Splunk Enterprise 8.1 Security Target

Acumen Security, LLC.

Document Version: 2.8

Date: January 28, 2021

Table Of Contents

1 Security Target Introduction 5

1.1 Security Target and TOE Reference 5

1.2 TOE Overview 5

1.3 TOE Description 5

1.3.1 Evaluated Configuration 5

1.3.1.1 Excluded Functionality 8

1.3.2 Physical Boundaries 8

1.3.3 Logical Boundaries 8

1.3.3.1 Cryptographic Support 8

1.3.3.2 User Data Protection 9

1.3.3.3 Identification and Authentication 9

1.3.3.4 Security Management 9

1.3.3.5 Privacy 9

1.3.3.6 Protection of the TSF 9

1.3.3.7 Trusted Path/Channels 10

1.3.4 TOE Documentation 10

2 Conformance Claims 11

2.1 CC Conformance 11

2.2 Protection Profile Conformance 11

2.3 Conformance Rationale 11

2.3.1 Technical Decisions 11

3 Security Problem Definition 13

3.1 Threats 13

3.2 Assumptions 13

3.3 Organizational Security Policies 13

4 Security Objectives 14

4.1 Security Objectives for the TOE 14

4.2 Security Objectives for the Operational Environment 15

5 Security Requirements 16

5.1 Conventions 17

5.2 Security Functional Requirements 17

5.2.1 Cryptographic Support (FCS) 17
5.2.2 User Data Protection (FDP) 22
5.2.3 Identification and Authentication (FIA) 22
5.2.4 Security Management (FMT) 23
5.2.5 Privacy (FPR) 24
5.2.6 Protection of TSF (FPT) 24
5.2.7 Trusted Path/Channel (FTP) 25
5.3 TOE SFR Dependencies Rationale for SFRs 26
5.4 Security Assurance Requirements 26
5.5 Rationale for Security Assurance Requirements 26
5.6 Assurance Measures 26
6 TOE Summary Specification 28

Revision History

Version	Date	Description
0.1	2/3/2020	Initial Draft.
0.2	2/4/2020	Updating TDs and SFRs.
0.3	1/12/2020	Updates made to SFRs.
0.4	2/25/2020	Updates made to SFRs.
0.5	3/1/2020	TSS updates.
0.6	3/24/2020	TSS updates.
0.7	3/25/2020	TSS updates.
0.8	3/26/2020	Minor updates to TSS based on vendor responses.
0.9	4/16/2020	Updates based on vendor feedback.
1.0	5/8/2020	Updates based on internal QA review.
1.1	5/27/2020	ST finalized for check-in.
1.2	6/11/2020	Updated TD.
1.3	6/18/2020	Updated TD.
1.4	8/4/2020	Addressing validator comments.
1.5	8/17/2020	Addressing validator comments.
1.6	8/20/2020	Addressing validator comments.
1.7	8/21/2020	Addressing validator comments and Updates based on Internal review.
1.8	8/26/2020	Addressing comments from Vendor.
1.9	8/27/2020	Updates based on Vendor feedback.
2.0	9/29/2020	Updates based on internal reviews.
2.1	10/13/2020	Updating Splunk version.
2.2	10/21/2020	Updated FPT_IDV_EXT.1 SFR and TSS section.
2.3	11/02/2020	Updates based on Internal comments.
2.4	11/03/2020	Updates based on Internal comments.
2.5	11/26/2020	Updates based on Internal QA review.
2.6	11/30/2020	Finalization
2.7	01/21/2021	Updates based on validator checkout comments
2.8	01/28/2021	Minor updates based on validator comments

1 Security Target Introduction

1.1 Security Target and TOE Reference

This section provides information needed to identify and control this ST and its TOE.

Category	Identifier
ST Title	Splunk Enterprise 8.1 Security Target
ST Version	2.8
ST Date	01/28/2021
ST Author	Acumen Security, LLC
TOE Identifier	Splunk Enterprise 8.1
TOE Software Version	8.1
TOE Developer	Splunk Inc
Key Words	Application, software

Table 1 TOE/ST Identification

1.2 TOE Overview

The Target of Evaluation (TOE) is the Splunk Enterprise v8.1 which runs on Red Hat Linux Enterprise (RHEL) v7.7 and v8.2 operating systems. Splunk collects data from various sources such as systems, devices, and interactions and presents the data for real time visibility and analysis. The TOE can be configured as a forwarder and an indexer. When the TOE is configured as the indexer, it will receive data from external sources such as web services, databases, and one or more instance of Splunk configured as a Forwarder. In Forwarder configuration, it will transmit all system generated data to the other instance of Splunk configured as an Indexer.

1.3 TOE Description

1.3.1 Evaluated Configuration

The TOE is the Splunk Enterprise v8.1 which is executed on RHEL operating system. The Splunk Enterprise is a software application that enables users to search, analyze, and visualize the data that is gathered from various components of an IT infrastructure or business industry. The evaluated version of the TOE is v8.1.

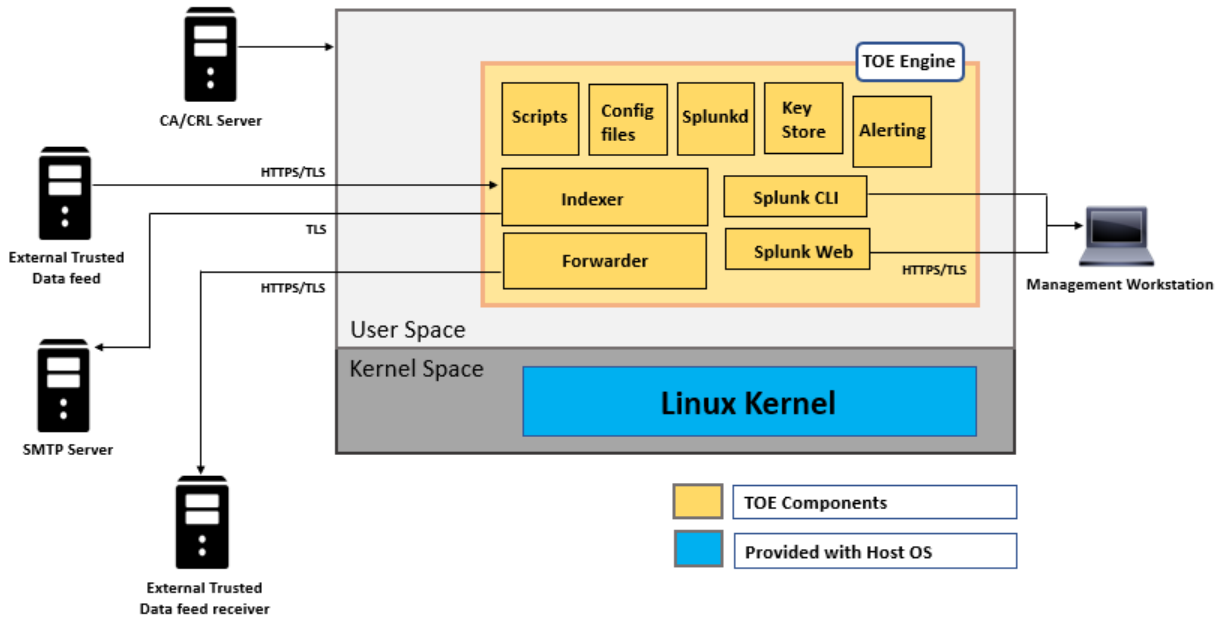

Figure 1: TOE Boundary Diagram

As noted in Figure 1, The TOE consists of many components: Splunkd, Splunk CLI, Splunk Web, Splunk Scripts, Splunk alerts, Splunk KeyStore and Splunk config files. Splunkd is the system process that handles indexing, searching, forwarding, and the Web interface that the user logs into Splunk Enterprise. The above Figure also shows the TOE uses the underlying host platform for storing the Scripts, Key store and Config files which are considered part of the application.

Splunk Web is the web-based user interface for the Splunk to manage the application using a graphical interface. The user logs into Splunk web interface with any supported browser. The communication between the browser and the Splunk Web is over HTTPS/TLS. An administrator must authenticate to Splunk Web using the username and password.

Splunkd is the system process that handles indexing, searching, forwarding, and the Web interface that the user logs into Splunk Enterprise. In order to start this process, the administrator must start the application by using the command "start splunk" in the application directory.

Splunk CLI provides a command line interface that is used to manage the application locally. The CLI service is provided by the underlying host platform. It is mainly used to navigate to the application directories and run the Splunk specific commands. It has the same functionality as Splunk Web except for the graphical representation.

Splunk alerts are actions which gets triggered when a specific criterion is met which is defined by the user. Different types of alerts can be configured in Splunk. In this evaluation email alerts are configured that sends an email to the SMTP server when an action is triggered.

Splunk Scripts can be created to automate the Splunk functionality, an administrator can create simple scripts to start and stop the Splunk application.

Splunk KeyStore is mainly used to store the data for keys, the KeyStore is accessed using the Gnome keyring. Splunk requires at least two partitions of the underlying platform to be LUKS encrypted.

Splunk Enterprise configuration settings are stored in configuration files. These files are available

on Splunk with an extension .conf and are easily readable and editable if the user has appropriate access. Below is the list of Splunk configuration files:

- inputs.conf
- outputs.conf
- server.conf
- alert_actions.conf
- web.conf

In the evaluated configuration the TOE is configured to act as both Indexer and Forwarder.

Splunk Indexer is a Splunk instance that is installed on the physical Red Hat Enterprise Server and is configured to receive the data from the Splunk Forwarder instance. The communication between the Indexer and the Forwarder is over HTTPS/TLS.

