OPSWAT

Security Target

MetaDefender Core & MetaDefender Kiosk
Evaluation Assurance Level (EAL): EAL4+, augmented with ALC_DVS.2, ALC_FLR.2, AVA_VAN.5

TOE Reference:	MetaDefender Core v5.14.2 & MetaDefender Kiosk v4.7.6
Version	v1.10
Date	2025-12-04
Classification:	PUBLIC

Version history

Version	Date	Author	Description
v1.0	2024-05-27	OPSWAT	The first version of the Security Target.
v1.1	2024-10-01	OPSWAT	Amendments based on the 1st analysis cycle
v1.2	2024-10-11	OPSWAT	Amendments based on the 2nd analysis cycle Kiosk version update from 4.6.8 to 4.7.1
v1.3	2025-03-07	OPSWAT	Amendments based on the 1st analysis cycle of ADV, AGD, ALC MD Core and Kiosk version upgrade
v1.4	2025-06-17	OPSWAT	Amendments based on the 2nd analysis cycle
v1.5	2025-07-08	OPSWAT	Amendments based on the 3rd analysis cycle
v1.6	2025-08-08	OPSWAT	Amendments based on anomalies reported in other classes
v1.7	2025-08-26	OPSWAT	Amendments based on the 4th analysis cycle
v1.8	2025-10-14	OPSWAT	Amendments based on the 5th analysis cycle
v1.9	2025-11-03	OPSWAT	Amendments based on the 6th analysis cycle
v1.10	2025-12-04	OPSWAT	Amendments based on the 7th analysis cycle

Table of Contents

1 Introduction

1.1 ST References

Table 1 - ST Reference

ST Title	MetaDefender Core & MetaDefender Kiosk Security Target
ST Version	v1.10
ST Creation Date	2025-12-04

1.2 TOE Reference

Table 2 - TOE Reference

TOE Name	MetaDefender Advanced Threat Prevention solution
TOE Reference	MetaDefender Core v5.14.2 & MetaDefender Kiosk v4.7.6
TOE Version and Release Date	MetaDefender Core • Version: v5.14.2 • Release date: 2025-05-28 MetaDefender Kiosk • Version: v4.7.6 • Release date: 2025-06-14

1.3 TOE Overview

This Security Target (ST) defines the OPSWAT MetaDefender Core & MetaDefender Kiosk Target of Evaluation (TOE) for the purposes of Common Criteria (CC) evaluation.

OPSWAT MetaDefender Kiosk is a cybersecurity solution for protecting your critical network and assets against removable media threats. MetaDefender Core is a backend component that provides centralized file analysis orchestration capabilities. MetaDefender Core is powered by a suite of cybersecurity technologies such as Multiscanning, Deep CDR, Proactive DLP, Adaptive Sandbox and others to detect, analyze and eliminate malware and zero-day attacks. MetaDefender Kiosk is a front-end component that is used as a media scanning workstation.

1.3.1 TOE Boundary

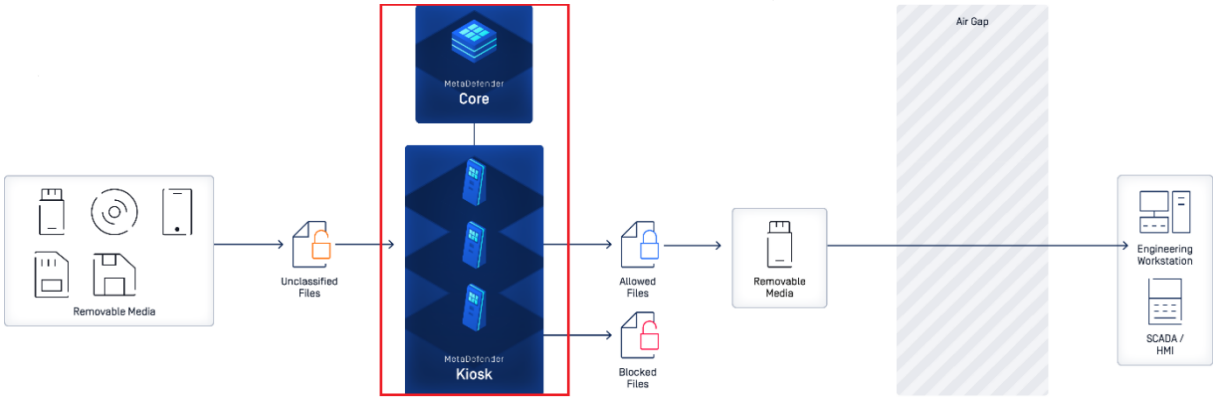
An example deployment of the TOE (enclosed in red) is shown in Figure 1. Note: It is not necessary that the TOE be deployed on an isolated network, nor is it necessary for the environment to include the Binary Armor or Secure File Transfer products.

The figure below illustrates the TOE architecture containing TOE and non-TOE components, that also defines TOE boundary.

TOE in the red box includes two different components (applications): MetaDefender Kiosk and MetaDefender Core (with engines / technologies underneath).

That also defines TOE boundary where MetaDefender Kiosk picks up files from certain sources (removeable devices) and submit to MetaDefender Core for file analysis and further processing.

Figure 1 - TOE Boundary



1.3.2 TOE Type

The TOE is a file-based threat detection and prevention solution.

1.3.3 TOE Usage and Major Security Features

The TOE is typically deployed into secure environments that require all portable media to be scanned on entry and/or exit.

MetaDefender Kiosk

Media such as USB devices, DVDs, card readers, SD cards, flash drives, mobile phones, or floppy disks, are scanned by MetaDefender Kiosk by inserting the media device into the appropriate drive. After the scan is complete, Kiosk generates a detailed report.

Figure 2 - Kiosk

The Kiosk has the following relevant usage characteristics:

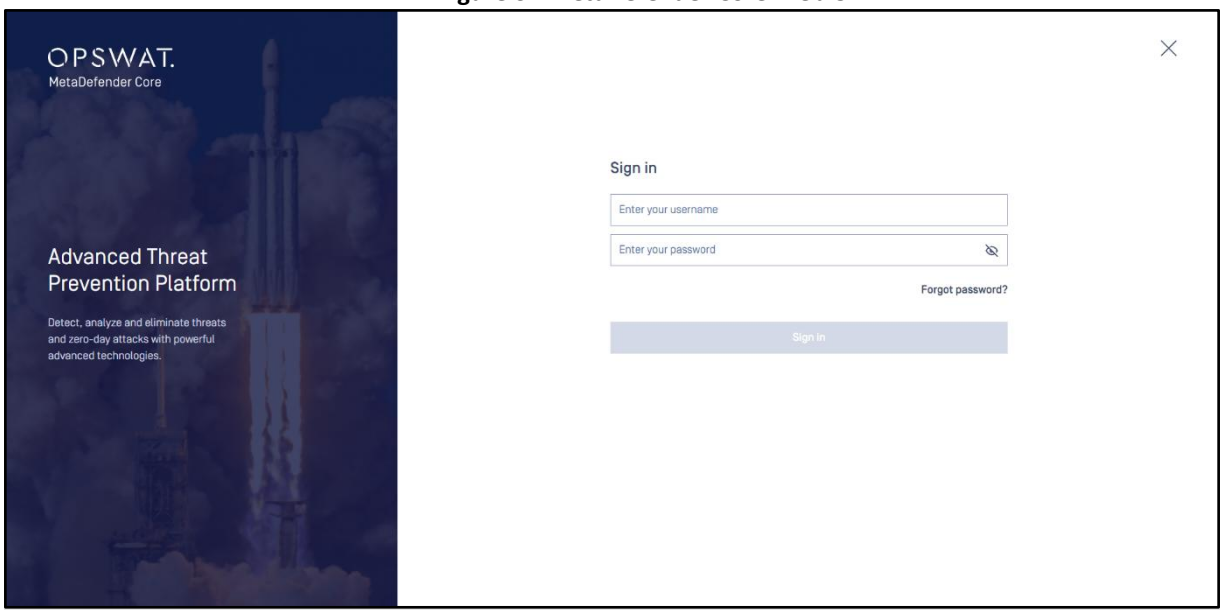
1. **Kiosk Management Console**: The MetaDefender Kiosk Management Console Web UI allows remote management via HTTPS.
2. **User Authentication**: MetaDefender supports scanning user authentication for audit and policy enforcement purposes.

MetaDefender Core

MetaDefender Kiosk uses MetaDefender Core to process files. MetaDefender Core has the following usage characteristics:

1. **REST API**: MetaDefender Core implements a REST API over HTTPS. All file processing (e.g. Kiosk or Web UI) occurs via this JSON-based interface.
2. **Core Management Console**: The MetaDefender Core Management Console Web UI allows remote management via HTTPS. Prior to authentication at the MetaDefender Core server's URL, the public file processing interface will be displayed. This page allows direct upload of files for processing (see Figure 3).
3. **File Processing**: MetaDefender Core has the following file processing capabilities:
    a. Scanning with multiple anti-malware engines
    b. Deep Content Disarm and Reconstruction (CDR) / Data sanitization
    c. File-based vulnerability assessment
    d. Proactive Data Loss Prevention (DLP)

Figure 3 - MetaDefender Core Web UI



1.3.4 Non-TOE Software/Firmware/Hardware

The TOE operates with the following components in the environment:

1. **MetaDefender Kiosk OS**: MetaDefender Kiosk requires a 64-bit Windows OS. The evaluated configuration assumes Windows 10. It provides a reliable time source for MetaDefender Kiosk. Note, the consumer version of Windows 10 reached end of life and not supported. Microsoft provides a Windows 10 IoT Enterprise LTSC 2021 for enterprise customers until which has a mainstream end of life until January 12, 2027, while an extended end of life until January 13, 2032:
    a. https://learn.microsoft.com/en-us/lifecycle/products/windows-10-iot-enterprise-ltsc-2021
2. **MetaDefender Kiosk Hardware**: MetaDefender Kiosk requires hardware that supports the above Windows OS and desired portable media peripherals. Hardware may be user supplied or purchased from OPSWAT. OPSWAT hardware options2 are described at:
    a. https://www.opswat.com/products/metadefender/kiosk/kiosk-options

3. **MetaDefender Core OS**: MetaDefender Core supports Windows and Unix based deployments.
   For Windows: Windows Server 2022
   For Debian: Debian 11, 12
   For Ubuntu: Ubuntu 20.04, 22.04
   It provides a reliable time source for MetaDefender Core.
4. **Network Environment**: Although the TOE can be deployed in a stand-alone nonnetworked environment, the evaluated configuration assumes a network environment that provides connectivity between the Core and Kiosk.
5. **AV SDKs**: AV SDKs listed in Table 3 are third-party anti-virus/anti-malware detection libraries. The OPSWAT Metascan engine module (listed in Table 4) utilizes AV SDKs for malware detection.

