Red Hat Enterprise Linux, Version 6.2 on 32 bit x86 Architecture

Version:	**2.6**
Status:	**Released**
Last Update:	**2014-08-12**
Classification:	**Public**

Trademarks

Red Hat and the Red Hat logo are trademarks or registered trademarks of Red Hat, Inc. in the United States, other countries, or both.

atsec is a trademark of atsec information security GmbH

Linux is a registered trademark of Linus Torvalds.

UNIX is a registered trademark of The Open Group in the United States and other countries.

IBM, IBM logo, bladecenter, eServer, iSeries, OS/400, , POWER3, POWER4, POWER4+, pSeries, System p, POWER5, POWER5+, POWER6, POWER6+, POWER7, POWER7+, System x, System z, S390, xSeries, zSeries, zArchitecture, and z/VM are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Intel, Xeon, and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

This document is based in parts on the Red Hat Enterprise Linux Version 6.2 Security Target, Copyright © 2012 by Red Hat, Inc. and atsec information security corp.

Legal Notice

This document is provided AS IS with no express or implied warranties. Use the information in this document at your own risk.

This document may be reproduced or distributed in any form without prior permission provided the copyright notice is retained on all copies. Modified versions of this document may be freely distributed provided that they are clearly identified as such, and this copyright is included intact.

Revision History

Revision	Date	Author(s)	Changes to Previous Revision
2.0	2013-10-08	Stephan Mueller	Initial version of ST
2.1	2013-11-07	Stephan Mueller	Evaluator comments
2.2	2013-12-20	Stephan Mueller	Editorial changes
2.3	2014-02-06	Stephan Mueller	Remove init from partial RELRO claim
2.4	2014-03-17	Stephan Mueller	Refer to 6.2.z packages
2.5	2014-03-26	Stephan Mueller	Editorial fixes
2.6	2014-08-12	Stephan Mueller	Changes based on BSI comments

Table of Contents

List of Tables

1 Introduction

1.1 Security Target Identification

Title:	Red Hat Enterprise Linux, Version 6.2 on 32 bit x86 Architecture
Version:	2.6
Status:	Released
Date:	2014-08-12
Sponsor:	Red Hat, Inc.
Developer:	Red Hat, Inc.
Certification Body:	BSI
Certification ID:	BSI-DSZ-CC-0924
Keywords:	Security Target, Common Criteria, Linux Distribution, Embedded Linux

1.2 TOE Identification

The TOE is Red Hat Enterprise Linux on 32 bit x86 Architecture Version 6.2.

1.3 TOE Type

The TOE type is a Linux-based general-purpose operating system.

1.4 TOE Overview

1.4.1 Configurations defined with this ST

This security target documents the security characteristics of the Red Hat Enterprise Linux distribution (abbreviated with RHEL throughout this document).

1.4.2 Overview description

Red Hat Enterprise Linux is a highly-configurable Linux-based operating system which has been developed to provide a good level of security as required in commercial environments. It also meets all requirements of the Operating System protection profile [OSPP] together with the following extended packages specified for the OSPP:

- Extended package for Audit

1.4.3 Compliance with STIG and other standards

The evaluated configuration draws from many standards, including the US STIG standard. It is possible to achieve full compliance with STIG in the evaluated configuration. However, to prevent violation of other configuration standards, the evaluated configuration does not claim full compliance with STIG.

1.4.4 Required Hardware and Software

The following hardware / firmware allows the installation of the TOE:

- Northrop Grumman Payload Control Element (PCE) Server 309-C20213

The covered system is based on an Intel x86 32 bit CPU.

1.4.5 Intended Method of Use

1.4.5.1 General-purpose computing environment

The TOE is a Linux-based multi-user multi-tasking operating system. The TOE may provide services to several users at the same time. After successful login, the users have access to a general computing environment, allowing the start-up of user applications, issuing user commands at shell level, creating and accessing files. The TOE provides adequate mechanisms to separate the users and protect their data. Privileged commands are restricted to administrative users.

The TOE is intended to operate in a networked environment with other instantiations of the TOE as well as other well-behaved peer systems operating within the same management domain. All those systems need to be configured in accordance with a defined common security policy.

It is assumed that responsibility for the safeguarding of the user data protected by the TOE can be delegated to human users of the TOE if such users are allowed to log on and spawn processes on their behalf. All user data is under the control of the TOE. The user data is stored in named objects which are covered by access rights.

The TOE enforces controls such that access to named objects can only take place in accordance with the access restrictions placed on that object by its owner, and by administrative users. Ownership of named objects may be transferred under the control of the access control policies implemented by RHEL.

Discretionary access rights (e.g. read, write, execute) can be assigned to named objects with respect to subjects identified with their UID, GID and supplemental GIDs. Once a subject is granted access to an object, the content of that object may be used freely to influence other objects accessible to this subject.

1.4.5.2 Operating environment

The TOE permits one or more processors and attached peripheral and storage devices to be used by multiple applications assigned to different UIDs to perform a variety of functions requiring controlled shared access to the data stored on the system. With different UIDs proper access restrictions to resources assigned to processes can be enforced using the access control mechanisms provided by the TOE. Such installations and usage scenarios are typical for systems accessed by processes or users local to, or with otherwise protected access to, the computer system.

Note: The TOE provides the platform for installing and running arbitrary services. These additional services are not part of the TOE. The TOE is solely the operating system which provides the runtime environment for such services.

All human users, if existent, as well as all services offered by RHEL are assigned unique user identifiers within the single host system that forms the TOE. This user identifier is used together with the attributes and roles assigned to the user identifier as the basis for access control decisions. The TOE authenticates the claimed identity of the user before allowing the user to perform any further actions. Services may be spawned by the TOE without the need for user-interaction. The TOE internally maintains a set of identifiers associated with processes, which are derived from the unique user identifier upon login of the user or from the configured user identifier for a TOE-spawned service. Some of those identifiers may change during the execution of the process according to a policy implemented by the TOE.

1.4.6 Major Security Features

The primary security features of the TOE are specified as part of the logical boundary description.

These primary security features are supported by domain separation and reference mediation, which ensure that the features are always invoked and cannot be bypassed.

1.5 TOE Description

1.5.1 Introduction

Red Hat Enterprise Linux is a general purpose, multi-user, multi-tasking Linux based operating system. It provides a platform for a variety of applications.

The RHEL evaluation covers a potentially distributed network of systems running the evaluated versions and configurations of RHEL as well as other peer systems operating within the same management domain. The hardware platforms selected for the evaluation consist of machines which are available when the evaluation has completed and to remain available for a substantial period of time afterwards.

The TOE Security Functions (TSF) consist of functions of RHEL that run in kernel mode plus some trusted processes. These are the functions that enforce the security policy as defined in this Security Target. Tools and commands executed in user mode that are used by an administrative user need also to be trusted to manage the system in a secure way. But as with other operating system evaluations they are not considered to be part of this TSF.

The hardware, the BootProm firmware and potentially other firmware layers between the hardware and the TOE are considered to be part of the TOE environment.

The TOE includes standard networking applications, such as SSH.

System administration tools include the standard command line tools. A graphical user interface for system administration or any other operation is not included in the evaluated configuration.

In addition to providing a general-purpose computing environment, the TOE provides the following mechanisms allowing a wider deployment:

- Audit: The TOE can be deployed as an audit server that receives audit logs from other TOE instances. These audit logs are stored locally. The TOE provides search and review facilities to authorized administrators for all audit logs.

The TOE environment also includes applications that are not evaluated, but are used as unprivileged tools to access public system services. For example a network server using a port above 1024 may be used as a normal application running without root privileges on top of the TOE. The additional documentation specific for the evaluated configuration provides guidance how to set up such applications on the TOE in a secure way.

1.5.2 TOE boundaries

1.5.2.1 Physical

The Target of Evaluation is based on the following system software:

- Red Hat Enterprise Linux in the above mentioned version
- The following patches from EUS 6.2.z:
    - Fix RHSA-2013:1519-1

- Fix RHBA-2012:0338
- Fix RHEA-2012:0065
- Fix RHBA-2012:0134
- Fix RHBA-2012:1319-2
- Fix RHBA-2012:0337
- Fix RHBA-2012:0344
- Fix RHBA-2012:0339
- Fix RHSA-2012:0699-01
- Fix RHEA-2012:0486
- Fix RHSA-2012:0451

The TOE and its documentation are supplied on ISO images distributed via the Red Hat Network. The TOE includes a package holding the additional user and administrator documentation.

In addition to the installation media, the following documentation is provided:

- Evaluated Configuration Guide
- Manual pages for all applications, configuration files and system calls

The hardware applicable to the evaluated configuration is listed above. The analysis of the hardware capabilities as well as the firmware functionality is covered by this evaluation to the extent that the following capabilities supporting the security functionality are analyzed and tested:

- Memory separation capability
- Unavailability of privileged processor states to untrusted user code (like the hypervisor state or the SMM)
- Full testing of the security functionality on all listed hardware systems

1.5.2.2 Logical

The primary security features of the TOE are:

- Auditing: The Lightweight Audit Framework (LAF) is designed to be an audit system making Linux compliant with the requirements from Common Criteria. LAF is able to intercept all system calls as well as retrieving audit log entries from privileged user space applications. The subsystem allows configuring the events to be actually audited from the set of all events that are possible to be audited.
- Cryptographic support: The TOE provides cryptographically secured communication channels as well as cryptographic primitives that unprivileged users can utilize for unspecified purposes. The TOE provides cryptographically secured communication to allow remote entities to log into the TOE. For interactive usage, the SSHv2 protocol is provided.
- Packet filter: The TOE provides a stateless and stateful packet filter for regular IP-based communication. Layer 3 (IP) and layer 4 (TCP, UDP, ICMP) network protocols can be controlled using this packet filter. Ethernet frames routed through bridges are controlled by a separate packet filter which implements a stateless packet filter for the TCP/IP protocol family.
- Identification and Authentication: User identification and authentication in the TOE includes all forms of interactive login (e.g. using the SSH protocol or log in at the local console) as well as identity changes through the su or sudo command. These all rely on explicit authentication information provided interactively by a user.

- Discretionary Access Control: DAC allows owners of named objects to control the access permissions to these objects. These owners can permit or deny access for other users based on the configured permission settings. The DAC mechanism is also used to ensure that untrusted users cannot tamper with the TOE mechanisms.
- Confidentiality protected data storage: Using dm_crypt, the Linux operating system offers administrators and users cryptographically protected block device storage space. Only with the passphrase can the session key used for encryption or decryption be obtained and used. Any data stored on the block devices protected by dm_crypt is encrypted and cannot be accessed even when the TOE is not operational unless the TOE is operational and the block device session key is unlocked.
- Security Management: The security management facilities provided by the TOE are usable by authorized users and/or authorized administrators to modify the configuration of TSF.
- Protection mechanisms: The TOE provides mechanisms to prevent common buffer overflow and similar attacks. These mechanisms are used for the TSF and are available to untrusted code.

The TOE provides many more functions and mechanisms. The evaluation ensures that all these additional functions do not interfere with the above mentioned security mechanisms in the evaluated configuration. Mechanisms and functions that would interfere with the operation of the security functions are disallowed in the evaluated configuration and the Evaluation Configuration Guide provides instructions to the administrator on how to disable them. Note: TOE mechanism which provide additional restrictions to the above claimed security functions are allowed in the evaluated configuration. For example, Linux Containers are provided with the TOE and permitted in the evaluated configuration even though they have not been subject to this evaluation. Linux Containers provide further restrictions on, for example, the security function of discretionary access control mechanism for IPC objects and therefore cannot breach the security functionality. The following table enumerates mechanisms that are provided with the TOE but which are excluded from the evaluation:

Functions	**Exclusion discussion**
SELinux	SELinux provides additional access control mechanisms which are uncovered in this evaluation.
KVM	The virtualization support offered by KVM is unclaimed as the underlying hardware lacks respective support.
eCryptFS	eCryptFS are not allowed to be used in the evaluated configuration. The encryption capability provided with this file system is therefore unavailable to any user.
SMACK	The mandatory access control functionality offered by the SMACK LSM is not assessed by the evaluation and disabled in the evaluated configuration.
IPSEC	IPSEC is allowed to be used with the TOE, but the ST does not make any claims about the cryptographic aspects of IPSEC.
SSL / TLS tunnels	The TOE provides the stunnel application which can be used to establish SSL and TLS tunnels with remote peers. This application however was excluded from evaluation assessment.

Functions	Exclusion discussion
Linux Containers	As mentioned above, the Linux Containers are not subject to assessment in this evaluation.
GSS-API Security Mechanisms	The GSS-API is used to secure the connection between different audit daemons. The security mechanisms used by the GSS-API, however, is not part of the evaluation. Therefore, A.CONNECT applies to the audit-related communication link.

Table 1: Non-evaluated functionalities

Note: Packages and mechanisms not covered with security claims and subsequent assessments during the evaluation or disabling the respective functionality in the evaluated configuration result from resource constraints during the evaluation but does not imply that the respective package or functionality is implemented insecurely.

1.5.2.3 Configurations

The evaluated configurations are defined as follows:

- The CC evaluated package set must be selected at install time in accordance with the description provided in the Evaluated Configuration Guide and installed accordingly.
- The TOE supports the use of IPv4 and IPv6, both are also supported in the evaluated configuration. IPv6 conforms to the following RFCs:
  - RFC 2460 specifying the basic IPv6 protocol
  - IPv6 source address selection as documented in RFC 6724
  - Linux implements several new socket options (IPV6_RECVPKTINFO, IPV6_PKTINFO, IPV6_RECVHOPOPTS, IPV6_HOPOPTS, IPV6_RECVDSTOPTS, IPV6_DSTOPTS, IPV6_RTHDRDSTOPTS, IPV6_RECVRTHDR, IPV6_RTHDR, IPV6_RECVHOPOPTS, IPV6_HOPOPTS, IPV6_{RECV,}TCLASS) and ancillary data in order to support advanced IPv6 applications including ping, traceroute, routing daemons and others. The following section introduces Internet Protocol Version 6 (IPv6). For additional information about referenced socket options and advanced IPv6 applications, see RFC 3542
  - Transition from IPv4 to IPv6: dual stack, and configured tunneling according to RFC 4213.
- The default configuration for identification and authentication are the defined password-based PAM modules as well as by the certificate based authentication for OpenSSH. Support for other authentication options, e.g. smart card authentication, is not included in the evaluation configuration.
- If the system console is used, it must be connected directly to the TOE and afforded the same physical protection as the TOE.

Deviations from the configurations and settings specified with the Evaluated Configuration Guide are not permitted.

The TOE comprises a single system (and optional peripherals) running the TOE software listed. Cluster configurations are not permitted in the evaluated configuration.

1.5.2.4 TOE Environment

Several TOE systems may be interlinked in a network, and individual networks may be joined by bridges and/or routers, or by TOE systems which act as routers and/or gateways. Each of the TOE systems implements its own security policy. The TOE does not include any synchronization function for those policies. As a result a single user may have user accounts on each of those systems with different UIDs, different roles, and other different attributes. (A synchronization method may optionally be used, but it not part of the TOE and must not use methods that conflict with the TOE requirements.)

If other systems are connected to a network they need to be configured and managed by the same authority using an appropriate security policy that does not conflict with the security policy of the TOE. All links between this network and untrusted networks (e. g. the Internet) need to be protected by appropriate measures such as carefully configured firewall systems that prohibit attacks from the untrusted networks. Those protections are part of the TOE environment.

1.5.2.5 Security Policy Model

The security policy for the TOE is defined by the security functional requirements in chapter 6. The following is a list of the subjects and objects participating in the policy.

Subjects:

- Processes acting on behalf of a human user or technical entity.

Named objects:

- File system objects in the following allowed file systems:
  - Ext4 - standard file system for general data
  - iso9660 - ISO9660 file system for CD-ROM and DVD
  - tmpfs - the temporary file system backed by RAM
  - rootfs - the virtual root file system used temporarily during system boot
  - procfs - process file system holding information about processes, general statistical data and tunable kernel parameters
  - sysfs - system-related file system covering general information about resources maintained by the kernel including several tunable parameters for these resources
  - devpts - pseudoterminal file system for allocating virtual TTYs on demand
  - devtmpfs - temporary file system that allows the kernel to generate character or block device nodes
  - binfmt_misc - configuration interface allowing the assignment of executable file formats with user space applications
  - securityfs - interface for loadable security modules (LSM) to provide tunables and configuration interfaces to user space
  - selinuxfs - interface for allowing user space components to interact with the SELinux module inside the kernel, including managing the SELinux policy.

  Please note that the TOE supports a number of additional virtual (i.e. without backing of persistent storage) file systems which are only accessible to the TSF - they are not or cannot be mounted. All above mentioned virtual file systems implement access decisions based DAC attributes inferred from the underlying process' DAC attributes. Additional restrictions may apply for specific objects in this file system.
- Inter Process Communication (IPC) objects:

    - Semaphores
    - Shared memory
    - Message queues
    - Named pipes
    - UNIX domain socket special files
- Network sockets (irrespectively of their type - such as Internet sockets, netlink sockets)
- Block device objects
- at and cron job queues maintained for each user

TSF data:

- TSF executable code
- Subject meta data - all data used for subjects except data which is not interpreted by the TSF and does not implement parts of the TSF (this data is called user data)
- Named object meta data - all data used for the respective objects except data which is not interpreted by the TSF and does not implement parts of the TSF (this data is called user data)
- User accounts, including the security attributes defined by FIA_ATD.1
- Audit records
- Session keys for dm_crypt block devices and passphrases protecting the session keys

User data:

- Non-TSF executable code used to drive the behavior of subjects
- Data not interpreted by TSF and stored or transmitted using named objects

2 CC Conformance Claim

This Security Target is CC Part 2 extended and CC Part 3 conformant, with a claimed Evaluation Assurance Level of EAL4, augmented by ALC_FLR.3.

This Security Target claims conformance to the following Protection Profiles and PP packages, if any:

- [OSPP]: BSI Operating System Protection Profile. Version 2.0 as of 2010; strict conformance.
- [OSPP-AUD]: BSI OSPP Extended Package - Advanced Audit. Version 2.0 as of 2010; strict conformance.

Common Criteria [CC] version 3.1 revision 4 is the basis for this conformance claim.

3 Security Problem Definition

3.1 Threat Environment

Threats to be countered by the TOE are characterized by the combination of an asset being subject to a threat, a threat agent and an adverse action.

The definition of threat agents and protected assets that follows is applicable to the OSPP base, as well as to the OSPP extended packages, unless noted otherwise.

3.1.1 Assets

Assets to be protected are:

- Persistent storage objects used to store user data and/or TSF data, where this data needs to be protected from any of the following operations:
  - Unauthorized read access
  - Unauthorized modification
  - Unauthorized deletion of the object
  - Unauthorized creation of new objects
  - Unauthorized management of object attributes
- Transient storage objects, including network data
- TSF functions and associated TSF data
- The resources managed by the TSF that are used to store the above-mentioned objects, including the metadata needed to manage these objects.

3.1.2 Threat Agents

Threat agents are external entities that potentially may attack the TOE. They satisfy one or more of the following criteria:

- External entities not authorized to access assets may attempt to access them either by masquerading as an authorized entity or by attempting to use TSF services without proper authorization.
- External entities authorized to access certain assets may attempt to access other assets they are not authorized to either by misusing services they are allowed to use or by masquerading as a different external entity.
- Untrusted subjects may attempt to access assets they are not authorized to either by misusing services they are allowed to use or by masquerading as a different subject.

Threat agents are typically characterized by a number of factors, such as expertise, available resources, and motivation, with motivation being linked directly to the value of the assets at stake. The TOE protects against intentional and unintentional breach of TOE security by attackers possessing an enhanced-basic attack potential.

3.1.3 Threats countered by the TOE

T.ACCESS.TSFDATA

A threat agent might read or modify TSF data without the necessary authorization when the data is stored or transmitted.

T.ACCESS.USERDATA

A threat agent might gain access to user data stored, processed or transmitted by the TOE without being appropriately authorized according to the TOE security policy.

T.ACCESS.TSFFUNC

A threat agent might use or modify functionality of the TSF without the necessary privilege to grant itself or others unauthorized access to TSF data or user data.

T.ACCESS.COMM

A threat agent might access a communication channel that establishes a trust relationship between the TOE and another remote trusted IT system or masquerade as another remote trusted IT system.

T.RESTRICT.NETTRAFFIC

A threat agent might get access to information or transmit information to other recipients via network communication channels without authorization for this communication attempt by the information flow control policy.

T.IA.MASQUERADE

A threat agent might masquerade as an authorized entity including the TOE itself or a part of the TOE in order to gain unauthorized access to user data, TSF data, or TOE resources.

T.IA.USER

A threat agent might gain access to user data, TSF data or TOE resources with the exception of public objects without being identified and authenticated.

T.UNOBSERVED_AUDIT

A threat agent might violate security policies but go undetected because there is too much audit data or too many local audit facilities in an enterprise network, causing the audit administrator to review and administer these audit facilities infrequently.

T.ACCESS.CP.USERDATA

A threat agent might gain access to user data at rest which is confidentiality protected without possessing the authorization of the owner, either at runtime of the TOE or when the TSF are inactive.

3.2 Assumptions

3.2.1 Environment of use of the TOE

3.2.1.1 Physical

A.PHYSICAL

It is assumed that the IT environment provides the TOE with appropriate physical security, commensurate with the value of the IT assets protected by the TOE.

3.2.1.2 Personnel

A.MANAGE

The TOE security functionality is managed by one or more competent individuals. The system administrative personnel are not careless, willfully negligent, or hostile, and will follow and abide by the instructions provided by the guidance documentation.

A.AUTHUSER

Authorized users possess the necessary authorization to access at least some of the information managed by the TOE and are expected to act in a cooperating manner in a benign environment.

A.TRAINEDUSER

Users are sufficiently trained and trusted to accomplish some task or group of tasks within a secure IT environment by exercising complete control over their user data.

3.2.1.3 Procedural

A.DETECT

Any modification or corruption of security-enforcing or security-relevant files of the TOE, user or the underlying platform caused either intentionally or accidentally will be detected by an administrative user.

A.PEER.MGT

All remote trusted IT systems trusted by the TSF to provide TSF data or services to the TOE, or to support the TSF in the enforcement of security policy decisions are assumed to be under the same management control and operate under security policy constraints compatible with those of the TOE.

A.PEER.FUNC

All remote trusted IT systems trusted by the TSF to provide TSF data or services to the TOE, or to support the TSF in the enforcement of security policy decisions are assumed to correctly implement the functionality used by the TSF consistent with the assumptions defined for this functionality.

