Oracle Solaris 11.4

Security Target

Version 1.3

February 2021

Document prepared by

www.lightshipsec.com

Document History

Version	Date	Author	Description
1.0	09 Nov 2020	G Nickel	Update TOE version
1.1	19 Nov 2020	G Nickel	Update IDR version
1.2	25 Jan 2021	L Turner	Update TLS and SSH.
1.3	8 Feb 2021	L Turner	Finalize for certification.

Table of Contents

List of Tables

1 Introduction

1.1 Overview

1 This Security Target (ST) defines the Oracle Solaris 11.4 Target of Evaluation (TOE) for the purposes of Common Criteria (CC) evaluation.

2 Oracle Solaris is a UNIX-based operating system designed to deliver a consistent platform to run enterprise applications.

1.2 Identification

Table 1: Evaluation identifiers

Target of Evaluation	Oracle Solaris 11.4 Build: 11.4 SRU 26.0.1 (11.4-11.4.26.0.1.75.4) with IDR 4534 v3
Security Target	Oracle Solaris 11.4 Security Target, v1.3

1.3 Conformance Claims

3 This ST supports the following conformance claims:

a) CC version 3.1 revision 5

i) CC Part 2 extended

ii) CC Part 3 extended

b) Protection Profile for General Purpose Operating Systems, v4.2.1

c) Extended Package for Secure Shell (SSH), v1.0, package-augmented

d) NIAP Technical Decisions per Table 2

Table 2: NIAP Technical Decisions

TD #	Name	Rationale if n/a
0525	Updates to Certificate Revocation (FIA_X509_EXT.1)	
0501	Cryptographic selections and updates for OS PP	
0496	GPOS PP adds allow-with statement for VPN Client V2.1	
0493	X.509v3 certificates when using digital signatures for Boot Integrity	
0463	Clarification for FPT_TUD_EXT	
0446	Missing selections for SSH	
0441	Updated TLS Ciphersuites for OS PP	

TD #	Name	Rationale if n/a
0420	Conflict in FCS_SSHC_EXT.1.1 and FCS_SSHS_EXT.1.1	
0386	Platform-Provided Verification of Update	
0365	FCS_CKM_EXT.4 selections	
0332	Support for RSA SHA2 host keys	
0331	SSH Rekey Testing	
0240	FCS_COP.1.1(1) Platform provided crypto for encryption/decryption	

1.4 Terminology

Table 3: Terminology

Term	Definition
Address Space Layout Randomization (ASLR)	An anti-exploitation feature which loads memory mappings into unpredictable locations. ASLR makes it more difficult for an attacker to redirect control to code that they have introduced into the address space of a process.
Administrator	An administrator is responsible for management activities, including setting policies that are applied by the enterprise on the operating system. This administrator could be acting remotely through a management server, from which the system receives configuration policies. An administrator can enforce settings on the system which cannot be overridden by non-administrator users.
Application (app)	Software that runs on a platform and performs tasks on behalf of the user or owner of the platform, as well as its supporting documentation.
Application Programming Interface (API)	A specification of routines, data structures, object classes, and variables that allows an application to make use of services provided by another software component, such as a library. APIs are often provided for a set of libraries included with the platform.
Common Criteria (CC)	Common Criteria for Information Technology Security Evaluation.
Common Evaluation Methodology (CEM)	Common Evaluation Methodology for Information Technology Security Evaluation.
Credential	Data that establishes the identity of a user, e.g. a cryptographic key or password.

Term	Definition
Critical Security Parameters (CSP)	Information that is either user or system defined and is used to operate a cryptographic module in processing encryption functions including cryptographic keys and authentication data, such as passwords, the disclosure or modification of which can compromise the security of a cryptographic module or the security of the information protected by the module.
Data At Rest (DAR) Protection	Countermeasures that prevent attackers, even those with physical access, from extracting data from non-volatile storage. Common techniques include data encryption and wiping.
Data Execution Prevention (DEP)	An anti-exploitation feature of modern operating systems executing on modern computer hardware, which enforces a non-execute permission on pages of memory. DEP prevents pages of memory from containing both data and instructions, which makes it more difficult for an attacker to introduce and execute code.
Developer	An entity that writes OS software. For the purposes of this document, vendors and developers are the same.
Extended Package (EP)	An implementation-independent set of security requirements for a specific subset of products described.
General Purpose Operating System	A class of OSes designed to support a wide-variety of workloads consisting of many concurrent applications or services. Typical characteristics for OSes in this class include support for third-party applications, support for multiple users, and security separation between users and their respective resources. General Purpose Operating Systems also lack the real-time constraint that defines Real Time Operating Systems (RTOS). RTOSes typically power routers, switches, and embedded devices.
Host-based Firewall	A software-based firewall implementation running on the OS for filtering inbound and outbound network traffic to and from processes running on the OS.
IDR	Interim Diagnostic or Relief (Oracle software patch)
KMIP	Key Management Interoperability Protocol
Operating System (OS)	Software that manages physical and logical resources and provides services for applications. The terms TOE and OS are interchangeable in this document.
Oracle Integrated Lights Out Manager (ILOM)	Oracle Integrated Lights Out Manager (ILOM) is the service processor embedded on all Oracle's x86 and SPARC servers.
OSPP	Protection Profile for General Purpose Operating Systems

Term	Definition
Personally Identifiable Information (PII)	Any information about an individual maintained by an agency, including, but not limited to, education, financial transactions, medical history, and criminal or employment history and information which can be used to distinguish or trace an individual's identity, such as their name, social security number, date and place of birth, mother's maiden name, biometric records, etc., including any other personal information which is linked or linkable to an individual. [OMB]
Protection Profile (PP)	An implementation-independent set of security requirements for a category of products.
Security Target (ST)	A set of implementation-dependent security requirements for a specific product.
Sensitive Data	Sensitive data may include all user or enterprise data or may be specific application data such as PII, emails, messaging, documents, calendar items, and contacts. Sensitive data must minimally include credentials and keys. Sensitive data shall be identified in the OS's TSS by the ST author.
SRU	Support Repository Update (Oracle software update)
Target of Evaluation (TOE)	The product under evaluation. In this case, Oracle Solaris.
TOE Security Functionality (TSF)	The security functionality of the product under evaluation.
TOE Summary Specification (TSS)	A description of how a TOE satisfies the SFRs in a ST.
Security Functional Requirement (SFR)	A requirement for security enforcement by the TOE.
Security Assurance Requirement (SAR)	A requirement to assure the security of the TOE.
Unified Extensible Firmware Interface (UEFI) Secure Boot	UEFI Secure Boot defines how platform firmware can authenticate a digitally signed UEFI image, such as an operating system loader.
User	A user is subject to configuration policies applied to the operating system by administrators. On some systems under certain configurations, a normal user can temporarily elevate privileges to that of an administrator. At that time, such a user should be considered an administrator.
ZFS	Zettabyte file system (ZFS) is the default Oracle Solaris disk-based and root file system. The ZFS file system has some features which are not found in any other file system, such as, creating storage pools, snapshots, and using copy-on write semantics.

2 TOE Description

2.1 Type

4 The TOE is a general-purpose OS.

2.2 Usage

5 The expected use cases (as defined by the OSPP) for the TOE are:

a) **Server System.** The OS provides a platform for server-side services, either on physical or virtual hardware.

b) **Cloud System.** The OS provides a platform for providing cloud services running on physical or virtual hardware.