Table 3 - AV SDKs

Name	Version for Windows	Version for Linux
Ahnlab	3.27.0.8-2308	3.26.1.4-2253
Antiy	3.0.3.1-1880	
Avira	4.15.23-2168	4.15.23-2163
Bitdefender	3.0.1.306-2122	3.0.1.297-1800
Bkav Pro	8.2.25-422	
ClamAV	1.4.1-2297	1.4.1-2392
CMC	2.3.5-353	2.3.5-150
Comodo	6.5.0.1195-2241	
CrowdStrike	1.10.1-1691	1.10.1-1482
Cylance	1.2.1-484	1.2.0-539
Emsisoft	2021.05.7597-2001	
Eset	1.0.0-2093	1.0.0-1933
Filseclab	1.0.2.2123-1948	
Gridinsoft	1.0.203-281	
Huorong	1.0.0-1800	
Ikarus	6.3.23-2165	6.3.23-2110
K7	4.0.0.6-2240	4.0.0.5-2276
Lionic	8.23-1793	8.23-1600
McAfee	6700-2106	6700-1866
NANOAV	1.0.146.25796-1701	1.0.38.74417-29-18
NETGATE	25.0.650.0-1397-35	
Quick Heal	18.0-2046	18.0-1539
RocketCyber	12_02_2021-1818	22_05_2024-262
Scrutiny	3.2.4-1278	
Sophos	4.20-3.92.0-1996	4.18-3.90.0-1798
Systweak	1.0.0.2-1785	
Tachyon	2020.4.22.1-2115	20240927-1951

Varist	6.6.1-2178	6.6.1-2095
Vir.IT eXplorer	9.5.80-2229	
VirusBlokAda	5.3.1-1932	
Webroot SMD	1.4.114-1770	1.4.114-1767
Microsoft Defender	1.0.0-1329	
XVirus	4.2.3-1717	4.2.3-354
Zillya!	1.2.0.11-1992	

1.3.5 Excluded Functionality

The following security related functionality that is available in MetaDefender Core & MetaDefender Kiosk has not been evaluated:

1. Use with Vault Server
2. Email password recovery
3. Custom scanners
4. Yara rule sources
5. Cloud based scanning by 3rd party malware engines
6. Sending files to MetaDefender Cloud
7. Decryption / unlock of password protected files
8. Kiosk visitor management
9. Single Sign-On
10. Active Directory

1.4 TOE Description

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

1.4.1 Physical Scope of the TOE

The TOE includes the following software:

1. OPSWAT MetaDefender Core v5.14.2 (Windows and Linux version) with the modules developed by OPSWAT
2. OPSWAT MetaDefender Kiosk v4.7.6

Table 4 - Modules

Name	Type	Version for Windows	Version for Linux
Metascan Engine	Module	5.14.2[1]	5.14.2
Proactive DLP	Module	2.22.1-1738248658	2.22.1-3958
Deep CDR	Module	7.3.2-21425	7.3.2-21425
InSights Threat Intelligence	Module	2.1.0-293	2.1.0-293
Sandbox	Module	2.2.0-280	2.2.0-326
SBOM	Module	3.1.0-351	3.1.0-351

[1] It is part of the MetaDefender Platform, so it is versioned together with it.

Reputation	Module	2.1.2-1728564486	2.1.2-1728562722
File Based Vulnerability Assessment	Module	4.2.416.0-154	4.57-236
Country of Origin	Module	1.1.0-292	1.1.0-292
Archive Extraction	Utility	7.3.2-6679	7.3.2-6679
Archive Compression	Utility	7.3.2-6679	7.3.2-6679
File Type	Utility	7.3.1-7712	7.3.1-7712
Yara	Utility	4.2.0-370	4.2.0-370

Note: Not every engine is available for both operating systems. If a version is indicated for an engine in the above table, it means that the corresponding engine is available in that version for the operating system indicated in the column title.

Metascan Engine module integrates the 3rd party engines, which are SDKs from AV manufactures. The Metascan Engine module utilizes the detection capabilities of 3rd party AV engine(SDKs) but has its own policy for threat analysis. These SDKs are listed in Table 3 for Windows and Linux separately.

Within the context of the TOE, an engine is a composable and self-contained processing component that performs cybersecurity analysis on a submitted file under the control of the MetaDefender Core platform.

Each engine executes as an independent process forming part of the TOE Security Functionality (TSF). The engines, including the modules and utilities listed in table 4, collectively implement the TOE's detection and policy enforcement capabilities defined by the SFRs (e.g., FTR_SCN.1 – File Scanning and FTR_SCN.2 – Scan Result Management).

1.4.1.1 Guidance Documents

The TOE includes the following guidance documents:

1. MetaDefender Core - v5.14.2_2025-12-04 [MDCore]
    a. AF39B4CAD8D535993A7AAA83134A80202CDE830E28FDAC387BA7A2387CB781FD
2. MetaDefender Kiosk - v4.7.6_2025-12-04[MDKiosk]
    a. A930037A9B3EAEFAC901ED4630530C2E49FE40248E7818A992BD2DCBB07D5C8D
3. AGD Documentation MetaDefender Core & MetaDefender Kiosk [AGD]
    a. EEDDE2A51E95779986204CBB764878B339D2C3E15952D50A15FC8FF60BAD855C

1.4.1.2 Installation packages

- MetaDefender Core:
    - ometascan-5.14.2-1-x64.msi
    - ometascan_5.14.2-1_amd64.deb
- MetaDefender Kiosk: MetaDefender_Kiosk_4.7.6. 3642.exe

1.4.2 Logical Scope of the TOE

The TOE provides the following security functions:

1. **File Threat Analysis**: The TOE orchestrates the analysis of files for threats and generates associated scanning session reports. Based on scan results, files are handled according to administrator defined policies for 'Blocked Files' and 'Allowed Files'. Scan types include:
    a. **Scanning with multiple anti-malware engines**: File scanning in your environment (no data is shared outside) using over 30 anti-malware engines, including signature-based detection, AI/NGAV and heuristic detection.
    b. **Deep CDR / Data sanitization:** Remove active content from common types of document and image files by either converting the file format or removing hidden exploitable objects such as scripts and macros.
    c. **File-based Vulnerability Assessment**: Ability to identify all known vulnerabilities in binaries (applications, patches, firmware updates) that might be used to exploit and compromise the end-user system once installed/deployed.
    d. **Proactive DLP**: Detect, redact, watermark, or block sensitive data in supported file types. Sensitive data may include credit card numbers, social security numbers or any specific data pattern using a regular expression.
    e. **File Handling:** The Kiosk manages file handling actions for both blocked and allowed files during processing. For blocked files, actions include reporting, stopping the session, removal, sanitization, and copying to specified locations. For allowed files, actions include reporting, wiping and copying back to original media, sanitization, and copying to specified locations.
    f. **Reporting:** After media processing, the session results show completion status, counts of allowed and blocked files, and total files processed. If any file wasn't processed by MetaDefender, a warning is displayed. The results page includes buttons for viewing allowed and blocked files and initiating file transfer and printing. The Blocked File Details screen provides detailed information on blocked files.
2. **Protected Communications**: The TOE makes use of HTTPS/TLS to protect communication with remote administrators and between the Kiosk and Core.
3. **User Authentication Support**: The TOE supports authenticating users as follows:
    a. **Kiosk Scanning User**: Kiosk scanning users are authenticated using Windows Login. Guest users may also perform scans depending on the defined policy.
    b. **Kiosk Management Console User:** Kiosk administrators are authenticated by means of a username and password against a local database.
    c. **Core REST API / Management Console User**: Core users are authenticated by means of a username and password against a local database.
4. **Security Management**: The TOE enables secure management of its security functions, including enforcing role-based access control, generating security audit events and performing trusted software updates, including updates to engines and signatures,

using digital signatures. The TOE generates keys according to the referred standards and also provides key destruction.

2 Conformance Claims

Table 5 - Conformance Claims

Common Criteria Conformance	Common Criteria for Information Technology Security Evaluation, CC Part 2 extended, CC Part 3 conformant
Common Criteria version	Version 3.1 Revision 5
PP Conformance	-
Evaluation Assurance Level	EAL4+, augmented with ALC_DVS.2, ALC_FLR.2, AVA_VAN.5

3 Security Problem Definition

This section defines the security problem to be addressed by the TOE and its operational environment and includes the following:

- Assets
- Secure Usage Assumptions,
- Threats, and
- Organizational Security Policies (OSPs).

3.1 Organizational Security Policies

The TOE shall comply with the following Organizational Security Policies (OSP) as security rules, procedures, practices, or guidelines imposed by an organization upon its operations.

Table 6 - OSPs

OSP	Description
OSP.AUDIT	The TOE shall be capable of auditing the use of scanning and management functions.

3.2 Assets

Table 7 - Assets

Assets	**Description**
A.Data	The data transferred (e.g., files, documents, etc.) from the Kiosk to the Core then back to the user.
A.PersonalData	The personal information handled (read, stored, sent, received) by the Kiosk or the Core.

3.3 Assumptions

Table 8 - Assumptions

Assumption	**Description**
A.ADMIN	Administrators are trusted and follow guidance.
A.USER	Non-administrative users of the TOE are trusted and follow guidance.
A.PHYSICAL	TOE components are protected from unauthorized physical access.
A.TIME	The IT environment will provide a reliable time source.

3.4 Threats

Table 9 - Threats

Threat	Description
T.FILE_THREAT	Attackers use file-based malware on portable media to compromise TOE protected IT resources.
T.DATA_LOSS	Attackers exfiltrate sensitive information, or users inadvertently transfer sensitive information between domains (e.g. high to low), on portable media.
T.MGMT	Attackers compromise the integrity of TOE security policies via TOE management interfaces.
T.COMMS	Attackers compromise the confidentiality or integrity of communication between TOE components or between the TOE and remote users.

4 Security Objectives

Table 10 - Security Objectives for the TOE

Objective	Description
O.DETECT	The TOE shall enable detection of file-based threats and respond according to a defined policy.
O.DLP	The TOE shall detect sensitive data in submitted files and respond according to a defined policy.
O.ACCESS	The TOE shall prevent unauthorized access to management interfaces.
O.MGMT_AUDIT	The TOE shall audit usage of management interfaces.
O.KIOSK_AUDIT	The TOE shall audit the use of kiosk scanning functions.
O.COMMS	The TOE shall protect communication between TOE components and between the TOE and remote users.
O.UPDATE	The TOE shall authenticate software updates.

Table 11 - Security Objectives for the Operational Environment

Objective	Description
OE.ADMIN	Administrators shall be trustworthy and follow guidance.
OE.USER	Non-administrative users of the TOE shall be trustworthy and follow guidance.
OE.PHYSICAL	TOE components shall be protected from unauthorized physical access.
OE.TIME	The IT environment will provide a reliable time source.

4.1 Security Objectives Rationale

This section demonstrates that the stated security objectives counter all identified threats, enforce policies, and uphold assumptions.

The following tables provide a mapping of security objectives for the TOE and security objectives for the operational environment of the TOE to the defined threats, policies, and assumptions, illustrating that each security objective covers at least one threat, enforces a policy or upholds an assumption and that each threat, policy or assumption is covered by at least one security objective.

The tables below provide information regarding:

- the identified security objectives providing effective countermeasures for the threats;
- the identified security objectives providing complete coverage of each organizational security policy;
- the identified security objectives upholding each assumption.

4.1.1 Security Objectives Rationale related to Threats

Table 12 - Threats

Threats	Objectives	Rationale
T.FILE_THREAT	O.DETECT	Mitigates this threat by detected and responding to file-based threats.
T.DATA_LOSS	O.DLP	Mitigates this threat by detecting sensitive data in scanned media and responding according to a defined policy.
T.MGMT	O.ACCESS	Mitigates this threat by preventing unauthorized access to management interfaces.
	O.UPDATE	Mitigates this threat by authenticating software updates (that are received via management interfaces).
T.COMMS	O.COMMS	Mitigates this threat by requiring protected communication between TOE components and between the TOE and remote users.

4.1.2 Security Objectives Rationale relating to Assumptions

Each of the Assumptions in section 3.3 is directly matched by a security objective for the operational environment in section 4.1.2 Table 12. The wording of each objective for the operational environment includes the wording of each assumption, and no further rationale is therefore given here.

Table 13 - Assumptions

Assumptions	Objectives	Rationale
A.ADMIN	OE.ADMIN	Restates the assumption as an environmental objective.
A.USER	OE.USER	Restates the assumption as an environmental objective.
A.PHYSICAL	OE.PHYSICAL	Restates the assumption as an environmental objective.
A.TIME	OE.TIME.	Restates the assumption as an environmental objective.

4.1.3 Security Objectives Rationale relating to OSPs

An Organizational Security Policy (OSP) is a set of security rules, procedures, or guidelines imposed by an organization on the operational environment of the TOE. The table below provides a mapping of the OSPs to the Security Objectives.

Table 14 - OSP

OSP	Objective	Rationale
OSP.AUDIT	O.MGMT_AUDIT	Satisfies this OSP by requiring audit of the usage of management interfaces.
	O.KIOSK_AUDIT	Satisfies this OSP by requiring audit of the usage of Kiosk scanning functions.

5 Extended Components Definition

5.1 Conventions

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

a. Assignment: *Indicated with italicized text*.
b. Refinement: **Indicated with bold text and ~~strikethroughs~~**.
c. Selection: Indicated with underlined text.
d. Assignment within a Selection: *Indicated with italicized and underlined text.*
e. Iteration: Indicated by adding a string starting with "/" (e.g. "FCS_COP.1/Hash").