3.2.1.4 Connectivity

A.CONNECT

All connections to and from remote trusted IT systems and between physically-separate parts of the TSF not protected by the TSF itself are physically or logically protected within the TOE environment to ensure the integrity and confidentiality of the data transmitted and to ensure the authenticity of the communication end points.

3.3 Organizational Security Policies

P.ACCOUNTABILITY

The users of the TOE shall be held accountable for their security-relevant actions within the TOE.

P.USER

Authority shall only be given to users who are trusted to perform the actions correctly.

P.PROTECT_SSH_KEY

When using SSH with key-based authentication, organizational procedures must exist that ensure users protect their private SSH key component against its use by any other user.

Note: The protection of the key can be established by access permissions to the file holding the key (when using the OpenSSH client, the key file permissions are automatically verified and the key is rejected if the permissions are not restrictive), or by encrypting the key with a passphrase. Making the SSH private key available to any other user is akin to telling that user the password for password-based authentication.

P.CP.ANCHOR

Users shall control the confidentiality protection anchor for their confidentiality-protected user data, and reset/replace/modify it if desired.

4 Security Objectives

4.1 Objectives for the TOE

O.AUDITING

The TSF must be able to record defined security-relevant events (which usually include security-critical actions of users of the TOE). The TSF must protect this information and present it to authorized users if the audit trail is stored on the local system. The information recorded for security-relevant events must contain the time and date the event happened and, if possible, the identification of the user that caused the event, and must be in sufficient detail to help the authorized user detect attempted security violations or potential misconfiguration of the TOE security features that would leave the IT assets open to compromise.

O.CRYPTO.NET

The TSF must allow authorized users to remotely access the TOE using a cryptographically-protected network protocol that ensures integrity and confidentiality of the transported data and is able to authenticate the end points of the communication. Note that the same protocols may also be used in the case where the TSF is physically separated into multiple parts that must communicate securely with each other over untrusted network connections.

O.DISCRETIONARY.ACCESS

The TSF must control access of subjects and/or users to named resources based on identity of the object. The TSF must allow authorized users to specify for each access mode which users/subjects are allowed to access a specific named object in that access mode.

O.NETWORK.FLOW

The TOE shall mediate communication between sets of TOE network interfaces, between a network interface and the TOE itself, and between subjects in the TOE and the TOE itself in accordance with its security policy.

O.SUBJECT.COM

The TOE shall mediate communication between subjects acting with different subject security attributes in accordance with its security policy.

O.I&A

The TOE must ensure that users have been successfully authenticated before allowing any action the TOE has defined to provide to authenticated users only.

O.MANAGE

The TSF must provide all the functions and facilities necessary to support the authorized users that are responsible for the management of TOE security mechanisms and must ensure that only authorized users are able to access such functionality.

O.TRUSTED_CHANNEL

The TSF must be designed and implemented in a manner that allows for establishing a trusted channel between the TOE and a remote trusted IT system that protects the user data and TSF data transferred over this channel from disclosure and undetected modification and prevents masquerading of the remote trusted IT system.

O.REMOTE_AUDIT

The TOE shall be able to process audit trails of remote trusted IT systems and to administer the audit functions of remote trusted IT systems according to a centrally-defined policy.

O.ANALYZE_AUDIT

The TOE shall provide audit trail analysis tools allowing administrators to analyze large amounts of audit data for possible or actual security violations.

O.CP.USERDATA

The TOE shall be able to protect the confidentiality of user data at rest separately for each user where the user can select the data which is being maintained under confidentiality protection.

O.CP.ANCHOR

The TOE shall allow each user to manage the trust anchor for the confidentiality protection of his own user data.

O.RUNTIME.PROTECTION

The TOE shall offer a runtime protection mechanism for applications to mitigate the effects of buffer overruns potentially present in applications and associated libraries.

4.2 Objectives for the Operational Environment

OE.ADMIN

Those responsible for the TOE are competent and trustworthy individuals, capable of managing the TOE and the security of the information it contains.

OE.REMOTE

If the TOE relies on remote trusted IT systems to support the enforcement of its policy, those systems provide the functions required by the TOE and are sufficiently protected from any attack that may cause those functions to provide false results.

OE.INFO_PROTECT

Those responsible for the TOE must establish and implement procedures to ensure that information is protected in an appropriate manner. In particular:

- All network and peripheral cabling must be approved for the transmittal of the most sensitive data held by the system. Such physical links are assumed to be adequately protected against threats to the confidentiality and integrity of the data transmitted.
- DAC protections on security-relevant files (such as audit trails and authentication databases) shall always be set up correctly.
- Users are authorized to access parts of the data managed by the TOE and are trained to exercise control over their own data.

OE.INSTALL

Those responsible for the TOE must establish and implement procedures to ensure that the hardware, software and firmware components that comprise the system are distributed, installed and configured in a secure manner supporting the security mechanisms provided by the TOE.

OE.MAINTENANCE

Authorized users of the TOE must ensure that the comprehensive diagnostics facilities provided by the product are invoked at every scheduled preventative maintenance period.

OE.PHYSICAL

Those responsible for the TOE must ensure that those parts of the TOE critical to enforcement of the security policy are protected from physical attack that might compromise IT security objectives. The protection must be commensurate with the value of the IT assets protected by the TOE.

OE.RECOVER

Those responsible for the TOE must ensure that procedures and/or mechanisms are provided to assure that after system failure or other discontinuity, recovery without a protection (security) compromise is achieved.

OE.TRUSTED.IT.SYSTEM

The remote trusted IT systems implement the protocols and mechanisms required by the TSF to support the enforcement of the security policy.

These remote trusted IT systems are under the same management domain as the TOE, are managed based on the same rules and policies applicable to the TOE, and are physically and logically protected equivalent to the TOE.

4.3 Security Objectives Rationale

4.3.1 Coverage

The following table provides a mapping of TOE objectives to threats and policies, showing that each objective counters or enforces at least one threat or policy, respectively.

Objective	Threats / OSPs
O.AUDITING	P.ACCOUNTABILITY
O.CRYPTO.NET	T.ACCESS.TSFDATA T.ACCESS.USERDATA T.ACCESS.TSFFUNC
O.DISCRETIONARY.ACCESS	T.ACCESS.TSFDATA T.ACCESS.USERDATA
O.NETWORK.FLOW	T.RESTRICT.NETTRAFFIC
O.SUBJECT.COM	T.ACCESS.TSFDATA T.ACCESS.USERDATA
O.I&A	T.IA.MASQUERADE T.IA.USER
O.MANAGE	T.ACCESS.TSFFUNC P.ACCOUNTABILITY P.USER
O.TRUSTED_CHANNEL	T.ACCESS.COMM
O.REMOTE_AUDIT	T.UNOBSERVED_AUDIT
O.ANALYZE_AUDIT	T.UNOBSERVED_AUDIT
O.CP.USERDATA	T.ACCESS.CP.USERDATA
O.CP.ANCHOR	P.CP.ANCHOR
O.RUNTIME.PROTECTION	T.ACCESS.TSFDATA T.ACCESS.USERDATA

Table 2: Mapping of security objectives to threats and policies

The following table provides a mapping of the objectives for the Operational Environment to assumptions, threats and policies, showing that each objective holds, counters or enforces at least one assumption, threat or policy, respectively.

Objective	Assumptions / Threats / OSPs
OE.ADMIN	A.MANAGE A.AUTHUSER A.TRAINEDUSER
OE.REMOTE	A.CONNECT T.ACCESS.COMM

Objective	Assumptions / Threats / OSPs
OE.INFO_PROTECT	A.PHYSICAL A.MANAGE A.AUTHUSER A.TRAINEDUSER P.USER P.PROTECT_SSH_KEY
OE.INSTALL	A.MANAGE A.DETECT
OE.MAINTENANCE	A.DETECT
OE.PHYSICAL	A.PHYSICAL
OE.RECOVER	A.MANAGE A.DETECT
OE.TRUSTED.IT.SYSTEM	A.PEER.MGT A.PEER.FUNC A.CONNECT

Table 3: Mapping of security objectives for the Operational Environment to assumptions, threats and policies

4.3.2 Sufficiency

The following rationale provides justification that the security objectives are suitable to counter each individual threat and that each security objective tracing back to a threat, when achieved, actually contributes to the removal, diminishing or mitigation of that threat:

Threat	Rationale for security objectives
T.ACCESS.TSFDATA	The threat of accessing TSF data without proper authorization is removed by: ● O.CRYPTO.NET requiring cryptographically-protected communication channels for data including TSF data controlled by the TOE in transit between trusted IT systems. ● O.DISCRETIONARY.ACCESS requiring that data, including TSF data stored with the TOE, have discretionary access control protection. ● O.SUBJECT.COM requiring the TSF to mediate communication between subjects. ● O.RUNTIME.PROTECTION requiring the TSF to provide functionality to mitigate the effect of protentially present buffer overrun dedicated TSF applications and their libraries.
T.ACCESS.USERDATA	The threat of accessing user data without proper authorization is removed by: ● O.CRYPTO.NET requiring cryptographically-protected communication channels for data including user data controlled by the TOE in transit between trusted IT systems.

Threat	Rationale for security objectives
	• O.DISCRETIONARY.ACCESS requiring that data including user data stored with the TOE, have discretionary access control protection. • O.SUBJECT.COM requiring the TSF to mediate communication between subjects. • O.RUNTIME.PROTECTION requiring the TSF to provide functionality to mitigate the effect of protentially present buffer overrun for specifically compiled user applications and their libraries.
T.ACCESS.TSFFUNC	The threat of accessing TSF functions without proper authorization is removed by: • O.CRYPTO.NET requiring cryptographically-protected communication channels to limit which TSF functions are accessible to external entities. • O.MANAGE requiring that only authorized users utilize management TSF functions.
T.ACCESS.COMM	The threat of accessing a communication channel that establishes a trust relationship between the TOE and another remote trusted IT system is removed by: • O.TRUSTED_CHANNEL requiring that the TOE implements a trusted channel between itself and a remote trusted IT system protecting the user data and TSF data transferred over this channel from disclosure and undetected modification and prevents masquerading of the remote trusted IT system. • OE.REMOTE requiring that those systems providing the functions required by the TOE are sufficiently protected from any attack that may cause those functions to provide false results.
T.RESTRICT.NETTRAFFIC	The threat of accessing information or transmitting information to other recipients via network communication channels without authorization for this communication attempt is removed by: • O.NETWORK.FLOW requiring the TOE to mediate the communication between itself and remote entities in accordance with its security policy.
T.IA.MASQUERADE	The threat of masquerading as an authorized entity in order to gain unauthorized access to user data, TSF data or TOE resources is removed by: • O.I&A requiring that each entity interacting with the TOE is properly identified and authenticated before allowing any action the TOE is defined to provide to authenticated users only.
T.IA.USER	The threat of accessing user data, TSF data or TOE resources without being identified and authenticated is removed by: • O.I&A requiring that each entity interacting with the TOE is properly identified and authenticated before allowing any action the TOE has defined to provide to authenticated users only.

Threat	Rationale for security objectives
T.UNOBSERVED_AUDIT	The threat of undetected violations of security policies due to too much audit data or too many local audit facilities in an enterprise network is removed by: • O.ANALYZE_AUDIT requiring the TOE to provide trail analysis tools allowing administrators to analyze large amounts of audit data for possible or actual security violations. • O.REMOTE_AUDIT requiring the TOE to process audit trails of remote trusted IT systems and to administer the audit functions of remote trusted IT systems according to a centrally-defined policy.
T.ACCESS.CP.USERDATA	The threat of gaining access to user data at rest which is confidentiality protected without possessing the authorization of the owner, either at runtime of the TOE or when the TSF are inactive is removed by: • O.CP.USERDATA requiring the TOE to be able to protect the confidentiality of user data at rest separately for each user.

Table 4: Sufficiency of objectives countering threats

The following rationale provides justification that the security objectives for the environment are suitable to cover each individual assumption, that each security objective for the environment that traces back to an assumption about the environment of use of the TOE, when achieved, actually contributes to the environment achieving consistency with the assumption, and that if all security objectives for the environment that trace back to an assumption are achieved, the intended usage is supported:

Assumption	Rationale for security objectives
A.PHYSICAL	The assumption on the IT environment to provide the TOE with appropriate physical security, commensurate with the value of the IT assets protected by the TOE is covered by: • OE.INFO_PROTECT requiring the approval of network and peripheral cabling. • OE.PHYSICAL requiring physical protection.
A.MANAGE	The assumptions on the TOE security functionality being managed by one or more trustworthy individuals is covered by: • OE.ADMIN requiring trustworthy personnel managing the TOE. • OE.INFO_PROTECT requiring personnel to ensure that information is protected in an appropriate manner. • OE.INSTALL requiring personnel to ensure that components that comprise the system are distributed, installed and configured in a secure manner supporting the security mechanisms provided by the TOE. • OE.RECOVER requiring personnel to assure that after system failure or other discontinuity, recovery without a protection (security) compromise is achieved.

Assumption	Rationale for security objectives
A.AUTHUSER	The assumption on authorized users to possess the necessary authorization to access at least some of the information managed by the TOE and to act in a cooperating manner in a benign environment is covered by: ● OE.ADMIN ensuring that those responsible for the TOE are competent and trustworthy individuals, capable of managing the TOE and the security of the information it contains. ● OE.INFO_PROTECT requiring that DAC protections on security-relevant files (such as audit trails and authentication databases) shall always be set up correctly and that users are authorized to access parts of the data maintained by the TOE.
A.TRAINEDUSER	The assumptions on users to be sufficiently trained and trusted to accomplish some task or group of tasks within a secure IT environment by exercising complete control over their user data is covered by: ● OE.ADMIN requiring competent personnel managing the TOE. ● OE.INFO_PROTECT requiring that those responsible for the TOE must establish and implement procedures to ensure that information is protected in an appropriate manner and that users are trained to exercise control over their own data.
A.DETECT	The assumption that modification or corruption of security-enforcing or security-relevant files will be detected by an administrative user is covered by: ● OE.INSTALL requiring an administrative user to ensure that the TOE is distributed, installed and configured in a secure manner supporting the security mechanisms provided by the TOE. ● OE.MAINTENANCE requiring an administrative user to ensure that the diagnostics facilities are invoked at every scheduled preventative maintenance period, verifying the correct operation of the TOE. ● OE.RECOVER requiring an administrative user to ensure that procedures and/or mechanisms are provided to assure that after system failure or other discontinuity, recovery without a protection (security) compromise is achieved.
A.PEER.MGT	The assumption on all remote trusted IT systems to be under the same management control and operate under security policy constraints compatible with those of the TOE is covered by: ● OE.TRUSTED.IT.SYSTEM requiring that these remote trusted IT systems are under the same management domain as the TOE, and are managed based on the same rules and policies applicable to the TOE.
A.PEER.FUNC	The assumption on all remote trusted IT systems to correctly implement the functionality used by the TSF consistent with the assumptions defined for this functionality is covered by: ● OE.TRUSTED.IT.SYSTEM requiring that the remote trusted IT systems implement the protocols and mechanisms required by the TSF to support the enforcement of the security policy.

Assumption	Rationale for security objectives
A.CONNECT	The assumption on all connections to and from remote trusted IT systems and between physically separate parts of the TSF not protected by the TSF itself are physically or logically protected is covered by: • OE.REMOTE requiring that remote trusted IT systems provide the functions required by the TOE and are sufficiently protected from any attack that may cause those functions to provide false results. • OE.TRUSTED.IT.SYSTEM demanding the physical and logical protection equivalent to the TOE.

Table 5: Sufficiency of objectives holding assumptions

The following rationale provides justification that the security objectives are suitable to cover each individual organizational security policy, that each security objective that traces back to an OSP, when achieved, actually contributes to the implementation of the OSP, and that if all security objectives that trace back to an OSP are achieved, the OSP is implemented:

OSP	Rationale for security objectives
P.ACCOUNTABILITY	The policy to hold users accountable for their security-relevant actions within the TOE is implemented by: • O.AUDITING providing the TOE with audit functionality. • O.MANAGE allowing the management of this function.
P.USER	The policy to match the trust given to a user and the actions the user is given authority to perform is implemented by: • O.MANAGE allowing appropriately-authorized users to manage the TSF. • OE.INFO_PROTECT, which requires that users are trusted to use the protection mechanisms of the TOE to protect their data.
P.PROTECT_SSH_KEY	The policy to match the trust given to a user to protect his SSH private key is implemented by: • OE.INFO_PROTECT, which requires that users are trusted to exercise the control over their own data.
P.CP.ANCHOR	The policy that users shall control the confidentiality protection anchor for their confidentiality-protected user data, and reset/replace/modify it if desired is implemented by: • O.CP.ANCHOR allowing each user to manage the trust anchor for the confidentiality protection of his own user data.

Table 6: Sufficiency of objectives enforcing Organizational Security Policies

5 Extended Components Definition

The Security Target uses the extended components of FDP_RIP.3 as well as FIA_USB.2 defined by [OSPP]. They are not re-defined here again.

In addition, the Security Target defines the extended component of FCS_RNG and FDP_CDP families for usage within this ST.

5.1 Class FCS: Cryptographic support

5.1.1 Random number generation (RNG)

Family behaviour

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

Component levelling

FCS_RNG.1 is not hierarchical to any other component within the FCS_RNG family.

Management: FCS_RNG.1

The following actions could be considered for the management functions in FMT:

a) There are no management activities foreseen.

Audit: FCS_RNG.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) Minimal: There are no actions defined to be auditable.

b) Basic: There are no actions defined to be auditable.

c) Detailed: There are no actions defined to be auditable.

5.1.1.1 FCS_RNG.1 - Random number generation (Class DRG.2)

Hierarchical to: No other components.

Dependencies: No dependencies.

FCS_RNG.1.1 The TSF shall provide a deterministic random number generator that implements:

a) DRG.2.1: If initialized with a random seed [selection: **using a PTRNG of class PTG.2 as random source, using a PTRNG of class PTG.3 as random source , using an NPTRNG of class NTG.1 , [assignment: other requirements for seeding]**], the internal state of the RNG shall [selection: **have [assignment: amount of entropy], have [assignment: work factor], require [assignment: guess work]**].

b) DRG.2.2: The RNG provides forward secrecy.

c) DRG.2.3: The RNG provides backward secrecy.

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

a) DRG.2.4: The RNG, initialized with a random seed [assignment: **requirements for seeding**], generates output for which [assignment: **number of strings**] strings of bit length 128 are mutually different with probability [assignment: **probability**].

b) DRG.2.5: Statistical test suites cannot practically distinguish the random numbers from output sequences of an ideal RNG. The random numbers must pass test procedure A [assignment: **additional test suites**].

Rationale

The quality of the random number generator is defined using this SFR. The quality metric required in FCS_RNG.1.1 is detailed in the German Scheme AIS 20 and AIS 31

5.2 Class FDP: User data protection

5.2.1 Confidentiality protection (FDP_CDP)

Component levelling

The FDP_CDP family contains only one component: FDP_CDP.1.

FDP_CDP.1 is therefore not hierarchical to any other component within the FDP_CDP family.

FDP_CDP.1 Confidentiality protection for data at rest, requires that the TSF ensures that the user data is stored within containers controlled by the TSF protected against accesses while the TSF are executing as well as when the TSF are not enforced.

Management: FDP_CDP.1

The following actions could be considered for the management functions in FMT:

a) The following actions could be considered for the management functions in FMT: Management of confidentiality protection trust anchor.

Audit: FDP_CDP.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) Minimal: The identity of any user or subject using the data storage mechanism.

b) Basic: The identity of any unauthorised user or subject attempting to use the data exchange mechanisms.

c) Detailed: The identity of any unauthorised user or subject attempting to use the data exchange mechanisms.

5.2.1.1 FDP_CDP.1 - Confidentiality for data at rest

Hierarchical to:	No other components.
Dependencies:	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control]

FDP_CDP.1.1 The TSF shall enforce the [assignment: **access control SFP(s) and/or information flow control SFP(s)**] to store user data at rest in containers controlled by the TSF in a manner protected from unauthorised disclosure.

Rationale

This family provides requirements that address the protection of the confidentiality of user data while it is at rest within containers controlled by the TSF. This family differs from FDP_UCT which covers the confidentiality to be maintained during the transmission of user data between the TOE and another IT product.

6 Security Requirements

6.1 TOE Security Functional Requirements

The following table shows the security functional requirements for the TOE, and the operations performed on the components according to CC part 2: iteration (Iter.), refinement (Ref.), assignment (Ass.) and selection (Sel.).

