Parameter Name	TcpIpNdpDefensiveProcessing		
Parent Container	TcpIpNdpArNudConfig		
Description	If enabled the NDP shall only process Neighbor Advertisements which are received in reaction to a previously transmitted Neighbor Solicitation as well as skipping updates to the Neighbor Cache based on received Neighbor Solicitations. If disabled all Neighbor Advertisements and Solicitations shall be processed as specified in RFC4861. [RFC4861 7.2.5. Receipt of Neighbor Advertisements]		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00133]		
Parameter Name	TcpIpNdpDelayFirstProbeTime		
Parent Container	TcpIpNdpArNudConfig		
Description	Delay before sending the first NUD probe in (s). [RFC4861 7.3.3. Node Behavior] Default: DELAY_FIRST_PROBE_TIME = 5 seconds		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 60]		
Default value	5		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00129]		
Parameter Name	TcpIpNdpMaxNeighborCacheSize		
Parent Container	TcpIpNdpArNudConfig		
Description	Maximum number of entries in the neighbor cache. [RFC4861 5.1. Conceptual Data Structures]		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 254		
Default value	5		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00135]		
Parameter Name	TcpIpNdpMaxRandomFactor		
Parent Container	TcpIpNdpArNudConfig		
Description	Maximum random factor used for randomization [RFC4861 10. Protocol Constants] Default: 15 (MAX_RANDOM_FACTOR = 1.5)		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 100		
Default value	15		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00134]		
Parameter Name	TcpIpNdpMinRandomFactor		
Parent Container	TcpIpNdpArNudConfig		
Description	Minimum random factor used for randomization [RFC4861 10. Protocol Constants] Default: 5 (MIN_RANDOM_FACTOR = 0.5)		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 100		
Default value	5		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00136]		
Parameter Name	TcpIpNdpNeighborUnreachabilityDetectionEnabled		
Parent Container	TcpIpNdpArNudConfig		
Description	Neighbor Unreachability Detection is used to remove unused entries from the neighbor cache. This feature is a basic feature of NDP and should be turned on.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00132]		
Parameter Name	TcpIpNdpNumMulticastSolicitations		
Parent Container	TcpIpNdpArNudConfig		
Description	Maximum number of multicast solicitations that will be sent when performing address resolution. [RFC4861 7.2.2. Sending Neighbor Solicitations] Default: MAX_MULTICAST_SOLICIT = 3		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	3		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00131]		
Parameter Name	TcpIpNdpNumUnicastSolicitations		
Parent Container	TcpIpNdpArNudConfig		
Description	Maximum number of unicast solicitations that will be sent when performig Neighbor Unreachability Detection. [RFC4861 7.3.3. Node Behavior] Default: MAX_UNICAST_SOLICIT = 3		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	3		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00171]		
Parameter Name	TcpIpNdpPacketQueueEnabled		
Parent Container	TcpIpNdpArNudConfig		
Description	Enables (TRUE) or disables (FALSE) support of a NDP Packet Queue according to IETF RFC 4861, section 7.2.2.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00137]**		
Parameter Name	TcpIpNdpRandomReachableTimeEnabled		
Parent Container	TcpIpNdpArNudConfig		
Description	If enabled the value of ReachableTime will be multiplied with a random value between MIN_RANDOM_FACTOR and MAX_RANDOM_FACTOR in order to prevent multiple nodes from transmitting at exactly the same time [RFC4861 6.3.2. Host Variables / ReachableTime]		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

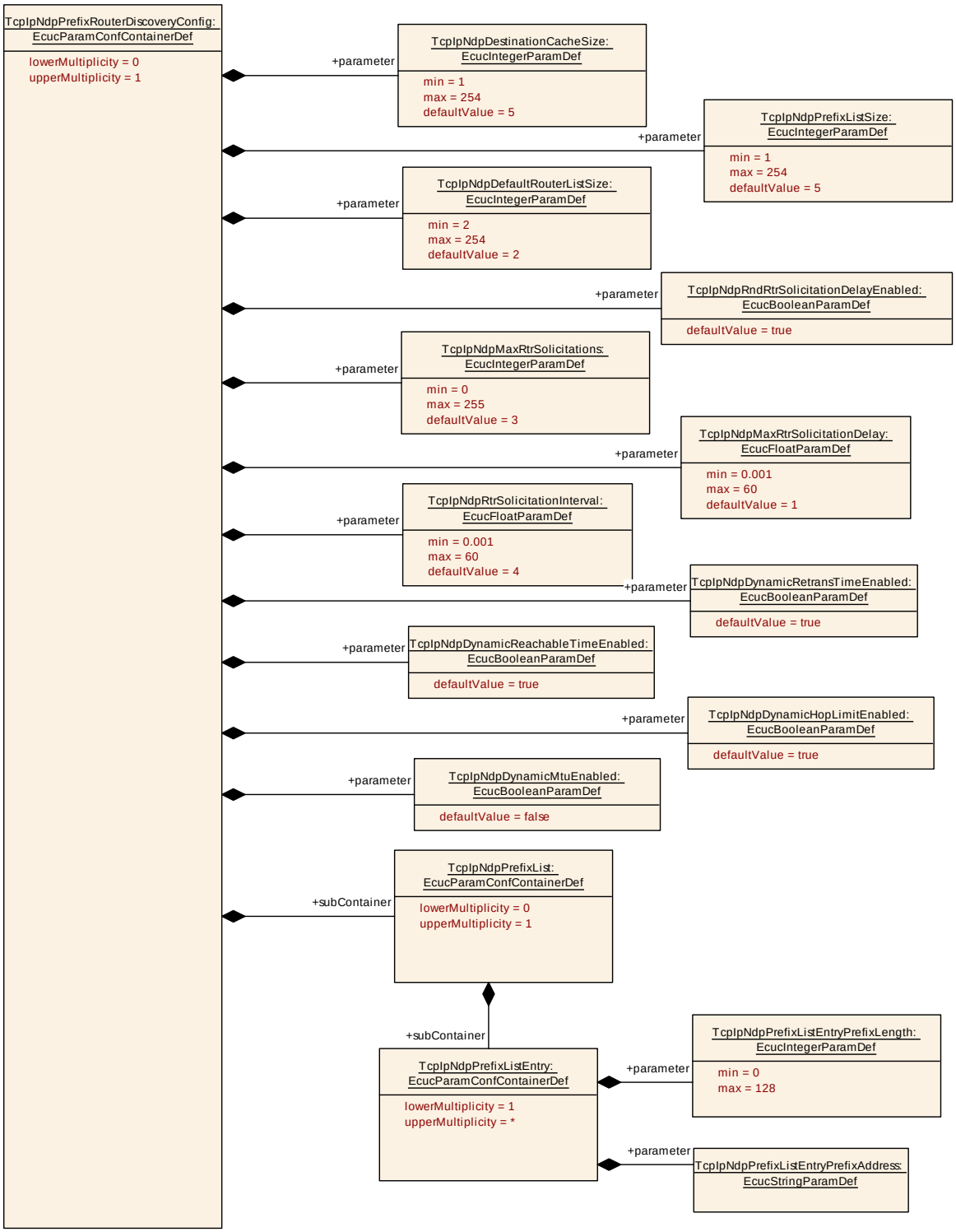

Figure 10.22: TcpIpNdpPrefixRouterDiscoveryConfig

10.2.32 TcpIpNdpPrefixRouterDiscoveryConfig

SWS Item	[ECUC_TcpIp_00124]
Container Name	TcpIpNdpPrefixRouterDiscoveryConfig
Parent Container	TcpIpNdpConfig
Description	Specifies the configuration parameters for NDP Prefix and Router Discovery.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00139]		
Parameter Name	TcpIpNdpDefaultRouterListSize		
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig		
Description	Maximum number of default router entries. [RFC4861 5.1. Conceptual Data Structures]		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	2 .. 254		
Default value	2		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00138]		
Parameter Name	TcpIpNdpDestinationCacheSize		
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig		
Description	Maximum number of entries in the destination cache. [RFC4861 5.1. Conceptual Data Structures]		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 254		
Default value	5		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00147]
Parameter Name	TcpIpNdpDynamicHopLimitEnabled
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig
Description	If enabled the default hop limit may be reconfigured based on received Router Advertisements. [RFC4861 6.3.4. Processing Received Router Advertisements]
Multiplicity	1
Type	EcucBooleanParamDef
Default value	true

▽

△

Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00148]**		
Parameter Name	TcpIpNdpDynamicMtuEnabled		
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig		
Description	Allow dynamic reconfiguration of link MTU via Router Advertisements. [RFC4861 4.6.4. MTU]		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00146]**		
Parameter Name	TcpIpNdpDynamicReachableTimeEnabled		
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig		
Description	If enabled the default Reachable Time value may be reconfigured based on received Router Advertisements. [RFC4861 6.3.4. Processing Received Router Advertisements] Default: Enabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00145]**		
Parameter Name	TcpIpNdpDynamicRetransTimeEnabled		
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig		
Description	If enabled the default Retransmit Timer value may be reconfigured based on received Router Advertisements. [RFC4861 6.3.4. Processing Received Router Advertisements] Default: Enabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	false		

▽

△

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00143]**		
Parameter Name	TcpIpNdpMaxRtrSolicitationDelay		
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig		
Description	Maximum delay before the first Router Solicitation will be sent after interface initialization in (s). [RFC4861 6.3.7. Sending Router Solicitations] Default: MAX_RTR_SOLICITATION_DELAY = 1 second		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0.001 .. 60]		
Default value	1		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00142]**		
Parameter Name	TcpIpNdpMaxRtrSolicitations		
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig		
Description	Maximum number of Router Solicitations that will be sent before the first Router Advertisement has been received. 0 = No Router Solicitations will be sent. This has no impact on handling Router Advertisements. [RFC4861 6.3.7. Sending Router Solicitations] Default: MAX_RTR_SOLICITATIONS = 3 transmissions		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	3		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00140]**
Parameter Name	TcpIpNdpPrefixListSize
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig
Description	Maximum number of entries in the on-link prefix list. [RFC4861 5.1. Conceptual Data Structures]
Multiplicity	1

▽

△

Type	EcucIntegerParamDef		
Range	1 .. 254		
Default value	5		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00141]**		
Parameter Name	TcpIpNdpRndRtrSolicitationDelayEnabled		
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig		
Description	If enabled the first router solicitation will be delayed randomly from [0...MAX_RTR_SOLICITATION_DELAY]. Otherwise the first router solicitation will be sent after exactly MAX_RTR_SOLICITATION_DELAY milliseconds. [RFC4861 6.3.7. Sending Router Solicitations] Default: Enabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00144]**		
Parameter Name	TcpIpNdpRtrSolicitationInterval		
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig		
Description	Interval between consecutive Router Solicitations in (s). [RFC4861 6.3.7. Sending Router Solicitations] Default: RTR_SOLICITATION_INTERVAL = 4 seconds		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0.001 .. 60]		
Default value	4		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpNdpPrefixList	0..1	Specifies a list of prefixes to be treated as "on-link" according to IETF RFC 4861 Section 5.1.

10.2.33 TcpIpNdpPrefixList

SWS Item	[ECUC_TcpIp_00205]
Container Name	TcpIpNdpPrefixList
Parent Container	TcpIpNdpPrefixRouterDiscoveryConfig
Description	Specifies a list of prefixes to be treated as "on-link" according to IETF RFC 4861 Section 5.1.
Configuration Parameters	

Included Containers		
Container Name	Multiplicity	Scope / Dependency
TcpIpNdpPrefixListEntry	1..*	Single entry in the prefix list.

10.2.34 TcpIpNdpPrefixListEntry

SWS Item	[ECUC_TcpIp_00206]
Container Name	TcpIpNdpPrefixListEntry
Parent Container	TcpIpNdpPrefixList
Description	Single entry in the prefix list.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00208]		
Parameter Name	TcpIpNdpPrefixListEntryPrefixAddress		
Parent Container	TcpIpNdpPrefixListEntry		
Description	The prefix of an IP address. This prefix can be used for on-link determination.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00207]	
Parameter Name	TcpIpNdpPrefixListEntryPrefixLength	
Parent Container	TcpIpNdpPrefixListEntry	
Description	The number of leading bits in the Prefix that are valid.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 128	
Default value	–	
Post-Build Variant Value	false	

▽

△

Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

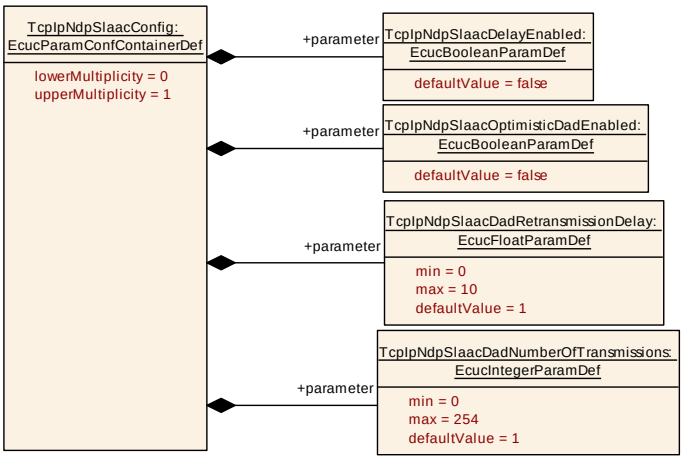

Figure 10.23: TcpIpNdpSlaacConfig

10.2.35 TcpIpNdpSlaacConfig

SWS Item	[ECUC_TcpIp_00122]
Container Name	TcpIpNdpSlaacConfig
Parent Container	TcpIpNdpConfig
Description	Specifies the configuration parameters for StateLess Address AutoConfiguration.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00128]	
Parameter Name	TcpIpNdpSlaacDadNumberOfTransmissions	
Parent Container	TcpIpNdpSlaacConfig	
Description	Number of Neighbor Solicitations that have to be unanswered in order to set an autoconfigurated address to PREFERRED (usable) state. [RFC4861 5.1. Node Configuration Variables] Default: DupAddrDetectTransmits = 1 Setting this value to 0 turns off DAD.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 254	
Default value	1	
Post-Build Variant Value	false	

▽

△

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00127]**		
Parameter Name	TcpIpNdpSlaacDadRetransmissionDelay		
Parent Container	TcpIpNdpSlaacConfig		
Description	Sets the maximum value for the address configuration delay (s). According to [RFC4861 5.4.2. Sending Neighbor Solicitation Messages] this value should be the same as MAX_RTR_SOLICITATION_DELAY. Default: MAX_RTR_SOLICITATION_DELAY = 1 second		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 10]		
Default value	1		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00125]**		
Parameter Name	TcpIpNdpSlaacDelayEnabled		
Parent Container	TcpIpNdpSlaacConfig		
Description	If enabled transmission of the first DAD Neighbor Solicitation will be delayed by a random value from [0...MAX_DAD_DELAY]. "This serves to alleviate congestion when many nodes start up on the link at the same time, such as after a power failure, and may help to avoid race conditions when more than one node is trying to solicit for the same address at the same time." "The delay will avoid similar congestion when multiple nodes are going to configure addresses by receiving the same single multicast router advertisement." [RFC4861 5.4.2. Sending Neighbor Solicitation Messages] Default: True		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00126]**
Parameter Name	TcpIpNdpSlaacOptimisticDadEnabled
Parent Container	TcpIpNdpSlaacConfig
Description	Enable Optimistic Duplicate Address Detection (DAD) according to RFC4429.

▽

△

Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

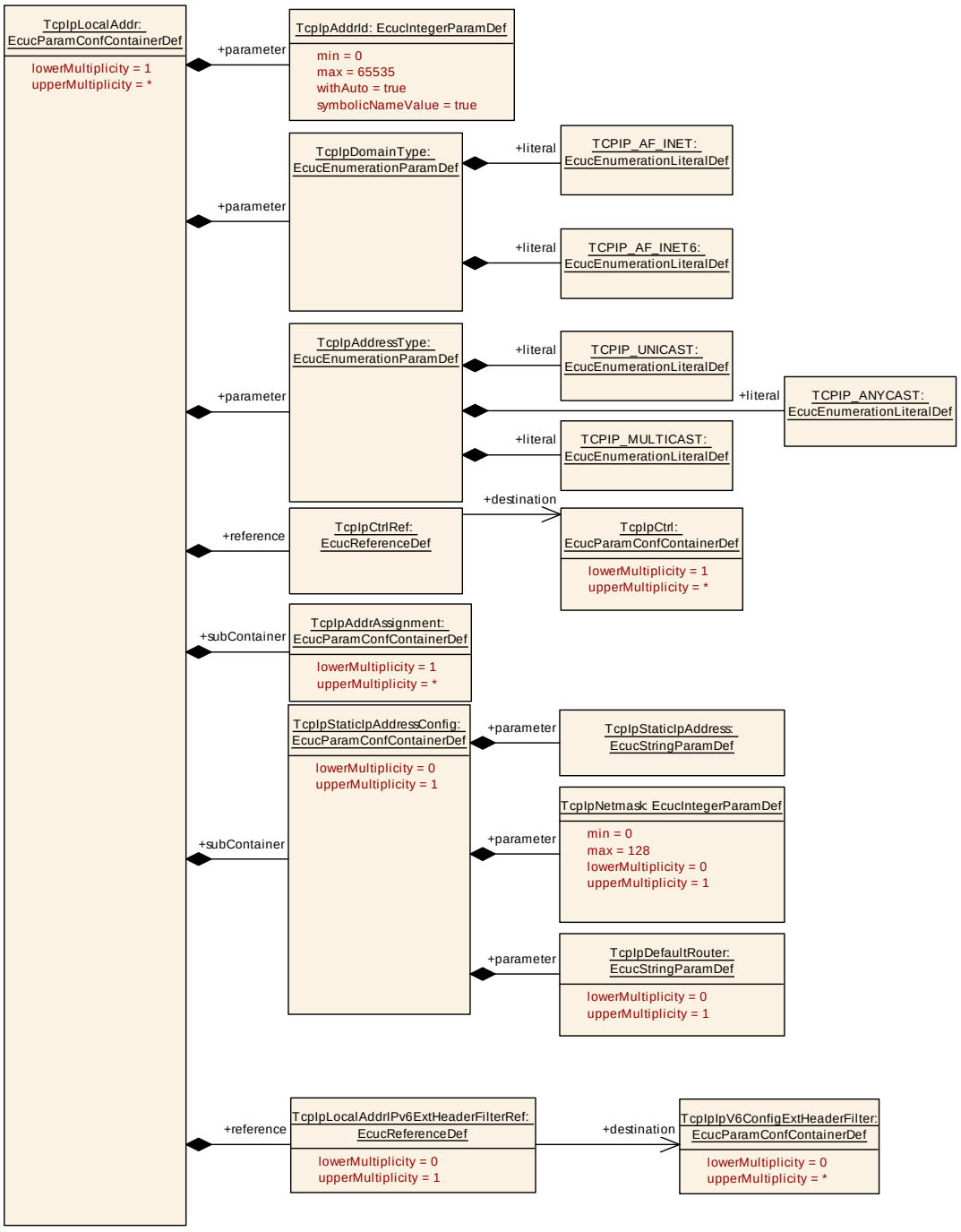

Figure 10.24: TcpIpLocalAddr

10.2.36 TcpIpLocalAddr

SWS Item	**[ECUC_TcpIp_00020]**
Container Name	TcpIpLocalAddr
Parent Container	TcpIpConfig
Description	Specifies the local IP (Internet Protocol) addresses used for IP communication.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00031]**		
Parameter Name	TcpIpAddressType		
Parent Container	TcpIpLocalAddr		
Description	Address type.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TCPIP_ANYCAST	Anycast address	
	TCPIP_MULTICAST	Multicast address.	
	TCPIP_UNICAST	Unicast address	
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00029]**		
Parameter Name	TcpIpAddrId		
Parent Container	TcpIpLocalAddr		
Description	IP address table identifier assigned by TCP/IP stack.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU withAuto = true		

SWS Item	**[ECUC_TcpIp_00030]**		
Parameter Name	TcpIpDomainType		
Parent Container	TcpIpLocalAddr		
Description	Address family.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TCPIP_AF_INET	IPv4 address	
	TCPIP_AF_INET6	IPv6 address	
Post-Build Variant Value	true		

△

Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00032]**		
Parameter Name	TcpIpCtrlRef		
Parent Container	TcpIpLocalAddr		
Description	Reference to a TcpIpCtrl specifying the EthIf Controller where the IP address shall be assigned.		
Multiplicity	1		
Type	Reference to TcpIpCtrl		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00200]**		
Parameter Name	TcpIpLocalAddrIPv6ExtHeaderFilterRef		
Parent Container	TcpIpLocalAddr		
Description	Reference to a set of IPv6 Extension Headers which are allowed for this local IPv6 address. Note: this parameter is only relevant if the related TcpIpDomainType is TCPIP_AF_INET6.		
Multiplicity	0..1		
Type	Reference to TcpIpIpV6ConfigExtHeaderFilter		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	dependency: only relevant if TcpIpDomainType = TCPIP_AF_INET6		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpAddrAssignment	1..*	This container is a subcontainer of TcpIpLocalAddr and specifies the assignment policy for the IP address.
TcpIpStaticIpAddressConfig	0..1	This container is a subcontainer of TcpIpLocalAddr and specifies a static IP address including directly related parameters.