The following table identifies the extended classes, families and components which are incorporated into this ST, and a rationale for their creation.

Table 15 - Extended components

Title	Rational
Class FPT: Protection of the TSF	This class contains families of functional requirements that relate to the integrity and management of the mechanisms that constitute the TSF and to the integrity of TSF data.
FPT_TUD: Trusted Update	The existing families of the CC do not address trusted firmware and software update.
Class FTR: Advanced Threat Prevention	The existing classes of the CC do not precisely address this class of security functionality, which is not specific to TSF data (FPT), User data (FDP), Security audit (FAU) or any other aspect covered by existing classes.
FTR_SCN: File Threat Scanning	The existing families of the CC do not address file-based threat scanning.
FTR_CDR: Content Disarm and Remove	The existing families of the CC do not address data sanitization / functionality to remove active content from files.
FTR_FVA: File-based Vulnerability Assessment	The existing families of the CC do not address functionality to identify known vulnerabilities in binaries.
FTR_DLP: Data Loss Prevention	The existing families of the CC do not address data loss prevention functions.

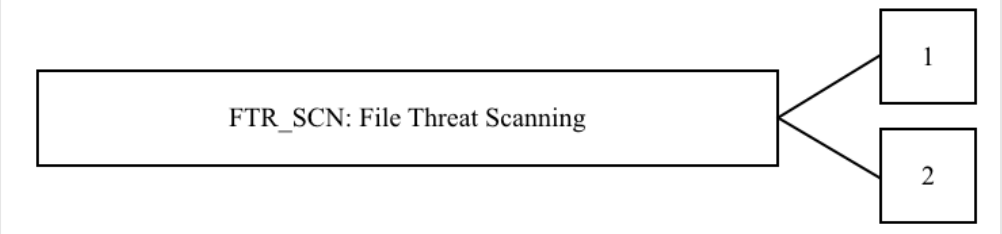
FTR_TAR: Threat Analysis Report	The existing families of the CC do not address production of threat reports.

5.2 File Threat Scanning (FTR_SCN)

Family Behavior

This family provides requirements that address file-based threat scanning.

Component Leveling



FTR_SCN.1 specifies how files can be submitted for scanning.

FTR_SCN.2 specifies the scanning engines that are used for scanning.

Management: FTR_SCN.1

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

a. Management of related policy configuration.

Audit: FTR_SCN.1

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

a. None

FTR_SCN.1	*Submission of Files for Scanning*

Hierarchical to: No other components.

Dependencies: No other components.

FTR_SCN.1.1 The TSF shall support the following methods of file submission for scanning: [assignment: *list of the ways that files may be submitted for scanning*].

Management: FTR_SCN.2

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

a. Management of related policy configuration.

Audit: FTR_SCN.2

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

a. None

FTR_SCN.2	*Supported Scanning Engines*

Hierarchical to: No other components

Dependencies: FTR_SCN.1 Submission of Files for Scanning

FTR_SCN.2.1 The TSF shall [selection: use 3rd party, implement its own] scanning engines that perform the following types of threat scanning: [assignment: *list of supported scanning engines and associated scan types (this may be individual engines/scan types or identification of standards-based engine types that the TOE supports)*].

FTR_SCN.2.2 The scanning engines used by the TSF shall be [selection: local to the TOE, cloud based, [assignment: *other – describe where the scanning engines reside*]].

FTR_SCN.2.3 The TSF shall submit the following information and artifacts with a scan request: [assignment: *list of information and artifacts*].

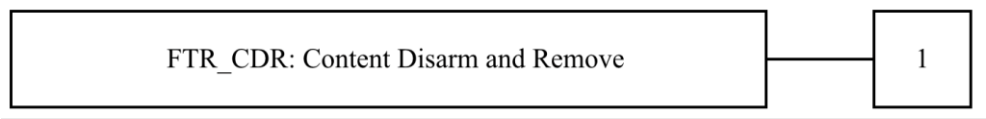
FTR_SCN.2.4 The TSF shall, at a minimum, receive the following information in the scan result: [assignment: *list of information*].

5.3 Content Disarm and Remove (FTR_CDR)

Family Behavior

This family provides requirements that address removing active content from files, such as embedded macros or other objects.

Component Leveling



FTR_CDR.1 specifies requirements for removing active content from files.

Management: FTR_CDR.1

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

a. Management of related policy configuration

Audit: FTR_CDR.1

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

a. None

FTR_CDR.1	*Content Disarm and Remove*

Hierarchical to: No other components.

Dependencies: FTR_SCN.1 Submission of Files for Scanning

FTR_CDR.1.1 The TSF shall support removal of active content from the following types of files: [assignment: *list of supported file types*].

FTR_CDR.1.2 The TSF shall support removal of the following types of active content from files: [assignment: *list of active content*].

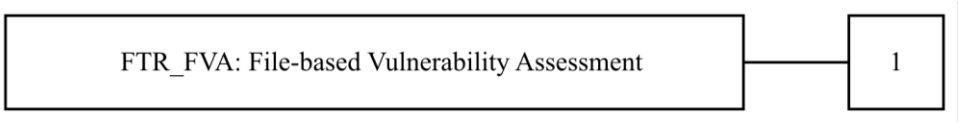
FTR_CDR.1.3 The TSF shall use the following methods to remove active content from files: [assignment: *list and describe methods used to remove active content*].

5.4 File-based Vulnerability Assessment (FTR_FVA)

Family Behavior

This family provides requirements that address identifying publicly known vulnerabilities in files.

Component Leveling



FTR_FVA.1 specifies requirements for identifying publicly known vulnerabilities in files.

Management: FTR_FVA.1

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

a. Management of related policy configuration

Audit: FTR_FVA.1

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

a. None

FTR_FVA.1	*File-based Vulnerability Assessment*

Hierarchical to: No other components.

Dependencies: FTR_SCN.1 Submission of Files for Scanning

FTR_FVA.1.1 The TSF shall enforce vulnerability assessment of the following type files: [assignment: *supported file types*].

FTR_FVA.1.2 The TSF shall use the following reference sources for public vulnerabilities: [assignment: *vulnerability databases used*].

FTR_FVA.1.3 The TSF shall identify files of the supported file type that contain public vulnerabilities from the reference sources.

FTR_FVA.1.4 The TSF shall support the following actions when vulnerabilities are detected: [assignment: *actions*].

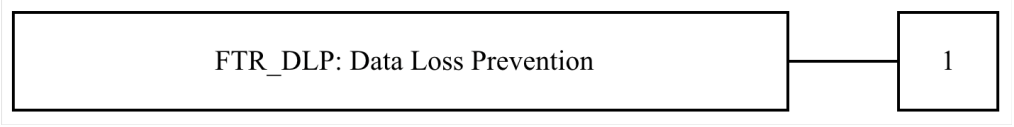
5.5 Data Loss Prevention (FTR_DLP)

Family Behavior

This family provides requirements that address techniques to prevent the loss of sensitive data such as credit card numbers, social security numbers or any specific data pattern. Techniques include:

a. Detecting sensitive data
b. Redacting sensitive data
c. Watermarking documents containing sensitive data
d. Enforcing defined policies for files containing sensitive data

Component Leveling



FTR_DLP.1 specifies requirements to prevent the loss of sensitive data.

Management: FTR_DLP.1

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

a. Management of related policy configuration

Audit: FTR_DLP.1

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

a. None

FTR_DLP.1	*File-based Data Loss Prevention*

Hierarchical to: No other components.

Dependencies: FTR_SCN.1 Submission of Files for Scanning

FTR_DLP.1.1 The TSF shall support applying data loss prevention techniques to the following types of files: [assignment: *supported file types*].

FTR_DLP.1.2 The TSF shall be able to identify the following types of sensitive data in files: [assignment: *types of sensitive data*].

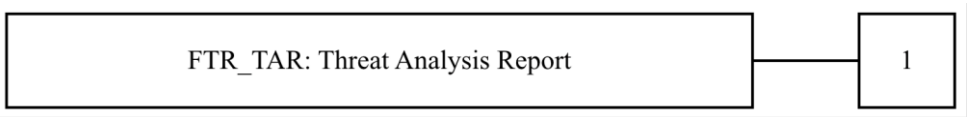
FTR_DLP.1.3 The TSF shall support the following actions when sensitive data is detected: [assignment: *actions*].

5.6 Threat Analysis Report (FTR_TAR)

Family Behavior

This family provides requirements that address production of threat reports.

Component Leveling



FTR_TAR.1 specifies requirements for production of threat reports.

Management: FTR_TAR.1

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

a. Management of related policies

Audit: FTR_TAR.1

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

a. None

FTR_TAR.1	*Threat Analysis Report*
Hierarchical to:	No other components.
Dependencies:	FTR_SCN.2 Supported Scanning Engines, or FTR_CDR.1 Content Disarm and Remove, or FTR_FVA.1 File-based Vulnerability Assessment, or FTR_DLP.1 File-based Data Loss Prevention FMT_SMR.1 Security Roles

FTR_TAR.1.1 The TSF shall support generation of the following reports: [assignment: *list of reports*].

FTR_TAR.1.2 The TSF reports shall contain the following information: [assignment: *for each report, list the information contained in the report*].

FTR_TAR.1.3 The TSF shall allow the following roles to view the reports: [assignment: *roles*].

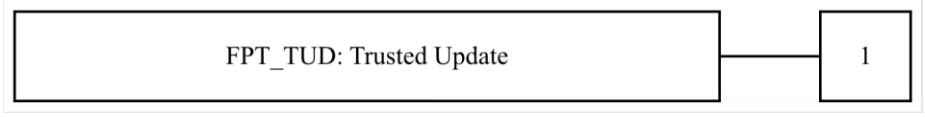
FTR_TAR.1.4 The TSF shall [selection: not store reports, store reports as follows: [assignment: *describe how reports are stored and the rules for retaining reports*]].

5.7 Trusted Update (FPT_TUD)

Family Behavior

This family provides requirements that address trusted updates to the TSF.

Component Leveling



FPT_TUD.1 specifies requirements to update the TOE firmware and software, including the ability to verify the updates prior to installation.

Management: FPT_TUD.1

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

a. None

Audit: FPT_TUD.1

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

a. None

FPT_TUD.1	*Trusted Update*

Hierarchical to: No other components.

Dependencies: FCS_COP.1 Cryptographic Operation

FPT_TUD.1.1 The TSF shall provide [assignment: *Administrators*] the ability to query the currently executing version of the TOE firmware/software and [selection: the most recently installed version of the TOE firmware/software; no other TOE firmware/software version].

FPT_TUD.1.2 The TSF shall provide [assignment: *Administrators*] the ability to manually initiate updates to TOE firmware/software and [selection: support automatic checking for updates, support automatic updates, no other update mechanism].

FPT_TUD.1.3 The TSF shall provide means to authenticate firmware/software updates to the TOE using a [selection: digital signature mechanism, published hash] prior to installing those updates.

6 Security Requirements

This section defines the SFRs, and SARs met by the TOE.

6.1 Conventions

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

a. Assignment: *Indicated with italicized text*.
b. Refinement: **Indicated with bold text and ~~strikethroughs~~**.
c. Selection: Indicated with underlined text.
d. Assignment within a Selection: *Indicated with italicized and underlined text.*
e. Iteration: Indicated by adding a string starting with "/" (e.g. "FCS_COP.1/Hash").

6.2 TOE Security Functional Requirements

List of the SFRs along with their description and the operations performed on them.

Table 16 - SFRs

Name	Description	S	A	R	I
FTR_SCN.1	Submission of Files for Scanning		X		

FTR_SCN.2	Supported Scanning Engines	X	X		
FTR_CDR.1	Content Disarm and Remove		X		
FTR_FVA.1	File-based Vulnerability Assessment		X		
FTR_DLP.1	File-based Data Loss Prevention		X		
FTR_TAR.1	Threat Analysis Report	X	X		
FAU_GEN.1	Audit Data Generation	X	X		
FAU_GEN.2	User Identity Association				
FCS_CKM.1	Cryptographic key generation		X		
FCS_CKM.4	Cryptographic key destruction		X		
FCS_COP.1	Cryptographic Operation		X		
FIA_UAU.1	Timing of authentication		X		
FIA_UAU.7	Protected authentication feedback		X		
FIA_UID.1	Timing of identification		X		
FMT_MOF.1	Management of security functions behavior	X	X		
FMT_SMF.1	Specification of Management Functions		X		
FMT_SMR.1	Security roles		X		
FPT_ITT.1	Basic internal TSF data transfer protection	X		X	
FPT_TUD.1	Trusted Update	X	X		
FTP_TRP.1	Trusted path	X	X	X	

Note: S = Selection, A = Assignment, R = Refinement, I = Iteration

6.2.1 Advanced Threat Prevention (FTR)

FTR_SCN.1 *Submission of Files for Scanning*

Hierarchical to: No other components.