Security functional group	Security functional requirement	Base security functional component	Source	Operations			
				Iter.	Ref.	Ass.	Sel.
General-purpose computing environment	FAU_GEN.1 Audit data generation		OSPP	No	No	Yes	No
	FAU_GEN.2 User identity association		OSPP	No	No	No	No
	FAU_SAR.1 Audit review		OSPP	No	No	Yes	No
	FAU_SAR.2 Restricted audit review		OSPP	No	No	No	No
	FAU_SAR.3(AUD) Selectable audit review [OSPP-AUD]	FAU_SAR.3	OSPP-AUD	No	No	Yes	No
	FAU_SEL.1 Selective audit		OSPP	No	No	Yes	No
	FAU_STG.1 Protected audit trail storage		OSPP	No	No	No	Yes
	FAU_STG.3 Action in case of possible audit data loss		OSPP	No	Yes	Yes	No
	FAU_STG.4 Prevention of audit data loss		OSPP	No	Yes	Yes	Yes
	FCS_CKM.1(SYM) Cryptographic key generation	FCS_CKM.1	OSPP	Yes	Yes	Yes	No
	FCS_CKM.1(RSA) Cryptographic key generation	FCS_CKM.1	OSPP	Yes	No	Yes	No
	FCS_CKM.1(DSA) Cryptographic key generation	FCS_CKM.1	OSPP	Yes	No	Yes	Yes
	FCS_CKM.2(NET) Cryptographic key distribution	FCS_CKM.2	OSPP	No	No	Yes	No
	FCS_CKM.4 Cryptographic key destruction		OSPP	No	No	Yes	No
	FCS_COP.1(NET) Cryptographic operation	FCS_COP.1	OSPP	Yes	No	Yes	Yes
	FCS_COP.1(CP) Cryptographic operation	FCS_COP.1	CC Part 2	Yes	No	Yes	No

Security functional group	Security functional requirement	Base security functional component	Source	Operations			
				Iter.	Ref.	Ass.	Sel.
	FCS_RNG.1(SSH-DFLT) Random number generation (Class DRG.2)	FCS_RNG.1	ECD	Yes	No	Yes	Yes
	FCS_RNG.1(SSH-FIPS) Random number generation (Class DRG.2)	FCS_RNG.1	ECD	Yes	Yes	Yes	Yes
	FCS_RNG.1(DM-INIT) Random number generation (Class DRG.2)	FCS_RNG.1	ECD	Yes	No	Yes	Yes
	FCS_RNG.1(DM-RUN) Random number generation (Class DRG.2)	FCS_RNG.1	ECD	Yes	No	Yes	Yes
	FCS_RNG.1(DM-FIPS) Random number generation (Class DRG.2)	FCS_RNG.1	ECD	Yes	Yes	Yes	Yes
	FDP_ACC.1(PSO) Subset access control	FDP_ACC.1	OSPP	Yes	No	Yes	No
	FDP_ACC.1(TSO) Subset access control	FDP_ACC.1	OSPP	Yes	No	Yes	No
	FDP_ACF.1(PSO) Security attribute based access control	FDP_ACF.1	OSPP	Yes	No	Yes	No
	FDP_ACF.1(TSO) Security attribute based access control	FDP_ACF.1	OSPP	Yes	No	Yes	No
	FDP_IFC.2(NI) Complete information flow control	FDP_IFC.2	OSPP	No	No	Yes	No
	FDP_IFF.1(NI-IPTables) Simple security attributes	FDP_IFF.1	OSPP	Yes	Yes	Yes	Yes
	FDP_IFF.1(NI-ebtables) Simple security attributes	FDP_IFF.1	OSPP	Yes	Yes	Yes	Yes
	FDP_ITC.2(BA) Import of user data with security attributes	FDP_ITC.2	OSPP	No	No	Yes	No
	FDP_RIP.2 Full residual information protection		OSPP	No	No	No	Yes
	FDP_RIP.3 Full residual information protection of resources		OSPP	No	No	No	Yes
	FIA_AFL.1 Authentication failure handling		OSPP	No	No	Yes	Yes
	FIA_ATD.1(HU) User attribute definition	FIA_ATD.1	OSPP	Yes	No	Yes	No
	FIA_ATD.1(TU) User attribute definition	FIA_ATD.1	OSPP	Yes	No	Yes	No

Security functional group	Security functional requirement	Base security functional component	Source	Operations			
				Iter.	Ref.	Ass.	Sel.
	FIA_SOS.1 Verification of secrets		OSPP	No	No	No	No
	FIA_UAU.1 Timing of authentication		OSPP	No	No	Yes	No
	FIA_UAU.5 Multiple authentication mechanisms		OSPP	No	No	Yes	No
	FIA_UAU.7 Protected authentication feedback		OSPP	No	No	No	No
	FIA_UID.1 Timing of identification		OSPP	No	No	Yes	No
	FIA_USB.2 Enhanced user-subject binding		OSPP	No	No	Yes	No
	FPT_FLS.1(FULL) Failure with preservation of secure state - full buffer overflow protection	FPT_FLS.1	CC Part 2	Yes	No	Yes	No
	FPT_FLS.1(PARTIAL) Failure with preservation of secure state - partial buffer overflow protection	FPT_FLS.1	CC Part 2	Yes	No	Yes	No
	FPT_STM.1 Reliable time stamps		OSPP	No	No	No	No
	FPT_TDC.1(BA) Inter-TSF basic TSF data consistency	FPT_TDC.1	CC Part 2	No	No	Yes	No
	FTA_SSL.1 TSF-initiated session locking		OSPP	No	No	Yes	No
	FTA_SSL.2 User-initiated locking		OSPP	No	No	Yes	No
	FTP_ITC.1 Inter-TSF trusted channel		OSPP	No	No	Yes	Yes
Confidentiality protection of data at rest	FDP_ACC.2(CP) Complete access control	FDP_ACC.2	CC Part 2	Yes	No	Yes	No
	FDP_ACF.1(CP) Security attribute based access control	FDP_ACF.1	CC Part 2	Yes	No	Yes	No
	FDP_CDP.1(CP) Confidentiality for data at rest	FDP_CDP.1	ECD	No	No	Yes	No
Management related functionality	FMT_MSA.1(PSO) Management of object security attributes	FMT_MSA.1	OSPP	Yes	No	Yes	Yes
	FMT_MSA.1(TSO) Management of object security attributes	FMT_MSA.1	OSPP	Yes	No	Yes	No
	FMT_MSA.1(CP) Management of security attributes	FMT_MSA.1	CC Part 2	Yes	No	Yes	Yes

Security functional group	Security functional requirement	Base security functional component	Source	Operations			
				Iter.	Ref.	Ass.	Sel.
	FMT_MSA.3(PSO) Static attribute initialisation	FMT_MSA.3	OSPP	Yes	No	Yes	No
	FMT_MSA.3(TSO) Static attribute initialisation	FMT_MSA.3	OSPP	Yes	No	Yes	No
	FMT_MSA.3(NI) Static attribute initialisation	FMT_MSA.3	OSPP	Yes	No	Yes	Yes
	FMT_MSA.3(CP) Static attribute initialisation	FMT_MSA.3	CC Part 2	Yes	No	Yes	Yes
	FMT_MSA.4(PSO) Security attribute value inheritance	FMT_MSA.4	OSPP	No	No	Yes	No
	FMT_MTD.1(AE) Management of TSF data	FMT_MTD.1	OSPP	Yes	No	Yes	No
	FMT_MTD.1(AS) Management of TSF data	FMT_MTD.1	OSPP	Yes	No	Yes	Yes
	FMT_MTD.1(AT) Management of TSF data	FMT_MTD.1	OSPP	Yes	No	Yes	Yes
	FMT_MTD.1(AF) Management of TSF data	FMT_MTD.1	OSPP	Yes	No	Yes	Yes
	FMT_MTD.1(NI) Management of TSF data	FMT_MTD.1	OSPP	Yes	No	Yes	Yes
	FMT_MTD.1(IAT) Management of TSF data	FMT_MTD.1	OSPP	Yes	No	Yes	No
	FMT_MTD.1(IAF) Management of TSF data	FMT_MTD.1	OSPP	Yes	No	Yes	No
	FMT_MTD.1(IAU) Management of TSF data	FMT_MTD.1	OSPP	Yes	No	Yes	No
	FMT_MTD.1(SSH) Management of TSF data	FMT_MTD.1	CC Part 2	Yes	No	Yes	Yes
	FMT_MTD.1(SSL) Management of TSF data	FMT_MTD.1	CC Part 2	Yes	No	Yes	Yes
	FMT_MTD.1(AUD-AE) Management of TSF data [OSPP-AUD]	FMT_MTD.1	OSPP-AUD	Yes	No	Yes	No
	FMT_MTD.1(AUD-AS) Management of TSF data [OSPP-AUD]	FMT_MTD.1	OSPP-AUD	Yes	No	Yes	Yes
	FMT_MTD.1(AUD-AT) Management of TSF data [OSPP-AUD]	FMT_MTD.1	OSPP-AUD	Yes	No	Yes	Yes

Security functional group	Security functional requirement	Base security functional component	Source	Operations			
				Iter.	Ref.	Ass.	Sel.
	FMT_MTD.1(AUD-AF) Management of TSF data [OSPP-AUD]	FMT_MTD.1	OSPP-AUD	Yes	No	Yes	Yes
	FMT_MTD.1(CP-AN) Management of TSF data	FMT_MTD.1	CC Part 2	Yes	No	Yes	Yes
	FMT_MTD.1(CP-UD) Management of TSF data	FMT_MTD.1	CC Part 2	Yes	No	Yes	Yes
	FMT_REV.1(OBJ) Revocation	FMT_REV.1	OSPP	Yes	No	Yes	No
	FMT_REV.1(USR) Revocation	FMT_REV.1	OSPP	Yes	Yes	Yes	No
	FMT_SMF.1 Specification of management functions		OSPP	No	No	Yes	No
	FMT_SMR.2 Security management roles		CC Part 2	No	No	Yes	No

Table 7: Security functional requirements for the TOE

6.1.1 General-purpose computing environment

6.1.1.1 Audit data generation (FAU_GEN.1)

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 basic level of audit; and

c) all modifications to the set of events being audited;

d) all user authentication attempts;

e) all denied accesses to objects for which the access control policy defined in the OSPP base applies;

f) explicit modifications of access rights to objects covered by the access control policies; and

g) **no further rules** .

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 identity (if applicable), and outcome 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;

i. User identity (if applicable); and

ii. **no further rules** .

6.1.1.2 User identity association (FAU_GEN.2)

FAU_GEN.2.1 For audit events resulting from actions of identified users, the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

Application Note: *The TOE maintains a "Login UID", which is inherited by every new process spawned. This allows the TOE to identify the "real" originator of an event, regardless if he has changed his real and / or effective and filesystem UID e. g. using the su command or executing a setuid or setgid program.*

6.1.1.3 Audit review (FAU_SAR.1)

FAU_SAR.1.1 The TSF shall provide **the root user** with the capability to read **all audit 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: *The audit records are stored in ASCII format and can therefore be read with a normal editor or pager. In addition, the TOE provides specific tools that support the interpretation of the audit trail.*

Application Note: *The audit trail is stored in a file that is readable to the root user only.*

6.1.1.4 Restricted audit review (FAU_SAR.2)

FAU_SAR.2.1 The TSF shall prohibit all users read access to the audit records, except those users that have been granted explicit read-access.

Application Note: *The protection of the audit records is based on the Unix permission bit settings defined by FDP_ACC.1(PSO) supported by FDP_ACF.1(PSO).*

6.1.1.5 Selectable audit review [OSPP-AUD] (FAU_SAR.3(AUD))

FAU_SAR.3.1 The TSF shall provide the ability to perform

a) searches based on pattern matching,

b) searches based on string matching,

c) searches based on exclusion of strings or patterns,

d) sorting,

e) extraction,

f) **no other operations**,

g) combination of any of the aforementioned operations in any order, where the result of one operation is the input for the next operation

of audit data based on the following attributes:

a) Identity of the remote trusted IT system that created the audit data;

b) **User identity**;

c) **Group identifier (real and effective)**;

d) **Event type**;

e) **Outcome (success/failure)**;

f) **Login from a specific remote hostname**;

g) **Login user ID**;
h) **Process ID**;
i) **Role that enabled access**;
j) **Date and time of the audit event**;
k) **Object name**;
l) **Type of access**;
m) **Any combination of the above items**.

6.1.1.6 Selective audit (FAU_SEL.1)

FAU_SEL.1.1 The TSF shall be able to select the set of events to be audited from the set of all auditable events based on the following attributes:
a) Type of audit event;
b) Subject or user identity;
c) Outcome (success or failure) of the audit event;
d) Named object identity;
e) **Access types on a particular object;**
f) **System call number;**

Application Note: *The TOE provides an application that allows specification of the audit rules which injects the rules into the kernel for enforcement. The Linux kernel auditing mechanism obtains all audit events and decides based on this rule set whether an event is forwarded to the audit daemon for storage.*

6.1.1.7 Protected audit trail storage (FAU_STG.1)

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

FAU_STG.1.2 The TSF shall be able to **prevent** unauthorised modifications to the audit records in the audit trail.

Application Note: *The protection of the audit records is based on the Unix permission bit settings defined by FDP_ACC.1(PSO) supported by FDP_ACF.1(PSO). The audit trail is readable and writeable to the root user only.*

Application Note: *FAU_STG.1(AUD) from the OSPP extended package for auditing is merged with this SFR.*

6.1.1.8 Action in case of possible audit data loss (FAU_STG.3)

FAU_STG.3.1 The TSF shall **notify an authorized administrator** if the audit trail exceeds **a root-user selectable, pre-defined size limit of the audit trail** or if any of the following **condition** is detected that may result in a loss of audit records *: no other condition* .

Application Note: *The term "authorized administrator" refers to the user that is notified by the auditd daemon. This daemon can be configured to notify different users in different ways. The administrator of the system must ensure that the auditd is configured to send the notification to the intended recipient.*

Application Note: *The alarm generated by the TOE can be configured to be a syslog message or the execution of an administrator-specified application. This message or action of executing the application is generated when the audit trail capacity exceeds the limit defined in the auditd.conf file.*

Application Note: *The information of the threshold limit is done in the configuration file of the auditd daemon. This file is only writeable to the root user.*

6.1.1.9 Prevention of audit data loss (FAU_STG.4)

FAU_STG.4.1 The TSF shall *be able to* **ignore the audited events** and **perform one of the following administrator-defined actions:**

a) Stop all processes that attempt to generate an audit record;

b) Switch to single user mode;

c) Halt the system

if the audit trail is full.

Application Note: *The SFR lists all configuration possibilities that apply to the case when the audit trail is full (i.e. the disk is full). Even though the SFR mentions the "ignoring of audit events" separate from the other options, all options should be seen as equal where the root user can select one of these options.*

6.1.1.10 Cryptographic key generation (FCS_CKM.1(SYM))

FCS_CKM.1.1 The TSF shall generate symmetric cryptographic keys in accordance with a specified cryptographic key generation algorithm capable of generating a random bit sequence and specified cryptographic key sizes:

a) *AES*128 bits,

b) *Triple-DES*168 bits,

c) *AES*256 bits,

d) **Twofish, Serpent: 128 bits, 192 bits, and 256 bits,**

e) **AES: 192 bits**

f) **HMAC-SHA-1: 160 bits**

g) **HMAC-SHA-256: 256 bits**

h) **HMAC-SHA-384: 384 bits**

i) **HMAC-SHA-512: 512 bits**

j) **PBKDF2 using SHA-1, SHA-256, SHA-384 or SHA-512 for disk encryption: key encryption key size equal to the size of the device encryption key to be protected**

that meet the following: **cryptographic key generation algorithm based on:**

a) the key agreement and key derivation function specified in FCS_CKM.2(NET) using random numbers derived from the random number generator defined in FCS_RNG.1(SSH-DFLT) for use in OpenSSH applications when FIPS 140-2 mode is not configured;

b) the key agreement and key derivation function specified in FCS_CKM.2(NET) using random numbers derived from the random number generator defined in FCS_RNG.1(SSH-FIPS) for use in OpenSSH applications when FIPS 140-2 mode is configured;

c) FCS_RNG.1(DM-RUN) random number generator for use during initialization of confidentiality-protected disks during normal operation of the TOE when FIPS 140-2 mode is not configured.

d) FCS_RNG.1(DM-INIT) for use during initialization of confidentiality-protected disks at initial installation time when FIPS 140-2 mode is not configured.

e) FCS_RNG.1(DM-FIPS) for use during initialization of confidentiality-protected disks when FIPS 140-2 mode is configured.

f) PBKDF2: SP800-132 section 5.4 option 2a.

6.1.1.11 Cryptographic key generation (FCS_CKM.1(RSA))

FCS_CKM.1.1 The TSF shall generate RSA cryptographic keys in accordance with a specified cryptographic key generation algorithm defined in U.S. NIST FIPS PUB 186-3 appendix B.3 and specified cryptographic key sizes:

a) 2048 bits,

b) **1024 bits,**

c) **3072 bits**

that meet the following:

a) U.S. NIST FIPS PUB 186-3,

b) **no other standard.**

Application Note:

The TOE supports the generation of RSA keys for the OpenSSH host key as well as the OpenSSH user keys using the ssh-keygen(1) application. The following random number generator is used to support the key generation:

- *FCS_RNG.1(SSH-DFLT) for use in OpenSSH applications when FIPS 140-2 mode is not configured;*
- *FCS_RNG.1(SSH-FIPS) for use in OpenSSH applications when FIPS 140-2 mode is configured;*

6.1.1.12 Cryptographic key generation (FCS_CKM.1(DSA))

FCS_CKM.1.1 The TSF shall generate DSA cryptographic keys in accordance with a specified cryptographic key generation algorithm defined in U.S. NIST FIPS PUB 186-3 appendix B.1 and specified cryptographic key sizes:

a) **L=1024, N=160 bits;**

that meet the following:

a) U.S. NIST FIPS PUB 186-3,

b) **no other standard.**

Application Note:

The TOE supports the generation of DSA keys for the OpenSSH host key as well as the OpenSSH user keys using the ssh-keygen(1) application. The following random number generator is used to support the key generation:

- *FCS_RNG.1(SSH-DFLT) for use in OpenSSH applications when FIPS 140-2 mode is not configured;*
- *FCS_RNG.1(SSH-FIPS) for use in OpenSSH applications when FIPS 140-2 mode is configured;*

6.1.1.13 Cryptographic key distribution (FCS_CKM.2(NET))

FCS_CKM.2.1 The TSF shall distribute cryptographic keys in accordance with the following specified cryptographic key distribution method that meets the following:

a) Diffie-Hellman key agreement method with diffie-hellman-group1-sha1 defined for the SSH protocol by RFC4253 supported by RFC2409;

b) Diffie-Hellman key agreement method with diffie-hellman-group14-sha1 defined for the SSH protocol by RFC4253 supported by RFC3526;

c) Diffie-Hellman key agreement method with diffie-hellman-group-exchange-sha1 defined for the SSH protocol by RFC4253 together with RFC4419;

d) Diffie-Hellman key agreement method with diffie-hellman-group-exchange-sha256 defined for the SSH protocol by RFC4253 together with RFC4419;

e) Public DSS, RSA host key exchange defined for the SSH protocol by [RFC4253];

f) Pseudo-Random-Function for deriving the IV, the session key and the HMAC key from the Diffie-Hellman shared secret using the hash type specified for the selected Diffie-Hellman group as defined for the SSH protocol by RFC4253.

Application Note: *DSS defined in RFC4253 for the host key exchange is compliant with DSA defined in FIPS 186-3.*

6.1.1.14 Cryptographic key destruction (FCS_CKM.4)

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method of **zerorization** that meets the following: **vendor-specific zeroization**.

Application Note:

The "vendor-specific zeroization" covers to the following concepts:

- *Memory objects: Overwriting the memory with zeros at the time the memory is released.*
- *Asymmetric key components stored in files: The object reuse functionality for objects defined with FDP_RIP.2 also covers this SFR.*

6.1.1.15 Cryptographic operation (FCS_COP.1(NET))

FCS_COP.1.1 The TSF shall perform encryption, decryption, integrity verification, peer authentication in accordance with the following cryptographic algorithms, cryptographic key sizes and applicable standards:

- **a) SSH allowing the use of TDES in CBC mode with 168 bits key size, and HMAC-SHA1 defined by RFC 4253;**
- **b) SSH allowing the use of AES in CBC mode with 128 bits and 256 bits key size, and HMAC-SHA1 defined by RFC 4253;**
- **c) SSH allowing the use of AES in CBC mode with 192 bits key size, and HMAC-SHA1 defined by RFC 4253;**
- **d) SSH allowing the use of AES in CTR mode with 128 bits, 192 bits and 256 bits key size, and HMAC-SHA1 defined by RFC 4253;**
- **e) SSH allowing the use of DSA with L=1024 bits, N=160 bits with the format of ssh-dss for "publickey" authentication defined by RFC 4252;**
- **f) SSH allowing the use of RSA signature verification method RSASSA-PKCS1-v1_5 with key sizes of 1024 bits, 2048 bits, 3072 bits with the format of ssh-rsa for "publickey" authentication defined by RFC 4252;**

Application Note: *The SSH protocol allows access to the console of the TOE.*

6.1.1.16 Cryptographic operation (FCS_COP.1(CP))

FCS_COP.1.1 The TSF shall perform **encryption, decryption** in accordance with a specified cryptographic algorithm **formed with any permutation of the following types of cryptographic primitives:**

- **a) Ciphers: AES, Serpent, Twofish with key sizes specified in FCS_CKM.1(SYM);**
- **b) Block chaining modes: CBC, GCM, XTS;**
- **c) IV-Handling mechanisms:**
  1. **GCM, XTS: plain64 - The initialization vector is the 64-bit little-endian version of the sector number, padded with zeros if necessary.**
  2. **CBC: essiv - The sector number is encrypted with the bulk cipher using a salt as key. The salt is derived from the cipher key used for encrypting the data with via hashing using the hashes of either SHA-1, SHA-256, SHA-384 and SHA-512.**
  3. **GCM, XTS: benbi - The initialization vector is the 64-bit big-endian version of the sector number, padded with zeros if necessary.**

and cryptographic key sizes **as allowed by the cipher specifications:**

- **a) AES: FIPS 197**
- **b) Serpent: A Proposal for the Advanced Encryption Standard**
- **c) Twofish: A 128-Bit Block Cipher**
- **d) SHA-1 and SHA-2: FIPS 180-4**

that meet the following: **LUKS-based dm-crypt Linux partition encryption schema.**

Application Note: *The list of cryptographic primitives allowed by the TOE may be reduced when booting the system in FIPS 140-2 compliant mode. The list of allowed cryptographic primitives is given in the Security Policy for the kernel crypto API FIPS 140-2 module.*

Application Note: *The list of cryptographic primitives is consistent with the requirements defined in BSI TR-02102 version 1.0, except that the XTS block chaining mode is allowed and SHA-1 is added. XTS is standardized later than the mentioned document and commonly used for disk encryption mechanisms. Furthermore, the concerns for SHA-1 regarding collisions are not considered applicable in the context of disk encryption.*

Application Note: *The master volume key (device encryption key) is encrypted with the same cipher selected for the data encryption. The key encryption key used to perform the encryption and decryption operation of the master volume key is obtained via PBKDF2 as defined in FCS_CKM.1(SYM).*

6.1.1.17 Random number generation (Class DRG.2) (FCS_RNG.1(SSH-DFLT))

FCS_RNG.1.1 The TSF shall provide a deterministic random number generator that implements:

a) DRG2.1: If initialized with a random seed using **/dev/random of class NTG.1 as random source** , the internal state of the RNG shall **have a minimum entropy of 48 bits**.

b) DRG2.2: The DRNG provides forward secrecy.

c) DRG2.3: The DRNG provides backward secrecy.

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

a) DRG.2.4: The RNG initialized with a random seed **holding 96 bits of entropy** generates output for which **2******19** strings of bit length 128 are mutually different with probability **of more than 1 - 2******-10**.

b) DRG.2.5: The test suite A **and no other test suite** cannot distinguish the random numbers from output sequences of ideal RNGs.

Application Note:

The OpenSSH applications use the deterministic RNG from OpenSSL to generate random numbers. Every time the ssh client is invoked, the ssh-keygen application is used or a new SSH connection is processed by sshd, the deterministic random number generator is seeded with data from /dev/random. Note, the OpenSSL library provides two separate deterministic RNGs, the default used in normal mode and an ANSI X9.31 compliant DRNG in FIPS 140-2 mode. This SFR covers the DRNG provided in default mode.

