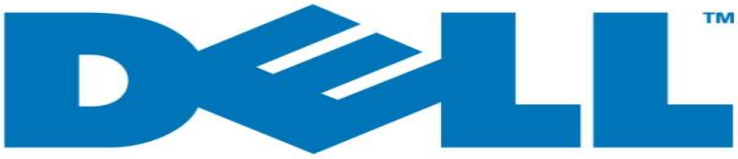


Dell, Inc.
Dell Data Protection | Encryption Personal Edition Non-Recoverable Security Target

Version 1.10

September 29, 2017

Dell, Inc.
One Dell Way
Round Rock, TX 78682

DOCUMENT INTRODUCTION

Prepared By:

Common Criteria Consulting LLC
15804 Laughlin Lane
Silver Spring, MD 20906
http://www.consulting-cc.com

Prepared For:

Dell, Inc.
One Dell Way
Round Rock, TX 78682
http://www.dell.com

REVISION HISTORY

Rev	Description
1.0	August 24, 2016, Initial release
1.1	November 15, 2016, Addressed lab ORs and TRRT responses
1.2	February 1, 2017, Clarified encryption policy constraints
1.3	February 28, 2017, Addressed certifier ORs
1.4	March 24, 2017, Addressed lab ORs
1.5	April 7, 2015, Restored information concerning security updates
1.6	April 17, 2017, Address lab ORs and updates from test preparation
1.7	May 9, 2017, Removed power management states and TOE version change
1.8	July 10, 2017, Addressed certifier comment regarding RDRAND
1.9	August 25, 2017, Addressed TDs and Windows versions
1.10	September 29, 2017, Updated documentation references

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS LIST

AES ..... **Advanced Encryption Standard**
API ..... **Application Program Interface**
AS ..... **Application Software**
ASLR ..... **Address Space Layout Randomization**
CAPI ..... **Cryptography Application Program Interface**
CAVP ..... **Cryptographic Algorithm Validation Program**
CBC ..... **Cipher Block Chaining**
CC ..... **Common Criteria**
CSP ..... **Critical Security Parameter**
DDPE ..... **Dell Data Protection | Encryption**
DPAPI ..... **Data Protection Application Program Interface**
DRBG ..... **Deterministic Random Bit Generator**
EAL ..... **Evaluation Assurance Level**
EMS ..... **External Media Shield**
FEK ..... **File Encrypting Key**
FFE ..... **File Folder Encryption**
FIPS ..... **Federal Information Processing Standards**
GCM ..... **Galois/Counter Mode**
HCA ..... **Hardware Crypto Accelerator**
HMAC ..... **Hash-based Message Authentication Code**
I&A ..... **Identification and Authentication**
I/O ..... **Input/Output**
IT ..... **Information Technology**
KEK ..... **Key Encrypting Key**
LMC ..... **Local Management Console**
LMS ..... **Local Management Server**
MB ..... **MegaByte**
NIAP ..... **National Information Assurance Partnership**
NIST ..... **National Institute of Standards and Technology**
NRPE ..... **Non-Recoverable Personal Edition**
OS ..... **Operating System**
PBKDF ..... **Password-Based Key Derivation Function**
PII ..... **Personally Identifiable Information**
PE ..... **Personal Edition**
PP ..... **Protection Profile**
SDE ..... **Software Disk Encryption**
SFR ..... **Security Functional Requirement**
SHA ..... **Secure Hash Algorithm**
SP ..... **Special Publication**
SQL ..... **Structured Query Language**
ST ..... **Security Target**
TOE ..... **Target of Evaluation**
TSF ..... **TOE Security Function**
XTS ..... **XEX-based Tweaked-codebook mode with ciphertext Stealing**

1. Security Target Introduction

This Security Target (ST) describes the objectives, requirements and rationale for the Dell Data Protection | Encryption Personal Edition Non-Recoverable Version 8.14. The language used in this Security Target is consistent with the *Common Criteria for Information Technology Security Evaluation, Version 3.1*. As such, the spelling of terms is presented using the internationally accepted English.

1.1 Security Target Reference

Dell Data Protection | Encryption Personal Edition Non-Recoverable Security Target, Version 1.10, dated September 29, 2017

1.2 TOE Reference

Dell Data Protection | Encryption Personal Edition Version 8.14.0 (build 2) installed in non-recoverable mode

1.3 TOE Overview

1.3.1 Usage and Major Security Features

Dell Data Protection | Encryption (DDPE) is a policy-based solution that protects data stored on a workstation or laptop drives. The DDPE portfolio of products delivers a high level of protection, fills critical security gaps and provides encryption policies for multiple endpoints and operating systems from a single management console.

DDPE Personal Edition (PE) provides software-based, data-centric encryption that protects data without disrupting IT processes or end user productivity. Personal Edition enables:

- Encryption of data by specific file types or folders
- Integration with existing processes for authentication
- Ability to encrypt based on end user profiles, data and groups within your organization

The evaluation is limited to PE installed in non-recoverable mode (NRPE).

When a new user is activated, the default encryption policy for the TOE is copied and becomes the initial policy for the user. Administrators may modify the encryption policies for users.

User login is provided by Windows. Authorized Windows users are, to NRPE, either unmanaged or activated. Initially all Windows users are unmanaged. In this state none of the files they create are encrypted, and they are unable to access any of the files NRPE has encrypted.

NRPE transparently encrypts files for activated users according to the user's encryption policy. The policy may specify files, folders and/or file extension types to encrypt. The policy includes both files encrypted with a common (to all users on an endpoint) key as well as files encrypted with a user-specific key. A common key enables file sharing between activated users, while the user-specific key limits access to just the user that creates a file. For example, a user's home folder is typically encrypted with the user-specific key, while a group project folder is encrypted with a common key.

1.3.2 TOE type

Data Protection

1.3.3 Required Non-TOE Hardware/Software/Firmware

TOE components are installed on endpoints that must satisfy the requirements specified in the following table. Minimum hardware requirements are determined by the Windows operating system variant being used.

Table 1 Endpoint Non-TOE Requirements

Item	Minimum Requirement
Platform	Dell Precision, Latitude or OptiPlex Note that all of these platforms include an Intel or AMD processor that supports the RDRAND CPU instruction
Other Hardware	Network Interface
Disk Space	110 MB free space
Operating System	Windows 8.1 Update 0-1 (64-bit): Enterprise, Pro Windows 10 (64-bit): Enterprise, Pro
Other Software	Microsoft Visual C++ 2012 Redistributable Package x64 (automatically installed with TOE) Microsoft Visual C++ 2015 Redistributable Package x64 (automatically installed with TOE) NuGet System.Data.SQLite 1.0.105 Microsoft .NET Framework v4.0

1.4 TOE Description

NRPE is software that runs on an endpoint, which may be a workstation or a laptop. NRPE provides on-device policy enforcement for access control and user data encryption. Policies are user and endpoint specific.

NRPE encrypts and decrypts endpoint-resident data files according to the configured policy. The policy includes common files (multiple users have access to files via a common endpoint-specific key), or may be user-specific (these files are only accessible to the user that creates them). Encryption and decryption of the data is transparent to the end user.

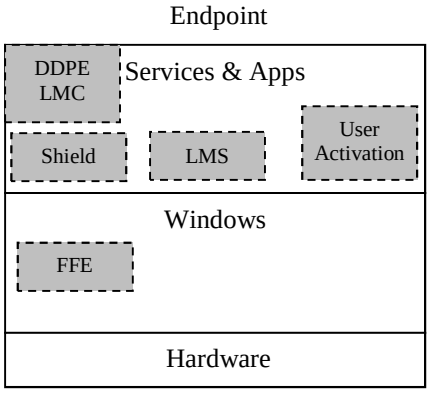
NRPE supports multiple users per endpoint, and a specific encryption policy may be configured for each user.