2.3 Logical Scope

6 The TOE provides the following security functions:

a) **Secure Administration.** The TOE enables secure management of its security functions, including:

   i) User authentication with passwords

   ii) Configurable password policies

   iii) Role Based Access Control

   iv) Management of security functions

b) **Protected Communications.** The TOE protects the integrity and confidentiality of communications as follows:

   i) **SSH.** The TOE implements an SSH server to protect communication with remote users and management servers.

   ii) **TLS.** The TOE implements TLS client capabilities to protect communication with TLS servers (e.g. OpenSSL).

c) **Protected Storage.** The TOE implements storage encryption to protect sensitive data.

d) **Trusted Update.** The TOE ensures the authenticity and integrity of software updates to itself and installed applications through digital signatures.

e) **Security Audit.** The TOE generates logs of security relevant events.

f) **Self-Test.** The TOE performs a suite of self-tests to ensure the correct operation and enforcement of its security functions.

g) **Self-Protection.** The TOE implements execution environment-based mitigations:

   i) Address Space Layout Randomization

   ii) Stack buffer overflow protection using stack canaries

h) **Cryptographic Operations.** The TOE implements multiple cryptographic modules. Relevant Cryptographic Algorithm Validation Program (CAVP) certificates are shown in section 6.2.

2.4 Physical Scope

2.4.1 Software

7 The TOE is the following software:

a) Oracle Solaris 11.4

8 The TOE is downloaded by users from the Oracle Software Delivery Cloud at https://edelivery.oracle.com/

2.4.2 Guidance Documents

9 The TOE includes the following guidance documents:

a) [CC Guide] - Oracle Solaris 11.4 Common Criteria Guide, v1.3 (PDF) - https://www.oracle.com/corporate/security-practices/assurance/development/external-security-evaluations/common-criteria/certifications.html

b) [Info] - Oracle Solaris 11.4 Information Library - https://docs.oracle.com/cd/E37838_01/

2.4.3 Non-TOE Components

10 The TOE operates with the following components in the environment:

a) **TLS Server.** The TOE is capable of securely communicating with a TLS server.

b) **Hardware Platforms.** The TOE was tested on the hardware platforms shown in Table 4. A full Hardware Compatibility list may be obtained from: https://www.oracle.com/webfolder/technetwork/hcl/data/s11ga/index.html

Table 4: Hardware Platforms

Model	CPU
Oracle SPARC T8-2	SPARC M8
Oracle Server X8-2	Intel Xeon Gold 5200 series

2.5 Excluded Functionality

11 This CC evaluation only covers the functionality identified in section 2.3 when Oracle Solaris is configured in accordance with [CC Guide].

3 Security Problem Definition

12 The Security Problem Definition is reproduced from the OSPP.

3.1 Threats

Table 5: Threats

Identifier	Description
T.NETWORK_ATTACK	An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may engage in communications with applications and services running on or part of the OS with the intent of compromise. Engagement may consist of altering existing legitimate communications.
T.NETWORK_ EAVESDROP	An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may monitor and gain access to data exchanged between applications and services that are running on or part of the OS.
T.LOCAL_ATTACK	An attacker may compromise applications running on the OS. The compromised application may provide maliciously formatted input to the OS through a variety of channels including unprivileged system calls and messaging via the file system.
T.LIMITED_ PHYSICAL_ACCESS	An attacker may attempt to access data on the OS while having a limited amount of time with the physical device.

3.2 Assumptions

Table 6: Assumptions

Identifier	Description
A.PLATFORM	The OS relies upon a trustworthy computing platform for its execution. This underlying platform is out of scope of this PP.
A.PROPER_USER	The user of the OS is not willfully negligent or hostile, and uses the software in compliance with the applied enterprise security policy. At the same time, malicious software could act as the user, so requirements which confine malicious subjects are still in scope.
A.PROPER_ADMIN	The administrator of the OS is not careless, willfully negligent or hostile, and administers the OS within compliance of the applied enterprise security policy.

3.3 Organizational Security Policies

13 None defined.

4 Security Objectives

14 The security objectives are reproduced from the OSPP.

4.1 Security Objectives for the TOE

Table 7: Security Objectives for the TOE

Identifier	Description
O.ACCOUNTABILITY	Conformant OSes ensure that information exists that allows administrators to discover unintentional issues with the configuration and operation of the operating system and discover its cause. Gathering event information and immediately transmitting it to another system can also enable incident response in the event of system compromise.
O.INTEGRITY	Conformant OSes ensure the integrity of their update packages. OSes are seldom if ever shipped without errors, and the ability to deploy patches and updates with integrity is critical to enterprise network security. Conformant OSes provide execution environment-based mitigations that increase the cost to attackers by adding complexity to the task of compromising systems.
O.MANAGEMENT	To facilitate management by users and the enterprise, conformant OSes provide consistent and supported interfaces for their security-relevant configuration and maintenance. This includes the deployment of applications and application updates through the use of platform-supported deployment mechanisms and formats, as well as providing mechanisms for configuration and application execution control.
O.PROTECTED_ STORAGE	To address the issue of loss of confidentiality of credentials in the event of loss of physical control of the storage medium, conformant OSes provide data-at-rest protection for credentials. Conformant OSes also provide access controls which allow users to keep their files private from other users of the same system.
O.PROTECTED_ COMMS	To address both passive (eavesdropping) and active (packet modification) network attack threats, conformant OSes provide mechanisms to create trusted channels for CSP and sensitive data. Both CSP and sensitive data should not be exposed outside of the platform.

4.2 Security Objectives for the Operational Environment

Table 8: Security Objectives for the Operational Environment

Identifier	Description
OE.PLATFORM	The OS relies on being installed on trusted hardware.

Identifier	Description
OE.PROPER_USER	The user of the OS is not willfully negligent or hostile, and uses the software within compliance of the applied enterprise security policy. Standard user accounts are provisioned in accordance with the least privilege model. Users requiring higher levels of access should have a separate account dedicated for that use.
OE.PROPER_ADMIN	The administrator of the OS is not careless, willfully negligent or hostile, and administers the OS within compliance of the applied enterprise security policy.

5 Security Requirements

5.1 Conventions

15 This document uses the following font conventions to identify the operations defined by the CC:

a) **Assignment.** Indicated with italicized text.

b) **Refinement.** Indicated with bold text (for added text) and strikethroughs (for deleted text).

c) **Selection.** Indicated with underlined text.

d) **Assignment within a Selection:** Indicated with italicized and underlined text.

e) **Iteration.** Indicated by adding a string starting with "/" (e.g. "FCS_COP.1/Hash").

5.2 Extended Components Definition

16 The following extended components are specified by the OSPP, which does not provide a formal definition of each component.

