Security Target for VMware ESXi 8.0g

Author:	Broadcom, Inc.
Version:	2.0
Date:	February 24, 2026
Cert-ID:	BSI-DSZ-CC-1087
Company:	Broadcom, Inc.

vmware by Broadcom

3401 Hillview Ave
Palo Alto, CA 94304
United States of America

https://www.vmware.com

Table of Contents

Page

1 INTRODUCTION 7
1.1 ST Reference 7
1.2 TOE Reference 7
1.3 Product Overview 7
1.4 TOE Overview 8
1.5 TOE Description 10
1.5.1 Logical Boundary 10
1.5.2 Hardware, Firmware, and Software Supplied by the TOE Environment 15
1.6 TOE Delivery 15
2 CONFORMANCE CLAIMS 16
2.1 CC Conformance Claims 16
2.2 PP Claim 16
2.3 Package Claim 16
2.4 Conformance Claim Rationale 16
3 SECURITY PROBLEM DEFINITION 17
3.1 External Entities 17
3.2 Assets 17
3.3 Assumptions 17
3.4 Threats 18
3.5 Organizational Security Policies (OSPs) 19
4 SECURITY OBJECTIVES 20
4.1 Security Objectives for the TOE 20
4.2 Security Objectives for the Environment 22
4.3 Security Objectives Rationale 22
5 EXTENDED COMPONENT DEFINITION 25
5.1 VMM isolation from VMs (FPT_VIV_EXT) 25
5.1.1 Family Behaviour 25
5.1.2 Component Levelling 25
5.1.3 Management 25
5.1.4 Audit 25
5.1.5 VMM isolation from VMs (FPT_VIV_EXT.1) 25
5.2 Hypercall Controls (FPT_HCL_EXT) 25
5.2.1 Family Behaviour 25
5.2.2 Component Levelling 26
5.2.3 Management 26
5.2.4 Audit 26
5.2.5 Hypercall Controls (FPT_HCL_EXT.1) 26
5.3 Removable Devices and Media (FPT_RDM_EXT) 26
5.3.1 Family Behaviour 26
5.3.2 Component Levelling 26
5.3.3 Management 26
5.3.4 Audit 26

5.3.5 Removable Devices and Media (FPT_RDM_EXT.1) 26
5.4 Generation of Random Numbers 27
5.4.1 Family Behaviour 27
5.4.2 Component Levelling 27
5.4.3 Management 27
5.4.4 Audit 27
5.4.5 Random number generation (FCS_RNG.1) 27
6 SECURITY REQUIREMENTS 27
6.1 Security Functional Requirements 28
6.1.1 Class FAU: Security Audit 29
6.1.2 Class FCS: Cryptographic Support 31
6.1.3 Class FDP: User Data Protection 33
6.1.4 Class FIA: Identification and Authentication (FIA) 36
6.1.5 Class FMT: Security Management (FMT) 37
6.1.6 Class FPT: Protection of the TSF (FPT) 38
6.1.7 Class FTP: Inter-TSF trusted channel (FTP_ITC) 39
6.2 Security Assurance Requirements 40
6.3 Security Requirements Rationale 41
6.3.1 Rationale for the Security Functional Requirements (SFRs) 41
6.4 Rationale for the Security Assurance Requirements (SARs) 46
6.4.1 Fulfilment of the Dependencies 46
7 TOE SUMMARY SPECIFICATION 47
7.1 SF1.Security Audit 47
7.2 SF2.Random Number Generation 48
7.3 SF3.User Data Protection 48
7.4 SF4.Identification and Authentication 50
7.5 SF5.Security Management 51
7.6 SF6.Protection of the TSF 53
7.7 SF7.Cryptographic Support and Encrypted Channels 56
7.8 TOE Summary Specification Rationale 60
8 ACRONYMS & DEFINITIONS 62

List of Tables

Table 1: External Entities 17
Table 2: Assets 17
Table 3: Assumptions 18
Table 4: Threats 19
Table 5: Security Objectives for the TOE 22
Table 6: Security Objectives for the Environment 22
Table 7: Rationale for Security Objectives 24
Table 8: SFRs 29
Table 9: Minimal Auditable Events 29
Table 10: Additional Auditable Events 30
Table 11: Cryptographic Key Generation 31
Table 12: Cryptographic Key Exchange 31
Table 13: Cryptographic Operations – RSA 32
Table 14: Information Flow Policy 34
Table 15: Remote Management Functions 38
Table 16: Management of security functions behaviour 38
Table 17: Management of security attributes 38
Table 18: Fulfilment of Security Objectives 44
Table 19: Fulfilment of SFR Dependencies 46
Table 20: Security Audit SFRs 47
Table 21: Cryptographic Support SFRs 48
Table 22: Guest Separation Rationale 49
Table 23: Identification and Authentication Rationale 50
Table 24: Security Management Rationale 52
Table 25: Protection of the TSF SFRs 54
Table 26: Encrypted Channels Rationale 59
Table 27: Algorithms / Ciphersuite 59
Table 28: Cryptographic Functions 60
Table 29: TSS Rationale 62
Table 30: List of Acronyms 63

List of Figures

Figure 1: Product Overview 8
Figure 2: TOE Boundary 12

1 Introduction

This section identifies the Security Target (ST), Target of Evaluation (TOE), document conventions, and terminology. It also provides TOE overview and describes the hardware and software that make up the TOE as well as the physical and logical boundaries of the TOE.

1.1 ST Reference

Document Title:	Security Target for the VMware ESXi 8.0g
Document Version:	2.0
Document Date:	2026-02-24

1.2 TOE Reference

TOE Name:	VMware ESXi
TOE Version:	8.0g
TOE Build Number:	25191479
Developer:	VMware, Inc.
Product Type:	Hypervisor
Certification-ID:	BSI-DSZ-CC-1087

1.3 Product Overview

The TOE is a VMware ESXi 8.0g hypervisor that provides an environment to host multiple virtual machines on industry standard x86-compatible hardware platforms (64-bit) and provides the management of resources and virtual machines. Figure 1 shows a sample deployment configuration of the TOE:

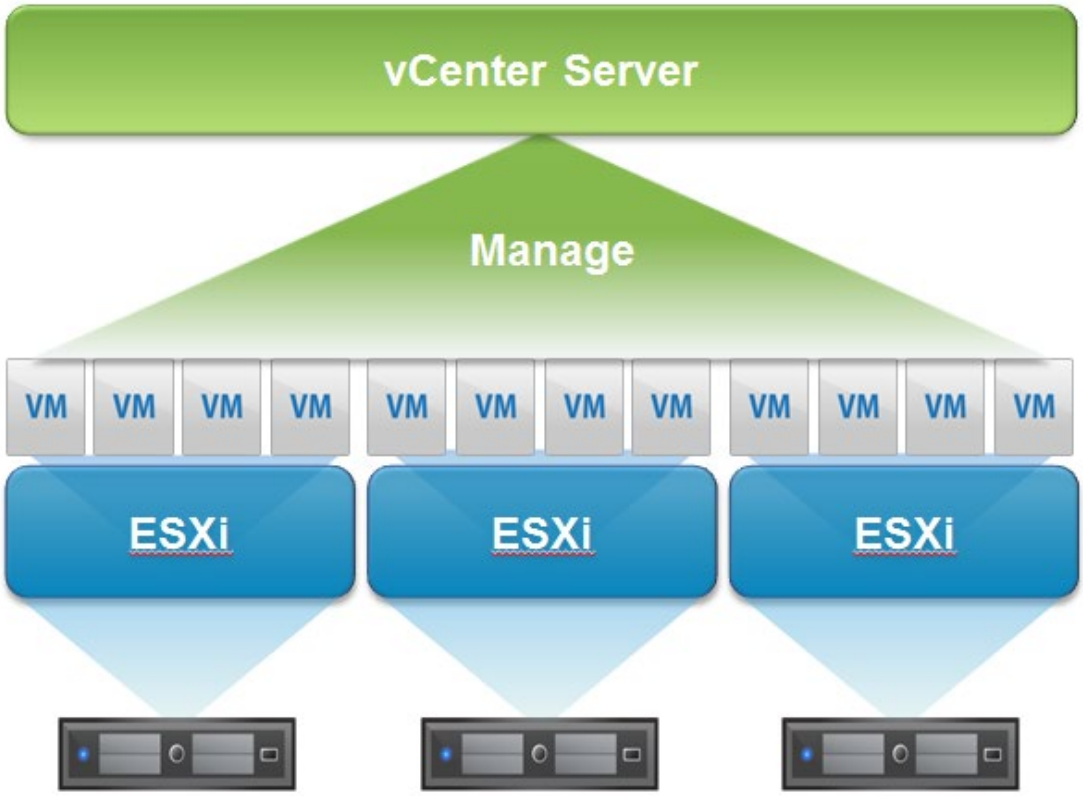


Figure 1: Product Overview

1.4 TOE Overview

The Target of Evaluation (TOE) is VMware ESXi 8.0g, a VMware software product implementing a hypervisor. The TOE is designed as a virtualization platform, providing the ability to implement and virtualize different workloads across multiple virtual machines (VMs) while providing isolation and efficient use of compute, storage, and network resources. This allows for implementation of cloud environments in support of wide array of various data center workloads or on-demand infrastructure.

The TOE provides functionality to enforce and support auditing, cryptographic operations, guest separation, encrypted channels, identification/authentication, security management, and protection of the TSF. The administrator role can provision VMs, configure virtual network and storage resources, and enforce data protection for virtual machines across virtual networks. Users can only interact directly with those VMs via the network; there is no non-administrator role access to the hypervisor's management functions.

One of VMware, Inc.’s core businesses is virtualization software. Specifically, VMware offers its virtualization solution which runs on supported industry standard x86_64-compatible hardware platforms. The foundational concept of virtualization technology is that a single physical hardware system is used to host multiple logical or “virtual” machines (VMs). VMware's architecture uses a hypervisor composed of a virtual machine monitor to provide isolation, a custom kernel (VMKernel) designed for the purpose of running virtual machine, and incidental management and configuration functionality for the hypervisor. The administrator role can further install guest operating systems within these virtual machines. The hypervisor has no provision for running general applications within the hypervisor.

In VMware’s virtualization solution, the following components are the essential building blocks that make up the virtualized computing environment:

- A host machine – x86_64-compatible hardware (part of the TOE's operating environment)
- Hypervisor (ESXi) (The TOE) – Enterprise class virtualization software from VMware that is installed on the host computer. The ESXi software provides the interface to run virtual machines on the host computer, and exposes two sets of APIs to manage virtual machines and the host computer.
- The virtual machines themselves, on the host machine (not part of the TOE)
- The guest operating system and application software that is installed on the virtual machine (not part of the TOE).

The four components described above make a very basic virtualized computing environment. That is, a single ESXi host provides the environment for one or more virtual machines. A more sophisticated deployment typical in enterprises has multiple physical hypervisor (ESXi) hosts running hundreds to hundreds of thousands of virtual machines. To effectively manage this type of environment, a typical deployment will include vCenter Server instances, which acts as a centralized management node, API gateway, and UI for deployments ranging from a single host to an entire datacenter. ESXi and vCenter Server are generally sold together under the brand name vSphere. vCenter Server is not included within the TOE. The exact software/firmware and hardware requirements for the TOE can be found in section 1.5.2.

All access to ESXi takes place via the reverse proxy endpoints, along with a small number of other authentication-based channels. These APIs and channels are used by a multitude of products and features offered by VMware to assist with centralized management of the TOE. What follows is a list of these supportive products and features, which are all external to the TOE:

- vSphere Web Services SDK (documentation, client code out-of-scope) - The vSphere Web Services SDK – not to be confused with the vSphere Web Services API which is in-scope -- is a set of documentation (vSphere API Reference, cited by the TOE Functional Specification), sample code, and client bindings to facilitate access to the vSphere API. The vSphere API is exposed by the TOE as a Web service, running on VMware vSphere server systems. The API provides access to the vSphere management components - the managed objects that you can use to manage, monitor, and control life-cycle operations of virtual machines and other virtual infrastructure components (datacenters, datastores, networks, and so on). The SDK provides optional bindings (for languages such as Java) which simplify the process of making API calls by translating language-specific access to the HTTPS access (VIM API or vAPI) which make up the TOE's TSFIs.
- vCenter Server Appliance (out-of-scope) - The vCenter Server group of services is built into a virtual appliance (currently VMware Photon). Services include vCenter Server for business logic, Distributed Resource Scheduler for cross-host resource scheduling, the Platform Services Controller for SSO functionality, vSphere Web Client for providing user interfaces to the administrator role, and a built-in VMware Certificate Authority for basic PKI needs
- VMware ESXi Host Client (out-of-scope) - The Host Client is a single-page HTML5 application that is used to connect to and manage a single ESXi host. It can be used to perform administrative tasks to manage host resources, such as virtual machines, networking, and storage, though vCenter's Web Client is intended to be the primary interface. The Host Client can also be helpful to troubleshoot individual virtual machines or hosts independent of vCenter Server and the vSphere Web Client
- VMware Remote Console (VMRC) (out-of-scope) - VMRC is a standalone thick client which provides high-performance remote console access to virtual machines. It provides screen, keyboard, and mouse access to virtual machines, similar to a KVM switch in a physical datacenter. In addition to the VIM API, VMRC uses authd and the MKS protocol for Mouse-Keyboard-Screen functionality.

The relationship between the vCenter Server and the hypervisor (ESXi) hosts is a one-to-many relationship: a single vCenter Server manages multiple ESXi hosts and all the virtual machines that reside on those hosts. In a typical deployment, the administrator interacts with vCenter Server which manages ESXis on behalf of the administrator. Direct administrator access to ESXi is atypical in large-scale deployments. The vCenter Server acts as a management console server, and is responsible for deploying, monitoring, and managing virtual machines that are distributed across multiple host computers running the ESXi software. On the client machines, the vSphere Client, vCLI, vSphere Web Client and the vSphere Host Client provide interfaces for the administrator role and users accessing vCenter and ESXi.

The TOE contains cryptographic implementations within the VMware OpenSSL FIPS Module and VMware VMKernel Cryptographic Module.

1.5 TOE Description

This section primarily addresses the physical and logical components of the TOE included in the evaluation.

1.5.1 Logical Boundary

VMware ESXi 8.0g, the TOE, is a virtualization layer that runs directly on industry standard x86-compatible hardware, providing an environment where multiple virtual machines are hosted on one physical host computer. ESXi abstracts processor, memory, storage, and networking resources to create an environment in which virtual machines can run a wide variety of different operating systems and applications. Each virtual machine acts as a separated guest and only communicates with other virtual machines using standard networking protocols.

The following subsystems are the major components within ESXI:

User Worlds:

The "userworlds" are collectively a set of daemons which implement the management functionality of an ESXi hypervisor. They accept incoming communication from the outside world over various protocols. Key daemons include the HTTPS reverse proxy (rhttpproxy) which serves as a TLS endpoint and forwards connections; hostd, which receives those connections and implements the VIM protocols; authd, serves as an endpoint for high-bandwidth services (file transfer and VM console display). Also included in "userworlds" are the various instances of the VMX process, which represents the resources of a running virtual machine - the management daemons directly control this process and its resource consumption.