NRPE supports a variety of platforms. The security functionality provided varies somewhat according to the platform in use. For this evaluation, only NRPE for Windows is included. Therefore, the security functionality described in this document describes the security functionality pertinent to the NRPE for Windows.

1.4.1 Physical Boundary

The following figure illustrates the NRPE components as they are installed on an endpoint. Each component is described in more detail following the figure.

Figure 1 - TOE Physical Boundary



DDPE LMC is the NRPE Local Management Console, a management application that enables a user to monitor TOE status. Authorized administrators may use LMC to configure user policies.

The Shield is a software module that coordinates all interactions between the other NRPE components.

LMS is the Local Management Server, which performs the functions of user activation and policy management. LMS manages the encrypted key material stored in the Vault (a database of key material used by NRPE). Access to the Vault is restricted by a random password generated during installation and stored on the endpoint via Windows DPAPI.

FFE (File Folder Encryption) is a Windows Filter driver that intercepts all file-based I/O. I/O to an unencrypted file is allowed to pass through with the data unchanged. I/O to an unauthorized encrypted file is blocked by the FFE driver. I/O to an authorized encrypted file is encrypted/decrypted by FFE. Authorization is determined by whether or not the requesting user has access to the key used to encrypt the file – either the Common Key or a User-specific Key.

User Activation is a software module that interfaces to the Windows OS login module. The Operational Environment (Windows) is responsible for identification and authentication of the End User. On initial login by each user, the user is "activated" by creating key material and policy for the user. Upon each successful authentication, User Activation determines the NRPE policy and key material to be used for I/O by the user.

The physical boundary includes the following guidance documentation:

1. *Dell Data Protection | Personal Edition Installation Guide v8.14 (2017-05)*
2. *Dell Data Protection | Personal Edition Addendum: Non-Recoverable Mode (2017-07)*
3. *Dell Data Protection | Encryption Personal Edition Non-Recoverable Common Criteria Supplement (2017-09)*

1.4.2 Logical Boundary

1.4.2.1 Cryptographic Support

Cryptographic functionality is included in the TOE. The TOE also uses cryptographic functionality provided by CAPI and DPAPI.

1.4.2.2 User Data Protection

Files of activated users are transparently encrypted using AES-CBC with 256-bit keys, as configured in user-specific encryption policies. Files may be encrypted with common keys in order to be shared, or with user-specific keys in order to limit access to the user that creates the file.

1.4.2.3 Identification and Authentication

User I&A is performed by Windows. Once valid credentials are supplied, Windows Login invokes the TOE, and the TOE determines if the supplied userid is authorized to access encrypted files (activated).

1.4.2.4 Management

Management functionality is available to TOE administrators to configure encryption policies.

1.4.2.5 Privacy

The TOE does not transmit any Personally Identifiable Information (PII) over a network.

1.4.2.6 Protection of the TSF

The TOE is constructed with anti-exploitation capabilities. KEKs and FEKs are stored in the LMS database; these items are protected by 256-bit AES-GCM. The TOE only uses supported platform APIs and is packaged with a defined set of third-party libraries. Trusted updates for the TOE can be downloaded in a form that allows them to be cryptographically verified and installed.

1.4.2.7 Trusted Path

The TOE does not transmit any data between itself and another trusted IT product.

1.5 Evaluated Configuration

The following configuration options must be adhered to:

1. The Windows system on which the TOE is installed must be configured for FIPS mode of operation.
2. Windows Fast User Switching functionality, which allows multiple users to be logged on to Windows simultaneously, is disabled.
3. The procedures specified in *Dell Data Protection | Personal Edition Installation Guide Addendum: Non-Recoverable Mode* are followed during installation. Specifically:
    a. Data Encryption Management Type is set to Personal Edition
    b. The “SECURE=1” parameter is included in the installation command to select non-recoverable mode.
4. The Sleep (S3) and Hibernate (S4) power management states must be disabled in Windows.
5. The encryption policies may be configured for each user subject to the following constraints.

Table 2 Encryption Policy Constraints

Policy Element	Allowed Values
SDE Encryption Enabled	False
HCA	False
Encryption Enabled	True
Common Encrypted Folders	List of folders whose files will be encrypted and will be accessible to all activated users (via Common Keys unique to the endpoint)
Common Encryption	AES 256
Application Data Encryption List	List of processes with specific extensions (e.g. notepad.exe) whose user files will be encrypted and will be accessible to all activated users (via Common Keys unique to the endpoint)
Application Data Encryption Key	Common
Encrypt Outlook Personal Folders	True or False
Encrypt Temp Files	True or False
Encrypt Temp Internet Files	True or False
Encryption User Profile Docs	True or False
Encrypt Windows Paging File	True or False
Secure Post-Encryption Cleanup	Three Pass Overwrite
Prevent Unsecured Hibernation	False
Secure Windows Hibernation File	False
Workstation Scan Priority	Norm
User Encrypted Folders	List of folders whose files will be encrypted and will only be accessible to the user that created each file (via User Keys unique to each user)
User Encryption Algorithm	AES 256
User Data Encryption Key	User
EMS Encrypt External Media	False
Port control System	Disabled
Suppress File Contention Notification	True or False
Allow Encryption Processing Only when Screen is locked	False

2. Conformance Claims

2.1 Common Criteria Conformance

Common Criteria version: Version 3.1 Revision 4, dated September 2012

Common Criteria conformance: Part 2 extended and Part 3 extended

2.2 Security Requirement Package Conformance

The TOE does not claim conformance to any security functional requirement or security assurance requirement packages.

2.3 Protection Profile Conformance

The TOE claims exact conformance to:

- Protection Profile for Application Software, Version: 1.2, 22 April 2016 [ASPP], and
- Application Software Protection Profile Extended Package: File Encryption. Version 1.0, 11/10/2014 [FEEP]

These documents are modified by the following NIAP Technical Decisions that apply to this TOE:

- TD0065 - Revision of FDP_PRT_EXT.1.2 requirement in APP SWFE EP v1.0
- TD0069 - Revision to FCS_COP.1(1) AA in SWFE EP v1.0
- TD0076 - Correction to SWFE Keychain Requirement
- TD0092 - FCS_KYC_EXT.1 - Key Integrity
- TD0107 - FCS_CKM - ANSI X9.31-1998, Section 4.1.for Cryptographic Key Generation
- TD0119 - FCS_STO_EXT.1.1 in PP_APP_v1.2
- TD0121 - FMT_MEC_EXT.1.1 Configuration Options
- TD0123 - GCM Mode Added to FCS_KYC_EXT.1.1, FCS_COP.1.1(1), FPT_KYP_EXT.1.1
- TD0172 - Additional APIs added to FCS_RBG_EXT.1.1
- TD0175 - Revision of FCS_CKM_EXT.4 requirement in APP SW FE EP v1.0
- TD0192 - Update to FCS_STO_EXT.1 Application Note
- TD0204 - Protection of Selected User Data
- TD0218 - Update to FPT_AEX_EXT.1.3 Assurance Activity
- TD0221 - FMT_SMF.1.1 - Assignments moved to Selections

3. Security Problem Definition

The Security Problem Definition is taken directly from [ASPP] and [FEEP], and is reproduced below. Exact conformance is claimed; all of the unconditional and selection-based threats, assumptions and organizational security policies stated in [ASPP] and [FEEP] are included and none have been added. References to optional and objective items not included in this Security Target have been removed from the following text.

3.1 Security Problem Definition from [ASPP]

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.

3.1.1 Threats

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.