Dependencies: No other components.

FTR_SCN.1.1 The TSF shall support the following methods of file submission for scanning:

- *user presentation of portable media at the Kiosk, and*
- *upload of files at the Core Management Console*[2]*.*
- *call REST API of MetaDefender Core to upload file content along with filename.*

FTR_SCN.2 *Supported Scanning Engines*

Hierarchical to: No other components

Dependencies: FTR_SCN.1 Submission of Files for Scanning

FTR_SCN.2.1 The TSF shall use 3rd party[3] scanning engines that perform the following types of threat scanning: *anti-malware scanning engines and scan types listed in the table below. Supported scan types:*

- *Signature: Signature-based malware scanning,*
- *Heuristics: Heuristics-based malware scanning,*

[2] [assignment: *list of the ways that files may be submitted for scanning*]

[3] [selection: use 3rd party, implement its own]

- *PUA/PUP: Scan for Potential Unwanted Applications (PUA) / Potentially Unwanted Program (PUP),*
- *AI/ML: Artificial Intelligence (AI) / Machine Learning (ML) supported malware scanning.*[4]

Table 17 - Supported scan types of Windows AV engines

	Scan Types (X = enabled by default, O = supported)			
Engine	Signature	Heuristics	PUA/PUP	AI/ML
Lionic	X			
AhnLab	X		O	
Antiy	X	O		
Avira	X	X	X	O
BKAV	X			
Bitdefender	X	X		
ClamAV	X	X	O	
CMC	X			
Comodo	X			
CrowdStrike				X
Cylance				X
Varist	X		O	
Emsisoft	X	O		
Eset	X	X	X	
Filseclab	X			
Gridinsoft	X	X		
Huorong	X			
Ikarus	X			
K7	X	O		
McAfee	X	X	O	
Microsoft Defender	X	X		X
NANO	X	X		
Netgate	X			
Tachyon	X	X		
Quick Heal	X	O	O	
RocketCyber				X
Scrutiny				X
Sophos	X	O	O	
Systweak	X			
VirIT	X		X	
Virus Blokada	X			
Webroot SMD	X			X
Xvirus	X			
Zillya	X	O		

[4] [assignment: *list of supported scanning engines and associated scan types (this may be individual engines/scan types or identification of standards-based engine types that the TOE supports)*]

Table 18 - Supported scan types of Linux AV engines

	Scan Types (X = enabled by default, O = supported)			
Engine	Signature	Heuristics	PUA/PUP	AI/ML
Lionic	X			
AhnLab	X		O	
Avira	X	X	X	O
Bitdefender	X	X		
ClamAV	X	X	O	
CMC	X			
CrowdStrike				X
Cylance				X
Varist	X		O	
Eset	X	X	X	
Ikarus	X			
K7	X	O		
McAfee	X	X	O	
NANO	X	X		
Tachyon	X	X		
Quick Heal	X	O	O	
RocketCyber				X
Sophos	X	O	O	
Webroot SMD	X			X
Xvirus	X			

FTR_SCN.2.2 The scanning engines used by the TSF shall be local to the TOE[5].

FTR_SCN.2.3 The TSF shall submit the following information and artifacts with a scan request: *file and filename*[6].

FTR_SCN.2.4 The TSF shall, at a minimum, receive the following information in the scan result:

- *Core Scan results:*
  - i. *Final verdict (Blocked or Allowed)*
  - ii. *Threat name (if any)*
  - iii. *Scan result (malicious/suspicious/potentially unwanted application/no threat found/etc.)*
  - iv. *All the engines participate in scanning.*
  - v. *Analysis Time in milliseconds.*
- *Kiosk Scan Results*

[5] [selection: local to the TOE, cloud based, [assignment: *other – describe where the scanning engines reside*]]
[6] [assignment: *list of information and artifacts*]

i. *User ID*

ii. *Profile (i.e. scan profile)*

iii. *Session ID*

iv. *Processing Time*

v. *Device Information (i.e. media scanned)*

vi. *File Processing Details*

vii. *Blocked Actions Taken*

viii. *Allows Actions Taken*

ix. *Detailed Scan Results* [7].

FTR_CDR.1	*Content Disarm and Remove*

Hierarchical to: No other components.

Dependencies: FTR_SCN.1 Submission of Files for Scanning

FTR_CDR.1.1 The TSF shall support removal of active content from the following types of files: *file types listed in the table below*[8].

FTR_CDR.1.2 The TSF shall support removal of the following types of active content from files: *content types listed in the table below*[9].

Table 19 - List of active content

File Type	Content Type
doc docm docx dot dotm dotx xls xlsm xlsx xlsb xlt xltx xltm ppt pptm pptx ppsx pps pot potx	Macro OLE Objects: * Attachment * Embedded binary file * Crafted embedded multimedia * Script enabled ActiveX Controls Hyperlink Crafted images Embedded font (not supported for vsdx, vsdm) Hidden text Comment Revision Metadata External media objects Mouse-Over Hyperlink/Click Hyperlink (pptx only) External image (docx only) Chart (not supported for xlsx)

[7] [assignment: *list of information*]
[8] [assignment: *list of supported file types*]
[9] [assignment: *list of active content*]

potm vsdx vsdm vsdx vssx vstx vsdm vssm vstm	
rtf	Embedded object Suspicious Drawing object Embedded HTML Metadata
csv	Formula injection
htm/html	Images Embedded Objects Embedded Java applets Href Metadata
pdf	Hyperlink Actions/JavaScript Annotation Attachments Multimedia Objects Images Embedded font Form fields (edit form, check box..) DTD Metadata
odt	Macro Embedded Object Hyperlink Images Embedded Font Metadata External image
jtd	Macro
jtdc	Hyperlink Embedded Objects Images Font Document View Styles
hwp	Embedded Objects * Flash Files * RTF * PCT Images Macro Hyperlinks
show	Embedded Objects Images

	Hyperlinks
cell	Embedded Objects Images Macro Hyperlinks
ics	Attachment Hyperlinks
xml	XML bomb / oversized payload Recursive payload Cdata injection XML injection VB Macro Script
mp3	Metadata EOF Frame ID3 tag
wav	Metadata
svg	JavaScript Cdata injection XML bomb
dwg jpg bmp png tiff	Macro Abnormal content Embedded malicious code: * HTML * PHP * JavaScript * exploit code Metadata WMF/EMF only: * Nonstandard EOF record * exploit codes

FTR_CDR.1.3 The TSF shall use the following methods to remove active content from files:

- *rebuild the file without the active content, or*
- *convert the file to a different file type*[10]*.*

FTR_FVA.1 *File-based Vulnerability Assessment*

Hierarchical to: No other components.

Dependencies: FTR_SCN.1 Submission of Files for Scanning

FTR_FVA.1.1 The TSF shall support vulnerability assessment of the following type files: *exe, dll, sys, msi, cab, dmg, zip, tar, gz, bz2, bin*[11].

[10] [assignment: *list and describe methods used to remove active content*]
[11] [assignment: *supported file types*]

FTR_FVA.1.2 The TSF shall use the following reference sources for public vulnerabilities: *National Vulnerability Database*[12], *Microsoft Security Update Guide*[13] [14].

FTR_FVA.1.3 The TSF shall identify files of the supported file type that contain public vulnerabilities from the reference sources.

FTR_FVA.1.4 The TSF shall support the following actions when vulnerabilities are detected: *report, block, allow, quarantine file on the local KIOSK, quarantine file on MD Core, delete file, copy/move file to another location, run a custom script applied to that file* [15].

FTR_DLP.1	*File-based Data Loss Prevention*

Hierarchical to: No other components.

Dependencies: FTR_SCN.1 Submission of Files for Scanning

FTR_DLP.1.1 The TSF shall support applying data loss prevention techniques to the following types of files: *Microsoft Offices (DOC, DOCX, XLS, XLSX, PPT, PPTX), OpenOffice (ODT, ODS, ODP), Adobe PDF, Text base (TXT, HTML, CSV, XML), Emails (EML, MSG), Images*[16].

FTR_DLP.1.2 The TSF shall be able to identify the following types of sensitive data in files:

- *credit card numbers*
- *social insurance numbers*
- *Ipv4 address*
- *CIDR range*
- *regular expression data pattern.*[17].

FTR_DLP.1.3 The TSF shall support the following actions when sensitive data is detected:

- *block file*
- *attempt to delete the file from the original media*
- *report the type of sensitive data that was identified.*[18]

FTR_TAR.1	*Threat Analysis Report*

Hierarchical to: No other components.

[12] https://nvd.nist.gov/vuln/data-feeds
[13] https://msrc.microsoft.com/update-guide
[14] [assignment: *vulnerability databases used*]
[15] [assignment: *actions*]
[16] [assignment: *supported file types*]
[17] [assignment: *types of sensitive data*]
[18] [assignment: *actions*]

Dependencies: FTR_SCN.2 Supported Scanning Engines, or

FTR_CDR.1 Content Disarm and Remove, or
FTR_FVA.1 File-based Vulnerability Assessment, or
FTR_DLP.1 File-based Data Loss Prevention
FMT_SMR.1 Security Roles

FTR_TAR.1.1 The TSF shall support generation of the following reports: *Kiosk Scan Results*[19].

FTR_TAR.1.2 The TSF reports shall contain the following information:

- *Kiosk Scan Results*
  - *User ID*
  - *Profile (i.e. scan profile)*
  - *Session ID*
  - *Processing Time*
  - *Device Information (i.e. media scanned)*
  - *File Processing Details*
  - *Blocked Actions Taken*
  - *Allows Actions Taken*
  - *Detailed Scan Results*

.[20]

FTR_TAR.1.3 The TSF shall allow the following roles to view the reports:

- *Roles assigned with permissions: Overall Results Only, Per Engine Results, Full Details*[21].

Application Note: Roles with Overall Results Only and Per Engine Results permissions may only view a subset of report information.

FTR_TAR.1.4 The TSF shall store reports as follows: *scan results are stored according to an administrator defined retention policy and reports are dynamically generated from stored scan results*[22].

6.2.2 Security Audit (FAU)

FAU_GEN.1	*Audit Data Generation*

Hierarchical to: No other components.

[19] [assignment: *list of reports*]
[20] [assignment: *for each report, list the information contained in the report*]
[21] [assignment: *roles*]
[22] [selection: not store reports, store reports as follows: [assignment: *describe how reports are stored and the rules for retaining reports*]]

Dependencies: FPT_STM.1 Reliable time stamps

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

a) Start-up and shutdown of the audit functions;

b) All auditable events for the not specified[23] level of audit; and

c) *Auditable events listed in the table below*[24].

Table 20 - Auditable events

Event	Additional Details
Log in to the Kiosk Management Console	-
User log in to the Kiosk application	-
Kiosk media insert \ removal	-
Core Scanning Request	Client IP address
Log in to the Core Management Console	-
User Management Operations	Operation Details

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

a) Date and time of the event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, *additional details specified in the above table*[25].

FAU_GEN.2 *User Identity Association*

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation
FIA_UID.1 Timing of identification

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

6.2.3 Cryptographic Support (FCS)

FCS_CKM.1 *Cryptographic key generation*

Hierarchical to: No other components.

[23] [selection, choose one of: minimum, basic, detailed, not specified]
[24] [assignment: *other specifically defined auditable events*]
[25] [assignment: *other audit relevant information*]

Dependencies: [FDP_CKM.2 Cryptographic key distribution, or
FCS_COP.1 Cryptographic operation]
FCS_CKM.4 Cryptographic key destruction

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm s*hown in the table below*[26] and specified cryptographic key sizes s*hown in the table below* [27] that meet the following: s*hown in the table below* [28].