6.1.1.18 Random number generation (Class DRG.2) (FCS_RNG.1(SSH-FIPS))

FCS_RNG.1.1 The TSF shall provide a deterministic random number generator *using the ANSI X9.31 Appendix A2.4 standard with AES-128 core* that implements:

a) DRG2.1: If initialized with a random seed using **/dev/random of class NTG.1 as random source** , the internal state of the RNG shall **have a minimum entropy of 48 bits**.

b) DRG2.2: The DRNG provides forward secrecy.

c) DRG2.3: The DRNG provides backward secrecy.

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

a) DRG.2.4: The RNG initialized with a random seed **holding 96 bits of entropy** generates output for which **2**19** strings of bit length 128 are mutually different with probability **of more than 1 - 2**-10**.

b) DRG.2.5: The test suite A **and no other test suite** cannot distinguish the random numbers from output sequences of ideal RNGs.

Application Note:

The OpenSSH applications use the deterministic RNG from OpenSSL to generate random numbers. Every time the ssh client is invoked, the ssh-keygen application is used or a new SSH connection is processed by sshd, the deterministic random number generator is seeded with data from /dev/random. Note, the OpenSSL library provides two separate deterministic RNGs, the default used in normal mode and an ANSI X9.31 compliant DRNG in FIPS 140-2 mode. This SFR covers the DRNG provided in FIPS 140-2 mode.

6.1.1.19 Random number generation (Class DRG.2) (FCS_RNG.1(DM-INIT))

FCS_RNG.1.1 The TSF shall provide a deterministic random number generator that implements:

a) DRG2.1: If initialized with a random seed using **/dev/urandom holding sufficient entropy due to usage and environmental constraints as random source** , the internal state of the RNG shall **have a minimum entropy of 48 bits**.

b) DRG2.2: The DRNG provides forward secrecy.

c) DRG2.3: The DRNG provides backward secrecy.

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

a) DRG.2.4: The RNG initialized with a random seed **holding 96 bits of entropy** generates output for which **2**19** strings of bit length 128 are mutually different with probability **of more than 1 - 2**-10**.

b) DRG.2.5: The test suite A **and no other test suite** cannot distinguish the random numbers from output sequences of ideal RNGs.

Application Note:

The usage environment together with the administrative guidance ensure that /dev/urandom contains sufficient entropy to generate the keys.

6.1.1.20 Random number generation (Class DRG.2) (FCS_RNG.1(DM-RUN))

FCS_RNG.1.1 The TSF shall provide a deterministic random number generator that implements:

a) DRG2.1: If initialized with a random seed using **/dev/random of class NTG.1 as random source** , the internal state of the RNG shall **have a minimum entropy of 48 bits**.

b) DRG2.2: The DRNG provides forward secrecy.

c) DRG2.3: The DRNG provides backward secrecy.

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

a) DRG.2.4: The RNG initialized with a random seed **holding 96 bits of entropy** generates output for which **2**19** strings of bit length 128 are mutually different with probability **of more than 1 - 2**-10**.

b) DRG.2.5: The test suite A **and no other test suite** cannot distinguish the random numbers from output sequences of ideal RNGs.

6.1.1.21 Random number generation (Class DRG.2) (FCS_RNG.1(DM-FIPS))

FCS_RNG.1.1 The TSF shall provide a deterministic random number generator *using the ANSI X9.31 Appendix A2.4 standard with AES-128 core* that implements:

a) DRG2.1: If initialized with a random seed using **/dev/random of class NTG.1 as random source** , the internal state of the RNG shall **have a minimum entropy of 48 bits**.

b) DRG2.2: The DRNG provides forward secrecy.

c) DRG2.3: The DRNG provides backward secrecy.

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

a) DRG.2.4: The RNG initialized with a random seed **holding 96 bits of entropy** generates output for which **2******19** strings of bit length 128 are mutually different with probability **of more than 1 - 2******-10**.

b) DRG.2.5: The test suite A **and no other test suite** cannot distinguish the random numbers from output sequences of ideal RNGs.

Application Note:

In FIPS 140-2 mode, libcryptsetup uses the ANSI X9.31 compliant DRNG provided with libgcrypt which is seeded by /dev/random during normal operation of the TOE.

Application Note:

In FIPS 140-2 mode, libcryptsetup uses the ANSI X9.31 compliant DRNG provided with libgcrypt which is seeded by /dev/urandom during initial installation time of the TOE. As no other user and no attacker is assumed to be present during the the initial installation time, /dev/urandom is considered to provide the same entropy as /dev/random.

6.1.1.22 Subset access control (FDP_ACC.1(PSO))

FDP_ACC.1.1 The TSF shall enforce the Persistent Storage Object Access Control Policy on

a) **Subjects: all subjects defined with the Security Policy Model**;

b) Objects:

i. Persistent Storage Objects of the following type **: all file system objects defined with the Security Policy Model**;

ii. **no other storage objects**;

c) Operations: **read, write, execute (regular files), search (directories).**

6.1.1.23 Subset access control (FDP_ACC.1(TSO))

FDP_ACC.1.1 The TSF shall enforce the Transient Storage Object Access Control Policy on

a) **Subjects: all subjects defined with the Security Policy Model**;

b) Objects:

i. Transient Storage Objects of the following type **: all IPC objects defined with the Security Policy Model**;

ii. **no other storage objects**;

c) Operations: **read, receive, write, send.**

6.1.1.24 Security attribute based access control (FDP_ACF.1(PSO))

FDP_ACF.1.1 The TSF shall enforce the Persistent Storage Object Access Control Policy to objects based on the following:

a) Subject security attributes: file system UID, file system GID, supplemental GIDs;

b) Object security attributes: owning UID, owning GID;

c) Access control security attributes maintained for each file system object governing access to that object:

i. ACL for specific UIDs (ACL_USER),

ii. ACL for specific GIDs (ACL_GROUP),

iii. Maximum ACL for the file system object (ACL_MASK),

iv. Permission bits for the owning UID (equals to ACL_USER_OBJ when using ACLs),

v. Permission bits for the owning GID (equals to ACL_GROUP_OBJ when using ACLs),

vi. Permission bits for "world" (equals to ACL_OTHER when using ACLs),

vii. The following permission bits: read, write, execute (for files), search (for directories),

viii. The following access rights applicable to the file system object: SAVETXT (directories), immutable (files),

d) Access control security attributes maintained for each partition holding a file system: read-only;

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

A subject must have search permission for every element of the pathname and the requested access for the object. A subject has a specific type access to an object if one of the following rules hold (the order of the rules is applicable on a first-match basis):

a) The subject's filesystem UID is identical with the owning UID of the object and the requested type of access is within the permission bits defined for the owning UID (permission bits) or by ACL_USER_OBJ (ACLs); or

b) ACLs: The subject's filesystem UID is identical with the UID specified with ACL_USER of the object and the requested type of access is within the permission bits defined in ACL_USER; or

c) The subject's filesystem GID or one of the subject's supplemental GIDs identical with the owning GID and the requested type of access is within the permission bits defined for the owning GID (permission bits), or by ACL_GROUP_OBJ when there is no ACL_MASK entry (ACLs), or by the ACL_MASK entry (ACLs); or

d) ACLs: The subject's filesystem GID or one of the subject's supplemental GIDs is identical with the GID specified with ACL_GROUP of the object and the requested type of access is within the permission bits defined in ACL_GROUP; or

e) The requested type of access is within the permission bits defined for "world" (permission bits) or by ACL_OTHER (ACLs).

Application Note: *The permission bits and the ACLs are inherently consistent as the TOE assigns the permission bits to ACLs when ACLs are used. Without any ACLs specified for an object, the TOE only uses the permission bits. If at least one ACL is present or when the ACL management tools are applied for objects even without any ACL set, the permission bits are interpreted as outlined above: the ACL entry of ACL_USER_OBJ contains the owning UID permission bits, the ACL entry of ACL_GROUP_OBJ contains the owning GID permission bits, and the ACL entry of ACL_OTHER contains the permission bits for "world". The ACL entries of ACL_USER_OBJ, ACL_GROUP_OBJ and ACL_OTHER are only a different representation of the permission bits to users, they are not separate attributes in addition to permission bits. The explicit specification of ACL_USER_OBJ, ACL_GROUP_OBJ and ACL_OTHER in the rule set above in addition to the permission bits is only intended to aid the evaluator or reader in understanding the overall ruleset.*

Application Note: *Due to the fact that the permission bits are an inherent part of the ACLs, there is no precedence issue between permission bits and ACLs.*

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules:

a) read and directory search operations are allowed for the subject with the capability of CAP_DAC_READ_SEARCH;

b) write and execute operations are allowed for the subject with the capability of CAP_DAC_OVERRIDE - the execute permission is granted if the file system object object is marked with at least one executable bit in its permission settings.

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to named objects based on the following rules:

a) Any file system object in a file system that is mounted as read-only cannot be modified, created or removed,

b) A file, a directory and a symbolic link in a file system that is mounted as read-only cannot be written to,

c) Any file system object marked as immutable cannot be modified or removed,

d) Any file system object stored in a directory marked with the SAVETXT bit cannot be modified or removed by subjects whose file system UID is not equal to the owning UID of the file system object unless the subject performing the operation possesses the CAP_FOWNER capability.

6.1.1.25 Security attribute based access control (FDP_ACF.1(TSO))

FDP_ACF.1.1 The TSF shall enforce the Transient Storage Object Access Control Policy to objects based on the following:

a) Subject security attributes: effective UID, file system UID, effective GID, file system GID, supplemental GIDs;

b) Object security attributes: owning UID, owning GID;

c) Access control security attributes maintained for each IPC object whose name is managed with a file governing access to that object: see FDP_ACF.1(PSO);

d) Access control security attributes maintained for any other IPC object governing access to that object:

i. Permission bits for the owning UID,

ii. Permission bits for the owning GID,

iii. Permission bits for "world",

iv. The following permission bits: read, write, execute,

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

a) IPC object whose name is managed with a file: see FDP_ACF.1(PSO);

b) Any other IPC object: A subject has a specific type access to an object if one of the following rules hold (the order of the rules is applicable on a first-match basis):

1. The subject's effective UID is identical with the owning UID of the object and the requested type of access is within the permission bits defined for the owning UID; or

2. The subject's effective GID or one of the subject's supplemental GIDs identical with the owning GID and the requested type of access is within the permission bits defined for the owning GID; or

3. The requested type of access is within the permission bits defined for "world".

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules:

a) IPC object whose name is managed with a file: see FDP_ACF.1(PSO);

b) Any other IPC object:

1. read and receive operations are allowed for the subject with the capability of CAP_DAC_READ_SEARCH;

2. write and send operations are allowed for the subject with the capability of CAP_DAC_OVERRIDE.

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to named objects based on the following rules:

a) IPC object whose name is managed with a file: see FDP_ACF.1(PSO);

b) Any other IPC object: none.

6.1.1.26 Complete information flow control (FDP_IFC.2(NI))

FDP_IFC.2.1 The TSF shall enforce the Network Information Flow Control Policy on

a) Subjects:

i. unauthenticated external IT entities that send and receive information mediated by the TOE;

ii. **standard Linux processes** that send and receive information mediated by the TOE;

b) Information:

i. Network data routed through the TOE;

ii. **No other information;**

and all operations that cause that information to flow to and from subjects covered by the SFP.

FDP_IFC.2.2 The TSF shall ensure that all operations that cause any information in the TOE to flow to and from any subject in the TOE are covered by an information flow control SFP.

Application Note: *The SFR applies only to host systems which implements the packet filtering functionality.*

6.1.1.27 Simple security attributes (FDP_IFF.1(NI-IPTables))

FDP_IFF.1.1 The TSF shall enforce the Network Information Flow Control Policy based on the following types of subject and information security attributes:

a) ~~Object~~*Information* security attribute: the logical or physical network interface through which the network data entered the TOE;

b) **TCP/IP information security attributes:**

i. Source and destination IP address,

ii. Source and destination TCP port number,

iii. Source and destination UDP port number,

iv. Network protocol of IP, TCP, UDP, ICMP

v. TCP header flags of SYN, ACK, FIN, RST, URG, PSH

vi. TCP sequence numbers;

c) **Information security attribute: IP packet identified as to be routed by the TSF.**

Application Note: *The refinement is applied due to an obvious error in the OSPP.*

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: **If the packet filter matches the analyzed packet and the rule accepts the packet, the packet is forwarded according to the network protocol stack's behavior** .

FDP_IFF.1.3 The TSF shall enforce the following rules:
Identification of network data using one or more of the following concepts:

a) Information security attribute matching;

b) **Matching based on the state of a TCP connection, Statistical analysis matching**;

Performing one or more of the following actions with identified network data:

a) Discard the network data **without any further processing, with sending a notification to the sender**;

b) Allow the network data to be processed unaltered by the TOE according to the routing information maintained by the TOE;

c) **No other actions**.

FDP_IFF.1.4 The TSF shall explicitly authorise an information flow based on the following rules: **If the network data is not matched by the rule set and the default rule of the packet filter is ACCEPT then the data is forwarded unaltered based on the normal operation of the host system's networking stack** .

FDP_IFF.1.5 The TSF shall explicitly deny an information flow based on the following rules: **If the network data is not matched by the rule set, one of the following default rules applies:**

a) DROP: the data is discarded;

b) REJECT: then the data is discarded and a notification is returned to the sender.

Application Note: *The default rule is configurable where exactly one of the above mentioned default rules can be selected at any given time.*

Application Note: *The SFRs FDP_IFF.1(NI-iptables) and FDP_IFF.1(NI-ebtables) both define different rule sets implemented by the TOE covering the FDP_IFF.1(NI) SFR from the OSPP base.*

6.1.1.28 Simple security attributes (FDP_IFF.1(NI-ebtables))

FDP_IFF.1.1 The TSF shall enforce the Network Information Flow Control Policy based on the following types of subject and information security attributes:

a) ~~Object~~*Information* security attribute: the logical or physical network interface through which the network data entered the TOE;

b) **TCP/IP information security attributes:**

i. Source and destination IP address,

ii. Source and destination TCP port number,

iii. Source and destination UDP port number,

iv. Network protocol of IP, TCP, UDP

v. TCP header flags of TOS

vi. Ethernet frames security attributes:

i. Source and destination MAC address,

ii. Ethernet protocol type.

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: **If the packet filter matches the analyzed packet and the rule accepts the packet, the packet is forwarded according to the network protocol stack's behavior** .

FDP_IFF.1.3 The TSF shall enforce the following rules:
Identification of network data using one or more of the following concepts:

a) Information security attribute matching;

b) **no other rules**;

Performing one or more of the following actions with identified network data:

a) Discard the network data **without any further processing**;

b) Allow the network data to be processed unaltered by the TOE according to the routing information maintained by the TOE;

c) **No other actions**.

FDP_IFF.1.4 The TSF shall explicitly authorise an information flow based on the following rules: **If the network data is not matched by the rule set and the default rule of the packet filter is ACCEPT then the data is forwarded unaltered based on the normal operation of the host system's networking stack** .

FDP_IFF.1.5 The TSF shall explicitly deny an information flow based on the following rules: **If the network data is not matched by the rule set, and the default rule of the packet filter is DROP then the data is discarded.**

Application Note: *The default rule is configurable where exactly one of the above mentioned default rules can be selected at any given time.*

6.1.1.29 Import of user data with security attributes (FDP_ITC.2(BA))

FDP_ITC.2.1 The TSF shall enforce the Persistent Storage Access Control Policy, Transient Storage Access Control Policy,Network Information Flow Control, **no other access control SFP(s) and/or information flow control SFP(s)** when importing user data, controlled under the SFP, from outside of the TOE.

FDP_ITC.2.2 The TSF shall use the security attributes associated with the imported user data.

FDP_ITC.2.3 The TSF shall ensure that the protocol used provides for the unambiguous association between the security attributes and the user data received.

FDP_ITC.2.4 The TSF shall ensure that interpretation of the security attributes of the imported user data is as intended by the source of the user data.

FDP_ITC.2.5 The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: **No additional importation control rules.**

6.1.1.30 Full residual information protection (FDP_RIP.2)

FDP_RIP.2.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the **allocation of the resource to** all objects.

6.1.1.31 Full residual information protection of resources (FDP_RIP.3)

FDP_RIP.3.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the **allocation of the resource to** all subjects or users.

Application Note: *The subject is represented by the data structures inside the kernel forming a process: all data structures anchored in the task_struct. The user is represented by its attributes defined by FIA_ATD.1.*

6.1.1.32 Authentication failure handling (FIA_AFL.1)

FIA_AFL.1.1 The TSF shall detect when an administrator-configurable number of unsuccessful authentication attempts for the authentication method **of password-based authentication** occur related to **consecutive unsuccessful authentication attempts**.

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been **met, surpassed**, the TSF shall

a) For all administrator accounts, "disable" the account for an authorized administrator configurable time period such that there can be no more than ten attempts per minute.

b) For all other accounts, disable the user logon account until it is re-enabled by the authorized administrator.

c) For all disabled accounts, any response to an authentication attempt given to the user shall not be based on the result of that authentication attempt.

6.1.1.33 User attribute definition (FIA_ATD.1(HU))

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual human users:

a) User identifier;

b) Group memberships;

c) User password;

d) Software token verification data;

e) Security roles;

f) **no other security attribute;**

Application Note: *Please see the application note for FIA_UAU.5 for a list of token-based authentication mechanisms and their associated tokens.*

6.1.1.34 User attribute definition (FIA_ATD.1(TU))

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual technical users:

a) the logical or physical network interface through which the network data entered the TOE;

b) identity of the logical or physical external interface through which the user connected to the TOE;

c) **no other attributes;**

Application Note: *Bullet a) of this SFR relates to FDP_IFC.2(NI) and the supporting information flow control rules specified with the iterations of FDP_IFF.1. In the Common Criteria scheme, external entities are always considered to be users. Therefore, every network data entity must be specified as user in this ST.*

6.1.1.35 Verification of secrets (FIA_SOS.1)

FIA_SOS.1.1 The TSF shall provide a mechanism to verify that secrets meet the following quality metric: the probability that a secret can be obtained by an attacker during the lifetime of the secret is less than 2^-20.

Application Note: *The TOE password change is implemented using the PAM library. The PAM module pam_passwdqc.so allows the specification of the quality of new passwords. The evaluated configuration requires a configuration of the PAM-based password change mechanism that meets the above mentioned criteria.*

Application Note: *The Evaluated Configuration Guide contains configuration suggestions for the password quality mechanism that covers the above mentioned probability. These configuration suggestions assume the worst-case scenario when attacking these settings.*

Application Note: *For key-based authentication methods, the evaluation of the RSA and DSA keys used for the SSH protocol will show the maximum lifetime of a key depending on its size.*

6.1.1.36 Timing of authentication (FIA_UAU.1)

FIA_UAU.1.1 The TSF shall allow

a) the information flow covered by the Network Information Flow Control Policy;

b) **Local console log-in: banner information;**

c) **SSH log-in: obtaining the list of allowed authentication methods;**

on behalf of the user to be performed before the user is authenticated.

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

6.1.1.37 Multiple authentication mechanisms (FIA_UAU.5)

FIA_UAU.5.1 The TSF shall provide the following authentication mechanisms:

a) Authentication based on username and password;

b) Authentication based on software token verification data;

c) **No other authentication mechanisms**

to support user authentication.

Application Note: *The TOE is able to maintain the following types of software tokens and their verification data:*

- *SSH user keys: The TOE as server part is able to store the public part of the SSH user key for the user account the user wants to access.*

FIA_UAU.5.2 The TSF shall authenticate any user's claimed identity according to the following rules:

a) Authentication based on username and password is performed for TOE-originated requests and credentials stored by the TSF;

b) Authentication based on software token verification data is performed for TOE-originated requests;

c) **For SSH, both, the password-based and key-based authentication methods can be enabled at the same time. In this case, the key-based authentication method is tried before the password-based authentication. If the key-based authentication succeeds, the user is authenticated. If the key-based authentication fails, the password-based authentication is applied. If the password-based authentication fails, the user login request is denied.**

6.1.1.38 Protected authentication feedback (FIA_UAU.7)

FIA_UAU.7.1 The TSF shall provide only obscured feedback to the user while the authentication is in progress.

6.1.1.39 Timing of identification (FIA_UID.1)

FIA_UID.1.1 The TSF shall allow

a) Console log-in: banner information;

b) SSH log-in: obtaining the list of allowed authentication methods;

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.

6.1.1.40 Enhanced user-subject binding (FIA_USB.2)

FIA_USB.2.1 The TSF shall associate the following user security attributes with subjects acting on the behalf of that user:

a) The user identity that is associated with auditable events;

b) The user security attributes that are used to enforce the Persistent Storage Object Access Control Policy;

c) The user security attributes that are used to enforce the Transient Storage Object Access Control Policy;

d) The software token that can be used for subsequent identification and authentication with the TSF or other remote IT systems;

e) Active roles;

f) Active groups;

g) **no other security attributes;**

FIA_USB.2.2 The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users:

a) Upon successful identification and authentication, the login UID, the real UID, the filesystem UID and the effective UID shall be those specified in the user entry for the user that has authenticated successfully;

b) Upon successful identification and authentication, the real GID, the filesystem GID and the effective GID shall be those specified via the primary group membership attribute in the user entry;

c) Upon successful identification and authentication, the supplemental GIDs shall be those specified via the supplemental group membership assignment for the user entry;

Application Note: *The various subject UIDs are all derived from the same numeric UID per user entry stored in the /etc/passwd file.*

Application Note: *The various subject GIDs except the supplemental GIDs are all derived from the same numeric GID per user entry stored in the /etc/passwd file.*

Application Note: *The subject's supplemental GIDs are derived from the username to group name mappings in the /etc/group file. As the TOE only maintains numeric IDs for subjects, the username and the group names need to be converted before instantiating the subject. The username to UID mapping is provided in /etc/passwd and the group name to GID mapping is provided in /etc/group.*

FIA_USB.2.3 The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users:

- **a) The effective and filesystem UID of a subject can be changed by the use of an executable with the SETUID bit set. In this case the program is executed with the effective and filesystem UID of the owning UID of the file storing the program. These newly set effective and filesystem UIDs are used for the DAC permission validation. The real and login UID remain unchanged.**
- **b) The effective and filesystem GID of a subject can be changed by the use of an executable with the SETGID bit set. In this case the program is executed with the effective and filesystem GID of the owning GID of the file storing the program. These newly set effective and filesystem GIDs are used for the DAC permission validation. The real GID remains unchanged.**
- **c) The real, effective and filesystem UID of a subject can be changed by the use of the set*uid system call family for the calling application. These system calls are restricted to processes possessing the CAP_SETUID capability.**
- **d) The real, effective and filesystem GID of a subject can be changed by the use of the set*gid system call family for the calling application. These system calls are restricted to processes possessing the CAP_SETUID capability.**
- **e) The set of supplemental GIDs of a subject can be changed by the use of the setgroups system call for the calling application. These system calls are restricted to processes possessing the CAP_SETUID capability.**

Application Note: *The applications "su" and "sudo" allow the calling user to change the filesystem and effective UID either to root or to other users provided the authentication to "su" or "sudo" was successful. Both application uses the SETUID bit with the owning UID of root as well as the set*uid system calls to change to other UIDs before spawning a new shell or the given command. As both applications rest on the above mentioned mechanisms, it is not listed as a separate mechanism to modify the calling user's UIDs.*

Application Note: *The login UID is set by the PAM modules by inserting the intended UID into the /proc/<PID>/loginuid file. This file can be written to only by subjects executing with the effective UID of zero (root) and only for the calling process' own loginuid file. However, there is no application except the PAM modules which access that proc file which implies that the login UID remains unchanged after login when operating the TOE. Authorized administrators are not intended to access that proc file.*

FIA_USB.2.4 The TSF shall enforce the following rules for the assignment of subject security attributes not derived from user security attributes when a subject is created: **No rules**.