Importantly, ESXi is not a traditional "operating system". Though it implements a kernel / user level separation, this is done for reliability reasons and not as traditional security mechanism. All user worlds run as the same POSIX user - one of many examples of how ESXi is neither Linux nor POSIX - with actual privilege isolation enforced via sandboxes more akin to ACLs. Each VMX process runs inside its own sandbox to protect the host and other virtual machines.

VMKernel:

The "vmkernel" is VMware's purpose-built kernel, specifically tuned for the unusual resource demands of a virtual machine. The vmkernel specializes in memory management - under normal load, as much as 95% of system memory can be in use for virtual machines, with the hypervisor operating in the remaining 5%. vmkernel also handles "co-scheduling", the necessity to run a virtual machines' CPUs simultaneously to avoid scheduler pathologies. vmkernel contains a network and storage stack, with unique entry points to allow a virtual machine to access storage and networking isolated from vmkernel's normal storage and network stacks. The vmkernel can also load hardware device drivers which implement its

native "vmkapi" interface; VMware publishes a development kit for partners to implement such drivers.

VMM (Virtual Machine Monitor):

The Virtual Machine Monitor (VMM, or "monitor") is a small (<1MB) kernel-mode module which implements the direct interface to a virtual machine by "virtualizing" CPU instructions. The VMM does so via Intel VT instructions (with EPT extensions), or AMD equivalents, AMD-V. In essence, the monitor handles setting up a synthetic environment, instructing the CPU to execute solely within that environment for a time via these VT instructions, then trapping and handling any special behaviors like synthetic input/output (I/O) to storage or networking. The VMM operates in very close conjunction with the VMX process, with performance-sensitive "fast-path" handling implemented in the VMM and less-sensitive "slow-path" handling implemented in the VMX.

VIM API (SOAP over HTTPS):

The VIM API, also known as "vSphere Web Services API" or VMOMI (VMware Object Modeling Interface), is a widely consumed interface to ESXi, accessible via port 443 (HTTPS). VIM is a SOAP-over-HTTPS interface described by several WSDL files which ESXi can serve over HTTPS. Authentication is handled via a session object, which (for ESXi) can accept username/password authentication; most API methods require a successful authentication to operate. The VIM API includes a permissions model, generally declaring a specific permission as necessary to make each API call on an object. Key objects represent a virtual machine, the host, various datastores, a folder to organize these objects, and so on.

In practice, ESXi operates with the administrator role. The same VIM API is implemented by vCenter Server, which offers a more sophisticated user model, but is not part of the TOE in this evaluation.

vAPI (REST over HTTPS):

vAPI is an IDL that supports both REST-like API access (JSON encoding) and gRPC access (Protobuf encoding), and includes an API metadata service to describe the interfaces through API access. Access is over port 443 (HTTPS). Authentication is typically handled by passing a token via an HTTP header when making API calls (SAML and JWT are supported); the token itself is issued through a token management API. Most API methods require a successful authentication to operate. The vAPI interface provides an improved model for typical workflows, and also represents deprecation of many uncommon workflows. Key vAPI objects represent token-based authentication and managing Trust Services (TPM-based attestation).

Reverse proxy endpoints:

ESXi uses a reverse proxy mechanism to redirect HTTPS connections to various service endpoints: most notably those that provide the VIM and vAPI interfaces. This allows multiple services to share a single port (port 443). The redirection is determined by the URL prefix. To ensure system security, the various endpoints authenticate their connections. There are also a small number of other service endpoints (for example authd) that use ports other than 443. These services also authenticate their connections.

authd (NFC and MKS):

The normal ESXi interface protocols (VIM) operate over HTTPS, a text encoding that is comparatively inefficient for high-bandwidth needs. The authd protocol (port 902) serves those needs instead.

The NFC protocol is used to transfer large data objects - uploading or downloading entire files, with a degree of access mediated by the VIM API (e.g. some connections can transfer the contents of a single virtual disk, some connections can transfer arbitrary files from a datastore). NFC is generally used to transfer a virtual machine's disks to or from a host; it is also often used by backup software to capture the state of virtual machines.

The MKS protocol is used to display a virtual machine's console. Again, this access is mediated via the VIM API and grants a ticket which, when presented, allows access to a particular virtual machine's console display. External programs (VMware Remote Console (VMRC) or vSphere Web Client) use this connection to display a VM console when necessary.

Guest (Virtual Machine, not part of the TOE):

ESXi runs unmodified guest operating systems as virtual machines. The guest operating system itself is not part of the TOE.

Hardware (not part of the TOE):

Interaction between: VMM (Monitor) / Intel VT instructions

The monitor uses VT instructions to isolate execution of a virtual machine from the host environment. Actual usage of VT is well-documented in Intel manuals.

The following figure shows the detailed TOE boundary of ESXi. The different subsystems and interfaces are described briefly below.

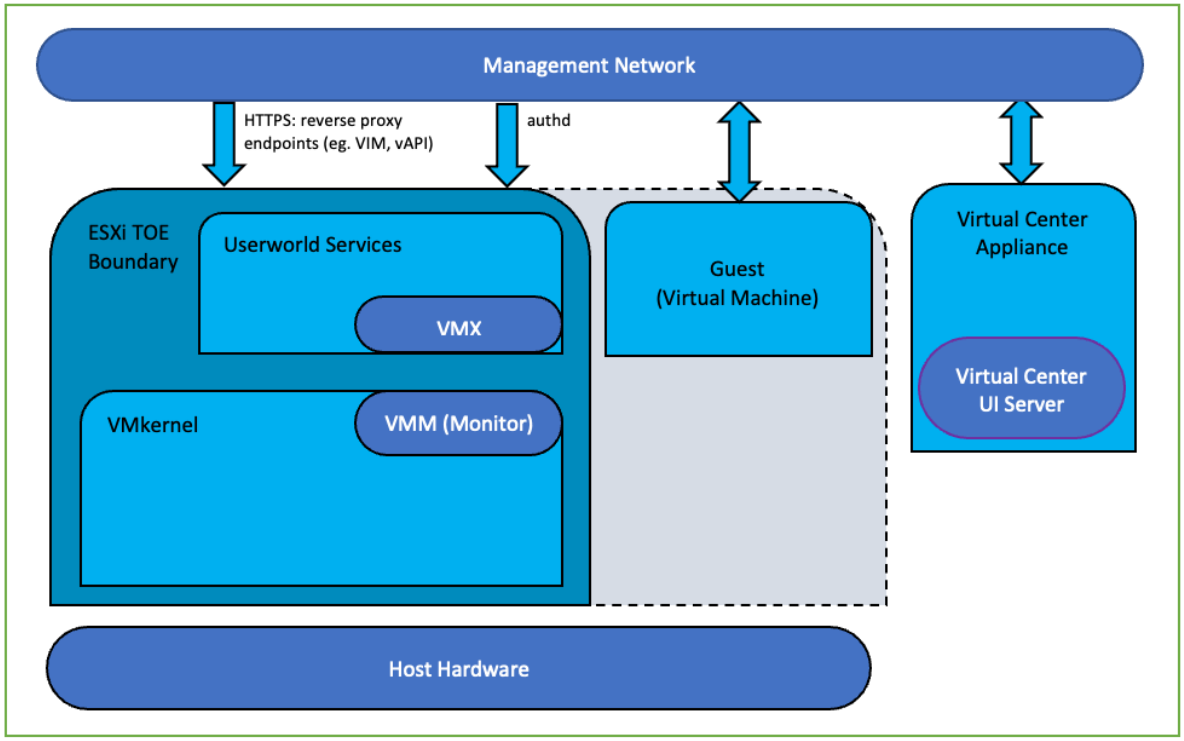


Figure 2: TOE Boundary

1.5.1.1 Out-of-scope software/features in the operational environment of the TOE which may be utilized

This software may be present in the OE of the TOE and do not affect any TSFs, because the TOE ensures software in the OE only accesses the TOE through TSFIs.

This list is provided to reconcile often used software in conjunction with the TOE to confirm that these software are outside the TOE.

- VMware vCenter Server Software, including vSphere Web Client
- VMware vCLI Software
- VMware PowerCLI Software
- VMware ESXi Host Client Software.

1.5.1.2 Out-of-scope software/features which may be utilized

The following additional security features do not impact the evaluated security functions.

Therefore, they are not evaluated and may be used in the evaluated configuration (see section 6.2 of [AGD]).

This list is provided to reconcile parts of the TOE which might be confused with TSFs.

- Core Dump Encryption
- Host Encryption Mode
- Remote syslog
- TPM and vTPM
- VM Encryption
- VM Snapshots management: snapshot creation and deletion, reverting a VM to a previous snapshot, deletion of all snapshots.

1.5.1.3 Out-of-scope software/features which shall not be utilized

The following additional security features may impact the evaluated security functions, and therefore are logically excluded from the TOE boundary and from this evaluation through Administrative Guidance (see section 6.1 of [AGD]):

- 3D Graphics
- 3rd party software
- Active Directory
- CIM and SNMP
- DCUI
- Host Profiles
- IPsec
- Kerberos NFSv4
- MOB (Managed Objet Browser)
- NFC
- Offline bundles
- PCI passthrough
- Physical optical drives
- Raw disks
- Remote ESXCLI
- SSH
- SCSI passthrough
- USB passthrough
- VM virtual disk sharing
- vMotion
- vSAN and vSAN encryption
- Floppy Disks
- Page Share Salting.

1.5.1.4 Security Audit

The auditing security function of the TOE is provided by ESXi. Audit data collected by ESXi is stored in flat files on the ESXi host. Each audit record generated includes the date and time of the event, the type of event, the subject identity (if applicable), and the outcome (success or failure) of the event.

Audit records can be retrieved using VIM APIs.

1.5.1.5 Random Number Generation

The TOE implements a random bit generation service for the purpose of giving good non-predictable random numbers. VMKernel collects entropy from a EAL4 (or above) certified HSM as a noise source to provide random data to the RNG. OpenSSL then seeds its SP800-90A DRBG through the use of the getrandom() syscall. The OpenSSL SP800-90A DRBG is used for random number generation (server/client hello/random), RSA and ECC (ephemeral) key generation used for TLS (see section 7.7).

1.5.1.6 Guest Separation

ESXi provides Guest Separation through execution isolation, volatile & non-volatile memory separation, and network data traffic separation in the TOE.

1.5.1.7 Encrypted Channels

The TOE protects VIM API, vAPI and more generally any reverse proxy endpoints through an HTTPS connection built upon TLS. Authd communication is a proprietary protocol which constitutes a legacy packet followed by a TLS session.

The TOE protects the confidentiality and integrity of all data as it passes between the TOE to another trusted IT product. The TOE achieves this by using OpenSSL which performs the encryption and the decryption of data that is being passed.

1.5.1.8 Identification and Authentication

When the administrator role attempts to log in to ESXi using a username and password as authentication, the provided credentials are compared with the securely stored password hash on the ESXi host.

Note that for the purposes of this ST, Administrative users are considered to be the only users of the TOE. VM users (individuals who access the guest operating system and applications within a virtual machine) are outside the scope of the TOE and are not discussed here.

1.5.1.9 Security Management

Security management specifies how ESXi manages several aspects of the TSF including TSF data and security functions. TSF data includes configuration data of the TOE, audit data, and system data.

VM administrators are administrators of one or more VMs on the ESXi host. VM administrators can access the VMs by directly logging into the ESXi host.

1.5.1.10 Protection of the TSF

The Protection of the TSF function provides the integrity and management of the mechanisms that comprise the TSF. Protection of the TOE from physical tampering is ensured by its environment. It is the responsibility of the administrator role to assure that physical connections made to the TOE remain intact and unmodified. The TOE protects the confidentiality and integrity of all data as it is transmitted to or from the TOE to another trusted IT product by transmitting data solely through encrypted channels (TLS) as described above in “Encrypted Channels”. (Protection of guest data is outside the scope of the TOE).

In addition the TOE protects itself by ensuring an isolation between Guest VMs or the platform and ensures the correctness of hypercalls from the Guest VMs to the hypervisor. Finally the TOE ensures that there is no possibility to bypass the TOE security by connecting removable devices and media.

To protect against execution environment-based vulnerabilities, the TOE leverages Address-Space randomization and memory execution protection. The randomization protects against buffer overflow attacks while the Memory Execution Protection marks memory regions as non-executable to protect against unauthorized execution of machine code. The TOE also can enable several categories of speculative execution mitigation (Spectre) at various isolation versus performance tradeoffs.

1.5.2 Hardware, Firmware, and Software Supplied by the TOE Environment

The following hardware, firmware and software, which are supplied by the IT environment, are excluded from the TOE boundary but needed for operation (that is, are part of the operating environment).

- ESXi host hardware. This may be Dell R740 servers with Intel Xeon processor(s) and a minimum of two storage devices and at least one network adapter. All other R740 server hardware variances (with the exception of explicitly excluded hardware as defined by Section 1.5.1.3) are acceptable since they are outside the scope of the evaluation (for example rack form factor, number of drive bays, number of hardware USB ports, included graphics card, amount of memory, TPM, RAID, etc.).
- iDRAC (integrated Dell Remote Access Controller) with either an Enterprise or Datacenter iDRAC license (not Express)
- Network hardware, including any cabling/routers/switches
  Operating Systems and applications running in VMs
- Client machines used by vSphere Administrators, including web browsers for UI interaction
- An HSM providing random numbers to the TOE, certified according to EAL4 or higher, providing at least a Shannon entropy of 0.91 bits/bit.

1.6 TOE Delivery

The delivery of the TOE is secured in a way that any user can determine the authenticity of the software package. After downloading the deliverables from the designated Broadcom portals, the user can check the integrity and authenticity by calculating and comparing the SHA-256 of the downloaded deliverable with the reference provided in this Security Target, which gets published on the BSI website. Note that the hash values are also provided with the TOE product listing on the Broadcom Common Criteria Security Certification Portal.

Further details can be found in the Administration Guide for secure installation, administration and maintenance of the TOE. The following TOE items are part of the delivery:

Deliverables	Delivery Method
TOE (ESXi) 8.0g Install ISO File: VMware-VMvisor-Installer-8.0g-25191479.x86_64.iso SHA-256: 66121a7158a84a36b1d6a6d6a4d01374f9e75df06291477cf171e513de42c78d	Broadcom Support Portal
Installation and Operational Guidance for ESXi 8.0g [AGD] File: AGD_ESXi8g_Guidance_v1.0.pdf Date: 2026-02-13 SHA-256:	Common Criteria Security

04cc3c3ca7c59a804ca0bfee0267cc550799c65cd8256dbf7cc9d08a171f923e	Certification Portal
Entropy Administrator Guide for ESXi 8.0g File: AGD_ESXi8g_EntropyGuide_v1.0.pdf SHA-256: 3979e3483a039acac5c9df057f2acfbae9fcf68f4a9b4cc6e010a7eb6312b914	
Evaluated Configuration Documentation Package File: AGD_ESXi8g_DocumentationPackage_v1.0.zip SHA-256: 55f7810e1764adde3036c236119abd0cbe2e6b56a6e327fc911202c81b87304a	
VMware vSphere Management SDK File: VMware-vSphere-SDK-8.0.0-20566232.zip SHA-256: 230dc4dbc6ddb7ed549d63cc693c75d63e32b1f089903094cb8a757bac96bc5a	Broadcom Developer Portal

2 Conformance Claims

2.1 CC Conformance Claims

The Security target and the TOE claim conformance to:

- Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, Version 3.1, Revision 5, April 2017, CCMB-2017-04-001,
- Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components, Version 3.1, Revision 5, April 2017, CCMB-2017-04-002,
- Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Components, Version 3.1, Revision 5, April 2017, CCMB-2017-04-003.