10.2.37 TcpIpAddrAssignment

SWS Item	[ECUC_TcpIp_00033]		
Container Name	TcpIpAddrAssignment		
Parent Container	TcpIpLocalAddr		
Description	This container is a subcontainer of TcpIpLocalAddr and specifies the assignment policy for the IP address.		
Configuration Parameters			

SWS Item	[ECUC_TcpIp_00186]		
Parameter Name	TcpIpAssignmentLifetime		
Parent Container	TcpIpAddrAssignment		
Description	Defines the lifetime of a dynamically fetched IP address. If TcpIpAssignmentMethod = TCPIP_STATIC then TcpIpAssignmentLifetime shall be omitted.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	TCPIP_FORGET	After a dynamic IP address has been assigned just use it for this link-up time.	
	TCPIP_STORE	After a dynamic IP address has been assigned store the address persistently.	
Default value	TCPIP_FORGET		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00035]		
Parameter Name	TcpIpAssignmentMethod		
Parent Container	TcpIpAddrAssignment		
Description	Method of address assignment		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TCPIP_DHCP	Dynamic Assigned IP Address using DHCP	
	TCPIP_IPV6_ROUTER	Dynamic Configured IPv6 Address by Router Advertisement	
	TCPIP_LINKLOCAL	Linklocal IPv4/IPv6 Address Assignment	
	TCPIP_LINKLOCAL_DOIP	Linklocal IPv4/IPv6 Address Assignment using DoIP Parameters	
	TCPIP_STATIC	Static Assigned IP Address	
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00037]		
Parameter Name	TcpIpAssignmentPriority		
Parent Container	TcpIpAddrAssignment		
Description	Priority of assignment (1 is highest). If a new address from an assignment method with a higher priority is available, it overwrites the IP address previously assigned by an assignment method with a lower priority.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 3		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00036]		
Parameter Name	TcpIpAssignmentTrigger		
Parent Container	TcpIpAddrAssignment		
Description	Trigger of address assignment.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TCPIP_AUTOMATIC	Assignment shall be initiated automatically by TCP/IP stack.	
	TCPIP_MANUAL	Assignment shall be initiated manually via TcpIp_RequestIpAddrAssignment().	
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

No Included Containers

10.2.38 TcpIpStaticIpAddressConfig

SWS Item	[ECUC_TcpIp_00034]
Container Name	TcpIpStaticIpAddressConfig
Parent Container	TcpIpLocalAddr
Description	This container is a subcontainer of TcpIpLocalAddr and specifies a static IP address including directly related parameters.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00040]		
Parameter Name	TcpIpDefaultRouter		
Parent Container	TcpIpStaticIpAddressConfig		
Description	IP address of default router (gateway)		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00039]		
Parameter Name	TcpIpNetmask		
Parent Container	TcpIpStaticIpAddressConfig		
Description	Network mask of IPv4 address or address prefix of IPv6 address in CIDR Notation, i.e. decimal value between 0 and 32 (IPv4) or 0 and 128 (IPv6) that describes the number of significant bits defining the network number or prefix of an IP address.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 128		
Default value	–		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00038]
Parameter Name	TcpIpStaticIpAddress
Parent Container	TcpIpStaticIpAddressConfig
Description	Static IP Address. To specify any IP address for a certain EthIfCtrl, "ANY" has to be set as wildcard. See TcpIp_Bind() for more details.
Multiplicity	1
Type	EcucStringParamDef
Default value	–
Regular Expression	–

▽

△

Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

No Included Containers

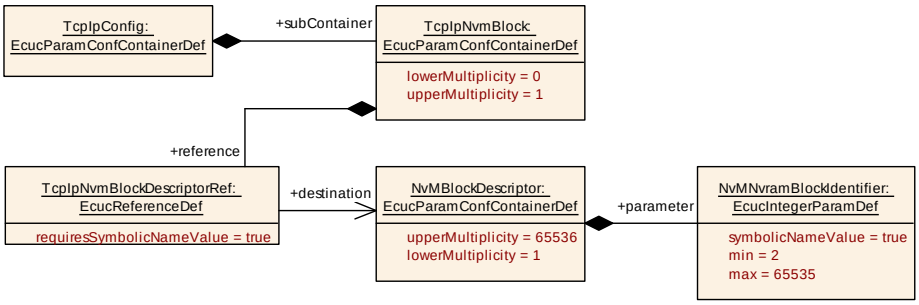

Figure 10.25: TcpIpNvm

10.2.39 TcpIpNvmBlock

SWS Item	**[ECUC_TcpIp_00184]**		
Container Name	TcpIpNvmBlock		
Parent Container	TcpIpConfig		
Description	Configuration of optional usage of Nvm in case the TcpIp module requires non volatile memory in the Ecu to store information (e.g. IP Address received via DHCP and shall be stored).		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_TcpIp_00185]**		
Parameter Name	TcpIpNvmBlockDescriptorRef		
Parent Container	TcpIpNvmBlock		
Description	Reference to the Nvm block description in the Nvm module configuration.		
Multiplicity	1		
Type	Symbolic name reference to NvMBlockDescriptor		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

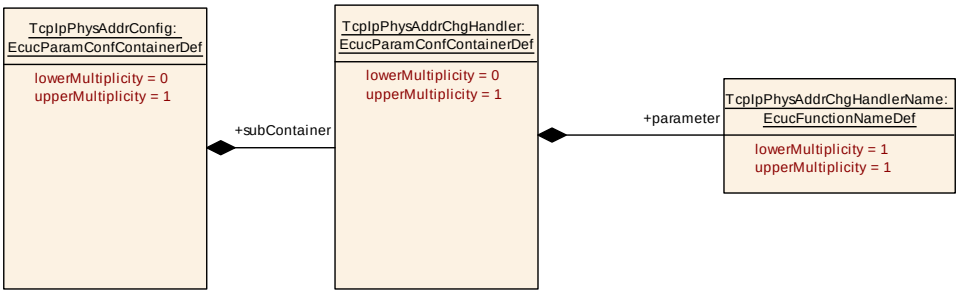

Figure 10.26: TcpIpPhysAddrConfig

10.2.40 TcpIpPhysAddrConfig

SWS Item	[ECUC_TcpIp_00083]
Container Name	TcpIpPhysAddrConfig
Parent Container	TcpIpConfig
Description	Specifies the physical address configuration.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpPhysAddrChgHandler	0..1	This container is a subcontainer of TcpIpPhysAddrConfig and specifies the configuration parameters for physical address change handler.

10.2.41 TcpIpPhysAddrChgHandler

SWS Item	[ECUC_TcpIp_00084]
Container Name	TcpIpPhysAddrChgHandler
Parent Container	TcpIpPhysAddrConfig
Description	This container is a subcontainer of TcpIpPhysAddrConfig and specifies the configuration parameters for physical address change handler.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00086]
Parameter Name	TcpIpPhysAddrChgHandlerName
Parent Container	TcpIpPhysAddrChgHandler
Description	This parameter defines the name of the physical address change function <Up>_PhysAddrTableChg.
Multiplicity	1
Type	EcucFunctionNameDef
Default value	–
Regular Expression	–
Post-Build Variant Value	false

▽

△

Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

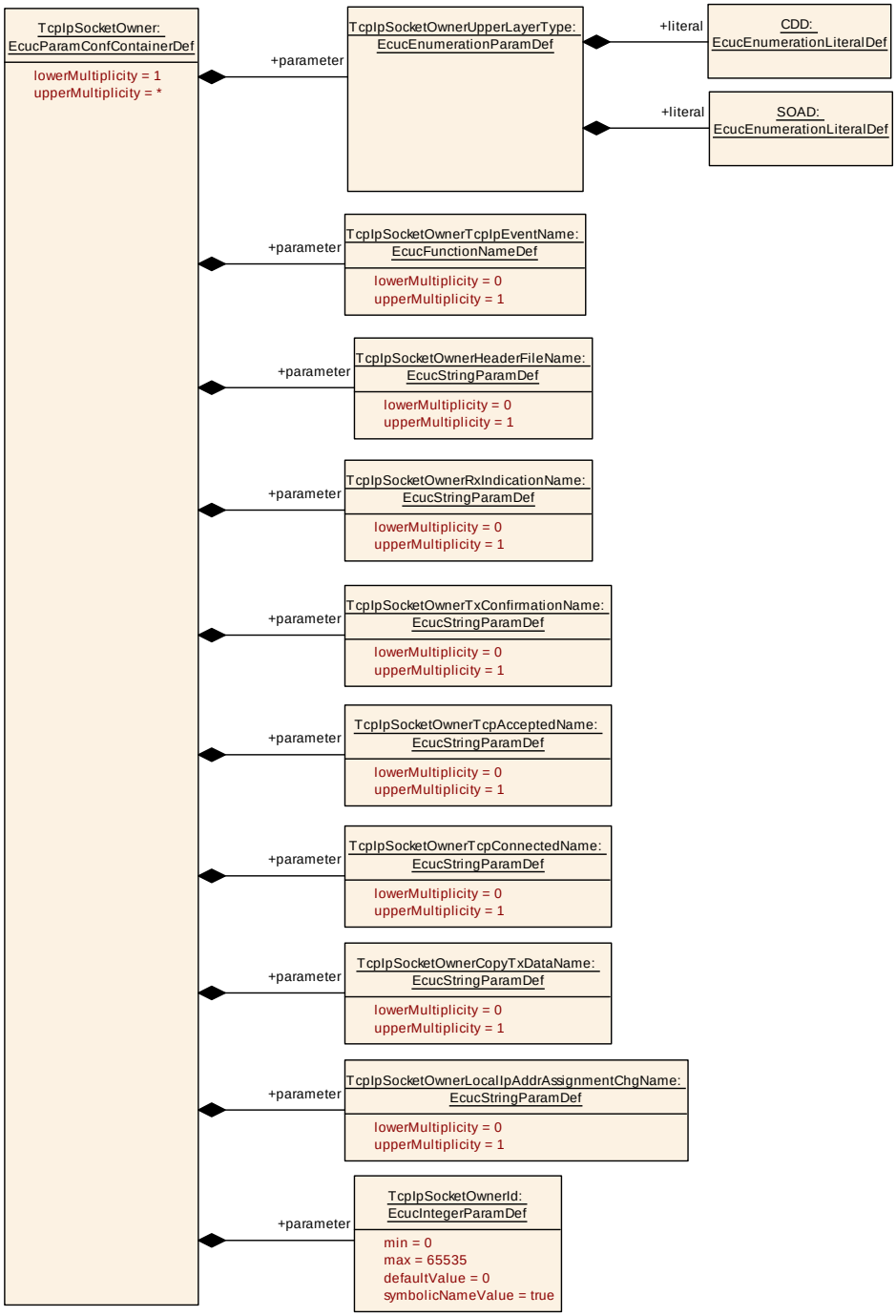

Figure 10.27: TcpIpSocketOwnerConfig

10.2.42 TcpIpSocketOwnerConfig

SWS Item	[ECUC_TcpIp_00172]
Container Name	TcpIpSocketOwnerConfig
Parent Container	TcpIpConfig
Description	Specifies the upper layer modules of TcpIp using the socket API.
Configuration Parameters	

Included Containers		
Container Name	Multiplicity	Scope / Dependency
TcpIpSocketOwner	1..*	This container is a subcontainer of TcpIpSocketOwnerConfig and specifies an upper layer of TcpIp that uses the socket API.

10.2.43 TcpIpSocketOwner

SWS Item	[ECUC_TcpIp_00173]
Container Name	TcpIpSocketOwner
Parent Container	TcpIpSocketOwnerConfig
Description	This container is a subcontainer of TcpIpSocketOwnerConfig and specifies an upper layer of TcpIp that uses the socket API.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00180]		
Parameter Name	TcpIpSocketOwnerCopyTxDataName		
Parent Container	TcpIpSocketOwner		
Description	This parameter defines the name of the <Up_CopyTxData> function of the TcpIpSocket Owner module. The function name shall only be configurable if TcpIpSocketOwner UpperLayerType is set to CDD.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local dependency: TcpIpSocketOwnerUpperLayerType		

SWS Item	[ECUC_TcpIp_00175]
Parameter Name	TcpIpSocketOwnerHeaderFileName
Parent Container	TcpIpSocketOwner

▽

△

Description	This parameter specifies the name of the header file containing the definition of the TcpIpSocketOwner module functions. The header file name shall only be configurable if TcpIpSocketOwnerUpperLayerType is set to CDD.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00316]**		
Parameter Name	TcpIpSocketOwnerId		
Parent Container	TcpIpSocketOwner		
Description	This value specifies the ID of the socket user.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	0		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00181]**		
Parameter Name	TcpIpSocketOwnerLocalIpAddrAssignmentChgName		
Parent Container	TcpIpSocketOwner		
Description	This parameter defines the name of the <Up_LocalIpAddrAssignmentChg> function of the TcpIpSocketOwner module. The function name shall only be configurable if TcpIpSocketOwnerUpperLayerType is set to CDD.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	

▽

△

Scope / Dependency	scope: local dependency: TcpIpSocketOwnerUpperLayerType

SWS Item	**[ECUC_TcpIp_00176]**		
Parameter Name	TcpIpSocketOwnerRxIndicationName		
Parent Container	TcpIpSocketOwner		
Description	This parameter defines the name of the <Up_RxIndication> function of the TcpIpSocket Owner module. The function name shall only be configurable if TcpIpSocketOwner UpperLayerType is set to CDD.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local dependency: TcpIpSocketOwnerUpperLayerType		

SWS Item	**[ECUC_TcpIp_00178]**		
Parameter Name	TcpIpSocketOwnerTcpAcceptedName		
Parent Container	TcpIpSocketOwner		
Description	This parameter defines the name of the <Up_TcpAccepted> function of the TcpIp SocketOwner module. The function name shall only be configurable if TcpIpSocket OwnerUpperLayerType is set to CDD.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local dependency: TcpIpSocketOwnerUpperLayerType		

SWS Item	**[ECUC_TcpIp_00179]**
Parameter Name	TcpIpSocketOwnerTcpConnectedName
Parent Container	TcpIpSocketOwner
Description	This parameter defines the name of the <Up_TcpConnected> function of the TcpIp SocketOwner module. The function name shall only be configurable if TcpIpSocket OwnerUpperLayerType is set to CDD.
Multiplicity	0..1
Type	EcucStringParamDef
Default value	–

▽

△

Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local dependency: TcpIpSocketOwnerUpperLayerType		

SWS Item	**[ECUC_TcpIp_00197]**		
Parameter Name	TcpIpSocketOwnerTcpIpEventName		
Parent Container	TcpIpSocketOwner		
Description	This parameter defines the name of the <Up_TcpIpEvent> function of the TcpIpSocket Owner module. The function name shall only be configurable if TcpIpSocketOwner UpperLayerType is set to CDD.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local dependency: TcpIpSocketOwnerUpperLayerType		

SWS Item	**[ECUC_TcpIp_00177]**		
Parameter Name	TcpIpSocketOwnerTxConfirmationName		
Parent Container	TcpIpSocketOwner		
Description	This parameter defines the name of the <Up_TxConfirmation> function of the TcpIp SocketOwner module. The function name shall only be configurable if TcpIpSocket OwnerUpperLayerType is set to CDD.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local dependency: TcpIpSocketOwnerUpperLayerType		

SWS Item	**[ECUC_TcpIp_00174]**
Parameter Name	TcpIpSocketOwnerUpperLayerType
Parent Container	TcpIpSocketOwner

▽

△

Description	This parameter specifies the type of the upper layer module.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CDD	Complex Driver	
	SOAD	Socket Adaptor	
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

No Included Containers

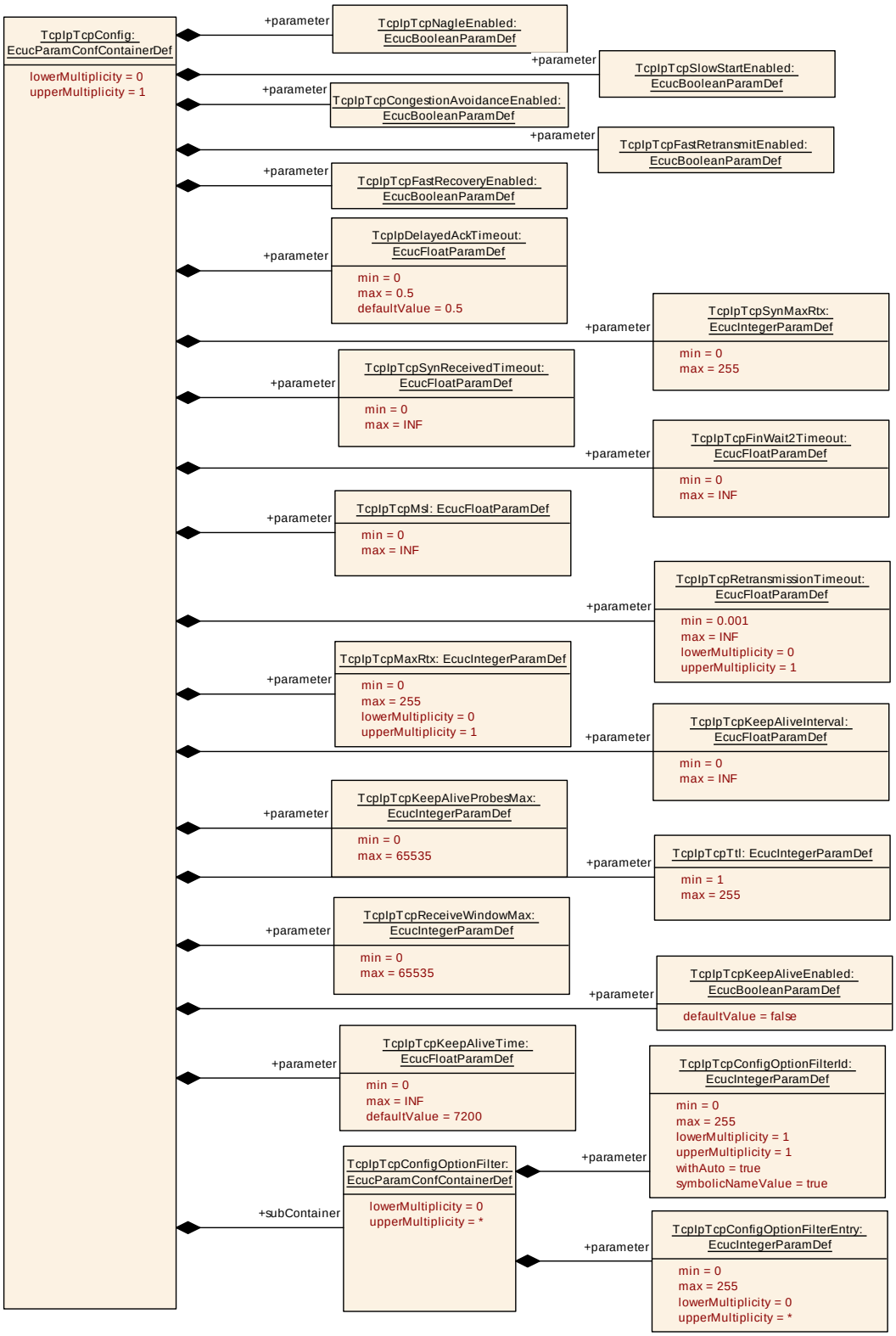

Figure 10.28: TcpIpTcpConfig

10.2.44 TcpIpTcpConfig

SWS Item	**[ECUC_TcpIp_00025]**
Container Name	TcpIpTcpConfig
Parent Container	TcpIpConfig
Description	Specifies the configuration parameters of the TCP (Transmission Control Protocol) sub-module.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00318]**		
Parameter Name	TcpIpDelayedAckTimeout		
Parent Container	TcpIpTcpConfig		
Description	The maximal time an acknowledgment is delayed for transmission in seconds.For further details, see also IETF RfC 1122 section 4.2.3.2.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	]0 .. 0.5]		
Default value	0.5		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00061]**		
Parameter Name	TcpIpTcpCongestionAvoidanceEnabled		
Parent Container	TcpIpTcpConfig		
Description	Enables (TRUE) or disables (FALSE) support of TCP congestion avoidance algorithm according to IETF RFC 5681.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00063]**		
Parameter Name	TcpIpTcpFastRecoveryEnabled		
Parent Container	TcpIpTcpConfig		
Description	Enables (TRUE) or disables (FALSE) support of TCP Fast Recovery according to IETF RFC 5681.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants

△

	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00062]**		
Parameter Name	TcpIpTcpFastRetransmitEnabled		
Parent Container	TcpIpTcpConfig		
Description	Enables (TRUE) or disables (FALSE) support of TCP Fast Retransmission according to IETF RFC 5681.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00066]**		
Parameter Name	TcpIpTcpFinWait2Timeout		
Parent Container	TcpIpTcpConfig		
Description	Timeout in [s] to receive a FIN from the remote node (after this node has initiated connection termination), i.e. maximum time waiting in FINWAIT-2 for a connection termination request from the remote TCP.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. INF]		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00082]**		
Parameter Name	TcpIpTcpKeepAliveEnabled		
Parent Container	TcpIpTcpConfig		
Description	Enables (TRUE) or disables (FALSE) TCP Keep Alive Probes according to IETF RFC 1122 chapter 4.2.3.6		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00070]		
Parameter Name	TcpIpTcpKeepAliveInterval		
Parent Container	TcpIpTcpConfig		
Description	Specifies the interval in [s] between subsequent keepalive probes.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. INF]		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: TcpIpTcpKeepAliveEnabled		

SWS Item	[ECUC_TcpIp_00071]		
Parameter Name	TcpIpTcpKeepAliveProbesMax		
Parent Container	TcpIpTcpConfig		
Description	Maximum number of times that a TCP Keep Alive is retransmitted before the connection is closed.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: TcpIpTcpKeepAliveEnabled		

SWS Item	[ECUC_TcpIp_00087]		
Parameter Name	TcpIpTcpKeepAliveTime		
Parent Container	TcpIpTcpConfig		
Description	Specifies the time in [s] between the last data packet sent (simple ACKs are not considered data) and the first keepalive probe. Note: Setting this configuration parameter to a value smaller or equal to the value of TcpIpMainFunctionPeriod results in the transmission of keep alive probes within every MainFunction cycle.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. INF]		
Default value	7200		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD

△

Scope / Dependency	scope: local dependency: TcpIpTcpKeepAliveEnabled

SWS Item	**[ECUC_TcpIp_00069]**		
Parameter Name	TcpIpTcpMaxRtx		
Parent Container	TcpIpTcpConfig		
Description	Maximum number of times that a TCP segment is retransmitted before the TCP connection is closed. This parameter is only valid if TcpIpTcpRetransmissionTimeout is configured. Note: This parameter also applies for FIN retransmissions.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	dependency: TcpIpTcpRetransmissionTimeout		

SWS Item	**[ECUC_TcpIp_00067]**		
Parameter Name	TcpIpTcpMsl		
Parent Container	TcpIpTcpConfig		
Description	Maximum segment lifetime in [s]. (Note: TIME-WAIT = 2 x TcpIpTcpMsl - to ensure that the remote node received the acknowledgment to its connection termination request.)		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. INF]		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00059]**		
Parameter Name	TcpIpTcpNagleEnabled		
Parent Container	TcpIpTcpConfig		
Description	Enables (TRUE) or disables (FALSE) support of Nagle's algorithm according to IETF RFC 1122 (chapter 4.2.3.4 When to Send Data). If enabled the Nagle's algorithm is activated per default for all TCP sockets, but can be deactivated via TcpIp_Change Parameter() API.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

▽

△

Scope / Dependency	scope: local

SWS Item	**[ECUC_TcpIp_00073]**		
Parameter Name	TcpIpTcpReceiveWindowMax		
Parent Container	TcpIpTcpConfig		
Description	Default value of maximum receive window in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00068]**		
Parameter Name	TcpIpTcpRetransmissionTimeout		
Parent Container	TcpIpTcpConfig		
Description	Timeout in [s] before an unacknowledged TCP segment is sent again. If the timeout is disabled or set to INF, no TCP segments shall be retransmitted. Value can be overwritten by TcpIp_ChangeParameter() API for a particular connection.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[0.001 .. INF]		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00327]**		
Parameter Name	TcpIpTcpSackEnabled		
Parent Container	TcpIpTcpConfig		
Description	Defines if the SACK (selective acknowledgement) mechanism shall be supported according to IETF RFC 2018. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00060]		
Parameter Name	TcpIpTcpSlowStartEnabled		
Parent Container	TcpIpTcpConfig		
Description	Enables (TRUE) or disables (FALSE) support of TCP slow start algorithm according to IETF RFC 5681.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00064]		
Parameter Name	TcpIpTcpSynMaxRtx		
Parent Container	TcpIpTcpConfig		
Description	Maximum number of times that a TCP SYN is retransmitted. Note: SYN will be retried after TcpIpTcpRetransmissionTimeout. The connection will be dropped if no matching connection request has been received after the last TCP SYN has been sent and TcpIp TcpRetransmissionTimeout has been expired.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00065]		
Parameter Name	TcpIpTcpSynReceivedTimeout		
Parent Container	TcpIpTcpConfig		
Description	Timeout in [s] to complete a remotely initiated TCP connection establishment, i.e. maximum time waiting in SYN-RECEIVED for a confirming connection request acknowledgment after having both received and sent a connection request.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. INF]		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00072]		
Parameter Name	TcpIpTcpTtl		
Parent Container	TcpIpTcpConfig		
Description	Default Time-to-live value of outgoing TCP packets.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00329]		
Parameter Name	TcpIpTcpWindowScale		
Parent Container	TcpIpTcpConfig		
Description	Defines the TCP window scale. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 14		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00328]		
Parameter Name	TcpIpTcpWindowScaleOptionEnabled		
Parent Container	TcpIpTcpConfig		
Description	Defines if the TCP window scale option and mechanism shall be supported according to IETF RFC 7323, chapter 2. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpTcpConfigOptionFilter	0..*	This container describes the white list for the filtering of TCP options, i.e. segments containing TCP options not listed here shall be silently dropped.

