SECURITY TARGET

VOTIRO SECURE DATA SANITIZATION ENGINE

Reference	Votiro SDS ST	Status	Released
Version	1.2	Release Date	07 September 2017
Owner	Votiro	Pages	28 (Including Cover)

LIST OF CONTENTS

LIST OF TABLES

1 DOCUMENT INFORMATION

Prepared for

Votiro
126 Yigal Alon St.
Tel Aviv 67443, Israel
https://www.votiro.com/

Prepared by

BAE Systems Applied Intelligence Pty Limited
Level 1, 14 Childers Street
Canberra ACT 2601
Australia
http://www.baesystems.com/ai

1.1 Amendment history

Version	Date	Revisions
0.1	29/03/2017	Initial internal draft
0.2	10/04/2017	Draft release to Votiro
1.0	01/05/2017	Final release
1.1	08/05/2017	Edits to CDR Functionality

1.2 Copyright statement

2 SECURITY TARGET INTRODUCTION (ASE_INT)

2.1 ST Identification

ST Title	Security Target - Votiro Secure Data Sanitization Engine
ST Version	1.2
ST Release Date	07 September 2017

2.2 TOE Identification

TOE Name	Votiro Secure Data Sanitization Engine
TOE Version	7.1
Protection Profile(s)	N/A
CC Identification	• Common Criteria for Information Technology (IT) Security Evaluation, Version 3.1 (Revision 5), April 2017 • Common Methodology for Information Technology Security Evaluation, Evaluation methodology, Version 3.1 (Revision 5), April 2017

2.3 Document organisation

This document is divided into the following sections:

- Section 2 (Security Target Introduction (ASE_INT)) provides the introductory material for the ST;
- Section 3 (Conformance Claims (ASE_CCL)) provides the conformance claims for the evaluation;
- Section 4 (Security Problem Definition (ASE_SPD)) provides the security problem to be addressed by the TOE and the operational environment of the TOE;
- Section 5 (Security Objectives (ASE_OBJ)) defines the security objectives for the TOE and the environment;
- Section 6 (Extended Components Definition (ASE_ECD)) provides a definition and justification for any extended components from CC Parts 2 or 3 that have been developed for the evaluation;
- Section 7 (Security Functional Requirements (ASE_REQ)) contains the functional and assurance requirements derived from the Common Criteria, Part 2 and 3, respectively that must be satisfied by the TOE; and
- Section 8 (TOE summary specification (ASE_TSS)) provides a summary of the TOE specification, identifying the IT security functions provided by the TOE.
- Section 9 (Rationale) provides a justification of the selected security objectives and security requirements.

2.4 References

[1] Common Criteria Part 1 (Introduction and general model), Version 3.1 Revision 5, April 2017

[2] Common Criteria Part 2 (Security functional components), Version 3.1 Revision 5, April 2017

[3] Common Criteria Part 3 (Security assurance components), Version 3.1 Revision 5, April 2017

[4] Common Criteria Evaluation Methodology (CEM), Version 3.1 Revision 5, April 2017

2.5 TOE overview

The Target of Evaluation (TOE) is Votiro Secure Data Sanitization Engine (SDS).

Votiro SDS is designed to supplement the traditional sandboxing approach for securing files entering a network. It can help detect and prevent 0-day and other advanced threats that are designed to work around the general sandboxing approach. The TOE allows users to submit files to it via API calls which are then processed according to configurable policy rules.

The TOE uses these policy rules to sort files before allowing, denying or deconstructing the files according to the ruleset. These rules can be based on file type or file signature status (e.g. only allowing signed PDF files). SDS supported files are passed onto the file specific sanitizer engine to be deconstructed by the TOE in order to detect and remove harmful content embedded within the files. The TOE then reconstructs the files in such a way as to maintain the end user experience prior to making the safe version of the file available for download via the API system.

Votiro SDS can be installed to seamlessly integrate with an organisation's email system to inspect and process all incoming email traffic in accordance with its configured policies. Votiro SDS also integrates third party virus inspection capabilities. Note however that both email integration and virus inspection have not been tested as part of this evaluation.

The sanitizer engines implement several proprietary methods of sanitizing files such as introducing "noise" or microchanges into raster image files in order to disrupt any hidden/embedded payloads that might be malicious. It is capable of recursively inspect files such as excel files embedded within word documents or nested archive formats.