The TOE is CC Part 2 [CC2] extended due to the use of FPT_VIV_EXT.1, FPT_HCL_EXT.1, FPT_RDM_EXT.1, and FCS_RNG.1 and is CC Part 3 [CC3] conformant.

2.2 PP Claim

The Security Target does not claim conformance to a Protection Profile.

2.3 Package Claim

The Security Target Claims to be conformant to the Assurance Packet EAL4 augmented by ALC_FLR.2.

2.4 Conformance Claim Rationale

The Security Target does not claim conformance to a Protection Profile. Thus a conformance rationale is not required.

3 Security Problem Definition

This section defines the security problem which the TOE and its operational environment are supposed to address. Specifically, the security problem consists of the following:

- Any known or assumed threats countered by the TOE or its operational environment.
- Any organizational security policies with which the TOE must comply.
- Any assumptions about the security aspects of the environment and/or of the manner in which the TOE is intended to be used.

This section identifies assumptions as *A.Assumption*, threats as *T.Threat,* and policies as *P.Policy*.

3.1 External Entities

External Entity	Description
Administrator Role	The administrator is the only role supported by the TOE in this evaluation. Although ESXi provides several default (predefined) roles, the only role appropriate for the certified configuration when you create administrator accounts is the "Administrator" role.
Attacker	An (unauthenticated) remote or local attacker
VM	One or more Guest Operating Systems running on the TOE including all applications

Table 1: External Entities

3.2 Assets

Assets	Description
VM_DATA	Data sent/received by a guest operating system or application inside a guest operating system flowing through the network interfaces. Basic threats: Modification (Integrity), Disclose (Confidentiality)
TSF_DATA	Security related data (e.g. Cryptographic keys, VM Configuration Data, TOE Configuration Data, Log Files). Basic threats: Modification (Integrity), Disclose (Confidentiality).

Table 2: Assets

3.3 Assumptions

This section describes the security aspects of the environment in which the TOE is intended to operate. The following specific conditions are assumed to exist in an environment where the TOE is employed:

Assumptions	Description
A.PLATFORM_INTEGRITY	The platform has not been compromised prior to installation of the VS

Assumptions	Description
A.PHYSICAL	Physical security of the TOE and the data it contains is assumed to be provided by the environment.
A.NETWORK	The network environment provides the ability to create physically or logically partitioned networks.
A.TRUSTED_ADMIN	The TOE administrator role is trusted to follow and apply all relevant administrator guidance.
A.TIME	The operational environment provides reliable time stamps to the TOE.
A.ENTROPY	An external entropy source, such as an HSM, is provided that generates entropy that is EAL4+ certified at PTG.2.

Table 3: Assumptions

3.4 Threats

This section identifies the threats to the assets against which protection is required by the TOE or by the security environment. The table below lists threats applicable to the TOE and its operational environment:

Threats	Description
T.DATA_LEAKAGE	A VM directly discloses VM_DATA associated to another VM, or TSF_DATA other than those he is entitled to. It is a fundamental property of VMs that the domains encapsulated by different VMs remain separate unless data sharing is permitted by policy. For this reason, all Virtualization Systems shall support a policy that prohibits information transfer between VMs. It shall be possible to configure VMs such that data cannot be moved between domains from VM to VM, or through virtual or physical network components under the control of the VS. When VMs are configured as such, it shall not be possible for data to leak between domains, neither by the express efforts of software or users of a VM, nor because of vulnerabilities or errors in the implementation of the VMM or other VS components. If it is possible for data (VM_DATA or TSF_DATA) to leak between domains when prohibited by policy, then an adversary on one domain or network can obtain data from another domain. Such cross-domain data leakage can, for example, cause classified information, corporate proprietary information, or personally identifiable information to be made accessible to unauthorized entities.
T.UNAUTHORIZED_MODIFICATION	An attacker tries to directly modify TSF DATA or VM_DATA without holding the explicit rights (entitlement) to change the portion of data. System integrity is a core security objective for Virtualization Systems. To achieve system integrity, the integrity of each VMM component must be established and maintained. Malware running on the platform could be able to undetectably modify Virtualization System components while the system is running or at rest if the system integrity is not sufficiently protected by the TOE. Likewise, malicious code running within a virtual machine could be able to modify

Threats	Description
	Virtualization System components, (VM_DATA or TSF_DATA).
T.VMM_COMPROMISE	A VM tries to gain access to VMM functionality restricted from him, in order to modify or disclose TSF_DATA or VM_DATA they are not entitled to. The Virtualization System is designed to provide the appearance of exclusivity to the VMs and is designed to separate or isolate their functions except where specifically shared. Failure of security mechanisms could lead to unauthorized intrusion into or modification of the VM, or bypass of the VM altogether by an attacker. This must be prevented to avoid compromising the Virtualization System.
T.PLATFORM_COMPROMISE	A VM tries to directly modify or disclose a portion of TSF_DATA or VM_DATA without having the explicit access rights to those. The VS must be capable of protecting the platform from threats that originate within VMs and operational networks connected to the VS. The hosting of untrusted—even malicious—domains by the VS cannot be permitted to compromise the security and integrity of the platform on which the VS executes. If an attacker can access the underlying platform in a manner not controlled by the VMM, the attacker might be able to modify system firmware or software including VM_DATA and TSF_DATA—compromising both the Virtualization System and the underlying platform.
T.UNAUTHORIZED_ACCESS	An attacker tries to gain access to VMM functionality or other TOE security functionality, to disclose or modify TSF_DATA or VM_DATA without having the entitlement to do so. Functions performed by the management layer include VM configuration, virtualized network configuration, allocation of physical resources, and reporting. Only certain authorized system users (the administrator role) are allowed to exercise management functions. Virtualization Systems are often managed remotely over communication networks. Members of these networks can be both geographically and logically separated from each other, and pass through a variety of other systems which may be under the control of an adversary, and offer the opportunity for communications to be compromised. An adversary with access to an open management network could inject commands into the management infrastructure. This would provide an adversary with administrator role privilege on the platform, and administrative control over the VMs and virtual network connections. The adversary could also gain access to the management network by hijacking the management network channel (assets: VM_DATA and TSF_DATA).

Table 4: Threats

3.5 Organizational Security Policies (OSPs)

This Security Target contains no Organisational Security Policies (OSPs).

4 Security Objectives

Security objectives are concise, abstract statements of the intended solution to the problem defined by the security problem definition. This high-level solution is divided into two parts: the security objectives for the TOE, and the security objectives for the TOE's operational environment. This section identifies the security objectives for the TOE and its supporting environment.

4.1 Security Objectives for the TOE

Security Objective for the TOE	Description
O.VM_ISOLATION	VMs are the fundamental subject of the system. The VMM is responsible for applying the system security policy (SSP) to the VM and all resources. As basic functionality, the VMM must support a security policy that mandates no information transfer between VMs. The VMM must support the necessary mechanisms to isolate the resources of all VMs. The VMM partitions a platform's physical resources for use by the supported virtual environments. Depending on the use case, a VM may require a completely isolated environment with exclusive access to system resources, or share some of its resources with other VMs. It must be possible to enforce a security policy that prohibits the transfer of data between VMs through shared devices. When the platform security policy allows the sharing of resources across VM boundaries, the VMM must ensure that all access to those resources is consistent with the policy. Devices, whether virtual or physical, are resources requiring access control. The VMM must enforce access control in accordance to system security policy. Physical devices are platform devices with access mediated via the VMM per the O.VMM_Integrity objective. Virtual devices may include virtual storage devices and virtual network devices. The VMM must support the mechanisms to isolate all resources associated with virtual networks and to limit a VM's access to only those virtual networks for which it has been configured. The VMM must also support the mechanisms to control the configurations of virtual networks according to the SSP.
O.VMM_INTEGRITY	Integrity is a core security objective for Virtualization Systems. To achieve system integrity, the integrity of each VMM component must be established and maintained. This

Security Objective for the TOE	Description
	objective concerns only the integrity of the Virtualization System—not the integrity of software running inside of Guest VMs or of the physical platform. The overall objective is to ensure the integrity of critical components of a Virtualization System. Integrity is maintained in a running system by careful protection of the VMM from untrusted users and software. For example, it must not be possible for software running within a Guest VM to exploit a vulnerability in a device or hypercall interface and gain control of the VMM.
O.PLATFORM_INTEGRITY	The integrity of the VMM depends on the integrity of the hardware and software on which the VMM relies. Although the VS does not have complete control over the integrity of the platform, the VS should as much as possible try to ensure that no users or software hosted by the VS is capable of undermining the integrity of the platform.
O.DOMAIN_INTEGRITY	While the VS is not responsible for the contents or correct functioning of software that runs within Guest VMs, it is responsible for ensuring that the correct functioning of the software within a Guest VM is not interfered with by other VMs.
O.MANAGEMENT_ACCESS	VMM management functions include VM configuration, virtualized network configuration, allocation of physical resources, and reporting. Only the authorized system users (with administrator role) are allowed to exercise management functions. Because of the privileges exercised by the VMM management functions, it must not be possible for the VMM's management components to be compromised without administrator notification. This means that unauthorized users cannot be permitted access to the management functions, and the management components must not be interfered with by Guest VMs or unprivileged users on other networks—including operational networks connected to the TOE. VMMs include a set of management functions that collectively allow administrators to configure and manage the VMM, as well as configure Guest VMs. These management functions are specific to the virtualization system, distinct from any other management functions that might exist for the internal management of any given Guest VM. These VMM management functions are privileged, with the security of the entire system relying on their proper use. The VMM management functions can be classified into

Security Objective for the TOE	Description
	different categories and the policy for their use and the impact to security may vary accordingly. The VMM maintains configuration data for every VM on the system. This configuration data must be protected. The mechanisms used to establish, modify and verify configuration data are part of the VS management functions and must be protected as such. Virtualization Systems are often managed remotely. For example, an administrator can start and shut down VMs, and manage virtualized network connections. Communications with the management infrastructure must be protected from Guest VMs and operational networks.
O.AUDIT	The purpose of audit is to capture and protect data about what happens on a system so that it can later be examined to determine what has happened in the past.

Table 5: Security Objectives for the TOE

4.2 Security Objectives for the Environment

The security objectives for the operational environment are addressed below:

Security Objective for the Environment	Description
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment. As a consequence, no local attacker can mount attacks like e.g T.UNAUTHORIZED_MODIFICATION.
OE.NETWORK	The network environment provides the ability to provide physically or logically partitioned networks.
OE.TRUSTED_ADMIN	TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.
OE.TIME	The operational environment provides reliable time stamps to the TOE.
OE.ENTROPY	An external entropy source, such as an HSM, is provided that generates entropy that is EAL4+ certified at PTG.2.

Table 6: Security Objectives for the Environment

4.3 Security Objectives Rationale

The rationale for the Threats, Assumptions, and Objectives for this Security Target can be found in the following table:

Threat, Assumption	Security Objective	Rationale
T.DATA_LEAKAGE	O.VM_ISOLATION O.DOMAIN_INTEGRITY	Logical separation of VMs and enforcement of domain integrity Prevent unauthorized transmission of data from one VM to another.
T.UNAUTHORIZED_MODIFICATION	O.VMM_INTEGRITY O.AUDIT OE.PHYSICAL	Enforcement of VMM integrity prevents the bypass of enforcement mechanisms and auditing ensures that abuse of legitimate authority can be detected.
T.VMM_COMPROMISE	O.VMM_INTEGRITY O.VM_ISOLATION	Maintaining the integrity of the VMM and ensuring that VMs execute in isolated domains mitigate the risk that the VMM can be compromised or bypassed.
T.PLATFORM_COMPROMISE	O.PLATFORM_INTEGRITY	The environment ensures that nobody has logical or physical access which would be required to compromise the platform.
T.UNAUTHORIZED_ACCESS	O.MANAGEMENT_ACCESS OE.PHYSICAL	Ensuring that TSF management functions cannot be executed without authorization prevents untrusted subjects from modifying the behavior of the TOE in an unanticipated manner.
A.PLATFORM_INTEGRITY	OE.PHYSICAL	If the underlying platform has not been compromised prior to installation of the TOE, its integrity can be assumed to be intact.
A.PHYSICAL	OE.PHYSICAL	If the TOE is deployed in a location that has appropriate physical safeguards, it can be assumed to be physically secure.
A.NETWORK	OE.NETWORK	The management and VM networks must be separate to protect the management of the TOE from compromise.
A.TRUSTED_ADMIN	OE.TRUSTED_ADMIN	Providing guidance to administrators and ensuring that individuals are properly trained and vetted before being given administrative responsibilities will ensure that they are trusted.
A.TIME	OE.TIME	If the environment gives a reliable time stamp it is expected that the security functions will work properly, either through NTP, PTP or relying on the underlying

Threat, Assumption	Security Objective	Rationale
		hardware to keep the system clock accurate.
A.ENTROPY	OE.ENTROPY	An external entropy source, such as an HSM, is provided that generates entropy that is EAL4+ certified at PTG.2.

Table 7: Rationale for Security Objectives

Note that OE.PHYSICAL does not per-se counter attacks by a remote attacker. These attacks still must be countered by the TOE's security functionality.

5 Extended Component definition

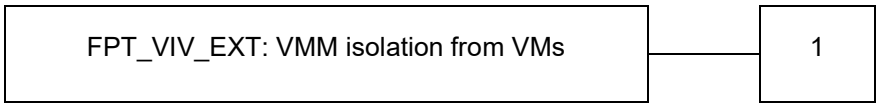
5.1 VMM isolation from VMs (FPT_VIV_EXT)

The additional family VMM isolation from VMs of the Class FPT (Protection of the TSF) is defined here to describe the specific IT security functional requirements of the TOE. The TOE shall prevent attacks from software running in a VM against other VMs running on the same host machine.

5.1.1 Family Behaviour

The family defines requirements to mitigate attacks from software running in a VM against other VMs running on the same host machine. Thereby the TOE shall ensure that a guest VM is unable to degrade or disrupt the functioning of other VMs, the CMM or the underlying platform itself.

5.1.2 Component Levelling



5.1.3 Management

There are no management activities foreseen.

5.1.4 Audit

There are no actions defined to be auditable.

5.1.5 VMM isolation from VMs (FPT_VIV_EXT.1)

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_VIV_EXT.1.1 The TSF must ensure that software running in a VM is not able to degrade or disrupt the functioning of other VMs, the VMM, or the Platform.

FPT_VIV_EXT.1.2 The TSF must ensure that a Guest VM is unable to invoke platform code that runs at a privilege level equal to or exceeding that of the VMM without involvement of the VMM.

5.2 Hypercall Controls (FPT_HCL_EXT)

The additional family Hypercall Controls of the Class FPT (Protection of the TSF) is defined here to describe the specific IT security functional requirements of the TOE. The TOE shall prevent attacks by using the hypercall interface for guest.

5.2.1 Family Behaviour

The family defines requirements to mitigate attacks using the hypercall interface for guest VMs which can be used to invoke functionality provided by the VMM.