10.2.45 TcpIpTcpConfigOptionFilter

SWS Item	[ECUC_TcpIp_00202]		
Container Name	TcpIpTcpConfigOptionFilter		
Parent Container	TcpIpTcpConfig		
Description	This container describes the white list for the filtering of TCP options, i.e. segments containing TCP options not listed here shall be silently dropped.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Configuration Parameters			

SWS Item	[ECUC_TcpIp_00204]		
Parameter Name	TcpIpTcpConfigOptionFilterEntry		
Parent Container	TcpIpTcpConfigOptionFilter		
Description	TCP option kind allowed by this filter.		
Multiplicity	0..*		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00203]		
Parameter Name	TcpIpTcpConfigOptionFilterId		
Parent Container	TcpIpTcpConfigOptionFilter		
Description	Identification of the TCP option filter.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local withAuto = true		

No Included Containers

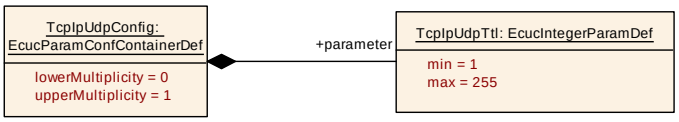

Figure 10.29: TcpIpUdpConfig

10.2.46 TcpIpUdpConfig

SWS Item	**[ECUC_TcpIp_00026]**
Container Name	TcpIpUdpConfig
Parent Container	TcpIpConfig
Description	Specifies the configuration parameters of the UDP (User Datagram Protocol) sub-module
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00075]**		
Parameter Name	TcpIpUdpTtl		
Parent Container	TcpIpUdpConfig		
Description	Default Time-to-live value of outgoing UDP packets.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

No Included Containers

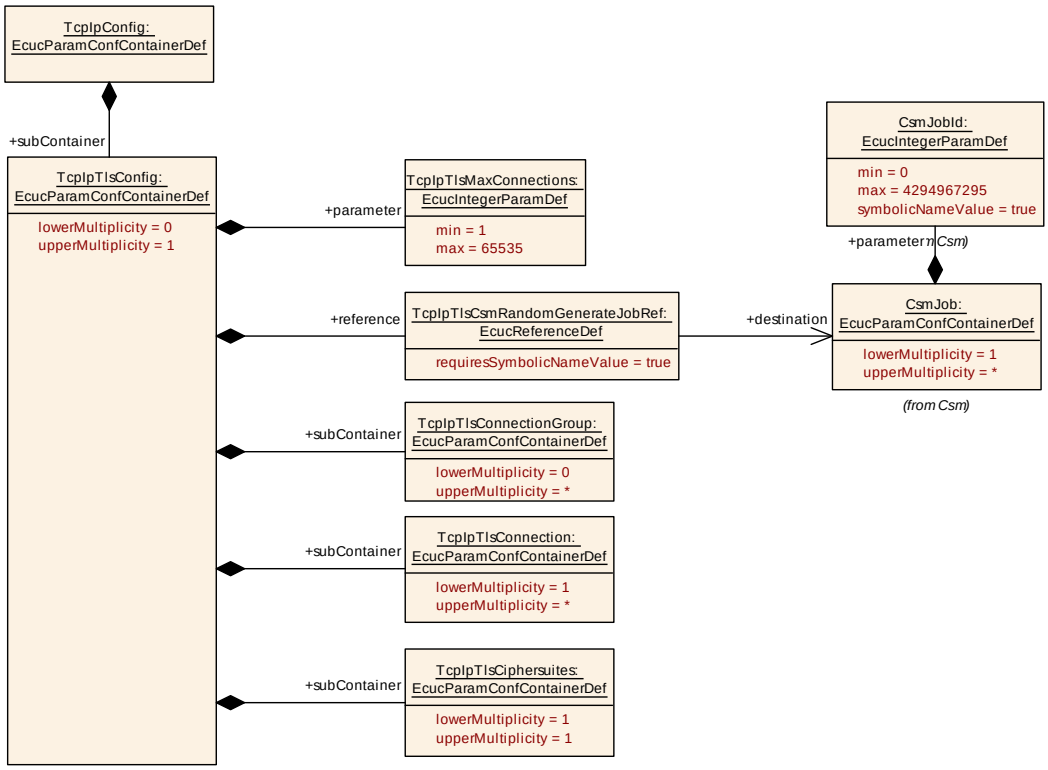

Figure 10.30: TcpIpTlsConfig

10.2.47 TcpIpTlsConfig

SWS Item	**[ECUC_TcpIp_00219]**
Container Name	TcpIpTlsConfig
Parent Container	TcpIpConfig
Description	Specifies the configuration parameters of the TLS (Transport Layer Security) sub module.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00220]**		
Parameter Name	TcpIpTlsMaxConnections		
Parent Container	TcpIpTlsConfig		
Description	Defines the max. number of TLS connections that can be opened at the same time.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00221]		
Parameter Name	TcpIpTlsCsmRandomGenerateJobRef		
Parent Container	TcpIpTlsConfig		
Description	Reference to a CSM job to generate a random value.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	Multiplicity	Scope / Dependency
TcpIpTlsCiphersuites	1	This container provides the information about supported ciphersuites used by TLS.
TcpIpTlsConnection	1..*	This container defines the properties of a TLS connection
TcpIpTlsConnectionGroup	0..*	This optional container is used to collect all TlsConnections that belong to a TlsConnectionGroup. The intention of a TLS connection group is to share resources among TLS connections collected in a group, because only one connection of a group can be used at a time.

10.2.48 TcpIpTlsConnectionGroup

SWS Item	[ECUC_TcpIp_00224]
Container Name	TcpIpTlsConnectionGroup
Parent Container	TcpIpTlsConfig
Description	This optional container is used to collect all TlsConnections that belong to a Tls ConnectionGroup. The intention of a TLS connection group is to share resources among TLS connections collected in a group, because only one connection of a group can be used at a time.
Configuration Parameters	

No Included Containers

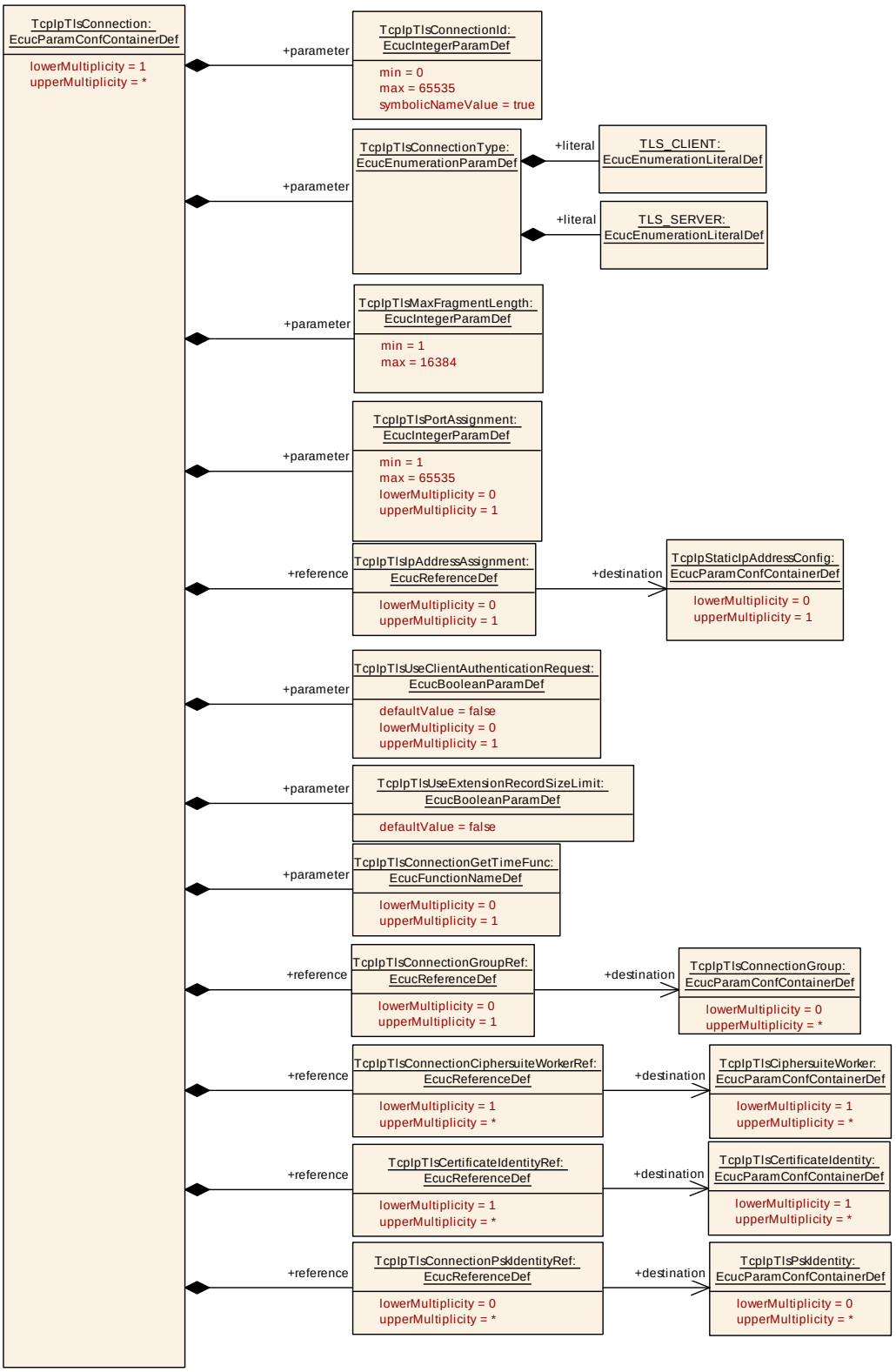

Figure 10.31: TcpIpTlsConnection

10.2.49 TcpIpTlsConnection

SWS Item	[ECUC_TcpIp_00223]
Container Name	TcpIpTlsConnection
Parent Container	TcpIpTlsConfig
Description	This container defines the properties of a TLS connection
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00232]		
Parameter Name	TcpIpTlsConnectionGetTimeFunc		
Parent Container	TcpIpTlsConnection		
Description	Defines the function name for the Up_TlsGetCurrentTimeStamp() callback.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: This definition is needed if a connection specific time shall be provided with the client hello message. If not present, the time will be set to 0.		

SWS Item	[ECUC_TcpIp_00225]		
Parameter Name	TcpIpTlsConnectionId		
Parent Container	TcpIpTlsConnection		
Description	Identifier of the connection. The set of configured identifiers shall be consecutive and gapless.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00226]
Parameter Name	TcpIpTlsConnectionType
Parent Container	TcpIpTlsConnection
Description	Specifies if the TLS connection is a server or a client.

△

Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TLS_CLIENT	–	
	TLS_SERVER	–	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00227]**		
Parameter Name	TcpIpTlsMaxFragmentLength		
Parent Container	TcpIpTlsConnection		
Description	Specifies the max length in bytes of a TLS fragment that is sent as a block. If ISO 15118-2 shall be supported, the range is 512 .. 16384.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 16384		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00285]**		
Parameter Name	TcpIpTlsPortAssignment		
Parent Container	TcpIpTlsConnection		
Description	Specifies the port address that is used for TLS communication.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00230]**
Parameter Name	TcpIpTlsUseClientAuthenticationRequest
Parent Container	TcpIpTlsConnection

▽

△

Description	Defines if client authentication shall be applied for this TLS connection.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	dependency: Informs the TLS_SERVER that a client authentication shall be requested. Can be omitted on TLS_CLIENT side.		

SWS Item	**[ECUC_TcpIp_00334]**		
Parameter Name	TcpIpTlsUseExtensionCertificateStatusRequest		
Parent Container	TcpIpTlsConnection		
Description	Defines if the optional extension status_request_v2 shall be supported. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00332]**		
Parameter Name	TcpIpTlsUseExtensionMaxFragmentLength		
Parent Container	TcpIpTlsConnection		
Description	Defines if the optional extension for max_fragment_length shall be supported. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: TcpIpTlsUseExtensionRecordSizeLimit is set to FALSE		

SWS Item	[ECUC_TcpIp_00231]		
Parameter Name	TcpIpTlsUseExtensionRecordSizeLimit		
Parent Container	TcpIpTlsConnection		
Description	Defines if the security extension for record_size_limit shall be supported as defined in IETF RFC 8449, chapter 4.1.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00333]		
Parameter Name	TcpIpTlsUseExtensionTrustedCAKeys		
Parent Container	TcpIpTlsConnection		
Description	Defines if the optional extension for trusted_ca_keys shall be supported **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00235]		
Parameter Name	TcpIpTlsCertificateIdentityRef		
Parent Container	TcpIpTlsConnection		
Description	References the container that contains the certificate and identity information.		
Multiplicity	1..*		
Type	Reference to TcpIpTlsCertificateIdentity		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: There shall be only one TlsCertificateIdentity reference if server name identification is not used.		

SWS Item	[ECUC_TcpIp_00234]		
Parameter Name	TcpIpTlsConnectionCiphersuiteWorkerRef		
Parent Container	TcpIpTlsConnection		
Description	References the container that contains the jobs and keys to process the application data.		
Multiplicity	1..*		
Type	Reference to TcpIpTlsCiphersuiteWorker		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00233]		
Parameter Name	TcpIpTlsConnectionGroupRef		
Parent Container	TcpIpTlsConnection		
Description	Assigns the TLS connection to a connection group.		
Multiplicity	0..1		
Type	Reference to TcpIpTlsConnectionGroup		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00236]		
Parameter Name	TcpIpTlsConnectionPskIdentityRef		
Parent Container	TcpIpTlsConnection		
Description	References the container that contains information about pre-shared keys.		
Multiplicity	0..*		
Type	Reference to TcpIpTlsPskIdentity		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	

△

Scope / Dependency	scope: local dependency: A reference to PskIdentity container is only useful if at least one CiphersuiteDefinition is referenced offering a PSK ciphersuite. Multiplicity might be reduced to 1 to provide a unique PSK identification depending on the TLS protocol version and/or ifit is used for the TLS server or client.

SWS Item	**[ECUC_TcpIp_00229]**		
Parameter Name	TcpIpTlsIpAddressAssignment		
Parent Container	TcpIpTlsConnection		
Description	Contains additional information about the endpoint IP address information. If this reference is present, the IP address of the connecting socket shall also be checked if a TLS connection shall be assigned automatically to a socket.		
Multiplicity	0..1		
Type	Reference to TcpIpStaticIpAddressConfig		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: If this item is not present but TcpIpTlsPortAssignment is defined, then IP address information is not relevant for the TLS connection assignment. If TcpIpTlsPort Assignment is not defined this item has no affect and shall not be defined, too.		

No Included Containers

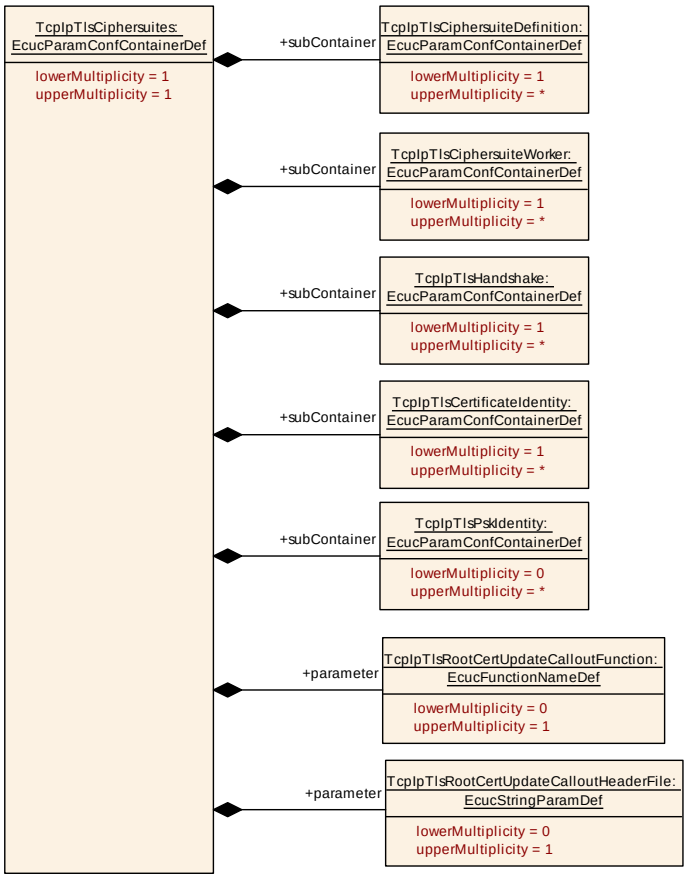

Figure 10.32: TcpIpTlsCiphersuites

10.2.50 TcpIpTlsCiphersuites

SWS Item	[ECUC_TcpIp_00222]
Container Name	TcpIpTlsCiphersuites
Parent Container	TcpIpTlsConfig
Description	This container provides the information about supported ciphersuites used by TLS.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00330]
Parameter Name	TcpIpTlsRootCertUpdateCalloutFunction
Parent Container	TcpIpTlsCiphersuites
Description	This optional parameter specifies the name of a callout function that is used when a new, valid, root certificate is received during a TLS handshake. Can be used to perform actions based on the received certificate. **Tags:** atp.Status=draft
Multiplicity	0..1
Type	EcucFunctionNameDef
Default value	–

△

Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00331]**		
Parameter Name	TcpIpTlsRootCertUpdateCalloutHeaderFile		
Parent Container	TcpIpTlsCiphersuites		
Description	This optional parameter specifies the name of the header file containing the definition for the function specified in TcpIpTlsRootCertUpdateCalloutFunction. **Tags:** atp.Status=draft		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpTlsCertificateIdentity	1..*	This container provides information about the certificates used for ciphersuites.
TcpIpTlsCiphersuiteDefinition	1..*	This container provides the static information of a ciphersuite used by TLS.
TcpIpTlsCiphersuiteWorker	1..*	This container provides the jobs and keys necessary for TLS data transmission and reception.
TcpIpTlsHandshake	1..*	This container provides information that is needed to process a handshake. It contains the appropriate references to jobs and keys of the CSM to perform the key exchange cryptographic for the ciphersuite and involved certificates.
TcpIpTlsPskIdentity	0..*	This container provides information about static definition of pre-shared keys. It is used during the handshake to negotiate pre-shared keys between a client and a server. Note: The callbacks for pre-shared keys are an alternative to the static definition. The callbacks allow to define the associated keys at runtime if pre-shared keys are used but no static definition is available. The container definition is used for static configuration.

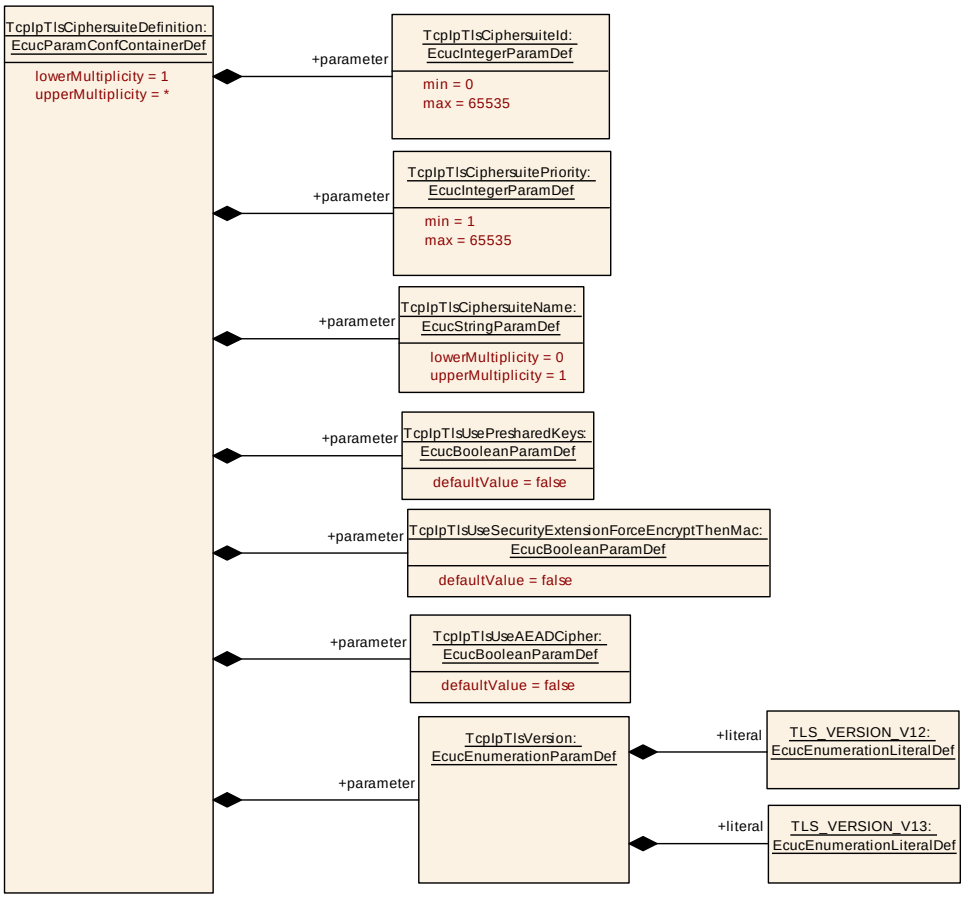

Figure 10.33: TcpipTlsCiphersuiteDefinition

10.2.51 TcpIpTlsCiphersuiteDefinition

SWS Item	[ECUC_TcpIp_00237]
Container Name	TcpIpTlsCiphersuiteDefinition
Parent Container	TcpIpTlsCiphersuites
Description	This container provides the static information of a ciphersuite used by TLS.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00242]	
Parameter Name	TcpIpTlsCiphersuiteId	
Parent Container	TcpIpTlsCiphersuiteDefinition	
Description	ID that represents the ciphersuite according to IETF, e.g. RFC4492, Sect. 6, RFC8446, Appendix B.4 or RFC5246, Appendix A.5.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 65535	
Default value	–	
Post-Build Variant Value	false	

▽

△

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00244]**		
Parameter Name	TcpIpTlsCiphersuiteName		
Parent Container	TcpIpTlsCiphersuiteDefinition		
Description	Provides a verbal name for the ciphersuite. The name should be the one defined in the respective RFC, e.g. TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 (TLS 1.2) or TLS_AES_128_GCM_SHA256 (TLS 1.3)		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00243]**		
Parameter Name	TcpIpTlsCiphersuitePriority		
Parent Container	TcpIpTlsCiphersuiteDefinition		
Description	Defines the priority of the cipher. The higher the number the lower the priority.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00247]**		
Parameter Name	TcpIpTlsUseAEADCipher		
Parent Container	TcpIpTlsCiphersuiteDefinition		
Description	Specifies if the ciphersuite supports AEAD for data en-/decryption.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00245]		
Parameter Name	TcpIpTlsUsePresharedKeys		
Parent Container	TcpIpTlsCiphersuiteDefinition		
Description	Defines if this ciphersuite uses pre-shared keys. If so, additional configuration or callbacks will be used for pre-shared key negotiation.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00246]		
Parameter Name	TcpIpTlsUseSecurityExtensionForceEncryptThenMac		
Parent Container	TcpIpTlsCiphersuiteDefinition		
Description	Defines if the security extension according to IETF RFC 7366 shall be supported. This is useful for ciphersuites using CBC mode.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00248]		
Parameter Name	TcpIpTlsVersion		
Parent Container	TcpIpTlsCiphersuiteDefinition		
Description	Declares the TLS version that this ciphersuite shall be used for.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TLS_VERSION_V12	–	
	TLS_VERSION_V13	–	
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