Table 9: Extended Components

Requirement	Title
FCS_RBG_EXT.1	Random Bit Generation
FCS_STO_EXT.1	Storage of Sensitive Data
FCS_SSH_EXT.1	SSH Protocol
FCS_SSHC_EXT.1	SSH Protocol - Client
FCS_SSHS_EXT.1	SSH Protocol - Server
FCS_TLSC_EXT.1	TLS Client Protocol
FDP_IFC_EXT.1	Information flow control
FDP_ACF_EXT.1	Access Controls for Protecting User Data
FIA_X509_EXT.1	X.509 Certificate Validation
FIA_X509_EXT.2	X.509 Certificate Authentication
FMT_MOF_EXT.1	Management of security functions behavior
FMT_SMF_EXT.1	Specification of Management Functions
FPT_ACF_EXT.1	Access controls
FPT_ASLR_EXT.1	Address Space Layout Randomization

Requirement	Title
FPT_SBOP_EXT.1	Stack Buffer Overflow Protection
FPT_TST_EXT.1	Boot Integrity
FPT_TUD_EXT.1	Trusted Update
FPT_TUD_EXT.2	Trusted Update for Application Software
FTP_ITC_EXT.1	Trusted channel communication

5.3 Functional Requirements

Table 10: Summary of SFRs

Requirement	Title	Type
FAU_GEN.1	Audit Data Generation (Refined)	Mandatory
FCS_CKM.1	Cryptographic Key Generation (Refined)	Mandatory
FCS_CKM.2	Cryptographic Key Establishment (Refined)	Mandatory
FCS_CKM_EXT.4	Cryptographic Key Destruction	Mandatory
FCS_COP.1(1)	Cryptographic Operation - Encryption/Decryption (Refined)	Mandatory
FCS_COP.1(2)	Cryptographic Operation - Hashing (Refined)	Mandatory
FCS_COP.1(3)	Cryptographic Operation - Signing (Refined)	Mandatory
FCS_COP.1(4)	Cryptographic Operation - Keyed-Hash Message Authentication (Refined)	Mandatory
FCS_COP.1/SSH	FCS_COP.1/SSH Cryptographic Operation - Encryption/Decryption (Refined)	Selection
FCS_RBG_EXT.1	Random Bit Generation	Mandatory
FCS_SSH_EXT.1	SSH Protocol	Selection
FCS_SSHS_EXT.1	SSH Protocol - Server	Selection
FCS_STO_EXT.1	Storage of Sensitive Data	Mandatory
FCS_TLSC_EXT.1	TLS Client Protocol	Mandatory
FDP_ACF_EXT.1	Access Controls for Protecting User Data	Mandatory
FIA_AFL.1	Authentication Failure Management (Refined	Mandatory

Requirement	Title	Type
FIA_UAU.5	Multiple Authentication Mechanisms (Refined)	Mandatory
FIA_X509_EXT.1	X.509 Certificate Validation	Mandatory
FIA_X509_EXT.2	X.509 Certificate Authentication	Mandatory
FMT_MOF_EXT.1	Management of security functions behavior	Mandatory
FMT_SMF_EXT.1	Specification of Management Functions	Mandatory
FPT_ACF_EXT.1	Access controls	Mandatory
FPT_ASLR_EXT.1	Address Space Layout Randomization	Mandatory
FPT_SBOP_EXT.1	Stack Buffer Overflow Protection	Mandatory
FPT_TST_EXT.1	Boot Integrity	Mandatory
FPT_TUD_EXT.1	Trusted Update	Mandatory
FPT_TUD_EXT.2	Trusted Update for Application Software	Mandatory
FTP_ITC_EXT.1	Trusted channel communication	Mandatory
FTP_TRP.1	Trusted Path	Mandatory

5.3.1 Security Audit (FAU)

FAU_GEN.1 Audit Data Generation

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

a) Start-up and shut-down of the audit functions;

b) All auditable events for the [not specified] level of audit; and [

- Authentication events (Success/Failure);
- Use of privileged/special rights events (Successful and unsuccessful security, audit, and configuration changes);
- Privilege or role escalation events (Success/Failure);
    - [File and object events (Successful and unsuccessful attempts to create, access, delete, modify, modify permissions),
    - User and Group management events (Successful and unsuccessful add, delete, modify, disable, enable, and credential change),
    - Audit and log data access events (Success/Failure),
    - Cryptographic verification of software (Success/Failure),

- Attempted application invocation with arguments (Success/Failure e.g. due to software restriction policy),
- System reboot, restart, and shutdown events (Success/Failure),
- Kernel module loading and unloading events (Success/Failure),
- Administrator or root-level access events (Success/Failure)].

FAU_GEN.1.2 The **OS** 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 (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [*none*].

5.3.2 Cryptographic Support (FCS)

FCS_CKM.1 Cryptographic Key Generation (Refined)

FCS_CKM.1.1 The **OS** shall generate **asymmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm [

- RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3,
- ECC schemes using "NIST curves" P-256, P-384 and [P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4,
- FFC Schemes using safe primes that meet the following: 'NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes,
- FFC Schemes using Diffie-Hellman group 14 that meet the following: RFC 3526,

~~] and specified cryptographic key sizes [*assignment: cryptographic key sizes*] that meet the following: [*assignment: list of standards*]~~.

Application Note: This SFR is altered by TD0501.

FCS_CKM.2 Cryptographic Key Establishment (Refined)

FCS_CKM.2.1 The OS shall **implement functionality to perform cryptographic key establishment** in accordance with a specified cryptographic key **establishment** method:[

- RSA-based key establishment schemes that meets the following: RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.2,

- Finite field-based key establishment schemes that meets the following: NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography,
- Key establishment scheme using Diffie-Hellman group 14 that meets the following: RFC 3526,

] ~~that meets the following: [*assignment: list of standards*].~~

Application Note: This SFR is altered by TD0501.

Application Note: The TOE implements FFC Schemes using "safe-prime" groups (identified in Appendix D of SP 800-56A Revision 3)

FCS_CKM_EXT.4 Cryptographic Key Destruction

FCS_CKM_EXT.4.1 The OS shall destroy cryptographic keys and key material in accordance with a specified cryptographic key destruction method [

- For volatile memory, the destruction shall be executed by a [
  - removal of power to the memory,]
- For non-volatile memory that consists of [the invocation of an interface provided by the underlying platform that [
  - instructs the underlying platform to destroy the abstraction that represents the key]]].

FCS_CKM_EXT.4.2 The OS shall destroy all keys and key material when no longer needed.

Application Note: This SFR is altered by TD0365

FCS_COP.1(1) Cryptographic Operation - Encryption/Decryption (Refined)

FCS_COP.1.1(1) The **OS** shall perform [*encryption/decryption services for data*] in accordance with a specified cryptographic algorithm [

- AES-CBC (as defined in NIST SP 800-38A)
-] and [
- AES-GCM (as defined in NIST SP 800-38D)
- AES-CCM (as defined in NIST SP 800-38C)

] and cryptographic key sizes [128-bit, 256-bit] ~~that meet the following: [*assignment: list of standards*].~~

FCS_COP.1(2) Cryptographic Operation - Hashing (Refined)

FCS_COP.1.1(2) The **OS** shall perform [*cryptographic hashing services*] in accordance with a specified cryptographic algorithm [*SHA-1 and* [

- SHA-256,
- SHA-384,
- SHA-512,

]] **and message digest sizes 160 bits and** [

- 256 bits,
- 384 bits,
- 512 bits,

] that meet the following: [FIPS Pub 180-4].

FCS_COP.1(3) Cryptographic Operation - Signing (Refined)

FCS_COP.1.1(3) The **OS** shall perform [*cryptographic signature services (generation and verification)*] in accordance with a specified cryptographic algorithm [

- RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 4,
- ECDSA schemes using "NIST curves" P-256, P-384 and [P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5

] ~~and cryptographic key sizes [assignment: cryptographic algorithm] that meet the following: [assignment: list of standards].~~

FCS_COP.1(4) Cryptographic Operation - Keyed-Hash Message Authentication (Refined)

FCS_COP.1.1(4) The **OS** shall perform [*keyed-hash message authentication services*] in accordance with a specified cryptographic algorithm [SHA-1, SHA-256, SHA-384, SHA-512] **with key sizes** [*160 bits, 256 bits, 384 bits, 512 bits*] **and message digest sizes** [160 bits, 256 bits, 384 bits, 512 bits] that meet the following: [*FIPS Pub 198-1 The Keyed-Hash Message Authentication Code and FIPS Pub 180-4 Secure Hash Standard*].