5.2.2 Component Levelling

5.2.3 Management

There are no management activities foreseen.

5.2.4 Audit

There are no actions defined to be auditable.

5.2.5 Hypercall Controls (FPT_HCL_EXT.1)

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_HCL_EXT.1.1 The TSF shall provide a Hypercall interface for Guest VMs to use to invoke functionality provided by the VMM.

FPT_HCL_EXT.1.2 The TSF shall validate the parameters passed to the Hypercall interface prior to execution of the VMM functionality exposed by that interface.

5.3 Removable Devices and Media (FPT_RDM_EXT)

The additional family Removable Devices and Media of the Class FPT (Protection of the TSF) is defined here to describe the specific IT security functional requirements of the TOE. The TOE shall prevent attacks by implementing controls for handling the transfer of virtual and physical removable media between different information domains.

5.3.1 Family Behaviour

The family defines requirements for the secure control of different logical or physical medias and the information transfer between different information domains.

5.3.2 Component Levelling

5.3.3 Management

There are no management activities foreseen.

5.3.4 Audit

There are no actions defined to be auditable.

5.3.5 Removable Devices and Media (FPT_RDM_EXT.1)

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_RDM_EXT.1.1 The TSF shall implement controls for handling the transfer of virtual and physical removable media and virtual and physical removable media devices between VMs.

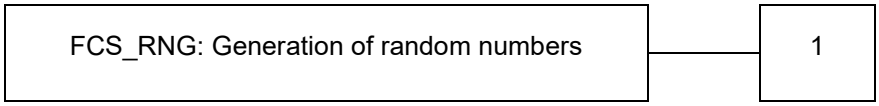
FPT_RDM_EXT.1.2 The TSF shall enforce the following rules when [*assignment: list of devices*] are switched between VMs: [*assignment: list of actions*].

5.4 Generation of Random Numbers

5.4.1 Family Behaviour

This family defines quality requirements for the generation of random numbers that are intended to be used for cryptographic purposes.

5.4.2 Component Levelling



5.4.3 Management

There are no management activities foreseen.

5.4.4 Audit

There are no actions defined to be auditable.

5.4.5 Random number generation (FCS_RNG.1)

Hierarchical to: No other components.

Dependencies: No dependencies.

FCS_RNG.1.1 The TSF shall provide a [selection: physical, non-physical true, deterministic, hybrid physical, hybrid deterministic] random number generator that implements: [assignment: list of security capabilities].

FCS_RNG.1.2 The TSF shall provide random numbers that meet [assignment: a defined quality metric].

6 Security Requirements

This chapter describes the Security Functional and the Assurance Requirements which have to be fulfilled by the TOE. Those requirements comprise functional components from [CC2] and the assurance components as defined for the EAL4 augmented by ALC_FLR.2 from [CC3].

The following notations are used:

- **Refinement** operations (denoted by **bold text**): is used to add details to a requirement, and thus further restricts a requirement. In case that a word has been deleted from the original text this refinement is indicated by ~~**crossed-out bold**~~ text
- **Selection** operation (denoted by underlined text): is used to select one or more options provided by the [CC] in stating a requirement
- **Assignment** operations (denoted by *italicized text*): is used to assign a specific value to an unspecified parameter, such as the length of password
- **Iteration** operation: are identified with a suffix in the name of the SFR.

6.1 Security Functional Requirements

The functional security requirements for this Security Target consist of the components from Part 2 of the CC, and those that were explicitly stated, all of which are summarized in the following table:

Requirement Class	Requirement Name	Description
FAU Security Audit	FAU_GEN.1	Audit Data Generation
	FAU_SAR.1	Audit Review
	FAU_STG.1	Protected Audit Trail Storage
FCS Cryptographic support	FCS_CKM.1/RSA	Cryptographic Key Generation (RSA)
	FCS_CKM.1/TLS	Cryptographic Key Generation (TLS)
	FCS_CKM.2/TLS	Cryptographic Key Distribution (TLS)
	FCS_CKM.4	Cryptographic Key Destruction
	FCS_COP.1/AES	Cryptographic Operation (Symmetric Encryption/Decryption)
	FCS_COP.1/RSA	Cryptographic Operation (Signature Generation/Verification)
	FCS_COP.1/ECDHE	Cryptographic Operation (Shared Secret Computation)
	FCS_COP.1/PRF	Cryptographic Operation (Key Derivation)
	FCS_COP.1/HMAC	Cryptographic Operation (Keyed Message Authentication Code)
	FCS_RNG.1	Random Number Generation
FDP User Data Protection	FDP_ACC.1	Subset Access Control
	FDP_ACF.1	Security attribute-based access control
	FDP_IFC.1	Subset information flow control
	FDP_IFF.1	Simple security attributes
	FDP_ITC.1	Import of user data without security attributes
	FDP_RIP.1	Residual Information Protection
FIA Identification and Authentication	FIA_AFL.1	Authentication Failure Handling
	FIA_SOS.1	Verification of secrets
	FIA_SOS.2	Generation of Secrets
	FIA_UAU.2	User authentication before any action
	FIA_UID.1	Timing of identification
FMT Security Management	FMT_MOF.1	Management of security functions behaviour
	FMT_MSA.1	Management of security attributes
	FMT_MSA.3	Static attribute initialisation
	FMT_SMR.1	Security Roles
	FMT_SMF.1	Specification of Management Functions
FPT Protection of the TSF	FPT_VIV_EXT.1	VMM Isolation from VMs
	FPT_HCL_EXT.1	Hypercall Controls

Requirement Class	Requirement Name	Description
	FPT_RDM_EXT.1	Removable Devices and Media
FTP Trusted path/channel	FTP_ITC.1	Inter-TSF trusted channel

Table 8: SFRs

6.1.1 Class FAU: Security Audit

6.1.1.1 FAU_GEN.1 Audit Data Generation

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;
b) All auditable events for the [minimum] level of audit in Table 9; and
c) [*Specifically defined auditable events listed in Table* 10].

SFRs	Minimum Audit Record Content
FDP_IFF.1	Decisions to permit requested information flows
FIA_AFL.1	The reaching of the threshold for the unsuccessful authentication attempts and the actions taken and the subsequent, if appropriate, restoration to the normal state.
FIA_SOS.1	Rejection by the TSF of any tested secret
FIA_SOS.2	
FIA_UAU.2	Unsuccessful use of the authentication mechanism
FIA_UID.1	Unsuccessful use of the user identification mechanism, including the user identity provided;
FMT_SMR.1	Modifications to the group of users that are part of a role;
FMT_SMF.1	Use of the management functions
FTP_ITC.1	Failure of the trusted channel functions

Table 9: Minimal Auditable Events

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

a) Date and time of the event; Type of event; Subject **and object** identity (if applicable), and the outcome (success or failure) of the event; and
b) For each audit event type, based on the auditable event definitions of the functional components included in the ~~**PP/**~~ST, [*Additional information listed in Table 9: Minimal Auditable Events, and Table 10: Additional Auditable Events below:*].

Requirements	Audit Event	Additional Audit Record Content
FAU_GEN.1	None	None
FAU_SAR.1	None	None
FAU_STG.1	None	None
FCS_CKM.1/RSA	None	None
FCS_CKM.1/TLS	None	None

Requirements	Audit Event	Additional Audit Record Content
FCS_CKM.2/TLS	None	None
FCS_CKM.4	None	None
FCS_COP.1/RSA	None	None
FCS_COP.1.AES	None	None
FCS_COP.1/ECDHE	None	None
FCS_COP.1/PRF	None	None
FCS_COP.1/HMAC	None	None
FCS_RNG.1	Entropy Failure	Failure of the entropy source
FDP_ACC.1	None	None
FDP_ACF.1	None	None
FDP_IFC.1	None	None
FDP_ITC.1	None	None
FDP_RIP.1	None	None
FMT_MOF.1	None	None
FMT_MSA.1	None	None
FMT_MSA.3	None	None
FPT_VIV_EXT.1	None	None
FPT_HCL_EXT.1	Invalid parameter to hypercall detected	Hypercall interface for which access was attempted
FPT_RDM_EXT.1	Connection/disconnection of removable media or device to/from a VM Ejection/insertion of removable media or device from/to an already connected VM	VM Identifier, Removable media/device identifier, event description or identifier (connect/disconnect, ejection/insertion, etc.)
FPT_VIV_EXT.1	N/A (There is no way for a VM to interfer with another VM. Therefore, there is no audit event)	N/A

Table 10: Additional Auditable Events

6.1.1.2 FAU_SAR.1 Audit Review

FAU_SAR.1.1 The TSF shall provide [*administrators*] with the capability to read [*all information*] from the audit records.

FAU_SAR.1.2 The TSF shall provide the audit records in a manner suitable for the user to interpret the information.

Application Note: Audit events are displayed by an external tool that uses the VIM API.

6.1.1.3 FAU_STG.1 Protected Audit Trail Storage

FAU_STG.1.1 The TSF shall protect the stored audit records in the audit trail from unauthorized deletion.

FAU_STG.1.2 The TSF shall be able to [prevent] unauthorized modifications to the stored audit records in the audit trail.

6.1.2 Class FCS: Cryptographic Support

6.1.2.1 FCS_CKM.1/RSA Cryptographic Key Generation (RSA)

FCS_CKM.1.1/RSA The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [*RSA schemes*] and specified cryptographic key sizes [*of 3072-bit or greater*] that meet the following: [*FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.1*].

6.1.2.2 FCS_CKM.1/TLS Cryptographic Key Generation (TLS)

FCS_CKM.1.1/TLS The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [*TLS1.2 Key Derivation Function (KDF) with PRF-SHA384]* and specified cryptographic key sizes *[as denoted in Table 11]* that meet the following: *[As denoted in Table 11].*

Operation/Purpose	Algorithms / Cipher Suite	Key Lengths	Standard
Key Derivation Function	PRF based on HMAC with SHA-384 (PRF-SHA384)	Variable	[FIPS180-4] [RFC2104] [RFC5246]

Table 11: Cryptographic Key Generation

Application Note: Key Generation denotes derivation of session keys from the shared secret.

6.1.2.3 FCS_CKM.2/TLS Cryptographic Key Distribution (TLS)

FCS_CKM.2.1/TLS The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method [*see Table 11*] that meets the following: *[see Table 12*].

Operation/Purpose	Algorithms / Cipher Suite	Key Lengths	Standard
Key Agreement	Ephemeral elliptic curve DH key exchange supports the P-256, P-384, and the P-521 curves.	256 Bits (P-256), 384 Bits (P-384), 521 Bits (P-521)	[RFC5246] [RFC8422] [SEC2]

Table 12: Cryptographic Key Exchange

Application Note: Key Distribution refers to the negotiation of the TLS shared secret.

6.1.2.4 FCS_CKM.4 Cryptographic Key Destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [*zeroisation*] that meets the following: [*no standard*].

Application Note: Cryptographic Key Destruction is used for keys used for TLS connection.

6.1.2.5 FCS_COP.1/AES Cryptographic Operation (Symmetric Encryption/Decryption)

FCS_COP.1.1/AES The TSF shall perform [*authenticated encryption/decryption*[1]] in accordance with a specified cryptographic algorithm: [*AES-GCM*] and cryptographic key sizes [*256 bits*] that meet the following: [*FIPS Pub 197, RFC5288, RFC5246, RFC5116, and SP 800-38D*].

6.1.2.6 FCS_COP.1/RSA Cryptographic Operation (Signature Generation/Verification)

FCS_COP.1.1/RSA The TSF shall perform [*signature generation/verification*] in accordance with a specified cryptographic algorithm: [*RSA Signature Schemes*] and cryptographic key size [*3072 bits*] that meet the following: [*FIPS Pub 186-4*] using the following supported scheme and hash algorithm combinations:

Operation / Purpose	Algorithms / Cipher Suite	Key Lengths	Standard
Signature Generation/ Verification	RSASSA-PSS (via rsa_pss_rsae schemes) • with SHA-256 • with SHA-384 • with SHA-512	3072 bits	FIPS Pub 186-4
Signature Generation/ Verification	RSASS-PKCS-v1_5 • with SHA-256 • with SHA-384 • with SHA-512	3072 bits	FIPS Pub 186-4

Table 13: Cryptographic Operations – RSA

Application Note: The TOE supports both RSASS-PKCS1-v1_5 (legacy) and RSASSA-PSS (high security) within TLS 1.2 connections, negotiated via the standard *signature_algorithms* extension.

6.1.2.7 FCS_COP.1/ECDHE Cryptographic Operation (Shared Secret Computation)

FCS_COP.1.1/ECDHE The TSF shall perform [*shared secret computation*] in accordance with a specified cryptographic algorithm: [*EC Diffie-Hellman Ephemeral*] and cryptographic key sizes [*P-256, P-384 and P-521*] that meet the following: [*SP 800-56Arev3*].

6.1.2.8 FCS_COP.1/PRF Cryptographic Operation (Key Derivation Function)

FCS_COP.1.1/PRF The TSF shall perform [*key derivation*] in accordance with a specified cryptographic algorithm: [*TLSv1.2 Pseudo Random*

[1] AES encrypt/decrypt operations are performed using the AES-NI implementation provided by the Intel (x86_64) processor part of the hardware platforms on which the TOE runs, as referenced in section TOE Overview. The user of the TOE cannot disable AES-NI or use a different implementation of AES.

Function (PRF) with SHA-384] and cryptographic key sizes [*not applicable*] that meet the following: [*RFC5246*].

6.1.2.9 FCS_COP.1/HMAC Cryptographic Operation (Keyed Hash-based Message Authentication Code)

FCS_COP.1.1/HMAC The TSF shall perform [*keyed message authentication code generation and verification*] in accordance with a specified cryptographic algorithm: [*HMAC-SHA-256*] and cryptographic key sizes [*256 bits*] that meet the following: [*FIPS 198*].

6.1.2.10 FCS_RNG.1 Random Number Generator

FCS_RNG.1.1 The TSF shall provide a [*hybrid deterministic*] random number generator that implements: [

- *DRG.4.1: The internal state of the RNG shall contain at least 200 bits of min-entropy when initialized with a random seed of 384 Bits using a PTRNG of class PTG.2 as the random source.*
- *DRG.4.2: The RNG provides forward secrecy.*
- *DRG.4.3: The RNG provides backward secrecy even if the current internal state is known.*
- *DRG.4.4: The RNG provides enhanced forward secrecy on condition that 2^{24} random numbers have been requested, or after one hour and seven minutes of time has elapsed.*
- *DRG.4.5: The internal state of the RNG is seeded by an PTRNG of class PTG.2.*

FCS_RNG.1.2 The TSF shall provide random numbers that meet [

- *DRG.4.6: The RNG generates output for which 2^{34} strings of bit length 128 are mutually different with probability $1-2^{-16}$.*
- *DRG.4.7: Statistical test suites cannot practically distinguish the random numbers from output sequences of an ideal RNG. The random numbers must pass test procedure A [and no other test suites.*]

Application Note: The DRG.4 utilizes an [SP800-90ARev1] conformant CTR-DRBG instance with a 256-bit security strength.

6.1.3 Class FDP: User Data Protection

6.1.3.1 FDP_ACC.1 Subset access control

FDP_ACC.1.1 The TSF shall enforce the [*policy of admin-only access*] on [*CA certificate and SAML validation rules*].