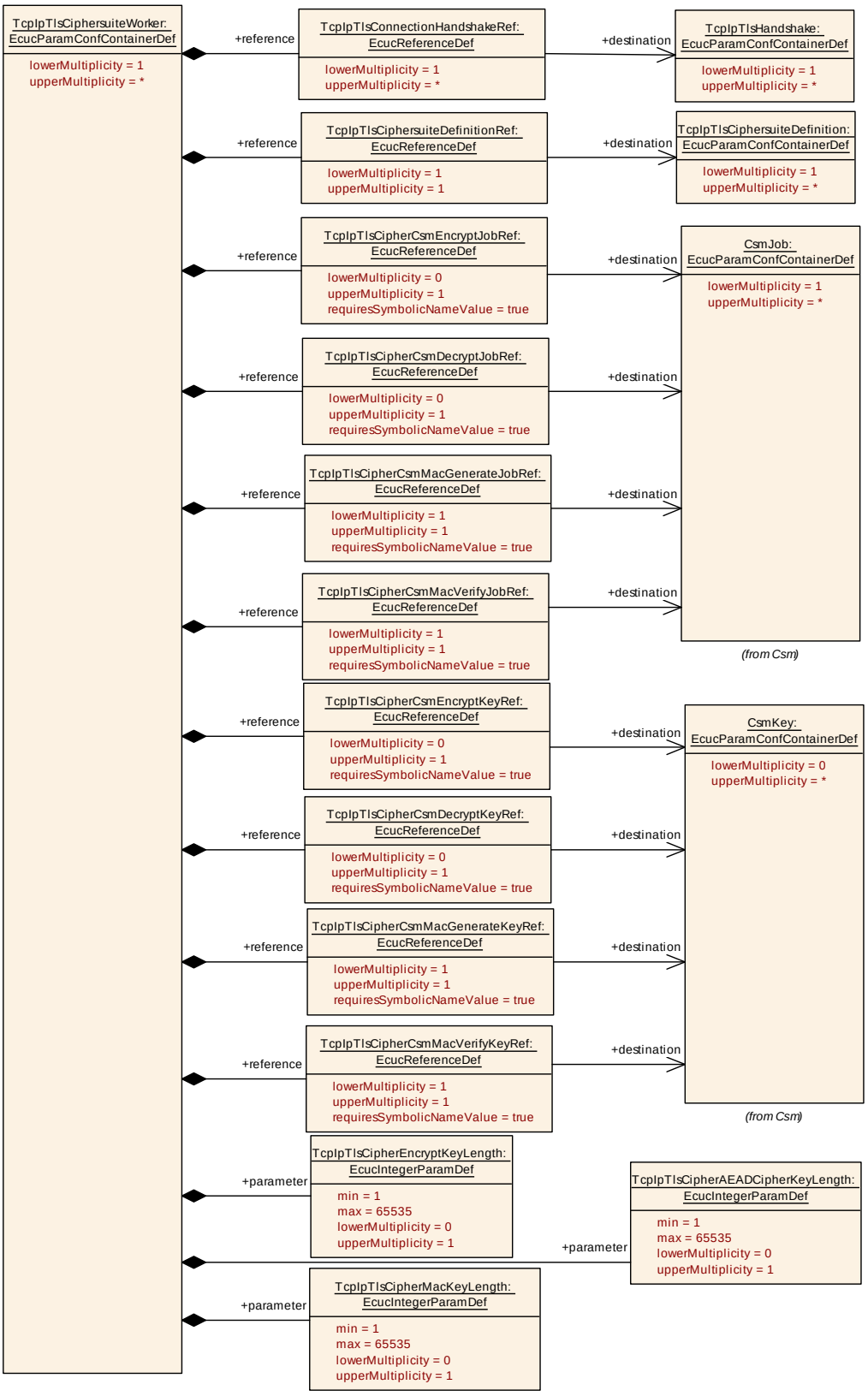

Figure 10.34: TcpIpTlsCiphersuiteWorker

10.2.52 TcpIpTlsCiphersuiteWorker

SWS Item	[ECUC_TcpIp_00238]
Container Name	TcpIpTlsCiphersuiteWorker
Parent Container	TcpIpTlsCiphersuites
Description	This container provides the jobs and keys necessary for TLS data transmission and reception.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00254]		
Parameter Name	TcpIpTlsCipherAEADCipherKeyLength		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Defines the key length for en- / decryption with authentication data (AEAD).		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: This value shall only be set if the cipher uses AEAD. If such a worker is selected, then Csm_AEADEncrypt() and Csm_AEADDecrypt() shall be used and AEAD shall be supported. Required to be set when TcpIpTlsCipherDefinition/TcpIpTls AEADCipher is set to TRUE.		

SWS Item	[ECUC_TcpIp_00253]		
Parameter Name	TcpIpTlsCipherEncryptKeyLength		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Defines the key length used for en- or decryption. The key length is valid for (symmetric) encryption and decryption.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00257]		
Parameter Name	TcpIpTlsCipherMacKeyLength		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Specifies the length of the MAC key		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00255]		
Parameter Name	TcpIpTlsCipherCsmDecryptJobRef		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Reference to a CSM job to perform the data decryption operation		
Multiplicity	0..1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00256]		
Parameter Name	TcpIpTlsCipherCsmDecryptKeyRef		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Reference to a CSM key associated to the CSM job that performs the data decryption operation		
Multiplicity	0..1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00251]		
Parameter Name	TcpIpTlsCipherCsmEncryptJobRef		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Reference to a CSM job to perform the data encryption operation		
Multiplicity	0..1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants

▽

△

	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00252]**		
Parameter Name	TcpIpTlsCipherCsmEncryptKeyRef		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Reference to a CSM key associated to the CSM job that performs the data encryption operation		
Multiplicity	0..1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00258]**		
Parameter Name	TcpIpTlsCipherCsmMacGenerateJobRef		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Reference to a CSM job to perform the MAC generate operation		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00259]**		
Parameter Name	TcpIpTlsCipherCsmMacGenerateKeyRef		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Reference to a CSM key associated to the CSM job that performs the MAC generate operation		
Multiplicity	1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00260]**
Parameter Name	TcpIpTlsCipherCsmMacVerifyJobRef
Parent Container	TcpIpTlsCiphersuiteWorker
Description	Reference to a CSM job to perform the MAC verify operation

▽

△

Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00261]**		
Parameter Name	TcpIpTlsCipherCsmMacVerifyKeyRef		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Reference to a CSM key associated to the CSM job that performs the MAC verify operation		
Multiplicity	1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00250]**		
Parameter Name	TcpIpTlsCiphersuiteDefinitionRef		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	Reference to a ciphersuite definition container		
Multiplicity	1		
Type	Reference to TcpIpTlsCiphersuiteDefinition		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00249]**		
Parameter Name	TcpIpTlsConnectionHandshakeRef		
Parent Container	TcpIpTlsCiphersuiteWorker		
Description	References the container that contains the jobs and keys for handshake operation. Referencing multiple handshake containers allow to share them between workers and to choose the next unused during the handshake.		
Multiplicity	1..*		
Type	Reference to TcpIpTlsHandshake		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants

▽

△

	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

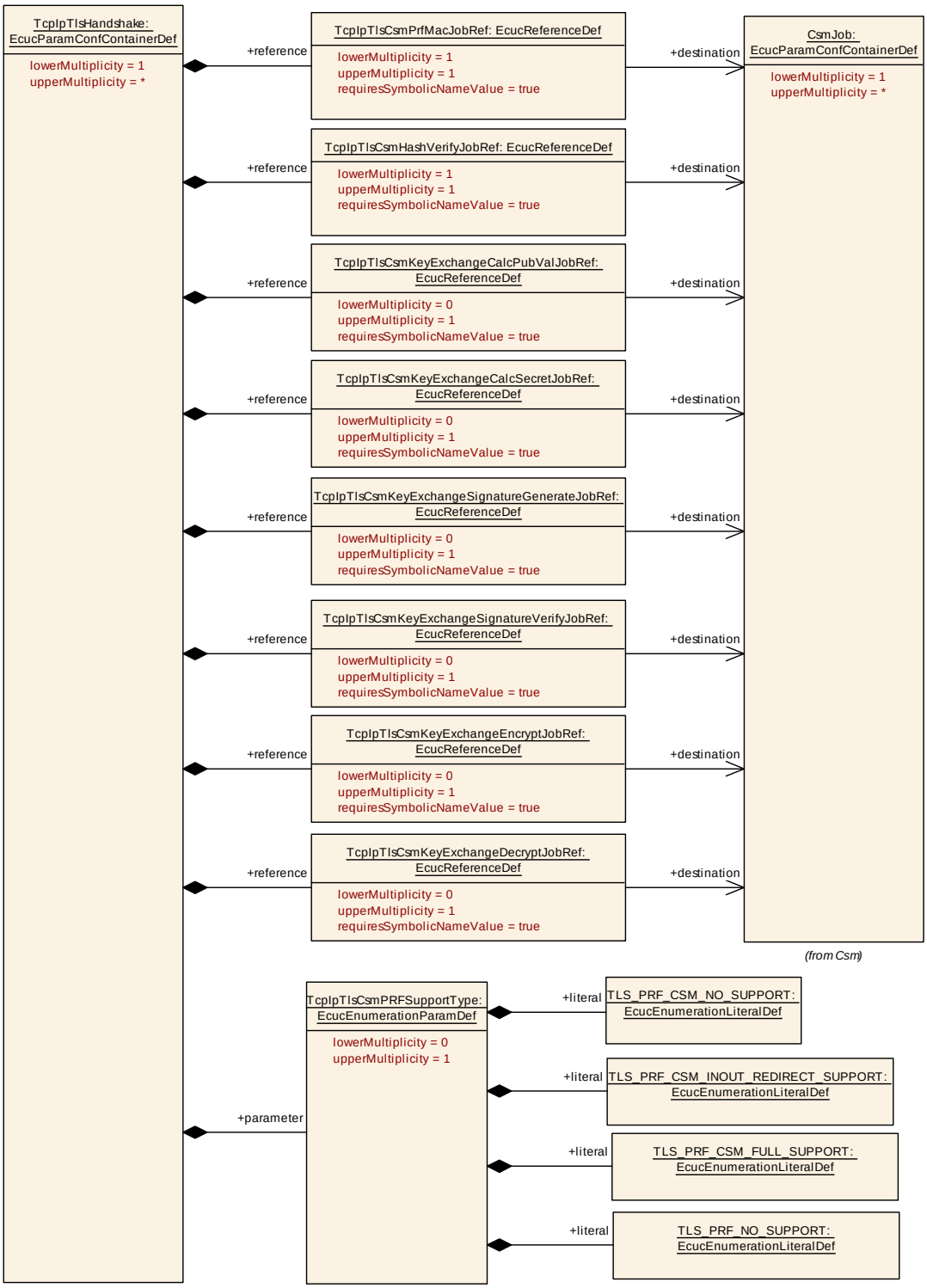

Figure 10.35: TcpIpTlsHandshake

10.2.53 TcpIpTlsHandshake

SWS Item	[ECUC_TcpIp_00239]
Container Name	TcpIpTlsHandshake
Parent Container	TcpIpTlsCiphersuites
Description	This container provides information that is needed to process a handshake. It contains the appropriate references to jobs and keys of the CSM to perform the key exchange cryptographic for the ciphersuite and involved certificates.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00264]		
Parameter Name	TcpIpTlsCsmPRFSupportType		
Parent Container	TcpIpTlsHandshake		
Description	Specifies how the CSM job supports the PRF operation.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	TLS_PRF_CSM_FULL_SUPPORT	–	
	TLS_PRF_CSM_INOUT_REDIRECT_SUPPORT	–	
	TLS_PRF_CSM_NO_SUPPORT	–	
	TLS_PRF_NO_SUPPORT	–	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00265]		
Parameter Name	TcpIpTlsCsmHashVerifyJobRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM job to perform the hash operation for the whole handshake.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00267]
Parameter Name	TcpIpTlsCsmKeyExchangeCalcPubValJobRef
Parent Container	TcpIpTlsHandshake
Description	Reference to a CSM job to perform the DH Key Exchange algorithm operation

▽

△

Multiplicity	0..1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00269]**		
Parameter Name	TcpIpTlsCsmKeyExchangeCalcSecretJobRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM job to perform the Key Exchange algorithm operation		
Multiplicity	0..1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: Only required if asynchronous job is used for key exchange calculation.		

SWS Item	**[ECUC_TcpIp_00276]**		
Parameter Name	TcpIpTlsCsmKeyExchangeDecryptJobRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM job to perform data decryption, e.g. with RSA key exchange operation.		
Multiplicity	0..1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00277]		
Parameter Name	TcpIpTlsCsmKeyExchangeDecryptKeyRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM key to perform data decryption, e.g. with RSA, used for exchange operation.		
Multiplicity	0..1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00274]		
Parameter Name	TcpIpTlsCsmKeyExchangeEncryptJobRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM job to perform data encryption, e.g. with RSA key exchange operation.		
Multiplicity	0..1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00275]		
Parameter Name	TcpIpTlsCsmKeyExchangeEncryptKeyRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM key to perform data encryption, e.g. with RSA, used for exchange operation.		
Multiplicity	0..1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants

▽

△

	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00268]**		
Parameter Name	TcpIpTlsCsmKeyExchangeKeyRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM key used for Diffie Hellman (DH) key exchange operation.		
Multiplicity	0..1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00270]**		
Parameter Name	TcpIpTlsCsmKeyExchangeSignatureGenerateJobRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM job to perform signature generation for DH operation		
Multiplicity	0..1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00271]**		
Parameter Name	TcpIpTlsCsmKeyExchangeSignatureGenerateKeyRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM key to perform signature generation for DH operation		
Multiplicity	0..1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	

▽

△

	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00272]**		
Parameter Name	TcpIpTlsCsmKeyExchangeSignatureVerifyJobRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM job to perform signature verification for DH operation		
Multiplicity	0..1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00273]**		
Parameter Name	TcpIpTlsCsmKeyExchangeSignatureVerifyKeyRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM key to perform signature verification for DH operation		
Multiplicity	0..1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00266]**
Parameter Name	TcpIpTlsCsmMasterSecretKeyRef
Parent Container	TcpIpTlsHandshake
Description	This is the reference to the master key that is calculated during the session.
Multiplicity	0..1
Type	Symbolic name reference to CsmKey
Post-Build Variant Multiplicity	false
Post-Build Variant Value	false

▽

△

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00262]**		
Parameter Name	TcpIpTlsCsmPrfMacJobRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM job to perform the PRF hash operation		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00263]**		
Parameter Name	TcpIpTlsCsmPrfMacKeyRef		
Parent Container	TcpIpTlsHandshake		
Description	Reference to a CSM key associated to the CSM job that performs the PRF hash operation		
Multiplicity	1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

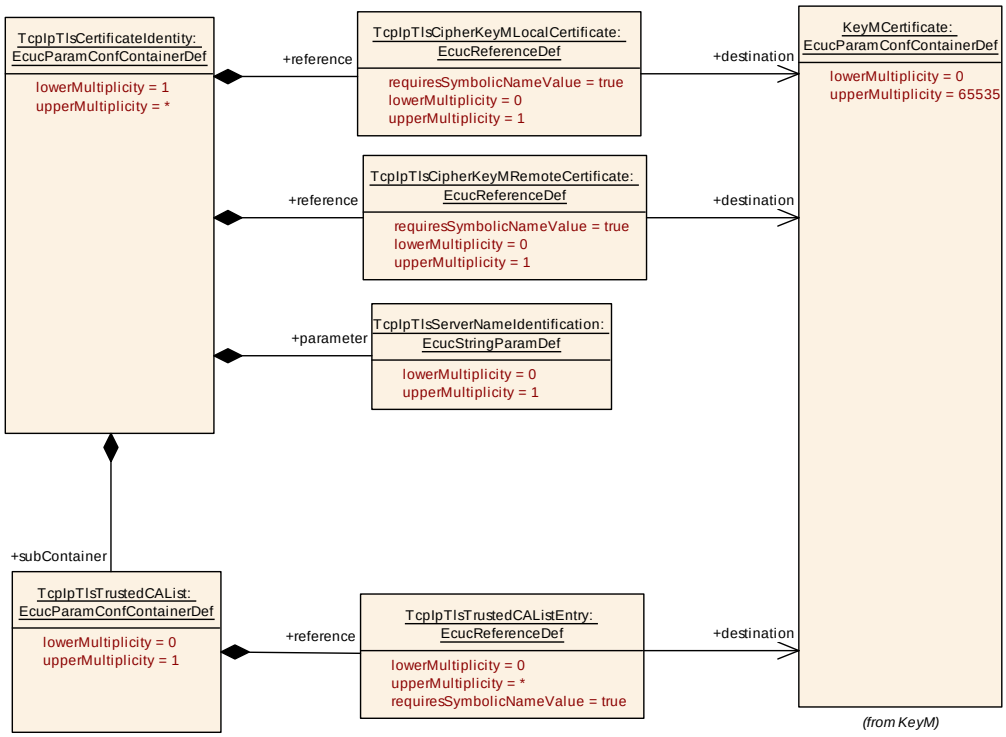

Figure 10.36: TcpIpTlsCertificateIdentity

10.2.54 TcpIpTlsCertificateIdentity

SWS Item	**[ECUC_TcpIp_00240]**
Container Name	TcpIpTlsCertificateIdentity
Parent Container	TcpIpTlsCiphersuites
Description	This container provides information about the certificates used for ciphersuites.
Configuration Parameters	

SWS Item	**[ECUC_TcpIp_00278]**		
Parameter Name	TcpIpTlsServerNameIdentification		
Parent Container	TcpIpTlsCertificateIdentity		
Description	Defines a server identification name. If present, the name will be added as an extension with the "TLS client hello" handshake message. The TLS server will check for the name to identify the server certificate.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

▽

△

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: Only needed if server name authentication is used.		

SWS Item	**[ECUC_TcpIp_00286]**		
Parameter Name	TcpIpTlsCipherKeyMLocalCertificate		
Parent Container	TcpIpTlsCertificateIdentity		
Description	Reference to a KeyM certificate used to address the local certificate.		
Multiplicity	0..1		
Type	Symbolic name reference to KeyMCertificate		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: Required if TcpIpTlsConnectionType is TLS_SERVER. Also required if TcpIpTlsConnectionType is TLS_CLIENT and the server requests a bidirectional authentication.		

SWS Item	**[ECUC_TcpIp_00287]**		
Parameter Name	TcpIpTlsCipherKeyMRemoteCertificate		
Parent Container	TcpIpTlsCertificateIdentity		
Description	Reference to KeyM certificate container to reference the remote certificate.		
Multiplicity	0..1		
Type	Symbolic name reference to KeyMCertificate		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: This optional parameter is needed by the TLS_CLIENT and is used to verify the certificate provided by the TLS_SERVER. It is also required by the TLS_SERVER if bidirectional authentication will be requested. Otherwise, this parameter can be omitted.		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpTlsTrustedCAList	0..1	This container contains references to trusted CA certificates, whose names are sent in the client's ClientHello message, if *TcpIpTlsUseExtensionTrustedCAKeys* is set to TRUE. **Tags:** atp.Status=draft

SWS Item	[ECUC_TcpIp_00335]		
Container Name	TcpIpTlsTrustedCAList		
Parent Container	TcpIpTlsCertificateIdentity		
Description	This container contains references to trusted CA certificates, whose names are sent in the client's ClientHello message, if *TcpIpTlsUseExtensionTrustedCAKeys* is set to TRUE. **Tags:** atp.Status=draft		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	[ECUC_TcpIp_00336]		
Parameter Name	TcpIpTlsTrustedCAListEntry		
Parent Container	TcpIpTlsTrustedCAList		
Description	Reference to a KeyM certificate of a root CA. The list is sent by the TLS client in the Client Hello's trusted_ca_keys extension **Tags:** atp.Status=draft		
Multiplicity	0..*		
Type	Symbolic name reference to KeyMCertificate		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

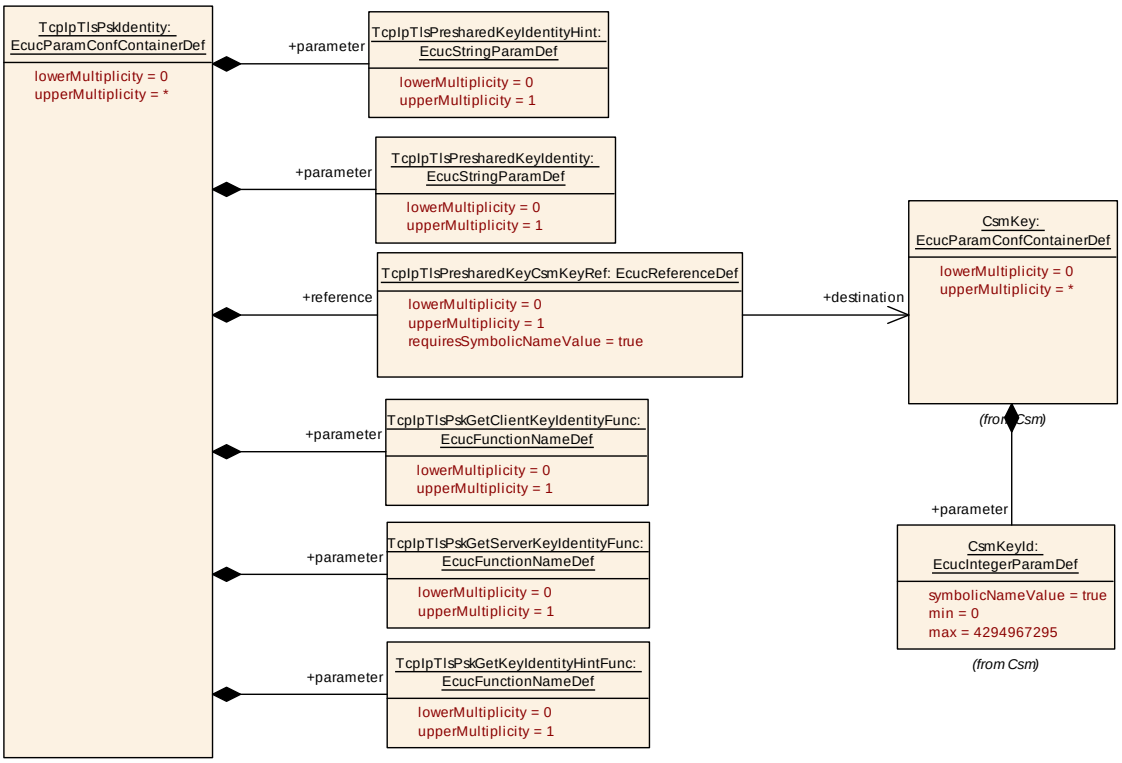

Figure 10.37: TcpIpTlsPskIdentity

10.2.55 TcpIpTlsPskIdentity

SWS Item	[ECUC_TcpIp_00241]
Container Name	TcpIpTlsPskIdentity
Parent Container	TcpIpTlsCiphersuites
Description	This container provides information about static definition of pre-shared keys. It is used during the handshake to negotiate pre-shared keys between a client and a server. Note: The callbacks for pre-shared keys are an alternative to the static definition. The callbacks allow to define the associated keys at runtime if pre-shared keys are used but no static definition is available. The container definition is used for static configuration.
Configuration Parameters	

SWS Item	[ECUC_TcpIp_00284]
Parameter Name	TcpIpTlsPresharedKeyIdentity
Parent Container	TcpIpTlsPskIdentity
Description	This item provides the key identification. The TLS client selects the pre-shared key based on the identification hint provided by the server and returns the key identification name back to the server.
Multiplicity	0..1
Type	EcucStringParamDef
Default value	–
Regular Expression	–
Post-Build Variant Multiplicity	false

△

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: The callback function < Up_TlsClientGetPskIdentity> is used if the ciphersuite defines pre-shared key but this parameter is not present.		