Splunk Indexer also communicates with SMTP server for sending email alerts, the communication between the Indexer and the SMTP server is over HTTPS/TLS.

Splunk Forwarder is a Splunk instance that is installed on the physical Red Hat Enterprise Server and is configured to send the data to the Splunk Indexer instance. The communication between the Forwarder and the Indexer is over HTTPS/TLS.

The external trusted data feed is an external data source for transmitting non-TSF related data to the TOE Indexer for populating Splunk's datastore.

The external trusted data feed receiver is an external data source for receiving non-TSF related data from the TOE Forwarder for populating Splunk's datastore.

In the evaluated configuration, the software was installed on the following hardware:

- Dell PowerEdge R430 Server with Intel Xeon E5-2630 v4 (Broadwell)

Note: The TOE is the application software only. The host platforms are not part of the evaluation.

The TOE supports secure connectivity with several other IT environment devices as described below:

Component	Required	Usage/Purpose Description
External Trusted Data Feed	Yes	External data source for transmitting non-TSF related data to the TOE indexer for populating Splunk's datastore. The external data source must use HTTPS/TLS to communicate with the TOE.
External Trusted Data Feed Receiver	Yes	External data source for receiving non-TSF related data from the TOE forwarder. The external data source must use HTTPS/TLS to communicate with the TOE.
Host Platform	Yes	A general-purpose computer on which the RHEL operating system and the TOE is installed.
Management Workstation	Yes	Used to remotely manage the TOE via HTTPS/TLS interface.
SMTP Server	Yes	External data source for receiving non-TSF related data from the TOE Indexer. The external data source must use HTTPS/TLS to communicate with the TOE.
CRL Server	Yes	Server which contains updated revocation list for the TOE.

Table 2 IT Environment Components

1.3.1.1 Excluded Functionality

The following components are included with the Splunk Enterprise v8.1 product but are separately licensed and not considered to be within the TOE boundary:

- Data Fabric Search

Functionality or components that are part of the product but are not part of the TOE relevant functionality are listed below:

- HTTPS administrative interface – port 8089
- The KV store service, port 8191
- The TOE's ability to search and index information is not part of the evaluation. However, the data is needed in order to stimulate events for testing PP related functionality.

1.3.2 Physical Boundaries

The TOE is a software application running on Dell PowerEdge R430 Server with Intel Xeon E5-2630 v4 (Broadwell) processor, and it includes 1TB disk and 32GB RAM. The TOE is Splunk Enterprise v8.1 which runs on Red Hat Linux Enterprise (RHEL) v7.7 and v8.2 64-bit operating system.

1.3.3 Logical Boundaries

The TOE provides the security functionality required by [SWAPP] and [TLS v1.1 package].

1.3.3.1 Cryptographic Support

The TOE platform provides HTTPS/TLS functionality to securely communicate with trusted entities. TOE is shipped with the OpenSSL which performs the TOE's cryptographic operations. TOE leverages the services of the underlying platform to generate entropy for deterministic random bit generator and key store to store the key data.

The following table contains the CAVP algorithm certificates:

Algorithm	Related SFRs	Description	Modes Supported	CAVP Certificate #
AES	FCS_COP.1(1)	Used for Symmetric Encryption/Decryption	GCM (256,128), CBC (256)	C1827 C1828
DRBG	FCS_RBG_EXT.1.1	Deterministic random bit generation	CTR_DRBG AES	C1827 C1828
ECDSA	FCS_CKM_EXT.1.1 FCS_COP.1(3) FCS_CKM.2	186-4 Key Pair Generation and Private Key Validation Signature Generation and Signature Verification ECC Key Establishment	P-256, P-384 and P-521	C1827 C1828 A878 A879
HMAC	FCS_COP.1(4)	Keyed-Hash Message Authentication	HMAC-SHA-256 and HMAC-SHA-384	C1827 C1828

Algorithm	Related SFRs	Description	Modes Supported	CAVP Certificate #
SHS	FCS_COP.1(2)	Cryptographic Hashing Services	SHA-256, SHA-384 and SHA-512	C1827 C1828

Table 3 CAVP Certificate References

1.3.3.2 User Data Protection

The TOE is installed on the encrypted partition of the underlying host platform to secure its data. The private key data for the certificates is stored on the secret storage that can be accessed with the password set to encrypt the partition. Prior to the Installation of TOE the hard drive on the host machine should be encrypted using LUKS. The TOE depends on the underlying platform's network connectivity for its management purpose, sending email alerts to the SMTP server and sending data to the external trusted data feed receiver (TOE Indexer) or receiving the data from the external trusted data feed (TOE Forwarder).

1.3.3.3 Identification and Authentication

The TOE relies on X.509v3 certificate validation functions provided by the platform to authenticate the certificate(s) during the establishment of the HTTPS/TLS trusted channel. If the certificate is found to be invalid the TOE rejects such certificate. Certificate with the unknown revocation status is accepted if the TOE is unable to validate the certificate through CRL.

1.3.3.4 Security Management

The TOE is not shipped with the default credentials used for the Initial authentication. Once the TOE is installed on the RHEL server all the directories and configuration files that are related to the TOE are protected and has the write access to only the user that performed the installation. The TOE has several configuration files that makes communication possible between the other network entities. An administrator can configure the supported TLS cipher suites and curves in these files for the secure communication with the entities and can also query the TOE version.

1.3.3.5 Privacy

The TOE does not request any personally identifiable information (PII) with the intent to transmit the data over the network, thus maintaining privacy of the security administrators and the users.

1.3.3.6 Protection of the TSF

The TOE's platform performs cryptographic self-tests at startup which ensures the TOE's ability to properly operate. The updates must be downloaded manually and installed using the platform's package manager. The TOE platform also verifies all software updates via digital signature wherein the administrator must install the public key of the TOE's developer to check the integrity of any available updates. The TOE uses platform APIs and includes only 3rd party libraries. It also implements stack-based buffer overflow protection along with ASLR (address space layout randomization) and allocating memory for both writing and execution for just-in-time compilation. The TOE supports SElinux and is one of the pre-requisites before installing the TOE application.

1.3.3.7 Trusted Path/Channels

The TOE is a software application. It supports HTTPS/TLS for secure remote administration communication for WebUI. HTTPS/TLS is used for secure communication channel between the TOE indexer and external trusted data feeds (TOE Forwarder), the TOE acting as an Indexer uses TLS to securely send email alerts to a remote SMTP server. The TOE when configured as a Forwarder uses HTTPS/TLS for sending a data to an external data feed receiver (TOE Indexer).

1.3.4 TOE Documentation

- Splunk Enterprise v8.1 Security Target v2.8 [ST]
- Splunk Enterprise v8.1 Common Criteria Guide v0.9 [AGD]

2 Conformance Claims

2.1 CC Conformance

This TOE is conformant to:

- Common Criteria for Information Technology Security Evaluations Part 1, Version 3.1, Revision 5, April 2017
- Common Criteria for Information Technology Security Evaluations Part 2, Version 3.1, Revision 5, April 2017: Part 2 extended
- Common Criteria for Information Technology Security Evaluations Part 2, Version 3.1, Revision 5, April 2017: Part 3 extended

2.2 Protection Profile Conformance

This TOE is conformant to:

- Protection Profile for Application Software, Version 1.3, dated 01 March 2019 [SWAPP]
- Functional Package for Transport Layer Security (TLS), Version 1.1, dated 01 March 2019 [TLS-PKG]

2.3 Conformance Rationale

This Security Target provides exact conformance to Version 1.3 of the Protection Profile for Application Software and Version 1.1 of the Functional Package for Transport Layer Security (TLS). The security problem definition and security objectives in this Security Target are taken from the Protection Profile unmodified. The security requirements in this Security Target are all taken from the Protection Profile and Functional Package performing only operations defined there.

2.3.1 Technical Decisions

All NIAP Technical Decisions (TDs) issued to date that are applicable to [SWAPP] and [TLS-PKG] have been addressed. The following tables identify all applicable TDs.

The following technical decisions were applied for this evaluation:

Identifier	Applicable	Exclusion Rationale (if applicable)
TD0554: iOS/iPadOS/Android AppSW Virus Scan	No	
TD0548: Integrity for installation tests in AppSW PP 1.3	Yes	
TD0544: Alternative testing methods for FPT_AEX_EXT.1.1	No	Product is not Android.
TD0543: FMT_MEC_EXT.1 evaluation activity update	No	Product is not Windows.
TD0540: Expanded AES Modes in FCS_COP	Yes	
TD0521: Updates to Certificate Revocation (FIA_X509_EXT.1)	Yes	
TD0519: Linux symbolic links and FMT_CFG_EXT.1	Yes	
TD0515: Use Android APK manifest in test	No	Product is not Android.
TD0510: Obtaining random bytes for iOS/macOS	No	Product is not iOS.

TD0498: Application Software PP Security Objectives and Requirements Rationale	Yes	
TD0495: FIA_X509_EXT.1.2 Test Clarification	Yes	
TD0486: Removal of PP-Module for VPN Clients from allowed with list	Yes	
TD0473: Support for Client or Server TOEs in FCS_HTTPS_EXT	Yes	
TD0465: Configuration Storage for .NET Apps	No	This is not a windows application.
TD0445: User Modifiable File Definition	Yes	
TD0444: IPsec selections	Yes	
TD0437: Supported Configuration Mechanism	Yes	
TD0435: Alternative to SELinux for FPT_AEX_EXT.1.3	Yes	
TD0434: Windows Desktop Applications Test	No	Product is not Windows.
TD0427: Reliable Time Source	Yes	
TD0416: Correction to FCS_RBG_EXT.1 Test Activity	Yes	

Table 4 SWAPP Technical Decisions

Identifier	Applicable	Exclusion Rationale (if applicable)
TD0513: CA Certificate loading	Yes	
TD0499: Testing with pinned certificates	Yes	
TD0469: Modification of test activity for FCS_TLSS_EXT.1.1 test 4.1	Yes	
TD0442: Updated TLS Ciphersuites for TLS Package	Yes	

Table 5 Functional Package for TLS v1.1 Technical Decisions

3 Security Problem Definition

The security problem definition has been taken from [SWAPP] and is reproduced here for the convenience of the reader. The security problem is described in terms of the threats that the TOE is expected to address, assumptions about the operational environment, and any organizational security policies that the TOE is expected to enforce.