2.6 TOE description

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

2.6.1 Physical scope of the TOE

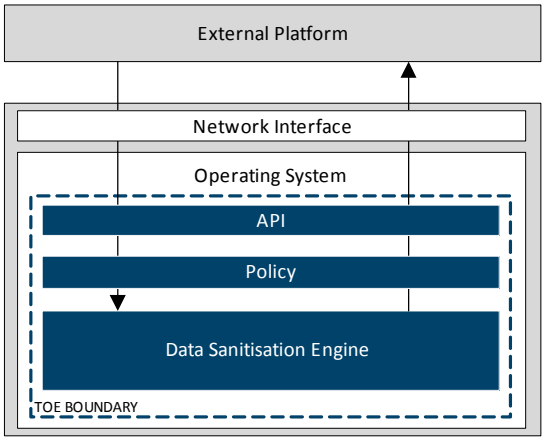

Figure 1 – TOE architecture

Votiro SDS is a software based TOE which consists of an API system, the SDS-WS (Secure Data Sanitization-Webservice) system and the Management System.

The API system allows API calls to be made to the TOE for the submission of files for processing and enables periodic polling of the TOE via API calls for the status of the file. Once processing is complete another API call allows the submitter to retrieve the processed file.

The SDS system of the TOE performs file inspection and processing in order to reduce the levels of active and malicious content within the submitted files. This can involve the removal of metadata, custom styles, printer settings and more.

The management system of the TOE handles the initial configuration of the TOE and processes the XML files that are used to configure individual policy and file processing settings.

The TOE is designed to run on a Windows Server 2016 OS and will work on hardware that meets the following requirements.

Table 1 – SDS WS Hardware Requirements.

Required Item	Minimum Specification
CPU	Quad Core Intel 2.66 GHz CPU
RAM	At least 16 GB of DDR3 RAM
Operating system drive	At least 30GB of free space
Workspace drive	At least 100 GB of free space
Network Card	Any

The TOE relies on the OS to provide cryptographic services for the API calls and accurate timesource information for logging. These services are provided by default in a standard installation of windows server 2016.

Additionaly the following software must be installed on ther Windows server in order to enable the correct installation of the TOE.

- Microsoft Visual C++ 2013 redistributable package.
- Microsoft Visual C++ 2015 redistributable package.
- Microsoft .NET Framework 3.5.1 and 4.5.

2.6.2 Logical scope of the TOE

The TOE provides the following logical security functions.

Table 2 – Logical scope of the TOE.

TSF	Description
Security Audit	The TOE is capable of generating audit logs of relevant security events including all file transactions and configuration changes. File transactions are identified by the file name whilst configuration changes are implicitly attributed to the administrator of the underlying OS.
User Data Protection	The TOE is capable of receiving files and processing them according to a predefined flow policy. These files can then be allowed, denied or handed off to the CDR subsystem of the TOE. The TOE also ensures that any residual information is deleted after it is no longer needed.
Security Management	The TOE is capable of accepting configuration via XML files
Privacy	The TOE is capable of maintaining the anonymity of its file deconstruction process.
Protection of the TSF	The TOE is capable of providing reliable timestamps and in the event of an error failing in a secure manner.
Resource Utilisation	The TOE is capable of limiting resource utilisation in order to prevent a DOS situation.

2.6.3 Summary of out-of-scope items

Votiro SDS contains several capabilities that are not in scope of this evaluation. The following features have not been examined as part of this evaluation:

- Flow Application Server (Management service for storage/retrieval of files);
- Microsoft SQL Database (Sub-component of the Flow Application Server);
- Exchange SDS Connector (Interface with Microsoft Exchange); and
- Antivirus Capability.

3 CONFORMANCE CLAIMS (ASE_CCL)

3.1 CC conformance claim

The ST and TOE are conformant to version 3.1 (Revision 5) of the Common Criteria for Information Technology Security Evaluation.

- **Part 2 extended**. Conformant with Common Criteria for Information Technology Security Evaluation Part 2: Security functional requirements, version 3.1, Revision 5.
- **Part 3 conformant**. Conformant with Common Criteria for Information Technology Security Evaluation Part 3: Security assurance requirements, version 3.1, Revision 5.

3.2 Protection Profile conformance claim

N/A

4 SECURITY PROBLEM DEFINITION (ASE_SPD)

This section describes the nature of the security problem that the TOE is designed to address. The security problem is described through:

- a set of **threats** that the TOE must mitigate,
- specific **assumptions** about the security aspects of the environment (both IT related and non-IT related elements) in which the TOE will operate, and
- relevant **organisational security policies** that specify rules or guidelines that must be followed by the TOE and/or the operational environment.

4.1 Threats

Table 3 – Identified threats

Threat	Description
T.UNAUTHORISED_ACCESS	An unauthorised user or third party may gain unauthorised access to a device containing the TOE, allowing direct access to TOE data for export, modification or other purposes.
T.DENIAL_OF_SERVICE	A user may attempt to submit multiple malicious files in such a way that it leads to a denial of TOE service.
T.MALICIOUS_FILES	Malicious content contained within files may adversely affect the confidentiality, integrity and/or availability of files/services within the environment.
T.UNAUTHORISED_DOWNLOAD	An unauthorised user may access or download a file that was uploaded by another user of the TOE.