6.1.1.41 Failure with preservation of secure state - full buffer overflow protection (FPT_FLS.1(FULL))

FPT_FLS.1.1 The TSF shall preserve a secure state when the following types of failures occur:

a) Execution of code on a process' or thread's stack;

b) Modification of a function return address on the process' or thread's stack to jump to previously known processor instructions;

c) Modification of process section other than the segment that holds compile time initialized data and segment holding the mapping of all uninitialized variables

for the runtime instances of the following binaries:

i. all user-provided applications and their depending libraries that are compiled and linked with the following properties:

1. presence of the ELF program header entry of PT_GNU_STACK and the absense of the PF_X bit in the p_flags ELF header flags;

2. presence of the ELF program header entry of PT_GNU_RELRO with the memory range information covering the following ELF sections: .tdata, .preinit_array, .init_array, .fini_array, .ctors, .dtors, .jcr, .data.rel.ro, .dynamic, .got including .got.plt;

Application Note: *The secure state implied with this functionality covers the following aspects where the following list explains the implication of each bullet above:*

a) When exploiting buffer overruns of an application, the attacker cannot feed code onto the stack and execute it.

b) When exploiting buffer overruns of an application, the modify of the return address stored on a stack to jump to a previously known code segment is much harder to achieve by an attacker. Due to the address randomization, any memory address of code already present with the application or loaded libraries will be different with each startup of the application.

c) The ELF header sections listed above are set read-only using the mprotect system call by the loader before the application gains control. When exploiting buffer overruns, the attacker cannot modify information in those memory sections. These sections store offset tables required for the dynamic linking mechanism and, if abused, allow attackers to modify the jump addresses of object accesses. Full protection against this type of attack can only be achieved if the application and all depending shared libraries are compiled linked with full protection enabled. When at least one shared library the application depends on or the application itself is compiled and linked with partial protection (see FPT_FLS.1(PARTIAL)), only partial protection against this type of attack is available for the given application.

Application Note: *To enforce the functionality in bullet a), /proc/sys/kernel/exec-shield must contain either a 1 or 2.*

Application Note: *To enforce the functionality in bullet b), /proc/sys/kernel/randomize_va_space must contain a 2 (1 implies that the address randomization is enabled except for the brk system call).*

Application Note: *During standard compilation of applications, the stack execution protection is enabled. To ensure the presence of the PT_GNU_STACK ELF program header entry and the absence of the PF_X bit in the p_flags ELF header flags, the following considerations must be applied by a programmer as any of the following operations disable the stack execution protection:*

- *The following linker option must **not** be used: "-z execstack" (gcc: "-Wl,-z,execstack").*

- *The following assembler option must **not** be used: "--execstack" (gcc: "-Wa,--execstack").*
- *Modifications of an ELF program header entry in an already compiled binary which change the PT_GNU_STACK and PF_X flags (like using the execstack(8) application) must **not** be performed.*
- *The application or library code must not contain trampolines such as nested functions pushed onto the stack which passed as pointers to functions as this would also enable the stack execution support.*

Application Note: *To ensure the presence of PT_GNU_RELRO covering the proper ELF sections, the application must be linked with the provided linker using the linker options of "-z relro -z now" (using the GCC compiler using the compiler options of "-Wl,-z,relro,-z,now" can be used which are passed to the linker). In addition, an application must be compiled as PIE with the gcc option of "-fPIE". Contrary, a shared library must be compiled as PIC with the gcc option of "-fPIC".*

6.1.1.42 Failure with preservation of secure state - partial buffer overflow protection (FPT_FLS.1(PARTIAL))

FPT_FLS.1.1 The TSF shall preserve a secure state when the following types of failures occur:

a) Execution of code on a process' or thread's stack;

b) Modification of a function return address on the process' or thread's stack to jump to previously known processor instructions;

c) Modification of process sections other than the segment that holds compile time initialized data, the segment holding the mapping of all uninitialized variables, and the procedure linking table (PLT)

for:

i. the runtime instances of the following applications as a subset of the depending libraries provide partial protection:

1. **sshd**
2. **rsyslogd and associated modules**
3. **udevd**
4. **xinetd**
5. **crond**
6. **aide**
7. **sftp-server**
8. **su**

ii. all libraries in the TSF

iii. all user-provided applications and their depending libraries that are compiled and linked with the following properties:

1. **presence of the ELF program header entry of PT_GNU_STACK and the absense of the PT_X bit in the p_flags ELF header flags;**
2. **presence of the ELF program header entry of PT_GNU_RELRO with the memory range information covering the following ELF sections: .tdata, .preinit_array, .init_array, .fini_array, .ctors, .dtors, .jcr, .data.rel.ro, .dynamic, .got excluding .got.plt;**

Application Note: *The only difference between full and partial RELRO is that in partial RELRO the .got.plt ELF section is left unprotected and is therefore read/writable. This difference allows lazy bindings during the dynamic linking process.*

Application Note: *All application notes from FPT_FLS.1(FULL) apply except the following.*

Application Note: *To ensure the presence of PT_GNU_RELRO covering the proper ELF sections, the application must be linked with the provided linker using the linker options of "-z relro" (using the GCC compiler using the compiler options of "-Wl,-z,relro" can be used which are passed to the linker). In addition, an application must be compiled as PIE with the gcc option of "-fPIE". Contrary, a shared library must be compiled as PIC with the gcc option of "-fPIC".*

6.1.1.43 Reliable time stamps (FPT_STM.1)

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps.

6.1.1.44 Inter-TSF basic TSF data consistency (FPT_TDC.1(BA))

FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret **the following data types:**

a) Packet filter: protocol headers for the network protocols covered by the packet filter;

when shared between the TSF and another trusted IT product.

FPT_TDC.1.2 The TSF shall use **the following interpretation rules:**

a) Packet filter: protocol headers specification provided in RFC 791 (IP), RFC 793 (TCP), RFC 768 (UDP), RFC 792 (ICMP);

when interpreting the TSF data from another trusted IT product.

6.1.1.45 TSF-initiated session locking (FTA_SSL.1)

FTA_SSL.1.1 The TSF shall lock an interactive session to a human user maintained by the TSF after **an administrator-configurable time interval of user inactivity** by:

a) clearing or overwriting TSF controlled display devices, making the current contents unreadable;

b) disabling any activity of the user's TSF controlled access/TSF controlled display devices other than unlocking the session.

Application Note: *The management aspect of configuring the time interval is covered by FMT_MTD.1(SSL).*

FTA_SSL.1.2 The TSF shall require the following events to occur prior to unlocking the session:

a) Successful re-authentication with the credentials of the user owning the session using **password based authentication**;

b) **No other events** .

6.1.1.46 User-initiated locking (FTA_SSL.2)

FTA_SSL.2.1 The TSF shall lock an interactive session maintained by the TSF, by:

a) clearing or overwriting TSF controlled display devices, making the current contents unreadable;

b) disabling any activity of the user's TSF controlled data access/TSF controlled display devices other than unlocking the session.

FTA_SSL.2.2 The TSF shall require the following events to occur prior to unlocking the session:

a) Successful re-authentication with the credentials of the user owning the session using **password based authentication**;

b) **No other events** .

6.1.1.47 Inter-TSF trusted channel (FTP_ITC.1)

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 and disclosure using the following mechanisms:

a) Cryptographically-protected communication channel using **SSH protocol version 2** ;

b) **No other trusted communication channel.**

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 **no other conditions or functions**.

Application Note: *The SSH protocol implements a bi-directional authentication mechanism as follows:*

- *Server-side authentication: the user identification and authentication via user name and password / SSH user key allows the server to authenticate the client.*
- *Client-side authentication: the SSH host key verification performed by the SSH client during each connection attempt allows the client to authenticate the server.*

6.1.2 Confidentiality protection of data at rest

6.1.2.1 Complete access control (FDP_ACC.2(CP))

FDP_ACC.2.1 The TSF shall enforce the **Confidentiality Access Control Policy** on

a) Subjects: all subjects defined with the Security Policy Model

b) Objects:

i. Persistent Storage Objects of the following type : all file system objects defined with the Security Policy Model.

and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.2 The TSF shall ensure that all operations between any subject controlled by the TSF and any object controlled by the TSF are covered by an access control SFP.

6.1.2.2 Security attribute based access control (FDP_ACF.1(CP))

FDP_ACF.1.1 The TSF shall enforce the **Confidentiality Access Control Policy** to objects based on the following:

- **a) Subject security attributes: none as all subjects maintained by the TOE are covered;**
- **b) Persistent storage object security attributes: all persistent storage objects located on the protected block device;**
- **c) Block device object security attributes: session key used to encrypt and decrypt and data processed on that block device;**
- **d) User security attributes: passphrase that protects the session key using the LUKS protection mechanism.**

Application Note: *The SFR mentions two different object attributes that are relevant to the security policy. The first is the session key used to encrypt data stored on the block device. However, file system objects (which contain the information the user wants to protect) are only covered by the encryption, if they are stored on the encrypted block device. Therefore, the storage location of the file system objects is another object security attribute as it decides about the protection status of the object.*

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

- **a) Access granting when TSF are active: Every user with access to the mount point of the encrypted block device is granted access when the encrypted block device is unlocked and mounted;**
- **b) Access granting when TSF are active: Every user not in the possession of the passphrase to unlock the encrypted block device is denied access to data stored on that block device;**
- **c) Access granting when TSF are inactive: Every user not in the possession of the passphrase to unlock the encrypted block device is denied access to data stored on that block device.**

Application Note: *The TOE provides the dm_crypt mechanism as a block device encryption. When the session key for the encryption and decryption operation is provided to the kernel, the encrypted block device is unlocked. At this point, the contents - the file system - is accessible to the kernel and can be mounted. If the session key is locked, all data is encrypted on the block device with a symmetric of either Triple-DES or AES.*

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: **no explicit access authorization to any subject.**

Application Note: *When the block device is unlocked and mounted, it behaves exactly the same way as any other mounted file system. Note that any file system specific access control mechanisms like permission bits, ACLs are added to the protection mechanism*

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **none.**

6.1.2.3 Confidentiality for data at rest (FDP_CDP.1(CP))

FDP_CDP.1.1 The TSF shall enforce the **Confidentiality Access Control Policy** to store user data at rest in containers controlled by the TSF in a manner protected from unauthorised disclosure.

6.1.3 Management related functionality

6.1.3.1 Management of object security attributes (FMT_MSA.1(PSO))

FMT_MSA.1.1 The TSF shall enforce the Persistent Storage Object Access Control Policy to restrict the ability to modify, **change_default** the security attributes of the objects covered by the SFP to the owner of the object and **users with processes granted the CAP_CHOWN, CAP_FOWNER, CAP_FSETID capabilities**.

6.1.3.2 Management of object security attributes (FMT_MSA.1(TSO))

FMT_MSA.1.1 The TSF shall enforce the Transient Storage Object Access Control Policy to restrict the ability to modify the security attributes of the objects covered by the SFP to the owner of the object and **users with processes granted the CAP_FOWNER capability**.

6.1.3.3 Management of security attributes (FMT_MSA.1(CP))

FMT_MSA.1.1 The TSF shall enforce the **Confidentiality Access Control Policy** to restrict the ability to **modify, transfer, delete** the security attributes **of the block device objects covered by the SFP** to **the owner of the object**.

Application Note: *The SFR applies to the management of the session key that encrypts the data on the block device. Only the owner, i.e. the user that is in possession of the passphrase protecting the session, is able to modify the key, to transfer it (i.e. to protect it with an additional passphrase) or to delete the session key.*

6.1.3.4 Static attribute initialisation (FMT_MSA.3(PSO))

FMT_MSA.3.1 The TSF shall enforce the Persistent Storage Object 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

a) root user for a global setting applied during logon;

b) each user for a setting applicable to his processes;

c) users with write permissions to a directory for setting default ACLs

to specify alternative initial values to override the default values when an object or information is created.

Application Note: *The global default value for permission bits is specified with the umask value which specifies the permission bits for newly created objects. This value has an initial setting of 022 or the value specified in /etc/login.defs. Only the root user can manage that initial value as this file is writeable to root only. Users can change their umask value at any time using the umask(2) system call. For ACLs, the default ACL is provided for for the root directory which, in case of absence of a default ACL entry is consistent with the umask.*

6.1.3.5 Static attribute initialisation (FMT_MSA.3(TSO))

FMT_MSA.3.1 The TSF shall enforce the Transient Storage Object 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

a) root user for a global setting applied during logon;

b) each user for a setting applicable to his processes

to specify alternative initial values to override the default values when an object or information is created.

Application Note: *The global default value for permission bits is specified with the umask value which specifies the permission bits for newly created objects. This value has an initial setting of 022 or the value specified in /etc/login.defs. Only the root user can manage that initial value as this file is writeable to root only. Users can change their umask value at any time using the umask(2) system call.*

6.1.3.6 Static attribute initialisation (FMT_MSA.3(NI))

FMT_MSA.3.1 The TSF shall enforce the Network Information Flow Control Policy to provide **permissive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the **users with processes granted the CAP_NET_ADMIN capability** to specify alternative initial values to override the default values when an object or information is created.

Application Note: *The default value specified in this SFR applies to the default for the packet filter after boot. The administrator can configure alternative default values as outlined in FDP_IFF.1(NI-IPTables) as well as FDP_IFF.1(NI-ebtables).*

Application Note: *The iptables and ebtables commands use a netlink interfact to the kernel which requires that the caller possesses the CAP_NET_ADMIN capability.*

6.1.3.7 Static attribute initialisation (FMT_MSA.3(CP))

FMT_MSA.3.1 The TSF shall enforce the **Confidentiality 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 **nobody** to specify alternative initial values to override the default values when an object or information is created.

Application Note: *Restrictive default values apply to the protection of the session key: the session key is created and immediately protected with a passphrase. Therefore, only the creator of the session key is initially able to access the locked block device.*

6.1.3.8 Security attribute value inheritance (FMT_MSA.4(PSO))

FMT_MSA.4.1 The TSF shall use the following rules to set the value of security attributes for Persistent Storage Objects:

a) The newly created object's owning UID is set to the effective UID of the calling subject;

b) The newly created object's owning GID is set to the effective GID of the calling subject with the following exception for file system objects: if the parent directory holding the newly created file system object is marked with the SETGID permission bit, the owning GID of the newly created file system object is set to the owning GID of the parent directory;

c) The newly created object's permission bits are derived from the calling subject's umask value by masking out the umask bits from the permission bit set granting full access;

d) The newly created object's ACLs are derived from the default ACL specified for the parent directory the newly created file system object is stored in, if existant. Otherwise, no ACL is set.

.

6.1.3.9 Management of TSF data (FMT_MTD.1(AE))

FMT_MTD.1.1 The TSF shall restrict the ability to query, modify the set of audited events to **processes with the capability CAP_AUDIT_CONTROL**.

Application Note: *This SFR applies to FAU_SEL.1.*

Application Note: *Using the audit tools which in turn use the netlink interface, an administrator can configure the audit rules.*

6.1.3.10 Management of TSF data (FMT_MTD.1(AS))

FMT_MTD.1.1 The TSF shall restrict the ability to clear **delete, configure the storage location** the audit storage to **the root user**.

Application Note: *This SFR applies to FAU_STG.1 where the directory used for storing the audit trail is configured.*

Application Note: *The configuration of these parameters is performed with the configuration file /etc/auditd/auditd.conf which is writeable to the root user only.*

6.1.3.11 Management of TSF data (FMT_MTD.1(AT))

FMT_MTD.1.1 The TSF shall restrict the ability to modify **add, delete** the

a) threshold of the audit trail when an action is performed;

b) action when the threshold is reached

to **the root user**.

Application Note: *This SFR applies to FAU_STG.3.*

Application Note: *The configuration of these parameters is performed with the configuration file /etc/auditd/auditd.conf which is writeable to the root user only.*

6.1.3.12 Management of TSF data (FMT_MTD.1(AF))

FMT_MTD.1.1 The TSF shall restrict the ability to modify **add, delete** the actions to be taken in case of audit storage failure to **the root user**.

Application Note: *This SFR applies to FAU_STG.4.*

Application Note: *The configuration of these parameters is performed with the configuration file /etc/auditd/auditd.conf which is writeable to the root user only.*

6.1.3.13 Management of TSF data (FMT_MTD.1(NI))

FMT_MTD.1.1 The TSF shall restrict the ability to query, modify, delete **change_default** the security attributes for the rules governing the

a) identification of network packets by the packet filter;

b) actions performed on the identified network packets by the packet filter;

to **users with processes granted the CAP_NET_ADMIN capability**.

Application Note: *This SFR applies to FDP_IFF.1(NI).*

Application Note: *The iptables and ebtables commands use a netlink interfact to the kernel which requires that the caller possesses the CAP_NET_ADMIN capability.*

6.1.3.14 Management of TSF data (FMT_MTD.1(IAT))

FMT_MTD.1.1 The TSF shall restrict the ability to modify the threshold for unsuccessful authentication attempts to **the root user**.

Application Note: *This SFR applies to FIA_AFL.1.*

Application Note: *The configuration of these parameters is performed with the PAM configuration files which are writeable to the root user only.*

6.1.3.15 Management of TSF data (FMT_MTD.1(IAF))

FMT_MTD.1.1 The TSF shall restrict the ability to re-enable the authentication to the account subject to authentication failure to **the root user**.

Application Note: *This SFR applies to FIA_AFL.1.*

Application Note: *The account locking information is stored in the directory /var/log/faillock. Using the pam_faillock application which modifies this file, the account can be unlocked. The DAC permissions of that file ensure that only the root user can write to it.*

6.1.3.16 Management of TSF data (FMT_MTD.1(IAU))

FMT_MTD.1.1 The TSF shall restrict the ability to initialize, modify, delete the user security attributes to

a) the root user,

b) users authorized to modify their own authentication data

.

Application Note: *This SFR applies to FIA_ATD.1, FIA_UAU.1, and FIA_UID.1.*

Application Note: *The configuration of these parameters is performed with the configuration files /etc/passwd and /etc/shadow which are writeable to the root user only.*

6.1.3.17 Management of TSF data (FMT_MTD.1(SSH))

FMT_MTD.1.1 The TSF shall restrict the ability to **modify** the **authentication methods provided by the OpenSSH server** to **the root user**.

Application Note: *This SFR applies to FIA_UAU.5.*

Application Note: *The configuration of this parameter is performed with the configuration file /etc/sshd_config which is writeable to the root user only.*

6.1.3.18 Management of TSF data (FMT_MTD.1(SSL))

FMT_MTD.1.1 The TSF shall restrict the ability to **modify** the **time interval of user inactivity for locking an interactive session** to

a) the root user for system wide settings,

b) each user for his own sessions, if allowed by the root user.

Application Note: *This SFR applies to FTA.SSL.1.*

Application Note: *The time interval is configured in /etc/screenrc which is writeable to root only. Normal users can configure the time interval in ~/.screenrc. The screen application enforcing the session locking can be configured to execute with /etc/profile or /etc/login.csh. The root user can place screen execution commands in these Shell startup files that prevent the loading of ~/.screenrc.*

6.1.3.19 Management of TSF data [OSPP-AUD] (FMT_MTD.1(AUD-AE))

FMT_MTD.1.1 The TSF shall restrict the ability to query, modify the set of events audited by a remote trusted IT system to **processes with the capability CAP_AUDIT_CONTROL** .

Application Note: *This SFR applies to FAU_SEL.1 of the OSPP base.*

6.1.3.20 Management of TSF data [OSPP-AUD] (FMT_MTD.1(AUD-AS))

FMT_MTD.1.1 The TSF shall restrict the ability to clear **delete, configure the storage location** the audit storage configuration of a remote trusted IT system to **the root user**.

Application Note: *This SFR applies to FAU_STG.1.*

Application Note: *The configuration of these parameters is performed with the configuration file /etc/auditsp/audisp-remote.conf which is writeable to the root user only.*

6.1.3.21 Management of TSF data [OSPP-AUD] (FMT_MTD.1(AUD-AT))

FMT_MTD.1.1 The TSF shall restrict the ability to modify **add, delete** the following configurations of a remote trusted IT system:

a) threshold of the audit trail when an action is performed;

b) action when the threshold is reached

to **the root user**.

Application Note: *This SFR applies to FAU_STG.3 of the OSPP base.*

Application Note: *The configuration of these parameters is performed with the configuration file /etc/auditd/auditd.conf which is writeable to the root user only.*

6.1.3.22 Management of TSF data [OSPP-AUD] (FMT_MTD.1(AUD-AF))

FMT_MTD.1.1 The TSF shall restrict the ability to modify **add, delete** the actions to be taken by a remote trusted IT system in case of audit storage failure to **the root user** .

Application Note: *This SFR applies to FAU_STG.4 of the OSPP base.*

Application Note: *The configuration of these parameters is performed with the configuration file /etc/auditd/auditd.conf which is writeable to the root user only.*

6.1.3.23 Management of TSF data (FMT_MTD.1(CP-AN))

FMT_MTD.1.1 The TSF shall restrict the ability to **modify** the **confidentiality protection anchor** to **the owner of the dm-crypt partition** .

Application Note: *The trust anchor is the passphrases that protect the master key for a dm-crypt partition.*

6.1.3.24 Management of TSF data (FMT_MTD.1(CP-UD))

FMT_MTD.1.1 The TSF shall restrict the ability to **enable, disable** the **security attributes governing the enforcement of the Confidentiality Access Control Polity on an object** to **the owner of the dm-crypt partition** .