SWS Item	**[ECUC_TcpIp_00279]**		
Parameter Name	TcpIpTlsPresharedKeyIdentityHint		
Parent Container	TcpIpTlsPskIdentity		
Description	Provides the identity hint for a pre-shared key. This information is transmitted by the TLS Server to provide its identification to the TLS client. The TLS client uses the same information to select the pre-shared key.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: The callback function <Up_TlsServerGetPskIdentityHint> is used if the ciphersuite defines pre-shared key but this parameter is not present.		

SWS Item	**[ECUC_TcpIp_00281]**		
Parameter Name	TcpIpTlsPskGetClientKeyIdentityFunc		
Parent Container	TcpIpTlsPskIdentity		
Description	Defines the function name for the Up_TlsClientGetPskIdentity() callback.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	

▽

△

	Post-build time	–	
Scope / Dependency	scope: local dependency: This definition is needed if a pre-shared key ciphersuite is used and TcpIpTlsPresharedKeyIdentity configuration parameter is not present. In this case, the callback function will be used to query the key identification.		

SWS Item	**[ECUC_TcpIp_00283]**		
Parameter Name	TcpIpTlsPskGetKeyIdentityHintFunc		
Parent Container	TcpIpTlsPskIdentity		
Description	Defines the function name for the Up_TlsServerGetPskIdentityHint() callback.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: This definition is needed if a pre-shared key ciphersuite is used and TcpIpTlsPresharedKeyGetKeyIdentityHint configuration parameter is not present. In this case, the callback function will be used to query the key identity hint.		

SWS Item	**[ECUC_TcpIp_00282]**		
Parameter Name	TcpIpTlsPskGetServerKeyIdentityFunc		
Parent Container	TcpIpTlsPskIdentity		
Description	Defines the function name for the Up_TlsServerGetPskIdentity () callback.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: This definition is needed if a pre-shared key ciphersuite is used and TcpIpTlsPresharedKeyIdentity configuration parameter is not present. In this case, the callback function will be used to query the key identification.		

SWS Item	[ECUC_TcpIp_00280]		
Parameter Name	TcpIpTlsPresharedKeyCsmKeyRef		
Parent Container	TcpIpTlsPskIdentity		
Description	Reference to a CSM key associated to the CSM job that performs the PRF hash operation		
Multiplicity	0..1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: Callback <Up_Tls[Server|Client]GetPskIdentity> is used instead if this parameter is not present.		

No Included Containers

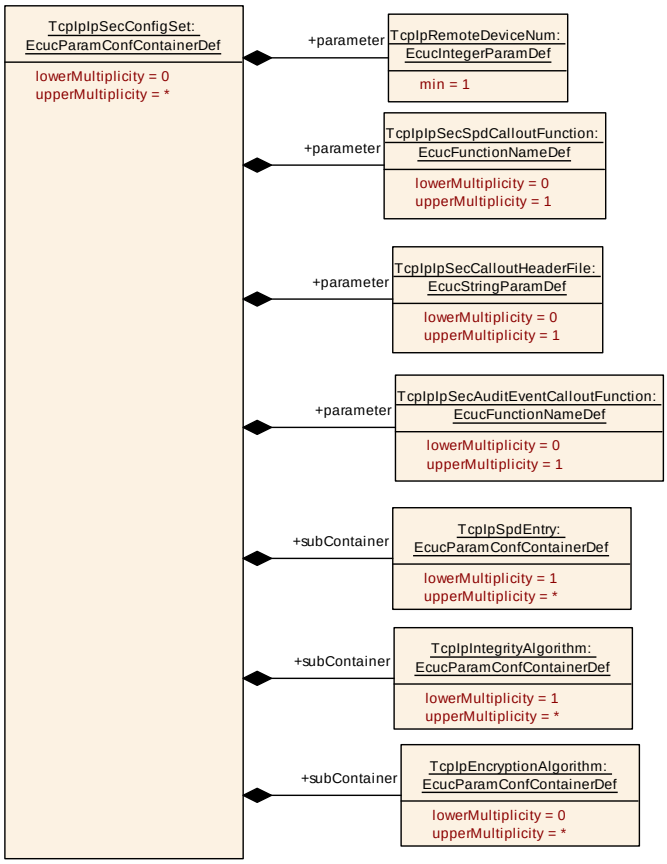

Figure 10.38: TcpIpIpSecConfig

10.2.56 TcpIpIpSecConfigSet

SWS Item	[ECUC_TcpIp_00288]		
Container Name	TcpIpIpSecConfigSet		
Parent Container	TcpIpConfig		
Description	Specifies the IPsec configuration.		
Post-Build Variant Multiplicity	false		
Configuration Parameters			

SWS Item	[ECUC_TcpIp_00292]		
Parameter Name	TcpIpIpSecAuditEventCalloutFunction		
Parent Container	TcpIpIpSecConfigSet		
Description	This parameter specifies the name of a callout function that will be called for each auditable event.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00291]		
Parameter Name	TcpIpIpSecCalloutHeaderFile		
Parent Container	TcpIpIpSecConfigSet		
Description	This parameter specifies the name of the header file containing the definition for the functions specified in TcpIpIpSecSpdCalloutFunction and TcpIpIpSecAuditEvent		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00290]		
Parameter Name	TcpIpIpSecSpdCalloutFunction		
Parent Container	TcpIpIpSecConfigSet		
Description	This parameter specifies the name of a callout function that shall be called for each Rx/Tx messag, after the IPsec has processed all corresponding SPD entries and has determined the policy. The callout function allows it to override the applied policy.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00289]		
Parameter Name	TcpIpRemoteDeviceNum		
Parent Container	TcpIpIpSecConfigSet		
Description	Amount of remote clients which will negotiate a Security Association (SA).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 18446744073709551615		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	Multiplicity	Scope / Dependency
TcpIpEncryptionAlgorithm	0..*	Container for configuration of supported encryption algorithm transforms. This container is used to configure supported algorithms for ESP. The transform algorithm must be configured in the Crypto module.
TcpIpIntegrityAlgorithm	1..*	Container for configuration of supported integrity algorithm transforms. This container is used to configure supported algorithms for AH. The transform algorithm must be configured in the Crypto module.
TcpIpSpdEntry	1..*	Entry of the Security Policy Database (SPD).

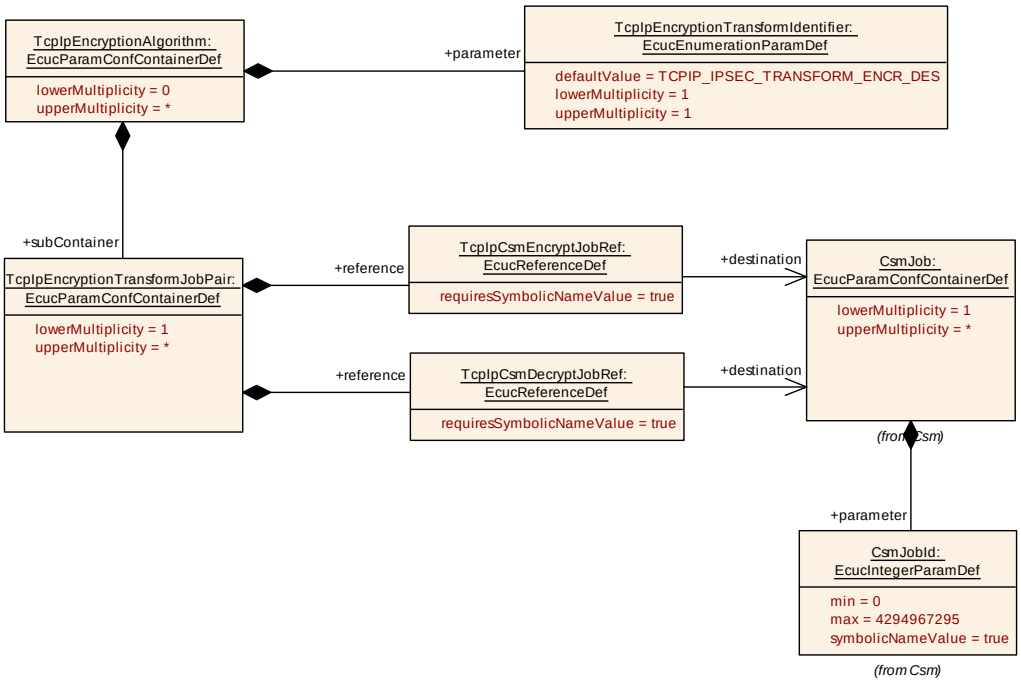

Figure 10.39: TcpIpEncryptionAlgorithm

10.2.57 TcpIpEncryptionAlgorithm

SWS Item	**[ECUC_TcpIp_00317]**		
Container Name	TcpIpEncryptionAlgorithm		
Parent Container	TcpIpIpSecConfigSet		
Description	Container for configuration of supported encryption algorithm transforms. This container is used to configure supported algorithms for ESP. The transform algorithm must be configured in the Crypto module.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_TcpIp_00311]**	
Parameter Name	TcpIpEncryptionTransformIdentifier	
Parent Container	TcpIpEncryptionAlgorithm	
Description	Encryption algorithm transform identifier. Parameter values are defined as per IETF RFC 7296 3.3.2	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	TCPIP_IPSEC_TRANSFORM_ENCR_3DES	–

▽

△

	TCPIP_IPSEC_TRANSFORM_ENCR_3IDEA	–
	TCPIP_IPSEC_TRANSFORM_ENCR_AES_CBC	–
	TCPIP_IPSEC_TRANSFORM_ENCR_AES_CCM_12	–
	TCPIP_IPSEC_TRANSFORM_ENCR_AES_CCM_16	–
	TCPIP_IPSEC_TRANSFORM_ENCR_AES_CCM_8	–
	TCPIP_IPSEC_TRANSFORM_ENCR_AES_CCM_8_IIV	–
	TCPIP_IPSEC_TRANSFORM_ENCR_AES_CTR	–
	TCPIP_IPSEC_TRANSFORM_ENCR_AES_GCM_12	–
	TCPIP_IPSEC_TRANSFORM_ENCR_AES_GCM_16	–
	TCPIP_IPSEC_TRANSFORM_ENCR_AES_GCM_16_IIV	–
	TCPIP_IPSEC_TRANSFORM_ENCR_AES_GCM_8	–
	TCPIP_IPSEC_TRANSFORM_ENCR_BLOWFISH	–
	TCPIP_IPSEC_TRANSFORM_ENCR_CAMELLIA_CBC	–
	TCPIP_IPSEC_TRANSFORM_ENCR_CAMELLIA_CCM_12	–
	TCPIP_IPSEC_TRANSFORM_ENCR_CAMELLIA_CCM_16	–
	TCPIP_IPSEC_TRANSFORM_ENCR_CAMELLIA_CCM_8	–
	TCPIP_IPSEC_TRANSFORM_ENCR_CAMELLIA_CTR	–
	TCPIP_IPSEC_TRANSFORM_ENCR_CAST	–
	TCPIP_IPSEC_TRANSFORM_ENCR_CHACHA20_POLY1305	–
	TCPIP_IPSEC_TRANSFORM_ENCR_CHACHA20_POLY1305_II	–
	TCPIP_IPSEC_TRANSFORM_ENCR_DES	–
	TCPIP_IPSEC_TRANSFORM_ENCR_DES_IV32	–
	TCPIP_IPSEC_TRANSFORM_ENCR_DES_IV64	–
	TCPIP_IPSEC_TRANSFORM_ENCR_IDEA	–
	TCPIP_IPSEC_TRANSFORM_ENCR_NULL	–
	TCPIP_IPSEC_TRANSFORM_ENCR_RC5	–
Default value	TCPIP_IPSEC_TRANSFORM_ENCR_DES	
Post-Build Variant Multiplicity	false	

▽

△

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpEncryptionTransformJobPair	1..*	Container for storing the CSM integrity transform job references for performing authentication. Valid for ESP and AH.At least one Integrity transform job pair needs to be configured for each Integrity Algorithm.

10.2.58 TcpIpEncryptionTransformJobPair

SWS Item	**[ECUC_TcpIp_00312]**		
Container Name	TcpIpEncryptionTransformJobPair		
Parent Container	TcpIpEncryptionAlgorithm		
Description	Container for storing the CSM integrity transform job references for performing authentication. Valid for ESP and AH.At least one Integrity transform job pair needs to be configured for each Integrity Algorithm.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_TcpIp_00314]**		
Parameter Name	TcpIpCsmDecryptJobRef		
Parent Container	TcpIpEncryptionTransformJobPair		
Description	The referenced Csm job is used for the execution of the CsmMacVerify primitive needed for this transform. Must be a valid decryption job of the parent type.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00313]		
Parameter Name	TcpIpCsmEncryptJobRef		
Parent Container	TcpIpEncryptionTransformJobPair		
Description	The referenced Csm job is used for the execution of the CsmMacGenerate primitive needed for this transform. Must be a valid encryption job of the parent type.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

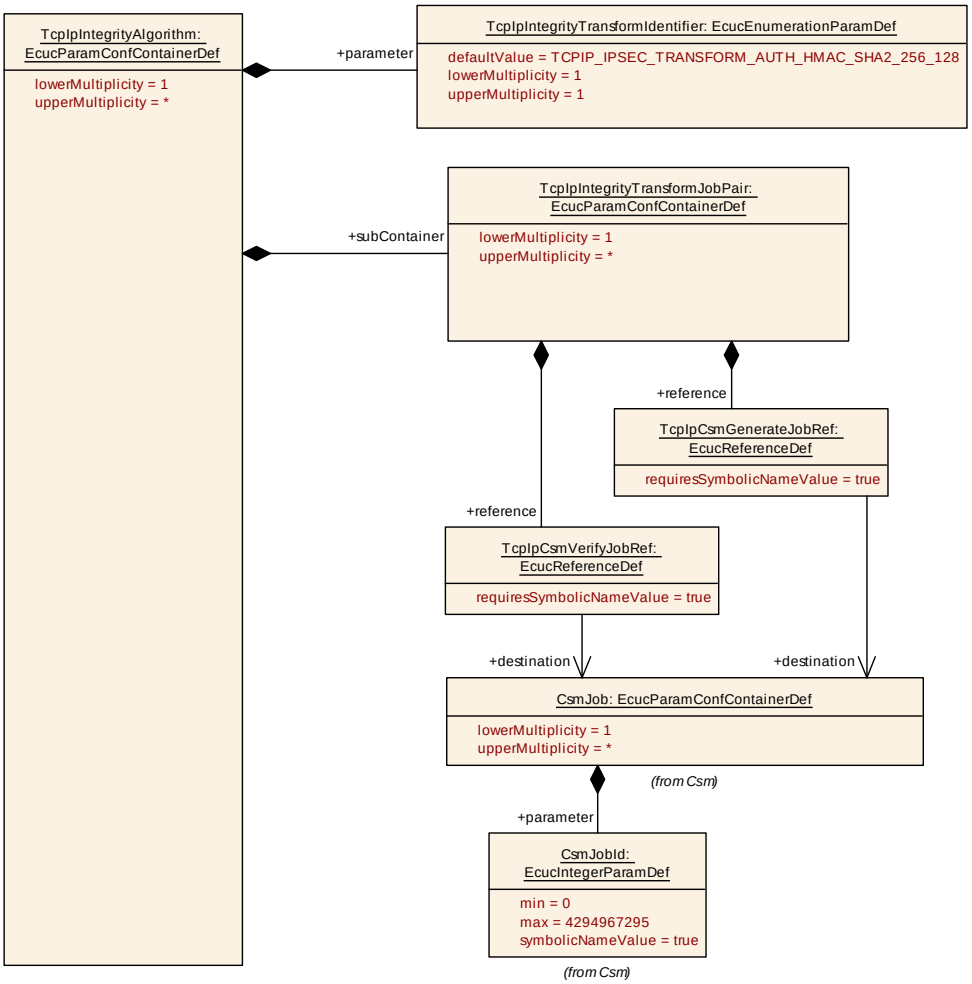

Figure 10.40: TcpIpIntegrityAlgorithm

10.2.59 TcpIpIntegrityAlgorithm

SWS Item	**[ECUC_TcpIp_00294]**		
Container Name	TcpIpIntegrityAlgorithm		
Parent Container	TcpIpIpSecConfigSet		
Description	Container for configuration of supported integrity algorithm transforms. This container is used to configure supported algorithms for AH. The transform algorithm must be configured in the Crypto module.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_TcpIp_00307]**	
Parameter Name	TcpIpIntegrityTransformIdentifier	
Parent Container	TcpIpIntegrityAlgorithm	
Description	Integrity algorithm transform identifier. Parameter values are defined as per IETF RFC 7296 3.3.2	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	TCPIP_IPSEC_TRANSFORM_AUTH_AES_128_GMAC	–
	TCPIP_IPSEC_TRANSFORM_AUTH_AES_192_GMAC	–
	TCPIP_IPSEC_TRANSFORM_AUTH_AES_256_GMAC	–
	TCPIP_IPSEC_TRANSFORM_AUTH_AES_CMAC_96	–
	TCPIP_IPSEC_TRANSFORM_AUTH_AES_XCBC_96	–
	TCPIP_IPSEC_TRANSFORM_AUTH_DES_MAC	–
	TCPIP_IPSEC_TRANSFORM_AUTH_HMAC_MD5_128	–
	TCPIP_IPSEC_TRANSFORM_AUTH_HMAC_MD5_96	–
	TCPIP_IPSEC_TRANSFORM_AUTH_HMAC_SHA1_160	–
	TCPIP_IPSEC_TRANSFORM_AUTH_HMAC_SHA1_96	–
	TCPIP_IPSEC_TRANSFORM_AUTH_HMAC_SHA2_256_128	–
	TCPIP_IPSEC_TRANSFORM_AUTH_HMAC_SHA2_384_192	–
	TCPIP_IPSEC_TRANSFORM_AUTH_HMAC_SHA2_512_256	–
	TCPIP_IPSEC_TRANSFORM_AUTH_KPDK_MD5	–
Default value	TCPIP_IPSEC_TRANSFORM_AUTH_HMAC_SHA2_256_128	
Post-Build Variant Multiplicity	false	

△

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
TcpIpIntegrityTransformJobPair	1..*	Container for storing the CSM integrity transform job references for performing authentication. Valid for ESP and AH.At least one Integrity transform job pair needs to be configured for each Integrity Algorithm.

10.2.60 TcpIpIntegrityTransformJobPair

SWS Item	**[ECUC_TcpIp_00308]**		
Container Name	TcpIpIntegrityTransformJobPair		
Parent Container	TcpIpIntegrityAlgorithm		
Description	Container for storing the CSM integrity transform job references for performing authentication. Valid for ESP and AH.At least one Integrity transform job pair needs to be configured for each Integrity Algorithm.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_TcpIp_00309]**		
Parameter Name	TcpIpCsmGenerateJobRef		
Parent Container	TcpIpIntegrityTransformJobPair		
Description	The referenced Csm job is used for the execution of the CsmMacGenerate primitive needed for this transform. Must be a valid MAC generate job of the parent type.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00310]**		
Parameter Name	TcpIpCsmVerifyJobRef		
Parent Container	TcpIpIntegrityTransformJobPair		
Description	The referenced Csm job is used for the execution of the CsmMacVerify primitive needed for this transform. Must be a valid MAC verify job of the parent type.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

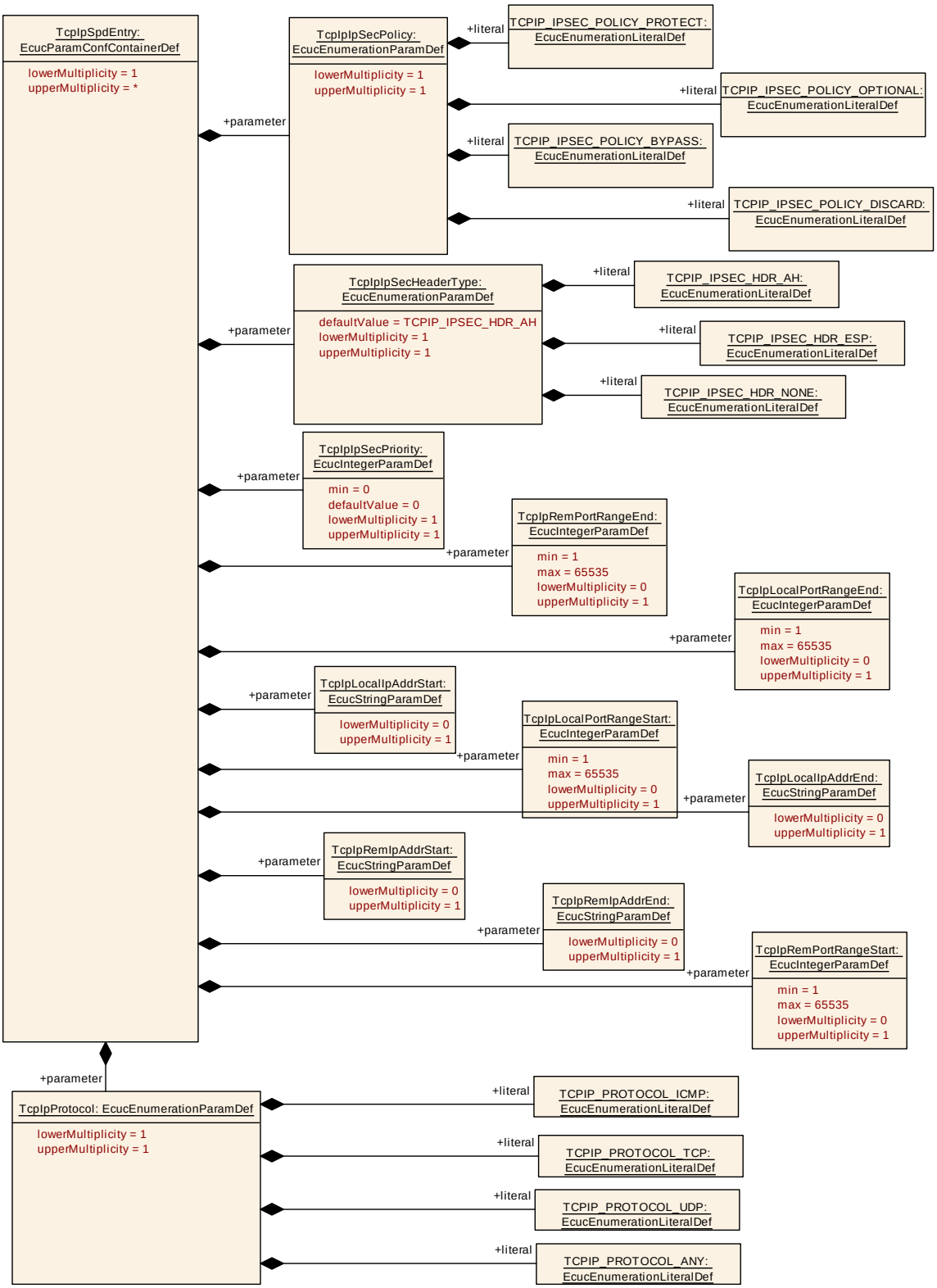

Figure 10.41: TcpIpSpdEntry

10.2.61 TcpIpSpdEntry

SWS Item	**[ECUC_TcpIp_00293]**		
Container Name	TcpIpSpdEntry		
Parent Container	TcpIpIpSecConfigSet		
Description	Entry of the Security Policy Database (SPD).		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_TcpIp_00297]**		
Parameter Name	TcpIpIpSecHeaderType		
Parent Container	TcpIpSpdEntry		
Description	Header type specifying the IPsec security mechanism.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TCPIP_IPSEC_HDR_AH	–	
	TCPIP_IPSEC_HDR_ESP	–	
	TCPIP_IPSEC_HDR_NONE	–	
Default value	TCPIP_IPSEC_HDR_AH		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00295]**	
Parameter Name	TcpIpIpSecPolicy	
Parent Container	TcpIpSpdEntry	
Description	Policy for usage of IPsec.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	TCPIP_IPSEC_POLICY_ BYPASS	–
	TCPIP_IPSEC_POLICY_ DISCARD	–
	TCPIP_IPSEC_POLICY_ OPTIONAL	–
	TCPIP_IPSEC_POLICY_ PROTECT	–
Post-Build Variant Multiplicity	false	
Post-Build Variant Value	false	