4.2 Assumptions

Table 4 – Assumptions

Assumption	Description
A.NO_EVIL	It is assumed that there will be one or more competent administrators assigned to manage the Votiro SDS server, its platform and the security of the information both of them contain. It is also assumed that the administrator(s) are not careless, wilfully negligent, nor hostile, and will follow and abide by the instructions provided by the administration documentation.
A.INSTALL	It is assumed that Votiro SDS is delivered, installed, configured and set up in accordance with documented delivery and installation/setup procedures.
A.PHYSICAL_PROTECT	It is assumed that Votiro SDS server and its associated platforms will be located within facilities providing controlled access to the TOE.
A.LOGICAL_PROTECT	The network environment in which the TOE is located ensures that only authorised users can are connect to it via its API. Thus the TOE is assumed to be deployed within a secure network environment.
A.TIME	The operational environment of the TOE will provide reliable time sources for use by the TOE.
A.CRYPTO	The operational environment of the TOE will provide Approved cryptographic functions for use by the TOE.

4.3 Organisational security policies

There are no organisational security policies for this Security Target.

5 SECURITY OBJECTIVES (ASE_OBJ)

5.1 Security objectives for the TOE

Table 5 – Security objectives for the TOE

Objective	Objective statement
O.AUDIT	The TOE shall generate audit data for security relevant events.
O.RESOURCE_MANAGMENT	The TOE shall manage resources in order to prevent over allocation of TOE capacity.
O. POLICY	The TOE shall process received files in accordance with its configured policy.
O.CDR	The TOE shall process files to reduce the level of active content.
O.ANONYMITY	The TOE shall not provide any method for users to determine whether files have been processed previously.
O.FAILSAFE	The TOE shall fail in such a way that it maintains the security of the TSF
O.FILE_RETRIEVAL	The TOE shall only allow a file to be retrieved by the original uploader.

5.2 Security objectives for the environment

Table 6 – Security objectives for the environment

Objective	Objective statement
OE.INSTALL	The operational environment shall ensure that the TOE is delivered, installed, configured and set up in accordance with documented delivery and installation/setup procedures. The operational environment shall enable the administrator to ensure that the platforms on which the TOE is installed on support the secure operation of the TOE.
OE.LOGICAL_SECURITY	The operational environment shall provide sufficient logical security (such as firewalls) to protect the TOE from external (and internal) unauthorised users.
OE.NO_EVIL	The operational environment shall provide one or more competent administrators assigned to manage the TOE, its platform and the security of the information both of them contain. The operational environment will ensure that the administrator(s) are not careless, wilfully negligent, nor hostile, and will follow and abide by the instructions provided by the administration documentation.
OE.PHYSICAL_SECURITY	Physical security mechanisms, commensurate with the value of the TOE and the data it contains, are provided by the environment to prevent unauthorised access to the TOE and TSF data.
OE.TIME	The operational environment of the TOE will provide reliable time sources for use by the TOE
OE.CRYPTO	The operational environment of the TOE will provide cryptographic functionality to the TOE.

6 EXTENDED COMPONENTS DEFINITION (ASE_ECD)

6.1 User Data Protection (FDP)

6.1.1 FDP_CDR_EXT.1 Content Disarm and Reconstruction

Hierarchical to: No other components.

Dependencies: FDP_IFF.1 Simple security attributes

Table 7 – FDP_CDR_EXT.1 SFRs.

FDP_CDR_EXT.1.1	The TSF shall be capable of deconstructing the following file types [**assignment: list of file types**]
FDP_CDR_EXT.1.2	The TSF shall be capable of performing the following actions on the deconstructed files **[*assignment: list of actions*]**
FDP_CDR_EXT.1.3	The TSF shall be capable of reconstructing files.
FDP_CDR_EXT.1.4	The TSF shall be capable of recursively deconstructing files.
FDP_CDR_EXT.1.5	The TSF shall be capable of identifying the true file type of deconstructed files.

This extended SFR sits within the FDP Class (User Data Protection) as it allows the TOE to provide an increased level of inspection to files that have been processed according to the information flow policy defined within FDP_IFF.1.

FDP_CDR_EXT.1 allows the TOE to extend its basic information flow policy by providing for the deconstruction of files for the detection of potentially malicious content. Once ‘disarmed’, these files are re-constructed and a clean version provided to the end user. TOEs claiming this SFR are also capable of recursively examining files within archives (to a defined level of recursion) and allows for the detection of the ‘true’ filetype of a received object based on internal attributes such as metadata.

This extended SFR has been defined because CC Part 2 does not contain any SFRs related to the deconstruction and inspection of files.