Table 21 - Key generation and destruction

Algorithm	Key Size	Function (Cryptographic Operation)	Key generation method	Standard
AES	256 128	Authenticated Encryption, Authenticated Decryption Symmetric encryption and decryption, Password Storage	OpenSSL	[FIPS 197], [SP 800-38A]
HMAC-SHA	256, 128 Salt length: 16, 32	Message authentication Signature generation and verification	OpenSSL	[FIPS 198-1], PKCS#5
SHA-256	Salt length: 32	Password hashing with salt	OpenSSL	[FIPS 180-4], [RFC4122]
RSA	2048, 4096	Asymmetric key pair generation and verification Signature Verification	OpenSSL	[FIPS 186-5], ANSI X9.31-1998, and PKCS #1 v2.1 (PSS and PKCS1.5)
PBKDF2	128 Salt length: 16	Authenticated Encryption	OpenSSL	[FIPS 198-1], PBKDF2

FCS_CKM.4 *Cryptographic key destruction*

Hierarchical to: No other components.

Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or
FCS_CKM.1 Cryptographic key generation]

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method *zeroization*[29] that meets the following: *[FIPS 140-2]* [30].

FCS_COP.1 *Cryptographic Operation*

Hierarchical to: No other components.

[26] [assignment: *cryptographic key generation algorithm*]
[27] [assignment: *cryptographic key sizes*]
[28] [assignment: *list of standards*]
[29] [assignment: *cryptographic key destruction method*]
[30] [assignment: *list of standards*]

Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or
FCS_CKM.1 Cryptographic key generation]
FCS_CKM.4 Cryptographic key destruction

FCS_COP.1.1 The TSF shall perform *cryptographic operations shown in the table below*[31] in accordance with a specified cryptographic algorithm *shown in the table below*[32] and cryptographic key sizes *shown in the table below*[33] that meet the following: *standards shown in the table below*[34].

Table 22 - Cryptographic operations

Operation	Algorithm	Key Size	Standards	CAVP
Symmetric encryption and decryption	AES-GCM	128 256	ISO 18033-3 [18033-3], ISO 19772 [ISO-19772]	C1903 C1904
Key exchange	ECDHE	128	NIST SP 800- 56A [SP800-56A]	
Message digest	SHA2-256 SHA2-384	N/A	ISO/IEC 10118-3:2004 [10118-3]	
Message authentication	HMAC-SHA2-256 HMAC-SHA2-384	256 384	ISO/IEC 9797-2:2011 [9797-2]	
Signature Generation and Verification	RSA	2048 3072	FIPS 186-5 [FIPS-186-5]	

6.2.4 Identification and Authentication (FIA)

FIA_UAU.1 *Timing of authentication*

Hierarchical to: No other components.

Dependencies: FIA_UID.1 Timing of identification

FIA_UAU.1.1 The TSF shall allow *Kiosk: presentation of files for scanning and view session reports (if guest submission is configured), Core: File Submission for Scanning*[35] on behalf of the user to be performed before the user is authenticated.

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

FIA_UAU.7 *Protected authentication feedback*

[31] [assignment: *list of cryptographic operations*]
[32] [assignment: *cryptographic algorithm*]
[33] [assignment: *cryptographic key sizes*]
[34] [assignment: *list of standards*]
[35] [assignment: *list of TSF mediated actions*]

Hierarchical to: No other components.

Dependencies: FIA_UAU.1 Timing of authentication

FIA_UAU.7.1 The TSF shall provide only *bullets*[36] to the user while the authentication is in progress.

FIA_UID.1	*Timing of identification*

Hierarchical to: No other components.

Dependencies: No dependencies.

FIA_UID.1.1 The TSF shall allow *actions per FIA_UAU.1.1*[37] on behalf of the user to be performed before the user is identified.

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

6.2.5 Security Management (FMT)

FMT_MOF.1	*Management of security functions behaviour*

Hierarchical to: No other components.

Dependencies: FMT_SMR.1 Security roles
FMT_SMF.1 Specification of Management Functions

FMT_MOF.1.1 The TSF shall restrict the ability to determine the behavior of, disable, enable, modify the behavior of[38] the functions *shown in the below table*[39] to *roles in the below table*[40].

Table 23 - Roles and behaviors

Role	Functions	Permissions
Kiosk		
Administrator	All	All
Auditor	Session Logs	Determine the behavior of
User/Guest	Application UI	Determine the behavior of
	File discovery and transfer	Determine the behavior of
Core		
Administrators	All	All
Security administrators	All except User Management, Licensing, Config History, Dashboard\Executive Report, Settings\Security, Settings\Network, Settings\Data Retention,	All

[36] [assignment: *list of feedback*]
[37] [assignment: *list of TSF-mediated actions*]
[38] [selection: determine the behaviour of, disable, enable, modify the behaviour of]
[39] [assignment: *list of functions*]
[40] [assignment: *the authorised identified roles*]

	Settings\Export/Import, Settings\Central Management.	
Security auditor	All except Dashboard\Executive Report, Inventory\Password Storage, Settings\Email Notification, Settings\Network, Settings\Export/Import, Settings\Central Management.	Determine the behavior of
Help desk	All except Dashboard\ Executive Report and System Health, User Management, Licensing, Inventory\Password Storage, Settings\Email Notification, Settings\Security, Settings\Network, Settings\Module Update, Settings\Data Retention, Settings\Health Check, Settings\Export/Import, Settings\Central Management, History\Config history, History\Quarantine, Inventory\Certificates, Inventory/Webhook authentication.	Determine the behavior of
Anonymous	File processing	

FMT_SMF.1 *Specification of Management Functions*

Hierarchical to: No other components.

Dependencies: No dependencies.

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

Kiosk Management Console:

- *View Dashboard*
- *Configure TSF*

Core Management Console:

- *View Dashboard*
- *Configure TSF*

.[41]

FMT_SMR.1	*Security roles*

Hierarchical to: No other components.

Dependencies: FIA_UID.1 Timing of identification

FMT_SMR.1.1 The TSF shall maintain the roles *identified in FMT_MOF.1.1*[42].

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

6.2.6 Protection of the TSF (FPT)

FPT_ITT.1	*Basic internal TSF data transfer protection*

Hierarchical to: No other components.

Dependencies: No dependencies.

FPT_ITT.1.1 The TSF shall protect TSF data from disclosure **and** modification[43] when it is transmitted between separate parts of the TOE.

FPT_TUD.1	*Trusted Update*

Hierarchical to: No other components.

Dependencies: FCS_COP.1 Cryptographic Operation

FPT_TUD.1.1 The TSF shall provide *all roles*[44] the ability to query the currently executing version of the TOE firmware/software and no other TOE firmware/software version[45].

FPT_TUD.1.2 The TSF shall provide *Core Admin*[46] the ability to manually initiate updates to TOE firmware/software and support automatic updates[47].

FPT_TUD.1.3 The TSF shall provide means to authenticate firmware/software updates to the TOE using a digital signature mechanism[48] prior to installing those updates.

6.2.7 Trusted Path/Channels (FTP)

FTP_TRP.1	*Trusted path*

Hierarchical to: No other components.

Dependencies: No dependencies.

[41] [assignment: *list of management functions to be provided by the TSF*]
[42] [assignment: *the authorised identified roles*]
[43] [selection: disclosure, modification]
[44] [assignment: *Administrators*]
[45] [selection: the most recently installed version of the TOE firmware/software; no other TOE firmware/software version]
[46] [assignment: *Administrators*]
[47] [selection: support automatic checking for updates, support automatic updates, no other update mechanism]
[48] [selection: digital signature mechanism, published hash]

FTP_TRP.1.1 The TSF shall provide a communication path between itself and remote[49] users that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from modification **and** disclosure[50].

FTP_TRP.1.2 The TSF shall permit remote users[51] to initiate communication via the trusted path.

FTP_TRP.1.3 The TSF shall require the use of the trusted path for *MetaDefender Core WebUI/REST API and Kiosk Management Console WebUI*[52].

6.3 TOE Security Assurance Requirements

This section defines the assurance requirements for the TOE. Assurance requirements are taken from the CC Part 3 and are EAL4+, augmented with ALC_DVS.2, ALC_FLR.2, AVA_VAN.5.

Table 24 - Assurance Requirements

Assurance Requirements		
Class ASE: Security Target evaluation	ASE_CCL.1	Conformance claims
	ASE_ECD.1	Extended components definition
	ASE_INT.1	ST introduction
	ASE_OBJ.2	Security objectives
	ASE_REQ.2	Derived security requirements
	ASE_SPD.1	Security problem definition
	ASE_TSS.1	TOE summary specification
Class ALC: Life Cycle Support	ALC_CMC.4	Production support, acceptance procedures and automation
	ALC_CMS.4	Problem tracking CM coverage
	ALC_DEL.1	Delivery procedures
	ALC_DVS.2	Development security
	ALC_FLR.2	Flaw reporting procedures
	ALC_LCD.1	Developer defined life-cycle model
	ALC_TAT.1	Well-defined development tools
Class ADV: Development	ADV_ARC.1	Security architecture description
	ADV_FSP.4	Complete functional specification

[49] [selection: remote, local]
[50] [selection: modification, disclosure, [assignment: *other types of integrity or confidentiality violation*]].
[51] [selection: the TSF, local users, remote users]
[52] [selection: initial user authentication, [assignment: *other services for which trusted path is required*]]

	ADV_TDS.3	Basic modular design
	ADV_IMP.1	Implementation representation of the TSF
Class AGD: Guidance documents	AGD_OPE.1	Operational user guidance
	AGD_PRE.1	Preparative procedures
Class ATE: Tests	ATE_COV.2	Analysis of coverage
	ATE_FUN.1	Functional testing
	ATE_IND.2	Independent testing – sample
	ATE_DPT.1	Testing: basic design
Class AVA: Vulnerability assessment	AVA_VAN.5	Advanced methodical vulnerability analysis

6.4 Security Requirements Rationale

This section provides the rationale for necessity and sufficiency of security requirements, demonstrating that each of the security objectives is addressed by at least one security requirement, and that every security functional requirement is directed toward solving at least one objective.

6.4.1 Security Requirements Coverage

The table in section 6.4.1.1 provides a mapping between the Security Functional Requirements and the Security Objectives, illustrating that each Security Functional Requirement covers at least one Objective and that each Objective is covered by at least one Security Functional Requirement.

6.4.1.1 Security Functional Requirements Related to Security Objectives

The following table should give a rationale that all Security Objectives are covered by at least one SFR and to show that there is no Security Objective not covered and no SFR used that is not required.

Table 25 - Security Functional Requirements Related to Security Objectives

	O.DETECT	O.DLP	O.ACCESS	O.MGMT_AUDIT	O.KIOSK_AUDIT	O.COMMS	O.UPDATE
FTR_SCN.1	X	X					
FTR_SCN.2	X						
FTR_CDR.1	X						
FTR_FVA.1	X						
FTR_DLP.1		X					
FTR_TAR.1	X	X					

FAU_GEN.1				X	X		
FAU_GEN.2				X	X		
FCS_CKM.1						X	X
FCS_CKM.4						X	X
FCS_COP.1						X	X
FIA_UAU.1			X				
FIA_UAU.7			X				
FIA_UID.1			X				
FMT_MOF.1			X				
FMT_SMF.1	X	X	X				
FMT_SMR.1			X				
FPT_ITT.1						X	
FPT_TUD.1							X
FTP_TRP.1						X	

Table 26 - Suitability of SFRs

Objectives	SFR supports the objective by requiring:
O.DETECT	FTR_SCN.1 – submission of files for scanning FTR_SCN.2 – scanning engines to detect malware FTR_CDR.1 – removal of potentially malicious content from files FTR_FVA.1 – detection of application files containing known vulnerabilities FTR_TAR.1 – reporting scanning results FMT_SMF.1 – specification of file handling / blocking policies Together the SFRs will result in the detection of file-based threats and response according to policy.
O.DLP	FTR_SCN.1 – submission of files for scanning FTR_DLP.1 – detection of sensitive data within files FTR_TAR.1 – reporting scanning results FMT_SMF.1 – specification of file handling / blocking policies Together the SFRs will result in the detection sensitive data in files and response according to policy.
O.ACCESS	FIA_UAU.7 – protection of authentication feedback FIA_UAU.1 – authentication of users FIA_UID.1 – identification of users FMT_MOF.1 – management of security functions behavior FMT_SMF.1 – specification of management functions FMT_SMR.1 – security roles Together the SFRs will result in protection of the management interfaces from unauthorized access.
O.MGMT_AUDIT	FAU_GEN.1 – audit events for the Core Management Console and Kiosk Management Console FAU_GEN.2 – user identity in audit events Together the SFRs will result in audited use of management interfaces.
O.KIOSK_AUDIT	FAU_GEN.1 – audit events for the Kiosk Scanning Functions FAU_GEN.2 – user identity in audit events Together the SFRs will result in audited use of scanning functions.
O.COMMS	FCS_CKM.1 – secure key generation FCS_CKM.4 – secure destruction of the keys FCS_COP.1 – cryptographic operations in support of communications FPT_ITT.1 – secure communication between the Kiosk and Core

	FTP_TRP.1 – secure communication with remote administrators Together the SFRs will result in protected communications between TOE components and between the TOE and remote users.
O.UPDATE	FCS_CKM.1 – secure key generation FCS_CKM.4 – secure destruction of the keys FCS_COP.1 – cryptographic operations in support of trusted updated FPT_TUD.1 – digitally signed software updates Together the SFRs will result in authenticated software updates.