3.1.2 Assumptions

A.PLATFORM

The TOE relies upon a trustworthy computing platform 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 within compliance of the applied enterprise security policy.

3.1.3 Organizational Security Policies

There are no Organizational Security Policies for the application.

3.2 Security Problem Definition from [FEEP]

The primary asset that is being protected is the sensitive user data stored on a system. The threat model thus focuses on a host machine that has been compromised by an unauthorized user. This section addresses threats to the TOE only.

3.2.1 Threats

A threat consists of a threat agent, an asset, and an adverse action of that threat agent on that asset. The model in this EP only addresses risks that arise from the host machine being compromised by an unauthorized user.

For this EP, the TOE is not expected to defend against all threats related to malicious software that may reside in user data files. For instance, the TOE is not responsible for detecting malware in the data selected by the user for encryption (that is a responsibility of the host environment). Once the file encryption product is operational in a host system, the threats against the data from potentially malicious software on the host are also not in the threat model of this EP. For example, there are no requirements in this EP addressing a malicious host capturing a password-based authorization factor, nor a malicious process reading the memory of an application program that is operating on a decrypted file.

Note that this EP does not repeat the threats identified in the AS PP, though they all apply given the conformance and hence dependence of this EP on the AS PP. Note also that while the AS PP contains only threats to the ability of the TOE to provide its security functions, this EP focuses on threats to resources in the operational environment. Together the threats of the AS PP and those defined in this EP define the comprehensive set of security threats addressed by a file encryption TOE.

Compromise of Keying MaterialAttacks against the encryption product could take several forms; for example, if there is a weakness in the random number generation mixing algorithm or the data sources used in random number generation are guessable, then the output may be guessable as well. If an attacker can guess the output of the pseudorandom number generator (PRNG) at the time an encryption key is made, then the output may be used to recreate the keying material and decrypt the protected files. As the encryption program runs, it will store a variety of information in memory. Some of this information, such as random bit generation (RBG) inputs, RBG output, copies of the plaintext file, and other keying material, could be very valuable to an attacker who wishes to decrypt an encrypted file. If the encryption product does

not wipe these memory spaces appropriately, an attacker may be able to recreate the encryption key and access encrypted files.

(T.KEYING_MATERIAL_COMPROMISE**)**

Brute Force AttackThe protection of the data involves encrypting said data assuming an attacker may have significant computing resources at their disposal. Several ciphers have already been broken through brute-force attacks because the length of the keys used in those ciphers was too short to provide protection against a concerted computing effort to discover those keys. Because protection of the data may rely on a chaining of keys and encryption mechanisms, there are many opportunities for brute force attacks against each potential key in the chain, such that the weakest link in the chain of factors/keys will determine the overall strength against a brute force attack.

(T.KEYSPACE_EXHAUST)

Plaintext CompromiseUnlike full disk encryption, selectable encryption products also need to protect against data leaks to other applications on the machine. Many file creators and editors store temporary files as the user is working on a file, and restore files if the machine experiences an interrupt while a file is open. Any of these files, if not properly protected or deleted, could leak information about a protected file to an attacker. Other applications might also access volatile or non-volatile memory released by the file encryption product, and the software used to create files prior to encryption may retain information about the file even after it has been encrypted. As the user creates and saves a new document, the plaintext will be stored on the machine's hard drive. An attacker could then search for the plaintext of the sensitive, encrypted information. An attacker may not even have to access the encrypted file for the protected information to be compromised. When the user wishes to encrypt the document, this plaintext file should be replaced with the new encrypted version. For non-mobile devices, it is expected that if the volatile and/or non-volatile memory space where the plaintext file was stored is merely released back to the machine without being first wiped clean of the data that was stored there, then the information the user wishes to protect will still be accessible. While protection of the encryption algorithm itself is vital, memory must also be properly managed by the file encryption product or the TOE platform in order for security to remain intact. For mobile devices, it is assumed that the File Encryption product will not be responsible for providing memory management cleanup and the environment's platform has met the Mobile Device Fundamentals Protection Profile.

Additionally, some encryption products offer to create backup files. These files are meant to be used in the event an encrypted file becomes corrupted and incapable of being decrypted. Each

backup file is a valuable resource to protect information that the user cannot afford to lose; however, it also may provide another route for an attacker to access the encrypted information. If the backup file is insufficiently protected, then the attacker may choose to attempt to break into it, rather than the copy of the encrypted file that the user would typically access.

(T.PLAINTEXT_COMPROMISE)

TSF FailureSecurity mechanisms of the TOE generally build up from a primitive set of mechanisms (e.g., memory management, privileged modes of process execution) to more complex sets of mechanisms. Failure of the primitive mechanisms could lead to a compromise in more complex mechanisms, resulting in a compromise of the TSF.

(T.TSF_FAILURE)

Unauthorized Data AccessThe central functionality of the TOE is the protection of resources under its control through encryption. In a shared resource environment, users on a system may have access to administrative-level tools that are capable of over-riding a system's access control protections. Further, if the system were to be lost or the system's storage device stolen, the attacker could then look directly at the storage device using low-level forensic tools in an attempt to access data for which they are not authorized. However, the need to protect the data in these scenarios should not interfere with the data-owner's (or another user that has been granted access to those data) ability to read or manipulate the data.

(T.UNAUTHORIZED_DATA_ACCESS)

Flawed Authentication Factor VerificationWhen a user enters an authorization factor, the TOE is required to ensure that the authorization factor is valid prior to providing any data to the user; the purpose of verification is to ensure the FEK is correctly derived. If the data is decrypted with an incorrectly derived FEK (the FEK is conditioned from the password/passphrase or is decrypted by the KEK), then unpredictable data will be provided to the user. If verification is not performed in a secure manner, keying material or user data may be exposed or weakened.

(T.UNSAFE_AUTHFACTOR_VERIFICATION)

Data Spoofing (optional)For certain modes of encryption, it is possible for a malicious person to modify ciphertext data to force unintended modification to the underlying plaintext data, without the user being notified. There are various failures that may occur on the part of the TOE, to include: failure to verify the integrity of the data prior to decryption, failure to provide integrity on the sensitive data, failure to use a cryptographic or secure hashing code and failure to differentiate the File Authentication Key (FAK) from the FEK; the FAK is any secret value used as input to a keyed hashing function or as part of an asymmetric authentication process.

(T.PLAINTEXT_DATA_SPOOFING)

Threat	Description of Threat
T.KEYING_MATERIAL_COMPROMISE	An attacker can obtain unencrypted key material (the KEK, the FEK, authorization factors, and random numbers, or any other values from which a key is derived) that the TOE has written to volatile memory, and use these values to gain unauthorized access to sensitive encrypted user data.
T.KEYSPACE_EXHAUST	An unauthorized user may attempt a brute force attack to determine cryptographic keys or authorization factors to gain unauthorized access to user or TSF data.
T.PLAINTEXT_COMPROMISE	An attacker may obtain unauthorized read access to sensitive plaintext material (the input to the file encryption) that the TOE has written to volatile memory as a result of the creation of a temporary file or improper memory clean-up.
T.TSF_FAILURE	Security mechanisms of the TOE may fail, leading to a compromise of the TSF.
T.UNAUTHORIZED_DATA_ACCESS	An unauthorized user that has access to filesystem on which a protected resource resides may gain access to data for which they are not authorized according to the TOE security policy.
T.UNSAFE_AUTHFACTOR_VERIFICATION	An attacker can take advantage of an unsafe method for performing verification of an authorization factor, resulting in exposure of the KEK, FEK, or user data.
T.PLAINTEXT_DATA_SPOOFING	An attacker can take advantage of certain encryption modes to modify the underlying plaintext without user awareness.