3.1 Threats

The following threats are drawn directly from the [SWAPP].

ID	Threat
T.NETWORK_ATTACK	An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may engage in communications with the application software or alter communications between the application software and other endpoints in order to compromise it.
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 the application and other endpoints.
T.LOCAL_ATTACK	An attacker can act through unprivileged software on the same computing platform on which the application executes. Attackers may provide maliciously formatted input to the application in the form of files or other local communications.
T.PHYSICAL_ACCESS	An attacker may try to access sensitive data at rest.

Table 6 Threats

3.2 Assumptions

The following assumptions are drawn directly from the [SWAPP].

ID	Assumption
A.PLATFORM	The TOE relies upon a trustworthy computing platform with a reliable time clock for its execution. This includes the underlying platform and whatever runtime environment it provides to the TOE.
A.PROPER_USER	The user of the application software is not willfully negligent or hostile, and uses the software in compliance with the applied enterprise security policy.
A.PROPER_ADMIN	The administrator of the application software is not careless, willfully negligent, or hostile, and administers the software in compliance with the applied enterprise security policy.

Table 7 Assumptions

3.3 Organizational Security Policies

There are no OSPs for the application

4 Security Objectives

The security objectives have been taken from [SWAPP] and are reproduced here for the convenience of the reader.

4.1 Security Objectives for the TOE

The following security objectives for the TOE were drawn directly from the [SWAPP].

ID	TOE Objective
O.INTEGRITY	Conformant TOEs ensure the integrity of their installation and update packages, and also leverage execution environment-based mitigations. Software is seldom, if ever, shipped without errors. The ability to deploy patches and updates to fielded software with integrity is critical to enterprise network security. Processor manufacturers, compiler developers, execution environment vendors, and operating system vendors have developed execution environment-based mitigations that increase the cost to attackers by adding complexity to the task of compromising systems. Application software can often take advantage of these mechanisms by using APIs provided by the runtime environment or by enabling the mechanism through compiler or linker options. Addressed by: FDP_DEC_EXT.1, FMT_CFG_EXT.1, FPT_AEX_EXT.1, FPT_TUD_EXT.1
O.QUALITY	To ensure quality of implementation, conformant TOEs leverage services and APIs provided by the runtime environment rather than implementing their own versions of these services and APIs. This is especially important for cryptographic services and other complex operations such as file and media parsing. Leveraging this platform behavior relies upon using only documented and supported APIs. Addressed by: FMT_MEC_EXT.1, FPT_API_EXT.1, FPT_API_EXT.2, FPT_LIB_EXT.1, FPT_TUD_EXT.2, FCS_CKM.1(1)
O.MANAGEMENT	To facilitate management by users and the enterprise, conformant TOEs 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. This also includes providing control to the user regarding disclosure of any PII. Addressed by: FMT_SMF.1, FPT_IDV_EXT.1, FPT_TUD_EXT.1, FPR_ANO_EXT.1, FCS_COP.1(3)
O.PROTECTED_STORAGE	To address the issue of loss of confidentiality of user data in the event of loss of physical control of the storage medium, conformant TOEs will use data-at-rest protection. This involves encrypting data and keys stored by the TOE in order to prevent unauthorized access to this data. This also includes unnecessary network communications whose consequence may be the loss of data. Addressed by: FDP_DAR_EXT.1, FCS_STO_EXT.1, FCS_RBG_EXT.1, FCS_CKM.1(3), FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(4)
O.PROTECTED_COMMS	To address both passive (eavesdropping) and active (packet modification) network attack threats, conformant TOEs will use a trusted channel for sensitive data. Sensitive data includes cryptographic keys, passwords, and any other data specific to the application that should not be exposed outside of the application. Addressed by: FTP_DIT_EXT.1, FCS_RBG_EXT.1, FCS_RBG_EXT.2, FCS_CKM_EXT.1, FCS_CKM.2, FCS_HTTPS_EXT.1, FDP_NET_EXT.1, FIA_X509_EXT.1

Table 8 Objectives for the TOE

4.2 Security Objectives for the Operational Environment

The following security objectives for the operational environment assist the TOE in correctly providing its security functionality. These track with the assumptions about the environment.

ID	Objective for the Operation Environment
OE.PLATFORM	The TOE relies upon a trustworthy computing platform for its execution. This includes the underlying operating system and any discrete execution environment provided to the TOE.
OE.PROPER_USER	The user of the application software is not willfully negligent or hostile, and uses the software within compliance of the applied enterprise security policy.
OE.PROPER_ADMIN	The administrator of the application software is not careless, willfully negligent or hostile, and administers the software within compliance of the applied enterprise security policy.

Table 9 Objectives for the environment

5 Security Requirements

This section identifies the Security Functional Requirements for the TOE and/or Platform. The Security Functional Requirements included in this section are derived from Part 2 of the Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 5 and all international interpretations.

Requirement	Description
FCS_RBG_EXT.1	Random Bit Generation Services
FCS_RBG_EXT.2	Random Bit Generation from Application
FCS_CKM_EXT.1	Cryptographic Key Generation Services
FCS_CKM.1(1)	Cryptographic Asymmetric Key Generation
FCS_CKM.2	Cryptographic Key Establishment
FCS_COP.1(1)	Cryptographic Operation - Encryption/Decryption
FCS_COP.1(2)	Cryptographic Operation - Hashing
FCS_COP.1(3)	Cryptographic Operation - Signing
FCS_COP.1(4)	Cryptographic Operation - Keyed-Hash Message Authentication
FCS_HTTPS_EXT.1 / Client	HTTPS Protocol
FCS_HTTPS_EXT.1/ Server	HTTPS Protocol
FCS_HTTPS_EXT.2	HTTPS Protocol with Mutual Authentication
FCS_STO_EXT.1	Storage of Credentials
FCS_TLS_EXT.1	TLS Protocol
FCS_TLSC_EXT.1	TLS Client Protocol
FCS_TLSC_EXT.2	TLS Client Support for Mutual Authentication
FCS_TLSC_EXT.5	TLS Client Support for Supported Groups Extension
FCS_TLSS_EXT.1	TLS Server Protocol
FDP_DEC_EXT.1	Access to Platform Resources
FDP_NET_EXT.1	Network Communications
FDP_DAR_EXT.1	Encryption Of Sensitive Application Data
FIA_X509_EXT.1	X.509 Certificate Validation
FIA_X509_EXT.2	X.509 Certificate Authentication
FMT_MEC_EXT.1	Supported Configuration Mechanism
FMT_CFG_EXT.1	Secure by Default Configuration
FMT_SMF.1	Specification of Management Functions
FPR_ANO_EXT.1	User Consent for Transmission of Personally Identifiable Information
FPT_API_EXT.1	Use of Supported Services and APIs
FPT_AEX_EXT.1	Anti-Exploitation Capabilities
FPT_TUD_EXT.1	Integrity for Installation and Update
FPT_TUD_EXT.2	Integrity for Installation and Update
FPT_LIB_EXT.1	Use of Third Party Libraries

FTP_DIT_EXT.1	Protection of Data in Transit
FPT_IDV_EXT.1	Software Identification and Versions

Table 10 SFRs

5.1 Conventions

The CC defines operations on Security Functional Requirements: assignments, selections, assignments within selections and refinements. This document uses the following font conventions to identify the operations defined by the CC:

- Assignment: Indicated with *italicized* text;
- Refinement: Indicated with **bold** text;
- Selection: Indicated with underlined text;
- Iteration: Indicated by appending the iteration number in parenthesis, e.g., (1), (2), (3);
- Where operations were completed in the PP itself, the formatting used in the PP has been retained.

Explicitly stated SFRs are identified by having a label 'EXT' after the requirement name for TOE SFRs. Formatting conventions outside of operations matches the formatting specified within the PP.

5.2 Security Functional Requirements

5.2.1 Cryptographic Support (FCS)

FCS_RBG_EXT.1 Random Bit Generation Services

FCS_RBG_EXT.1.1

The application shall [

- *implement DRBG functionality*

] for its cryptographic operations.

FCS_RBG_EXT.2 Random Bit Generation from Application

FCS_RBG_EXT.2.1

The application shall perform all deterministic random bit generation (DRBG) services in accordance with NIST Special Publication 800-90A using [*CTR_DRBG (AES)*]

FCS_RBG_EXT.2.2

The deterministic RBG shall be seeded by an entropy source that accumulates entropy from a platform-based DRBG and [

- *no other 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_CKM_EXT.1 Cryptographic Key Generation Services

FCS_CKM_EXT.1.1

The application shall [

- *implement asymmetric key generation*

].

FCS_CKM.1(1) Cryptographic Asymmetric Key Generation

FCS_CKM.1.1(1)

The **application** shall [

- *implement functionality*

] **to generate asymmetric cryptographic keys in accordance with a specified cryptographic key generation algorithm** [

- ***[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]*** ,

].