6.1.3.25 Revocation (FMT_REV.1(OBJ))

FMT_REV.1.1 The TSF shall restrict the ability to revoke object security attributes defined by SFPs associated with the corresponding object under the control of the TSF to

a) DAC permissions: owners of the object and authorized administrator;

b) Other security attributes: authorized administrator.

Application Note: *The privileges that constitute an authorized administrator are defined in the above mentioned FMT_* SFRs which specify the privileges needed to modify object security attributes. The same privileges are required to revoke these security attributes.*

FMT_REV.1.2 The TSF shall enforce the following rules:

a) The access rights associated with an object shall be enforced when an access check is made;

b) **no specification of other revocation rules**.

Application Note: *Revocation of security attributes for named objects imply the revocation of access granted to users other than the owner of the object. Note that the DAC ownership management (which can be also considered as a form of access revocation) is specified in FMT_MSA.1(PSO).*

6.1.3.26 Revocation (FMT_REV.1(USR))

FMT_REV.1.1 The TSF shall restrict the ability to revoke user security attributes defined by the SFP associated with the corresponding user under the control of the TSF to **authorized administrators** .

Application Note: *The privileges that constitute an authorized administrator are defined in the above mentioned FMT_* SFRs which specify the privileges needed to modify object security attributes. The same privileges are required to revoke these security attributes.*

FMT_REV.1.2 The TSF shall enforce the *following* rules *:*

a) The enforcement of the revocation of security-relevant authorizations with the next user-subject binding process during the next authentication of the user;

b) No other rules

Application Note: *The changes are enforced for a new session when the user affected by the change initiates that new session.*

6.1.3.27 Specification of management functions (FMT_SMF.1)

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions:

a) Management of auditing;

b) Management of cryptographic network protocols;

c) Management of Persistent Storage Object Access Control Policy;

d) Management of Network Information Flow Control Policy;

e) Management of identification and authentication policy;

f) Management of user security attributes;

g) **no other rules;**

Application Note: *The given list is kept generic intentionally. This ST specifies one iteration of FMT_MTD.1 per management function required by an SFR. For each FMT_MTD.1 iteration, a corresponding application note refers to the covered SFR(s).*

6.1.3.28 Security management roles (FMT_SMR.2)

FMT_SMR.2.1 The TSF shall maintain the roles:

a) User role with the following rights:

i. Users are authorized to modify their own user password;

ii. Users are authorized to modify the access control permissions for the named objects they own;

iii. **no other rights;**

b) **Configurations stored by user space: administrative users defined by the access permissions to the configurations mentioned in the other management SFRs;**

c) **Functions provided by the kernel: administrative users defined by capabilities mentioned in other management SFRs;**

FMT_SMR.2.2 The TSF shall be able to associate users with roles.

FMT_SMR.2.3 The TSF shall ensure that the conditions

a) No role-based access control

are satisfied.

Application Note: *Administrative actions can only be performed when the calling subject possesses the above mentioned capabilities which in the TOE configuration is only provided to processes executing with the effective UID or file system UID of zero (also called the root user). As the account for the root user is disabled for direct logon, authorized administrators are defined as users who are assigned to the "wheel" group. This group allows the use of the "su" application which is the only way to assume the root user capabilities. In addition, the "sudo" application allows granting users the privilege to execute commands with a different user ID, including the root user.*

Application Note: *This SFR is hierarchical to the PP SFR of FMT_SMR.1 which satisfies the strict conformance claim.*

6.2 Security Functional Requirements Rationale

6.2.1 Coverage

The following table provides a mapping of SFR to the security objectives, showing that each security functional requirement addresses at least one security objective.

Security functional requirements	Objectives
FAU_GEN.1	O.AUDITING
FAU_GEN.2	O.AUDITING
FAU_SAR.1	O.AUDITING
FAU_SAR.2	O.AUDITING
FAU_SAR.3(AUD)	O.ANALYZE_AUDIT
FAU_SEL.1	O.AUDITING
FAU_STG.1	O.AUDITING
FAU_STG.3	O.AUDITING
FAU_STG.4	O.AUDITING
FCS_CKM.1(SYM)	O.CRYPTO.NET
FCS_CKM.1(RSA)	O.CRYPTO.NET
FCS_CKM.1(DSA)	O.CRYPTO.NET
FCS_CKM.2(NET)	O.CRYPTO.NET
FCS_CKM.4	O.CRYPTO.NET
FCS_COP.1(NET)	O.CRYPTO.NET
FCS_COP.1(CP)	O.CP.USERDATA
FCS_RNG.1(SSH-DFLT)	O.CRYPTO.NET
FCS_RNG.1(SSH-FIPS)	O.CRYPTO.NET
FCS_RNG.1(DM-INIT)	O.CP.USERDATA

Security functional requirements	Objectives
FCS_RNG.1(DM-RUN)	O.CP.USERDATA
FCS_RNG.1(DM-FIPS)	O.CP.USERDATA
FDP_ACC.1(PSO)	O.DISCRETIONARY.ACCESS
FDP_ACC.1(TSO)	O.SUBJECT.COM
FDP_ACF.1(PSO)	O.DISCRETIONARY.ACCESS
FDP_ACF.1(TSO)	O.SUBJECT.COM
FDP_IFC.2(NI)	O.NETWORK.FLOW
FDP_IFF.1(NI-IPTables)	O.NETWORK.FLOW
FDP_IFF.1(NI-ebtables)	O.NETWORK.FLOW
FDP_ITC.2(BA)	O.DISCRETIONARY.ACCESS, O.NETWORK.FLOW, O.SUBJECT.COM
FDP_RIP.2	O.AUDITING, O.CRYPTO.NET, O.DISCRETIONARY.ACCESS, O.I&A, O.NETWORK.FLOW, O.SUBJECT.COM
FDP_RIP.3	O.AUDITING, O.CRYPTO.NET, O.DISCRETIONARY.ACCESS, O.I&A, O.NETWORK.FLOW, O.SUBJECT.COM
FIA_AFL.1	O.I&A
FIA_ATD.1(HU)	O.I&A
FIA_ATD.1(TU)	O.NETWORK.FLOW
FIA_SOS.1	O.I&A
FIA_UAU.1	O.I&A
FIA_UAU.5	O.I&A
FIA_UAU.7	O.I&A
FIA_UID.1	O.I&A, O.NETWORK.FLOW
FIA_USB.2	O.I&A
FPT_FLS.1(FULL)	O.RUNTIME.PROTECTION

Security functional requirements	Objectives
FPT_FLS.1(PARTIAL)	O.RUNTIME.PROTECTION
FPT_STM.1	O.AUDITING
FPT_TDC.1(BA)	O.DISCRETIONARY.ACCESS, O.NETWORK.FLOW, O.SUBJECT.COM
FTA_SSL.1	O.I&A
FTA_SSL.2	O.I&A
FTP_ITC.1	O.TRUSTED_CHANNEL
FDP_ACC.2(CP)	O.CP.USERDATA
FDP_ACF.1(CP)	O.CP.USERDATA
FDP_CDP.1(CP)	O.CP.USERDATA
FMT_MSA.1(PSO)	O.MANAGE
FMT_MSA.1(TSO)	O.MANAGE
FMT_MSA.1(CP)	O.CP.USERDATA
FMT_MSA.3(PSO)	O.MANAGE
FMT_MSA.3(TSO)	O.MANAGE
FMT_MSA.3(NI)	O.MANAGE
FMT_MSA.3(CP)	O.CP.USERDATA
FMT_MSA.4(PSO)	O.MANAGE
FMT_MTD.1(AE)	O.MANAGE
FMT_MTD.1(AS)	O.MANAGE
FMT_MTD.1(AT)	O.MANAGE
FMT_MTD.1(AF)	O.MANAGE
FMT_MTD.1(NI)	O.MANAGE
FMT_MTD.1(IAT)	O.MANAGE
FMT_MTD.1(IAF)	O.MANAGE
FMT_MTD.1(IAU)	O.MANAGE
FMT_MTD.1(SSH)	O.MANAGE
FMT_MTD.1(SSL)	O.MANAGE
FMT_MTD.1(AUD-AE)	O.REMOTE_AUDIT

Security functional requirements	Objectives
FMT_MTD.1(AUD-AS)	O.REMOTE_AUDIT
FMT_MTD.1(AUD-AT)	O.REMOTE_AUDIT
FMT_MTD.1(AUD-AF)	O.REMOTE_AUDIT
FMT_MTD.1(CP-AN)	O.CP.ANCHOR
FMT_MTD.1(CP-UD)	O.CP.USERDATA
FMT_REV.1(OBJ)	O.MANAGE
FMT_REV.1(USR)	O.MANAGE
FMT_SMF.1	O.MANAGE
FMT_SMR.2	O.MANAGE

Table 8: Mapping of security functional requirements to security objectives

6.2.2 Sufficiency

The following rationale provides justification for each security objective for the TOE, showing that the security functional requirements are suitable to meet and achieve the security objectives:

Security objectives	Rationale
O.AUDITING	The events to be audited are defined in [FAU_GEN.1] and are associated with the identity of the user that caused the event [FAU_GEN.2]. Authorized users are provided the capability to read the audit records [FAU_SAR.1], while all other users are denied access to the audit records [FAU_SAR.2]. The authorized user must have the capability to specify which audit records are generated [FAU_SEL.1]. The TOE prevents the audit log from being modified or deleted [FAU_STG.1] and ensures that the audit log is not lost due to resource shortage [FAU_STG.3, FAU_STG.4]. To support auditing, the TOE is able to maintain proper time stamps [FPT_STM.1]. The protection of reused resources ensures that no data leaks from other protected sources [FDP_RIP.2, FDP_RIP.3].
O.CRYPTO.NET	The cryptographically-protected network protocol [FCS_COP.1(NET)] is supported by the generation of symmetric keys [FCS_CKM.1(SYM)], as well as asymmetric keys [FCS_CKM.1(RSA), FCS_CKM.1(DSA)] where the functionality is based on the random number generator as defined by [FCS_RNG.1(SSH-DFLT), FCS_RNG.1(SSH-FIPS)]. As part of the cryptographic network protocol, the TOE securely exchanges the symmetric key with a remote trusted IT system [FCS_CKM.2(NET)]. The TOE ensures that all keys are zeroized upon de-allocation [FCS_CKM.4]. The protection of reused resources ensures that no data leaks from other protected sources [FDP_RIP.2, FDP_RIP.3].

Security objectives	Rationale
O.DISCRETIONARY.ACCESS	The TSF must control access to resources based on the identity of users that are allowed to specify which resources they want to access for storing their data. The access control policy must have a defined scope of control [FDP_ACC.1(PSO)]. The rules for the access control policy are defined [FDP_ACF.1(PSO)]. When import of user data is allowed, the TOE must ensure that user data security attributes required by the access control policy are correctly interpreted [FDP_ITC.2(BA), FPT_TDC.1(BA)]. The protection of reused resources ensures that no data leaks from other protected sources [FDP_RIP.2, FDP_RIP.3].
O.NETWORK.FLOW	The network information flow control mechanism controls the information flowing between different entities [FDP_IFC.2(NI)]. The TOE implements a rule-set governing the information flow [FDP_IFF.1(NI-IPTables), FDP_IFF.1(NI-ebtables)]. To facilitate the information flow control, the information must be identified [FIA_UID.1] based on security attributes the TOE can maintain [FIA_ATD.1(TU)]. The TOE must ensure that security attributes of the network data required by the information flow control policy are correctly interpreted [FDP_ITC.2(BA), FPT_TDC.1(BA)]. The protection of reused resources ensures that no data leaks from other protected sources [FDP_RIP.2, FDP_RIP.3].
O.SUBJECT.COM	The TSF must control the exchange of data using transient storage objects between subjects based on the identity of users. The access control policy must have a defined scope of control [FDP_ACC.1(TSO)]. The rules for the access control policy are defined [FDP_ACF.1(TSO)]. When import of user data is allowed, the TOE must ensure that user data security attributes required by the access control policy are correctly interpreted [FDP_ITC.2(BA), FPT_TDC.1(BA)]. The protection of reused resources ensures that no data leaks from other protected sources [FDP_RIP.2, FDP_RIP.3].
O.I&A	The TSF must ensure that only authorized users gain access to the TOE and its resources. Users authorized to access the TOE must use an identification and authentication process [FIA_UID.1, FIA_UAU.1]. Multiple I&A mechanisms are allowed as specified in [FIA_UAU.5]. To ensure authorized access to the TOE, authentication data is protected [FIA_ATD.1(HU), FIA_UAU.7]. Proper authorization for subjects acting on behalf of users is also ensured [FIA_USB.2]. The appropriate strength of the authentication mechanism is ensured [FIA_SOS.1]. To support the strength of authentication methods, the TOE is capable of identifying and reacting to unsuccessful authentication attempts [FIA_AFL.1]. In addition, user-initiated and TSF-initiated session locking [FTA_SSL.1, FTA_SSL.2] protect the authenticated user's session. The protection of reused resources ensures that no data leaks from other protected sources [FDP_RIP.2, FDP_RIP.3] are present.
O.MANAGE	The TOE provides management interfaces globally defined in [FMT_SMF.1] for:

Security objectives	Rationale
	• the access control policies [FMT_MSA.1(PSO), FMT_MSA.1(TSO), FMT_MSA.3(PSO), FMT_MSA.3(TSO)]; • the information flow control policy [FMT_MSA.3(NI), FMT_MTD.1(NI)]; • the auditing aspects [FMT_MTD.1(AE), FMT_MTD.1(AS), FMT_MTD.1(AT), FMT_MTD.1(AF)]; • the identification and authentication aspects [FMT_MTD.1(IAT), FMT_MTD.1(IAF), FMT_MTD.1(IAU), FMT_MTD.1(SSH)]. • the session locking threshold [FMT_MTD.1(SSL)]. Persistently stored user data is stored either in hierarchical or relational fashion, which implies an inheritance of security attributes from parent object [FMT_MSA.4(PSO)]. The rights management for the different management aspects is defined with [FMT_SMR.2]. The management interfaces for the revocation of user and object attributes is provided with [FMT_REV.1(OBJ) and FMT_REV.1(USR)].
O.TRUSTED_CHANNEL	The TOE provides a trusted channel protecting communication between a remote trusted IT system and itself [FTP_ITC.1].
O.REMOTE_AUDIT	In addition, the TOE shall allow the administration of the audit functions of remote trusted IT systems according to a centrally-defined policy defined with [FMT_MTD.1(AUD-AE), FMT_MTD.1(AUD-AS), FMT_MTD.1(AUD-AT), FMT_MTD.1(AUD-AF)].
O.ANALYZE_AUDIT	The extensive audit search and filtering mechanism is defined in [FAU_SAR.3(AUD)].
O.CP.USERDATA	The confidentiality protection mechanism for user data at rest is provided with the access control policy specified with [FDP_ACC.2] and [FDP_ACF.1] and supported by the cryptographic operations defined in [FCS_COP.1(CP)]. In addition, the confidentiality mechanism is defined with [FDP_CDP.1]. The the confidentiality protection is implemented with cryptographic mechanisms where the symmetric keys are derived from a random number generator as defined by [FCS_RNG.1(DM-INIT), FCS_RNG.1(DM-RUN), FCS_RNG.1(DM-FIPS)]. The management of the confidentiality protection mechanism is covered by [FMT_MSA.1] and [FMT_MSA.3] covering the general management aspects. [FMT_MTD.1(CP-UD)] allows owners of user data to select which of their data is covered by the confidentiality protection mechanism.
O.CP.ANCHOR	The management of the trust anchor for the confidentiality protection mechanism is specified with [FMT_MTD.1(CP-AN)].
O.RUNTIME.PROTECTION	Using the runtime protection mechanisms offered to applications is specified by [FPT_FLS.1(FULL), FPT_FLS.1(PARTIAL)].

Table 9: Security objectives for the TOE rationale

6.2.3 Security requirements dependency analysis

The following table demonstrates the dependencies of SFRs modeled in CC Part 2 and how the SFRs for the TOE resolve those dependencies:

Security functional requirement	Dependencies	Resolution
FAU_GEN.1	FPT_STM.1	FPT_STM.1
FAU_GEN.2	FAU_GEN.1	FAU_GEN.1
	FIA_UID.1	FIA_UID.1
FAU_SAR.1	FAU_GEN.1	FAU_GEN.1
FAU_SAR.2	FAU_SAR.1	FAU_SAR.1
FAU_SAR.3(AUD)	FAU_SAR.1	FAU_SAR.1
FAU_SEL.1	FAU_GEN.1	FAU_GEN.1
	FMT_MTD.1	FMT_MTD.1(AE)
FAU_STG.1	FAU_GEN.1	FAU_GEN.1
FAU_STG.3	FAU_STG.1	FAU_STG.1
FAU_STG.4	FAU_STG.1	FAU_STG.1
FCS_CKM.1(SYM)	[FCS_CKM.2 or FCS_COP.1]	FCS_COP.1(NET)
	FCS_CKM.4	FCS_CKM.4
FCS_CKM.1(RSA)	[FCS_CKM.2 or FCS_COP.1]	FCS_COP.1(NET)
	FCS_CKM.4	FCS_CKM.4
FCS_CKM.1(DSA)	[FCS_CKM.2 or FCS_COP.1]	FCS_COP.1(NET)
	FCS_CKM.4	FCS_CKM.4
FCS_CKM.2(NET)	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	FCS_CKM.1(SYM) FCS_CKM.1(RSA) FCS_CKM.1(DSA)
	FCS_CKM.4	FCS_CKM.4
FCS_CKM.4	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	FCS_CKM.1(SYM)
FCS_COP.1(NET)	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	FCS_CKM.1(SYM) FCS_CKM.1(RSA) FCS_CKM.1(DSA)
	FCS_CKM.4	FCS_CKM.4
FCS_COP.1(CP)	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	FCS_CKM.1(SYM)
	FCS_CKM.4	FCS_CKM.4

Security functional requirement	Dependencies	Resolution
FCS_RNG.1(SSH-DFLT)	No dependencies.	
FCS_RNG.1(SSH-FIPS)	No dependencies.	
FCS_RNG.1(DM-INIT)	No dependencies.	
FCS_RNG.1(DM-RUN)	No dependencies.	
FCS_RNG.1(DM-FIPS)	No dependencies.	
FDP_ACC.1(PSO)	FDP_ACF.1	FDP_ACF.1(PSO)
FDP_ACC.1(TSO)	FDP_ACF.1	FDP_ACF.1(TSO)
FDP_ACF.1(PSO)	FDP_ACC.1	FDP_ACC.1(PSO)
	FMT_MSA.3	FMT_MSA.3(PSO)
FDP_ACF.1(TSO)	FDP_ACC.1	FDP_ACC.1(TSO)
	FMT_MSA.3	FMT_MSA.3(TSO)
FDP_IFC.2(NI)	FDP_IFF.1	FDP_IFF.1(NI-IPTables) FDP_IFF.1(NI-ebtables)
FDP_IFF.1(NI-IPTables)	FDP_IFC.1	FDP_IFC.2(NI)
	FMT_MSA.3	FMT_MSA.3(NI)
FDP_IFF.1(NI-ebtables)	FDP_IFC.1	FDP_IFC.2(NI)
	FMT_MSA.3	FMT_MSA.3(NI)
FDP_ITC.2(BA)	[FDP_ACC.1 or FDP_IFC.1]	FDP_ACC.1(PSO) FDP_ACC.1(TSO) FDP_IFC.2(NI)
	[FTP_ITC.1 or FTP_TRP.1]	FTP_ITC.1
	FPT_TDC.1	FPT_TDC.1(BA)
FDP_RIP.2	No dependencies.	
FDP_RIP.3	No dependencies.	
FIA_AFL.1	FIA_UAU.1	FIA_UAU.1
FIA_ATD.1(HU)	No dependencies.	
FIA_ATD.1(TU)	No dependencies.	
FIA_SOS.1	No dependencies.	
FIA_UAU.1	FIA_UID.1	FIA_UID.1

Security functional requirement	Dependencies	Resolution
FIA_UAU.5	No dependencies.	
FIA_UAU.7	FIA_UAU.1	FIA_UAU.1
FIA_UID.1	No dependencies.	
FIA_USB.2	FIA_ATD.1	FIA_ATD.1(HU)
FPT_FLS.1(FULL)	No dependencies.	
FPT_FLS.1(PARTIAL)	No dependencies.	
FPT_STM.1	No dependencies.	
FPT_TDC.1(BA)	No dependencies.	
FTA_SSL.1	FIA_UAU.1	FIA_UAU.1
FTA_SSL.2	FIA_UAU.1	FIA_UAU.1
FTP_ITC.1	No dependencies.	
FDP_ACC.2(CP)	FDP_ACF.1	FDP_ACF.1(CP)
FDP_ACF.1(CP)	FDP_ACC.1	FDP_ACC.2(CP)
	FMT_MSA.3	FMT_MSA.3(CP)
FDP_CDP.1(CP)	[FDP_ACC.1 or FDP_IFC.1]	FDP_ACC.2(CP)
FMT_MSA.1(PSO)	[FDP_ACC.1 or FDP_IFC.1]	FDP_ACC.1(PSO)
	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MSA.1(TSO)	[FDP_ACC.1 or FDP_IFC.1]	FDP_ACC.1(PSO)
	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MSA.1(CP)	[FDP_ACC.1 or FDP_IFC.1]	FDP_ACC.2(CP)
	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MSA.3(PSO)	FMT_MSA.1	FMT_MSA.1(PSO)
	FMT_SMR.1	FMT_SMR.2
FMT_MSA.3(TSO)	FMT_MSA.1	FMT_MSA.1(TSO)
	FMT_SMR.1	FMT_SMR.2

Security functional requirement	Dependencies	Resolution
FMT_MSA.3(NI)	FMT_MSA.1	See OSPP rationale.
	FMT_SMR.1	FMT_SMR.2
FMT_MSA.3(CP)	FMT_MSA.1	FMT_MSA.1(CP)
	FMT_SMR.1	FMT_SMR.2
FMT_MSA.4(PSO)	[FDP_ACC.1 or FDP_IFC.1]	FDP_ACC.1(PSO)
FMT_MTD.1(AE)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(AS)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(AT)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(AF)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(NI)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(IAT)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(IAF)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(IAU)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(SSH)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(SSL)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(AUD-AE)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1

Security functional requirement	Dependencies	Resolution
FMT_MTD.1(AUD-AS)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(AUD-AT)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(AUD-AF)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(CP-AN)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_MTD.1(CP-UD)	FMT_SMR.1	FMT_SMR.2
	FMT_SMF.1	FMT_SMF.1
FMT_REV.1(OBJ)	FMT_SMR.1	FMT_SMR.2
FMT_REV.1(USR)	FMT_SMR.1	FMT_SMR.2
FMT_SMF.1	No dependencies.	
FMT_SMR.2	FIA_UID.1	FIA_UID.1

Table 10: TOE SFR dependency analysis

6.3 Security Assurance Requirements

The security assurance requirements for the TOE are the Evaluation Assurance Level 4 components as specified in [CC] part 3, augmented by ALC_FLR.3.