3.2.2 **Assumptions**

Assumption	Description of Assumption
A.AUTHORIZED_USER	Authorized users of the host machine are well-trained, not actively working against the protection of the data, and will follow all provided guidance.
A.AUTH_FACTOR	An authorized user will be responsible for ensuring that all externally derived authorization factors have sufficient strength and entropy to reflect the sensitivity of the data being protected. This can apply to password- or passphrase-based, ECC CDH, and RSA authorization factors.
A.EXTERNAL_FEK_PROTECTION	External entities that implement ECC CDH or RSA that are used to encrypt and decrypt a FEK have the following characteristics: • meet National requirements for the cryptographic mechanisms implemented; • require authentication via a pin or other mechanisms prior to allowing access to protected information (the decrypted FEK, or the private key); • implement anti-hammer provisions where appropriate (for example, when a pin is the authentication factor).
A.SHUTDOWN	An authorized user will not leave the machine in a mode where sensitive information persists in non-volatile storage (e.g., power it down or enter a power managed state, such as a “hibernation mode”).
A.STRONG_OE_CRYPTO	All cryptography implemented in the Operational Environment and used by the TOE will meet the requirements listed in Appendix C of this EP. This includes generation of external token authorization factors by a RBG.
A.PLATFORM_STATE	The platform on which the TOE resides is free of malware that could interfere with the correct operation of the product.
A.AUTHORIZED_CONFIGURATION	Access and ability to modify the cryptographic configuration files may be done only by authorized users.
A.KEK_SECURITY	The KEK will be derived from a strong entropy source, attaining equal or greater bit strength to that of the block cipher it is used in.
A.FILE_INTEGRITY	When the file is in transit, it is not modified, otherwise if that possibility exists, the appropriate selections in Appendix B are chosen for Data Authentication.

3.2.3 **Organizational Security Policy**

There are no additional OSPs for the File Encryption product.

4. Security Objectives

The Security Objectives are taken directly from [ASPP] and [FEEP], and is reproduced below. Exact conformance is claimed; all of the objectives stated in [ASPP] and [FEEP] are included and none have been added. References to optional and objective items not included in this Security Target have been removed from the following text.

4.1 Security Objectives from [ASPP]

4.1.1 Security Objectives for the TOE

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, and 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_LIB_EXT.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_TUD_EXT.1.5, FPR_ANO_EXT.1

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

4.1.2 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.

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.

4.1.3 Security Objectives Rationale

This section describes how the assumptions, threats, and organizational security policies map to the security objectives.

Threat, Assumption, or OSP	Security Objectives	Rationale
T.LOCAL_ATTACK	O.QUALITY	The objective O.QUALITY protects against the use of mechanisms that weaken the TOE with regard to attack by other software on the platform.
T.PHYSICAL_ACCESS	O.PROTECTED_STORAGE	The objective O.PROTECTED_STORAGE protects against unauthorized attempts to access physical storage used by the TOE.

Threat, Assumption, or OSP	Security Objectives	Rationale
A.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.

4.2 Security Objectives from [FEEP]

4.2.1 Security Objectives for the TOE

The Security Problem described in Section 2 will be addressed by a combination of cryptographic capabilities. Compliant TOEs will provide security functionality that addresses threats to the TOE and enforces policies that are imposed by law and regulation. The following subsections provide a description of the security objectives required to meet the threats/policies previously discussed. The description of these security objectives are in addition to that described in the AS PP.

Note: in each subsection below particular security objectives are identified (highlighted by O.) and they are matched with the associated security functional requirements (SFRs) that provide the mechanisms to satisfy the objectives.

The Security Objectives are the requirements for the Target of Evaluation (TOE) and for the Operational Environment derived from the threats in Section 2.

Protection of Key Material (O.KEY_MATERIAL_PROTECTION)

The TOE must ensure that plaintext key material used in performing its operations is cleared once it is no longer needed. Key material must be identified; its use and intermediate storage areas must also be identified; and then those storage areas must be cleared in a timely manner and without interruptions. For example, authorization factors are only needed until the KEK is formed; at that point, volatile memory areas containing the authorization factors should be cleared.

[FCS_CKM_EXT.4, FDP_PRT_EXT.1 (optional: FDP_PM_EXT.1)]

Encryption Using a Strong FEK and KEK (O.FEK_SECURITY)

In order to ensure that brute force attacks are infeasible, the TOE must ensure that the cryptographic strength of the keys and authorization factors used to generate and protect the keys is sufficient to withstand attacks in the near-to-mid-term future. Password/passphrase

complexity and conditioning requirements are also levied to help ensure that a brute force attack against these authorization factors (when used) has a similar level of resistance.

[FCS_CKM_EXT.2, FMT_SMF.1, FCS_COP.1(1), FCS_IV_EXT.1, FPT_FEK_EXT.1, (selectable: FCS_CKM.1, FCS_CKM_EXT.1]

Removal of Plaintext Data (O.WIPE_MEMORY)

To address the threat of unencrypted copies of data being left in non-volatile memory or temporary files where it may be accessed by an unauthorized user, the TOE will ensure that plaintext data it creates is securely erased when no longer needed. The TOE's responsibility is to utilize the appropriate TOE platform method for secure erasure, but the TOE is not responsible for verifying that the secure erasure occurred as this will be the responsibility of the TOE platform.

[FDP_PRT_EXT.1 (optional: FDP_PRT_EXT.2)]

Protection of Data (O.AUTHORIZATION, O.PROTECT_DATA)

The TOE will encrypt data to protect the data from unauthorized access. Encrypting the file or set of files will protect the user data even when low-level tools that bypass operating system protections such as discretionary and mandatory access controls are available to an attacker. Users that are authorized to access the data must provide *authorization factors* to the TOE in order for the data to be decrypted and provided to the user.

[FCS_CKM_EXT.1, FDP_PRT_EXT.1, FMT_SMF.1, FCS_COP.1(1)]

Safe Authentication Factor Verification (O.SAFE_AUTHFACTOR_VERIFICATION)

In order to avoid exposing information that would allow an attacker to compromise or weaken any factors in the chain keys generated or protected by the authorization factors, the TOE will verify the valid authorization factor prior to the FEK being used to decrypt the data being protected.

[FIA_AUT_EXT.1]

Data Authentication (O.DATA_AUTHENTICATION)

For certain encryption modes, it is feasible to maliciously modify the ciphertext data to cause unintended modifications to plaintext data, without user notification. The TOE may provide a method for authenticating the sensitive data and using an approved data authentication method.

[FCS_CKM_EXT.4]

Objective	Objective Description

Objective	Objective Description
O.AUTHORIZATION	The TOE must enforce the entry of authorization factor(s) by authorized users to be able to encrypt and decrypt user data.
O.CORRECT_TSF_OPERATION	The TOE will provide the capability to test the TSF to ensure the correct operation of the TSF in its operational environment.
O.PROTECT_DATA	The TOE will decrypt/encrypt all user data that is provided to the file encryption program in order to protect it while it is not being activity accessed by the user.
O.FEK_SECURITY	The TOE will encrypt the FEK using a KEK created from one or more authorization factors so that a threat agent who does not have the authorization factor(s) will be unable to gain access to the user data by obtaining the FEK. The size of the FEK will be large enough to make a brute force attack infeasible.
O.KEY_MATERIAL_PROTECTION	The TOE shall ensure that unencrypted keys or keying material are properly removed from memory after use.
O.MANAGE	The TOE will provide all the functions and facilities necessary to support the authorized administrators in their management of the security of the TOE, and restrict these functions and facilities from unauthorized use.
O. SAFE_AUTHFACTOR_VERIFICATION	The TOE shall perform verification of the authorization factors in such a way that the KEK, FEK, or user data are not inadvertently exposed.
O.WIPE_MEMORY	The TOE shall ensure that non-volatile memory space corresponding to sensitive plaintext material (encryption input) is wiped from the TOE's memory. This includes temporary files that may have been created.
O.DATA_AUTHENTICATION *(optional)*	The TOE shall verify the integrity of the plaintext data using an approved data authentication method.