▽

△

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00296]**	
Parameter Name	TcpIpIpSecPriority	
Parent Container	TcpIpSpdEntry	
Description	Priority of the SPD entry. The processing of entries is based on priority, starting with the highest priority "0". The first matching SPD entry defines the policy.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 18446744073709551615	
Default value	0	
Post-Build Variant Multiplicity	false	
Post-Build Variant Value	false	
Scope / Dependency	scope: local	

SWS Item	**[ECUC_TcpIp_00301]**		
Parameter Name	TcpIpLocalIpAddrEnd		
Parent Container	TcpIpSpdEntry		
Description	End value of the remote IP address range.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00300]**
Parameter Name	TcpIpLocalIpAddrStart
Parent Container	TcpIpSpdEntry
Description	Start value of the local IP address range.
Multiplicity	0..1
Type	EcucStringParamDef
Default value	–

▽

△

Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00299]**		
Parameter Name	TcpIpLocalPortRangeEnd		
Parent Container	TcpIpSpdEntry		
Description	End value of the local port range.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00298]**		
Parameter Name	TcpIpLocalPortRangeStart		
Parent Container	TcpIpSpdEntry		
Description	Start value of the local port range.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00306]		
Parameter Name	TcpIpProtocol		
Parent Container	TcpIpSpdEntry		
Description	Relevant IP protocol. Note: As specified in IETF Rfc 4301 section 6, ICMP error messages will always be BYPASSed. The policy for TCPIP_PROTOCOL_ICMP only applies to ICMP non-error messages. (Echo reply/response).		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TCPIP_PROTOCOL_ANY	–	
	TCPIP_PROTOCOL_ICMP	–	
	TCPIP_PROTOCOL_TCP	–	
	TCPIP_PROTOCOL_UDP	–	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00303]		
Parameter Name	TcpIpRemIpAddrEnd		
Parent Container	TcpIpSpdEntry		
Description	End value of the remote IP address range.		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_TcpIp_00302]
Parameter Name	TcpIpRemIpAddrStart
Parent Container	TcpIpSpdEntry
Description	Start value of the remote IP address range.
Multiplicity	0..1
Type	EcucStringParamDef
Default value	–

▽

△

Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00305]**		
Parameter Name	TcpIpRemPortRangeEnd		
Parent Container	TcpIpSpdEntry		
Description	End value of the remote port range.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_TcpIp_00304]**		
Parameter Name	TcpIpRemPortRangeStart		
Parent Container	TcpIpSpdEntry		
Description	Start value of the remote port range.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

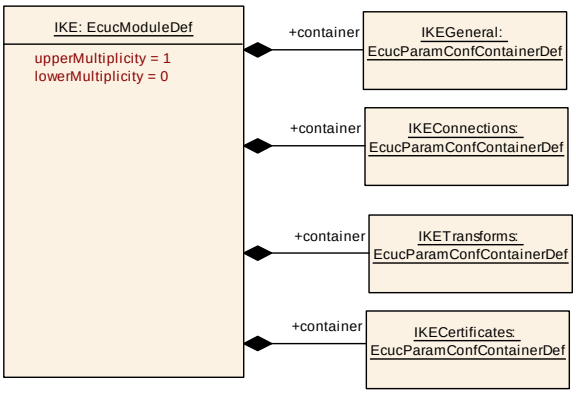

Figure 10.42: IKE

10.2.62 IKE

SWS Item	[ECUC_IKE_00001]
Module Name	IKE
Description	Description for the Internet Key Exchange.
Post-Build Variant Support	false
Supported Config Variants	VARIANT-PRE-COMPILE

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
IKECertificates	1	Container for configuration of IKE certificates.
IKEConnections	1	Container for configuration of IKE connections.
IKEGeneral	1	General module settings.
IKETransforms	1	Container for configuration of IKE transforms.

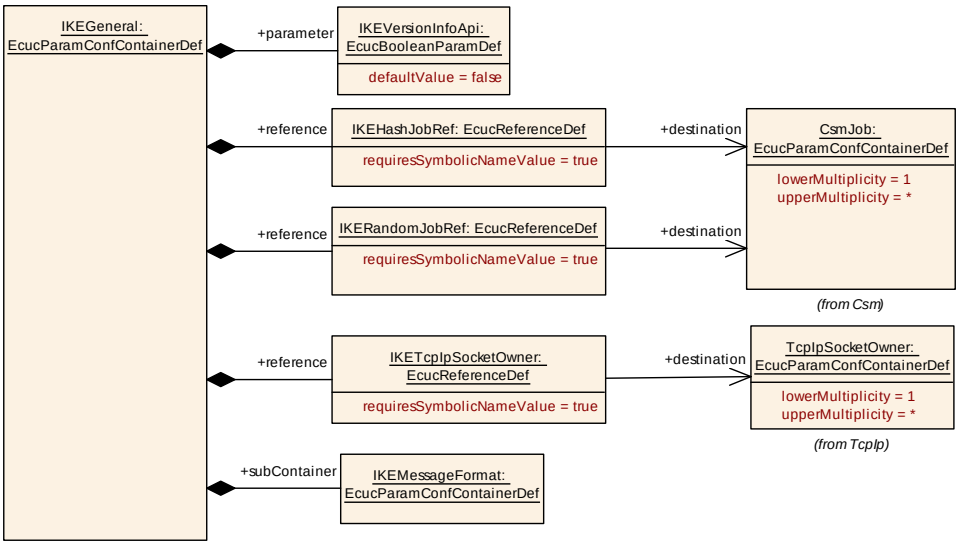

Figure 10.43: IKEGeneral

10.2.63 IKEGeneral

SWS Item	[ECUC_IKE_00002]		
Container Name	IKEGeneral		
Parent Container	IKE		
Description	General module settings.		
Configuration Parameters			

SWS Item	[ECUC_IKE_00008]		
Parameter Name	IKEVersionInfoApi		
Parent Container	IKEGeneral		
Description	Pre-processor switch to enable and disable availability of the API IKE_GetVersionInfo(). • True: API IKE_GetVersionInfo() is available. • False: API IKE_GetVersionInfo() is not available.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00009]
Parameter Name	IKEHashJobRef
Parent Container	IKEGeneral
Description	The referenced crypto job is used to calculate the SHA-1 hash of the Subject Public Key Info element needed for the encoding of the certification authorities.
Multiplicity	1
Type	Symbolic name reference to CsmJob
Scope / Dependency	scope: local

SWS Item	[ECUC_IKE_00010]		
Parameter Name	IKERandomJobRef		
Parent Container	IKEGeneral		
Description	The referenced crypto job is used for random number generation.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants

▽

△

	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00011]**		
Parameter Name	IKETcpIpSocketOwner		
Parent Container	IKEGeneral		
Description	The ID of the socket user.		
Multiplicity	1		
Type	Symbolic name reference to TcpIpSocketOwner		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
IKEMessageFormat	1	In order to deserialize the byte stream of IKE messages to data structures memory is statically allocated. Use the parameters in this container to minimize the used memory. But, configuring too low maximum values might result in unsuccessful deserializations of received IKE messages.

10.2.64 IKEMessageFormat

SWS Item	**[ECUC_IKE_00012]**		
Container Name	IKEMessageFormat		
Parent Container	IKEGeneral		
Description	In order to deserialize the byte stream of IKE messages to data structures memory is statically allocated. Use the parameters in this container to minimize the used memory. But, configuring too low maximum values might result in unsuccessful deserializations of received IKE messages.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_IKE_00014]**
Parameter Name	IKEMaxAttributesPerTransform
Parent Container	IKEMessageFormat

▽

△

Description	The maximum number of attributes a transform may contain.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 2		
Default value	2		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00020]**		
Parameter Name	IKEMaxCertPayloadsPerMessage		
Parent Container	IKEMessageFormat		
Description	The maximum number of Certificate payloads an IKE message may contain.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4		
Default value	4		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00021]**		
Parameter Name	IKEMaxCertreqPayloadsPerMessage		
Parent Container	IKEMessageFormat		
Description	The maximum number of Certificate Request payloads an IKE message may contain.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4		
Default value	4		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

▽

△

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00022]**		
Parameter Name	IKEMaxDeletePayloadsPerMessage		
Parent Container	IKEMessageFormat		
Description	The maximum number of Delete payloads an IKE message may contain.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 2		
Default value	2		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00024]**		
Parameter Name	IKEMaxInitMessageSize		
Parent Container	IKEMessageFormat		
Description	The maximum size of incoming IKE_INIT messages.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	256 .. 3000		
Default value	512		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00023]**
Parameter Name	IKEMaxNonceSize
Parent Container	IKEMessageFormat
Description	The maximum size of incoming nonces. Must be be at least 32 bytes and at least half the key size of the largest configured pseudorandom function (PRF).

▽

△

Multiplicity	1		
Type	EcucIntegerParamDef		
Range	32 .. 512		
Default value	64		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00019]**		
Parameter Name	IKEMaxNotifyPayloadsPerMessage		
Parent Container	IKEMessageFormat		
Description	The maximum number of Notify payloads an IKE message may contain.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 10		
Default value	10		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00018]**		
Parameter Name	IKEMaxPayloadsPerMessage		
Parent Container	IKEMessageFormat		
Description	The maximum number of payloads an IKE message may contain.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 20		
Default value	20		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants

▽

△

	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00015]**		
Parameter Name	IKEMaxProposalsPerSaPayload		
Parent Container	IKEMessageFormat		
Description	The maximum number of proposals a SA payload may contain.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 5		
Default value	5		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00017]**		
Parameter Name	IKEMaxSpisPerDeletePayload		
Parent Container	IKEMessageFormat		
Description	The maximum number of SPIs a Delete payload may contain.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 2		
Default value	2		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00016]**		
Parameter Name	IKEMaxTrafficSelectorsPerTsPayload		
Parent Container	IKEMessageFormat		
Description	The maximum number of traffic selectors a Traffic Selector payload may contain.		
Multiplicity	1		
Type	EcucIntegerParamDef		

▽

△

Range	1 .. 2		
Default value	2		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00013]**		
Parameter Name	IKEMaxTransformsPerProp		
Parent Container	IKEMessageFormat		
Description	The maximum number of transforms a proposal may contain.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 5		
Default value	5		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

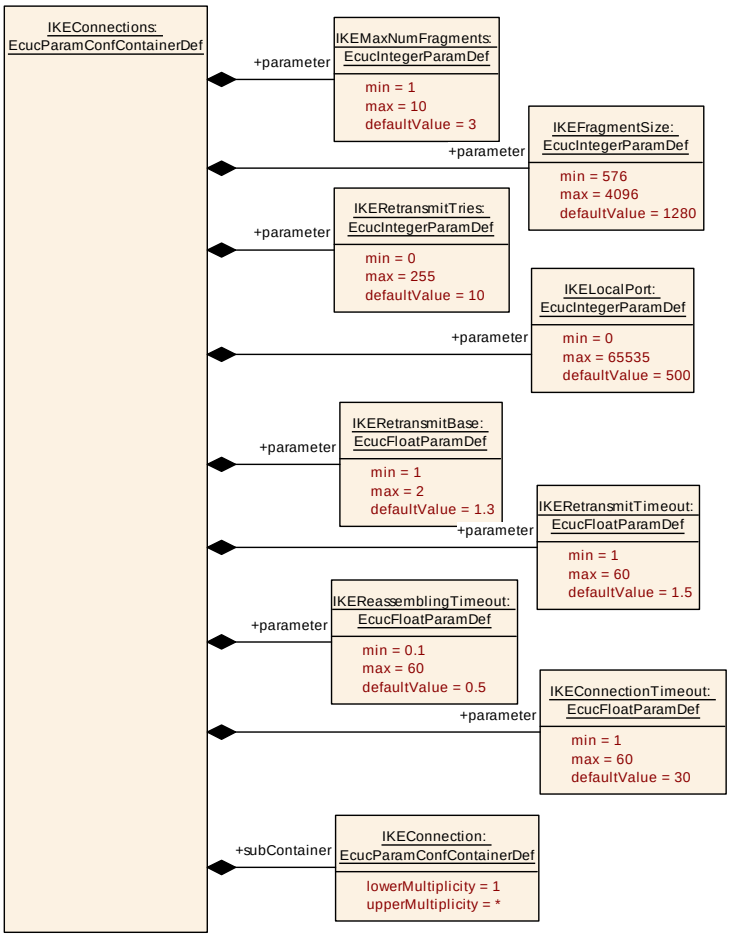

Figure 10.44: IKEConnections

10.2.65 IKEConnections

SWS Item	[ECUC_IKE_00003]
Container Name	IKEConnections
Parent Container	IKE
Description	Container for configuration of IKE connections.
Configuration Parameters	

SWS Item	[ECUC_IKE_00055]
Parameter Name	IKEConnectionTimeout
Parent Container	IKEConnections
Description	Timeout for etsablishing a connection in order to handle a "half open" state.
Multiplicity	1
Type	EcucFloatParamDef
Range	[1 .. 60]
Default value	30
Post-Build Variant Multiplicity	false

▽

△

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00049]**		
Parameter Name	IKEFragmentSize		
Parent Container	IKEConnections		
Description	The maximum size of IKE fragment messages when fragmentation is used. The resulting buffer size for subsequent fragment messages is (Number of Fragments * Fragment Size). This fragment size is the maximum IP datagram size, used for both RX and TX.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	576 .. 4096		
Default value	1280		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00051]**		
Parameter Name	IKELocalPort		
Parent Container	IKEConnections		
Description	The local port is the UDP port to listen to.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	500		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00048]		
Parameter Name	IKEMaxNumFragments		
Parent Container	IKEConnections		
Description	The maximum number of fragment messages into which the an IKE message might be divided. If this value is set to 1, fragmentation is not supported. The resulting buffer size for subsequent fragment messages is (Number of Fragments * Fragment Size). Used for both RX and TX and affects size of TX buffer.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 10		
Default value	3		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00054]		
Parameter Name	IKEReassemblingTimeout		
Parent Container	IKEConnections		
Description	The timeout for reassembling a fragmented message. All fragments of a message must be received within this interval, Otherwise all so far received fragments are discarded.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0.1 .. 60]		
Default value	0.5		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00052]	
Parameter Name	IKERetransmitBase	
Parent Container	IKEConnections	
Description	The base used for calculation of the exponantioal back-off of the retransmit timeouts.	
Multiplicity	1	
Type	EcucFloatParamDef	
Range	[1 .. 2]	
Default value	1.3	

▽

△

Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00053]**		
Parameter Name	IKERetransmitTimeout		
Parent Container	IKEConnections		
Description	The initial retransmit timeout.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[1 .. 60]		
Default value	1.5		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00050]**		
Parameter Name	IKERetransmitTries		
Parent Container	IKEConnections		
Description	The maximum number of retransmits of a request before giving up.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	10		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
IKEConnection	1..*	Container for configuration of IKE connection.

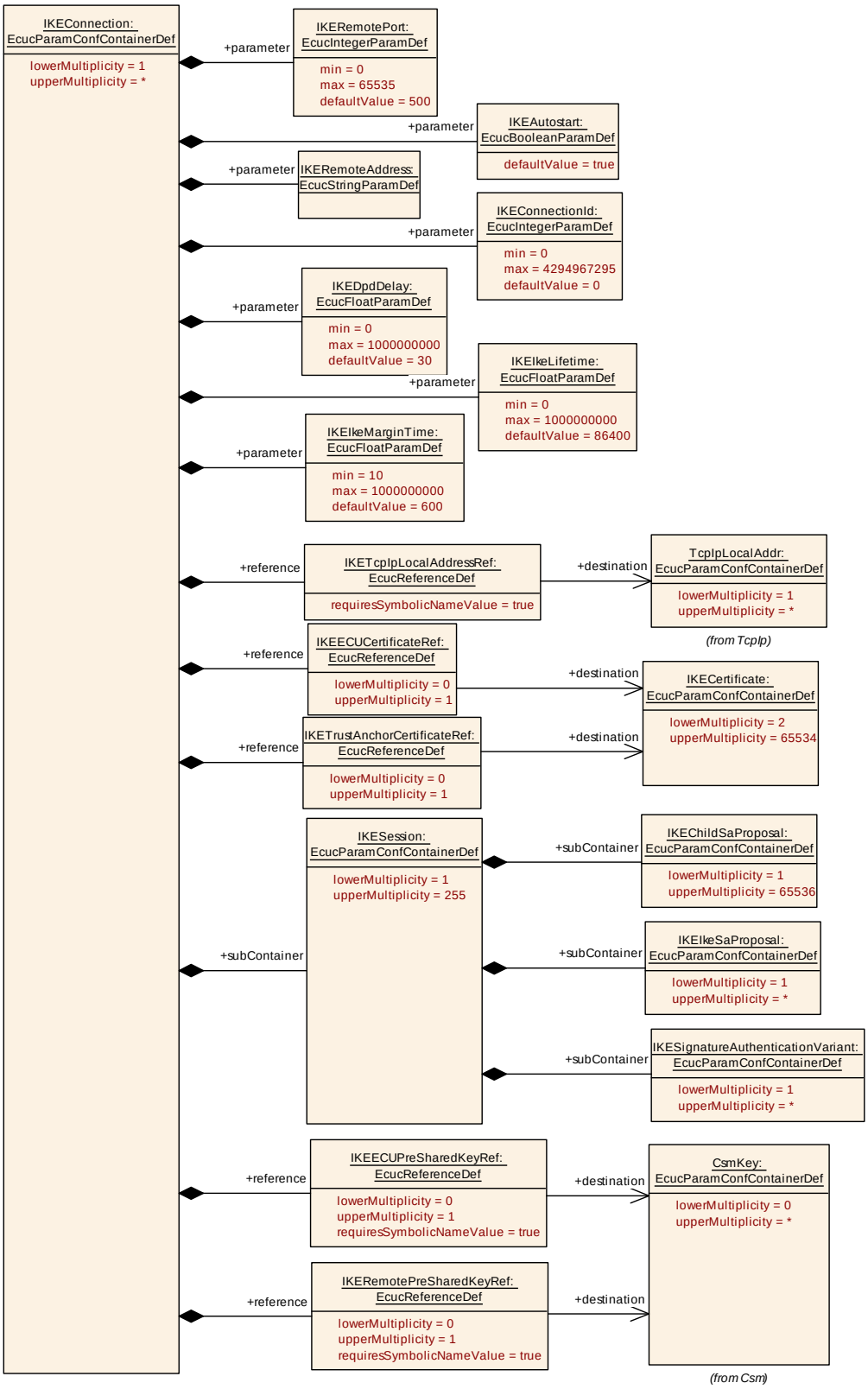

Figure 10.45: IKEConnection

10.2.66 IKEConnection

SWS Item	[ECUC_IKE_00056]		
Container Name	IKEConnection		
Parent Container	IKEConnections		
Description	Container for configuration of IKE connection.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	[ECUC_IKE_00058]		
Parameter Name	IKEAutostart		
Parent Container	IKEConnection		
Description	If enabled, IKE wil automatically initiate an IKE SA on this connection after start-up of the module.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00060]		
Parameter Name	IKEConnectionId		
Parent Container	IKEConnection		
Description	Identifier of the connection.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	0		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00063]		
Parameter Name	IKEDpdDelay		
Parent Container	IKEConnection		
Description	Specifies the interval in which Dead Peer Detection (DPD) packets shall be sent in the absence of other traffic Set to 0 to disable sending DPD packets.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 1000000000]		
Default value	30		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00064]		
Parameter Name	IKEIkeLifetime		
Parent Container	IKEConnection		
Description	Specifies the time after which an IKE SA is terminated. Set to 0 if IKE SA never expires.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 1000000000]		
Default value	86400		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00065]
Parameter Name	IKEIkeMarginTime
Parent Container	IKEConnection
Description	Specifies how many seconds before expiry an IKE SA should be renegotiated.
Multiplicity	1
Type	EcucFloatParamDef
Range	[10 .. 1000000000]
Default value	600
Post-Build Variant Multiplicity	false
Post-Build Variant Value	false

▽

△

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00059]**		
Parameter Name	IKERemoteAddress		
Parent Container	IKEConnection		
Description	The remote address is the IP address of the ECU which a IKE connection shall be established with, e.g. 192.168.50.101.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	–		
Regular Expression	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00057]**		
Parameter Name	IKERemotePort		
Parent Container	IKEConnection		
Description	The remote port is the UDP port of the ECU which a IKE connection shall be established with.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	500		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00067]		
Parameter Name	IKEECUCertificateRef		
Parent Container	IKEConnection		
Description	The ECU certificate is the end-entity certificate. The referenced certificate is the ECU certificate which cortains the public key used for authentication during the IKE connection setup.		
Multiplicity	0..1		
Type	Reference to IKECertificate		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00088]		
Parameter Name	IKEECUPreSharedKeyRef		
Parent Container	IKEConnection		
Description	The ECU's pre-shared key.		
Multiplicity	0..1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00089]		
Parameter Name	IKERemotePreSharedKeyRef		
Parent Container	IKEConnection		
Description	The referenced key is the key which is used to identify the remote ECU during the IKE connection setup.		
Multiplicity	0..1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	

△

	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00066]**		
Parameter Name	IKETcpIpLocalAddressRef		
Parent Container	IKEConnection		
Description	IP address table identifier assigned by TCP/IP stack.		
Multiplicity	1		
Type	Symbolic name reference to TcpIpLocalAddr		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00068]**		
Parameter Name	IKETrustAnchorCertificateRef		
Parent Container	IKEConnection		
Description	The referenced certificate is the Trust Anchor certificate which is used to identify the trusted Certification Authorities during the IKE connection setup.		
Multiplicity	0..1		
Type	Reference to IKECertificate		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
IKESession	1..255	Container for configuration of IKE session.