7 SECURITY FUNCTIONAL REQUIREMENTS (ASE_REQ)

7.1 Overview

This section defines the security requirements satisfied by the TOE. Each requirement has been extracted from version 3.1 (Rev 4) of the CC, Part 2 providing functional requirements and Part 3 providing assurance requirements.

Part 2 of the CC defines an approved set of operations that may be applied to security functional requirements. Following are the approved operations and the document conventions that are used within this ST to depict their application:

- Assignment: The assignment operation provides the ability to specify an identified parameter within a requirement. Assignments are depicted using italicised text and are surrounded by square brackets as follows [*assignment*].
- Assignment within a selection: Indicated with [*italicised and underlined text*]
- Selection: The selection operation allows the specification of one or more items from a list. Selections are depicted using bold text and are surrounded by square brackets as follows [**selection**].
- Refinement: The refinement operation allows the addition of extra detail to a requirement. Refinements are indicated using **bolded text** for additions and ~~strike-through~~ for deletions.
- Iteration: The iteration operation allows a component to be used more than once with varying operations. Iterations are depicted by placing a number at the end of the component identifier (e.g. FCS_COP.1(1) and FCS_COP.1(2)).

The security functional requirements are expressed using the notation stated above and are identified in the table below.

Table 8 – Security functional requirements

Identifier	Title
Audit (FAU)	
FAU_GEN.1	Audit data generation
User Data Protection (FDP)	
FDP_IFC.1	Subset information flow control
FDP_IFF.1	Simple security attributes
FDP_CDR_EXT.1	Content Deconstruction and Reconstruction
FDP_RIP.2	Full residual information protection
Security Management (FMT)	
FMT_MSA.1	Management of Security Attributes
FMT_MSA.3	Static attribute initialisation
FMT_SAE.1	Time-limited authorisation
FMT_SMF.1	Specification of Management Functions
FMT_SMR.1	Security Roles

Identifier	Title
Privacy (FPR)	
FPR_UNL.1	Unlinkability
Protection of the TSF (FPT)	
FPT_FLS.1	Failure with preservation of secure state
FPT_STM.1	Reliable time stamps
Resource utilisation (FRU)	
FRU_FLT.1	Degraded fault tolerance
FRU_RSA.1	Maximum quotas

7.2 Audit (FAU)

7.2.1 FAU_GEN.1 Audit data generation

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**] level of audit; and c) [*See **Table 9 – Auditable Events** for additional auditable events*].
FAU_GEN.1.2	The TSF shall record within each audit record at least the following information: Date and time of the event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event; and For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [*See **Table 9 – Auditable Events** for other audit information*].

Table 9 – Auditable Events

Auditable event	Additional Audit Information
Submission of File	Item ID, Filename, temporary storage filepath
Retrieval of File	Item ID, Filename
CDR processing	File processing policy name and details
Exception	Item ID, Filename
Deletion of old files.	olderThan deletion time
XML Error	Xml file, exception stack trace
Startup	N/A
Shutdown	N/A

7.3 User Data Protection (FDP)

7.3.1 FDP_IFC.1 Subset information flow control (File Flow Policy)

FDP_IFC.1.1	The TSF shall enforce the [*file flow policy*] on [*all files received by the TOE*].

7.3.2 FDP_IFF.1 Simple security attributes

FDP_IFF.1.1	The TSF shall enforce the [*file flow policy*] based on the following types of subject and information security attributes: [• *Subjects: File submitter;* • *Information: Submitted files; and* • *Operations: Allow, Deny, CDR*].
FDP_IFF.1.2	The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [• *if the policy is configured with an explicit Allow rule; or* • *or if the file is processed in accordance with FDP_CDR_EXT.1*].
FDP_IFF.1.3	The TSF shall enforce the [*no additional information flow control SFP rules*].
FDP_IFF.1.4	The TSF shall explicitly authorise an information flow based on the following rules: [*no explicit authorization rules*].
FDP_IFF.1.5	The TSF shall explicitly deny an information flow based on the following rules: [*if there is currently no policy for the submitted File Type*].

7.3.3 FDP_CDR_EXT.1 Content Deconstruction and Reconstruction

FDP_CDR_EXT.1.1	The TSF shall be capable of deconstructing the following file types: [***file types defined in* Table 10 – Supported file types and operations**]
FDP_CDR_EXT.1.2	The TSF shall be capable of performing the following actions on the deconstructed files: [***as defined in Table 10 – Supported file types and operations***]
FDP_CDR_EXT.1.3	The TSF shall be capable of reconstructing files.
FDP_CDR_EXT.1.4	The TSF shall be capable of recursively deconstructing files.
FDP_CDR_EXT.1.5	The TSF shall be capable of identifying the true file type of deconstructed files.