4.2.2 **Security Objectives for the TOE's Operational Environment**

The Operational Environment of the TOE implements technical and procedural measures to assist the TOE in correctly providing its security functionality (which is defined by the security objectives for the TOE). The security objectives for the Operational Environment consist of a set of statements describing the goals that the Operational Environment should achieve.

This section defines the security objectives that are to be addressed by the IT domain or by non-technical or procedural means. The assumptions identified in Section A.1.2 are incorporated as security objectives for the environment.

Objective	Objective Description
OE.AUTHORIZATION_FACTOR_STRENGTH	An authorized user will be responsible for ensuring that all externally derived authorization factors have sufficient strength and entropy to reflect the sensitivity of the data being protected. This can apply to password- or passphrase-based, ECC CDH, and RSA authorization factors.
OE.POWER_SAVE	The non-mobile operational environment must be configurable so that there exists at least one mechanism that will cause the system to power down after a period of time in the same fashion as the user electing to shutdown the system (A.SHUTDOWN). Any such mechanism (e.g., sleep, hibernate) that does not conform to this requirement must be capable of being disabled. The mobile operational environment must be configurable such that there exists at least one mechanism that will cause the system to lock upon a period of time.
OE.STRONG_ENVIRONMENT_ CRYPTO	The Operating environment will provide a cryptographic function capability that is commensurate with the requirements and capabilities of the TOE.
OE.TRAINED_USERS	Authorized users of the host machine will be trained to follow all provided guidance.

5. Extended Components Definition

The extended components included in this ST have been drawn from [ASPP] and [FEEP], and reflect NIAP Technical Decisions published as of 27 July 2017. [ASPP] and [FEEP] do not include extended SFR definitions; therefore, none are included in this ST. No extended components other than those drawn from [ASPP] and [FEEP] are included in this ST.

The following extended components from [ASPP] and [FEEP] are included in this ST:

- FCS_CKM_EXT.1 Key Encrypting Key (KEK) Support
- FCS_CKM_EXT.2 Cryptographic key generation (FEK)
- FCS_CKM_EXT.4 Cryptographic Key Destruction
- FCS_IV_EXT.1 Initialization Vector Generation
- FCS_KYC_EXT.1 Key Chaining and Key Storage
- FCS_RBG_EXT.1 Random Bit Generation Services
- FCS_STO_EXT.1 Storage of Credentials
- FDP_DAR_EXT.1 Encryption Of Sensitive Application Data
- FDP_DEC_EXT.1 Access to Platform Resources
- FDP_NET_EXT.1 Network Communications
- FDP_PRT_EXT.1 Protection of Selected User Data
- FIA_AUT_EXT.1 User Authorization
- FMT_CFG_EXT.1 Secure by Default Configuration
- FMT_MEC_EXT.1 Supported Configuration Mechanism
- FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information
- FPT_AEX_EXT.1 Anti-Exploitation Capabilities
- FPT_API_EXT.1 Use of Supported Services and APIs
- FPT_FEK_EXT.1 File Encryption Key (FEK) Support
- FPT_KYP_EXT.1 Protection of Key and Key Material (FPT_KYP_EXT)
- FPT_TUD_EXT.1 Integrity for Installation and Update
- FPT_LIB_EXT.1 Use of Third Party Libraries
- FTP_DIT_EXT.1 Protection of Data in Transit

6. Security Requirements

This section contains the functional requirements that are provided by the TOE. These requirements consist of functional components from Part 2 of the CC.

The CC defines operations on security requirements. The font conventions listed below state the conventions used in this ST to identify the operations.

Assignment: indicated in underlined text

Selection: indicated in italics

Assignments within selections: indicated in italics and underlined text

Refinement: indicated with bold text

Iterations of security functional requirements may be included. If so, iterations are specified at the component level and all elements of the component are repeated. Iterations are identified by numbers in parentheses following the component or element.

6.1 TOE Security Functional Requirements

6.1.1 Class FCS

6.1.1.1 FCS_CKM.1(2) Cryptographic Symmetric Key Generation

FCS_CKM.1.1(2)

The application shall generate symmetric cryptographic keys using a Random Bit Generator as specified in FCS_RBG_EXT.1 and specified cryptographic key sizes [*256 bit*].

6.1.1.2 FCS_CKM_EXT.1 Key Encrypting Key (KEK) Support

FCS_CKM_EXT.1.1

The TSF shall support KEK in the following manner based on the selection chosen in FPT_FEK_EXT.1: [*using a Random Bit Generator as specified in FCS_RBG_EXT.1 (from the AS PP) and with entropy corresponding to the security strength of AES key sizes of [256 bit]*].

FCS_CKM_EXT.1.2

All KEKs shall be [*256-bit*] keys corresponding to at least the security strength of the keys encrypted by the KEK.

6.1.1.3 FCS_CKM_EXT.2 Cryptographic key generation (FEK)

FCS_CKM_EXT.2.1

The TSF shall generate FEK cryptographic keys [*using a Random Bit Generator as specified in FCS_RBG_EXT.1 (from the AS PP) and with entropy corresponding to the security strength of AES key sizes of [256 bit]*].

FCS_CKM_EXT.2.2

The TSF shall create a unique FEK for each file (or set of files) using the mechanism on the client as specified in FCS_CKM_EXT.2.1.

FCS_CKM_EXT.2.3

The FEKs must be generated by the TOE.

6.1.1.4 FCS_CKM_EXT.4 Extended: Cryptographic Key Destruction

FCS_CKM_EXT.4.1

The TSF shall destroy cryptographic keys 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*]

].

Application Note: This SFR corresponds to FCS_CKM_EXT.4 as modified by TD0175.

6.1.1.5 FCS_COP.1(1a) Cryptographic operation (Data Encryption)

FCS_COP.1.1(1a)

Refinement: The application shall [*implement AES encryption*] ~~**shall**~~ **to** perform **data encryption and decryption** in accordance with a specified cryptographic algorithm **AES used in** [*CBC (as defined in NIST SP 800-38A)*] mode and cryptographic key sizes [*256-bits*].

Application Note: This SFR iteration corresponds to FCS_COP.1(1) from [FEEP] (as modified by TD0123) and presents AES encryption performed by the TOE.

6.1.1.6 FCS_COP.1(1b) Cryptographic operation (Data Encryption)

FCS_COP.1.1(1b)

Refinement: The application shall [*implement platform-provided AES encryption*] ~~**shall**~~ **to** perform **data encryption and decryption** in accordance with a specified cryptographic algorithm **AES used in** [*GCM (as defined in NIST SP 800-38D)*] mode and cryptographic key sizes [*256-bits*].

Application Note: This SFR iteration corresponds to FCS_COP.1(1) from [FEEP] (as modified by TD0123) and presents AES encryption performed by the platform and invoked by the TOE. AES-GCM is invoked via DPAPI and CAPI.

Application Note: This SFR also describes the key encryption used in key chains per FCS_KYC_EXT.1.

6.1.1.7 FCS_IV_EXT.1 Initialization Vector Generation

FCS_IV_EXT.1.1

The application shall [*generate IVs*] in accordance with **[FEEP]** Appendix **G**: Initialization Vector Requirements for NIST-Approved Cipher Modes.