FCS_CKM.2 Cryptographic Key Establishment

FCS_CKM.2.1

The application shall [*implement functionality*] to perform cryptographic key establishment in accordance with a specified cryptographic key establishment method:

[

- ***[Elliptic curve-based key establishment schemes]*** *that meets the following:* ***[NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"] ,***

].

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

FCS_COP.1.1(1)

The **application** shall perform *encryption/decryption* in accordance with a specified cryptographic algorithm [

- *AES-CBC (as defined in NIST SP 800-38A) mode,*
- *AES-GCM (as defined in NIST SP 800-38D) mode,*

] and cryptographic key sizes [*128-bit, 256-bit*].

FCS_COP.1(2) Cryptographic Operation - Hashing

FCS_COP.1.1(2)

The **application** shall perform *cryptographic hashing* services in accordance with a specified cryptographic algorithm [

- *SHA-256,*
- *SHA-384,*

- *SHA-512,*

] and message digest sizes [

- *256,*
- *384,*
- *512,*

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

FCS_COP.1(3) Cryptographic Operation - Signing

FCS_COP.1.1(3)

The **application** shall perform *cryptographic signature services (generation and verification)* in accordance with a specified cryptographic algorithm [

- ***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*

] .

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

FCS_COP.1.1(4)

The **application** shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm

- HMAC-SHA-256

and [

- *SHA-384,*

] with key sizes [*384 (in bits) used in HMAC*] and message digest sizes 256 and [*384*] bits that meet the following: FIPS Pub 198-1 *The Keyed-Hash Message Authentication Code* and FIPS Pub 180-4 *Secure Hash Standard*.

FCS_HTTPS_EXT.1/ Client HTTPS Protocol

FCS_HTTPS_EXT.1.1/Client

The application shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2/Client

The application shall implement HTTPS using TLS as defined in the TLS package.

FCS_HTTPS_EXT.1.3/Client

The application shall [*not establish the application-initiated connection*] if the peer certificate is deemed invalid.

FCS_HTTPS_EXT.1/ Server HTTPS Protocol

FCS_HTTPS_EXT.1.1/Server

The application shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2/Server

The application shall implement HTTPS using TLS as defined in the TLS package.

FCS_HTTPS_EXT.2 HTTPS Protocol with Mutual Authentication

FCS_HTTPS_EXT.2.1

The application shall [*not establish the connection*] if the peer certificate is deemed invalid.

FCS_STO_EXT.1 Storage of Credentials

FCS_STO_EXT.1.1

The application shall [

- *invoke the functionality provided by the platform to securely store [credentials for keyring]*

] to non-volatile memory.

FCS_TLS_EXT.1 TLS Protocol

FCS_TLS_EXT.1.1

The product shall implement [

- *TLS as a client,*
- *TLS as a server,*

].

FCS_TLSC_EXT.1 TLS Client Protocol

FCS_TLSC_EXT.1.1

The product shall implement TLS 1.2 (RFC 5246) and [no earlier TLS versions] as a client that supports the cipher suites [

- *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,*
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289,*
- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,*

] and also supports functionality for [

- *mutual authentication,*

].

FCS_TLSC_EXT.1.2

The product shall verify that the presented identifier matches the reference identifier according to RFC 6125.

FCS_TLSC_EXT.1.3

The product shall not establish a trusted channel if the server certificate is invalid [

- *with no exceptions,*

].

FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication

FCS_TLSC_EXT.2.1

The product shall support mutual authentication using X.509v3 certificates.

FCS_TLSC_EXT.5 TLS Client Support for Supported Groups Extension

FCS_TLSC_EXT.5.1

The product shall present the Supported Groups Extension in the Client Hello with the

supported groups [

- *secp256r1,*
- *secp384r1,*
- *secp521r1*

].

FCS_TLSS_EXT.1 TLS Server Protocol

FCS_TLSS_EXT.1.1

The product shall implement TLS 1.2 (RFC 5246) and [*no earlier TLS versions*] as a server that supports the cipher suites [

- *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,*
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289,*
- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,*

] and also supports functionality for [

- *mutual authentication,*

].

FCS_TLSS_EXT.1.2

The product shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0 and [*TLS 1.1*].

FCS_TLSS_EXT.1.3

The product shall perform key establishment for TLS using [

- *ECDHE parameters using elliptic curves [secp256r1, secp384r1, secp521r1] and no other curves,*

].

FCS_TLSS_EXT.2 TLS Server Support for Mutual Authentication

FCS_TLSS_EXT.2.1

The product shall support authentication of TLS clients using X.509v3 certificates.

FCS_TLSS_EXT.2.2

The product shall not establish a trusted channel if the client certificate is invalid.

FCS_TLSS_EXT.2.3

The product shall not establish a trusted channel if the Distinguished Name (DN) or Subject Alternative Name (SAN) contained in a certificate does not match one of the expected identifiers for the client.

5.2.2 User Data Protection (FDP)

FDP_DEC_EXT.1 Access to Platform Resources

FDP_DEC_EXT.1.1

The application shall restrict its access to [

- *network connectivity,*

].

FDP_DEC_EXT.1.2

The application shall restrict its access to [

- *no sensitive information repositories,*

] .

FDP_NET_EXT.1 Network Communications

FDP_NET_EXT.1.1

The application shall restrict network communication to [

- *respond to [remote administration requests (web server), receipt of non-TSF related data from/to external trusted data feeds (indexer functionality)],*
- *[transmission of alerts to environmental SMTP server, transmission of non-TSF related data from/to external trusted data feeds (forwarder functionality]*

] .

FDP_DAR_EXT.1 Encryption Of Sensitive Application Data

FDP_DAR_EXT.1.1

The application shall [

- *leverage platform-provided functionality to encrypt sensitive data,*

] in non-volatile memory.

5.2.3 Identification and Authentication (FIA)

FIA_X509_EXT.1 X.509 Certificate Validation

FIA_X509_EXT.1.1

The application 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 application 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 application shall validate that any CA certificate includes caSigning purpose in the key usage field
- The application shall validate the revocation status of the certificate using [*CRL as specified in RFC 5759*] .
- The application shall validate the extendedKeyUsage (EKU) 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 EKU 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.

FIA_X509_EXT.1.2

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

FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1

The application shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [*HTTPS , TLS*].

FIA_X509_EXT.2.2

When the application cannot establish a connection to determine the validity of a certificate, the application shall [*accept the certificate*].

5.2.4 Security Management (FMT)

FMT_MEC_EXT.1 Supported Configuration Mechanism

FMT_MEC_EXT.1.1

The application shall invoke the mechanisms recommended by the platform vendor for storing and setting configuration options.

FMT_CFG_EXT.1 Secure by Default Configuration

FMT_CFG_EXT.1.1

The application shall provide only enough functionality to set new credentials when configured with default credentials or no credentials.

FMT_CFG_EXT.1.2

The application shall be configured by default with file permissions which protect the application binaries and data files from modification by normal unprivileged users.

FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1

The TSF shall be capable of performing the following management functions [

- *[enable/disable supported TLS cipher suites, and query the version of the TOE].*

] .

5.2.5 Privacy (FPR)

FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information

FPR_ANO_EXT.1.1

The application shall [

- *not transmit PII over a network*

] .

5.2.6 Protection of TSF (FPT)

FPT_API_EXT.1 Use of Supported Services and APIs

FPT_API_EXT.1.1

The application shall use only documented platform APIs.

FPT_AEX_EXT.1 Anti - Exploitation Capabilities

FPT_AEX_EXT.1.1

The application shall not request to map memory at an explicit address except for [*none*].

FPT_AEX_EXT.1.2

The application shall [

- *allocate memory regions with write and execute permissions for only [just-in-time compilation functions sljit, libffi, luajit]*

] .

FPT_AEX_EXT.1.3

The application shall be compatible with security features provided by the platform vendor.

FPT_AEX_EXT.1.4

The application shall not write user-modifiable files to directories that contain executable files unless explicitly directed by the user to do so.

FPT_AEX_EXT.1.5

The application shall be built with stack-based buffer overflow protection enabled.

FPT_TUD_EXT.1 Integrity for Installation and Update

FPT_TUD_EXT.1.1

The application shall [*provide the ability*] to check for updates and patches to the application software.

FPT_TUD_EXT.1.2

The application shall [*provide the ability*] to query the current version of the application software.

FPT_TUD_EXT.1.3

The application shall not download, modify, replace, or update its own binary code.

FPT_TUD_EXT.1.4

The application installation package and its updates shall be digitally signed such that its platform can cryptographically verify them prior to installation.

FPT_TUD_EXT.1.5

The application is distributed [*as an additional software package to the platform OS*]

FPT_TUD_EXT.2 Integrity for Installation and Update

FPT_TUD_EXT.2.1

The application shall be distributed using the format of the platform-supported package manager.

FPT_TUD_EXT.2.2

The application shall be packaged such that its removal results in the deletion of all traces of

the application, with the exception of configuration settings, output files, and audit/log events.

FPT_LIB_EXT.1 Use of Third-Party Libraries

FPT_LIB_EXT.1.1

The application shall be packaged with only [*3rd party libraries as listed in Table 15*].

FPT_IDV_EXT.1 Software Identification and Versions

FPT_IDV_EXT.1.1

The application shall be versioned with [*SWID tags that comply with minimum requirements from ISO/IEC 19770-2:2015*].

5.2.7 Trusted Path/Channel (FTP)

FTP_DIT_EXT.1 Protection of Data in Transit

FTP_DIT_EXT.1.1

The application shall [

- *encrypt all transmitted [sensitive data] with [HTTPS in accordance with FCS_HTTPS_EXT.1, TLS as defined in the TLS Package,*

] between itself and another trusted IT product.