6.4.1.2 Security Assurance Requirements Rationale

EAL4+, augmented with ALC_DVS.2, ALC_FLR.2, AVA_VAN.5 was chosen since it is best suited to address the stated security objectives of the TOE. EAL4+, augmented with ALC_DVS.2, ALC_FLR.2, AVA_VAN.5 challenges vendors to use best (rather than average) commercial practices, and at the same time it allows the vendor to evaluate their product at a detailed level while benefitting from the Common Criteria Recognition Agreement, which is a recognized agreement in many countries of the world. The chosen assurance level appropriately challenges the threats defined in the TOE environment. At EAL4+, augmented with ALC_DVS.2, ALC_FLR.2, AVA_VAN.5, penetration testing is performed by the evaluator assuming an attack potential of High.

6.5 Requirements Dependency Rationale

6.5.1 Rationale Showing that Dependencies are Satisfied

The SFRs in this ST satisfy all the required dependencies listed in the Common Criteria. The table in this section lists each requirement to which the TOE claims conformance and indicates whether the dependent requirements are included. As it is indicated by the table, all dependencies are fulfilled.

6.5.1.1 Security Functional Requirements Dependencies

The following table provides a summary of the SFRs and their dependencies

Table 27 - Summary of Security Functional Requirements Dependencies

Component	Dependency	Which is:
FTR_SCN.1	None	-
FTR_SCN.2	FTR_SCN.1	Included
FTR_CDR.1	FTR_SCN.1	Included
FTR_FVA.1	FTR_SCN.1	Included
FTR_DLP.1	FTR_SCN.1	Included
FTR_TAR.1	FTR_SCN.2	Included
	FTR_CDR.1	Included
	FTR_FVA.1	Included
	FTR_DLP.1	Included
	FMT_SMR.1	Included
FAU_GEN.1	FPT_STM.1	Not included Per OE.TIME, the environment provides reliable time.
FAU_GEN.2	FAU_GEN.1	Included
	FIA_UID.1	Included
FCS_CKM.1	[FCS_CKM.2, or FCS_COP.1]	FCS_COP.1 included FCS_CKM.4 included

	FCS_CKM.4	
FCS_CKM.4	[FDP_ITC.1, or FDP_ITC.2, or FCS_CKM.1]	FCS_CKM.1 included
FCS_COP.1	FDP_ITC.1, or FDP_ITC.2, or FCS_CKM.1	Included
	FCS_CKM.4	Included
FIA_UAU.1	FIA_UID.1	Included
FIA_UAU.7	FIA_UAU.1	Included
FIA_UID.1	None	-
FMT_MOF.1	FMT_SMR.1	Included
	FMT_SMF.1	Included
FMT_SMF.1	None	-
FMT_SMR.1	FIA_UID.1	Included
FPT_ITT.1	None	-
FPT_TUD.1	FCS_COP.1	Included
FTP_TRP.1	None	-

6.5.1.2 Security Assurance Requirements Dependencies

The following table provides a summary of the SARs and their dependencies.

Table 28 - Summary of Security Assurance Requirements Dependencies

Component	Depends On:	Which is:
ADV_ARC.1	ADV_FSP.1	hierarchically higher component ADV_FSP.4 is included
	ADV_TDS.1	hierarchically higher component ADV_TDS.3 is included.
ADV_FSP.4	ADV_TDS.1	hierarchically higher component ADV_TDS.3 is included
ADV_IMP.1	ADV_TDS.3	included
	ALC_TAT.1	included
ADV_TDS.3	ADV_FSP.4	included
AGD_OPE.1	ADV_FSP.1	hierarchically higher component ADV_FSP.4 is included
AGD_PRE.1	no dependencies	not applicable
ALC_CMC.4	ALC_CMS.1	hierarchically higher component ALC_CMS.4 is included
	ALC_DVS.1	hierarchically higher component ALC_DVS.2 is included.
	ALC_LCD.1	included
ALC_CMS.4	no dependencies	not applicable
ALC_DEL.1	no dependencies	not applicable
ALC_DVS.2	no dependencies	not applicable

ALC_FLR.2	no dependencies	not applicable
ALC_LCD.1	no dependencies	not applicable
ALC_TAT.1	ADV_IMP.1	included
ASE_INT.1	no dependencies	not applicable
ASE_CCL.1	ASE_INT.1	included
	ASE_ECD.1	included
	ASE_REQ.1	hierarchically higher component ASE_REQ.2 is included
ASE_SPD.1	no dependencies	not applicable
ASE_OBJ.2	ASE_SPD.1	included
ASE_ECD.1	no dependencies	not applicable
ASE_REQ.2	ASE_OBJ.2	included
	ASE_ECD.1	included
ASE_TSS.1	ASE_INT.1	included
	ASE_REQ.1	hierarchically higher component ASE_REQ.2 is included
	ADV_FSP.1	hierarchically higher component ADV_FSP.4 is included
ATE_COV.2	ADV_FSP.2	hierarchically higher component ADV_FSP.4 is included
	ATE_FUN.1	included
ATE_FUN.1	ATE_COV.1	hierarchically higher component ATE_COV.2 is included
ATE_IND.2	ADV_FSP.2	hierarchically higher component ADV_FSP.4 is included
	AGD_OPE.1	included
	AGD_PRE.1	included
	ATE_COV.1	hierarchically higher component ATE_COV.2 is included
	ATE_FUN.1	included
ATE_DPT.1	ADV_ARC.1	included
	ADV_TDS.2	hierarchically higher component ADV_TDS.3 is included
	ATE_FUN.1	included
AVA_VAN.5	ADV_ARC.1	included
	ADV_FSP.4	included
	ADV_IMP.1	included
	ADV_TDS.3	included
	AGD_OPE.1	included
	AGD_PRE.1	included
	ATE_DPT.1	included

7 TOE Summary Specification

7.1 File Threat Analysis

This security function implements the SFRs shown in the table below.

Table 29 - File Threat Analysis SFRs

Requirement	Title
FTR_SCN.1	Submission of Files for Scanning
FTR_SCN.2	Supported Scanning Engines
FTR_CDR.1	Content Disarm and Remove
FTR_FVA.1	File-based Vulnerability Assessment
FTR_DLP.1	File-based Data Loss Prevention
FTR_TAR.1	Threat Analysis Report
FMT_SMF.1	Specification of Management Functions

The TOE orchestrates the scanning and analysis of files for threats and generates associated session reports. According to the administrator defined workflow/policies, the file analysis goes through multiple phases to detect if the file is either malicious, contains sensitive or vulnerable data and if it's safe to be consumed by the end-user.

Methods of file submission for scanning with:

- user presentation of portable media at the Kiosk, and
- upload of files to the Core Management Console
- call REST API of MetaDefender Core to upload file content along with filename.

The following type files are accepted: exe, dll, sys, msi, cab, dmg, zip, tar, gz, bz2, bin.

MetaDefender Kiosk will submit all the selected files to MetaDefender Core for analysis. The submission happens as follows:

a) MetaDefender Kiosk will call MetaDefender Core's REST API to upload the file (POST /file)
b) MetaDefender Core will respond with a unique identifier for that file (data_id)
c) Multiple files will be submitted in parallel, until MetaDefender Core's queue is full
d) MetaDefender Core executes a multi-stage analysis flow for each submitted file
e) MetaDefender Kiosk will check periodically to see if the analysis is complete (short polling), using the previously received data_id
f) MetaDefender Kiosk will retrieve results and perform the associated file handling actions according to the administrator defined policy
g) MetaDefender Kiosk will present the summary report to the user

The TSF relevant technologies are described in the following sections.

The following information will be in the scan result:

- Final verdict (Blocked or Allowed)
- Threat name (if any)
- Scan result (malicious/suspicious/potentially unwanted application/no threat found/etc.)
- All the engines participate in scanning.

- Analysis Time in milliseconds

Scan results are stored according to an administrator defined retention policy (please see chapter Data Retention of [MDCore]) and reports are dynamically generated from stored scan results (please see chapter Restriction of [MDCore]).

The TOE supports generation of the following reports: Kiosk Scan Results:

- Kiosk Scan Results
  - User ID
  - Profile (i.e. scan profile)
  - Session ID
  - Processing Time
  - Device Information (i.e. media scanned)
  - File Processing Details
  - Blocked Actions Taken
  - Allows Actions Taken
  - Detailed Scan Results

The following actions are available when vulnerabilities are detected:

- report,
- block,
- allow,
- quarantine file on the local KIOSK,
- quarantine file on MD Core,
- delete file,
- copy/move file to another location,
- run a custom script applied to that file.

Roles assigned with permissions (roles with Overall Results Only and Per Engine Results permissions may only view a subset of report information):

- Overall Results Only,
- Per Engine Results,
- Full Details.

7.1.1 Scanning with multiple anti-malware engines

The engines are downloaded to the Core and are stored locally.

MetaDefender Core offers the ability to license different packages for the multi-scanning module (called Metascan). Depending on the license key, the licensed modules are loaded into MetaDefender Core.

This model ensures the end-customer can analyze all the files in their environment, with no data being submitted to either OPSWAT or the engine vendors for analysis. This guarantees that the solution works in online, offline and even air-gapped environments.

Once a file is submitted, the file is passed to all licensed engines for analysis, making use of parallel processing to ensure the highest throughput. When all the engines have completed the analysis, the result is provided back to MetaDefender Core.

7.1.2 Deep CDR / Data sanitization

Deep CDR allows users to sanitize productivity documents, by removing embedded active objects that might drive a malicious behavior (macros, OLE objects, ActiveX controls, etc. In Office docs).

Workflow configuration will allow the administrator to define, at file type level, which type will be sanitized and the method to use for sanitization, either:

a) CDR method (rebuild the file without the active content)
b) Filetype conversion method (convert a file to a different filetype, which will break the active content, but will also change the usability of the file, e.g. Excel file to PDF).

The supported file types and content are listed at FTR_CDR.1.2.

7.1.3 File-based Vulnerability Assessment

In case the submitted file is an application file (installer, patch, firmware update, etc.), the File based Vulnerability Assessment Engine will map the file to its known vulnerabilities.

OPSWAT maintains a repository of known applications (and files belonging to those applications) and performs matching at file level to known vulnerabilities for those applications based on the following vulnerability databases:

a) National Vulnerability Database https://nvd.nist.gov/
b) Microsoft Security Update Guide https://portal.msrc.microsoft.com/en-us/security-guidance

7.1.4 Proactive DLP

The TOE supports applying data loss prevention techniques to

a) Microsoft Office files (DOC, DOCX, XLS, XLSX, PPT, PPTX)
b) OpenOffice files (ODT, ODS, ODP),
c) Adobe PDF,
d) Text base files (TXT, HTML, CSV, XML),
e) Email files (EML, MSG)
f) Images (TOE uses optical character recognition in this case)

The TOE is able to identify the following types of sensitive data in files based on the administrator defined policy:

a) credit card numbers
b) social insurance numbers
c) IPv4 address
d) CIDR range
e) regular expression data pattern

MetaDefender Core performs the analysis and will identify the sensitive data.