Refinement Rationale: "[FEEP]" has been added to the SFR to clarify the Appendix reference. [FEEP] incorrectly references Appendix H as providing IV requirements; the actual location in [FEEP] is Appendix G.

6.1.1.8 FCS_KYC_EXT.1 Key Chaining and Key Storage

FCS_KYC_EXT.1.1

The TSF shall maintain a primary key chain of: [

- *KEKs originating from one or more authorization factors(s) to the FEK(s) using the following method(s): [utilization of the platform key storage; implement key encryption as specified in FCS_COP.1(1**b**) in [GCM] mode] while maintaining an effective strength of [*
- *[256 bits] for symmetric keys;]*

commensurate with the strength of the FEK] and [

- *No supplemental key chains*].

Refinement Rationale: "b" has been added to distinguish between SFR iterations for TOE-provided and platform-provided AES.

Application Note: This SFR is modified by TD0123.

6.1.1.9 FCS_RBG_EXT.1 Random Bit Generation Services

FCS_RBG_EXT.1.1

The application shall [*invoke platform provided DRBG functionality*] for its cryptographic operations.

6.1.1.10 FCS_STO_EXT.1 Storage of Credentials

FCS_STO_EXT.1.1

The application shall [*not store any credentials*] to nonvolatile memory.

6.1.2 Class FDP

6.1.2.1 FDP_DAR_EXT.1 Encryption Of Sensitive Application Data

FDP_DAR_EXT.1.1

The application shall [*implement functionality to encrypt sensitive data*] in nonvolatile memory.

6.1.2.2 FDP_DEC_EXT.1 Access to Platform Resources

FDP_DEC_EXT.1.1

The application shall restrict its access to [*no hardware resources*].

FDP_DEC_EXT.1.2

The application shall restrict its access to [*no sensitive information repositories*].

6.1.2.3 FDP_NET_EXT.1 Network Communications

FDP_NET_EXT.1.1

The application shall restrict network communication to [*no network communication*].

6.1.2.4 FDP_PRT_EXT.1 Protection of Selected User Data

FDP_PRT_EXT.1.1

The TSF shall perform encryption and decryption of the user-selected file (or set of files) in accordance with FCS_COP.1(1**a**).

Refinement Rationale: "a" has been added to distinguish between FCS_COP.1(1) iterations.

FDP_PRT_EXT.1.2

The application shall [*implement functionality*] to ensure that all sensitive data created by the TOE when decrypting/encrypting the user-selected file (or set of files) are destroyed in volatile and non-volatile memory when the data is no longer needed.

6.1.3 Class FIA

6.1.3.1 FIA_AUT_EXT.1 User Authorization

FIA_AUT_EXT.1.1

The application shall [*implement platform-provided functionality to provide user authorization*] based on [*password/passphrase authorization factors*].

6.1.4 Class FMT

6.1.4.1 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 it and its data from unauthorized access.

6.1.4.2 FMT_MEC_EXT.1 Supported Configuration Mechanism

FMT_MEC_EXT.1.1

The application shall [*store and protect configuration options as specified in FCS_COP.1(1**a**)*].

Application Note: The instance of FMT_MEC_EXT.1 from the [ASPP] is replaced by a different instance from the [FEEP] per TD0121.

Refinement: "a" is added to the FCS_COP.1(1) reference to clarify the SFR iteration being referenced.

Application Note: Configuration options are stored in a Windows file (the NRPE Vault). Information in the Vault is encrypted (AES-CBC).

6.1.4.3 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1

The TSF shall be capable of performing the following management functions: [*configure cryptographic functionality*].

Application Note: The instance of FMT_SMF.1 from the [ASPP] is replaced by a different instance from the [FEEP] as modified by TD0221.

6.1.5 Class FPR

6.1.5.1 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*].

6.1.6 Class FPT

6.1.6.1 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 [*not allocate any memory region with both write and execute permissions*].

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 compiled with stack-based buffer overflow protection enabled.

6.1.6.2 FPT_API_EXT.1 Use of Supported Services and APIs

FPT_API_EXT.1.1

The application shall only use supported platform APIs.

6.1.6.3 FPT_FEK_EXT.1 File Encryption Key (FEK) Support

FPT_FEK_EXT.1.1

The TSF shall [

- *Store a FEK in Non-Volatile memory conformant with FPT_KYP_EXT.1*].

6.1.6.4 FPT_KYP_EXT.1 Protection of Key and Key Material (FPT_KYP_EXT)

FPT_KYP_EXT.1.1

The TSF shall [*only store keys in non-volatile memory when [encrypted, as specified in FCS_COP.1(1**b**)]]*.

Application Note: This SFR is modified by TD0123.

Refinement Rationale: "b" has been added to distinguish between FCS_COP.1(1) iterations.

6.1.6.5 FPT_TUD_EXT.1 Integrity for Installation and Update

FPT_TUD_EXT.1.1

The application shall [*leverage the platform*] to check for updates and patches to the application software.

FPT_TUD_EXT.1.2

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

FPT_TUD_EXT.1.3

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_TUD_EXT.1.4

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

FPT_TUD_EXT.1.5

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

FPT_TUD_EXT.1.6

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

6.1.6.6 FPT_LIB_EXT.1 Use of Third Party Libraries

FPT_LIB_EXT.1.1

The application shall be packaged with only [

- Log4Net v1.2.13.0
- Spring v1.2.0
- Microsoft Visual C++ 2012 Redistributable Package x64 version 11.0.60610.1
- Microsoft Visual C++ 2015 Redistributable Package x64 version 14.0.24215.1
- Galasoft Model-View-ViewModel (MVVM) List (GalaSoft.MvvmLight.dll) version 3.0.0.31869
- Microsoft Expression Interactions for Windows Presentation Foundation (WPF) version 1.0.1327.0
- Microsoft System.Windows.Interactivity version 1.0.1343.0
- Antlr.org antlr.runtime version 2.7.6.2

- NuGet Common.Logging.dll version 2.3.1.0
- NuGet System.Data.SQLite 1.0.105
- NuGet SQLite.Interop.dll 1.0.105].

6.1.7 Class FTP

6.1.7.1 FTP_DIT_EXT.1 Protection of Data in Transit

FTP_DIT_EXT.1.1

The application shall [*not transmit any data*] between itself and another trusted IT product.

6.2 TOE Security Assurance Requirements

The Security Assurance Requirements are the EAL 1 components as specified in Part 3 of the Common Criteria. Note that these components are refined by the assurance activities stated in [ASPP] and [FEEP], which are included by reference.

Table 3 TOE Assurance Components Summary

Class	Component
ADV	ADV_FSP.1: Basic functional specification
AGD	AGD_OPE.1: Operational user guidance
	AGD_PRE.1: Preparative procedures
ALC	ALC_CMC.1: Labelling of the TOE
	ALC_CMS.1: TOE CM coverage
ATE	ATE_IND.1: Independent testing - conformance
AVA	AVA_VAN.1: Vulnerability survey

7. TOE Summary Specification

7.1 Security Functions

7.1.1 Cryptographic Support

The TOE environment is a Windows platform. Microsoft's Cryptography API (CAPI) is used for random number generation and key encryption using AES-GCM. Microsoft's Data Protection API (DPAPI) is used to protect (encrypt) user-specific KEKs used in the File Encryption Key (FEK) key chains using AES-GCM (CAVP #2832 and #4064). The DPAPI calls used are CryptProtectData and CryptUnprotectData.

The TOE incorporates the Dell-CREDANT Cryptographic Kernel Version 1.8 (Windows Kernel Mode and Windows User Mode), which provides the following algorithms:

Table 4 Cryptographic Algorithm Support

Operations	Algorithm	Size	Standards
Symmetric Encryption & Decryption	AES-CBC (CAVP #4562)	256	FIPS PUB 197 NIST SP 800-38A

The cryptographic support provided by the TOE related to specific SFRs is described in the following table.