5.3 TOE SFR Dependencies Rationale for SFRs

The Protection Profile for Application Software and TLS Package contains all the requirements claimed in this Security Target. As such, the dependencies are not applicable since the PP has been approved.

5.4 Security Assurance Requirements

The TOE assurance requirements for this ST are taken directly from the Protection Profile for Application Software which are derived from Common Criteria Version 3.1, Revision 5. The assurance requirements are summarized in the table below.

Assurance Class	Components	Components Description
Development	ADV_FSP.1	Basic functional specification
Guidance documents	AGD_OPE.1	Operational user guidance
	AGD_PRE.1	Preparative procedures
Life-cycle support	ALC_CMC.1	Labeling of the TOE
	ALC_CMS.1	TOE CM coverage
	ALC_TSU_EXT.1	Timely Security Updates
Security Target evaluation	ASE_CCL.1	Conformance claims
	ASE_ECD.1	Extended components definition
	ASE_INT.1	ST introduction
	ASE_OBJ.1	Security objectives for the operational environment
	ASE_REQ.1	Stated security requirements
	ASE_TSS.1	TOE summary specification
Tests	ATE_IND.1	Independent testing – conformance
Vulnerability assessment	AVA_VAN.1	Vulnerability survey

Table 11 Security Assurance Requirements

5.5 Rationale for Security Assurance Requirements

The functional specification describes the external interfaces of the TOE, such as the means for a user to invoke a service and the corresponding response of those services. The description includes the interface(s) that enforces a security functional requirement, the interface(s) that supports the enforcement of a security functional requirement, and the interface(s) that does not enforce any security functional requirements. The interfaces are described in terms of their purpose (general goal of the interface), method of use (how the interface is to be used), parameters (explicit inputs to and outputs from an interface that control the behavior of that interface), parameter descriptions (tells what the parameter is in some meaningful way), and error messages (identifies the condition that generated it, what the message is, and the meaning of any error codes). The development evidence also contains a tracing of the interfaces to the SFRs described in this ST.

5.6 Assurance Measures

The TOE satisfies the identified assurance requirements. This section identifies the Assurance Measures applied by [Vendor] to satisfy the assurance requirements. The table below lists the details.

SAR	How the SAR will be met
ADV_FSP.1	The functional specification describes the external interfaces of the TOE; such as the means for a user to invoke a service and the corresponding response of those services. The description includes the interface(s) that enforces a security functional requirement, the interface(s) that supports the enforcement of a security functional requirement, and the interface(s) that does not enforce any security functional requirements. The interfaces are described in terms of their purpose (general goal of the interface), method of use (how the interface is to be used), parameters (explicit inputs to and outputs from an interface that control the behavior of that interface), parameter descriptions (tells what the parameter is in some meaningful way), and error messages (identifies the condition that generated it, what the message is, and the meaning of any error codes).
AGD_OPE.1	The Administrative Guide provides the descriptions of the processes and procedures of how the administrative users of the TOE can securely administer the TOE using the interfaces that provide the features and functions detailed in the guidance.
AGD_PRE.1	The Installation Guide describes the installation, generation, and startup procedures so that the users of the TOE can put the components of the TOE in the evaluated configuration.
ALC_CMC.1	The Configuration Management (CM) documents describe how the consumer identifies the evaluated TOE. The CM documents identify the configuration items, how those configuration items are uniquely identified, and the adequacy of the procedures that are used to control and track changes that are made to the TOE. This includes details on what changes are tracked and how potential changes are incorporated.
ALC_TSU_EXT.1	Splunk uses a systematic method for identifying and providing security relevant updates to the TOEs users via its support infrastructure.
ATE_IND.1	Splunk will provide the TOE for testing.
AVA_VAN.1	Splunk will provide the TOE for testing.

Table 12 TOE Security Assurance Measures

6 TOE Summary Specification

This chapter identifies and describes how the Security Functional Requirements identified above are met by the TOE.

SFR	Rationale
FCS_RBG_EXT.1	The TOE implements its own DRBG functionality for cryptographic operations. For all the deterministic random bit generation services, an OpenSSL implementation of the AES_CTR DRBG is invoked by the TOE. The TOE depends on the underlying platform to the collect the seed entropy The RDRAND instruction is called enough times to produce 64 bits of entropy which is directly fed into the CTR_DRBG. The amount of entropy generated depends on the function that the DRBG is being used for. This amount of entropy generated is always greater than or equal to the security strength of the output data. The TOE does not have the ability to use an alternative DRBG.
FCS_RBG_EXT.2	The entropy source is described in detail in the Entropy Assessment Report.
FCS_CKM_EXT.1	The TOE uses asymmetric key generation services for HTTPS/TLS communications. These asymmetric keys are created on a separate machine and must be installed on the TOE during the installation. Please refer Table#3 CAVP Certificate References for ECDSA.
FCS_CKM.1(1)	The TOE implements functionality to generate cryptographic keys for TLS communications. The TOE supports 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. Please refer Table#3 CAVP Certificate References for ECDSA.
FCS_CKM.2	To establish HTTPS/TLS communications the TOE ensures that the Elliptic curve based key establishment schemes that conforms to NIST SP 800-56A are supported. Please refer Table#3 CAVP Certificate References for ECDSA.
FCS_COP.1(1)	For HTTPS/TLS communications, the TOE performs encryption/decryption using AES-CBC mode (as defined in NIST SP 800-38A) and AES-GCM (as defined in NIST SP 800-38D) modes. The key sizes supported are 128 bits and 256 bits. Please refer Table#3 CAVP Certificate References for AES.
FCS_COP.1(2)	For HTTPS/TLS communications, the TOE performs cryptographic hashing using SHA-256, SHA-384, SHA-512 cryptographic algorithms. The message digest sizes supported are 256, 384 and 512 bits. Please refer Table#3 CAVP Certificate References for SHS.
FCS_COP.1(3)	The TOE performs cryptographic digital signature services for X.509v3 certificate authentication and for software updates. The TOE supports ECDSA schemes using NIST curves P-256, P-384 and P-521. Please refer Table#3 CAVP Certificate References for ECDSA.
FCS_COP.1(4)	For HTTPS/TLS communications, the TOE performs keyed-hash message authentication using HMAC-SHA-256 and HMAC-SHA-384 cryptographic algorithms. The key size supported is 384 bits used in HMAC. The message digest sizes supported are 256 bits and 384 bits.

<table>
<tr><th>SFR</th><th>Rationale</th></tr>
<tr><td></td><td>Please refer Table#3 CAVP Certificate References for HMAC.</td></tr>
<tr><td>FCS_HTTPS_EXT.1 /Client</td><td>The TOE implements the HTTPS/TLS when it acts as a TLS client to transmit information to an external data feed. The TOE will not establish the connection if the peer certificate is deemed invalid. The TOE implements the HTTPS protocol that complies with RFC 2818 and leverages TLS as defined in the TLS package.</td></tr>
<tr><td>FCS_HTTPS_EXT.1/Server</td><td>The GUI is accessed via an HTTPS connection using the TLS implementation and the TOE acts as a server for GUI usage. The TOE also implements HTTPS over TLS for the Indexer function when the TOE receives information from an external trusted data feed. The TOE implements the HTTPS protocol that complies with RFC 2818 and leverages TLS as defined in the TLS package.</td></tr>
<tr><td>FCS_HTTPS_EXT.2</td><td>The TOE supports mutual authentication using X509 v3 certificates.
The TOE will present its client certificate when the TLS server requests the client for a certificate.</td></tr>
<tr><td>FCS_STO_EXT.1</td><td>The gnome Keyring application of the underlying platform stores all the credential data that is being used by the TOE.
gnome Keyring stores password data as well as passphrases that protect private keys and make them available to the TOE. The private keys are encrypted and stored in the file system of the underlying platform. Credentials are required to unlock the keyring.
gnome keyring can automatically unlock the 'login' keyring when the user logs into the TOE.

The data can be written to the gnome keyring at the Splunk CLI using the following command: “secret-storage –write <conf> <stanza> <param>”

<table>
<tr><td>The Following includes the credentials that are stored in the keyring and their purpose: Configuration File</td><td>Stanza</td><td>Parameter</td><td>Rationale</td></tr>
<tr><td>Alerts_action</td><td>email</td><td>auth_password</td><td>Used to store the SMTP password to the keyring when the TOE communicates to the SMTP server.</td></tr>
<tr><td>server</td><td>kvstore</td><td>sslPasssword</td><td>Used to store the passphrase of the key-value store private key in the keyring.</td></tr>
<tr><td>distsearch</td><td>tokenExchKeys</td><td>privateKeyPassphrase</td><td>Used when Splunk is started for the first time.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>SFR</th><th>Rationale</th></tr>
<tr><td></td><td>
<table>
<tr><td>inputs</td><td>SSL</td><td>sslPassword</td><td>Used to store the passphrase of the private key of the TOE configured as an Indexer for receiving the data from the external trusted data feed.</td></tr>
<tr><td>outputs</td><td>tcpout</td><td>sslPassword</td><td>Used to store the passphrase of the private key of the TOE configured as a forwarder for sending the data to the external trusted data feed.</td></tr>
<tr><td>audit</td><td>auditTrail</td><td>PrivateKeyPassphrase</td><td>Used when starting the TOE for the first time.</td></tr>
<tr><td>server</td><td>sslConfig</td><td>sslPassword</td><td>This is used to store the passphrase that protects the Splunkd server private key.</td></tr>
</table>
Table 13 Keys stored in Keyring
</td></tr>
<tr><td>FCS_TLS_EXT.1</td><td>The TOE behaves as a client and server. FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 requirements have been included in this evaluation. The selections are consistent with the selections noted in FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1.</td></tr>
<tr><td>FCS_TLSC_EXT.1</td><td>The TOE act as a TLS client for the trusted channel with SMTP server and when the Splunk instance is configured to transmit data (i.e. Forwarder functionality).