6.1.3.2 FDP_ACF.1 Security attribute-based access control

FDP_ACF.1.1 The TSF shall enforce the [*access to objects defined in FDP_ACC.1*] to objects based on the following: [*only admins*].

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [*operations*

accessing the objects defined in FDP_ACC.1 are only allowed for admins].

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: [*only admins are allowed to access objects mentioned in FDP_ACC.1*].

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [*non-admins users are not allowed to access any objects*].

6.1.3.3 FDP_IFC.1 Subset information flow control

FDP_IFC.1.1 The TSF shall enforce the [*Guest VM virtual network data sharing policy*] on [*the list of subjects, information, and operations that cause controlled information flow to and from controlled subjects as defined in Table 14: Information Flow Policy*].

Subjects	Information	Operations
Guest VMs	Ethernet Packets	Transmit/receive network data
Port Groups	Ethernet Packets	Permit or deny
Virtual Switches	Ethernet Packets	Permit or deny

Table 14: Information Flow Policy

6.1.3.4 FDP_IFF.1 Simple security attribute

FDP_IFF.1.1 The TSF shall enforce the [*Guest VM virtual network data sharing policy*] based on the following types of subject and information security attributes:[

Subject Security Attributes:

- *Guest VM:*
  - *Port Group*
- *Port Group:*
  - *VLAN tag*
- *Virtual Switch:*
  - *List of Port Groups*

Information Security Attributes:

- *Ethernet Packet optional VLAN tag*

].

FDP_IFF.1.2 The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [

Between Guest VMs:

- *Guest VMs are assigned the same Port Group, or*
- *The Port Groups assigned to the Guest VMs are part of the same Virtual Switch, and those Port Groups use the same VLAN tag or one of the reserved VLAN tags.*

]

Application note

Reserved VLAN tags are 0 (no VLAN tagging) and 4095 (trunk mode, where the tagging is preserved with the Ethernet Packet). No VLAN tag is equivalent to VLAN tag 0. Virtual Switches do not filter Ethernet Packets for Port Groups configured with a reserved VLAN tag. Ethernet Packets are simply passed through to such Port Groups.

FDP_IFF.1.3 The TSF shall enforce the [*requirement that virtual networks are the only mechanism that permits information flow between Guest VMs, as defined by the Guest VM virtual network data sharing policy*].

FDP_IFF.1.4 The TSF shall explicitly authorise an information flow based on the following rules: [

- *Within a Virtual Switch, Ethernet Packets are forwarded between Port Groups if the Port Groups have the same VLAN tag or one of the reserved VLAN tags.*

].

FDP_IFF.1.5 The TSF shall explicitly deny an information flow based on the following rules: [

- *Within the TSF, information flow between Virtual Switches is denied.*
- *Ethernet Packets associated with a VLAN will not be forwarded to Port Groups that are not configured with the same VLAN tag or not configured with one of the reserved VLAN tags.*

].

6.1.3.5 FDP_ITC.1 Import of user data without security attributes

FDP_ITC.1.1 The TSF shall enforce the [*that only ADMINs are allowed to import CAs*] when importing **user CA certificates permissible to validate SAML tokens for authentication, as well as the rules to map the SAML token to the administrator role**, controlled under the SFP, from outside of the TOE.

FDP_ITC.1.2 The TSF shall ignore any security attributes associated with the user data when imported from outside the TOE.

FDP_ITC.1.3 The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: [*SAML validation rules*].

6.1.3.6 FDP_RIP.1 Subset of residual information protection

FDP_RIP.1.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the [allocation of the resource to] the following objects: [*Other Guest VMs*].

Application Note: The resources in consideration are volatile and non-volatile memory.

6.1.4 Class FIA: Identification and Authentication (FIA)

6.1.4.1 FIA_AFL.1 Authentication Failure Handling

FIA_AFL.1.1 The TSF shall detect when [*an administrator role configurable positive integer within [1-10]*] unsuccessful authentication attempts occur related to [*authentication based on username and password*].

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been [met], the TSF shall: [*account lockout for 15 minutes*].

6.1.4.2 FIA_SOS.1 Verification of secrets

FIA_SOS.1.1 The TSF shall provide a mechanism to verify that secrets meet [*the following password definition rule:*

a. *Passwords shall be able to be composed of any combination of upper and lower case characters, digits, and the following special characters: "!", "@", "#", "$", "%", "^", "&", "*", "(", ")"; It must contain at least one uppercase, lowercase, one number and one special character.*

b. *Minimum Password length shall be configurable;*

c. *Passwords of at least 16 characters in length shall be required.*]

6.1.4.3 FIA_SOS.2 Generation of secrets

FIA_SOS.2.1 The TSF shall provide a mechanism to generate secrets that meet:

- [*128-bit tokens/session IDs generated from a DRG.4 RNG, as stated in FCS_RNG.1, making up at least 100 bits of entropy*
- *128-bit tickets with a lifetime for less than 5 minutes, generated from a DRG.4 RNG, as stated in FCS_RNG.1, making up 64 bits of entropy*].

FIA_SOS.2.2 The TSF shall be able to enforce the use of TSF generated secrets for [*session IDs, tokens or tickets for user-authentication*].

6.1.4.4 FIA_UAU.2 User authentication before any action

FIA_UAU.2.1 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

6.1.4.5 FIA_UID.1 Timing of identification

FIA_UID.1.1 The TSF shall allow [*none of the TSF-mediated actions*] on behalf of the user to be performed before the user is identified.

FIA_UID.1.2 The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

6.1.5 Class FMT: Security Management (FMT)

6.1.5.1 FMT_SMR.1 Security Roles

FMT_SMR.1.1 The TSF shall maintain the roles [*administrator*].

FMT_SMR.1.2 The TSF shall be able to associate users with roles.

6.1.5.2 FMT_SMF.1 Specification of Management

FMT_SMF.1.1 The TSF shall be capable of performing the following **remote** management functions **by an administrator role**: [*as listed in Table 15*]

Number	Management Function	Management subfunction
1.	Ability to enable/disable the transfer of data between Guest VMs	Inter-VM network communication
2.	Ability to control Guest Networking (e.g. configure virtual networking components to connect VMs to each other and to physical networks)	Create virtual network
		Destroy virtual network
		Linking virtual NICs to VMs
		Ability to separate the host/guest network (separation of management and operational)
3.	Ability to control virtual device backings	ISO (virtual CD/DVD)
		VMDK (virtual hard disk)
		USB storage
4.	Ability to configure user accounts with administrator role	Create account
		Update account
		Delete account
		Change account password
5.	Ability to manage VMs	Ability to create VM
		Ability to delete a VM
		Ability to register a VM
		Ability to unregister a VM
		Ability to power-on/resume a VM
		Ability to power-off a VM
		Ability to terminate a VM
		Ability to suspend a VM
6.	Ability to configure the limit of unsuccessful login attempts before temporary account lockout	Configure how many attempts before lockout
		Configure duration of the account lockout
7.	Ability to manage local audit	n/a
8.	Ability to configure remote connection inactivity timeout and define global password complexity	n/a
9.	Ability to control host power state	Power-Off

Number	Management Function	Management subfunction
		Reboot
10.	Ability to manage certificates	Import new host certificate
		Manage certificate store
11.	Ability to manage the system's available entropy	n/a

Table 15: Remote Management Functions

6.1.5.3 FMT_MOF.1 Management of security functions behavior

FMT_MOF.1.1 The TSF shall restrict the ability to [*modify the behaviour of*] the functions [*listed in Table 16]* to [*the administrator role*].

	Administrator Role	Non-Administrator Role
Ability to restrict all management functions to a specific user	X	

Table 16: Management of security functions behaviour

6.1.5.4 FMT_MSA.1 Management of security attributes

FMT_MSA.1.1 The TSF shall enforce the [*administrator role access control policy*] to restrict the ability to [*manage*] the security attributes [*listed in Table 17*] to [*the administrator role*].

Operation	Objects / Security Attributes
Add/Remove/Modify Hardware or virtual device resources	Guest VM settings
Information Flow Control, Allow, Separate or Deny	Host settings
Information Flow Control, Allow, Separate or Deny	Network settings
Allow / Deny Access	Guest VM peripheral settings
Set, Change	Administrator Role Password

Table 17: Management of security attributes

6.1.5.5 FMT_MSA.3 Static attribute initialisation

FMT_MSA.3.1 The TSF shall enforce the [*administrator role access control policy*] to provide [*restrictive*] default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the [*administrator role*] to specify alternative initial values to override the default values when an object or information is created.

6.1.6 Class FPT: Protection of the TSF (FPT)

6.1.6.1 FPT_VIV_EXT.1 VMM Isolation from VMs

FPT_VIV_EXT.1.1 The TSF must ensure that software running in a VM is not able to degrade or disrupt the functioning of other VMs, the VMM, or the Platform.

FPT_VIV_EXT.1.2 The TSF must ensure that a Guest VM is unable to invoke platform code that runs at a privilege level equal to or exceeding that of the VMM without involvement of the VMM.

6.1.6.2 FPT_HCL_EXT.1 Hypercall Controls

FPT_HCL_EXT.1.1 The TSF shall provide a Hypercall interface for Guest VMs to use to invoke functionality provided by the VMM.

FPT_HCL_EXT.1.2 The TSF shall validate the parameters passed to the Hypercall interface prior to execution of the VMM functionality exposed by that interface.

6.1.6.3 FPT_RDM_EXT.1 Removable Devices and Media

FPT_RDM_EXT.1.1 The TSF shall implement controls for handling the transfer of virtual and physical removable media and virtual and physical removable media devices between VMs.

FPT_RDM_EXT.1.2 The TSF shall enforce the following rules when [*removable media and images*] are switched between VMs: [

a) *The administrator role must granted explicit access for the media or device to be connected to the receiving VM.*

b) *A device or media can only be connected to one VM at a time.*]

6.1.7 Class FTP: Inter-TSF trusted channel (FTP_ITC)

6.1.7.1 FTP_ITC.1 Inter-TSF trusted channel

FTP_ITC.1.1 The TSF shall provide a communication channel between itself and another trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure **using the following mechanisms:**

1. **Cryptographic protection:**

a) Cryptographically-protected communication channel using Webservices (includes VIM & vAPI) API with a combination of the following cipher suite defined there:

- **Cryptographic ciphers and algorithms defined in FCS_COP.1/AES, FCS_COP.1/RSA, FCS_COP.1/ECDHE and FCS_COP.1/PRF to satisfy the TLS cipher suite TLS_ECDHE_RSA_AES_256_GCM_SHA384 defined in [RFC5246]**

b) Authenticated communication channel using TLS as defined in [RFC5246] for server authentication.

c) Authenticated communication channel using a password authentication scheme as defined in FIA_SOS.1, sessions IDs, tokens or tickets as defined in FIA_SOS.2. FCS_COP.1/HMAC is used to generate/verify token's MAC.

2. Physical or logical protection through separation of the management and operational networks.

FTP_ITC.1.2 The TSF shall permit [*the TSF, another trusted IT product*] to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for [*all security functions specified in the ST that interact with remote trusted IT systems and no other conditions or functions*].

6.2 Security Assurance Requirements

The following table summarizes the Security Assurance Requirements for EAL4 augmented by ALC_FLR.2 as defined in [CC3]:

Assurance Class	Assurance Component
ASE: Security Target evaluation	ASE_CCL.1 Conformance claims
	ASE_ECD.1 Extended components definition
	ASE_INT.1 ST introduction
	ASE_OBJ.2 Security Objectives
	ASE_REQ.2 Derived Security Requirements
	ASE_SPD.1 Security Problem Definition
	ASE_TSS.1 TOE summary specification
ADV: Development	ADV_ARC.1 Security architecture description
	ADV_FSP.4 Complete functional specification
	ADV_IMP.1 Implementation representation of the TSF
	ADV_TDS.3 Basic modular design
AGD: Guidance documents	AGD_OPE.1 Operational user guidance
	ADG_PRE.1 Preparative procedures
ALC: Life-cycle support	ALC_CMC.4 Production support, acceptance procedures and automation

Assurance Class	Assurance Component
	ALC_CMS.4 Problem tracking CM coverage
	ALC_DEL.1 Delivery procedures
	ALC_DVS.1 Identification of security measures
	ALC_LCD.1 Developer defined life-cycle model
	ALC_FLR.2 Flaw reporting procedures
	ALC_TAT.1 Well-defined development tools
ATE: Tests	ATE_COV.2 Analysis coverage
	ATE_DPT.1 Testing: basic testing
	ATE_FUN.1 Functional testing
	ATE_IND.2 Independent testing – sample
AVA: Vulnerability assessment	AVA_VAN.3: Focused vulnerability analysis

6.3 Security Requirements Rationale

6.3.1 Rationale for the Security Functional Requirements (SFRs)

6.3.1.1 Fulfilment of the Security Objectives

The following table shows that the security functional requirements fulfill the Security Objectives for the TOE:

Security Objective for the TOE	Security Functional Requirement	Rationale
O.VM_ISOLATION	FMT_SMF.1	Defines the requirements to configure virtual network components to ensure that network traffic visible to a guest operating system is not visible to other guest operating systems. Additionally it defines the requirements to control the access of guest operating systems to the different physical platform resources Additionally it defines the requirements that allows the administrator role to configure virtual networking components to connect different guest operating systems to each other by ensuring that only traffic is visible that shall be shared between different VMs.
	FDP_IFC.1	Defines the requirements for the VM separation and possible data transfer between different guest operating systems.
	FDP_RIP.1	Defines the requirements that ensures that previously used information of a guest operating system cannot be

Security Objective for the TOE	Security Functional Requirement	Rationale
		seen by other guest operating systems.
	FPT_VIV_EXT.1	Defines the requirements to ensure that software running in a VM is not able to degrade or disrupt the functioning of other guest operating systems.
	FPT_HCL_EXT.1	Defines the requirements for the TSF Hypercall interface for Guest VMs to use to invoke functionality provided by the VMM.
	FPT_RDM_EXT.1	Defines the requirements to ensure that virtual and physical removable media is controlled during transfer between different guest operating systems.
O.VMM_INTEGRITY	FDP_IFC.1	Defines the requirements for the VM separation and possible data transfer between different guest operating systems.
	FDP_IFF.1	Defines the requirements for simple security attributes.
	FMT_SMF.1	Defines the requirements to control the access of guest operating systems to the physical platform resources
O.PLATFORM_INTEGRITY	FAU_STG.1	Defines the requirements to store generated audit trails, most notably requirements for the protection of stored audit records.
	FPT_VIV_EXT.1	Defines the requirements to ensure that software running in a VM is not able to degrade or disrupt the functioning of the VMM (as well as other guest operating systems).
	FPT_HCL_EXT.1	Defines the requirements for the TSF Hypercall interface for Guest VMs to use to invoke functionality provided by the VMM, most notably parameter validation.
	FTP_ITC.1	Defines the requirements for the Inter-TSF trusted channel (TLS) for VIM and vAPI APIs, most notably protection of channel data.
O.DOMAIN_INTEGRITY	FMT_SMF.1	Defines the requirements that allow the administrator role to configure virtual networking components to connect different guest operating systems to each other by ensuring that only traffic shared between different VMs is visible to each other.
	FDP_RIP.1	Defines the requirements that ensure that previously used information of a

Security Objective for the TOE	Security Functional Requirement	Rationale
		guest operating system cannot be seen by other guest operating systems.
O.MANAGEMENT_ACCESS	FDP_ACC.1	Defines the access control requirement for the admin user.
	FDP_ACF.1	Defines what users can access which objects: only admins can access the objects defined in FDP_ACC.1.
	FDP_ITC.1	Defines the type of user data admin users can import within the TOE boundary.
	FIA_AFL.1	Defines the requirements that ensure that the administrator role account will be temporarily disabled if more than the configured number of unsuccessful login attempts are registered.
	FIA_SOS.1	Defines the requirements that enforce a policy for secure passwords.
	FIA_SOS.2	Defines the metric with which sessions IDs, tokens or tickets are being generated for user authentications.
	FIA_UAU.2	Defines the requirements that enforce user authentication before any action can be done.
	FIA_UID.1	Defines the requirements that enforce the user identification before any action be done.
	FMT_SMF.1	Allows for the administrator to define the password policy, create/update/delete an admin account, import or update a host certificate, change the password lockout policy and timeout.
	FMT_SMR.1	Defines the requirements that enforce a security policy that prohibits the sharing of data between different Guest VMs using the virtual network.
	FMT_MOF.1	Defines the requirements for managing the security functions behaviour.
	FMT_MSA.1	Defines the requirements for managing security attributes.
	FMT_MSA.3	Defines the requirements to initiate static attributes.
	FCS_CKM.1/TLS	Defines the cryptographic operations required by the TOE for
	FCS_CKM.2/TLS	

Security Objective for the TOE	Security Functional Requirement	Rationale
	FCS_COP.1/AES	communication with the environment through TLS. Authentication mechanism for tokens.
	FCS_COP.1/RSA	
	FCS_COP.1/ECDHE	
	FCS_COP.1/HMAC	
	FCS_COP.1/PRF	
	FCS_CKM.4	Defines the requirements around the secure deletion of cryptographic keys.
	FTP_ITC.1	Defines the requirements for the Inter-TSF trusted channel (TLS) for VIM and vAPI APIs.
	FCS_RNG.1	Defines the requirements to generate random keys.
O.AUDIT	FAU_GEN.1	Defines the requirements for audit generation.
	FAU_SAR.1	Defines the requirements to review the created audit trails.
	FAU_STG.1	Defines the requirements to store generated audit trails.