Table 10 – Supported file types and operations

File type	Supported operations
Microsoft Office	
doc (Word)	Recursive processing, Macro removal, ActiveX removal
docx (Word 2007+)	Recursive processing
xml (WordXML)	Recursive processing
xls,xlt (Excel)	Recursive processing, Macro removal, ActiveX removal
xlsx (Excel 2007+)	Recursive processing, ActiveX removal
xml (ExcelXML)	Rebuild
ppt, pps, pot (PowerPoint)	Recursive processing, Macro removal, ActiveX removal
pptx, ppsx (PowerPoint 2007+)	Recursive processing
potx (PowerPoint Template 2007+)	Recursive processing
sldx (PowerPoint Slide 2007+)	Recursive processing
dotx (Word Template 2007+)	Recursive processing, ActiveX removal
xltx (Excel Template 2007+)	Recursive processing, ActiveX removal
dot (Word Pre-2007 Template)	Recursive processing, Macro removal
Microsoft Office 2007+ with Macros	
xlsm (Excel with Macros)	Recursive processing, Macro removal, ActiveX removal
docm, dotm (Word with Macros)	Recursive processing, Macro removal, ActiveX removal
pptm, sldm (PowerPoint with Macros)	Recursive processing, Macro removal, ActiveX removal
Raster Image Files	
jpeg, jpg (JPEG)	Rebuild / Microchanges
gif (GIF)	Rebuild / Microchanges
tif (Tagged Image File)	Rebuild / Microchanges
bmp (Bitmap)	Rebuild / Microchanges
png (Portable Network Graphics)	Rebuild / Microchanges
Vector Image Files	
wmf (Windows Metafile)	Rebuild
emf (Enhanced Metafile)	Rebuild
Archives	
zip (ZIP Archive)	Recursive processing

File type	Supported operations
rar (RAR Archive)	Recursive processing
7z (7-Zip Archive)	Recursive processing
tar.gz (GZip-compressed Tar Archive)	Recursive processing
tar (Tar archive)	Recursive processing
RTF files	
Rtf (Rich Text Format)	Recursive processing
Adobe PDF	
Pdf (Portable Document Format)	Recursive processing, JavaScript removal

7.3.4 FDP_RIP.2 Full residual information protection

FDP_RIP.2.1	The TSF shall ensure that any previous information content of a resource is made unavailable upon the [**deallocation of the resource from**] all objects.

7.4 Security Management (FMT)

7.4.1 FMT_MSA.1 Management of Security Attributes

FMT_MSA.1.1	The TSF shall enforce the [*file flow policy]* to restrict the ability to [**modify**] the security attributes [*file flow policy rules*] to [*the administrator*].

7.4.2 FMT_MSA.3 Static attribute initialisation

FMT_MSA.3.1	The TSF shall enforce the [*file flow policy*] to provide [**restrictive**] default values for security attributes that are used to enforce the SFP.
FMT_MSA.3.2	The TSF shall allow the [*administrator*] to specify alternative initial values to override the default values when an object or information is created.

7.4.3 FMT_SAE.1 Time-limited authorisation

FMT_SAE.1.1	The TSF shall restrict the capability to specify an expiration time for [*non-retrieved files*] to [*administrators*].
FMT_SAE.1.2	For each of these security attributes, the TSF shall be able to [*delete files*] after the expiration time for the indicated security attribute has passed.

7.4.4 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1	The TSF shall be capable of performing the following management functions: [• *Configuration of XML files* • *Configuration of Windows services and file system*]

7.4.5 FMT_SMR.1 Security roles

FMT_SMR.1.1	The TSF shall maintain the roles [*administrator*].
FMT_SMR.1.2	The TSF shall be able to associate users with roles.

7.5 Privacy (FPR)

7.5.1 FPR_UNL.1 Unlinkability

FPR_UNL.1.1	The TSF shall ensure that [*administrators or file uploaders*] are unable to determine whether [*one or more file uploads*] [**were caused by the same user**].

7.6 Protection of the TSF (FPT)

7.6.1 FPT_FLS.1 Failure with preservation of secure state

FPT_FLS.1.1	The TSF shall preserve a secure state when the following types of failures occur: [*error in SDS engine, service halt, connection failure, subsystem error/exceptions/misconfiguration, policy misconfiguration or runtime errors*].

7.6.2 FPT_STM.1 Reliable time stamps

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

7.7 Resource utilisation (FRU)

7.7.1 FRU_FLT.1 Degraded fault tolerance

FRU_FLT.1.1	The TSF shall ensure the operation of [*file processing and policy enforcement*] when the following failures occur: [*error, exception or service fault*].

7.7.2 FRU_RSA.1 Maximum quotas

FRU_RSA.1.1	The TSF shall enforce maximum quotas of the following resources: [*concurrent file processing*] that [**subjects**] can use [**simultaneously**].

