Mauve Corp

Fictional but secure products

Common Criteria Certification
BSI-DSZ-CC-xyz BSI-CC-PP-00zz

TOE Design Specification

MAUVECORP MAUVEVPN CLIENT
Version 2.11

MauveCorp
Fliederweg 98
D-50020 Köln
certification@mauvecorp.com

Document Version 1.0-SNAPSHOT
2022-09-13
[Commit cc54422 / main]

Contents

List of Tables

List of Figures

1. Introduction

This documente contains the information necessary for the evaluation of the assurance component ADV_TDS.3 in the Common Criteria evaluation of MauveCorp MauveVPN Client.

This document describes the TOE design on tho levels of abstraction: subsystems and modules, as required by the assurance level ADV_TDS.3 determined in the Protection Profile. The subsystems are described on a higher level of abstraction, as outlined in section "A.4.2 Modules" of Part III of the Common Criteria [CC Part 3]. The modules' descriptions are correspondingly morte elaborate.

Remarks on Notation

Table 1.1 lists the typographical conventions used in this document and their usage. Often the categories are not as clear-cut as they seem. Occasionally, the same term is used on different levels of abstraction. It is not always possible to determine that level at first glance. For example, a term can be both a keyword taken from the specification and the name of a variable in the code. Typographic consistency supports the reader in determining the level of abstraction. The explanations in Table 1.1 shall motivate the differentiation.

Subsystems and modules are separated by a double colon, as in Subsystem::Module. Interfaces of modules are separated by two forward slashes: Subsystem::Module//Interface.

The quoted names of code elements – especially of the Java-based components – tend to be quite long. In such cases, hyphens are used to split the name across lines, . This avoids extraneous amounts of white space and supports readability.

Typographic Markup	Purpose
keywords	*Keywords* are terms that are directly taken from the specification. This can be the names of configuration parametes and their values. Also other highly specific terms can be typeset as *keywords*.
`code elements`	`code elements` are terms that are taken from the TOE's source code. This can be names of classes, methods and other types, but also their parameters or logical structures of the programming language employed by the TOE.
`file names`	*`file names`* relate to names of file system elements, such as files or directories.
security terms	Terms that are directly related to the Common Criteria framework are typeset as security terms. This can be SFRs or the names of SF and TSFI. Furthermore, the names of subsystems and modules that constitute the TOE are typeset in this way. This holds also for the names of certficiates and other key material.

Table 1.1.: Typographic Conventions

2. Description of Subsystems

This chapter describes the division of the TOE into its subsystems.

2.1. Subsystem VPN Client

Description

The subsystem VPN Client provides the VPN client that is used to create VPN tunnels in the WAN.

Implemented SFR

This subsystem fulfills the requirements levied upon the TOE by the SFR in Table 2.1, Table 2.2.

Interaction With Other Subsystems

The subsystem VPN Client requires functionalities from the subsystems Crypto Services.

Enforcing SFR	Purpose
FCS_CKM.2/IKE	Connection establishment, key distribution for VPN
FPT_TDC.1/Zert	Certificate validation
FTP_ITC.1/VPN	Establish/terminate VPN connections

Table 2.1.: Enforcing SFR of subsystem VPN Client

Supporting SFR	Purpose
FTP_TRP.1/Admin	Establish/terminate VPN connections

Table 2.2.: Supporting SFR of subsystem VPN Client

2.2. Subsystem NTP Client

Description

The subsystem NTP Client provides the NTP client that is used to synchronize the system time with an NTP server in the WAN.

Implemented SFR

This subsystem fulfills the requirements levied upon the TOE by the SFR in Table 2.3, Table 2.4.

Interaction With Other Subsystems

Enforcing SFR	Purpose
FPT_STM.1	Tiem service

Table 2.3.: Enforcing SFR of subsystem NTP Client

Supporting SFR	Purpose
(none)	

Table 2.4.: Supporting SFR of subsystem NTP Client

2.3. Subsystem Protection

Description

The subsystem Protection contains the functionalities that are used for the self-protection of the TOE. Provided are capabilities to diagnose the TOE with self-tests (Self Test and for deleting residual information from the TOE's memory (Memory Protection).

Implemented SFR

This subsystem fulfills the requirements levied upon the TOE by the SFR in Table 2.5, Table 2.6.