The TOE supports TLS v1.2. The following ciphersuites are supported:

● TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
● TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
● TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The reference identifier is configured by the administrator using the .conf files within the TOE. The reference identifiers supported are the Common Name and Subject Alternative Name (SAN). The TOE doesn’t support IP address, URI names, nor Service names. The TOE does not support certificate pinning and wildcard certificates.

When the TLS client receives a X.509 certificate from the server, the client will compare the reference identifier with the established Subject Alternative Names (SANs) in the</td></tr>
</table>

SFR	Rationale
	certificate. If a SAN is available and does not match the reference identifier, then the verification fails, and the channel is terminated. If there are no SANs of the correct type (FQDN name) in the certificate, then the TOE will compare the reference identifier to the Common Name (CN) in the certificate Subject. If there is no CN, then the verification fails, and the channel is terminated. If the CN exists and does not match, then the verification fails, and the channel is terminated. Otherwise, the reference identifier verification passes, and additional verification actions can proceed.
FCS_TLSC_EXT.2	The TOE supports mutual authentication using X509 v3 certificates. The TOE will present its client certificate when the TLS server requests the client for a certificate.
FCS_TLSC_EXT.5	The TOE supports Elliptic Curves Extension in the Client Hello with the following NIST curves: secp256r1, secp384r1, and secp521r1.
FCS_TLSS_EXT.1	The TOE behaves as a TLS server for the web GUI interface and For the Indexer functionality where the TOE is configured to receive information from an external trusted data feed. The server supports TLS protocol v1.2 and rejects SSL v2.0, SSL v3.0 , TLS v1.0 and TLSv1.1. The TOE supports the following ciphersuites: ● TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 ● TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 ● TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 The TLS server is capable of negotiating ciphersuites that include ECDHE key agreement schemes. The ECDHE key agreement parameters are restricted to secp256r1, secp384r1, secp521r1 key establishment parameters.
FCS_TLSS_EXT.2	The TOE supports mutual authentication of TLS using X509 v3 certificates. The reference identifiers supported are the Common Name and Subject Alternative Name (SAN). The TOE doesn't support IP addresses, URI names, Service names. The TOE does not support certificate pinning and wild card certificates. When the TLS server receives an X.509 certificate from the client, the server will compare the reference identifier with the established Subject Alternative Names (SANs) in the certificate. If a SAN is available and does not match the reference identifier, then the verification fails, and the channel is terminated. If there are no SANs of the correct type (FQDN name) in the certificate, then the TOE will compare the reference identifier to the Common Name (CN) in the certificate Subject. If there is no CN, then the verification fails, and the channel is terminated. If the CN exists and does not match, then the verification fails, and the channel is terminated. Otherwise, the reference identifier verification passes, and additional verification actions can proceed.
FDP_DEC_EXT.1	The TOE depends on its platform to provide the network connectivity for establishing communication channels. The TOE does not require access to any sensitive information repositories.
FDP_NET_EXT.1	The main function of the TOE is to collect data from multiple sources and to parse the data therefore the TOE requires network access to perform these functions. The functionalities of the TOE that require the network access are listed below:

SFR	Rationale
	• In order to facilitate the remote administration to the TOE. The port 8000 is used by the TOE for the webserver to respond to remote administration requests. The management port 8089 and application server port 8191 are initiated by the application for internal support. • The TOE requires access to the network when it behaves as a TLS server (TOE Indexer) to receive non TSF related data from the external data feeds (TOE Forwarder). In this evaluation port 9998 is used as a receiver port by the TOE Indexer. • The TOE requires access to the network when it behaves as a TLS client (TOE Forwarder) to transmit non-TSF data to the external trusted data feed receiver (TOE Indexer). • The TOE requires access to the network when it behaves as a TLS client (TOE Indexer) to transmit alerts to an external SMTP server.
FDP_DAR_EXT.1	The sensitive data in the TOE which is secured by the Operation environment is defined as follows: • The Encrypted private and the full certificate chain for splunkd server is found in the server configuration file under the sslConfig -stanza specified in the sslKeysFile parameter. • The DH parameter file in the server configuration file under the sslConfig stanza in the dhfile parameter. • The Trusted Root CAs list in a single .pem file which can be specified in the server configuration under the sslConfig stanza under the sslRootCAPath parameter. • The Encrypted private and the full certificate chain for KVStore server is found in the server configuration file under the kvstore stanza specified in the sslKeysPath parameter. • The Certificate revocation list file for KVStore can be found in the server configuration file under the kvstore stanza specified in the sslCRLPath parameter. The CRL files used by Splunk must be stored in the $SPLUNK_ETC/auth/crl directory. • The encrypted private key server certificate and the DH Param file that is used by the TOE when it behaves as a TLS server for WEB GUI access can be found in Web configuration file in the settings stanza under the privKeyPath, caCertPath, dhFile parameters. • The encrypted private key found in the Distsearch configuration file in the tokenExchKeys stanza under the privateKey and the publicKey parameters. • The TOE acts as a TLS Server when configured to receive information (i.e. indexer functionality) from an external trusted data feed. The full path to the server certificate for Splunk indexer functionality and the DH Param file can be found in the input configuration file under the SSL stanza in the serverCert and the dhfile parameter. • The TOE acts as a TLS client for when Splunk is configured to transmit non-TSF data (i.e. forwarder functionality). The full path to the client certificate on Splunk forwarder functionality can be found in the Outputs configuration file under the tcpout stanza in the sslCertPath parameter.

SFR	Rationale
	The Linux unified Key Setup encryption is used to secure the private keys and the filesystem objects that comprise the TOE by storing them on a partition drive. The passwd file in the $SPLUNK_ETC directory is used to store the credential data of the TOE which is in turn protected using LUKS. The user-seed.conf file can be used to override the credentials by a security administrator with the credential loaded in the gnome keyring.
FIA_X509_EXT.1	The TOE uses X.509v3 certificates as defined by RFC 5280 to support authentication for HTTPS/TLS communications. The TOE validates 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 application shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates. • The application shall validate the revocation status of the certificate using a Certificate Revocation List (CRL) as specified in RFC 5759] . • The application shall validate the extendedKeyUsage field according to the following rules: o RFC 5280 certificate validation and certificate path validation. o The certificate path must terminate with a trusted CA certificate. o The application 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 o The application shall validate that any CA certificate includes caSigning purpose in the key usage field o The application shall validate the revocation status of the certificate using [CRL as specified in RFC 5759] . o The application shall validate the extendedKeyUsage (EKU) field according to the following rules: o 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. o 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 EKU field. o 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. o 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. o 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. o 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. The TOE will treat a certificate as a CA certificate only if the basicConstraints extension is present and the CA flag is set to TRUE. Certificate revocation checking is performed using

SFR	Rationale
	CRL. If a connection to the CRL server cannot be established", the TOE will accept the certificate. The TOE provides an option to refresh CRL information during runtime using 'splunk reload crl' command in the CLI.
FIA_X509_EXT.2	The TOE includes a .conf file that is used to specify the imported certificates and keys which are being used by the TOE for HTTPS/TLS authentication. The TOE uses certificates by default without any configuration. The security administrator has the ability to specify the support of mutual authentication as per the requirement. As part of the verification process, CRL is used to determine whether the certificate is revoked or not. The security administrator has the ability to specify the CRL path on the TOE to check the revocation status of the certificate during authentication. If the CRL server cannot be contacted, then the TOE will choose to accept the certificate.
FMT_MEC_EXT.1	The TOE depends on the platform and invokes mechanisms recommended by the platform for storing and setting the configuration options. The administrator can make security related changes to the configuration files that reside in /etc/opt/splunk. To ensure that the configuration is in the correct location it can be confirmed by specifying the environment variable SPLUNK_EXT=/etc/opt/splunk. Settings required for CC configuration to satisfy various security functional requirements can be done using the configuration files stored in /etc/opt/splunk. The configuration files include: ● alerts_actions.conf: it is configured to send alerts to the SMTP server. ● inputs.conf: It is used to configure the TOE as a TLS server to receive information from an external trusted data feed. ● outputs.conf: It is used to configure the TOE as a TLS client to transmit non-TSF data to the external trusted data feed receiver. ● server.conf: Used for communications between splunkd and splunk web. ● Web.conf: It is used to configure the TOE as a TLS server for remote Web administration. The parameters like cipher suites, TLS version, reference identifier (CN and SAN), X.509 certificates, certificate validation and mutual authentication can be configured for both the server and the client communications.
FMT_CFG_EXT.1	During the initial startup of the TOE, the TOE prompts the security administrator to create a user with password and there are no default credentials. The TOE will ensure that 'other' users will not have access to SPLUNK_HOME and SPLUNK_ETC directory by overwriting file permissions if needed. For remote administration via WEB GUI, the TOE requires the user to authenticate with username and credentials. The TOE supports a non-root OS user called 'splunk'. The following file permissions are installed by the TOE in the SPLUNK_HOME and the SPLUNK_ETC directory by default. The 'Splunk' user has read-write-execute access. The 'splunk' group has read-execute access. The TOE does not grant access to any 'other' users.
FMT_SMF.1	The TOE provides the following security-related management functions: ● Ability to query the version of the TOE. ● Ability to enable or disable the supported TLS cipher suites. The TOE is managed via remote Web GUI and local CLI interfaces.
FPR_ANO_EXT.1	The TOE does not transmit PII over the network.