7.8 Security assurance requirements

Table 11 – Security assurance requirements

Assurance class	Assurance component
Security Target (ASE)	Conformance claims (ASE_CCL.1)
	Extended components definition (ASE_ECD.1)
	ST introduction (ASE_INT.1)
	Security objectives(ASE_OBJ.2)
	Derived security requirements (ASE_REQ.2)
	Security Problem Definition (ASE_SPD.1)
	TOE summary specification (ASE_TSS.1)
Development (ADV)	Security-enforcing functional specification (ADV_FSP.2)
	Basic design (ADV_TDS.1)
	Security architecture description (ADV_ARC.1)
Guidance documents (AGD)	Operational user guidance (AGD_OPE.1)
	Preparative procedures (AGD_PRE.1)
Life cycle support (ALC)	Use of a CM system (ALC_CMC.2)
	Parts of the TOE CM coverage (ALC_CMS.2)
	Delivery procedures (ALC_DEL.1)

Tests (ATE)	Independent testing – sample (ATE_IND.2)
	Evidence of coverage (ATE_COV.1)
	Functional testing (ATE_FUN.1)
Vulnerability assessment (AVA)	Vulnerability analysis (AVA_VAN.2)

8 TOE SUMMARY SPECIFICATION (ASE_TSS)

8.1 Overview

Votiro SDS is designed to supplement the traditional sandboxing approach for securing files entering a network. The TOE allows users to submit files to it via API calls which are then processed according to configurable policy rules.

The TOE uses these policy rules to sort files before applying one of several actions to it. These rules can be based on file type or digital signature status (e.g. only allow signed PDF files). SDS-supported files are passed onto the file specific sanitizer engine to be deconstructed by the TOE in order to detect and remove harmful content embedded within the files. The TOE then reconstructs the files in such a way as to maintain the end user experience whilst removing the possibly harmful content.

Sanitization actions that can be implemented include removing macros, metadata, printer settings from office files and adding “noise” or microchanges to raster image files in order to disrupt embedded code segments.

Non SDS-supported files can be filtered in a manner similar to a standard firewall with configurable rules to allow or deny each file type.

Once the file is reconstructed in a safe manner the submitter can then use another API call to retrieve the safe version of the file.

The TOE is configured entirely by the use of .XML configuration files. These allow the administrator to configure the general TOE settings as well as the specific file filtering rules and sanitizer actions.

8.2 Audit

The TOE is capable of generating audit logs for actions of the TOE. This includes changes to the configuration of the TOE and for each file processing action including submission and retrieval of files.

The file processing steps are identified by the filename and the unique identification number assigned to each transaction. The TOE generates its own timestamps which are used during the generation of audit logs. These timestamps are generated using the time source of the underlying operating system.

For a full list of auditable events and additional recorded audit information see Table 9 – Auditable Events.

8.3 User Data Protection

The TOE allows users to submit files to it via API calls which are then processed according to configurable policy rules.

Prior to processing the TOE inspects files to ensure the true file type is correctly identified. This identification relies upon the inspection of metadata and file payload in order to correctly identify file types.

SDS supported files are passed onto the file specific sanitizer engine to be deconstructed by the TOE in order to detect and remove harmful content embedded within the files. The TOE then reconstructs the files in such a way as to maintain the end user experience whilst removing the possibly harmful content.

Non SDS supported files can be filtered in a manner similar to a standard firewall with configurable rules to allow or deny each file type.

Once the file is reconstructed in a safe manner the submitter can then use another API call to retrieve the safe version of the file.

Table 10 – Supported file types and operations describes the file types and sanitisation actions are supported by the TOE.

All submitted files are deleted after a configurable time limit.

The TOE is capable of recursively processing files. For example it can unpack and process files contained within an archive or files that are embedded within other documents. The TOE can be configured with either a hard recursion limit which would cause it to reject anything nested over a certain limit or with a processing timelimit in order to prevent the use of ZipBomb files as a denial of service attack against the TOE.

The TOE is also capable of inspecting file protected archives and PDFs if provided with a password and can repack and password protect the file after processing.

Raster files, can be compressed as part of the SDS process to reduce file size. NoiseLevel is another option in raster image sanitisation that allows the TOE to sanitise the data portion of an image. This involves the introduction of random “noise” or microchanges to the image body in order to disrupt any embedded content. Noise affects the file size but this is an adequate trade off in order to disarm embedded shellcode.

The PDF sanitizer allows for the “flattening” of files to remove uneeded active content though this can have some impact on PDFs with form fields. The TOE is also capable of verifying certificates with the option of skipping the processing of files signed by a trusted source.The certificate verification process is handled by the underlying operational environment.

The office file sanitizer can remove remove active content such as macros, metadata, printersettings, ActiveX and embedded fonts.