Table 18: Fulfilment of Security Objectives

6.3.1.2 Fulfilment of the Dependencies

The following table shows how each dependency of the security functional requirements is fulfilled:

Security Functional Requirement	Dependency according to [CC2]	Dependency fulfilled
FAU_GEN.1	[FPT_STM.1]	This dependency is fulfilled by the environment. Please see assumption A.TIME.
FAU_SAR.1	[FAU_GEN.1]	FAU_GEN.1
FAU_STG.1	[FAU_GEN.1]	FAU_GEN.1
FCS_CKM.1/RSA	[FCS_CKM.2 or FCS_COP.1] and FCS_CKM.4	FCS_COP.1/RSA and FCS_CKM.4
FCS_CKM.2/TLS	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] and FCS_CKM.4	FCS_RNG.1 and FCS_CKM.4 For key distribution of TLS session secrets by ECDHE, the nonces generated by the hybrid deterministic DRG.4 are considered the raw key material – fulfils dependency FCS_CKM.1 See also FCS_COP.1/ECDHE
FCS_CKM.1/TLS	[FCS_CKM.2 or FCS_COP.1] and FCS_CKM.4	FCS_CKM.2/TLS and FCS_CKM.4

Security Functional Requirement	Dependency according to [CC2]	Dependency fulfilled
FCS_COP.1/AES	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] and FCS_CKM.4	FCS_COP.1/ECDHE, FCS_COP.1/PRF and FCS_CKM.4: FCS_COP.1/ECDHE and FCS_COP.1/PRF combined provide the AES keying material for the TLS encrypted channel for both endpoints.
FCS_COP.1/RSA		FCS_CKM.1/RSA and FCS_CKM.4
FCS_COP.1/ECDHE	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] and FCS_CKM.4	FCS_CKM.1/TLS, FCS_RNG.1 and FCS_CKM.4: OpenSSL uses the cryptographically secure random number generator to obtain the private key d of the Elliptic-Curve Diffie-Hellman shared secret. The shared secret (public key) is derived by multiplying d with the client's public Diffie-Hellman key.
FCS_COP.1/PRF	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] and FCS_CKM.4	FCS_CKM.1/TLS and FCS_CKM.4: the output of the ECDHE key agreement is in the input to the PRF. Note that the HMAC algorithm is not used for the purpose of packet authentication but rather internally by the TLSv1.2 PRF.
FCS_COP.1/HMAC	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] and FCS_CKM.4	FCS_RNG.1: OpenSSL generates HMAC keys randomly, outside of the TLS protocol.
FCS_CKM.4	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	FCS_CKM.1/RSA and FCS_CKM.1/TLS[2]
FCS_RNG.1	None	None
FDP_ACC.1	FDP_ACF.1	FDP_ACF.1
FDP_ACF.1	FDP_ACC.1 and FMT_MSA.3	FDP_ACC.1 and FMT_MSA.3
FDP_IFC.1	FDP_IFF.1	FDP_IFF.1
FDP_IFF.1	FDP_IFC.1 and FMT_MSA.3	FDP_IFC.1 and FMT_MSA.3
FDP_ITC.1	[FDP_ACC.1 or FDP_IFC.1] and FMT_MSA.3	FDP_ACC.1, FDP_IFC.1 and FMT_MSA.3
FDP_RIP.1	None	None
FIA_AFL.1	FIA_UAU.1	FIA_UAU.2
FIA_SOS.1	None	None
FIA_SOS.2	None	None
FIA_UAU.2	FIA_UID.1	FIA_UID.1
FIA_UID.1	None	None

[2] Note that all keys and secrets, and generally any data that is part of the TLS protocol will be zeroized by OpenSSL once the TLS session is over.

Security Functional Requirement	Dependency according to [CC2]	Dependency fulfilled
FMT_SMR.1	FIA_UID.1	FIA_UID.1
FMT_SMF.1	None	None
FMT_MOF.1	FMT_SMR.1 and FMT_SMR.1	FMT_SMR.1 and FMT_SMR.1
FMT_MSA.1	[FDP_ACC.1 or FDP_IFC.1] and FMT_SMR.1 and FMT_SMF.1	FDP_IFC.1 and FMT_SMR.1 and FMT_SMF.1
FMT_MSA.3	FMT_MSA.1 and FMT_SMR.1	FMT_MSA.1 and FMT_SMR.1
FPT_VIV_EXT.1	None	None
FPT_HCL_EXT.1	None	None
FPT_RDM_EXT.1	None	None
FTP_ITC.1	None	None

Table 19: Fulfilment of SFR Dependencies

6.4 Rationale for the Security Assurance Requirements (SARs)

The assurance package EAL4 augmented by ALC_FLR.2 has been chosen because it “permits a developer to gain maximum assurance from positive security engineering based on good commercial development practices which, though rigorous, do not require substantial specialist knowledge skills, and other resources. EAL4 augmented by ALC_FLR.2 is the highest level at which it is likely to be economically feasible to retrofit to an existing product line. EAL4 augmented by ALC_FLR.2is therefore applicable in those circumstances where developers or users require a moderate to high level of independently assured security in conventional commodity TOEs and are prepared to incur additional security specific engineering costs” [CC3 §111f].

6.4.1 Fulfilment of the Dependencies

The dependencies of the assurance requirements taken from EAL4 augmented by ALC_FLR.2 are fulfilled automatically.

7 TOE Summary Specification

This section presents information to detail how the TOE meets the security functional requirements described in previous sections of this ST.

This chapter describes the following security functions:

- Security Audit
- Cryptographic Support
- User Data Protection
- Identification and Authentication
- Security Management
- Protection of the TSF
- TOE Access
- Trusted Path/Channel
- Lifecycle and Updates.

7.1 SF1.Security Audit

SFRs	Rationale
FAU_GEN.1 Audit Data Generation	The auditing security function of the TOE is provided by ESXi. The TOE generates audit records regarding the start-up and shutdown of audit functions and as identified in Table 10: Additional Auditable Events, the TOE supports auditing of the Security Functional Requirements. Each audit record generated includes the date and time of the event, type of event, subject and object identity (if applicable), and the outcome (success or failure) of the event, and the additional information as defined in Table 10: Additional Auditable Events.
FAU_SAR.1 Audit Review	The audit data is stored locally to the TOE and is not sent anywhere outside the TOE boundary. The review of the audit records on the ESXi host is restricted to the ESXi System Administrator.
FAU_STG.1 Protected Audit Trail Storage	Within the TOE, audit records are stored as flat files on the ESXi host. ESXi provides the capability to review its audit records which are stored in /scratch/auditLog. These audit records are protected by the file system and restricted for review only by the administrator role.

Table 20: Security Audit SFRs

The TOE (ESXi) includes a security audit function for recording security-relevant behavior that occurs. The TSF generates audit records locally for all audit events listed in Table 10: Additional Auditable Events above. Each audit record includes date, time, applicable subject and objective identities, the outcome of the event, and any additional information required by the TOE's conformance claims on a per-event basis.

Specific examples of each audit record can be found in the supplemental administrative guidance.

Audit records are stored on the TOE's file system as flat files. They are protected from unauthorized access through file system permissions as well as through logical access controls on the TOE's management interfaces. Audit records can be reviewed using the TOE via the VIM API, but only the administrator role has the ability to do this. There is no interface to modify or manually delete stored audit records.

The TOE can be configured to specify the maximum size of local audit record storage. Local audit records are stored as flat files that are pre-allocated when the TOE is initially provisioned. When a file has reached its maximum capacity, the log is rolled over to the next file. This repeats in a FIFO order until all files have been filled, at which point the log is rolled back over to the first file, which is subsequently cleared to make room for the new audit data. Configuration of local audit storage retention does not affect the remote syslog[3] server.

7.2 SF2.Random Number Generation

SFRs	Rationale
FCS_RNG.1 Random Number Generator	The TOE implements a random bit generation service for the purpose of giving good non-predictable random numbers. VMKernel collects entropy from a EAL4 (or above) certified HSM as a noise source to provide random data to the RNG. OpenSSL then seeds its SP800-90A DRBG through the use of the getrandom() syscall. The OpenSSL SP800-90A DRBG is used for random number generation (server/client hello/random), RSA and ECC (ephemeral) key generation used for TLS (see section 7.7).

Table 21: Cryptographic Support SFRs

The TOE provides random numbers for the purpose of TLS through its OpenSSL library (v3.0.5). OpenSSL implements a SP800-90A DRBG that seeds itself from the getrandom() system call. getrandom() will not deliver any random number until enough entropy has been estimate in-kernel (based on the in-kernel entropy estimator).

The kernel obtains data itself from an HSM, which is outside of the TOE logical and physical boundary.

7.3 SF3.User Data Protection

SFRs	Rationale
FDP_IFC.1 Subset information flow control	A Guest VM cannot access the data of another Guest VM, or transfer data to another Guest VM other than through the mechanisms described in FDP_IFF.1 Simple security attribute (that is, virtual networking) when expressly enabled by an authorized user with administrator role. There are no known design or implementation flaws that permit the above mechanisms to be bypassed or defeated, or for data to be transferred through undocumented mechanisms. This claim does not apply to covert channels or architectural side-channels.
FDP_IFF.1 Simple security attribute	Sharing of data between VMs can be achieved using virtual networking. By default, data sharing is prohibited between VMs, even when virtual networking is not implemented.
FDP_RIP.1 Subset of residual information protection	Virtual machine memory deallocation acts just like it does in an operating system. The guest operating system frees a piece of physical memory by adding these memory page numbers to the guest free list, but the data of the “freed” memory may not be modified at all. As a result, when a particular piece of guest physical memory is freed, the mapped host physical memory will usually not change its state and only the guest free list will be changed. When a virtual machine requires memory, the vmkernel zeros each memory page out before being handed to the virtual machine. Normally, the virtual machine then has exclusive use of the memory page, and no other virtual machine can touch it or even detect it.

[3] Remote syslog is not in scope.

SFRs	Rationale
	Memory pages that are identical in two or more virtual machines are stored once in the host system's RAM, and each of the virtual machines has read-only access. Such shared pages are common, for example, if many virtual machines on the same host run the same OS. As soon as any one virtual machine attempts to modify a shared page, it gets its own private copy. Because shared memory pages are marked copy-on-write, it is impossible for one virtual machine to leak private information to another through this mechanism. Once a guest VM is deleted from the environment, its disk storage and memory are de-allocated under the same action and prior to allocation of the memory locations to a new VM. However, the memory and storage within the guest while present on the environment is under the control of the OS running on that VM To avoid information leaking among VMs, the hypervisor always writes zeroes to the host physical memory before assigning it to a VM. The VM is not aware that its memory pages are shared with other VMs or not. Because of this invisibility, sensitive information cannot be leaked from one VM to another. Moreover, sharing virtual memory between VMs is a feature that is disabled by default and should not be turned on.

Table 22: Guest Separation Rationale

A Guest VM cannot access the data of another Guest VM, or transfer data to another Guest VM other than through the mechanisms described in FDP_IFF.1 Simple security attribute (virtual networking) when expressly enabled by an authorized user with administrator role. There are no known design or implementation flaws that permit the above mechanisms to be bypassed or defeated, or for data to be transferred through undocumented mechanisms. This claim does not apply to covert channels or architectural side-channels.

Sharing of data between VMs can be achieved using virtual networking. By default, data sharing is prohibited between VMs, even when virtual networking is not implemented.

Virtual machine memory deallocation acts just like it does in an operating system. The guest operating system frees a piece of physical memory by adding these memory page numbers to the guest free list, but the data of the "freed" memory may not be modified at all. As a result, when a particular piece of guest physical memory is freed, the mapped host physical memory will usually not change its state and only the guest free list will be changed.

When a virtual machine requires memory, the vmkernel zeros each memory page out before being handed to the virtual machine. Normally, the virtual machine then has exclusive use of the memory page, and no other virtual machine can touch it or even detect it.

Memory pages that are identical in two or more virtual machines are stored once in the host system's RAM, and each of the virtual machines has read-only access. Such shared pages are common, for example, if many virtual machines on the same host run the same OS. As soon as any one virtual machine attempts to modify a shared page, it gets its own private copy. Because shared memory pages are marked copy-on-write, it is impossible for one virtual machine to leak private information to another through this mechanism.

Once a guest VM is deleted from the environment, its disk storage and memory are de-allocated under the same action and prior to allocation of the memory locations to a new VM. However, the memory and storage within the guest while present on the environment is under the control of the OS running on that VM.

To avoid information leaking among VMs, the hypervisor always writes zeroes to the host physical memory before assigning it to a VM.

The VM is not aware that its memory pages are shared with other VMs or not. Because of this invisibility, sensitive information cannot be leaked from one VM to another.

In the TOE User Data Protection of Guests are provided by hardware-based isolation mechanisms using "VT-X, EPT, AMD-V, RVI, and the IOMMU" to control direct access to the "CPU and PCI devices". The TOE clears the physical memory before allocation to Guests there by prevents any information leak from one Guest to another; this applies to both volatile and non-volatile storage.