Kiosk will enforce the file handling policy to either:

a) block the file
b) attempt to delete the file from the original media
c) report the type of sensitive data that was identified

7.1.5 File Handling

The Kiosk enforces file handing actions for processed files.

Handling actions for Blocked files:

a) **No action:** Report only.
b) **Stop processing**: The Kiosk session to stop processing immediately after the first blocked file is found. Kiosk will alert the user that a blocked file was found and go directly to the session summary after the user has acknowledged the message.
c) **Remove file:** Blocked file will be removed from the original media and optionally quarantined.
d) **Sanitized file handling**: If a file has been sanitized, the original file may either be replaced by the sanitized file, or the sanitized file will be copied to the original media and the original file will be left untouched.
e) **Copy to:** Copy blocked files to specified locations. The original file may optionally be deleted.

Handling actions for allowed files include:

a) **No action:** Report only.
b) **Wipe and copy to original media:** Will copy allowed files back to the original media after the original media has been formatted.
c) **Sanitized file handling**: If a file has been sanitized (i.e. via CDR), the original file may either be replaced by the sanitized file, or the sanitized file will be copied to the original media and the original file will be left untouched.
d) **Copy to:** Copy allowed files to specified locations. The original file may optionally be deleted.

7.1.6 Reporting

After media has been processed, the session results appear. If any file was not processed by MetaDefender, a warning will pop up indicating that not all files were processed.

The session results include whether processing was completed or aborted, the number of files allowed and blocked and the total number of files processed.

The session result page includes the following buttons:

a) **Allowed**: If allowed files are found, then the Allowed count will appear. Click this button to go to the Allowed file summary screen.
b) **Blocked**: If blocked files are found, then the Blocked count will appear. Click this button to go to the Blocked file summary screen.

c) **Copy & Print**: Clicking this button will begin the file transfer process to any destination configured. If printing is enabled, the session results will be printed to the default printer.

The Blocked File Details screen displays the blocked files detected by MetaDefender Kiosk during processing. The user may click a blocked file to view more details.

7.2 Protected Communications

This security function implements the SFRs shown in table below.

Table 30 - Protected Communications SFRs

Requirement	Title
FCS_CKM.1	Cryptographic key generation
FCS_CKM.4	Cryptographic key destruction
FCS_COP.1	Cryptographic Operation
FPT_ITT.1	Basic internal TSF data transfer protection
FTP_TRP.1	Trusted path

The TOE protects communication using TLS as follows:

a) MetaDefender Core. Implements a web server (nginx) that requires HTTPS for the REST API and Core Management Console Web GUI.
b) MetaDefender Kiosk. Implements a web server (nginx) that requires HTTPS for access to the Kiosk Management Console Web GUI. In addition, the Kiosk makes use of the Core's REST API by submitting request over HTTPS.

In all cases, the underlying TLS implementation is provided by OpenSSL. In the evaluated configuration, the TLS implementation has the following characteristics:

MetaDefender Kiosk

a) TLS 1.2 is supported
b) Supported ciphers:
    i. TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
    ii. TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

MetaDefender Core

a) TLS 1.2 is supported
b) Supported ciphers:
    i. TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
    ii. TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
    iii. TLS_ECDHE_RSA_WITH_ARIA_256_GCM_SHA384
    iv. TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
    v. TLS_ECDHE_RSA_WITH_ARIA_128_GCM_SHA256
    vi. TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
    vii. TLS_ECDHE_RSA_WITH_CAMELLIA_256_CBC_SHA384
    viii. TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
    ix. TLS_ECDHE_RSA_WITH_CAMELLIA_128_CBC_SHA256
    x. TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA

xi. TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
xii. TLS_RSA_WITH_AES_256_GCM_SHA384
xiii. TLS_RSA_WITH_AES_256_CCM
xiv. TLS_RSA_WITH_ARIA_256_GCM_SHA384
xv. TLS_RSA_WITH_AES_128_GCM_SHA256
xvi. TLS_RSA_WITH_AES_128_CCM
xvii. TLS_RSA_WITH_ARIA_128_GCM_SHA256
xviii. TLS_RSA_WITH_AES_256_CBC_SHA256
xix. TLS_RSA_WITH_CAMELLIA_256_CBC_SHA256
xx. TLS_RSA_WITH_AES_128_CBC_SHA256
xxi. TLS_RSA_WITH_CAMELLIA_128_CBC_SHA256
xxii. TLS_RSA_WITH_AES_256_CBC_SHA
xxiii. TLS_RSA_WITH_CAMELLIA_256_CBC_SHA
xxiv. TLS_RSA_WITH_AES_128_CBC_SHA
xxv. TLS_RSA_WITH_CAMELLIA_128_CBC_SHA

7.3 User Authentication

This security function implements the SFRs shown in table below.

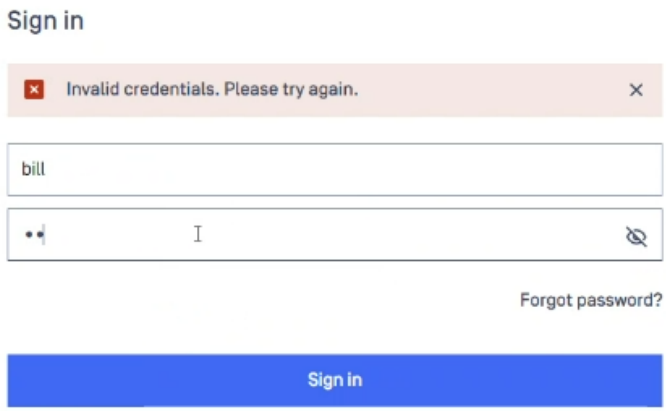
Table 31 - User Authentication SFRs

Requirement	Title
FIA_UAU.1	Timing of authentication
FIA_UAU.7	Protected authentication feedback
FIA_UID.1	Timing of identification

The TSF shall provide only a loading screen and login form to the user while the authentication is in progress. Users need to wait until the authentication finishes successfully.

The user will only see bullets[53] and see the number of the characters typed. When the login is successful the user will be redirected, if the login is unsuccessful an error message will be shown to the user.

Figure 4 - Login failure



[53] Points, which hide the typed characters for the password.

7.3.1 Kiosk Scanning Users

In the evaluated configuration, Kiosk scanning users are authenticated using Windows Login (i.e. the TOE invokes Windows Login). TOE administrators can choose to allow guest users, and whether to restrict the users by domain. If selected, only users on the same domain as the system are allowed to use MetaDefender Kiosk. If this is not selected, users will be able to enter authentication information for users on any domain as well as local system users.

7.3.2 Kiosk Management Console User

Kiosk administrates are authenticated by means of a username and password against a local database.

7.3.3 Core REST API / Management Console User

Core users are authenticated by means of a username and password against a local database.

7.4 Security Management

This security function implements the SFRs shown in table below.

Table 32 - Security Management SFRs

Requirement	Title
FAU_GEN.1	Audit Data Generation
FAU_GEN.2	User Identity Association
FCS_CKM.1	Cryptographic key generation
FCS_CKM.4	Cryptographic key destruction
FCS_COP.1	Cryptographic Operation
FMT_MOF.1	Management of security functions behavior
FMT_SMF.1	Specification of Management Functions
FMT_SMR.1	Security roles
FPT_TUD.1	Trusted Update

The users have the ability to determine the behavior of, disable, enable, or modify the behavior of the Kiosk and Core functions as shown in the Table 21 – Roles and behaviors.

Table 33 - Security management

Role	Functions	Permissions
Kiosk		
Administrator	All	All
Auditor	Dashboard	
	Logging History	
Core		
Administrators	All	All
Security administrators	All except User Management, Licensing, Config History, Dashboard\Executive Report, Settings\Security, Settings\Network, Settings\Data Retention, Settings\Export/Import, Settings\Central Management.	Can view and edit Dashboard\Usage and System Health, Workflow Management, Inventory, Settings\Health Check, Settings\Security. View only and cannot edit History\Processing, Quarantine, Update History.

		Cannot edit or view User Management, Licensing, Config History, Dashboard\Executive Report, Settings\Security, Settings\Network, Settings\Data Retention, Settings\Export and Import, Settings\Central Management.
Security auditor	All except Dashboard\Executive Report, Inventory\Password Storage, Settings\Email Notification, Settings\Network, Settings\Export/Import, Settings\Central Management.	Cannot view Dashboard\Executive Report, Inventory\Password Storage, Settings\Email Notification, Settings\Network, Settings\Export/Import, Settings\Central Management. View only and cannot edit other functions.
Help desk	All except Dashboard\Executive Report and System Health, User Management, Licensing, Settings\Security, Settings\Network, Settings\Module Update, Settings\Data Retention, Settings\Health Check, Settings\Export/Import, Settings\Central Management, History\Config history, History\Quarantine, Inventory\Certificates, Inventory\Webhook authentication.	View only and cannot edit Dashboard\Usage, Processing\History, Processing\Module Update, Workflow Management, Inventory\Modules, Inventory\Skip by Hash, Inventory\Post Actions, Inventory\External Scanners, Settings\General. Cannot edit or view Dashboard\ Executive Report and System Health, User Management, Licensing, Settings\Security, Settings\Network, Settings\Module Update, Settings\Data Retention, Settings\Health Check, Settings\Export/Import, Settings\Central Management, History\Config history, History\Quarantine, Inventory\Certificates, Inventory\Webhook authentication.

7.4.1 Kiosk Management Console

The Kiosk Management Console provides the following management functions:

a) Create / manage users

b) View the Dashboard - the first page that is seen when logging in to the MetaDefender Kiosk Management Console. This page provides a summary of all of the files that have been processed by MetaDefender Kiosk.
c) Configuration - the configuration pages allow administrators to configure all MetaDefender Kiosk settings that apply to all users of MetaDefender Kiosk.

Kiosk Management Console users are assigned to roles as defined at FMT_SMR.1. The TOE will enforce access control in accordance with the privileges assigned to each role.

7.4.2 Core Management Console

The Core Management Console provides the following management functions:

a) Create / manage users
b) View Dashboard - gives a general overview of MetaDefender Core status.
c) Configuration - the configuration pages allow administrators to configure all MetaDefender Core settings.

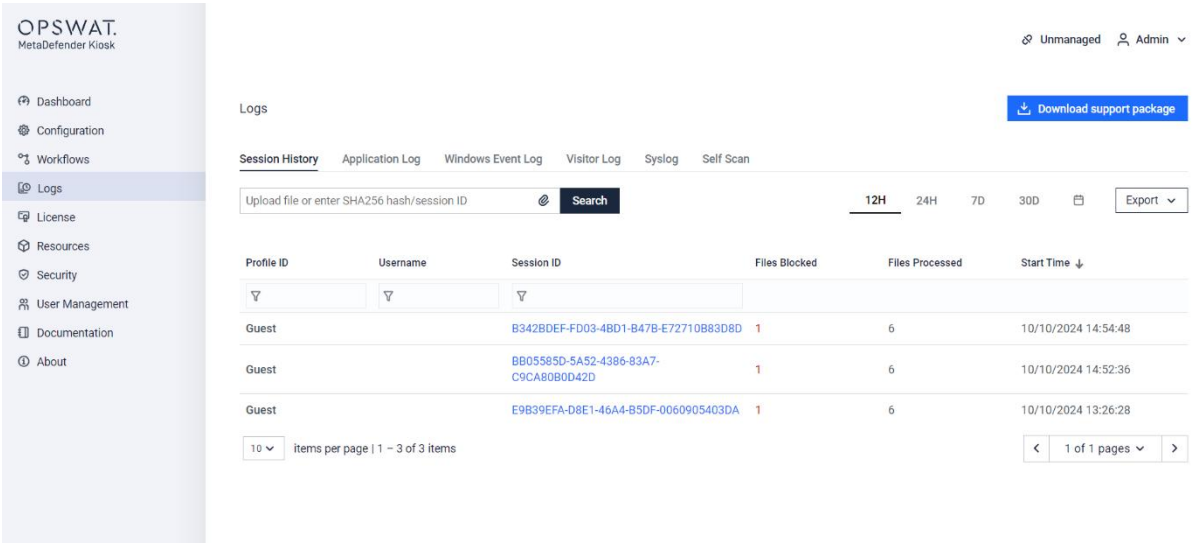
Core Management Console users are assigned to roles as defined at FMT_SMR.1. The TOE will enforce access control in accordance with the privileges assigned to each role.