Interaction With Other Subsystems

Enforcing SFR	Purpose
FDP_RIP.1	Memory cleanup
FPT_TST.1	Run self test

Table 2.5.: Enforcing SFR of subsystem Protection

Supporting SFR	Purpose
FPT_TST.1	Run self test

Table 2.6.: Supporting SFR of subsystem Protection

2.4. Subsystem Administration

Description

The subsystem Administration provides the administration application and management functions of the TOE.

Implemented SFR

This subsystem fulfills the requirements levied upon the TOE by the SFR in Table 2.7, Table 2.8.

Interaction With Other Subsystems

This subsystem requires functionalities from the subsystem Protection.

Enforcing SFR	Purpose
FTP_TRP.1/Admin	Provide management application

Table 2.7.: Enforcing SFR of subsystem Administration

Supporting SFR	Purpose
FTP_ITC.1/VPN	Manage VPN connections

Table 2.8.: Supporting SFR of subsystem Administration

2.5. Subsystem Crypto Services

Description

The subsystem Crypto Services provides cryptographic base functions required by other functionalities of the TOE. The base functions comprise cryptographic algorithms, key management and random number generation.

Implemented SFR

This subsystem fulfills the requirements levied upon the TOE by the SFR in Table 2.9, Table 2.10.

Interaction With Other Subsystems

Enforcing SFR	Purpose
FCS_CKM.1	Key creation for VPN and TLS
FCS_CKM.4	Key destruction for VPN and TLS
FCS_COP.1/Hash	Provide Hash-Algorithms
FCS_COP.1/HMAC	Provide HMAC-Algorithms
FCS_RNG.1/Hash_DRBG	Random number creation

Table 2.9.: Enforcing SFR of subsystem Crypto Services

Supporting SFR	Purpose
(none)	

Table 2.10.: Supporting SFR of subsystem Crypto Services

2.6. Subsystem TLS-Server

Description

The subsystem Crypto Services provides TLS connections required to connect to the management functions in subsystem Administration.

Implemented SFR

This subsystem fulfills the requirements levied upon the TOE by the SFR in Table 2.11, Table 2.12.

Interaction With Other Subsystems

Enforcing SFR	Purpose
FCS_CKM.2/TLS	Connection establishment, key distribution for TLS
FCS_COP.1/TLS.AES	Provide AES for TLS
FCS_COP.1/TLS.Auth	Authentication of TLS peer
FPT_TDC.1/TLS.Zert	Certificate validation
FTP_ITC.1/TLS	Establish/terminate TLS connections

Table 2.11.: Enforcing SFR of subsystem TLS-Server

Supporting SFR	Purpose
(none)	

Table 2.12.: Supporting SFR of subsystem TLS-Server

3. Description of Modules

This chapter describes the modules of the TOE, sorted by subsystem.

3.1. Modules for Subsystem VPN Client

This section describes the modules of subsystem VPN Client.

3.1.1. Module VPN Client::Core

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.1. The module is SFR-enforcing.

Enforcing SFR	
FCS_CKM.2/IKE	FTP_ITC.1/VPN
Supporting SFR	
FTP_TRP.1/Admin	

Table 3.1.: SFR of module VPN Client::Core

3.1.1.1. Description

The module Core of subsystem VPN Client provides interfaces and processes to…

3.1.1.2. Processes

3.1.1.2.1. Open Connection to VPN concentrator This process opens a connection to the VPN concentrator using IPSec and IKEv2. During the connection process, the peer presents its identity in the form of an X.509 certificate which must be verified. It is verified using the functionality reached by the interface VPN Client::Certificate Service//Check-VPN-Certificate.

Implemented SFR	
FTP_ITC.1/VPN	FCS_CKM.2/IKE

3.1.1.2.2. Close Connection to VPN concentrator This process opens a connection to the VPN concentrator.

Implemented SFR
FTP_ITC.1/VPN

3.1.1.3. Interfaces To Other Modules

3.1.1.3.1. Connect-to-VPN (Provided) This interface triggers the creation of a new VPN connection (see Section 3.1.1.2.1).

3.1.1.3.2. Disconnect-from-VPN (Provided) This interface closes the VPN connection (see Section 3.1.1.2.2).

3.1.1.3.3. Check-VPN-Certificate (Required) The interface VPN Client::Certificate Service//Check-VPN-Certificate is required for checking validity of the certificate of the VPN concentrator.