7.4 SF4.Identification and Authentication

SFRs	Rationale
FDP_ACC.1 Subset access control	This SFR defines the access control requirements: only admins are allowed to use the TSF and only when providing the proper certificate or credentials.
FDP_ACF.1 Security attribute-based access control	Only admins can access the objects defined in FDP_ACC.1, namely: CA certificates, SAML validation rules, and all management functions.
FIA_AFL.1 Authentication Failure Handling	The guidance documentation provides information on the usage of the authentication lockout mechanism. Account lockout (by default 15 minutes) is supported through the VIM API after the configured number of login attempts has been reached.
FIA_SOS.1 Verification of secrets	For the authentication implemented in the evaluated configuration, the following password policy is implemented for the evaluated TOE configuration: • Password length of at least 16 characters • Must contain at least one uppercase letter • Must contain at least one lowercase letter • Must contain at least one number • Must contain at least one special character of the following: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “).
FIA_SOS.2 Generation of secrets	The TOE uses its random number generator provided by OpenSSL libcrypto to generate random numbers which will be used to session IDs, tickets or tokens with at least 100 bits of entropy, with total length 128 bits.
FIA_UAU.2 User authentication before any action	The TOE provides a user authentication that needs to be successfully passed before any actions can occur. Therefore, a combination of username and password is used.
FIA_UID.1 Timing of identification	Users must be successfully identified by the TOE before any action can occur. A username and password are required in order to be successfully identified.

Table 23: Identification and Authentication Rationale

Users must be successfully identified by the TOE for all remote interfaces to the TOE. A username and password are required in order to be successfully identified.

For the authentication implemented in the evaluated configuration, the following password policy is implemented for the evaluated TOE configuration:

- Password length of at least 16 characters
- Must contain at least one uppercase letter
- Must contain at least one lowercase letter
- Must contain at least one number
- Must contain at least one special character of the following: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”.

When connecting remotely, a username and password is transmitted securely over TLS. If the credentials match valid entries and have sufficient permissions, the user is successfully logged in to the TOE and given the administrator role.

The TOE's reverse proxy mechanism provides service endpoints through HTTPS (port 443), where the URL prefix defines the endpoint. The VIM API and vAPI are both reverse proxy endpoints (represented for example by the /sdk and /api URL prefixes, respectively). In order to access the endpoints, authentication is required. This is accomplished by either providing a username and password directly for authentication, or by providing a ticket or a token that was generated from using a username and password as authentication. No specific browser is required in order for the session IDs, tokens or tickets to be securely generated and provided to the user. An admin user can import certificate trust anchors that are used for token (JWT and SAML) validation.

For VIM APIs, after successful authentication, the TOE generates a 128-bit random number (session identifier or session ID) that is sent back to the user through TLS as an HTTP cookie. That session ID is used as a cookie for further authentication. It has a validity of 30 minutes, after which the user needs to be re-authenticated with its username and password.

For vAPI, a token is generated by the TOE (or a 3rd party), signed by the TOE and provide in the same way to the user through TLS. That token has a validity of 5 minutes regardless of what API call is requested. The token's signature is verified with each vAPI request. A new token has to be generated after its validity has expired.

Account lockout (15 minutes by default) is supported through the VIM API after the configured number of login attempts has been reached.

7.5 SF5.Security Management

SFRs	Rationale
FMT_SMF.1 Specification of Management	The TOE administrator role can remotely configure the following management functions in order to ensure the correct behavior of security mechanisms: • Ability to enable/disable the transfer of data between Guest VMs • Ability to control Guest Networking (e.g. configure virtual networking components to connect VMs to each other and to physical networks) • Ability to control virtual device backings • Ability to configure user accounts with administrator role • Ability to manage VMs • Ability to configure the limit of unsuccessful login attempts before temporary account lockout • Ability to enable/disable auditing • Ability to remotely fetch audit records • Ability to configure remote connection inactivity timeout and define global password complexity.
FMT_SMR.1 Security Roles	The sole role defined for the evaluated configuration is administrator. The administrator role has access to perform all operations on the TOE.
FMT_MOF.1 Management of security functions behavior	All management functions are restricted to the administrator role since the administrator role is the only role supported by the TOE.
FMT_MSA.1 Management of security attributes	The following security attributes can only be managed by the administrator role on the TOE: <u>Guest VM settings:</u> The administrator role has the possibility to add, remove or modify the hardware or virtual device resources in order to maintain the TOE security and ensure sufficient system resources for guest operating systems. <u>Host settings:</u>

SFRs	Rationale
	Regarding information flow control, the administrator role has the possibility to allow, separate, or deny communication between different Guest VMs of a host system. Thereby it can be ensured the no assets leak between the different Guest VMs. Network settings: The administrator role can control the network settings in order to allow, separate or deny network traffic between the different Guest VMs. Thereby it can be ensured that no assets leak between the different Guest VMs. Guest VM peripheral settings: The administrator role has the possibility to allow or restrict the ability of Guest VM to access connected peripheral in order to avoid data leakage. Administrator Role Password: The administrator role has the possibility to set or change the administrator role password regarding the underlying password policy.
FMT_MSA.3 Static attribute initialisation	The administrator role is able to specify alternative initial values to override the default values.
FDP_IFF.1 Simple security attribute	Unless the administrator role explicity allows data sharing between the VMs, there is no implicit data sharing between the guests.
FDP_ITC.1 Import of user data without security attributes	The administrator role is the only role allowed to import CA certificates, SAML validation rules, and access any management function: the TSF needs to import an RSA certificate (public key) in order to verify the cryptographic signature of tokens send from an external entity to the TOE. Tokens's MAC are generated/verified through FCS_COP.1/HMAC.

Table 24: Security Management Rationale

The TOE has one administrator role that can perform all management functions defined in FMT_SMF.1. No other user can perform management functions. The TOE has several management interfaces that can be used for that purpose; the management functions supported by the TOE and the management interfaces on which those functions can be performed are shown in Table 15: Remote Management Functions. The ESXi Host Client is a GUI front-end for the VIM API; all functions executed on this interface call underlying VIM API functions that can also be invoked directly.

The TSF enforces separation of data sharing between Guest VMs and between management and operational networks. By default, data sharing between Guest VMs is prohibited per FDP_IFF.1, but this can be administratively enabled through the use of virtual networking. To ensure that the management and operational networks of the TOE remain separated, the administrator role can configure separate networks for management and operation. This can either be done through physical means, where separate NICs are used for each network, or through logical means, where the management and operational networks are on separate VLANs. This ensures that communication on the management network does not occur on the same network as operational traffic. Management traffic over the management network, whether physical or logical, is always handled through trusted channels that use TLS or TLS/HTTPS.

7.6 SF6.Protection of the TSF

SFRs	Rationale
FPT_VIV_EXT.1 VMM Isolation from VMs	There are no known design or implementation flaws that bypass or defeat VM isolation, or permit software running in a VM to degrade or disrupt the functioning of other VMs, the VMM, or the platform. The Guest separation security function is provided by the TOE. The TOE ensures that each virtual machine is isolated from any other virtual machines co-existing on the TOE. This isolation is provided at the virtualization layer of the TOE. The vmkernel of the TOE ensures that virtual machines are unable to directly interact with other virtual machines yet still allow for physical resources to be shared among the existing virtual machines. The ESXi VMKernel provides a virtual hardware environment which controls the host hardware and schedules the allocation of the underlying physical resources associated with each virtual machine. Each virtual machine runs its own operating system and applications: they cannot communicate with each other in unacceptable or unauthorized ways. The following mechanisms ensure this: • Shared memory access: The memory allocation mechanisms prevent the sharing of writable memory. Each VM is assigned memory that belongs exclusively to it. • Read-only memory: For efficiency, multiple VMs may use the same memory pages, and in these cases, the memory locations are shared, but in a read-only mode. This effectively saves memory without providing a communication channel between VMs. • Communication between VMs through standard network connections can be permitted or prevented as desired. These standard networking mechanisms are similar to those used to connect separate physical machines. Each virtual machine appears to run on its own processor, fully isolated from other virtual machines with its own registers, buffers, and other control structures. Most instructions are directly executed on the physical processor, allowing compute-intensive workloads to run at near-native speed. Memory appears contiguous to each virtual machine, but instead, noncontiguous physical pages are remapped efficiently and presented transparently to each virtual machine. The ESXi VMKernel mediates all access to physical hardware resources, including CPU, memory, and I/O devices, ensuring that VMs cannot circumvent this level of isolation and gain access to the physical hardware. A VM can only detect the virtual devices made available to it: - VM storage: Virtual machines use virtual disks to store OS, program files, and other data. Each VM is given its own virtual disk file that is not visible to other VMs. Virtual disks are accessed using virtual SCSI controllers. Virtual disk files reside on Virtual Machine File System (VMFS) or supported network based datastores. The virtual disk appears to the VM as if it is a SCSI drive attached to a physical SCSI controller. VMware supports parallel SCSI, iSCSI, network, NFS, Fibre Channel, or FcoE based storage. The physical access method is completely transparent to the guest OS and applications residing within a VM. - Removable storage: ESXi also provides virtualization of removable storage by enabling access to the logical CD, or remotely via the VIM API.

SFRs	Rationale
FPT_HCL_EXT.1 Hypercall Controls	Vendor documentation provides information on the hypercall interfaces, including all applicable functions, parameters, legal values, configuration settings, and how the functions are called.
FPT_RDM_EXT.1 Removable Devices and Media	During regular operation of the TOE, the administrator role controls access to removable media, whether physical or virtual, by means of explicit configuration to permit access. Removable physical media only applies to USB storage devices. Removable virtual media applies to virtual floppies and virtual optical device images (e.g. ISO images). ISO images are presented read-only (no write access is permitted).

Table 25: Protection of the TSF SFRs

There are no known design or implementation flaws that bypass or defeat VM isolation, or permit software running in a VM to degrade or disrupt the functioning of other VMs, the VMM, or the platform. The Guest separation security function is provided by the TOE. The TOE ensures that each virtual machine is isolated from any other virtual machines co-existing on the TOE. This isolation is provided at the virtualization layer of the TOE. The vmkernel of the TOE ensures that virtual machines are unable to directly interact with other virtual machines yet still allow for physical resources to be shared among the existing virtual machines.

The ESXi VMKernel provides a virtual hardware environment which controls the host hardware and schedules the allocation of the underlying physical resources associated with each virtual machine. Each virtual machine runs its own operating system and applications: they cannot communicate with each other in unacceptable or unauthorized ways. The following mechanisms ensure this:

- Shared memory access: The memory allocation mechanisms prevent the sharing of writable memory. Each VM is assigned memory that belongs exclusively to it.
- Read-only memory: For efficiency, multiple VMs may use the same memory pages, and in these cases, the memory locations are shared, but in a read-only mode. This effectively saves memory without providing a communication channel between VMs.
- Communication between VMs through standard network connections can be permitted or prevented as desired. These standard networking mechanisms are similar to those used to connect separate physical machines.

Each virtual machine appears to run on its own processor, fully isolated from other virtual machines with its own registers, buffers, and other control structures. Most instructions are directly executed on the physical processor, allowing compute-intensive workloads to run at near-native speed. Memory appears contiguous to each virtual machine, but instead, noncontiguous physical pages are remapped efficiently and presented transparently to each virtual machine.

The ESXi VMKernel mediates all access to physical hardware resources, including CPU, memory, and I/O devices, ensuring that VMs cannot circumvent this level of isolation and gain access to the physical hardware. A VM can only detect the virtual devices made available to it:

- VM storage: Virtual machines use virtual disks to store OS, program files, and other data. Each VM is given its own virtual disk file that is not visible to other VMs. Virtual disks are accessed using virtual SCSI controllers. Virtual disk files reside on Virtual Machine File System (VMFS) or supported network based datastores. The virtual disk appears to the VM as if it is a SCSI drive attached to a physical SCSI controller. VMware supports parallel SCSI, iSCSI, network, NFS, Fibre Channel, or FcoE based storage. The physical access method is completely transparent to the guest OS and applications residing within a VM.
- Removable storage: ESXi also provides virtualization of removable storage by enabling access to the logical CD or remotely via the VIM API.

Vendor documentation provides information on the hypercall interfaces, including all applicable functions, parameters, legal values, configuration settings, and how the functions are called.

During regular operation of the TOE, the administrator role controls access to removable media, whether physical or virtual, by means of explicit configuration to permit access. Removable physical media only applies to USB storage devices. Removable virtual media applies to virtual floppies and virtual optical device images (e.g. ISO images). ISO images are presented read-only (no write access is permitted).

Furthermore the TOE protects the channel between the different TOE parts and external IT components by a TLS connection.

There are no known design or implementation flaws that bypass or defeat VM isolation, or permit software running in a VM to degrade or disrupt the functioning of other VMs, the VMM, or the platform. Specifically, the TOE uses Intel's VT-x and VT-d hardware virtualization support to ensure that VMs are isolated from each other and the TOE, and cannot interfere with a VM's device access. VMware ESXi does not provide a mechanism or ability for guest software to directly call platform APIs or to directly generate physical System Management Interrupts (SMIs). System Management Mode (SMM) and SMIs are both virtualized, and thus handled by the virtual firmware (in guest) and not by the physical hardware. Virtual SMIs are not correlated with physical SMIs. In the evaluated configuration, no platform firmware, I/O ports, or MMIO registers are directly mapped into the address space or I/O space of the guest VM.

While ESXi does contain a host mechanism for updating microcode on the CPU, any attempt by guest software to update the microcode via the virtualized MSR (Model-Specific Register) will be logged and then dropped.

A proprietary annex to this ST provides information on the hypercall interfaces, including all applicable functions, parameters, legal values, configuration settings, and how the functions are called. The administrstor role on the TOE has the ability to disable these hypercall functions.

The following virtual devices are within the scope of evaluation:

- Network controllers:
  - Vlance
  - vmxnet2
  - e1000
  - vmxnet3
- Storage controllers:
  - IDE
  - AHCI (SATA)
  - BusLogic (SCSI)
  - LSILogic (SCSI)
  - PVSCSI (SCSI)
  - NVMe
  - USB controllers
  - UHCI (USB 1.0)
  - EHCI (USB 2.0)
  - XHCI (USB 3.0)
  - Sound

- Video graphics controller
- VMware SVGA II (not removable)
- “PC” devices
- Serial
- Parallel
- PS/2 (Keyboard, Mouse) (not removable).

Most devices are exposed as PCI devices where presence of appropriate PCI identifying information determines presence of a device. Some devices also have IO ports, either well known or relative to a base.

Parameters passed from Guest VMs to virtual device interfaces are thoroughly validated and all illegal values (as specified in the TSS) are rejected. Additionally, parameters passed from Guest VMs to virtual device interfaces are not able to degrade or disrupt the functioning of other VMs, the VMM, or the Platform. Thorough testing and architectural design reviews have been conducted to ensure the accuracy of these claims, and there are no known design or implementation flaws that bypass or defeat the security of the virtual device interfaces.

The ESXi VMM uses VT-x to reduce the use of binary translation. It also uses EPT to eliminate the need for shadow page tables.

The TOE leverages the capabilities of address-space randomization, memory execution protection, and stack buffer overflow protection to provide execution environment-based vulnerability mitigation mechanisms. These mechanisms assist in prevention of unintended machine code execution within the environment.