10.2.67 IKESession

SWS Item	[ECUC_IKE_00069]		
Container Name	IKESession		
Parent Container	IKEConnection		
Description	Container for configuration of IKE session.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Included Containers		
Container Name	Multiplicity	Scope / Dependency
IKEChildSaProposal	1..65536	Container for configuration of IKE Authentication Header (AH) or Encapsulating Security Payload (ESP) Security Association Proposals.
IKEIkeSaProposal	1..*	Container for configuration of IKE IKE Security Association Proposal.
IKESignatureAuthenticationVariant	1..*	Defining variants for the IKEv2 Authentication Method "Digitial Signature".

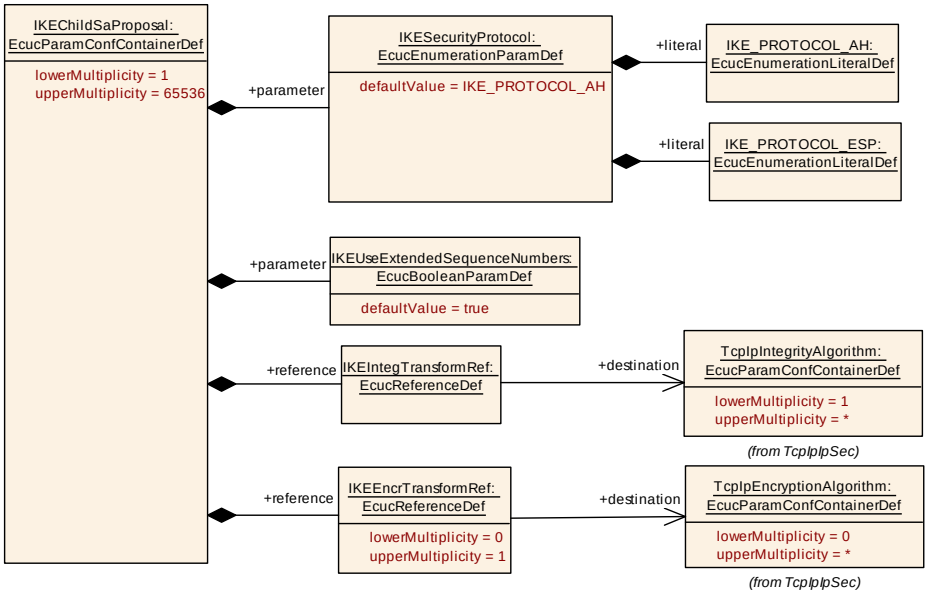

Figure 10.46: IKEChildSaProposal

10.2.68 IKEChildSaProposal

SWS Item	**[ECUC_IKE_00070]**		
Container Name	IKEChildSaProposal		
Parent Container	IKESession		
Description	Container for configuration of IKE Authentication Header (AH) or Encapsulating Security Payload (ESP) Security Association Proposals.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_IKE_00074]**		
Parameter Name	IKESecurityProtocol		
Parent Container	IKEChildSaProposal		
Description	The security protocol (i.e., AH or ESP) to be used for this Child SA.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	IKE_PROTOCOL_AH	–	
	IKE_PROTOCOL_ESP	–	
Default value	IKE_PROTOCOL_AH		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00075]**		
Parameter Name	IKEUseExtendedSequenceNumbers		
Parent Container	IKEChildSaProposal		
Description	Whether this Child SA should use Extended Sequence Numbers (ESN), i.e., 64-Bit instead of 32-Bit sequence numbers.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

▽

△

Scope / Dependency	scope: local

SWS Item	**[ECUC_IKE_00077]**		
Parameter Name	IKEEncrTransformRef		
Parent Container	IKEChildSaProposal		
Description	The referenced Encryption Algorithm is added to this proposal. Leave empty for AH and ESP in authentication-only mode.		
Multiplicity	0..1		
Type	Reference to TcpIpEncryptionAlgorithm		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00076]**		
Parameter Name	IKEIntegTransformRef		
Parent Container	IKEChildSaProposal		
Description	The referenced Integrity Algorithm is added to this proposal.		
Multiplicity	1		
Type	Reference to TcpIpIntegrityAlgorithm		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

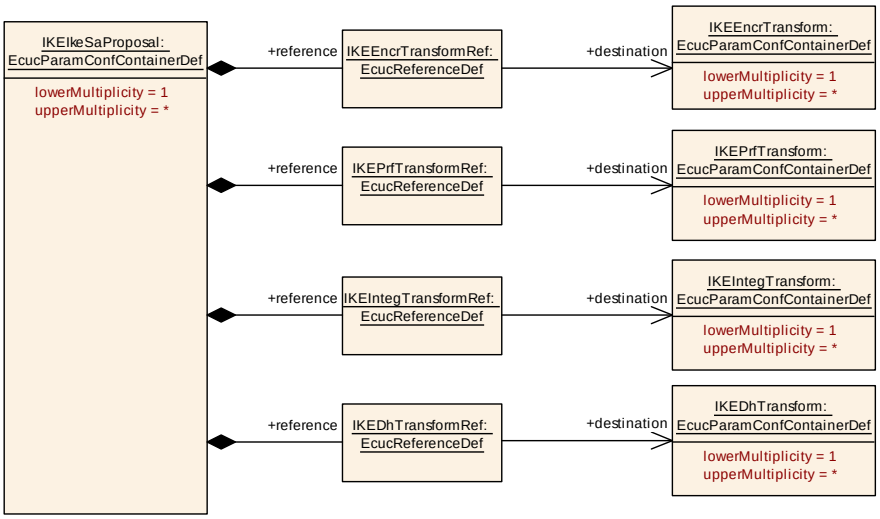

Figure 10.47: IKEIkeSaProposal

10.2.69 IKEIkeSaProposal

SWS Item	**[ECUC_IKE_00071]**		
Container Name	IKEIkeSaProposal		
Parent Container	IKESession		
Description	Container for configuration of IKE IKE Security Association Proposal.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_IKE_00082]**		
Parameter Name	IKEDhTransformRef		
Parent Container	IKEIkeSaProposal		
Description	The referenced Diffie-Hellman Group Transform is added to this proposal.		
Multiplicity	1		
Type	Reference to IKEDhTransform		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00079]		
Parameter Name	IKEEncrTransformRef		
Parent Container	IKEIkeSaProposal		
Description	The referenced Encryption Algorithm Transform is added to this proposal.		
Multiplicity	1		
Type	Reference to IKEEncrTransform		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00081]		
Parameter Name	IKEIntegTransformRef		
Parent Container	IKEIkeSaProposal		
Description	The referenced Integrity Algorithm Transform is added to this proposal.		
Multiplicity	1		
Type	Reference to IKEIntegTransform		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00080]		
Parameter Name	IKEPrfTransformRef		
Parent Container	IKEIkeSaProposal		
Description	The referenced Pseudorandom Function Transform is added to this proposal.		
Multiplicity	1		
Type	Reference to IKEPrfTransform		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

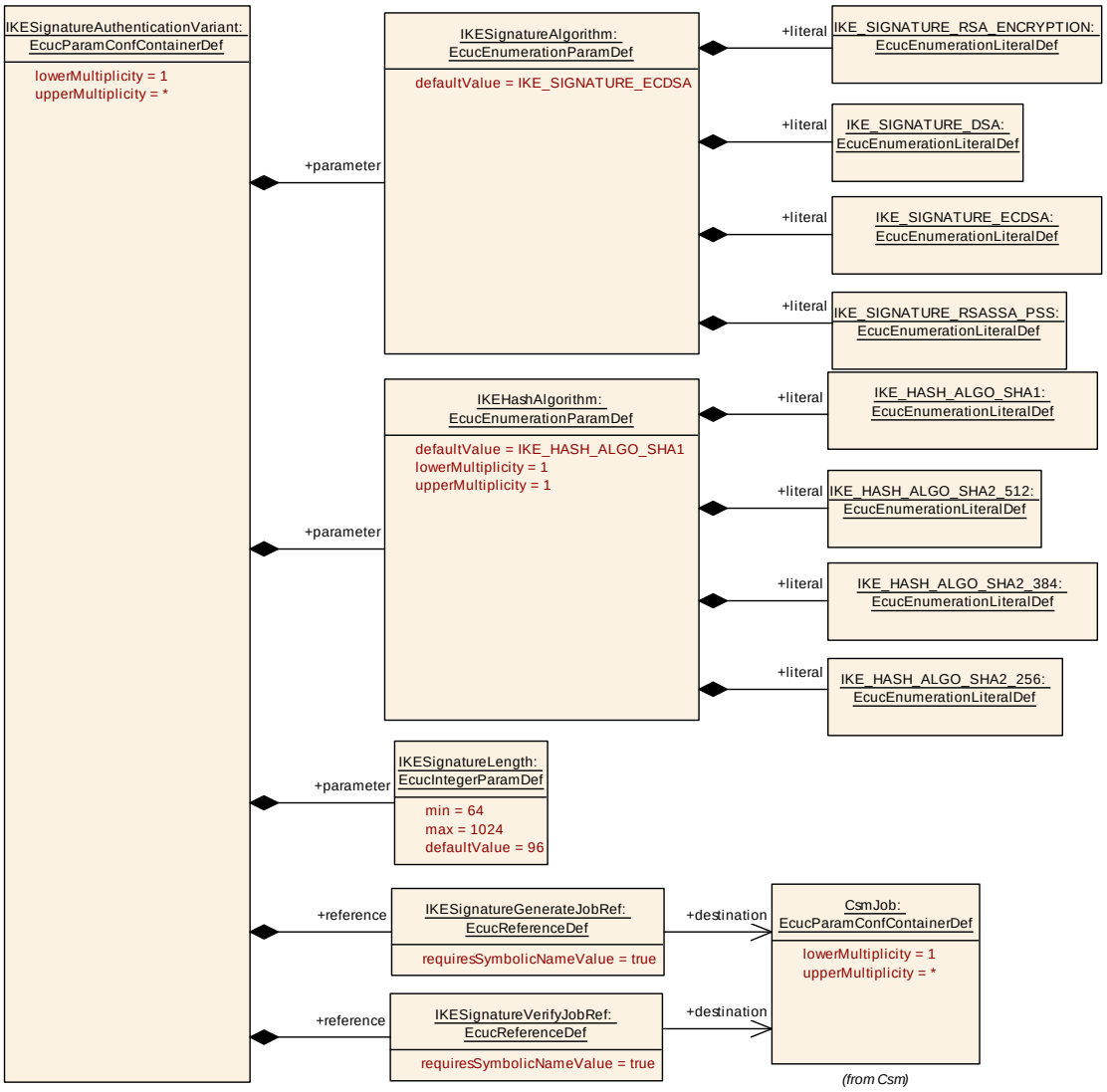

Figure 10.48: IKESignatureAuthenticationVariant

10.2.70 IKESignatureAuthenticationVariant

SWS Item	[ECUC_IKE_00072]		
Container Name	IKESignatureAuthenticationVariant		
Parent Container	IKESession		
Description	Defining variants for the IKEv2 Authentication Method "Digitial Signature".		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	

△

	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_IKE_00084]**		
Parameter Name	IKEHashAlgorithm		
Parent Container	IKESignatureAuthenticationVariant		
Description	Pre-hashing Algorithm. Please adapt to the referenced Csm jobs.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	IKE_HASH_ALGO_SHA1	–	
	IKE_HASH_ALGO_SHA2_256	–	
	IKE_HASH_ALGO_SHA2_384	–	
	IKE_HASH_ALGO_SHA2_512	–	
Default value	IKE_HASH_ALGO_SHA1		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00083]**		
Parameter Name	IKESignatureAlgorithm		
Parent Container	IKESignatureAuthenticationVariant		
Description	Signature Algorithm. Please adapt to the referenced Csm jobs.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	IKE_SIGNATURE_DSA	–	
	IKE_SIGNATURE_ECDSA	–	
	IKE_SIGNATURE_RSASSA_PSS	–	
	IKE_SIGNATURE_RSA_ ENCRYPTION	–	
Default value	IKE_SIGNATURE_ECDSA		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00085]		
Parameter Name	IKESignatureLength		
Parent Container	IKESignatureAuthenticationVariant		
Description	The length of a signature generated by the configured generation job and verified by the configured verification job. E.g. 64 for ECDSA-256 or 96 for ECDSA-386.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	64 .. 1024		
Default value	96		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00086]		
Parameter Name	IKESignatureGenerateJobRef		
Parent Container	IKESignatureAuthenticationVariant		
Description	The referenced Csm job is used for the execution of the CsmSignatureGenerate primitive needed for this transform.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	[ECUC_IKE_00087]		
Parameter Name	IKESignatureVerifyJobRef		
Parent Container	IKESignatureAuthenticationVariant		
Description	The referenced Csm job is used for the execution of the CsmSignatureVerify primitive needed for this transform.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants

▽

△

	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

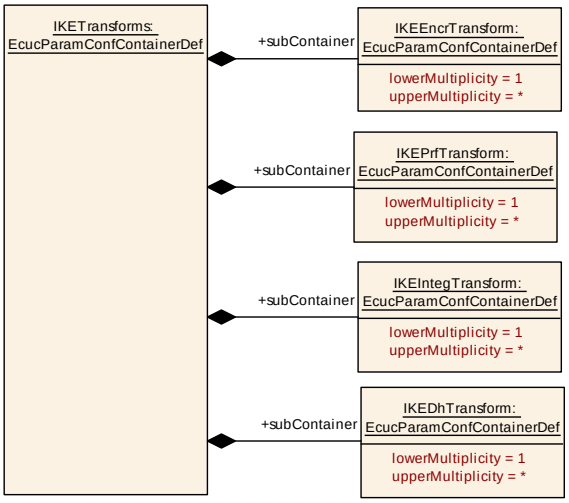

Figure 10.49: IKETransforms

10.2.71 IKETransforms

SWS Item	**[ECUC_IKE_00004]**
Container Name	IKETransforms
Parent Container	IKE
Description	Container for configuration of IKE transforms.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
IKEDhTransform	1..*	Container for configuration of Diffie-Hellman Group Transform.
IKEEncrTransform	1..*	Container for configuration of Encryption Algorithm Transform.
IKEIntegTransform	1..*	Container for configuration of Integrity Algorithm Transform.
IKEPrfTransform	1..*	Container for configuration of Pseudorandom Function Transform.

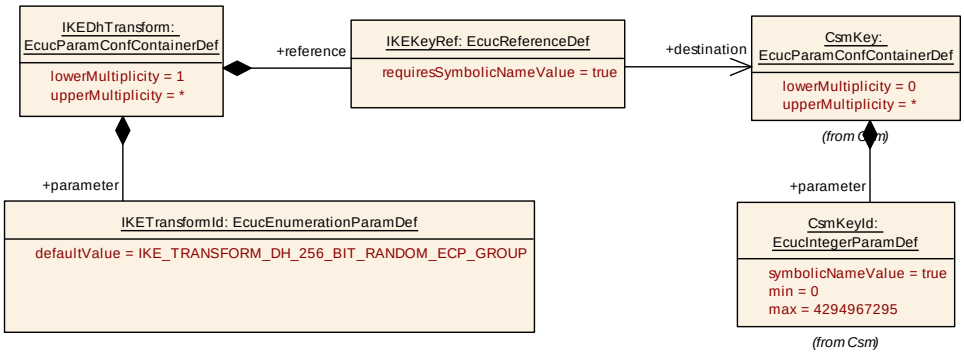

Figure 10.50: IKEDhTransform

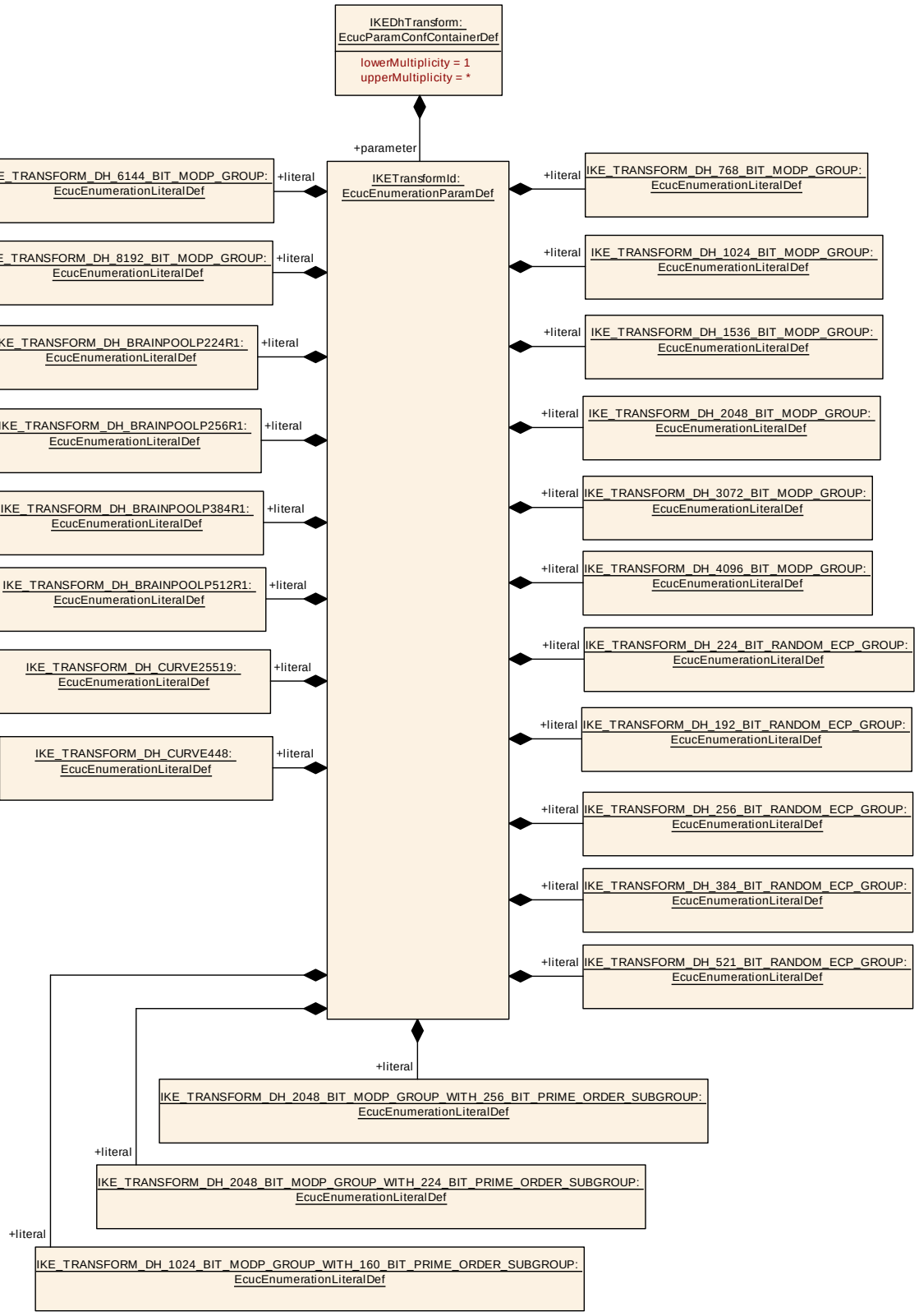

Figure 10.51: IKEEDhTransformIKETransformId

10.2.72 IKEDhTransform

SWS Item	[ECUC_IKE_00028]		
Container Name	IKEDhTransform		
Parent Container	IKETransforms		
Description	Container for configuration of Diffie-Hellman Group Transform.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	[ECUC_IKE_00038]	
Parameter Name	IKETransformId	
Parent Container	IKEDhTransform	
Description	Diffie-Hellman Group Transform ID.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	IKE_TRANSFORM_DH_1024_BIT_MODP_GROUP	–
	IKE_TRANSFORM_DH_1024_BIT_MODP_GROUP_WITH_160_BIT_PRIME_ORDER_SUBGROUP	–
	IKE_TRANSFORM_DH_1536_BIT_MODP_GROUP	–
	IKE_TRANSFORM_DH_192_BIT_RANDOM_ECP_GROUP	–
	IKE_TRANSFORM_DH_2048_BIT_MODP_GROUP	–
	IKE_TRANSFORM_DH_2048_BIT_MODP_GROUP_WITH_224_BIT_PRIME_ORDER_SUBGROUP	–
	IKE_TRANSFORM_DH_2048_BIT_MODP_GROUP_WITH_256_BIT_PRIME_ORDER_SUBGROUP	–
	IKE_TRANSFORM_DH_224_BIT_RANDOM_ECP_GROUP	–
	IKE_TRANSFORM_DH_256_BIT_RANDOM_ECP_GROUP	–
	IKE_TRANSFORM_DH_3072_BIT_MODP_GROUP	–
	IKE_TRANSFORM_DH_384_BIT_RANDOM_ECP_GROUP	–
	IKE_TRANSFORM_DH_4096_BIT_MODP_GROUP	–
	IKE_TRANSFORM_DH_521_BIT_RANDOM_ECP_GROUP	–
	IKE_TRANSFORM_DH_6144_BIT_MODP_GROUP	–

△

	IKE_TRANSFORM_DH_768_BIT_MODP_GROUP	–	
	IKE_TRANSFORM_DH_8192_BIT_MODP_GROUP	–	
	IKE_TRANSFORM_DH_BRAINPOOLP224R1	–	
	IKE_TRANSFORM_DH_BRAINPOOLP256R1	–	
	IKE_TRANSFORM_DH_BRAINPOOLP384R1	–	
	IKE_TRANSFORM_DH_BRAINPOOLP512R1	–	
	IKE_TRANSFORM_DH_CURVE25519	–	
	IKE_TRANSFORM_DH_CURVE448	–	
Default value	IKE_TRANSFORM_DH_256_BIT_RANDOM_ECP_GROUP		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00039]**		
Parameter Name	IKEKeyRef		
Parent Container	IKEDhTransform		
Description	The referenced Csm key is used for the execution of key management functions needed for this transform.		
Multiplicity	1		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