The following table shows the Security assurance requirements, and the operations performed on the components according to CC part 3: iteration (Iter.), refinement (Ref.), assignment (Ass.) and selection (Sel.).

Security assurance class	Security assurance requirement	Source	Operations			
			Iter.	Ref.	Ass.	Sel.
ADV Development	ADV_ARC.1 Security architecture description	CC Part 3	No	No	No	No
	ADV_FSP.4 Complete functional specification	CC Part 3	No	No	No	No
	ADV_IMP.1 Implementation representation of the TSF	CC Part 3	No	No	No	No
	ADV_TDS.3 Basic modular design	CC Part 3	No	No	No	No

Security assurance class	Security assurance requirement	Source	Operations			
			Iter.	Ref.	Ass.	Sel.
AGD Guidance documents	AGD_OPE.1 Operational user guidance	CC Part 3	No	No	No	No
	AGD_PRE.1 Preparative procedures	CC Part 3	No	No	No	No
ALC Life-cycle support	ALC_CMC.4 Production support, acceptance procedures and automation	CC Part 3	No	No	No	No
	ALC_CMS.4 Problem tracking CM coverage	CC Part 3	No	No	No	No
	ALC_DEL.1 Delivery procedures	CC Part 3	No	No	No	No
	ALC_DVS.1 Identification of security measures	CC Part 3	No	No	No	No
	ALC_FLR.3 Systematic flaw remediation	CC Part 3	No	No	No	No
	ALC_LCD.1 Developer defined life-cycle model	CC Part 3	No	No	No	No
	ALC_TAT.1 Well-defined development tools	CC Part 3	No	No	No	No
ASE Security Target evaluation	ASE_INT.1 ST introduction	CC Part 3	No	No	No	No
	ASE_CCL.1 Conformance claims	CC Part 3	No	No	No	No
	ASE_SPD.1 Security problem definition	CC Part 3	No	No	No	No
	ASE_OBJ.2 Security objectives	CC Part 3	No	No	No	No
	ASE_ECD.1 Extended components definition	CC Part 3	No	No	No	No
	ASE_REQ.2 Derived security requirements	CC Part 3	No	No	No	No
	ASE_TSS.1 TOE summary specification	CC Part 3	No	No	No	No
ATE Tests	ATE_COV.2 Analysis of coverage	CC Part 3	No	No	No	No
	ATE_DPT.1 Testing: basic design	CC Part 3	No	No	No	No
	ATE_FUN.1 Functional testing	CC Part 3	No	No	No	No
	ATE_IND.2 Independent testing - sample	CC Part 3	No	No	No	No
AVA Vulnerability assessment	AVA_VAN.3 Focused vulnerability analysis	CC Part 3	No	No	No	No

Table 11: Security assurance requirements

6.4 Security Assurance Requirements Rationale

The basis for the justification of EAL4 augmented with ALC_FLR.3 is the threat environment experienced by the typical consumers of the TOE. This matches the package description for EAL4 (enhanced-basic).

7 TOE Summary Specification

7.1 TOE Security Functionality

The following section explains how the security functions are implemented. The different TOE security functions cover the various SFR classes.

The primary security features of the TOE are:

- Audit
- Cryptographically secured communication channels
- Packet filter
- Identification and Authentication
- Discretionary Access Control
- Confidentiality protected data storage
- Security Management
- Protection mechanisms

7.1.1 Audit

The Lightweight Audit Framework (LAF) is designed to be an audit system for Linux compliant with the requirements from Common Criteria. LAF is able to intercept all system calls as well as retrieving audit log entries from privileged user space applications. The subsystem allows configuring the events to be actually audited from the set of all events that are possible to be audited. Those events are configured in a specific configuration file and then the kernel is notified to build its own internal structure for the events to be audited.

7.1.1.1 Audit functionality

The Linux kernel implements the core of the LAF functionality. It gathers all audit events, analyzes these events based on the audit rules and forwards the audit events that are requested to be audited to the audit daemon executing in user space.

Audit events are generated in various places of the kernel. In addition, a user space application can create audit records which needs to be fed to the kernel for further processing.

The audit functionality of the Linux kernel is configured by user space applications which communicate with the kernel using a specific netlink communication channel. This netlink channel is also to be used by applications that want to send an audit event to the kernel.

The kernel netlink interface is usable only by applications possessing the following capabilities:

- CAP_AUDIT_CONTROL: Performing management operations like adding or deleting audit rules, setting or getting auditing parameters;
- CAP_AUDIT_WRITE: Submitting audit records to the kernel which in turn forwards the audit records to the audit daemon.

Based on the audit rules, the kernel decides whether an audit event is discarded or to be sent to the user space audit daemon for storing it in the audit trail. The kernel sends the message to the audit daemon again using the above mentioned netlink communication channel. The audit daemon writes the audit records to the audit trail. An internal queuing mechanism is used for this purpose. When the queue does not have sufficient space to hold an audit record the TOE switches into single user mode, is halted or the audit daemon executes an administrator-specified notification action

depending on the configuration of the audit daemon. This ensures that audit records do not get lost due to resource shortage and the administrator can backup and clear the audit trail to free disk space for new audit logs.

Access to audit data by normal users is prohibited by the discretionary access control function of the TOE, which is used to restrict the access to the audit trail and audit configuration files to the system administrator only.

The system administrator can define the events to be audited from the overall events that the Lightweight Audit Framework using simple filter expressions. This allows for a flexible definition of the events to be audited and the conditions under which events are audited. The system administrator is also able to define a set of user IDs for which auditing is active or alternatively a set of user IDs that are not audited.

The system administrator can select files to be audited by adding them to a watch list that is loaded into the kernel.

This security function covers the SFRs of: FAU_GEN.1, FAU_GEN.2, FAU_SAR.2, FAU_SEL.1, FAU_STG.1, FAU_STG.3, FAU_STG.4, FPT_STM.1.

7.1.1.2 Audit trail

An audit record consists of one or more lines of text containing fields in a “keyword=value” tagged format. The following information is contained in all audit record lines:

- Type: indicates the source of the event, such as SYSCALL, PATH, USER_LOGIN, or LOGIN
- Timestamp: Date and time the audit record was generated
- Audit ID: unique numerical event identifier
- Login ID (“auid”), the user ID of the user authenticated by the system (regardless if the user has changed his real and / or effective user ID afterwards)
- Effective user ID: the effective user ID of the process at the time the audit event was generated
- Success or failure (where appropriate)
- Process ID of the subject that caused the event (PID)

This information is followed by event specific data. In some cases, such as SYSCALL event records involving file system objects, multiple text lines will be generated for a single event, these all have the same time stamp and audit ID to permit easy correlation.

The audit trail is stored in ASCII text. The TOE provides tools for managing ASCII files that can be used for post-processing of audit data. These tools include:

- less - reads the ASCII audit data
- ausearch - allows selective extraction of records from the audit trail using defined selection criteria
- sort - The audit records are listed in chronological order by default. The sort utility can be used together with ausearch to use a different sorting order.

The audit trail is stored in files which are accessible by root only.

This security function covers the SFRs of: FAU_SAR.1.

7.1.1.3 Centralized audit collection and management

The audit daemon of the TOE that stores the local audit trail is capable of remote auditing. The concept of remote auditing implies that one machine - the client - does not maintain the audit trail locally, but sends it to an audit server that centrally collects the audit data. The functionality of remote auditing contains the following two aspects implemented by the audit daemon:

- Audit server: The audit daemon can be configured to listen on a network port for other audit daemons to submit audit data. The communication is protected against eavesdropping using GSSAPI. Different protection mechanisms can be employed using GSSAPI, like Kerberos or SPNEGO (note: none of the security mechanisms is part of the evaluation). The audit daemon receives any audit records and stores them as part of the local audit trail together with the information about the source of the audit trail.
- Audit client: The audit daemon can be configured to send the audit trail off to an audit server via the network instead of storing the data locally.

The TOE is capable of providing the server side as well as the client side for remote auditing. For the client side, the TOE can always be configured to store the audit trail local to the client or to send it off to the server. Therefore, the TOE is always able to provide a self-sufficient audit trail if configured by the authorized administrator.

In conjunction with the centralized storage of the audit trail, the audit system of the TOE can also be administered centrally. The central management depends on using SSH to access the auditctl command on all administered servers:

- the administrator can either configure the audit functionality locally and use the scp tool to copy it to the remote target systems, or
- the administrator can configure the audit functionality directly on the remote systems using the ssh tool.

The remote management allows the configuration of all audit-related aspects on the remote system in the same way as the local audit system can be configured.

This security function covers the SFRs of: FAU_SAR.3(AUD), FMT_MTD.1(AUD-*).

7.1.2 Cryptographic services

The TOE provides cryptographically secured network communication channels to allow remote users to interact with the TOE. Using one of the following cryptographically secured network channels, a user can request the following services:

- OpenSSH: The OpenSSH application provides access to the command line interface of the TOE. Users may employ OpenSSH for interactive sessions as well as for non-interactive sessions. The console provided via OpenSSH provides the same environment as a local console. OpenSSH implements the SSHv2 protocol.

In addition to the cryptographically secured communication channels, the TOE also provides cryptographic algorithms for general use.

The cryptographic primitives for implementing the above mentioned cryptographic communication protocols are provided by OpenSSL.

7.1.2.1 SSHv2 Protocol

The TOE provides the Secure Shell Protocol Version 2 (SSH v2.0) to allow users from a remote host to establish a secure connection and perform a logon to the TOE.

The following table documents implementation details concerning the OpenSSH implementation's compliance to the relevant standards. It addresses areas where the standards permit different implementation choices such as optional features.

Reference	Description	Implementation Details
RFC 4253 chapter 5	Compatibility with old SSH versions	The OpenSSH implementation is capable of interoperating with clients and servers using the old 1.x protocol. That functionality is explicitly disabled in the evaluated configuration, it permits protocol version 2.0 exclusively.
RFC 4253 section 6.2	Compression	OpenSSH supports the OPTIONAL "zlib" compression method.
RFC 4253 section 6.3	Encryption	The ciphers supported in the evaluated configuration are listed in FCS_COP.1(NET) for the SSH protocol.
RFC 4252 chapter 7	Public Key Authentication Method: "publickey"	This REQUIRED authentication method is supported by OpenSSH but can be disabled by the administrator of the OpenSSH daemon.
RFC 4252 chapter 8	Password Authentication Method: "password"	This SHOULD authentication method is supported by OpenSSH but can be disabled by the administrator of the OpenSSH daemon.
RFC 4252 chapter 8	Password change request and setting new password	The OpenSSH implementation supports the optional password change mechanism in the evaluated configuration.
RFC 4252 chapter 9	Host-Based Authentication: "hostbased"	This OPTIONAL authentication method is disabled in the evaluated configuration.

Table 12: SSH implementation notes

The TOE supports the generation of RSA and DSA key pairs. These key pairs are used by OpenSSH for the host keys as well as for the per-user keys. When a user registers his public key with the user he wants to access on the server side, a key-based authentication can be performed instead of a password-based authentication. The key generation mechanism uses the Linux kernel random number generator. The evaluated configuration permits the import of externally-generated RSA/DSA key pairs.

This security function covers the SFRs of: FCS_CKM.1(RSA), FCS_CKM.1(DSA).

The TOE supports the following security functions of the SSH v2.0 protocol:

- Establishing a secure communication channel using the following cryptographic functions provided by the SSH v2.0 protocol:
  - Encryption as defined in section 4.3 of RFC 4253 - the keys are generated using the Linux kernel random number generator;
  - Diffie-Hellman key exchange as defined in section 6.1 of RFC 4253;
  - The keyed hash function for integrity protection as defined in section 4.4 of RFC 4253.

  Note: The protocol supports more cryptographic algorithms than the ones listed above. Those other algorithms are not covered by this evaluation and should be disabled or not used when running the evaluated configuration.

- Performing user authentication using the standard password-based authentication method the TOE provides for users (password authentication method as defined in chapter 5 of RFC 4252).
- Performing user authentication using a RSA or DSA key-based authentication method (public key authentication method as defined in chapter 5 of RFC 4252).
- Checking the integrity of the messages exchanged and close down the connection in case an integrity error is detected.

The OpenSSH applications of sshd, ssh and ssh-keygen use the OpenSSL random number generator seeded by /dev/random to generate cryptographic keys. OpenSSL provides different DRNGs depending whether the FIPS 140-2 mode is enabled in the system.

This security function covers the SFRs of: FCS_CKM.1(SYM), FCS_CKM.2(NET), FCS_CKM.4, FCS_COP.1(NET), FCS_RNG.1(SSH-DFLT), FCS_RNG.1(SSH-FIPS), FTP_ITC.1.

7.1.3 Packet filter

The Linux kernel's network stack implementation follows the layering structure of the network protocols. It implements the code for handling the link layer as well as the network layer. For those layers, independent filter mechanism are provided:

- Link layer: ebtables implements the filtering mechanism for bridges
- Network layer: netfilter/iptables implements the filtering mechanism for non-bridge interfaces

7.1.3.1 Network layer filtering

Netfilter

Netfilter is a framework for packet mangling, implemented in the Linux kernel network stack handling the network layer. The netfilter framework comprises of the following parts:

- The IP stack defines five hooks which are well-defined points in a network packet's traversal of the IP protocol stack. Each of the hooks, the network stack will call the netfilter framework allowing it to operate on the entire packet. Note: the netfilter framework provides such hooks in a number of network protocol implementations, but the TOE only supports IP as outlined above. Therefore, the ST specification only covers the IP protocol.
- The netfilter framework provides register functions for other kernel parts to listen to the different hooks. When a packet traverses one of the hooks and passed to the netfilter framework, it invokes every registered kernel part. These kernel parts then can examine the packet and possible alter it. As part of the examination, these kernel parts can instruct the netfilter framework to discard the packet, to allow it to pass, or to queue it to user space.
- When a packet is marked to be queued to user space, the netfilter framework handles the asynchronous communication with user space.

The netfilter framework implements the five hooks at the following points in the packet traversal chain:

- When the packet enters the network layer of the TOE and after applying some sanity checks, but before the routing table is consulted, the NF_IP_PRE_ROUTING hook is triggered.
- After passing the routing table decision and the routing code marks the packet to be targeted for another host, the NF_IP_FORWARD hook is triggered.

- After passing the routing table decision and the routing code marks the packet to be targeted for the local system, the NF_IP_LOCAL_IN hook is triggered.
- When the packet traversed all of the network stack and is about to be placed on the wire again, the NF_IP_POST_ROUTING hook is triggered.
- When a packet is generated locally, the NF_IP_LOCAL_OUT hook is triggered before the routing table is consulted.

IPTables

All communication on the network layer can be controlled by the IPTables framework.

The TOE implements a packet filter as part of the network stack provided with the Linux kernel. The combination of IPTables and netfilter implements the packet filter which provides stateful and stateless packet filtering for network communication by inspecting the IP header, the TCP header, UDP header and/or ICMP header of every network packet that passes the network stack.

The packet selection system called IP Tables uses the netfilter framework to implement the actual packet filtering logic on the network layer for the TCP/IP protocol family.

Note: IPTables is able to perform Network Address Translation (NAT) as well as Port Address Translation (PAT) for simple as well as more complex protocols. This mechanism is out of scope for the evaluation. Furthermore, packet mangling support is provided with IPTables which is also out of scope for the evaluation.

IPTables registers all hooks provided by the netfilter framework. The NAT/PAT mechanism uses the pre-routing and post-routing hooks whereas the packet filtering capability is enforced on the local-in, local-out and forwaring hooks.

IPTables consists of the following two components:

- In-kernel packet filter enforcement: The kernel-side of IPTables use the netfilter framework as indicated above. Three lists of packet filter rules are enforced by the kernel mechanism: one for each netfilter framework hook that applies to packet filtering. When a packet is analyzed by the IPTables kernel modules, they first select the applicable list based on the hook where the netfilter framework triggered IPTables. Each list contains zero or more rules which are iterated sequentially. A rule consists of a matching part (also called the "match extension") and an action part (also called the "target extension"). When a rule is applied to a packet, the kernel modules first applies the matching part of the rule. If the packet matches, the action part is enforced. If the action part contains a decision of the fate of the packet (to accept it, to drop it, or to drop it and sending a notification to the sender), the rule list validation stops for this packet. If the action part contains a modification instruction or log instruction for the packet, the rule list validation continues after performing this operation. When the rule list is iterated through and a packet could not be matched by a rule with a decision action (accept, drop), the default decision action applicable to the list is enforced. This default action is either to accept the packet, to drop the packet, or to drop the packet and send a notification to the sender.
- User space configuration application: The user space application iptables(1) allows the configuration of the IPTables kernel components. The application allows the specification of one rule per invocation where a rule contains the above mentioned matching part and action part. The tool also allows modification or deletion of existing rules as well as configuration of the default action. When using the tool, each invocation must specify the netfilter framework hook to which the rule applies to. See the man page of iptables(1) for more details.

7.1.3.2 Link layer filtering

ebtables chains

Similarly to the netfilter hooks in the network layer handling protocol, the link layer code handling the bridging functionality implements chains that are used by ebtables to apply filtering.

The netfilter framework is used to implement the ebtables chains at the following points in the packet traversal logic:

- When the packet enters the link layer of the TOE and after applying some sanity checks, but before the bridging decision made, the BROUTING chain is triggered.
- After passing the BROUTING chain but still before the bridging decision, the PREROUTING chain is triggered. When the bridging code decides that the frame is not intended for a bridge, it is forwarded to the network layer and processing on the link layer stops.
- After passing the bridge routing decision and the routing code marks the packet to be targeted for another host, the FORWARD chain is triggered.
- After passing the bridge routing decision and the routing code marks the packet to be targeted for the local system, the INPUT chain is triggered.
- When a packet is generated locally and the bridging decision marks the frame to be intended for a bridge, the OUTPUT chain is triggered.
- When the packet traversed all of the bridge logic and is about to be placed on the wire again, the POSTROUTING chain is triggered.

ebtables filtering rules

The packet selection system called ebtables uses the netfilter framework to implement the actual packet filtering logic for Ethernet frames routed through bridges.

Bridged networking communication is only visible on the link layer. The data encapsulated within the Ethernet frames are not routed through the host system's network stack and can therefore not be analyzed and controlled by the IPTables framework outlined above.

Even though the network stack of the host system does not analyze the bridged Ethernet frames any more, the ebtables framework receives the entire frame and is allowed to operate on that frame. Therefore, ebtables can analyze the protocol inside the Ethernet frame.

Please note that the ST author is perfectly aware that the term "packet" is used for the network layer (which includes the TCP/IP protocol family), whereas the term "frame" applies to the link layer (which includes the Ethernet frames). However, when speaking about ebtables that can analyze IP packets inside the link layer, this ST refers to them as packets/frames.

The ebtables framework can apply filters based on the following concepts:

- Matching of the frame/packet: Matches are based on Ethernet protocols or source and/or destination MAC addresses.
- Action to be performed on matched frames: Similarly to IPTables, ebtables must be configured to perform an action on the matched frame. The action can either be to discard the packet or to accept it.

This security function covers the SFRs of:

- Packet filtering rules: FDP_IFC.2(NI), FDP_IFF.1(NI-*)
- Interpretation of network protocol: FIA_UID.1, FDP_ITC.2, FPT_TDC.1(BA)

- Maintenance of rules: FIA_ATD.1(TU)

7.1.4 Identification and Authentication

User identification and authentication in the TOE includes all forms of interactive login (e.g. using the SSH protocol or log in at the local console) as well as identity changes through the su and sudo commands. These all rely on explicit authentication information provided interactively by a user. In addition, the key-based authentication mechanism of the OpenSSH server is another form of of authentication.

7.1.4.1 PAM-based identification and authentication mechanisms

Linux uses a suite of libraries called the "Pluggable Authentication Modules" (PAM) that allow an administrative user to choose how PAM-aware applications authenticate users. The TOE provides PAM modules that implement all the security functionality to:

- Provides login control and establishing all UIDs, GIDs and login ID for a subject
- Ensure the quality of passwords
- Enforce limits for accounts (such as the number of maximum concurrent sessions allowed for a user)
- Enforce the change of passwords after a configured time including the password quality enforcement
- Enforcement of locking of accounts after failed login attempts.
- Restriction of the use of the root account to certain terminals
- Restriction of the use of the su and sudo commands

The login processing sets the real, file system effective and login UID as well as the real, effective, file system GID and the set of supplemental GIDs of the subject that is created. It is of course up to the client application usually provided by a remote system to protect the user’s entry of a password correctly (e. g. provide only obscured feedback).

During login processing, the user is shown a banner. After successful authentication, the login time is recorded.

After a successful identification and authentication, the TOE initiates a session for the user and spawns the initial login shell as the first process the user can interact with. The TOE provides a mechanism to lock a session either automatically after a configurable period of inactivity for that session or upon the user's request.

This security function covers the SFRs of FDP_RIP.3, FIA_AFL.1, FIA_SOS.1, FIA_UAU.1, FIA_UID.1, FIA_UAU.5, FIA_UAU.7, FIA_USB.2.

7.1.4.2 User Identity Changing

Users can change their identity (i.e., switch to another identity) using one of the following commands provided with the TOE:

su command

The su command is intended for a switch to a another identity that establishes a new login session and spawns a new shell with the new identity. When invoking su, the user must provide the credentials associated with the target identity - i.e. when the user wants to switch to another user ID, it has to provide the password protecting the account of the target user.

The primary use of the su command within the TOE is to allow appropriately authorized individuals the ability to assume the root identity to perform administrative actions. In this system the capability to login as the root identity has been restricted to defined terminals only. In addition the use of the su command to switch to root has been restricted to users belonging to a special group. Users that don’t have access to a terminal where root login is allowed and are not member of that special group will not be able to switch their real, file system and effective user ID to root even if they would know the authentication information for root. Note that when a user executes a program that has the setuid bit set, only the effective user ID and file system ID are changed to that of the owner of the file containing the program while the real user ID remains that of the caller. The login ID is neither changed by the su command nor by executing a program that has the setuid or setgid bit set as it is used for auditing purposes.

sudo command

The sudo command is intended for giving users permissions to execute commands with another user identity. When invoking sudo, the user has to authenticate with this credentials.