During regular operation of the TOE, the administrator role controls access to removable media, whether physical or virtual, by means of explicit configuration to permit access. Removable physical media applies to USB storage devices. Removable virtual media applies to virtual floppies and virtual optical device images (e.g. ISO images). ISO images are presented read-only (no write access is permitted).

Audit records are emitted when a virtual machine is associated with a particular media, and when the association is removed.

7.7 SF7.Cryptographic Support and Encrypted Channels

SFR	Rationale
FCS_CKM.1/RSA Cryptographic Key Generation (RSA)	RSA key generation is used for the protection of TSF data communications by using a TLSv1.2 channel.
FCS_CKM.1/TLS Cryptographic Key Generation (TLS)	TLS Cryptographic Key Generation is used for protection of TSF data communicatio

<table>
<tr><th>SFR</th><th>Rationale</th></tr>
<tr><td></td><td>ns by generation of additional keys through a key derivation function (KDF) based on a key distributed through FCS_CKM.2/TLS.

The key derivation function (KDF) is the TLS 1.2 PRF-SHA384.</td></tr>
<tr><td>FCS_CKM.2/TLS Cryptographic Key Distribution (TLS)

FCS_CKM.2.1/TLS The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method [see Table 12] that meets the following: [see Table 12].

<table><tr><th>Operation/Purpose</th><th>Algorithms / Cipher Suite</th><th>Key Lengths</th></tr><tr><td>Key Agreement</td><td>Ephemeral elliptic curve DH key exchange supports the P-256, P-384, and the P-521 curves.</td><td>256 Bits (P-256
384 Bits (P-384
521 Bits (P-521</td></tr></table>Table 12: Cryptographic Key Exchange

Application Note: Key Distribution refers to the negotiation of the TLS shared secret.</td><td>TLS session key generation is used for the protection of TSF data communications by using a TLSv1.2 channel. Session keys are derived from the negotiated shared secret for encryption, decryption and authentication (integrity protection) of transferred data in the TLS channel.</td></tr>
<tr><td>FCS_CKM.4 Cryptographic Key Destruction</td><td>In the TOE both user world and ESXi Critical Security Parameters (cryptographic keys) are zeroised by the ESXi VMKernel when volatile memory pages are allocated.</td></tr>
<tr><td>FCS_COP.1/AES Cryptographic Operation (Symmetric Encryption/Decryption)</td><td>The TOE uses AES-GCM to provide packet confidentiality</td></tr>
</table>

SFR	Rationale
	and authentication through TLSv1.2.
FCS_COP.1/RSA Cryptographic Operation (Signature Generation/Verification)	The TOE provides RSA signature verification capabilities to provide ECDH parameters verification during the TLSv1.2 handshake.
FCS_COP.1/ECDHE Cryptographic Operation (Shared Secret Computation) (Key Derivation Function)	The TOE provides EC Diffie-Hellman ephemeral shared secret computation to establish a master secret between the TOE and a trusted external IT entity.
FCS_COP.1/PRF Cryptographic Operation (Key Derivation Function)	The TLS protocol relies on a PRF to derive all necessary keying material in order to establish a trusted TLS channel between the TOE and an external IT entity.
FCS_COP.1/HMAC Cryptographic Operation (Keyed Hash-based Message Authentication Code)	The TOE uses HMAC-SHA-256 to create and verify the integrity of MACs of tokens sent across internal trusted channels.
FTP_ITC.1 Inter-TSF trusted channel	The TOE implements the TLSv1.2 protocol to

SFR	Rationale
	establish trusted channels between itself and external IT entities. Externel IT entities are allowed to initiate TLS connections where the TOE acts as a server. This trusted channel (TLS/HTTPS) is used to protect TSF data.

Table 26: Encrypted Channels Rationale

The TOE uses cryptography to secure data in transit between itself and its operational environment.

All TOE cryptographic services are implemented by the OpenSSL cryptographic library. The TOE uses VMware OpenSSL FIPS Provider 3.0.0. The cryptographic algorithms supplied by the TOE are NIST-approved.

The following table identifies the cryptographic algorithms used by the TSF, the associated standards to which they conform.

Cipher Suite from RFC5289 (TLSv1.2)	Algorithms	Standard
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	AES	FIPS Pub 197
	GCM	SP800-38D RFC5288 RFC5116
	RSA	FIPS Pub 186-4
	Ephemeral elliptic curve Diffie-Hellman	RFC5246 RFC8422 SEC2
	TLSv1.2 PRF with SHA384	RFC5246

Table 27: Algorithms / Ciphersuite

Functions	Libraries	Standards
TLS Key Derivation Function (PRF-SHA384)	OpenSSL	FIPS PUB 186-4 RFC2104 RFC5246
RSA key pair generation	OpenSSL	FIPS PUB 186-4

ECC based key establishment (ECDHE according to NIST curves P-256, P-384 and P-521)	OpenSSL	RFC8422 RFC5246 SEC2
RSA signature generation/verification (3072 bits)	OpenSSL	FIPS PUB 186-4
(HMAC-)SHA-384	OpenSSL	FIPS PUB 198-1 FIPS PUB 180-4
AES-GCM (256 bits)	OpenSSL	SP 800-38D RFC5288 RFC5116
CTR_DRBG(AES)	OpenSSL	SP 800-90A

Table 28: Cryptographic Functions

The TOE generates RSA keys of size 3072 bits, and utilizes NIST curves P-256, P-384 and P-521 for ECDHE key negotiation in support of TLSv1.2. These ECC primitives and algorithms are implemented by OpenSSL in support of the ECDHE key establishment schemes that are used for TLS communications. To ensure sufficient key strength, the TOE implements an SP800-90A DRBG random number generator, using the AES-CTR_DRBG (see SF2.Random Number Generation).

The TOE's HMAC functions support the HMAC-SHA384 algorithm with a key of 384 bits, and a block size of 1024 bits. HMAC is used to support the TLS v1.2 PRF as defined in RFC5246.

The TOE uses TLS v1.2 for client and server communications. In the case where the TOE acts as a TLS server, all other TLS versions are rejected. The TLS client or server implementation supports the one TLS cipher suite TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 in the TOE's evaluated configuration.

If the TOE is prompted with a session using an old SSL or TLS version, the connection is rejected if the peer cannot use a newer version. During the server key exchange, the key agreement parameters are passed. These include a temporary key pair and public keys. A shared secret is computed from these parameters.

The TOE implements TLS as a server for remote administration using HTTPS. The TOE's implementation of HTTPS conforms to RFC 2818.

The TLS channel is used for HTTPS (for reverse proxy communications) to protect TSF Data. The SFR is giving an overview about the different security functional requirements that are used to define the TLS channel. In general the TOE permits the TSF or another trusted IT product to initiate communication via the trusted channel which is used for all security functions specified in the ST that interact with remote trusted IT systems.

In the TOE both user world and ESXi Critical Security Parameters (cryptographic keys) are zeroised by the VMKernel when volatile memory pages are allocated.

7.8 TOE Summary Specification Rationale

SFR	SF1	SF2	SF3	SF4	SF5	SF6	SF7
FAU_GEN.1 Audit Data Generation	X						
FAU_SAR.1 Audit Review	X						
FAU_STG.1 Protected Audit Trail Storage	X						
FCS_RNG.1 Random Number Generator		X					
FDP_ACC.1 Subset access control				X			

SFR	SF1	SF2	SF3	SF4	SF5	SF6	SF7
FDP_ACF.1 Security attribute-based access control				X			
FDP_IFC.1 Subset information flow control			X				
FDP_IFF.1 Simple security attribute			X		X		
FDP_RIP.1 Subset of residual information protection			X				
FDP_ITC.1 Import of user data without security attributes					X		
FIA_AFL.1 Authentication Failure Handling				X			
FIA_SOS.1 Verification of secrets				X			
FIA_SOS.2 Generation of secrets				X			
FIA_UAU.2 User authentication before any action				X			
FIA_UID.1 Timing of identification				X			
FMT_SMR.1 Security Roles					X		
FMT_SMF.1 Specification of Management					X		
FMT_MOF.1 Management of security functions behavior					X		
FMT_MSA.1 Management of security attributes					X		
FMT_MSA.3 Static attribute initialisation					X		
FPT_VIV_EXT.1 VMM Isolation from VMs						X	
FPT_HCL_EXT.1 Hypercall Controls						X	
FPT_RDM_EXT.1 Removable Devices and Media						X	
FTP_ITC.1 Inter-TSF trusted channel							X
FCS_CKM.1/RSA Cryptographic Key Generation (RSA)							X
FCS_CKM.1/TLS Cryptographic Key Generation (TLS)							X
FCS_CKM.2/TLS Cryptographic Key Distribution (TLS)							X
FCS_CKM.4 Cryptographic Key Destruction							X
FCS_COP.1/AES Cryptographic Operation (Symmetric Encryption/Decryption)							X
FCS_COP.1/RSA Cryptographic Operation (Signature Generation/Verification)							X

SFR	SF1	SF2	SF3	SF4	SF5	SF6	SF7
FCS_COP.1/ECDHE Cryptographic Operation (Shared Secret Computation)							X
FCS_COP.1/PRF Cryptographic Operation (Key Derivation Function)							X
FCS_COP.1/HMAC Cryptographic Operation (Keyed Hash-based Message Authentication Code)							X

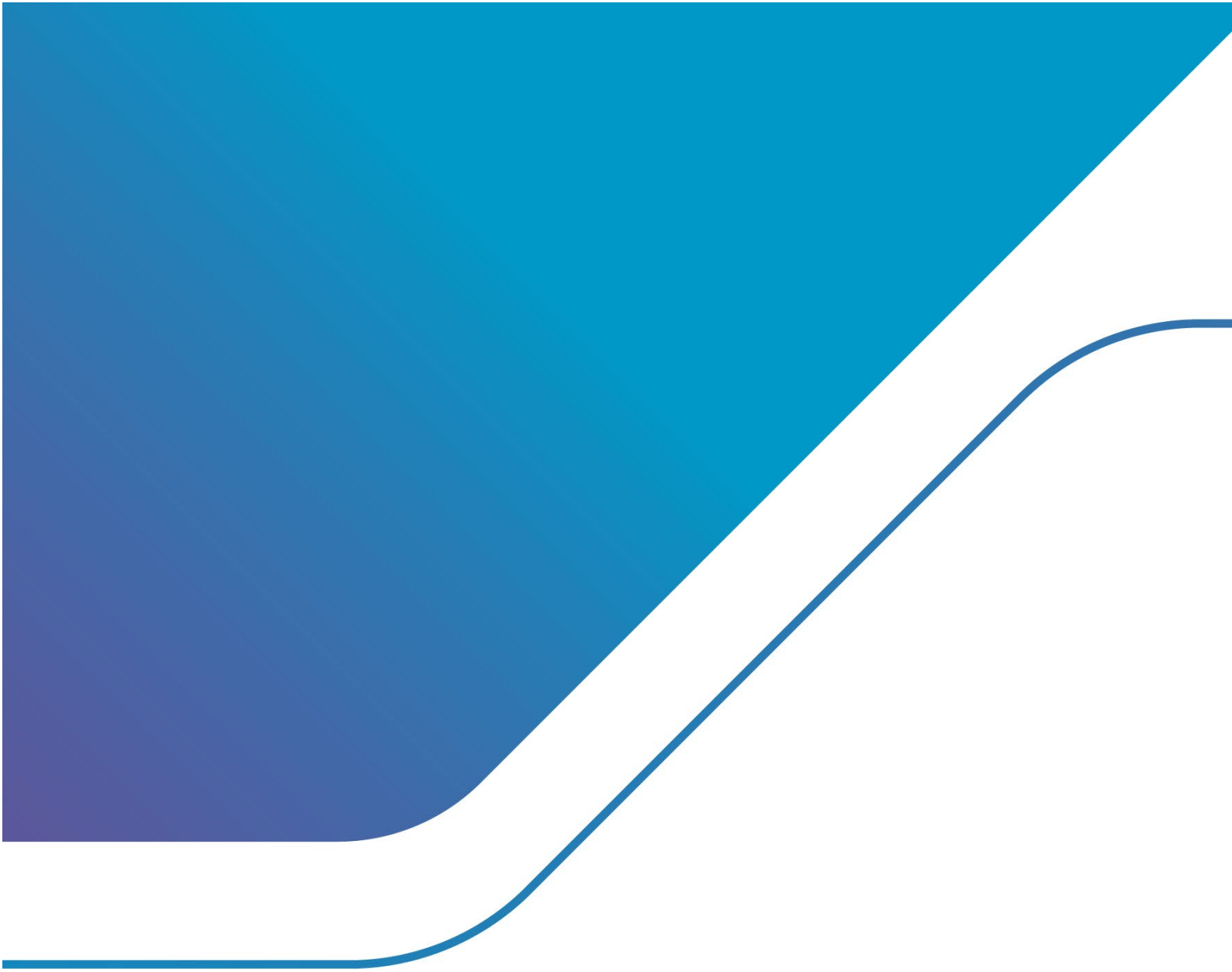
Table 29: TSS Rationale

8 Acronyms & Definitions

Acronym	Definition
ACL	Access Control Lists
ADOM	Administrative Domain
AMD-V	Advanced Micro Devices Virtualization (Technology)
API	Application Programming Interface
BSI	Bundesamt für Sicherheit in der Informationstechnik
CBC	Cipher Block Chaining
CC	Common Criteria
CEM	Common Evaluation Methodology
CSP	Critical Security Parameters
DRNG	Deterministic Random Number Generator
EPT	Extended Page Tables
ESXi	Elastic Sky X (i)
FCoE	Fibre Channel over Ethernet
FIPS	Federal Information Processing Standard
gRPC	global Remote Procedure Call
HSM	Hardware Security Module
HTML	HyperText Markup Language
HTTPS	Hypertext Transfer Protocol Secure
IDL	Interface Definition Language
IOMMU	Input/Output Memory Management Unit
iSCSI	Internet Small Computer System Interface
JWT	JavaScript Object Notation (JSON) Web Token
LDAP	Lightweight Directory Access Protocol
MKS	Mouse-Keyboard-Screen
NDPP	Network Device Protection Profile
NFC	Network File Copy
NFS	Network File Share
NTP	Network Time Protocol
OSP	Organizational Security Policy
POSIX	Portable Operating System Interface
PP	Protection Profile
PSC	Platform Services Controller
RAID	Redundant Array of Inexpensive Disks
RBG	Random Bit Generator
RVI	Rapid Virtualization Indexing
SAML	Security Assertion Markup Language
SAR	Security Assurance Requirement
SCSI	Small Computer System Interface

Acronym	Definition
SFP	Security Functional Policies
SFR	Security Functional Requirement
SSO	Single Sign-On
ST	Security Target
TLB	Translation Lookaside Buffer
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functionality
TPM	Trusted Platform Module
vCLI	vSphere Command-Line Interface
vCS	vCenter Server
vCSA	vCenter Server Appliance
VDOM	Virtual Domain
VIM	Virtual Infrastructure Manager
VM	Virtual Machine
VMDK	Virtual Machine Disk
VMM	Virtual Machine Manager
VMRC	VMware Remote Console
VMX	Virtual Machine eXecutable
VPP	Virtualization Protection Profile
VS	Virtualization Software
VT-X	Intel Virtualization Technology

Table 30: List of Acronyms



vmware
by Broadcom