Table 5 Cryptographic Support SFR Details

SFR	Description
FCS_CKM.1(2)	256-bit symmetric keys are generated via calls to the DRBG functionality of CAPI.
FCS_CKM_EXT.1	KEKs are 256-bit symmetric keys, which are generated via calls to BCryptGenRandom to obtain 256-bit random values. These functions are invoked with a length parameter of 32 bytes and a pointer to the memory location being used for the key. The TOE assumes a minimum entropy seeding of 256 bits for these random number functions. KEKs are encrypted by User DPAPI.
FCS_CKM_EXT.2	Unique FEKs are used to encrypt sets of files. FEKs are 256-bit symmetric keys, which are generated via calls to the DRBG functionality of CAPI. FEKs may be the shared Common Key or the user-unique User Keys. There is one Common Key for each TOE instance (shared by all users on that instance), and (in the evaluated configuration) one User Key for each activated user for each TOE instance. FEKs are encrypted by KEKs.
FCS_CKM_EXT.4	Plaintext keying material in volatile memory is destroyed by powering off the system. No plaintext keying material or cryptographic security parameters are stored on disk.
FCS_COP.1(1a) and (1b)	The algorithm and key sizes used by NRPE are determined by the values of the Common Encryption and User Encryption Algorithm parameters in the security policies. In the evaluated configuration, AES 256 is required to be specified for these parameters, resulting in all encryption using AES and all keys being 256 bits. In the evaluated configuration, AES-CBC is always used for encryption of user data within files, and AES-GCM is always used for key encryption within key chains.

SFR	Description
FCS_IV_EXT.1	For each unique key, a unique IV is generated via calls to the DRBG functionality of CAPI for each file that is encrypted.
FCS_KYC_EXT.1	Each key chain to a FEK utilizes both User DPAPI and key encryption (AES-GCM). User DPAPI is used upon successful login to decrypt a user-specific key. That key in turn is used to decrypt FEKs associated with that user (including the Common Key). All keys in the chain are AES-GCM encrypted with 256-bit keys. Additional details for the key chains are provided in the associated Key Management Document.
FCS_RBG_EXT.1	The TOE invokes the DRBG functionality of Microsoft CAPI (CAVP #489 and #868). User-space programs and the FFE driver invoke BCryptGenRandom. The DRBG functionality is invoked to generate AES-256 symmetric keys, IVs, and Salts.
FCS_STO_EXT.1	Credentials are not stored by the TOE. Windows login is used to authorize users for file access by incorporating User DPAPI in the key chains.

Additional information on key creation and destruction in provided in the Key Management Document (KMD).

7.1.2 User Data Protection

Authorized Windows users are, from a TOE perspective, either unmanaged or activated. Unmanaged users have no access to any files encrypted by the TOE, and no files they create will be encrypted. Activated users have been authorized to use TOE encryption functionality, have an assigned Common Key and User Key, and have an associated encryption policy.

The TOE selectively and transparently encrypts files according to the configured policy for activated users. Policies are user-specific, and are created from the default policy when each user is activated. Files "selected" for encryption may occur based upon the file name, the folder in which the file is created, or the extension designated for the file.

In addition, the following file types may be encrypted for activated users (per the encryption policy for the user):

- Outlook Personal Folders
- Temp Files
- Temp Internet Files
- User Profile Docs
- Windows Paging File

When based on file name or folder, the files may be encrypted with a user-specific User Key (when matching the User Encryption Folders policy parameter) or with the Common Key (when matching the Common Encrypted Folders or Application Data Encryption List policy configuration parameters). AES-CBC with 256-bit keys is used for all data encryption.

When a user attempts to open an encrypted file, FFE intercepts the operation and determines that the file is encrypted and identifies the Key ID of the FEK used for that file. FFE requests the Shield to retrieve the Key for that Key ID. The Shield interacts with LMS to determine if the

active user has access to the corresponding key, and if so, retrieves it and supplies the key to FFE. All activated users have access to the Common Key, and only the user that created a file has access to the User Key associated with that file. If the key used to encrypt the file is not accessible to the user, then anAccess Denied error is returned to the user.

When a new file is opened, FFE again intercepts the operation and determines if the file is supposed to be encrypted based on the encryption policy for the active user. If so, FFE retrieves the appropriate key from the Shield and begins encrypting the data in the file.

Additional information on key usage is provided in the KMD.

The user data protection functionality provided by the TOE related to specific SFRs is described in the following table.

Table 6 User Data Protection SFR Details

SFR	Description
FDP_DAR_EXT.1	The TOE implements file encryption functionality.
FDP_DEC_EXT.1	The TOE does not access hardware resources or sensitive information repositories of the platform.
FDP_NET_EXT.1	The TOE does not perform any network communication.
FDP_PRT_EXT.1	Files are transparently encrypted according to the configured encryption policy for the active user. For all files selected for encryption, AES-CBC with 256 bit keys is performed. When a pre-existing file is initially encrypted, a temporary copy is made and then re-written with encrypted data. Upon completion, the TOE implements a three-pass overwrite of the (no longer needed) temporary file. In the first pass, the entire file is overwritten with 0xAA, in the second pass with 0x55, and in the third pass with random data obtained from the DRBG functionality of CAPI. FEKs are used to encrypt sets of files. FEKs may be the shared Common Key or the user-unique User Keys. There is one Common Key for each TOE instance (shared by all users on that instance), and (in the evaluated configuration) one User Key for each activated user for each TOE instance.

7.1.3 Identification and Authentication

User I&A is performed by Windows. Once a user presents valid credentials, the Shield is invoked by Windows login. Subsequent behavior depends on the user's state within the TOE.

For unmanaged users, no access is provided to encrypted files.

For activated users, the protected (encrypted) User KEK from the LMS database and unprotected (decrypted) via User DPAPI. Using the User KEK, the user's associated FEKs are accessible, and therefore their associated files are accessible.

The I&A functionality provided by the TOE related to specific SFRs is described in the following table.

Table 7 I&A SFR Details

SFR	Description
FIA_AUT_EXT.1	User I&A is performed by Windows, and the TOE is invoked when valid credentials are supplied.

7.1.4 Management

Application configuration options are:

- Non-recoverable mode (required to be set during installation for the evaluated configuration)
- Administrator password
- Default encryption policy (used as the initial encryption policy for each activated user)
- User encryption policies (one per activated user)

Management functionality is available to TOE administrators to configure encryption policies. Configuration is performed via the LMC.

The management functionality provided by the TOE related to specific SFRs is described in the following table.

Table 8 Management SFR Details

SFR	Description
FMT_CFG_EXT.1	No file encryption is performed until users are activated, which requires an administrator to supply an administrator password. The administrator password must be specified during TOE installation (no default value exists). The TOE is installed with file permissions that limit the ability to modify the TOE and its data to Windows administrators.
FMT_MEC_EXT.1	Installation configuration parameters are stored in the NRPE vault. All configuration information stored in the vault (user encryption policies) is encrypted and access to that information is protected by DPAPI.
FMT_SMF.1	Administrator functionality includes the ability to configure encryption policies.

7.1.5 Privacy

The TOE does not transmit any Personally Identifiable Information (PII) over a network.

The privacy functionality provided by the TOE related to specific SFRs is described in the following table.

Table 9 Privacy SFR Details

SFR	Description
FPR_ANO_EXT.1	The TOE does not transmit any Personally Identifiable Information (PII) over a network.