3.1.1.3.4. Create-DHE-Key (Required) The interface Crypto Services::Key Management//Create-DHE-Key is required for creating an ephemeral key pair for the connection.

3.1.1.3.5. Destroy-Key (Required) The interface Crypto Services::Key Management//Destroy-Key is required for destroying the connection keys.

3.1.2. Module VPN Client::Certificate Service

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.2. The module is SFR-enforcing.

Enforcing SFR
FPT_TDC.1/Zert
Supporting SFR
(none)

Table 3.2.: SFR of module VPN Client::Certificate Service

3.1.2.1. Description

The module Certificate Service of subsystem VPN Client provides interfaces and processes to…

3.1.2.2. Processes

3.1.2.2.1. Verification of the VPN concentrator certificate The certificate is checked mathematically and for validity. The expiry date must be at least one day in the future. The SHA-256 hash of the certificate is calculated by calling the function Crypto Services::Algorithms//Get-Hash.

Implemented SFR
FPT_TDC.1/Zert

3.1.2.3. Interfaces To Other Modules

3.1.2.3.1. Check-VPN-Certificate (Provided) This interface is called to check the certificate of a VPN concentrator (see Section 3.1.2.2.1).

3.1.2.3.2. Get-Hash (Required) The interface Crypto Services::Algorithms//Get-Hash is required to calculate the hash value of the certificate.

3.2. Modules for Subsystem NTP Client

This section describes the modules of subsystem NTP Client.

3.2.1. Module NTP Client::Core

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.3. The module is SFR-enforcing.

Enforcing SFR
FPT_STM.1
Supporting SFR
(none)

Table 3.3.: SFR of module NTP Client::Core

3.2.1.1. Description

The module Core of subsystem NTP Client provides interfaces and processes for using a time service. The module implements the logical interface LI.WAN.NTP of the Functional Specification [ADV_FSP, Section 3.2.6].

3.2.1.2. Processes

3.2.1.2.1. Synchronizing System Clock This process synchronizes the system time by connecting to a remote NTP server via LI.WAN.NTP.

Implemented SFR
FPT_STM.1

3.2.1.3. Interfaces To Other Modules

3.2.1.3.1. Sync-Time (Provided) This interface can be called by other modules of the TOE to initiate synchronization of the system clock (see Section 3.2.1.2.1).

3.3. Modules for Subsystem Protection

This section describes the modules of subsystem Protection.

3.3.1. Module Protection::Memory Protection

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.4. The module is SFR-enforcing.

Enforcing SFR
FDP_RIP.1
Supporting SFR
(none)

Table 3.4.: SFR of module Protection::Memory Protection

3.3.1.1. Description

The module Memory Protection of subsystem Protection provides mechanisms for sanitizing system memory.

3.3.1.2. Processes

3.3.1.2.1. Overwriting Sensitive Data Sensitive data is overwritten by pseudo-random numbers generated by the TOE's random number generator in Crypto Services::Random Number Generator. The RNG is called via Crypto Services::Random Number Generator//Create-Random.

Implemented SFR
FDP_RIP.1

3.3.1.3. Interfaces To Other Modules

3.3.1.3.1. Clean-Memory (Provided) This interface triggers the memory sanitization (see Section 3.3.1.2.1).

3.3.2. Module Protection::Self Test

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.5. The module is SFR-enforcing.

Enforcing SFR
FPT_TST.1
Supporting SFR
FPT_TST.1

Table 3.5.: SFR of module Protection::Self Test

3.3.2.1. Description

The module Self Test of subsystem Protection provides functionality to run self-tests of the TSF.

3.3.2.2. Processes

3.3.2.2.1. Run Self-Tests This process runs a self test of the TSF. The process is run at boot-up and periodically every 24 hours. The process can also be called by an administrator.

Implemented SFR
FPT_TST.1

3.3.2.3. Interfaces To Other Modules

3.3.2.3.1. Run-Selftest (Provided) This interface calls the self test (see Section 3.3.2.2.1).

3.4. Modules for Subsystem Administration

This section describes the modules of subsystem Administration.

3.4.1. Module Administration::HTTP-Server

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.6. The module is SFR-enforcing.