<table>
<tr><th>SFR</th><th>Rationale</th></tr>
<tr><td>FPT_API_EXT.1</td><td>The Splunk TOE does not depend on the platform for most of the libraries and scripting languages like javascripts, lua, python as they are present as a part of the TOE.

The Platform APIs which are leveraged by the TOE are listed below:
<table>
<tr><td>__assert_fail</td><td>fmod</td><td>mkstemp</td><td>setresgid</td></tr>
<tr><td>__ctype_b_loc</td><td>fopen</td><td>mkstemp64</td><td>setresuid</td></tr>
<tr><td>__ctype_get_mb_cur_max</td><td>fopen64</td><td>mktime</td><td>setreuid</td></tr>
<tr><td>__ctype_tolower_loc</td><td>fork</td><td>mmap</td><td>setrlimit</td></tr>
<tr><td>__ctype_toupper_loc</td><td>forkpty</td><td>mmap64</td><td>setrlimit64</td></tr>
<tr><td>__cxa_atexit</td><td>fpathconf</td><td>modf</td><td>setsid</td></tr>
<tr><td>__duplocale</td><td>fprintf</td><td>mprotect</td><td>setsockopt</td></tr>
<tr><td>__errno_location</td><td>fputc</td><td>msync</td><td>setuid</td></tr>
<tr><td>__fdelt_chk</td><td>fputs</td><td>munmap</td><td>setvbuf</td></tr>
<tr><td>__finite</td><td>fread</td><td>nanosleep</td><td>shutdown</td></tr>
<tr><td>__fprintf_chk</td><td>free</td><td>nftw</td><td>sigaction</td></tr>
<tr><td>__fread_chk</td><td>freeaddrinfo</td><td>nice</td><td>sigaddset</td></tr>
<tr><td>__freelocale</td><td>freeifaddrs</td><td>nl_langinfo</td><td>sigaltstack</td></tr>
<tr><td>__fxstat</td><td>frexp</td><td>open</td><td>sigemptyset</td></tr>
<tr><td>__fxstat64</td><td>fscanf</td><td>open64</td><td>sigfillset</td></tr>
<tr><td>__h_errno_location</td><td>fseek</td><td>opendir</td><td>siginterrupt</td></tr>
<tr><td>__isinf</td><td>fseeko</td><td>openpty</td><td>signal</td></tr>
<tr><td>__isinff</td><td>fseeko64</td><td>pathconf</td><td>sigpoll</td></tr>
<tr><td>__isnan</td><td>fstatfs</td><td>pause</td><td>sigprocmask</td></tr>
<tr><td>__isoc99_fscanf</td><td>fstatvfs64</td><td>pclose</td><td>sin</td></tr>
<tr><td>__isoc99_sscanf</td><td>fsync</td><td>perror</td><td>sincos</td></tr>
<tr><td>__libc_current_sigrtmax</td><td>ftell</td><td>pipe</td><td>sinh</td></tr>
<tr><td>__libc_current_sigrtmin</td><td>ftello</td><td>popen</td><td>sleep</td></tr>
<tr><td>__libc_start_main</td><td>ftello64</td><td>posix_fadvise</td><td>snprintf</td></tr>
<tr><td>__lxstat</td><td>ftruncate</td><td>posix_memalign</td><td>socket</td></tr>
<tr><td>__lxstat64</td><td>ftruncate64</td><td>pow</td><td>socketpair</td></tr>
<tr><td>__memcpy_chk</td><td>funlockfile</td><td>prctl</td><td>sprintf</td></tr>
<tr><td>__memmove_chk</td><td>fwrite</td><td>pread</td><td>sqrt</td></tr>
<tr><td>__memset_chk</td><td>gai_strerror</td><td>pread64</td><td>sqrtf</td></tr>
<tr><td>__newlocale</td><td>getaddrinfo</td><td>preadv64</td><td>srand</td></tr>
<tr><td>__nl_langinfo_l</td><td>getc</td><td>pthread_attr_destroy</td><td>srand48</td></tr>
</table></td></tr>
</table>

SFR	Rationale			
	__open64_2	getcwd	pthread_attr_init	sscanf
	__pread64_chk	getdtablesize	pthread_attr_setscope	statfs
	__printf_chk	getegid	pthread_attr_setstacksize	statvfs
	__rawmemchr	getenv	pthread_barrier_destroy	statvfs64
	__read_chk	geteuid	pthread_barrier_init	stderr
	__realpath_chk	getgid	pthread_barrier_wait	stdin
	__snprintf_chk	getgrent	pthread_cond_broadcast	stdout
	__sprintf_chk	getgrgid_r	pthread_cond_destroy	stpcpy
	__stack_chk_fail	getgrnam_r	pthread_cond_init	strcasecmp
	__stpcpy_chk	getgroups	pthread_cond_signal	strcat
	__strcat_chk	gethostbyaddr	pthread_cond_timedwait	strchr
	__strcpy_chk	gethostbyname	pthread_cond_wait	strcmp
	__strdup	gethostname	pthread_condattr_destroy	strcoll
	__strncat_chk	getifaddrs	pthread_condattr_init	strcpy
	__strncpy_chk	getitimer	pthread_create	strcspn
	__sysv_signal	getloadavg	pthread_detach	strerror
	__tls_get_addr	getlogin	pthread_getspecific	strftime
	__uflow	getnameinfo	pthread_join	strftimel
	__uselocale	getopt_long	pthread_key_create	strlen
	__vfprintf_chk	getpagesize	pthread_key_create	strncasecmp
	__vsnprintf_chk	getpeername	pthread_key_delete	strncat
	__xmknod	getpgid	pthread_kill	strncmp
	__xpg_strerror_r	getpgrp	pthread_mutex_destroy	strncpy
	__xstat	getpid	pthread_mutex_init	strndup
	__xstat64	getppid	pthread_mutex_lock	strrchr
	abort	getpriority	pthread_mutex_trylock	strsignal

SFR	Rationale			
	accept	getpwent	pthread_mutex_unlock	strspn
	access	getpwnam	pthread_mutexattr_destroy	strstr
	acos	getpwnam_r	pthread_mutexattr_init	strtod
	alarm	getpwuid	pthread_mutexattr_settype	strtod
	alphasort64	getpwuid_r	pthread_once	strtof
	asctime_r	getresgid	pthread_rwlock_destroy	strtok
	asin	getresuid	pthread_rwlock_init	strtol
	atan	getrlimit	pthread_rwlock_rdlock	strtold
	atan2	getrlimit64	pthread_rwlock_tryrdlock	strtoll
	atoi	getrusage	pthread_rwlock_trywrlock	strtoul
	backtrace	getservbyname	pthread_rwlock_unlock	strtoull
	backtrace_symbols	getsid	pthread_rwlock_wrlock	strxfrm
	bind	getsockname	pthread_self	symlink
	bindtextdomain	getsockopt	pthread_setname_np	sync
	btowc	gettext	pthread_setspecific	syscall
	calloc	gettimeofday	pthread_sigmask	sysconf
	ceil	getuid	putc	sysinfo
	ceilf	getwc	putchar	system
	cfmakeraw	gmtime_r	putenv	tan
	chdir	hypot	puts	Tanh
	chmod	if_nametoindex	putwc	tcgetattr
	chown	inet_addr	pwrite	Tcgetpgrp
	chroot	inet_aton	pwrite64	Tcsetattr
	clearerr	inet_ntoa	pwritev64	tcsetpgrp
	clock	inet_ntop	qsort	tempnam
	clock_getres	inet_pton	qsort_r	textdomain
	clock_gettime	initgroups	raise	time
	close	ioctl	rand	textdomain

SFR	Rationale			
	closedir	isalnum	read	times
	confstr	isalpha	readdir	timespec_get
	connect	isatty	readdir_r	tmpfile
	cos	iscntrl	readdir64	tmpfile64
	cosh	isgraph	readlink	tmpnam_r
	ctermid	islower	readv	towlower
	ctime	isprint	realloc	towupper
	difftime	ispunct	recv	truncate
	dirname	isspace	recvfrom	ttyname
	dl_iterate_phdr	isupper	recvmsg	tzset
	dladdr	iswctype	remove	umask
	dlclose	isxdigit	rename	uname
	dlerror	kill	rewind	ungetc
	dlopen	killpg	rmdir	ungetwc
	dlsym	lchown	round	unlink
	dup	ldexp	scandir64	unsetenv
	dup2	link	sched_yield	usleep
	endgrent	listen	select	utime
	endpwent	localeconv	sem_destroy	utimes
	execv	localtime	sem_init	vsnprintf
	execve	localtime_r	sem_post	wait
	execvp	log	sem_timedwait	wait3
	exit	log10	sem_trywait	wait4
	exp	logf	sem_wait	waitpid
	fchdir	lrand48	send	wcrtomb
	fchmod	lrint	sendfile64	wcscmp
	fchown	lseek	sendmsg	wcscoll
	fclose	lseek64	sendto	wcsftime
	fcntl	malloc	setegid	wcslen
	fdatasync	mbrtowc	setenv	wcsnrtombs
	fdopen	mbsnrtowcs	seteuid	wcsxfrm
	feof	mbsrtowcs	setgid	wctod
	ferror	memchr	setgroups	wctype
	fesetround	memcmp	setitimer	wmemchr
	fflush	memcpy	setlinebuf	wmemcmp

<table>
<tr><th>SFR</th><th>Rationale</th></tr>
<tr><td></td><td>
<table>
<tr><td>fgetc</td><td>memmove</td><td>setlocale</td><td>wmemcpy</td></tr>
<tr><td>fgets</td><td>memrchr</td><td>setpgid</td><td>wmemmove</td></tr>
<tr><td>fileno</td><td>memset</td><td>setpgrp</td><td>wmemset</td></tr>
<tr><td>flockfile</td><td>mkdir</td><td>setpriority</td><td>write</td></tr>
<tr><td>floor</td><td>mkdtemp</td><td>setpwent</td><td>writev</td></tr>
<tr><td>floorf</td><td>mkfifo</td><td>setregid</td><td></td></tr>
</table>
Table 14 Platform APIs used by the TOE
</td></tr>
<tr><td>FPT_AEX_EXT.1</td><td>The TOE enables ASLR and stack protection by -fPIE, -pie and the -fstack-protector-strong compilation flags.
The TOE allocates memory regions with write and execute permissions to support just-in-time compilation functions like sljit which is used by PCRE library, libffi which is used to bind the Python code to C code, and luajit which is used to perform JIT compilation of Lua code by node.js java script run time environment.