If a specific file type or content is blocked entirely the TOE is capable of returning a blocked file report instead of a processed file which contains the reason why the file was blocked.

8.4 Management

Configuration of the TOE including general security configuration and file flow policy is conducted via .XML files. Individual file flow policies defined via XML can be called individually at an API level.

Malformed XML configuration files will cause a runtime exception.

These files can only configured by the administrator of the TOE. As the TOE itself does not maintain user roles it relies on the underlying operation system to manage authentication and identification of users.

It is assumed that the operator of the underlying OS is the authorised administrator of the TOE and as such any configuration changes are attributed to this user. The administrator is also cabable of making changes to or manipulating the underlying OS as needed. For example using windows services to restart the SDS service in order to apply changes from XML files.

The administrator is capable of specifying a lifetime for non-downloaded files and will delete them if they are not retrieved by the submitter within this time period.

8.5 Privacy

Votiro SDS protects the privacy of its file inspection transactions by assigning each transaction a unique twenty character randomly-generated identifier which is used by the submitter to poll the server for files processing status and, once processing is completed, retrieve the file from the server. The TOE utilises the underlying OS random bit generator for generation of these identifiers.

Once a file has been retrieved from the server it is immediately earmarked for deletion preventing it from been downloaded again without resubmitting it. This is done in order to ensure no residual information is retained in the event that Votiro SDS server is compromised.

In the case of password protected archives the passwords are stored in memory after the submission of the API request and deleted once no longer needed.

As the ID assigned to the file transaction is uniquely generated and not tied in any way to the properties of the actual file it is impossible to determine if a file has previously been submitted to the TOE.

8.6 Protection of the TSF

The TOE will never provide a file as safe if it has not been processed by the relevant SDS/CDR systems. In the event of an error in CDR engine, service halt, connection failure, subsystem error/exceptions/misconfiguration, policy misconfiguration or runtime error the TOE will fail in such a way that no unprocessed files are presented to the users of the system.

The TOE generates its own timestamps which are used during the generation of audit logs. These timestamps are generated using the time source of the underlying operating system.

8.7 Resource utilisation

The SDS system limits the amount of concurrent files that can be processed to an administrator configurable amount. This is done to ensure that the hardware resources are not over utilised. In the even that more files are submitted than can be concurrently processed excess files are placed into a queue in order to be processed once resources are available.

In the event of an error, exception or service fault occurring, the service will attempt to restart itself. If the same error occurs after a service restart the TOE will attempt another service restart. If a third occurrence of the error occurs the TOE will request a reboot of the underlying operating system in an attempt to fix the error.

9 RATIONALE

9.1 Security objectives rationale

9.1.1 Threat/Objectives Mapping

The following table provides a mapping of threats to objectives and adequate justification for the mapping.

Table 12 – Threats to objectives mapping

Threat	Objective	Justification
T.UNAUTHORISED_ACCESS	O.AUDIT O.ANONYMITY	An unauthorised user or third party may gain unauthorised access to a device containing the TOE, allowing direct access to TOE data for export, modification or other purposes. Audit: Allows for the tracking and determination of TOE initialisation/restart. Anonymity: Actions the deletion of previously processed data to remove any residual information that may be gained.
T.DENIAL_OF_SERVICE	O.RESOURCE_MANAGMENT O.FAILSAFE	A user may attempt to submit multiple malicious files in such a way that it leads to a denial of TOE service. Resource Management: Prevents excess files submissions from overloading TOE systems. Failsafe: In the event that TOE systems are compromised it will fail in such a way that it does not allow the transmission of unsafe files.
T.MALICIOUS_FILES	O.POLICY O.CDR O.FAILSAFE	Malicious files may compromise the environment of the TOE or the system it is protecting. Policy: Allows for the processing of receive files in accordance with its configuration. CDR: Allows the TOE to disarm potentially malicious content. Failsafe: In the event that TOE systems are compromised it will fail in such a way that it does not allow the transmission of unsafe files.
T.UNAUTHORISED_DOWNLOAD	O.FILE_RETRIEVAL O.ANONYMITY	An unauthorised user may access or download a file that was uploaded by another user of the TOE. File_Retreival: Prevents the accidental or malicious download of files uploaded by other users. Anonymity: Prevents attribution of currently in processing files and deletion of previously processed data removes any residual information that may be gained.

9.1.2 Objectives/SFRs Mapping

The following table provides a mapping of Objectives to SFRs and adequate justification for the mapping.