Enforcing SFR
FTP_TRP.1/Admin
Supporting SFR
(none)

Table 3.6.: SFR of module Administration::HTTP-Server

3.4.1.1. Description

The module HTTP-Server of subsystem Administration provides an HTTP Server for the management interface.

3.4.1.2. Processes

3.4.1.2.1. HTTP Communication With an Administrator This process implements the HTTP protocol.

Implemented SFR
FTP_TRP.1/Admin

3.4.1.3. Interfaces To Other Modules

3.4.1.3.1. HTTP-Endpoint (Provided) This interface implements the logical interface LI.LAN.HTTP_MGMT.

3.4.1.3.2. Set-Configuration (Required) The interface Administration::Management-App//Set-Configuration is required for saving configuration parameters.

3.4.1.3.3. Connect-Disconnect-VPN (Required) The interface Administration::Management-App//Connect-Disconnect-VPN is required by the HTTP server for opening/closing connections to the VPN concentrator.

3.4.2. Module Administration::Management-App

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.7. The module is SFR-supporting.

Enforcing SFR
(none)
Supporting SFR
FTP_ITC.1/VPN

Table 3.7.: SFR of module Administration::Management-App

3.4.2.1. Description

The module Management-App of subsystem Administration provides functionality to configure the TOE. Furthermore, the administrator can use the module to open/close a connection to the VPN concentrator and to run the TSF self tests.

3.4.2.2. Processes

3.4.2.2.1. Speichern der Konfiguration This process persists configuration parameters entered by the administrator via the HTTP server.

3.4.2.2.2. Opening and Closing the VPN Connection This process initiates the creation and destruction of a VPN connection. The interface VPN Client::Core//Connect-to-VPN is required for creating the connection. The interface VPN Client::Core//Disconnect-from-VPN is required for closing the connection.

3.4.2.2.3. Calling Self Tests This process initiates the self tests of module Protection::Self Test. The self tests are started by calling Protection::Self Test//Run-Selftest.

3.4.2.3. Interfaces To Other Modules

3.4.2.3.1. Set-Configuration (Provided) This interface calls the processes for managing the configuration data of the TOE. Configuration data can be read and written with this interface.

3.4.2.3.2. Connect-Disconnect-VPN (Provided) This interface is used to initiate the opening and closing of a VPN connection from the management interface.

3.4.2.3.3. Connect-to-VPN (Required) The interface VPN Client::Core//Connect-to-VPN is required for opening the VPN connection.

3.4.2.3.4. Disconnect-from-VPN (Required) The interface VPN Client::Core//Disconnect-from-VPNis required for closing the VPN connection.

3.4.2.3.5. Run-Selftest (Required) The interface is required Protection::Self Test//Run-Selftest for running the self tests.

3.5. Modules for Subsystem Crypto Services

This section describes the modules of subsystem Crypto Services.

3.5.1. Module Crypto Services::Algorithms

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.8. The module is SFR-enforcing.

Enforcing SFR	
FCS_COP.1/Hash	FCS_COP.1/HMAC
Supporting SFR	
(none)	

Table 3.8.: SFR of module Crypto Services::Algorithms

3.5.1.1. Description

Module Algorithms of subsystem Crypto Services provides cryptographic base functionalities.

3.5.1.2. Processes

3.5.1.2.1. Calculate Hash Values This process calculates SHA-2 hash values.

Implemented SFR
FCS_COP.1/Hash

3.5.1.2.2. Calculate HMAC This process calculates HMAC.

Implemented SFR
FCS_COP.1/HMAC

3.5.1.3. Interfaces To Other Modules

3.5.1.3.1. Get-Hash (Provided) This interface triggers the hash value calculation (see Section 3.5.1.2.1)

3.5.1.3.2. Get-HMAC (Provided) This interface triggers the HMAC calculation (see Section 3.5.1.2.2)

3.5.2. Module Crypto Services::Key Management

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.9. The module is SFR-enforcing.

Enforcing SFR	
FCS_CKM.1	FCS_CKM.4
Supporting SFR	
(none)	

Table 3.9.: SFR of module Crypto Services::Key Management

3.5.2.1. Description

The module Key Management of subsystem Crypto Services provides functionality to manage cryptographic keys.

3.5.2.2. Processes

3.5.2.2.1. Creating Keys Ephemeral Diffie-Hellman key pairs are created by reading random numbers from Crypto Services::Random Number Generator. The random data is wrapped into the appropriate data structures.