FCS_COP.1/SSH Cryptographic Operation - Encryption/Decryption (Refined)

FCS_COP.1.1/SSH The SSH software shall perform encryption/decryption services for data in accordance with a specified cryptographic algorithm AES-CTR (as defined in NIST SP 800-38A) mode and cryptographic key sizes [128-bit, 256-bit].

Application Note: This is FCS_COP.1(1) from the SSH EP.

FCS_RBG_EXT.1 Random Bit Generation

FCS_RBG_EXT.1.1 The OS shall perform all deterministic random bit generation (DRBG) services in accordance with NIST Special Publication 800-90A using [

- Hash_DRBG (any)].

FCS_RBG_EXT.1.2 The deterministic RBG used by the OS shall be seeded by an entropy source that accumulates entropy from a [

- software-based noise source,

- platform-based noise source

] with a minimum of [

- 256 bits

] of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

FCS_SSH_EXT.1 SSH Protocol

FCS_SSH_EXT.1.1 The SSH software shall implement the SSH protocol that complies with RFCs 4251, 4252, 4253, 4254 and [5656, 6668] as a [server].

FCS_SSHS_EXT.1 SSH Protocol - Server

FCS_SSHS_EXT.1.1 The SSH server shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based, and [password-based].

FCS_SSHS_EXT.1.2 The SSH server shall ensure that, as described in RFC 4253, packets greater than [*256 kilo*]bytes in an SSH transport connection are dropped.

FCS_SSHS_EXT.1.3 The SSH server shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: aes128-ctr, aes256-ctr, [aes128-cbc, aes256-cbc, aes128-gcm@openssh.com, aes256-gcm@openssh.com].

FCS_SSHS_EXT.1.4 The SSH server shall ensure that the SSH transport implementation uses [ssh-rsa, ecdsa-sha2-nistp256] and [ecdsa-sha2-nistp384] as its public key algorithm(s) and rejects all other public key algorithms.

FCS_SSHS_EXT.1.5 The SSH server shall ensure that the SSH transport implementation uses [hmac-sha1, hmac-sha1-96, hmac-sha2-256, hmac-sha2-512] and [implicit] as its MAC algorithm(s) and rejects all other MAC algorithm(s).

FCS_SSHS_EXT.1.6 The SSH server shall ensure that [diffie-hellman-group14-sha1] and [no other methods] are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHS_EXT.1.7 The SSH server shall ensure that the SSH connection be rekeyed after [no more than 1 Gigabyte of data has been transmitted, no more than 1 hour] using that key.

Application Note: This SFR is altered by TD0446.

FCS_STO_EXT.1 Storage of Sensitive Data

FCS_STO_EXT.1.1 The OS shall implement functionality to encrypt sensitive data stored in non-volatile storage and provide interfaces to applications to invoke this functionality.

FCS_TLSC_EXT.1 TLS Client Protocol

FCS_TLSC_EXT.1.1 The OS shall implement TLS 1.2 (RFC 5246) supporting the following cipher suites: [

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246 ,
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 5246,
- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246,
- TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246,
- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288,
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246,
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246,
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288,
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,

].

Application Note: The above element is altered by TD0441.

FCS_TLSC_EXT.1.2 The OS shall verify that the presented identifier matches the reference identifier according to RFC 6125.

FCS_TLSC_EXT.1.3 The OS shall only establish a trusted channel if the peer certificate is valid.

5.3.3 User Data Protection (FDP)

FDP_ACF_EXT.1 Access Controls for Protecting User Data

FDP_ACF_EXT.1.1 The OS shall implement access controls which can prohibit unprivileged users from accessing files and directories owned by other users.

5.3.4 Identification and Authentication (FIA)

FIA_AFL.1 Authentication failure handling (Refined)

FIA_AFL.1.1 The **OS** shall detect when [

- an administrator configurable positive integer within [1 - 15]

] unsuccessful authentication attempts occur related to **events with** [

- authentication based on user name and password].

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts for an account has been **met**, the **OS** shall: [Account Lockout].

FIA_UAU.5 Multiple Authentication Mechanisms (Refined)

FIA_UAU.5.1 The **OS** shall provide the following authentication mechanisms [

- authentication based on user name and password,
- for use in SSH only, SSH public key-based authentication as specified by the EP for Secure Shell

] to support user authentication.

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

- *Authentication on the local console is based on user name and password, authentication via the SSHv2 protocol first performs the certificate-based authentication which is followed by the user name and password authentication if the certificate-based authentication was unsuccessful*].

FIA_X509_EXT.1 X.509 Certificate Validation

FIA_X509_EXT.1.1 The OS shall implement functionality to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation
- The certificate path must terminate with a trusted CA certificate
- The OS shall validate a certificate path by ensuring the presence of the basicConstraints extension, that the CA flag is set to TRUE for all CA certificates, and that any path constraints are met.
- The TSF shall validate that any CA certificate includes caSigning purpose in the key usage field
- The OS shall validate the revocation status of the certificate using [OCSP as specified in RFC 6960, CRL as specified in RFC 5759].
- The OS shall validate the extendedKeyUsage field according to the following rules:
    - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing Purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
    - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
    - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the EKU field.
    - S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the EKU field.

- OCSP certificates presented for OCSP responses shall have the OCSP Signing Purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the EKU field.
- Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the EKU field. (conditional)

FIA_X509_EXT.1.2 The OS shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

Application Note: This SFR is altered by TD0525.

FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1 The OS shall use X.509v3 certificates as defined by RFC 5280 to support authentication for TLS and [no other protocols] connections.

5.3.5 Security Management (FMT)

FMT_MOF_EXT.1 Management of security functions behavior

FMT_MOF_EXT.1.1 The OS shall restrict the ability to perform the function indicated in the "Administrator" column in FMT_SMF_EXT.1.1 to the administrator.

FMT_SMF_EXT.1 Specification of Management Functions

FMT_SMF_EXT.1.1 The OS shall be capable of performing the following management functions:

Management Function	Administrator	User
Enable/disable [session timeout]	X	
Configure [session] inactivity timeout	X	
Configure local audit storage capacity	X	
Configure minimum password Length	X	
Configure minimum number of special characters in password	X	
Configure minimum number of numeric characters in password	X	
Configure minimum number of uppercase characters in password	X	
Configure minimum number of lowercase characters in password	X	

Management Function	Administrator	User
Configure lockout policy for unsuccessful authentication attempts through [limiting number of attempts during a time period]	X	
Configure host-based firewall	X	
Configure name/address of directory server with which to bind	X	
Configure name/address of audit/logging server to which to send audit/logging records	X	
Configure audit rules	X	
Configure name/address of network time server	X	
[*no additional functions specified*]		

5.3.6 Protection of the TSF (FPT)

FPT_ACF_EXT.1 Access controls

FPT_ACF_EXT.1.1 The OS shall implement access controls which prohibit unprivileged users from modifying:

- Kernel and its drivers/modules
- Security audit logs
- Shared libraries
- System executables
- System configuration files

FPT_ACF_EXT.1.2 The OS shall implement access controls which prohibit unprivileged users from reading:

- Security audit logs
- System-wide credential repositories

FPT_ASLR_EXT.1 Address Space Layout Randomization

FPT_ASLR_EXT.1.1 The OS shall always randomize process address space memory locations with [[*28*]] bits of entropy except for [*any untagged files, the kernel, non-Position-Independent-Executable applications, non-Position-Independent-Code shared libraries*].