7.4.3 Security Audit

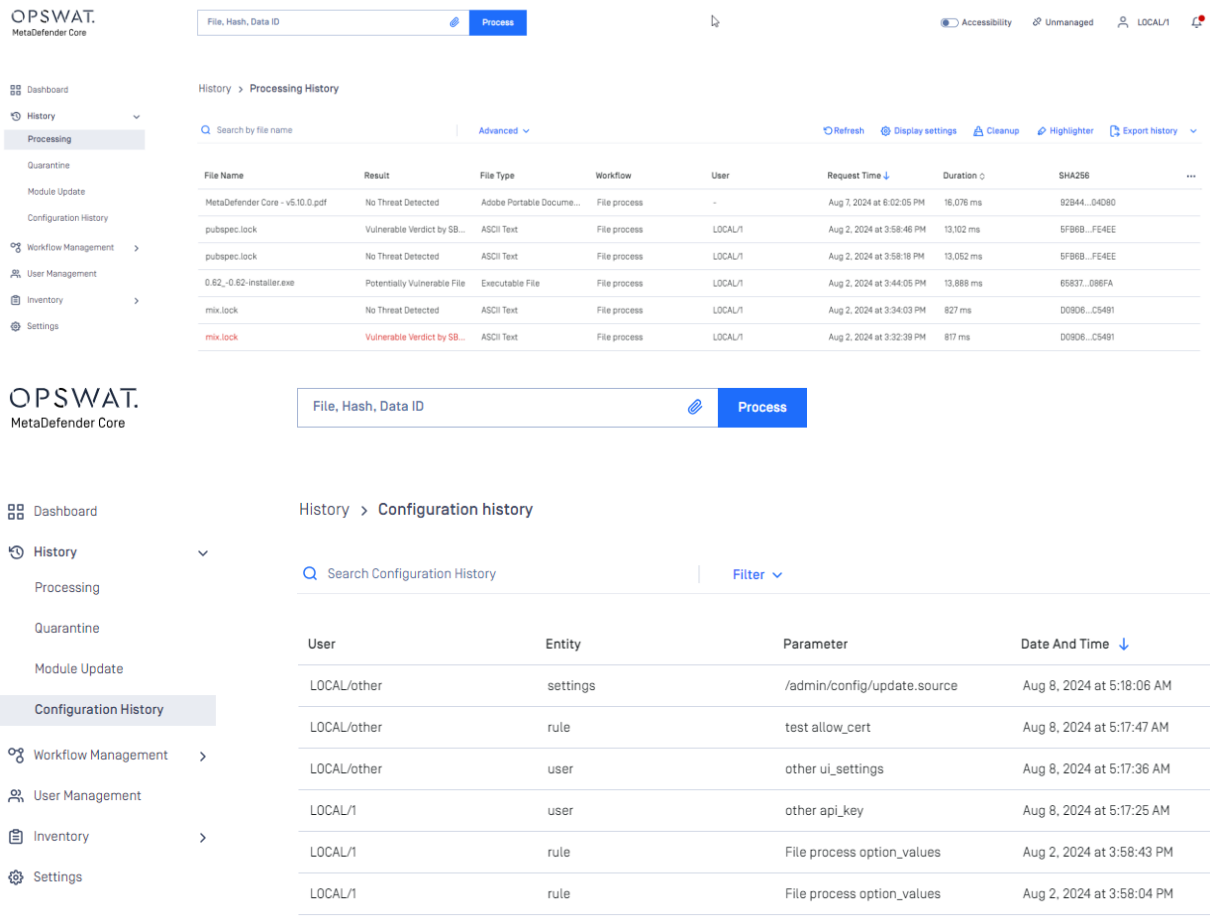
The TOE generates audit logs and stores them locally. Each TOE component (Kiosk and Core) maintains its own audit log. The audit events and details are described at FAU_GEN.1.

Each audit event includes the date and time of the event, type of event, subject identity (if applicable), and the outcome of the event.

Admin can audit user activities on the KIOSK Console to view details of the sessions performed by the user on the KIOSK.



2. MetaDefender Core provides a Core Management Console for Admin to audit user activities regarding file processing events and administrative events.



Besides Core Management Console, Admin can also check product logs and configure syslog for monitoring event and user activity (https://docs.opswat.com/mdcore/configuration/logging).

7.4.4 Trusted Update

MetaDefender Core implements trusted updates for signature files and installed 3rd party engines as follows:

a) All updates are digitally signed with the OPSWAT code signing private key (RSA / SHA2-256)
b) Updates are fetched from OPSWAT cloud infrastructure
c) Core uses the OPSWAT code signing public key to verify the digital signature prior to installing

Updates to the Core and Kiosk software are performed manually and are verified by the Microsoft Authenticode mechanism.

7.4.5 Key generation and destruction

The TOE generates cryptographic keys in accordance with a specified cryptographic key generation algorithm, the specified cryptographic key sizes and the met standards described in Table 20 – Key generation and destruction.

7.4.5.1 MetaDefender Core

PostgreSQL connection info:

A key for encrypting PostgreSQL credentials is generated by OpenSSL. This key is automatically generated only once during the installation.

During uninstallation, the key will be destroyed and cleaned up automatically with zeroization by the product.

The algorithm: AES

The key size: 256

Email server's password:

A key for encrypting Email server credentials is generated by OpenSSL. The key is automatically generated when Administrator first setup Email Configuration.

When removing Email Configuration, the product will destroy the key and clean it up with zeroization automatically.

The algorithm: AES

The key size: 256

Proxy server's password:

A key for encrypting Proxy server credentials is generated by OpenSSL. The key is automatically generated every time Administrator configures Proxy Configuration or disable/enable it.

During uninstallation, the product will destroy the key and clean it up with zeroization automatically.

The algorithm: AES

The key size: 256

User password:

Salt is automatically generated upon user creation or password update. Salt value is a UUID compliant with [RFC4122] and [FIPS 180-4]. The product hashes the user-input password with unique salt using SHA256.

During uninstallation, the product will destroy the salt and clean it up with zeroization automatically.

The algorithm: SHA256

Asymmetric key for webhook authentication:

A key for encrypting asymmetric key for webhook authentication is generated by OpenSSL. The key is automatically generated upon the first webhook authentication certificate is created.

When all webhook authentication certificates are deleted or during product uninstallation, the product will destroy the key and clean it up with zeroization automatically.

The algorithm: AES

The key size: 256

Password storage:

A key for encrypting passwords in its storage is generated by OpenSSL. The key is automatically generated when the first storage is created.

When all the storages are removed or during product uninstallation, the product will destroy the key and clean it up with zeroization automatically.

The algorithm: AES

The key size: 256

7.4.5.2 MetaDefender KIOSK

User password:

Salt is automatically generated upon user creation or password update. The product encrypts the user-input password with unique salt using PBKDF2.

When removing user or updating the user password, the product will destroy the salt and clean it up with zeroization automatically.

The algorithm: PBKDF2

The key size: 128

Exit KIOSK password:

Salt is automatically generated upon user configure the password to exit the KIOSK application. The product hashes the exit password with unique salt using SHA256.

When the administrator disables the Exit KIOSK password, the product will destroy the salt and clean it up with zeroization automatically.

The algorithm: HMAC-SHA

The key size: 256

8 Acronyms

Table 34 - Acronyms

Acronym	Meaning
AES	Advanced Encryption Standard
AI	Artificial Intelligence
ANSI X9.31-1998	Digital Signatures using Reversible Public Key Cryptography for the Financial Services Industry (rDSA). Version. 1.0
CAVP	Cryptographic Algorithm Validation Program
CC	Common Criteria
CDR	Deep Content Disarm and Reconstruction
CIDR	Classless Inter-Domain Routing
DES	Data Encryption Standard

DLP	Proactive Data Loss Prevention
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral
GCM	Galois Counter Mode
HMAC	Hash-based message authentication code
IT	Information Technology
ML	Machine Learning
OS	Operating System
OSP	Organizational Security Policy
PKCS	Password-Based Cryptography Specification
PKCS1.5	PKCS #1: RSA Encryption Version 1.5
PKCS #1 v2.1	Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1
PKCS #5	Password-Based Cryptography Specification Version 2.0
PP	Protection Profile
PSS	Probabilistic Signature Scheme
PUA	Potential Unwanted Applications
PUP	Potentially Unwanted Program
REST API	Representational State Transfer Application Programming Interface
RSA	Rivest–Shamir–Adleman
SA	Security Association
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
ST	Security Target
TOE	Target of Evaluation
TSF	TOE Security Functionality
Web UI	Web User Interface

9 Bibliography

[CC_P1] Common Criteria, Part 1: Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model, Version 3.1, Revision 5, April 2017, CCMB-2017-04-001

[CC_P2] Common Criteria, Part 2: Common Criteria for Information Technology Security Evaluation Part 2: Security Functional Requirements, Version 3.1, Revision 5, April 2017, CCMB-2017-04-002

[CC_P3] Common Criteria, Part 3: Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Requirements, Version 3.1, Revision 5, April 2017, CCMB-2017-04-003

[CEM] Common Methodology for Information Technology Security Evaluation, Evaluation Methodology, Version 3.1, Revision 5, April 2017, CCMB-2017-04-004

[FIPS 140-2] FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION Security Requirements for Cryptographic Modules May 25, 2001 (Change Notice 2, 12/3/2002) - https://doi.org/10.6028/NIST.FIPS.140-2

[FIPS 197] FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION Advanced Encryption Standard (AES), Published November 26, 2001; Updated May 9, 2023 - https://doi.org/10.6028/NIST.FIPS.197-upd1

[SP 800-38A] NIST Special Publication 800-38A Recommendation for Block 2001 Edition Cipher Modes of Operation, Date Published: December 2001 - https://doi.org/10.6028/NIST.SP.800-38A

[FIPS 198-1] FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION The Keyed-Hash Message Authentication Code (HMAC), July 2008 - https://doi.org/10.6028/NIST.FIPS.198-1

[FIPS 186-5] FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION Digital Signature Standard (DSS), February 3, 2023 - https://csrc.nist.gov/pubs/fips/186-5/final

[MDCore] MetaDefender Core - v5.14.2_2025-12-04.pdf, 2025-12-04

[MDKiosk] MetaDefender Kiosk - v4.7.6_2025-12-04.pdf, 2025-12-04

[AGD] AGD Documentation MetaDefender Core & MetaDefender Kiosk Evaluation Assurance Level (EAL): EAL4+, augmented with ALC_DVS.2, ALC_FLR.2, AVA_VAN.5, v1.9, 2025-12-04

[18033-3] ISO/IEC 18033-3:2010 Information technology — Security techniques — Encryption algorithms Part 3: Block ciphers Published (Edition 2, 2010)

[FIPS-186-5] FIPS 186-5 Digital Signature Standard (DSS) Date Published: February 3, 2023

[9797-2] ISO/IEC 9797-2:2021 Information security — Message authentication codes (MACs) Part 2: Mechanisms using a dedicated hash-function Published (Edition 3, 2021)

[10118-3] ISO/IEC 10118-3:2018 IT Security techniques — Hash-functions Part 3: Dedicated hash-functions Published (Edition 4, 2018)

[SP800-56A] NIST SP 800-56A Rev. 3 Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography Date Published: April 2018

[ISO-19772] ISO/IEC 19772:2020 Information security — Authenticated encryption Published (Edition 2, 2020)

[RFC4122] A Universally Unique IDentifier (UUID) URN Namespace, P. Leach, M. Mealling, R. Salz, July 2005

[FIPS 180-4] FIPS PUB 180-4 FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION Secure Hash Standard (SHS), Date Published: August 2015