Implemented SFR
FCS_CKM.1

3.5.2.2.2. Destroying keys Keys are destroyed by overwriting the data with pseudo-random numbers. Module Protection::Memory Protection is used for securely erasing the memory.

Implemented SFR
FCS_CKM.4

3.5.2.3. Interfaces To Other Modules

3.5.2.3.1. Create-DHE-Key (Provided) This interface is used to create a key pair (see Section 3.5.2.2.1).

3.5.2.3.2. Destroy-Key (Provided) This interface is used to destroy a key (see Section 3.5.2.2.2).

3.5.2.3.3. Create-Random (Required) The interface Protection::Memory Protection//Clean-Memory is required for generating random numbers (see Section 3.5.2.2.1)

3.5.2.3.4. Clean-Memory (Required) The interface Protection::Memory Protection//Clean-Memory is required for securely erasing the memory during key deletion (see Section 3.5.2.2.2).

3.5.3. Module Crypto Services::Random Number Generator

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.10. The module is SFR-enforcing.

Enforcing SFR
FCS_RNG.1/Hash_DRBG
Supporting SFR
(none)

Table 3.10.: SFR of module Crypto Services::Random Number Generator

3.5.3.1. Description

The module Random Number Generator of subsystem Crypto Services provides a secure random number generator.

3.5.3.2. Processes

3.5.3.2.1. Creating a Random Number This process generates n random bytes.

Implemented SFR
FCS_RNG.1/Hash_DRBG

3.5.3.3. Interfaces To Other Modules

3.5.3.3.1. Create-Random (Provided) This interface is provided to create random numbers (see Section 3.5.3.2.1).

3.6. Modules for Subsystem TLS-Server

This section describes the modules of subsystem TLS-Server.

3.6.1. Module TLS-Server::Core

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.11. The module is SFR-enforcing.

Enforcing SFR	
FCS_CKM.2/TLS	FCS_COP.1/TLS.Auth
FCS_COP.1/TLS.AES	FTP_ITC.1/TLS
Supporting SFR	
(none)	

Table 3.11.: SFR of module TLS-Server::Core

3.6.1.1. Description

Module Core of subsystem TLS-Server implements the TLS protocol. A TLS server is started to use the administration interface of the TOE.

3.6.1.2. Processes

3.6.1.2.1. Opening a TLS Connection This process accepts an incoming TLS connection from a client. During TLS handshake, ephemeral DH key exchange is performed. The reqired key pair is generated by module Key Management.

Implemented SFR	
FTP_ITC.1/TLS	FCS_COP.1/TLS.AES

3.6.1.2.2. Authenticating the Peer This proccess authenticates the peer. The peer's certificate is verified by calling TLS-Server::Certificate Service//Check-TLS-Certificate.

Implemented SFR
FCS_COP.1/TLS.Auth

3.6.1.2.3. Closing a TLS Connection This process closes the TLS connection. Upon closing the connection, the TLS session keys are destroyed by calling Destroy-Key.

Implemented SFR
FTP_ITC.1/TLS

3.6.1.3. Interfaces To Other Modules

3.6.1.3.1. TLS-Connection-Accept (Provided) This interface accecpts a TLS connection. It implements the TSFI LI.LAN.TLS.

3.6.1.3.2. TLS-Disconnect (Provided) This interface closes a TLS connection.

3.6.1.3.3. Check-TLS-Certificate (Required) The interface TLS-Server::Certificate Service//Check-TLS-Certificate is required for verifying the peer's certificate.

3.6.1.3.4. Create-DHE-Key (Required) The interface Crypto Services::Key Management//Create-DHE-Key is required for creating the ephemeral DH key pair.

3.6.1.3.5. Destroy-Key (Required) The interface Crypto Services::Key Management//Destroy-Key is required for destroying the TLS session keys.

3.6.2. Module TLS-Server::Certificate Service

This module fulfills the requirements levied upon the TOE by the SFR in Table 3.12. The module is SFR-enforcing.

Enforcing SFR
FPT_TDC.1/TLS.Zert
Supporting SFR
(none)

Table 3.12.: SFR of module TLS-Server::Certificate Service

3.6.2.1. Description

Module Certificate Service of subsystem TLS-Server provides functionalities for verifying X.509 certificates during the TLS handshake.