FPT_SBOP_EXT.1 Stack Buffer Overflow Protection

FPT_SBOP_EXT.1.1 The OS shall [employ stack-based buffer overflow protections].

FPT_TST_EXT.1 Boot Integrity

FPT_TST_EXT.1.1 The OS shall verify the integrity of the bootchain up through the OS kernel and [

- *[the boot loader, the kernel and kernel modules]*,

] prior to its execution through the use of [

- a digital signature using a hardware-protected asymmetric key].

FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1 The OS shall provide the ability to check for updates to the OS software itself and shall use a digital signature scheme specified in FCS_COP.1(3) to validate the authenticity of the response.

FPT_TUD_EXT.1.2 The OS shall [cryptographically verify] updates to itself using a digital signature prior to installation using schemes specified in FCS_COP.1(3).

Application Note: This SFR is altered by TD0386 and TD0463.

FPT_TUD_EXT.2 Trusted Update for Application Software

FPT_TUD_EXT.2.1 The OS shall provide the ability to check for updates to application software and shall use a digital signature scheme specified in FCS_COP.1(3) to validate the authenticity of the response.

FPT_TUD_EXT.2.2 The OS shall cryptographically verify the integrity of updates to applications using a digital signature specified by FCS_COP.1(3) prior to installation.

Application Note: This SFR is altered by TD0463.

5.3.7 Trusted Path/Channels (FTP)

FTP_ITC_EXT.1 Trusted channel communication

FTP_ITC_EXT.1.1 The OS shall use [

- TLS as conforming to FCS_TLSC_EXT.1,
- SSH as conforming to the EP for Secure Shell

] to provide a trusted communication channel between itself and authorized IT entities supporting the following capabilities: [[*remote user, TLS server]*] that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from disclosure and detection of modification of the channel data.

FTP_TRP.1 Trusted Path

FTP_TRP.1.1 The **OS** shall provide a communication path between itself and [remote, local] users that is logically distinct from other communication paths and

	provides assured identification of its endpoints and protection of the communicated data from [modification, disclosure].
FTP_TRP.1.2	The **OS** shall permit [local users, remote users] to initiate communication via the trusted path.
FTP_TRP.1.3	The **OS** shall require use of the trusted path for [[*all remote administrative actions*]].

5.4 Assurance Requirements

5.4.1 Summary of Requirements

17 The TOE security assurance requirements are summarized in Table 11.

Table 11: Assurance Requirements

Assurance Class	Components	Description
Security Target Evaluation	ASE_CCL.1	Conformance Claims
	ASE_ECD.1	Extended Components Definition
	ASE_INT.1	ST Introduction
	ASE_OBJ.2	Security Objectives
	ASE_REQ.2	Derived Security Requirements
	ASE_SPD.1	Security Problem Definition
	ASE_TSS.1	TOE Summary Specification
Development	ADV_FSP.1	Basic Functional Specification
Guidance Documents	AGD_OPE.1	Operational User Guidance
	AGD_PRE.1	Preparative User Guidance
Life Cycle Support	ALC_CMC.1	Labelling of the TOE
	ALC_CMS.1	TOE CM Coverage
	ALC_TSU_EXT.1	Timely Security Updates
Tests	ATE_IND.1	Independent Testing - conformance
Vulnerability Assessment	AVA_VAN.1	Vulnerability Analysis

5.4.2 Timely Security Updates (ALC_TSU_EXT.1)

18 Oracle's timely security update methodology is published here: https://www.oracle.com/corporate/security-practices/assurance/vulnerability/security-fixing.html

19 Oracle's security vulnerability reporting procedures for users are published here: https://www.oracle.com/corporate/security-practices/assurance/vulnerability/reporting.html

20 Oracle's security alerts are published here: https://www.oracle.com/security-alerts/

6 TOE Summary Specification

21 The following describes how the TOE fulfils each SFR included in section 5.3.

6.1 Security Audit

22 The auditing subsystem of Oracle Solaris keeps a record of how the system is being used. The audit service, *auditd*, is enabled by default. The audit service tracks auditable actions that occur on a system. These auditable actions are defined as Audit Events. Each audit event is connected to a system call or user command and is assigned to one or more Audit Classes (which are convenient containers for large numbers of audit events). */var/audit* stores audit records. The size of an audit file is not limited.

23 By default, when audit records fill the available disk space, the system tracks the number of dropped audit records. A warning is issued when one percent of available disk space remains.

24 The TOE must be configured in accordance with [CC Guide] to ensure the events and information listed below are generated.

6.1.1 Audit Data Generation (FAU_GEN.1)

25 The TOE generates the following audit events:

a) Start-up and shut-down of the audit functions

b) Authentication events (Success/Failure)

c) Use of privileged/special rights events (Successful and unsuccessful security, audit, and configuration changes)

d) Privilege or role escalation events (Success/Failure)

e) File and object events (Successful and unsuccessful attempts to create, access, delete, modify, modify permissions)

f) User and Group management events (Successful and unsuccessful add, delete, modify, disable, enable, and credential change)

g) Audit and log data access events (Success/Failure)

h) Cryptographic verification of software (Success/Failure)

i) Attempted application invocation with arguments (Success/Failure e.g. due to software restriction policy)

j) System reboot, restart, and shutdown events (Success/Failure)

k) Kernel module loading and unloading events (Success/Failure)

l) Administrator or root-level access events (Success/Failure)

26 Each audit event includes the following information:

a) Date and time of the event

b) Type of event

c) Subject identity (if applicable)

d) Outcome (success or failure) of the event

6.2 Cryptographic Support

27 When configured in accordance with [CC Guide] Oracle Solaris provides the following relevant cryptographic implementations:

a) **Oracle Solaris Kernel Cryptographic Framework (CAVP C1895).** Provides cryptography for the kernel module. Filesystem encryption and signature verification for kernel code after early boot is performed by this module.

b) **Oracle OpenSSL FIPS Object Module (CAVP C1651).** Provides cryptographic services for userland consumers. All TLS and SSH related cryptography are provided by this module.

c) **Kernel Loader (CAVP C1937).** The kernel loader contains its own cryptographic service used to perform signature verification of the kernel code during early boot.

d) **Oracle Solaris SSH Key Derivation Function (CAVP C1936).** Provides cryptographic services for SSH key derivation.

28 The Oracle Solaris Userland Cryptographic Framework also provides cryptography for any user application that calls into it however it does not provide any TOE related services.

6.2.1 Key Generation/Establishment (FCS_CKM.1 & FCS_CKM.2)

29 The TOE supports the following asymmetric cryptographic key generation algorithms:

a) RSA, Mod 2048, 3072

b) ECDSA, Curves P-256, P-384, P-521

c) FFS Safe Primes

d) Diffie-Hellman group 14

30 The TOE supports the following key establishment schemes:

a) RSA based schemes, Mod 2048, 3072 (TOE is sender)

b) FFC based schemes / safe primes

c) Diffie-Hellman group 14

31 Table 12 below identifies the scheme being used by each service.

Table 12: Key Generation/Establishment Mapping

Scheme	Usage	SFR	Service
RSA	Key Generation Key Establishment	FCS_TLSC_EXT.1	Outbound TLS
	Key Generation	FCS_SSHS_EXT.1	Remote Administration
FFC (safe primes)	Key Generation Key Establishment	FCS_TLSC_EXT.1	Outbound TLS
DH Group 14	Key Generation	FCS_TLSC_EXT.1	Outbound TLS

Scheme	Usage	SFR	Service
	Key Establishment	FCS_SSHS_EXT.1	Remote Administration

32 In the event of a decryption error, the OS only logs/outputs aggregate generic error messages.

6.2.2 Key Destruction (FCS_CKM_EXT.4)

33 Table 13 identifies the TOE relevant cryptographic keys and related destruction information. The Generator/Initiator column indicates the entity that causes the key to enter volatile memory.