Sudo is associated with sophisticated ruleset that can be engaged to specify which:

- source user ID
- originating from which host
- can access a command, a command with specific configuration flags, or all commands within a directory
- with which new user identity.

When switching identities, the real, file system and effective user ID and real, file system and effective group ID are changed to the one of the user specified in the command (after successful authentication as this user).

Note: The login ID is not retained for the following special case:

1. User A logs into the system.
2. User A uses su to change to user B.
3. User B now edits the cron or at job queue to add new jobs. This operation is appropriately audited with the proper login ID.
4. Now when the new jobs are executed as user B, the system does not provide the audit information that the jobs are created by user A.

The su command invokes the common authentication mechanism to validate the supplied authentication.

This security function covers the SFRs of FIA_USB.2.

7.1.4.3 Authentication Data Management

Each TOE instance maintains its own set of users with their passwords and attributes. Although the same human user may have accounts on different servers interconnected by a network and running an instantiation of the TOE, those accounts and their parameter are not synchronized on different TOE instances. As a result the same user may have different user names, different user Ids, different passwords and different attributes on different machines within the networked environment. Existing mechanism for synchronizing this within the whole networked system are not subject to this evaluation.

Each TOE instance within the network maintains its own administrative database by making all administrative changes on the local TOE instance. System administration has to ensure that all machines within the network are configured in accordance with the requirements defined in this Security Target.

The file /etc/passwd contains for each user the user’s name, the id of the user, an indicator whether the password of the user is valid, the principal group id of the user and other (not security relevant) information. The file /etc/shadow contains for each user a hash of the user's password, the userid, the time the password was last changed, the expiration time as well as the validity period of the password and some other information that are not subject to the security functions as defined in this Security Target. Users are allowed to change their passwords by using the passwd command. This application is able to read and modify the contents of /etc/shadow for the user’s password entry, which would ordinarily be inaccessible to a non-privileged user process. Users are also warned to change their passwords at login time if the password will expire soon, and are prevented from logging in if the password has expired.

The time of the last successful logins is recorded in the directory /var/log/faillock where one file per user is kept.

The TOE displays informative banners before or during the login to users. The banners can be specified with the files /etc/issue for log ins via the physical console or /etc/issue.net for remote log ins, such as via SSH. When performing a log in on the physical console, the banner is displayed above the username and password prompt. For log ins via SSH, the banner is displayed to the remote peer during the SSH-session handshake takes place. The remote SSH client will display the banner to the user. When using the provided OpenSSH client, the banner is displayed when the user instructs the OpenSSH client to log into the remote system.

This security function covers the SFRs of FIA_ATD.1(HU).

7.1.4.4 SSH key-based authentication

In addition to the PAM-based authentication outlined above, the OpenSSH server is able to perform a key-based authentication. When a user wants to log on, instead of providing a password, the user applies his SSH key. After a successful verification, the OpenSSH server considers the user as authenticated and performs the PAM-based operations as outlined above.

To establish a key-based authentication, a user first has to generate an RSA or DSA key pair. The private part of the key pair remains on the client side. The public part is copied to the server into the file .ssh/authorized_keys which resides in the home directory of the user he wants to log on as. When the login operation is performed the SSHv2 protocol tries to perform the "publickey" authentication using the private key on the client side and the public key found on the server side. The operations performed during the publickey authentication is defined in RFC 4252 chapter 7.

Users have to protect their private key part the same way as protecting a password. Appropriate permission settings on the file holding the private key is necessary. To strengthen the protection of the private key, the user can encrypt the key where a password serves as key for the encryption operation. See ssh-keygen(1) for more information.

This security function covers the SFRs of FIA_UAU.1, FIA_UID.1, FIA_UAU.5, FIA_SOS.1.

7.1.4.5 Session locking

The TOE uses the screen(1) application which locks the current session of the user either after an administrator-specified time of inactivity or upon the user's request.

To unlock the session, the user must supply his password. Screen uses PAM to validate the password and allows the user to access his session after a successful validation.

This security function covers the SFRs of FTA_SSL.1, FTA_SSL.2.

7.1.5 Discretionary Access Control

The general policy enforced is that subjects (i.e., processes) are allowed only the accesses specified by the policies applicable to the object the subject requests access to. Further, the ability to propagate access permissions is limited to those subjects who have that permission, as determined by the policies applicable to the object the subject requests access to.

A subject may possess one or more of the following capabilities which provide the following exemptions from the DAC mechanism:

- CAP_DAC_OVERRIDE: A process with this capability is exempt from all restrictions of the discretionary access control and can perform any action desired. For the execution of a file, the permission bit vector of that file must contain at least one execute bit.
- CAP_DAC_READ_SEARCH: A process with this capability overrides all DAC restrictions regarding read and search on files and directories.
- CAP_CHOWN: A process with this capability is allowed to make arbitrary changes to a file's UID or GID.
- CAP_FOWNER: Setting permissions and ownership on objects even if the process' UID does not match the UID of the object.
- CAP_FSETID: Don't clear SUID and SGID permission bits when a file is modified.

DAC provides the mechanism that allows users to specify and control access to objects that they own. DAC attributes are assigned to objects at creation time and remain in effect until the object is destroyed or the object attributes are changed. DAC attributes exist for, and are particular to, each type of named object known to the TOE. DAC is implemented with permission bits and, when specified, ACLs.

The outlined DAC mechanism applies only to named objects which can be used to store or transmit user data. Other named objects are also covered by the DAC mechanism but may be supplemented by further restrictions. These additional restrictions are out of scope for this evaluation. Examples of objects which are accessible to users that cannot be used to store or transmit user data are: virtual file systems externalizing kernel data structures (such as most of procfs, sysfs, binfmt_misc) and process signals.

During creation of objects, the TSF ensures that all residual contents is removed from that object before making it accessible to the subject requesting the creation.

When data is imported into the TOE (such as when mounting disks created by other trusted systems), the TOE enforces the permission bits and ACLs applied to the file system objects.

7.1.5.1 Permission bits

The TOE supports standard UNIX permission bits to provide one form of DAC for file system objects in all supported file systems. There are three sets of three bits that define access for three categories of users: the owning user, users in the owning group, and other users. The three bits in each set indicate the access permissions granted to each user category: one bit for read (r), one for write (w) and one for execute (x). Note that write access to file systems mounted as read only (e. g. CD-ROM) is always rejected (the exceptions are character and block device files which can still be written to as write operations do not modify the information on the storage media). Also, write access to file system objects marked as immutable is always rejected. The SAVETXT attribute is used for world-writeable temp directories preventing the removal of files by users other than the owner.

Each process has an inheritable “umask” attribute which is used to determine the default access permissions for new objects. It is a bit mask of the user/group/other read/write/execute bits, and specifies the access bits to be removed from new objects. For example, setting the umask to “002” ensures that new objects will be writable by the owner and group, but not by others. The umask is defined by the administrator in the /etc/login.defs file or 022 by default if not specified.

This security function covers the SFRs of FDP_ACC.1(PSO), FDP_ACF.1(PSO), FDP_ITC.1(BA), FDP_RIP.2, FPT_TDC.1(BA).

7.1.5.2 Access Control Lists (ACLs)

The TOE provides support for POSIX type ACLs to define a fine grained access control on a user basis. ACLs are supported for all file system objects stored with the following file systems:

- ext4
- tmpfs

An ACL entry contains the following information:

- A tag type that specifies the type of the ACL entry
- A qualifier that specifies an instance of an ACL entry type
- A permission set that specifies the discretionary access rights for processes identified by the tag type and qualifier

An ACL contains exactly one entry of three different tag types (called the "required ACL entries” forming the "minimum ACL"). The standard UNIX file permission bits as described in the previous section are represented by the entries in the minimum ACL.

A default ACL is an additional ACL which may be associated with a directory. This default ACL has no effect on the access to this directory. Instead the default ACL is used to initialize the ACL for any file that is created in this directory. If the new file created is a directory it inherits the default ACL from its parent directory. When an object is created within a directory and the ACL is not defined with the function creating the object, the new object inherits the default ACL of its parent directory as its initial ACL.

7.1.5.3 File system objects

Access to file system objects is generally governed by permission bits. For the ext4 file system, ACLs are supported.

File system objects access checks are performed when the object is initially opened, and are not checked on each subsequent access. Changes to access controls (i.e., revocation) are effective with the next attempt to open the object.

7.1.5.4 IPC objects

The TOE implements the following standard types of IPC mechanisms:

- SYSV Shared Memory
- SYSV and POSIX Message Queues
- SYSV Semaphores

Access to the above mentioned IPC mechanisms are governed by UNIX permission bits.

As the IPC objects of UNIX domain socket special files and Named Pipes are represented as file system objects, the access control mechanism covering file system objects are applicable to these IPC mechanisms too.

The TOE maintains IPC object types where each process has its own namespace for that object type: sockets - including network sockets. Access to the socket is only possible by the process whose socket namespace contains the socket reference. Setting of permissions for such objects can be handled using file descriptor passing.

This security function covers the SFRs of FDP_ACC.1(TSO), FDP_ACF.1(TSO).

7.1.5.5 at and cron jobs queues

at and cron jobs can only be accessed (read/added/modified/deleted) by the owning user. The TOE maintains at and cron job queues for each user.

The root user can always access every at or cron job queue.

The at or cron jobs are started with the UIDs/GIDs of the creator of the job.

7.1.6 Confidentiality protected data storage

File system objects are stored on block devices, such as partitions on hard disk. The Linux operating systems offers the use of an additional layer between the file systems and the physical block device to encrypt and decrypt any data transmitted between the file system and the block device. The dm_crypt functionality uses the Linux device mapper to provide such encryption and decryption operation that is transparent to the file system and therefore to the user.

Before mounting the block device that is protected by the dm_crypt encryption scheme, the owner of the encrypted block device has to provide a passphrase. This passphrase is used to decrypt the symmetric session key which is injected into the kernel. Using that session key kernel is now able to decrypt (to unlock) the block device and provides access to data stored on that block device. At this point, the file system can be mounted as the file system can now be read.

Once the dm_crypt protected block device is unlocked and mounted, it is accessible as any other file system. When it is unmounted and locked (i.e. the kernel is informed to discard the session key), all data on the block device is inaccessible. Even administrative users like the root user is not able to access any data any more. When an administrator would access the raw hardware hosting the block device, only encrypted data can be read.

For the cryptographic operation, the creator of the dm_crypt block device can select the cipher. When creating the dm_crypt block device, the session key is obtained from the Linux random number generator and stored on the block device encrypted with the user's passphrase. The key derivation mechanism from the user's password is based on the LUKS mechanism which is also conformant to the FIPS140-2 cryptographic standard.

The encryption and decryption operation of the block device is implemented by the kernel. To unlock the encrypted session key stored on the protected block device, the cryptsetup application performs the following steps:

1. obtain the user's passphrase
2. apply the LUKS key derivation mechanism on the passphrase
3. read the encrypted session key from the block device
4. decrypt the session key with the key derived from the user's passphrase
5. inject the decrypted session key into the kernel

Using the cryptsetup tool, the session key can also be transferred by encrypting it with another passphrase which can be given to another user. The transfer follows the same steps outlined for the unlocking operation, but instead of injecting the decrypted session key into the kernel, cryptsetup fetches the new passphrase from the user, applies the LUKS mechanism on that passphrase, encrypts the session key with the derived key and stores the encrypted session key in a separate area on the block device. At this point, the session key is now stored encrypted in two separate places.

Similarly, the cryptsetup tool can be used to erase the storage location of one encrypted session key which implies that the user owning the passphrase of the affected encrypted session key is not able to unlock the block device any more.

During setup time of an encrypted disk, the application cryptsetup uses data out of /dev/random directly as key material (normal mode, runtime), /dev/urandom directly as key material (normal mode, initial installation time), from the libgcrypt ANSI X9.31 DRNG seeded by /dev/random (FIPS 140-2 mode, runtime), or from the libgcrypt ANSI X9.31 DRNG seeded by /dev/urandom (FIPS 140-2 mode, initial installation time).

This security function covers the SFRs of FCS_COP.1(CP), FCS_RNG.1(DM-INIT), FCS_RNG.1(DM-RUN), FCS_RNG.1(DM-FIPS), FDP_ACC.2(CP), FDP_ACF.1(CP), FDP_CDP.1, FMT_MSA.1(CP), FMT_MSA.3(CP), FMT_MTD.1(CP-AN), FMT_MTD.1(CP-UD).

7.1.7 Security Management

The security management facilities provided by the TOE are usable by authorized users and/or authorized administrators to modify the configuration of TSF. The configuration of TSF are hosted in the following locations:

- Configuration files (or TSF databases)
- Data structures maintained by the kernel and within the kernel memory

The TOE provides applications to authorized users as well as authorized administrators to perform various administrative tasks. These applications are documented as part of the administrator and user guidance. These applications are either used to modify configuration files or to access parameters controlled and enforced by the kernel via kernel-provided interfaces to user space.

Configuration options are stored in different configuration files. These files are protected using the DAC mechanisms against unauthorized access where usually the root user only is allowed to write to the files. In some special cases (like for /etc/shadow), the file is even readable to the root user

only. It is the task of the persons responsible for setting up and administrating the system to ensure that the access control features of the TOE are used throughout the lifetime of the system to protect those databases. These configuration files are accessed using applications which are able to interpret the contents of these configuration files. Each TOE instance maintains its own TSF database. Synchronizing those databases is not performed in the evaluated configuration. If such synchronization is required by an organization it is the responsibility of an administrative user of the TOE to achieve this either manually or with some automated assistance.

To access data structures maintained by the kernel, applications use the kernel-provided interfaces, such as system calls, virtual file systems, netlink sockets, and device files. These kernel interfaces are restricted to authorized administrators or authorized users, if applicable, by either using DAC (for virtual file system objects) or special kernel-internal verification checks for each interface.

The TOE provides security management applications for all security-relevant settings listed throughout this ST, i.e. all iterations of FMT_MSA.1, FMT_REV.1, and FMT_MTD.1; FMT_SMR.2, FMT_SMF.1.

7.1.7.1 Approval and delegation of management functions

Using the sudo command, authorized administrators can approve that other users can perform management tasks. Once the administrator approves the operation, the /etc/sudoers file is modified to grant the user the right to perform the administrative operation.

Using the /etc/sudoers file, the administrator can specify the approval rules based on the following fine-grained properties:

- Specification of the command that can be executed. The command may contain wild cards.
- Specification of the target user ID or group ID the command shall be executed with.
- Specification of the user ID or group ID (where all members of the group are covered) which are allowed by this rule.

Using the sudo command and the associated /etc/sudoers configuration file, the administrative users, i.e. the users allowed to use the root UID are allowed to delegate parts or all of their authority to other users.

7.1.7.2 Privileges

Privileges to perform administrative actions are maintained by the TOE. These privileges are separated into privileges to act on data or access functionality in user space and in kernel space.

Functionality accessible in user space are applications that can be invoked by users. Also, data accessible in user space is either data maintained with an application or data stored in persistent or transient storage objects. Privileges are controlled by permissions to invoke applications and to access data. For example, the configuration files including the user databases of /etc/passwd and /etc/shadow are accessible to the root user only. Therefore, the root user is given the privilege to perform modifications on this configuration data which constitutes administrative actions.

Functionality and data maintained by the kernel must be accessed using system calls. The kernel implements a privilege check for functions and data that shall not be accessible by normal users. These privileges are controlled with capabilities that can be assigned to processes. If a process is assigned with a capability, it is allowed to request special operations that other processes cannot. To implement consistency with the Unix legacy, processes with the effective UID of zero are implicitly given all capabilities. However, these processes may decide to drop capabilities. Such capabilities are marked by names with the prefix of "CAP_" throughout this document. The Linux kernel

implements many more capabilities than mentioned in this document. These unmentioned capabilities protect functions that do not directly cover SFR functionality but need to be protected to ensure the integrity of the system and its resources.

7.1.8 Protection Mechanisms

The TOE provides functionality to mitigate the effects of a potentially present buffer overrun vulnerability in applications. This mitigation mechanism covers the following aspects:

- Configuring the memory holding the stack to be readable and writable only, but not executable. Per default, the all processes and threads use stacks which are non-executable.
- Performing address space randomization which affects position independent code (PIC) as well as position independent executables (PIE). All shared libraries are necessarily PIC, whereas only dedicated TSF applications are PIE. Users may compile their applications as PIE to allow address space randomization protect their applications.
- Marking the runtime memory of all parts of a binary as read-only apart from heap data before the loaded application gains control. This support is enabled for dedicated TSF applications and specifically compiled user applications. Partial protection is also possible which implies that also the .got.plt section is marked read/writable.

This security function covers the SFR of FPT_FLS.1(FULL), FPT_FLS.1(PARTIAL).

8 Abbreviations, Terminology and References

8.1 Abbreviations

ACL
Access Control List

API
Application Programming Interface

KVM
Kernel Virtualized Machine

HTTP
Hypertext Transfer Protocol

SFR
Security Functional Requirement

SSL
Secure Sockets Layer

ST
Security Target

TCP/IP
Transmission Control Protocol / Internet Protocol

TLS
Transport Layer Security

TOE
Target of Evaluation

TSF
TOE Security Functionality

VM
Virtual Machine

VPN
Virtual Private Network

8.2 Terminology

This section contains definitions of technical terms that are used with a meaning specific to this document. Terms defined in the [CC] are not reiterated here, unless stated otherwise.

Authentication Data
Authentication data is the data used by users or remote entities to authenticate their claimed identity.

Authorized Administrator
This term refers to a user in one of the defined administrative roles of a Linux system. The TOE associates the user with the UID of zero and named "root" with administrative authorities. Effectively, the UID zero is assigned with all Linux capabilities known to the Linux kernel. Every user who is allowed to log on as that root user, or to switch their UID to the root user is considered an authorized administrator. In addition, any user who is able to execute applications which grant one or more Linux capabilities to be used in an unconditional manner is considered an authorized administrator.

Category
A category is the non-hierarchical category of the lower MLS label defined with an SELinux label. Note: an SELinux label consists of four parts where the MLS label is one of them. The MLS label in turn is split into a higher and a lower MLS label part.

Classification
A sensitivity label associated with an object.

Clearance
A sensitivity label associated with a subject or user.

DAC
Discretionary Access Control implemented with permission bits and ACLs.

Data
Arbitrary bit sequences on persistent or transient storage media.

Dominate
Sensitivity label A dominates sensitivity label B if the hierarchical level of A is greater than or equal to the hierarchical level of B, and the category set of label A is a proper subset of or equal to the category set of label B. (cf. Incomparable sensitivity labels).

Guest
Software executing within a virtual machine environment. There can be zero or more guests executing concurrently on the host system.

Host
The host system provides the Linux environment that controls and manages the virtual machines. The host provides the execution environment for every virtual machine.

Information
Any data held within a server, including data in transit between systems.

IOMMU
Input / Output Memory Management Unit. This MMU allows the setup of multiple DMA areas for different virtual machines.

KVM
Kernel-based Virtual Machine.

MLS
Multi-level security

Named Object
In Linux, those objects that are covered by access control policies. The list of objects defined as named objects is provided with FDP_ACC.1.

Object
For Linux, objects are defined by FDP_ACC.1.

OSPP
Operating System Protection Profile

OSPP EP
Operating System Protection Profile Extended Package

PAM
Pluggable Authentication Module - the authentication functionality provided with Linux is highly configurable by selecting and combining different modules implementing different aspects of the authentication process.

Product
The term product is used to define software components that comprise the Linux system.

QEMU
The QEMU software component implements the virtual devices and virtual resources for virtual machines. There is one instance of QEMU per virtual machine. The QEMU software component is also identified as the "kvm" application on the host system.

SELinux
Linux kernel LSM module that is able to implement arbitrary security policies. An SELinux policy distributed with the TOE implements multi-level or multi-category security.

Sensitivity Label
The TOE attaches a sensitivity label to each named object. This label consists of a hierarchical sensitivity level and a set of zero or more categories. The policy defines the number and names of the sensitivity levels and categories.

Subject
There are two classes of subjects in Red Hat Enterprise Linux: i) untrusted internal subject - this is a Linux process running on behalf of some user or providing an arbitrary service, running outside of the TSF (for example, with no privileges); ii) trusted internal subject - this is a Linux process running as part of the TSF (for example: service daemons and the process implementing the identification and authentication of users).

Target Of Evaluation (TOE)
The TOE is defined as the Red Hat Enterprise Linux operating system, running and tested on the hardware and firmware specified in this Security Target. The BootPROM firmware as well as the hardware are not part of the TOE.

User
Any individual/person or technical entity (such as a service added by the administrator on top of the TOE) who has a unique user identifier and who interacts with the product.

User Security Attributes
Defined by functional requirement FIA_ATD.1, every user is associated with a number of security attributes which allow the TOE to enforce its security functions on this user. This also includes the user clearance which defines the maximum sensitivity label a user can have access to.

Virtual devices
See virtual resources for a generic explanation. This definition applies also to virtual devices, but with a focus to devices, such as disks, network cards, graphics cards, and similar.

Virtual machine
A virtual machine is an execution environment where the software executing within the virtual machine has access to the processor's user and supervisor state and resources defined by the host system. Resources include the number of processors, RAM size, physical devices, virtualized devices, communication channels to other virtual machines and the host system. For the KVM environment a virtual machine environment is controlled and provided by the Linux kernel hypervisor functionality plus the QEMU application instantiated for each virtual machine.

Virtual machine environment
See virtual machine.

Virtual resources
Virtual resources are resources that either do not physically exist and do not exist in the host system. Virtual resources are implemented by the virtual machine environment and are provided to the respective virtual machine. For example, virtual resources are special exceptions that can be triggered from the virtual machine environment to request services from the host system, such as para-virtualized drivers. Virtual devices can be considered one form of virtual resources.

8.3 References

CC	**Common Criteria for Information Technology Security Evaluation**	
	Version	3.1R4
	Date	September 2012
	Location	http://www.commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R4.pdf
	Location	http://www.commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R4.pdf
	Location	http://www.commoncriteriaportal.org/files/ccfiles/CCPART3V3.1R4.pdf
OSPP	**BSI Operating System Protection Profile**	
	Version	2.0
	Date	2010
OSPP-AUD	**BSI OSPP Extended Package - Advanced Audit**	
	Version	2.0
	Date	2010
RFC4253	**The Secure Shell (SSH) Transport Layer Protocol**	
	Date	January 2006
	Location	http://tools.ietf.org/html/rfc4253