3.6.2.2. Processes

3.6.2.2.1. Prüfung des Zertifikats This process checks the certificate presented by the peer during the TLS handshake.

Implemented SFR
FPT_TDC.1/TLS.Zert

3.6.2.3. Interfaces To Other Modules

3.6.2.3.1. Check-TLS-Certificate (Provided) This interface triggers the certificate verification.

3.6.2.3.2. Get-Hash (Required) The interface Crypto Services::Algorithms//Get-Hash is required for calculating the certificate hash value.

A. TLS Connections

The Protection Profiles defines the cipher suites required for TLS connections. The TOE uses provides exactly those cipher suites and no other. Table A.1 lists these cipher suites. Table A.2 lists the elliptic curves used for ECDHE.

Algorithmen / Cipher Suite	IANA ID	TLS 1.2 [RFC 5246]
TLS_DHE_RSA_WITH_AES_128_CBC_SHA	0x00, 0x33	✓
TLS_DHE_RSA_WITH_AES_256_CBC_SHA	0x00, 0x39	✓
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	0xc0, 0x13	✓
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA	0xc0, 0x14	✓
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	0xc0, 0x27	✓
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	0xc0, 0x28	✓
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	0xc0, 0x2f	✓
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	0xc0, 0x30	✓

Table A.1.: Cipher suites for TLS connections

Elliptic curve	IANA ID	Standard
secp256r1 (P-256)	23	[RFC 8422; ANSI X9.62]
secp384r1 (P-384)	24	[RFC 8422; ANSI X9.62]
brainpoolP256r1	26	[RFC 7027]
brainpoolP384r1	27	[RFC 7027]

Table A.2.: Elliptic curves for TLS connections

The TOE communicates with other trusted IT products over secure connections. Integrity and confidentiality of these connections is ensured by TLS v1.2. Table A.4 lists all connections the TOE can participate in. Table A.3 describes the columns of this table.

Column	Descriptions
ID	Smybolic name of this connection
Interface	Logical interface whose communication is secured.
Role	Describes client/serer role of the TOE.
Peer	Describes the peer in this TLS connection.
Protocol	Application protocol used in this connection.
Subsystem::Module	Name of the subsystem and the module from which the connection originates or that accepts the connections.
Port	IP-Port that the TOE opens for the connection. If the TOE is client, “dyn.” stands for ephemeral port assignment. “config” stands for a confgurable port number.
Identity of TOE	Certificate that the TOE uses to authenticate itself to the peer.
Identity of Peer	Certificate that the peer uses to authenticate itself to the TOE.
Authentication by	Process, data source or subsystem/module used by the TOE for authentication/verification.

Table A.3.: Legend for TLS connections

ID	Interface (protocol)	Role	Peer	Subsystem::Module	Port	Identity of TOE	Identity of Peer	Authentication by
TLS.1	LI.LAN.HTTP_MGMT	Server	Browser	Administration::HTTP-Server	443	Certificate from Mauve CA	Username/password	User administration in TOE

Table A.4.: TLS connections of MauveVPN Client

B. List of TSFI

The TOE provides the logical interfaces described in the protection profile [BSI-CC-PP-00zz]. They are repeated here.

LS.LAN is the TOE's interface to the local area network of the operating environment. In addition to the interfaces described in the protection profile, there are further protocol specific interfaces described here. Table B.1 lists these logical interfaces.

LS.WAN is the TOE's interface to the wide area network of the operating environment, the Internet. In addition to the interfaces described in the protection profile, there are further protocol specific interfaces described here. Table B.2 lists these logical interfaces.

LS.LED represents the logical interface to the display and the buttons of PS.LED.

Label	Client/Server	Purpose of the interface
LI.LAN.Ether	—	media access
LI.LAN.IP	—	access to the Internet layer
LI.LAN.TCP	—	access to the transport layer
LI.LAN.TLS	server	transport security with TLS 1.2
LI.LAN.UDP	—	access to the transport layer
LI.LAN.HTTP_MGMT	server	HTTP access to the management console

Table B.1.: Logical Interfaces on LI.LAN

Label	Client/Server	Purpose of the interface
LI.WAN.Ether	—	media access
LI.WAN.IP	—	access to the Internet layer
LI.WAN.TCP	—	access to the transport layer
LI.WAN.NTP	client	Obtaining time
LI.WAN.DHCP	client	Obtaining IP addresses in the WAN
LI.WAN.UDP	—	access to the transport layer
LI.WAN.IPSec	—	VPN data traffic