The TOE is compatible with security features provided by the platform through the SELinux profile which was specifically created by the TOE developer.
The TOE will not write user-modifiable files to directories that contain executable files unless explicitly directed by the user to do so.</td></tr>
<tr><td>FPT_TUD_EXT.1</td><td>The security administrators can verify the current version on the TOE either through Web GUI or CLI.
● The currently installed version can be verified in Help-> About in the Web GUI.
● The “splunk version” command can be used in the CLI to determine the current installed version on the TOE.

The Splunk verifies whether an update is available to the user when authenticated to the web UI. Whenever an update is available, Splunk informs the user with a message displayed in the “Messages” menu. The TOE notifies the user that an update is available but does not install the update automatically. In order to install the updates manually, the user will select the updated URL in the “Messages” menu which will redirect the user to Splunk’s customer portal site. The user authenticates himself to download the updates which comes in RPM software package format. This RPM package will be installed manually by the root administrator using the RPM application already available on the platform.
The RPM package consists of a public key which is installed initially. In order to verify the update against the installed public key, the user with root privileges should run the “rp -K <filename.rpm>” command. The authorized source for the digitally signed updates is "Splunk".</td></tr>
<tr><td>FPT_TUD_EXT.2</td><td>The $SPLUNK_HOME directory where the TOE is installed will be completely erased when the Splunk application is uninstalled. The configuration files or output files will be stored in etc/opt/splunk/ directory and the log files are stored in /opt/splunk/var/log and /opt/splunk/var/lib/splunk directory after uninstalling the application.</td></tr>
<tr><td>FPT_LIB_EXT.1</td><td>The TOE uses the third-party libraries as defined below:
<table>
<tr><td>lib4758cca.so</td><td>libsqlite3.so</td><td>_md5.so</td></tr>
<tr><td>libaep.so</td><td>libsqlite3.so.0</td><td>_multibytecodec.so</td></tr>
</table>
</td></tr>
</table>

SFR	Rationale		
	libatalla.so	libsqlite3.so.0.8.6	_multiprocessing.so
	libcapi.so	libssl.so	_random.so
	libchil.so	libssl.so.1.0.0	_sha256.so
	libcswift.so	libxml2.so	_sha512.so
	libgmp.so	libxml2.so.2	_ssl.so
	libgost.so	libxml2.so.2.9.9	_socket.so
	libnuron.so	libxmlsec1-openssl.so	_struct.so
	libpadlock.so	libxmlsec1-openssl.so.1	array.so
	libsureware.so	libxmlsec1-openssl.so.1.2.24	binascii.so
	libubsec.so	libxmlsec1.so	bz2.so
	libarchive.so	libxmlsec1.so.1	cPickle.so
	libarchive.so.13	libxmlsec1.so.1.2.24	cStringIO.so
	libarchive.so.13.3.3	libxslt.so	datetime.so
	libbson-1.0.so	libxslt.so.1	fcntl.so
	libbson-1.0.so.0	libxslt.so.1.1.30	future_builtins.so
	libbson-1.0.so.0.0.0	libz.so	itertools.so
	libbz2.so	libz.so.1	math.so
	libbz2.so.1	libz.so.1.2.11	operator.so
	libbz2.so.1.0.3	_bisect.so	parser.so
	libcrypto.so	_codecs_iso2022.so	pyexpat.so
	libcrypto.so.1.0.0	_codecs_jp.so	resource.so
	libexslt.so	_collections.so	select.so
	libexslt.so.0	_csv.so	strop.so
	libexslt.so.0.8.18	_ctypes.so	termios.so
	libjemalloc.so	_elementtree.so	time.so
	libjemalloc.so.2	_functools.so	unicodedata.so
	libmongoc-1.0.so	_hashlib.so	zlib.so
	libmongoc-1.0.so.0	_heapq.so	OpenSSL/SSL.so
	libmongoc-1.0.so.0.0.0	_io.so	OpenSSL/crypto.so
	libpcre2-8.so	_json.so	OpenSSL/rand.so
	libpcre2-posix.so	_locale.so	_elementpath.so

SFR	Rationale		
	builder.so	etree.so	clean.so
	diff.so	objectify.so	sax.so
	_xxtestfuzz.cpython-37m-x86_64-linux-gnu.so	_asyncio.cpython-37m-x86_64-linux-gnu.so	array.cpython-37m-x86_64-linux-gnu.so
	_bisect.cpython-37m-x86_64-linux-gnu.so	binascii.cpython-37m-x86_64-linux-gnu.so	_blake2.cpython-37m-x86_64-linux-gnu.so
	fcntl.cpython-37m-x86_64-linux-gnu.so	_bz2.cpython-37m-x86_64-linux-gnu.so	_struct.cpython-37m-x86_64-linux-gnu.so
	_codecs_iso2022.cpython-37m-x86_64-linux-gnu.so	math.cpython-37m-x86_64-linux-gnu.so	_codecs_jp.cpython-37m-x86_64-linux-gnu.so
	_testimportmultiple.cpython-37m-x86_64-linux-gnu.so	_contextvars.cpython-37m-x86_64-linux-gnu.so	select.cpython-37m-x86_64-linux-gnu.so
	_crypt.cpython-37m-x86_64-linux-gnu.so	termios.cpython-37m-x86_64-linux-gnu.so	_csv.cpython-37m-x86_64-linux-gnu.so
	unicodedata.cpython-37m-x86_64-linux-gnu.so	_ctypes.cpython-37m-x86_64-linux-gnu.so	xxlimited.cpython-37m-x86_64-linux-gnu.so
	_datetime.cpython-37m-x86_64-linux-gnu.so	zlib.cpython-37m-x86_64-linux-gnu.so	_decimal.cpython-37m-x86_64-linux-gnu.so
	_testbuffer.cpython-37m-x86_64-linux-gnu.so	_elementtree.cpython-37m-x86_64-linux-gnu.so	_hashlib.cpython-37m-x86_64-linux-gnu.so
	_heapq.cpython-37m-x86_64-linux-gnu.so	_json.cpython-37m-x86_64-linux-gnu.so	_md5.cpython-37m-x86_64-linux-gnu.so
	_testmultiphase.cpython-37m-x86_64-linux-gnu.so	_multibytecodec.cpython-37m-x86_64-linux-gnu.so	_uuid.cpython-37m-x86_64-linux-gnu.so
	_multiprocessing.cpython-37m-x86_64-linux-gnu.so	_opcode.cpython-37m-x86_64-linux-gnu.so	_pickle.cpython-37m-x86_64-linux-gnu.so
	_posixsubprocess.cpython-37m-x86_64-linux-gnu.so	_queue.cpython-37m-x86_64-linux-gnu.so	_random.cpython-37m-x86_64-linux-gnu.so

SFR	Rationale
	_sha1.cpython-37m-x86_64-linux-gnu.so _sha256.cpython-37m-x86_64-linux-gnu.so _sha3.cpython-37m-x86_64-linux-gnu.so _sha512.cpython-37m-x86_64-linux-gnu.so _socket.cpython-37m-x86_64-linux-gnu.so _ssl.cpython-37m-x86_64-linux-gnu.so parser.cpython-37m-x86_64-linux-gnu.so pyexpat.cpython-37m-x86_64-linux-gnu.so resource.cpython-37m-x86_64-linux-gnu.so _elementpath.cpython-37m-x86_64-linux-gnu.so builder.cpython-37m-x86_64-linux-gnu.so etree.cpython-37m-x86_64-linux-gnu.so clean.cpython-37m-x86_64-linux-gnu.so diff.cpython-37m-x86_64-linux-gnu.so objectify.cpython-37m-x86_64-linux-gnu.so sax.cpython-37m-x86_64-linux-gnu.so libjemalloc.so libjemalloc.so.2 **Table 15 TOE Libraries**
FPT_IDV_EXT.1.1	The application is versioned with SWID tags that comply with the minimum requirements from ISO/IEC 19770-2:2015.
FTP_DIT_EXT.1	The TOE leverages HTTPS/TLS v1.2 to encrypt transmitted data over trusted channels and trusted path. The TOE web GUI is accessed via an HTTPS connection using the TLS implementation. The TOE implements the HTTPS/TLS when it acts as a TLS server to receive information from an external trusted data feed. The TOE implements the HTTPS/TLS when it acts as a TLS client to transmit information to an external data feed.
ALC_TSU_EXT.1	Customers are provided access to support on Splunk.com website so that they are able to submit support issues. This is an HTTPS website that does require user authentication. All fixes will be issued in a patch to the Splunk software. All security relevant fixes will be released as a new package similar to any features implemented. The implementation flaws are addressed within 90 days of reporting the issues. The customers are notified of security related fixes directly from the Spunk Customer portal.

Table 16 TOE Summary Specification SFR Description