34 For volatile memory, destruction is executed by removal of power to the memory.

35 For non-volatile memory the destruction consists of the invocation of an interface provided by the underlying platform that instructs the underlying platform to destroy the abstraction that represents the key – this occurs at the level of the ZFS File System, further described here: https://docs.oracle.com/cd/E37838_01/html/E61017/index.html

Table 13: Key Destruction

Key	Generator / Initiator	Storage	Destruction
TLS Session Keys[1] (FCS_TLSC_EXT.1.1)	OpenSSL	Volatile	Removal of power
SSH Asymmetric Private Keys (FCS_SSHS_EXT.1.4)	OpenSSL	Persistent – encrypted (with ZFS DEK)	ZFS delete
		Volatile	Removal of power
SSH Session Keys (FCS_SSHS_EXT.1.3)	OpenSSL	Volatile	Removal of power
ZFS Data Encryption Keys (DEK) (FCS_STO_EXT.1.1)	Kernel Module	Persistent – encrypted (with ZFS KEK)	ZFS delete
		Volatile	Removal of power
ZFS Key Encryption Keys (KEK) (FCS_STO_EXT.1.1)	User AES key derived from user entered passphrase (PBKDF2) (Option 1)	Volatile	Removal of power

[1] Since the TOE only implements a TLS client without mutual authentication there are no persistent private keys within the scope of the TOE.

Key	Generator / Initiator	Storage	Destruction
	Kernel Module Raw AES key (Option 2)	External (e.g. removable media)	n/a – external to the TOE.
		Volatile	Removal of power
User Passwords	User User entered password	Persistent SHA-256 hash	ZFS delete

6.2.3 Encryption/Decryption (FCS_COP.1(1) & FCS_COP.1/SSH)

36 The TOE supports encryption/decryption as follows (algorithm-keysize-mode):

a) AES-128-CBC

b) AES-256-CBC

c) AES-128-GCM

d) AES-256-GCM

e) AES-128-CTR

f) AES-256-CTR

g) AES-128-CCM

h) AES-256-CCM

37 For AES CTR mode, the TOE implements a counter based on the Standard Incrementing Function as defined in Appendix B.1 of NIST SP800-38a. The counter values are unique as they start from a random Initialization Vector (IV).

6.2.4 Hashing/HMAC (FCS_COP.1(2) & FCS_COP.1(4))

38 The TOE supports hashing algorithms: SHA-1, SHA-256, SHA-384 and SHA-512 used in:

a) TLS

b) SSH

c) Digital Signatures

39 The TOE support keyed hash algorithms: HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512 used in:

a) TLS

b) SSH

6.2.5 Digital Signatures (FCS_COP.1(3))

40 The TOE supports the following digital signature generation and verification schemes:

a) RSA 2048, 3072

b) ECDSA P-256, P-384, P-512

41 Digital Signatures are used in TLS, SSH, trusted update and verified boot.

6.2.6 Random Bit Generation (FCS_RBG_EXT.1)

42 The TOE implements the following DRBGs:

a) **Oracle Solaris Kernel Cryptographic Framework** – Hash_DRBG(SHA-512)

b) **Oracle OpenSSL FIPS Object Module** – Hash_DRBG(SHA-512)

43 The TOE implements both platform and software noise sources to seed the DRBGs with a minimum of 256 bits of entropy.

6.2.7 SSH / Trusted Paths (FCS_SSH_EXT.1, FCS_SSHS_EXT.1, FTP_ITC_EXT.1 & FTP_TRP.1)

44 The TOE protects remote administrator communications via SSH with the following characteristics:

a) Public key (ssh-rsa, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384) and password-based authentication is supported.

b) If the SSH packets are greater than 256KB they are automatically dropped.

c) Supported encryption algorithms: aes128-ctr, aes256-ctr, aes128-cbc, aes256-cbc, aes128-gcm@openssh.com, aes256-gcm@openssh.com

d) Supported MAC algorithms: hmac-sha1, hmac-sha1-96, hmac-sha2-256, hmac-sha2-512 and implicit (when using @openssh.com encryption).

e) Supported key exchange methods: diffie-hellman-group14-sha1.

f) Connection re-keys occur after 1 hour or after an aggregate of 1 gig of data has been exchanged (whichever occurs first).

45 Administration via the local console is also supported. This access is logically distinct from other communication paths and user authentication is required prior to access being granted. Data is protected from modification and disclosure through physical security.

6.2.8 Storage of Sensitive Data (FCS_STO_EXT.1)

46 Oracle Solaris allows for transparent data encryption functionality to ZFS. All data and file system metadata (such as ownership, access control lists, quota information, and so on) are encrypted when stored persistently in a ZFS pool.

47 When configured in accordance with [CC Guide] the TOE will use ZFS file encryption (AES-GCM-128/256 or AES-CCM-128/256) to protect sensitive data, which includes all persistent keys identified in Table 13.

48 When an encrypted file system is created with ZFS, the TOE generates a data encryption key (DEK). The administrator never has access to the data encryption keys. Instead the administrator manages the wrapping key.

49 Wrapping keys are AES key encryption keys (KEK) of 128 or 256 bits. They are used to decrypt the DEK for a data set. The wrapping key can be changed at any time, even while the data set is mounted and shared. ZFS supports both user passphrase (key derivation) and raw AES keys.

50 In the case of raw AES keys, the [CC Guide] instructs the user to store these externally from the TOE (e.g. on suitably protected removable media or in a remote key management sever using KMIP).

6.2.9 TLS Client (FCS_TLSC_EXT.1 & FTP_ITC_EXT.1)

51 The TOE supports TLS 1.2 (OpenSSL) and the following cipher suites:

a) TLS_RSA_WITH_AES_128_CBC_SHA

b) TLS_RSA_WITH_AES_256_CBC_SHA

c) TLS_RSA_WITH_AES_128_CBC_SHA256

d) TLS_RSA_WITH_AES_256_CBC_SHA256

e) TLS_RSA_WITH_AES_128_GCM_SHA256

f) TLS_RSA_WITH_AES_256_GCM_SHA384

g) TLS_DHE_RSA_WITH_AES_128_CBC_SHA256

h) TLS_DHE_RSA_WITH_AES_256_CBC_SHA256

i) TLS_DHE_RSA_WITH_AES_128_GCM_SHA256

j) TLS_DHE_RSA_WITH_AES_256_GCM_SHA384

52 Reference identifiers may be specified using IPv4/IPv6 addresses or DNS Names using the OpenSSL API. Wildcards are supported. Certificate pinning is not supported.

6.3 User Data Protection

6.3.1 Access Control – Files & Directories (FDP_ACF_EXT.1)

53 Oracle Solaris implements UNIX permissions for controlling access to files and directories as described at https://docs.oracle.com/cd/E37838_01/html/E61022/secfile-60.html

54 In addition, file attributes may be used to add security to ZFS files as described at https://docs.oracle.com/cd/E37838_01/html/E61022/secfile-extatt.html

6.4 Identification and Authentication

6.4.1 Authentication Failure Handling (FIA_AFL.1)

55 Oracle Solaris allows the administrator to configure a login policy that locks a user's account after a configured number of failed authentication attempts. A locked account may be unlocked by the administrator or configured to unlock after a defined time period.