Table B.2.: Logical Interfaces on LI.WAN

C. Mapping of Modules to SFR

This Table shows the coverage of SFR by their *enforcing* and *supporting* modules

SFR	Relation	subsystem::module
FCS_CKM.1	Enforcing Supporting	Crypto Services::Key Management (none)
FCS_CKM.2/IKE	Enforcing Supporting	VPN Client::Core (none)
FCS_CKM.2/TLS	Enforcing Supporting	TLS-Server::Core (none)
FCS_CKM.4	Enforcing Supporting	Crypto Services::Key Management (none)
FCS_COP.1/Hash	Enforcing Supporting	Crypto Services::Algorithms (none)
FCS_COP.1/HMAC	Enforcing Supporting	Crypto Services::Algorithms (none)
FCS_COP.1/TLS.AES	Enforcing Supporting	TLS-Server::Core (none)
FCS_COP.1/TLS.Auth	Enforcing Supporting	TLS-Server::Core (none)
FCS_RNG.1/Hash_DRBG	Enforcing Supporting	Crypto Services::Random Number Generator (none)
FDP_RIP.1	Enforcing Supporting	Protection::Memory Protection (none)
FPT_TDC.1/TLS.Zert	Enforcing Supporting	TLS-Server::Certificate Service (none)
FPT_TDC.1/Zert	Enforcing Supporting	VPN Client::Certificate Service (none)
FPT_STM.1	Enforcing Supporting	NTP Client::Core (none)
FPT_TST.1	Enforcing Supporting	Protection::Self Test Protection::Self Test

Continued on next page

SFR	Relation	subsystem::module
FTP_ITC.1/TLS	Enforcing Supporting	TLS-Server::Core (none)
FTP_ITC.1/VPN	Enforcing Supporting	VPN Client::Core Administration::Management-App
FTP_TRP.1/Admin	Enforcing Supporting	Administration::HTTP-Server VPN Client::Core

Table C.1.: Mapping of SFR to modules

Bibliography

Protection Profiles And Technical Guidelines

[BSI-CC-PP-00zz] Bundesamt für Sicherheit in der Informationstechnik. *Schutzprofil: Anforderungen an den VPN Client. BSI-CC-PP-00zz*. Common Criteria Schutzprofil (Protection Profile). Version 1.1. Bundesamt für Sicherheit in der Informationstechnik (BSI), Feb. 5, 2020.

Developer Documentation

[ADV_FSP] Mauve Corp. *MauveCorp MauveVPN Client. Functiona Specification*. Common Criteria Component ADV_FSP.

Standards

[ANSI X9.62] Accredited Standards Committee X9. *ANSI X9.62, Public Key Cryptography For The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*. Standard. ANSI, Nov. 16, 2005.

[CC Part 3] The Common Criteria Recognition Agreement Members. *Common Criteria for Information Technology Security Evaluation. Part 3: Security assurance components*. Common Criteria. Version 3.1R5. Common Criteria Portal, Apr. 2017. URL: http://www.commoncriteriaportal.org/thecc.html.

RFC

[RFC 5246] T. Dierks and E. Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.2*. RFC 5246 (Proposed Standard). RFC. Updated by RFCs 5746, 5878, 6176, 7465, 7507, 7568, 7627, 7685, 7905, 7919. Fremont, CA, USA: RFC Editor, Aug. 2008. DOI: 10.17487/RFC5246. URL: https://www.rfc-editor.org/rfc/rfc5246.txt.

[RFC 7027] J. Merkle and M. Lochter. *Elliptic Curve Cryptography (ECC) Brainpool Curves for Transport Layer Security (TLS)*. RFC 7027 (Informational). RFC. Fremont, CA, USA: RFC Editor, Oct. 2013. DOI: 10.17487/RFC7027. URL: https://www.rfc-editor.org/rfc/rfc7027.txt.

[RFC 8422] Y. Nir, S. Josefsson, and M. Pegourie-Gonnard. *Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier*. RFC 8422 (Proposed Standard). RFC. Fremont, CA, USA: RFC Editor, Aug. 2018. DOI: 10.17487/RFC8422. URL: https://www.rfc-editor.org/rfc/rfc8422.txt.

Index of SFR