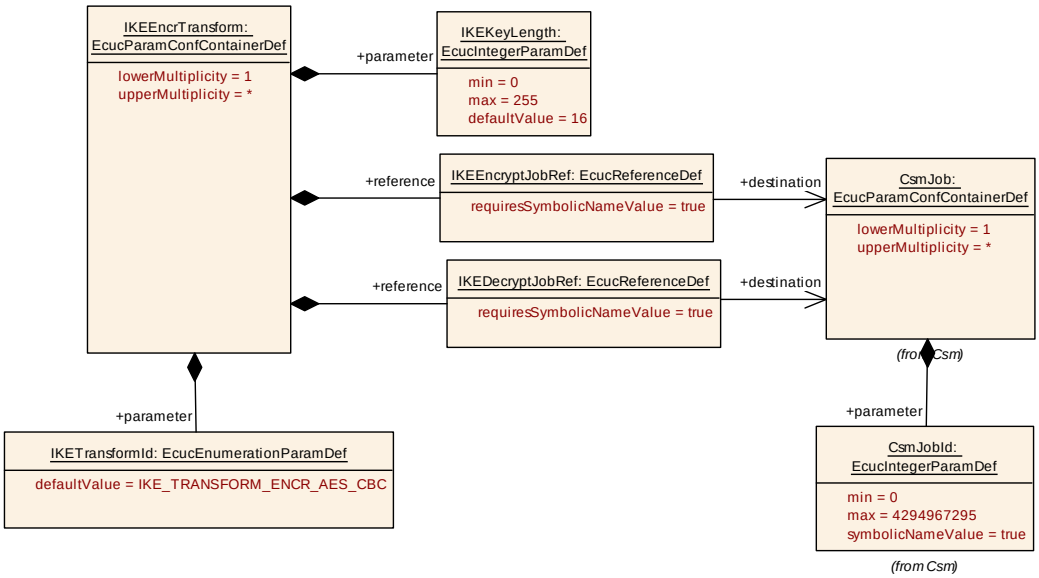

Figure 10.52: IKEEncrTransform

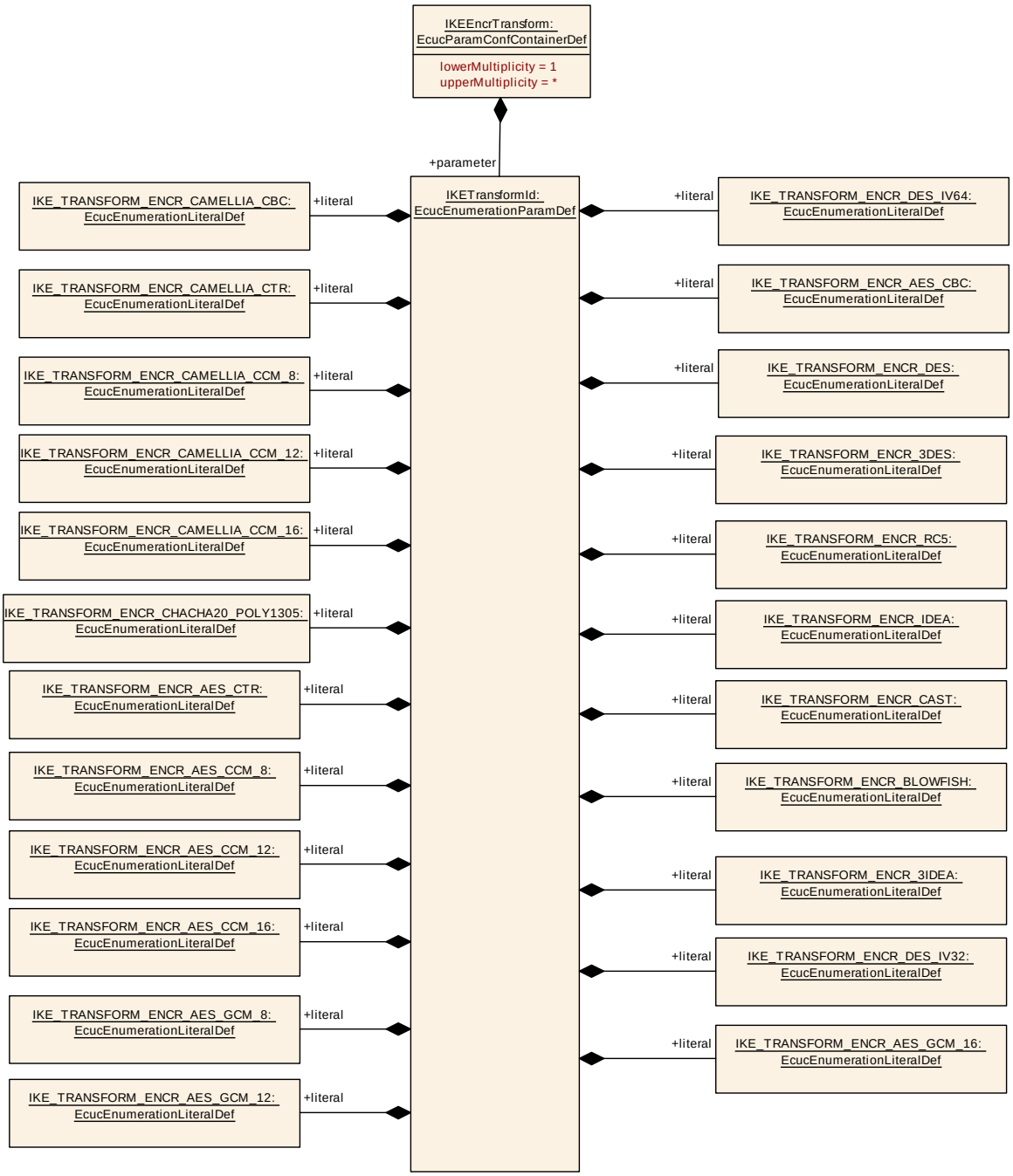

Figure 10.53: IKEEncrTransformIKETransformId

10.2.73 IKEEncrTransform

SWS Item	[ECUC_IKE_00025]
Container Name	IKEEncrTransform
Parent Container	IKETransforms

▽

△

Description	Container for configuration of Encryption Algorithm Transform.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_IKE_00030]**		
Parameter Name	IKEKeyLength		
Parent Container	IKEEncrTransform		
Description	The key length of the encryption algorithm in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	16		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00029]**	
Parameter Name	IKETransformId	
Parent Container	IKEEncrTransform	
Description	Encryption Algorithm Transform ID.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	IKE_TRANSFORM_ENCR_3DES	–
	IKE_TRANSFORM_ENCR_3IDEA	–
	IKE_TRANSFORM_ENCR_AES_CBC	–
	IKE_TRANSFORM_ENCR_AES_CCM_12	–
	IKE_TRANSFORM_ENCR_AES_CCM_16	–
	IKE_TRANSFORM_ENCR_AES_CCM_8	–
	IKE_TRANSFORM_ENCR_AES_CTR	–
	IKE_TRANSFORM_ENCR_AES_GCM_12	–
	IKE_TRANSFORM_ENCR_AES_GCM_16	–

▽

△

	IKE_TRANSFORM_ENCR_AES_GCM_8	–	
	IKE_TRANSFORM_ENCR_BLOWFISH	–	
	IKE_TRANSFORM_ENCR_CAMELLIA_CBC	–	
	IKE_TRANSFORM_ENCR_CAMELLIA_CCM_12	–	
	IKE_TRANSFORM_ENCR_CAMELLIA_CCM_16	–	
	IKE_TRANSFORM_ENCR_CAMELLIA_CCM_8	–	
	IKE_TRANSFORM_ENCR_CAMELLIA_CTR	–	
	IKE_TRANSFORM_ENCR_CAST	–	
	IKE_TRANSFORM_ENCR_CHACHA20_POLY1305	–	
	IKE_TRANSFORM_ENCR_DES	–	
	IKE_TRANSFORM_ENCR_DES_IV32	–	
	IKE_TRANSFORM_ENCR_DES_IV64	–	
	IKE_TRANSFORM_ENCR_IDEA	–	
	IKE_TRANSFORM_ENCR_RC5	–	
Default value	IKE_TRANSFORM_ENCR_AES_CBC		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00032]**		
Parameter Name	IKEDecryptJobRef		
Parent Container	IKEEncrTransform		
Description	The referenced Csm job is used for the execution of the CsmDecrypt primitive needed for this transform.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

▽

△

Scope / Dependency	scope: local

SWS Item	**[ECUC_IKE_00031]**		
Parameter Name	IKEEncryptJobRef		
Parent Container	IKEEncrTransform		
Description	The referenced Csm job is used for the execution of the CsmEncrypt primitive needed for this transform.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

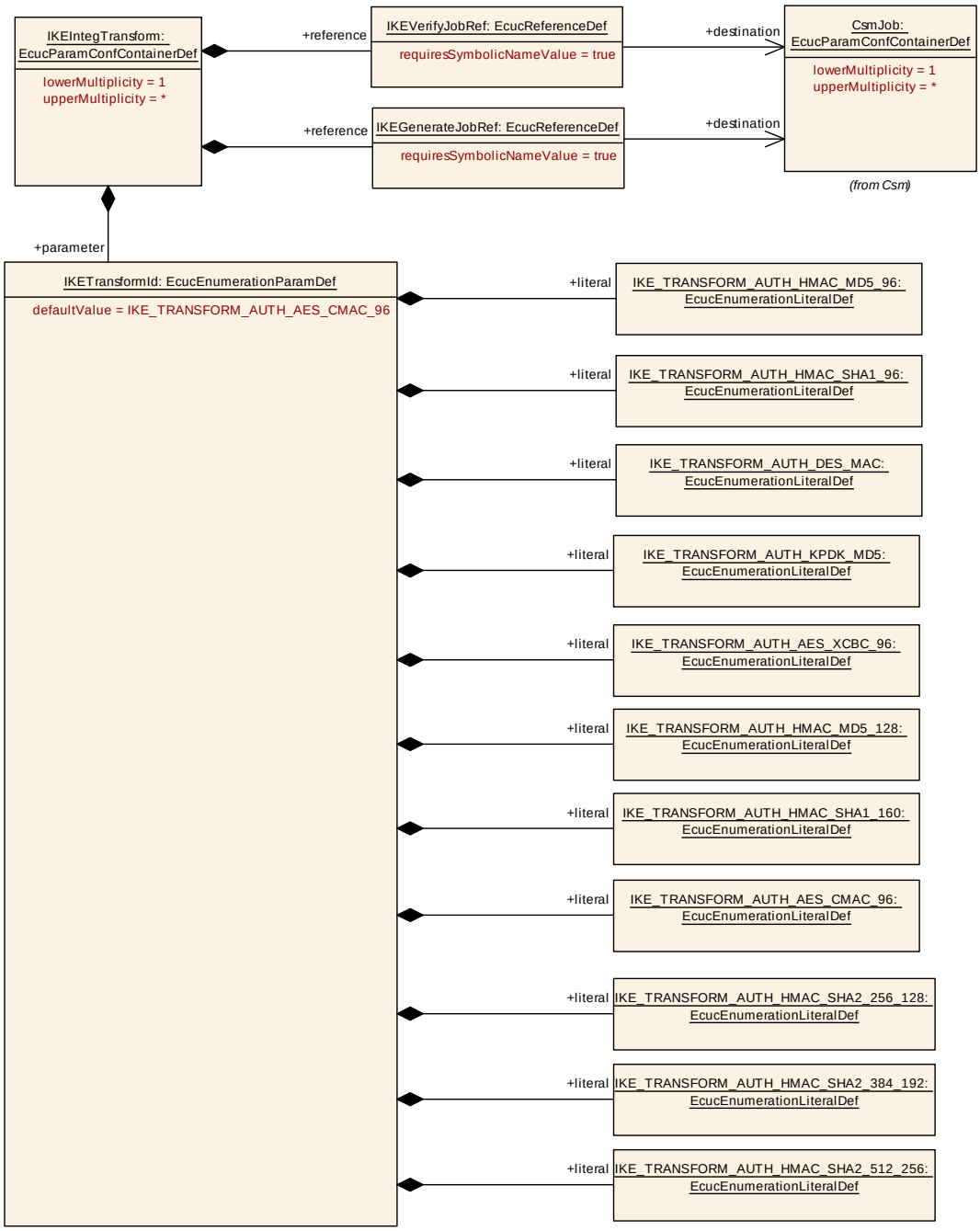

Figure 10.54: IKEIntegTransform

10.2.74 IKEIntegTransform

SWS Item	[ECUC_IKE_00027]
Container Name	IKEIntegTransform
Parent Container	IKETransforms

▽

△

Description	Container for configuration of Integrity Algorithm Transform.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_IKE_00037]**		
Parameter Name	IKETransformId		
Parent Container	IKEIntegTransform		
Description	Integrity Algorithm Transform ID.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	IKE_TRANSFORM_AUTH_AES_CMAC_96	–	
	IKE_TRANSFORM_AUTH_AES_XCBC_96	–	
	IKE_TRANSFORM_AUTH_DES_MAC	–	
	IKE_TRANSFORM_AUTH_HMAC_MD5_128	–	
	IKE_TRANSFORM_AUTH_HMAC_MD5_96	–	
	IKE_TRANSFORM_AUTH_HMAC_SHA1_160	–	
	IKE_TRANSFORM_AUTH_HMAC_SHA1_96	–	
	IKE_TRANSFORM_AUTH_HMAC_SHA2_256_128	–	
	IKE_TRANSFORM_AUTH_HMAC_SHA2_384_192	–	
	IKE_TRANSFORM_AUTH_HMAC_SHA2_512_256	–	
	IKE_TRANSFORM_AUTH_KPDK_MD5	–	
Default value	IKE_TRANSFORM_AUTH_AES_CMAC_96		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00036]**
Parameter Name	IKEGenerateJobRef
Parent Container	IKEIntegTransform

▽

△

Description	The referenced Csm job is used for the execution of the CsmMacGenerate primitive needed for this transform.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00035]**		
Parameter Name	IKEVerifyJobRef		
Parent Container	IKEIntegTransform		
Description	The referenced Csm job is used for the execution of the CsmMacVerify primitive needed for this transform.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

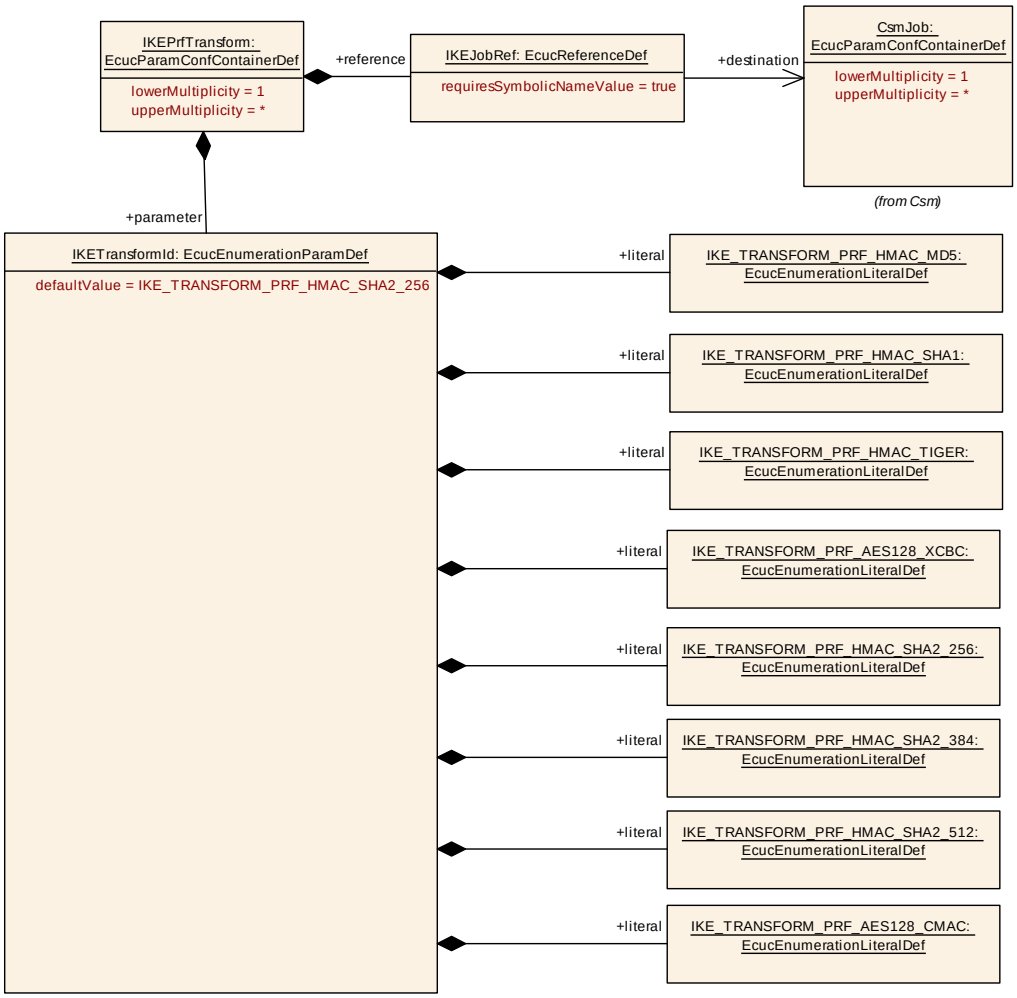

Figure 10.55: IKEPrfTransform

10.2.75 IKEPrfTransform

SWS Item	[ECUC_IKE_00026]		
Container Name	IKEPrfTransform		
Parent Container	IKETransforms		
Description	Container for configuration of Pseudorandom Function Transform.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	[ECUC_IKE_00033]
Parameter Name	IKETransformId
Parent Container	IKEPrfTransform

▽

△

Description	Pseudorandom Function Transform ID.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	IKE_TRANSFORM_PRF_AES128_CMAC	–	
	IKE_TRANSFORM_PRF_AES128_XCBC	–	
	IKE_TRANSFORM_PRF_HMAC_MD5	–	
	IKE_TRANSFORM_PRF_HMAC_SHA1	–	
	IKE_TRANSFORM_PRF_HMAC_SHA2_256	–	
	IKE_TRANSFORM_PRF_HMAC_SHA2_384	–	
	IKE_TRANSFORM_PRF_HMAC_SHA2_512	–	
	IKE_TRANSFORM_PRF_HMAC_TIGER	–	
Default value	IKE_TRANSFORM_PRF_HMAC_SHA2_256		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00034]**		
Parameter Name	IKEJobRef		
Parent Container	IKEPrfTransform		
Description	The referenced Csm job is used for the execution of the CsmMacGenerate primitive needed for this transform.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

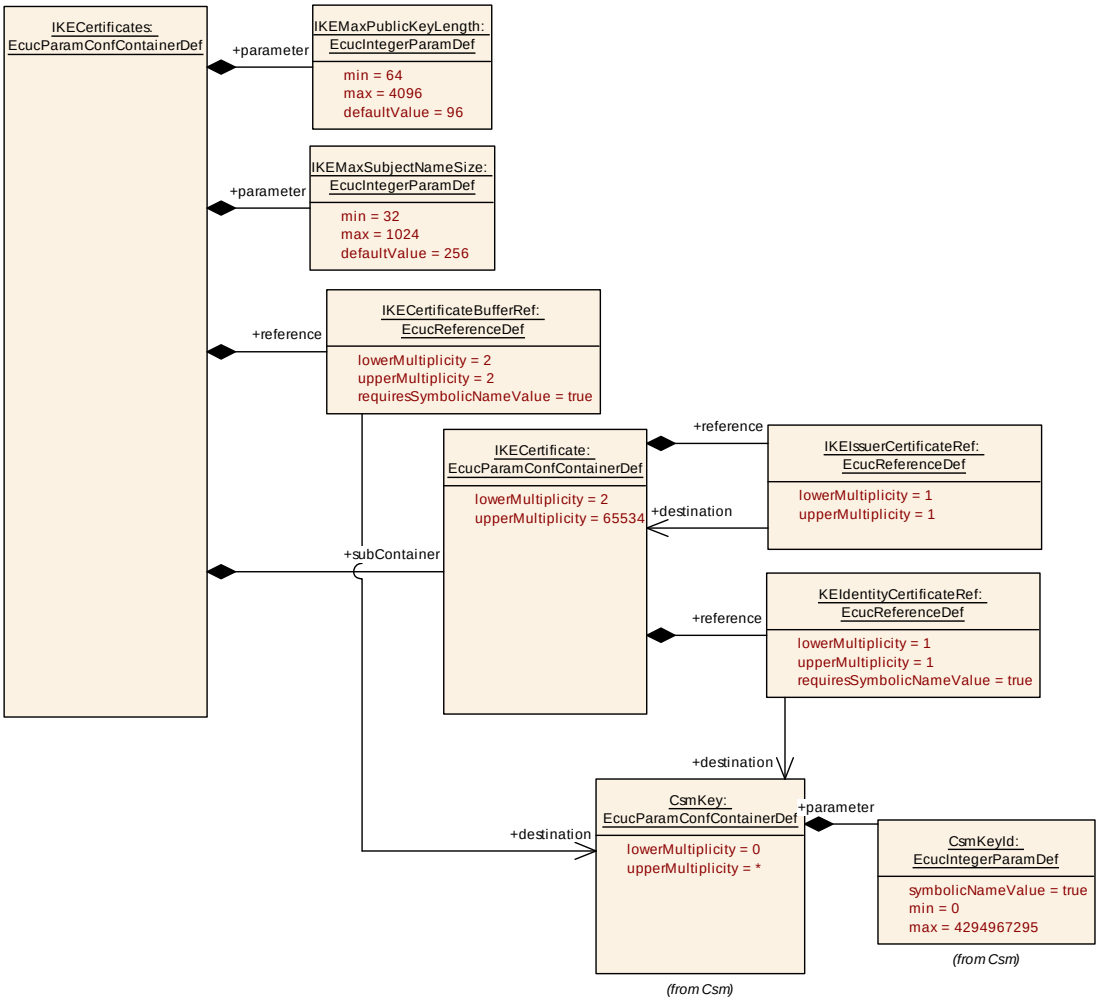

Figure 10.56: IKECertificates

10.2.76 IKECertificates

SWS Item	[ECUC_IKE_00005]
Container Name	IKECertificates
Parent Container	IKE
Description	Container for configuration of IKE certificates.
Configuration Parameters	

SWS Item	[ECUC_IKE_00042]
Parameter Name	IKEMaxPublicKeyLength
Parent Container	IKECertificates
Description	The maximum length of the public key in a certificate. Choose 64 for ECDSA-256, 96 for ECDSA-384, etc.
Multiplicity	1
Type	EcucIntegerParamDef

△

Range	64 .. 4096		
Default value	96		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00041]**		
Parameter Name	IKEMaxSubjectNameSize		
Parent Container	IKECertificates		
Description	The maximum size of the Subject Name field in certificates.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	32 .. 1024		
Default value	256		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00043]**		
Parameter Name	IKECertificateBufferRef		
Parent Container	IKECertificates		
Description	The referenced keys are used as buffers for temporarily storing the peer certificates.		
Multiplicity	2		
Type	Symbolic name reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
IKECertificate	2..65534	Container for configuration of an identity certificate and its issuer certificate. Use this container to configure a valid chain of certificates. The top-level certificate must be a self-signed certificate.

10.2.77 IKECertificate

SWS Item	**[ECUC_IKE_00044]**		
Container Name	IKECertificate		
Parent Container	IKECertificates		
Description	Container for configuration of an identity certificate and its issuer certificate. Use this container to configure a valid chain of certificates. The top-level certificate must be a self-signed certificate.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	**[ECUC_IKE_00045]**		
Parameter Name	IKEIssuerCertificateRef		
Parent Container	IKECertificate		
Description	The referenced certificate is the Issuer Certificate. The Issuer Certificate is used to identifiy the certificate authority (CA) which is the issuer of the Identity Certificate. The associated public key is used for verification of the certificate.		
Multiplicity	1		
Type	Reference to IKECertificate		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_IKE_00046]**
Parameter Name	KEIdentityCertificateRef
Parent Container	IKECertificate
Description	The referenced key is the Identity Certificate which is used to identifiy an entity and to associate that identiy with a public key.
Multiplicity	1
Type	Symbolic name reference to CsmKey
Post-Build Variant Multiplicity	false
Post-Build Variant Value	false

▽

△

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.3 Published Information

For details refer to the chapter 10.3 “Published Information” in SWS_BSWGeneral.