6.4.2 User Authentication (FIA_UAU.5)

56 When configured in accordance with [CC Guide] the TOE provides the following authentication mechanisms:

a) Username and password

b) SSH public key-based authentication

57 Oracle Solaris leverages the Pluggable Authentication Module (PAM) authentication mechanism. For password-based authentication, when the user provides the correct username and password, this is compared to the known user database and if they match then the user is granted access. Otherwise, the user will not be granted access to the TOE. Password authentication is based on a SHA-256 hash comparison when configured according to the [CC Guide].

58 When using public key-based authentication for SSH, the client creates a key pair and uploads the public key to the TOE. This is placed in a file called authorized_keys within the ~/.ssh directory in the user account's home directory on the TOE (server). The server uses the public key in this file to encrypt a challenge message to the client. If the client can prove that it was able to decrypt this message, it has demonstrated that it owns the associated private key. The server then can set up the environment for the client, and authentication is successful. If the public key authentication is unsuccessful, the user is prompted for a username and password.

6.4.3 Certificate Validation (FIA_X509_EXT.1 & FIA_X509_EXT.2)

59 The TOE is capable of performing validation of TLS server X.509 certificates.

60 Certificates are checked for several validation characteristics:

a) If the certificate 'notAfter' date is in the past, then this is an expired certificate which is considered invalid;

b) The certificate chain must terminate with a trusted CA certificate;

c) Server certificates consumed by the TOE TLS client must have a 'serverAuthentication' extendedKeyUsage purpose;

d) A trusted CA certificate is defined as any certificate loaded into the TOE trust store that has, at a minimum, a basicConstraints extension with the CA flag set to TRUE;

e) CA certificates must include the caSigning purpose in the key usage field.

61 Certificate revocation checking is performed using CRL or OCSP.

62 The X.509 certificates are validated using the certificate path validation algorithm defined in RFC 5280, which can be summarized as follows:

a) The public key algorithm and parameters are checked

b) The current date/time is checked against the validity period revocation status is checked

c) Issuer name of X matches the subject name of X+1

d) Name constraints are checked

e) Policy OIDs are checked

f) Policy constraints are checked; issuers are ensured to have CA signing bits

g) Path length is checked

h) Critical extensions are processed

63 If, during the entire trust chain verification activity, any certificate under review fails a verification check, then the entire trust chain is deemed untrusted.

6.5 Security Management

6.5.1 Administrator Privileges (FMT_MOF_EXT.1 & FMT_SMF_EXT.1)

64 Oracle Solaris is capable of performing the management functions marks with 'X' listed at FMT_SMF_EXT.1.1 in section 5.3.5 above.

65 Most management activities are restricted to the root user. 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.

66 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.

67 Due to privileges being controlled by permissions, this prevents users from performing management functions that they do not have access to.

6.6 Protection of the TSF

6.6.1 System File Protection (FPT_ACF_EXT.1)

68 The TOE implements access control to the following security relevant data:

a) /kernel: contains Kernel modules and device drivers

b) /usr/kernel: contains Kernel modules and device drivers

c) /var/audit: contains audit data

d) /lib, /lib64, /usr/lib and /usr/lib64 contains shared libraries

e) /bin, /sbin, /usr/bin, and /usr/sbin contains system executables.

f) /etc: contains system configuration files.

69 This access control prohibits unprivileged users from reading security audit logs and system-wide credential repositories.

6.6.2 Address Space Layout Randomization (FPT_ASLR_EXT.1)

70 Oracle Solaris tags many of its userland binaries with the address space layout randomization (ASLR) security extension. ASLR randomizes the starting address of key parts of an address space.

71 The default Oracle Solaris value for ASLR is *tagged-binaries*. Many binaries in Oracle Solaris are tagged to use ASLR with the following exceptions:

a) the kernel;

b) non-Position-Independent-Executable applications; and

c) non-Position-Independent-Code shared libraries.

6.6.3 Stack Buffer Overflow Protection (FPT_SBOP_EXT.1)

72 Attacks that cause buffers on the stack to overflow try to insert new code on the stack and cause the program to execute it. Removing execute permission from the stack memory prevents these attacks from succeeding.

73 Oracle Solaris provides the *nxheap* and *nxstack* security extensions to systematically make the stack and heap of all Oracle Solaris processes non-executable.

74 64-bit processes always have non-executable stacks. By default, 32-bit SPARC processes have executable stacks, however, the *nxstack* security extension, which is enabled by default, prevents the stacks of 32-bit processes from being executable.

75 Additional detail is provided here:
https://docs.oracle.com/cd/E37838_01/html/E61021/sysauth-nx.html#scrolltoc

6.6.4 Boot Integrity (FPT_TST_EXT.1)

76 In Oracle Solaris, boot verification is performed by means of *elfsign* signature. At the factory, Oracle Solaris kernel modules (ELF objects) are digitally signed. The signature is created by using the SHA-256 hash of selected ELF records in an object file. The SHA-256 hash is signed with an RSA-2048 private key. The public key is distributed from the */etc/certs/elfsign* directory while the private key is not distributed.

77 All keys are stored in the system's pre-boot environment, which is the software or firmware that runs prior to the booting of Oracle Solaris. The firmware loads and boots *platform/.../unix*. The pre-boot environment / TOE platforms provide hardware-backed protection of the keys as follows:

a) Oracle ILOM for SPARC

b) UEFI Secure Boot (BIOS menu) for x86

78 Verified boot automates the verification of the elfsign signatures of Oracle Solaris kernel modules. With verified boot, the administrator can create a verifiable chain of trust in the boot process beginning from system reset through the completion of the boot process.

79 During a system boot, each block of code that is started in the boot process verifies the next block that needs to be loaded. The sequence of verification and loading continues until the last kernel module is loaded.

80 When a power cycle is subsequently performed on the system, a new sequence of verification begins. The administrator can also configure verified boot to take the appropriate action in the event of verification failure.

81 The firmware verifies and then loads the Oracle Solaris */platform/.../unix* module, the initial Oracle Solaris module. In turn, the Oracle Solaris kernel runtime loader *krtld*, which is part of the unix module, verifies and loads the generic UNIX (*genunix*) module and subsequent modules.

82 The *boot_policy* property manages verified boot behavior when loading kernel modules during the boot process.

6.6.5 Software Updates (FPT_TUD_EXT.1 & FPT_TUD_EXT.2)

83 The Oracle Solaris Image Packaging System (IPS) is a framework that enables the following tasks:

a) List and search software packages

b) Install, update, and remove software

c) Upgrade to a new Oracle Solaris operating system release

84 The IPS interfaces allows administrators to restrict which packages can be installed, which versions of packages can be installed, and how installed software needs to be signed.

85 Oracle Solaris software is distributed in IPS packages. IPS packages are stored in IPS package repositories, which are populated by IPS publishers. IPS packages are installed into Oracle Solaris images.

86 Oracle Solaris provides the *pkg update* command (and other *pkg* commands) to check for updates to itself and installed applications (software packages).

87 When configured in accordance with the [CC Guide] Oracle Solaris verifies the integrity of updates to itself and applications using RSA 2048 / SHA-256 digital signature. Packages that fail verification will not be installed.