Table 13 – SFR to objectives mapping

Objective	SFR	Justification
O.AUDIT	FAU_GEN.1 FPT_STM.1	FAU_GEN.1 – This TOE generates audit logs. FPT_STM.1 – This TOE provides accurate timestamps to the generated audit logs.
O.RESOURCE_MANAGMENT	FRU_RSA.1 FMT_SMF.1	FRU_RSA.1 – The TOE allows the configuration of Maximun concurrent processing quotas. FMT_SMF.1 – The TOE can be configured via XML files to change the quotas.
O. POLICY	FDP_IFC.1 FDP_IFF.1 FMT_MSA.1 FMT_SMR.1 FMT_MSA.3 FMT_SMF.1	FDP_IFC.1 – The TOE enforces the File Flow Policy FDP_IFF.1 – The TOE enforces the policy based on set criteria and configuration. FMT_MSA.1 – The TOE enforces the policy by default and this can be edited by the administrator. FMT_SMR.1 – The TOE provides an administrator role which is enforced by the underlying OS FMT_MSA.3 – The TOE enforces santisation of files by default and these settings can be changed by the administrator. FMT_SMF.1 – The TOE policy can be configured via XML files.
O.CDR	FDP_CDR_EXT.1	FDP_CDR_EXT.1 – This SFR defines how the CDR engine processes submitted files
O.ANONYMITY	FPR_UNL.1 FMT_SAE.1	FPR_UNL.1 – The TOE anonymises file submitters. FMT_SAE.1 – The TOE deletes residual information after a timelimit is reached.
O.FAILSAFE	FPT_FLS.1 FRU_FLT.1	FPT_FLS.1 – The TOE will preserve a secure state in the event of an error. FRU_FLT.1 – The TOE will ensure that the File Flow Policy continues to be enforced in the event of an error.
O.FILE_RETRIEVAL	FDP_RIP.2 FMT_SAE.1	FDP_RIP.2 – The TOE deletes all residual information after processing. FMT_SAE.1 – The TOE deletes residual information after a timelimit is reached.

9.1.3 Assumptions to OE Rationale

The following table provides a mapping of the security objectives for the environment and their relevant assumptions, as well as a justification for the mapping.

Table 14 – Assumptions to objectives mapping

Assumption	Objective	Justification
A.NO_EVIL	OE.NO_EVIL	The objective is a direct instantiation of the assumption and is therefore met.
A.INSTALL	OE.INSTALL	The objective is a direct instantiation of the assumption and is therefore met.
A.PHYSICAL_PROTECT	OE.PHYSICAL_SECURITY	The objective is a direct instantiation of the assumption and is therefore met.
A.LOGICAL_PROTECT	OE.LOGICAL_PROTECT	The objective is a direct instantiation of the assumption and is therefore met.
A.TIME	OE.TIME	The objective is a direct instantiation of the assumption and is therefore met.

A.CRYPTO	OE.CRYPTO	The objective is a direct instantiation of the assumption and is therefore met.

9.2 Security Requirements Rationale

9.2.1 Dependency analysis

The following table provides a dependency analysis for the SFRs chosen within this security target. For SFRs that have not been included a rationale is provided after the table.

SFR	Dependencies	Dependency Met?
FAU_GEN.1	FPT_STM.1 Reliable time stamps	Yes
FDP_IFC.1	FDP_IFF.1 Simple security attributes	Yes
FDP_IFF.1	FDP_IFC.1 Subset information flow control FMT_MSA.3 Static attribute initialisation	Yes
FDP_CDR_EXT.1	FDP_IFF.1 Simple security attributes	Yes
FDP_RIP.2	N/A	Yes
FMT_MSA.1	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control] FMT_SMR.1 Security roles FMT_SMF.1 Specification of Management Functions	Yes
FMT_MSA.3	FMT_MSA.1 Management of security attributes FMT_SMR.1 Security role	Yes
FMT_SAE.1	FMT_SMR.1 Security roles FPT_STM.1 Reliable time stamps	Yes
FMT_SMF.1	N/A	Yes
FMT_SMR.1	FIA_UID.1 Timing of identification	Implicitly met by environment – user access control is managed by the underlying operational environment using Windows user permissions.
FPR_UNL.1	N/A	Yes
FPT_FLS.1	N/A	Yes
FPT_STM.1	N/A	Yes
FRU_FLT.1	FPT_FLS.1 Failure with preservation of secure state	Yes
FRU_RSA.1	N/A	Yes

9.3 Security assurance requirements justification

The assurance package for the evaluation is Evaluation Assurance Level 2 (EAL2).

EAL2 assurance requirements provide confidence in the security functionality of the TOE by analysis using a functional and interface specification, guidance documentation and the high-level design of the TOE, to understand the security behaviour.

The analysis is supported by independent testing of the TOE security functions, evidence of developer testing based on the functional specification, selective independent confirmation of the developer test results, strength of function analysis, and evidence of a developer search for obvious vulnerabilities.

EAL2 also provides assurance through a configuration list for the TOE, and evidence of secure delivery procedures.

--- END OF DOCUMENT ---