7.1.6 Protection of the TSF

The TOE is constructed with anti-exploitation capabilities:

- Memory is not mapped to explicit addresses
- Memory regions are not allocated with both write and execute permissions

- The TOE user-space components are protected by Microsoft's Enhanced Mitigation Experience Toolkit (EMET)
- The TOE only writes user-modifiable files to directories that contain executable files when explicitly directed by a user to do so
- The TOE is compiled with stack-based buffer overflow protection enabled
- The TOE only uses supported platform APIs

The supported APIs invoked by the TOE are:

- ACLUI.DLL
- ACTIVEDS.DLL
- ADSLDPC.DLL
- ADVAPI32.DLL
- ADVPACK.DLL
- AUTHZ.DLL
- AVRT.DLL
- BCP47LANGS.DLL
- BCRYPT.DLL
- BCRYPTPRIMITIVES.DLL
- CAPI.DLL
- DCOMP.DLL
- DEVOBJ.DLL
- DEVRTL.DLL
- DPAPI.DLL
- DRVSTORE.DLL
- DUI70.DLL
- DUSER.DLL
- DWRITE.DLL
- FLTLIB.DLL
- FMS.DLL
- GDI32.DLL
- IMM32.DLL
- KERNEL32.DLL
- KERNELBASE.DLL

- LINKINFO.DLL
- MFC42U.DLL
- MSCOREE.DLL
- MSI.DLL
- MSILTCFG.DLL
- MSIMG32.DLL
- MSLS31.DLL
- MSVCRT.DLL
- NCRYPT.DLL
- NSI.DLL
- NTDLL.DLL
- ODBC32.DLL
- OLE32.DLL
- OLEACC.DLL
- OLEAUT32.DLL
- OLEDLG.DLL
- PCACLI.DLL
- PCWUM.DLL
- POWRPROF.DLL
- PROPSYS.DLL
- REGAPI.DLL
- SECUR32.DLL
- SETUPAPI.DLL
- SPFILEQ.DLL
- SPPC.DLL
- USER32.DLL
- USERENV.DLL
- USP10.DLL
- UXTHEME.DLL
- VAULTCLI.DLL
- VERSION.DLL

- VIRTDISK.DLL
- W32TOPL.DLL
- WDI.DLL
- WINDOWSCODECS.DLL
- WINSTA.DLL
- WINTRUST.DLL
- WKSCLI.DLL

These list of third party dynamic libraries packaged with the TOE is:

- Log4Net v1.2.13.0
- Spring v1.2.0
- Microsoft Visual C++ 2012 Redistributable Package x64 version 11.0.60610.1
- Microsoft Visual C++ 2015 Redistributable Package x64 version 14.0.24215.1
- Galasoft Model-View-ViewModel (MVVM) List (GalaSoft.MvvmLight.dll) version 3.0.0.31869
- Microsoft Expression Interactions for Windows Presentation Foundation (WPF) version 1.0.1327.0
- Microsoft System.Windows.Interactivity version 1.0.1343.0
- Antlr.org antlr.runtime version 2.7.6.2
- NuGet Common.Logging.dll version 2.3.1.0
- NuGet System.Data.SQLite 1.0.105
- NuGet SQLite.Interop.dll 1.0.105.

KEKs and FEKs are stored in the LMS database in encrypted (256-bit AES-GCM) form.

The TOE may be uninstalled using the procedures documented in the *Dell Data Protection | Personal Edition Common Criteria Supplement*. Users will no longer have access to the contents of any of the encrypted files after using the procedure.

The TSF protection functionality provided by the TOE related to specific SFRs is described in the following table.

Table 10 Protection of the TSF SFR Details

SFR	Description
FPT_AEX_EXT.1	The required anti-exploitation capabilities are incorporated into the TOE. In Visual Studio, the /GS option is used to enable stack-based buffer overflow protection. In the Visual Studio linker, the /DYNAMICBASE and /NXCOMPAT options are used to enable Address Space Layout Randomization (ASLR).
FPT_API_EXT.1	The TOE only uses supported platform APIs.

SFR	Description
FPT_FEK_EXT.1	DPAPI and 256-bit AES-GCM are used to protect FEKs stored in the LMS database.
FPT_KYP_EXT.1	DPAPI and 256-bit AES-GCM is used to protect KEKs stored in the LMS database.
FPT_LIB_EXT.1	The list of third party libraries packaged with the TOE is supplied above.
FPT_TUD_EXT.1	A web browser may be used to connect to the Dell Support web portal to check for TOE updates. Files are distributed as .MSI or .EXE files, and are signed using the Microsoft Authenticode process. Administrators manually retrieve the installation packages from the web portal. Administrators manually install the packages, and Windows validates the signatures during the installation process. The TOE may be uninstalled; this process will remove any TOE-related files. The TOE does not initiate any download, modification, replacement or update to itself. The current version of the TOE may be queried via the LMC application. The digital signature for the update file is based on a certificate identifying Dell Inc as the authorized source, and that certificate is issued by Digicert as the Certificate Authority.

Additional information about how keys are protected and how metadata is associated with files is provided in the KMD.

7.1.7 Trusted Path

The TOE does not transmit any data between itself and another trusted IT product.

The trusted path functionality provided by the TOE related to specific SFRs is described in the following table.

Table 11 Trusted Path SFR Details

SFR	Description
FTP_DIT_EXT.1	The TOE does not transmit any data between itself and another trusted IT product.

7.1.8 Timely Security Updates

Security flaws are addressed by the same procedures used for all flaws. Flaws may be identified internally, by third-party suppliers of TOE components, or by customers. Public disclosures of vulnerabilities are handled as internally-identified flaws if the vulnerability has not been identified previously.

When identified internally or by third party suppliers, a ticket is opened in Dell Engineering's project tracking tool.

When identified by customers, customers report the flaws by contacting the Dell Technical Support group. Contact information can be found on Dell's web site. If protected communication is desired, users may follow the procedures for reporting potential vulnerabilities to Dell by sending email to vulnerability_research@dell.com and using the specified public key. Technical Support personnel collect information from the customer to initiate a support ticket in the Technical Support incident tracking system. Customers also receive Dell support login

credentials, enabling them to access the Dell Support site via secure connections to directly open a support ticket.

Technical Support determines if an incident may be a flaw in the product, and if so the incident is escalated to the Engineering group. At that point information from the incident tracking tool is used to populate a ticket in Engineering's project tracking tool.

All tickets are assigned a severity from 1 to 4, and security flaws are assigned either level 1 (Critical) or 2 (High). Security flaws are expected to be resolved within 90 days.

During investigation of the flaw, mitigations may be identified. The mitigations are documented in the ticket, tested by Quality Assurance (QA), and approved by Project Management (PM). Once approved, the Technical Support group is notified. Technical Support personnel update the incident with the mitigation procedure and contact the customer. Customers are also automatically emailed a notification when updates are made to their support tickets. Technical Support may also publish a Bulletin so that the information in available to all users via Dell's Knowledge Base repository.

Investigation of a flaw may determine that the root cause is within a component provided by a third party supplier. In this case, Engineering contacts the third party to coordinate resolution. While awaiting a fix, Engineering continues to investigate mitigations.

When a fix has been implemented by Engineering, it is provided to QA for testing. QA performs regression testing based upon the area of code that was modified. When a fix is approved by QA, it is released to PM for approval. Upon PM approval, the fix for a customer-reported flaw is released to Technical Support. Technical Support updates the support ticket, loads the fix onto the Dell portal so that it may be downloaded by customers, and notifies the customer about the fix. Technical Support may also publish a Bulletin so that the information is available to all users via Dell's Knowledge Base repository. If a resolution is provided by release of a new version, information about the fix is also included in the Release Notes.