7 Rationale

7.1 Conformance Claim Rationale

88 The following rationale is presented with regard to the PP conformance claims:

a) **TOE type.** As identified in section 2.1, the TOE is a general-purpose OS, consistent with the OSPP.

b) **Security problem definition.** As shown in section 3, the threats, OSPs and assumptions are reproduced directly from the OSPP.

c) **Security objectives.** As shown in section 4, the security objectives are reproduced directly from the OSPP.

d) **Security requirements.** As shown in section 5, the security requirements are reproduced directly from the OSPP. No additional requirements have been specified.

7.2 Security Objectives Rationale

89 All security objectives are drawn directly from the OSPP. Table 14 reproduces the rationale from the OSPP.

Table 14: Security Objectives Rationale

Threat, Assumption, or OSP	Security Objectives	Rationale
T.NETWORK_ATTACK	O.PROTECTED_COMMS, O.INTEGRITY, O.MANAGEMENT O.ACCOUNTABILITY	The threat T.NETWORK_ATTACK is countered by O.PROTECTED_COMMS as this provides for integrity of transmitted data. The threat T.NETWORK_ATTACK is countered by O.INTEGRITY as this provides for integrity of software that is installed onto the system from the network. The threat T.NETWORK_ATTACK is countered by O.MANAGEMENT as this provides for the ability to configure the OS to defend against network attack. The threat T.NETWORK_ATTACK is countered by O.ACCOUNTABILITY as this provides a mechanism for the OS to report behavior that may indicate a network attack has occurred.

Threat, Assumption, or OSP	Security Objectives	Rationale
T.NETWORK_EAVESDROP	O.PROTECTED_COMMS, O.MANAGEMENT	The threat T.NETWORK_EAVESDROP is countered by O.PROTECTED_COMMS as this provides for confidentiality of transmitted data. The threat T.NETWORK_EAVESDROP is countered by O.MANAGEMENT as this provides for the ability to configure the OS to protect the confidentiality of its transmitted data.
T.LOCAL_ATTACK	O.INTEGRITY O.ACCOUNTABILITY	The objective O.INTEGRITY protects against the use of mechanisms that weaken the TOE with regard to attack by other software on the platform. The objective O.ACCOUNTABILITY protects against local attacks by providing a mechanism to report behavior that may indicate a local attack is occurring or has occurred.
T.LIMITED_PHYSICAL_ACCESS	O.PROTECTED_STORAGE	The objective O.PROTECTED_STORAGE protects against unauthorized attempts to access physical storage used by the TOE.
A.PLATFORM OE.PLATFORM	OE.PLATFORM	The operational environment objective OE.PLATFORM is realized through A.PLATFORM.
A.PROPER_USER	OE.PROPER_USER	The operational environment objective OE.PROPER_USER is realized through A.PROPER_USER.
A.PROPER_ADMIN	OE.PROPER_ADMIN	The operational environment objective OE.PROPER_ADMIN is realized through A.PROPER_ADMIN.

7.3 Security Requirements Rationale

90 All security requirements are drawn directly from the OSPP. Table 15 presents a mapping between objectives and SFRs as presented in the OSPP.

Table 15: OSPP SFR Rationale

Objective	SFR Rationale
O.ACCOUNTABILITY	Addressed by: FAU_GEN.1, FTP_ITC_EXT.1 Rationale: FAU_GEN.1 defines the auditable events that must be generated to diagnose the cause of unexpected system behavior. FTP_ITC_EXT.1 provides a mechanism for the TSF to transmit the audit data to a remote system.
O.INTEGRITY	Addressed by: FPT_SBOP_EXT.1, FPT_ASLR_EXT.1, FPT_TUD_EXT.1, FPT_TUD_EXT.2, FCS_COP.1(2), FCS_COP.1(3), FCS_COP.1(4), FPT_ACF_EXT.1, FPT_SRP_EXT.1, FIA_X509_EXT.1, FPT_TST_EXT.1, FTP_ITC_EXT.1, FPT_W^X_EXT.1, FIA_AFL.1, FIA_UAU.5 Rationale: FPT_SBOP_EXT.1 enforces stack buffer overflow protection that makes it more difficult to exploit running code. FPT_ASLR_EXT.1 prevents attackers from exploiting code that executes in static known memory locations. FPT_TUD_EXT.1 and FPT_TUD_EXT.2 enforce integrity of software updates. FCS_COP.1(2), FCS_COP.1(3), and FCS_COP.1(4) provide the cryptographic mechanisms that are used to verify integrity values. FPT_ACF_EXT.1 guarantees the integrity of critical components by preventing unauthorized modifications of them. FPT_SRP_EXT.1 restricts the execution of unauthorized software. FPT_X509_EXT.1 provides X.509 certificates as a way of validating software integrity. FPT_TST_EXT.1 verifies the integrity of stored code. FPT_W^X_EXT.1 prevents execution of data in writable memory. FIA_UAU.5 provides mechanisms that prevent untrusted users from accessing the TSF and FIA_AFL.1 prevents brute-force authentication attempts. FTP_ITC_EXT.1 provides trusted remote communications which makes a remote authenticated session less susceptible to compromise.
O.MANAGEMENT	Addressed by: FMT_MOF_EXT.1, FMT_SMF_EXT.1, FTA_TAB.1, FTP_TRP.1 Rationale: FMT_SMF_EXT.1 defines the TOE's management functions and FMT_MOF_EXT.1 defines the privileges required to invoke them. FTP_TRP.1 provides one or more secure remote interfaces for management of the TSF and FTA_TAB.1 provides actionable warnings against misuse of these interfaces.
O.PROTECTED_STORAGE	Addressed by: FCS_STO_EXT.1, FCS_RBG_EXT.1, FCS_COP.1(1), FDP_ACF_EXT.1 Rationale: FCS_STO_EXT.1 provides a mechanism by which the TOE can designate data as ‘sensitive’ and subsequently

Objective	SFR Rationale
	require it to be encrypted. FCS_COP.1(1) defines the symmetric algorithm used to encrypt and decrypt sensitive data. FCS_RBG_EXT.1 defines the random bit generator used to create the symmetric keys used to perform this encryption and decryption. FDP_ACF_EXT.1 enforces logical access control on stored data.
O.PROTECTED_COMMS	Addressed by: FCS_TLSC_EXT.1, FCS_TLSC_EXT.2, FCS_TLSC_EXT.3, FCS_TLSC_EXT.4, FCS_DTLS_EXT.1, FCS_RBG_EXT.1, FCS_CKM.1, FCS_CKM.2, FCS_CKM_EXT.4, FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(3), FCS_COP.1(4), FDP_IFC_EXT.1, FIA_X509_EXT.1, FIA_X509_EXT.2, FTP_ITC_EXT.1 Rationale: FCS_TLSC_EXT.1, FCS_TLSC_EXT.2, FCS_TLSC_EXT.3, and FCS_TLSC_EXT.4 define the ability of the TOE to act as a TLS client as a method of enforcing protected communications. FCS_DTLS_EXT.1 defines the ability of the TOE to act as a DTLS client for the same purpose. FCS_CKM.1, FCS_CKM.2, FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(3), FCS_COP.1(4), and FCS_RBG_EXT.1 define the cryptographic operations and key lifecycle activity used to support the establishment of protected communications. FIA_X509_EXT.1 defines how the TSF validates x.509 certificates as part of establishing protected communications. FIA_X509_EXT.2 defines the trusted communication protocols for which the TOE must perform certificate validation operations. FDP_IFC_EXT.1 defines the extent to which the TSF provides an IPsec VPN as a protected communications method. FTP_ITC_EXT.1 defines the trusted communications channels supported by the TOE.