Wickr Enterprise Server 1.30.0
Security Target

Version 1.0
28 March 2023

Prepared for:

Wickr
W 31st Street
New York, NY 10001

Prepared by:

Accredited Testing and Evaluation Labs
6841 Benjamin Franklin Drive
Columbia, MD 21046

Contents

1. Security Target Introduction

This section identifies the Security Target (ST) and Target of Evaluation (TOE) identification, ST conventions, ST conformance claims, and the ST organization. The TOE is Wickr Enterprise Server 1.30.0.

Wickr Enterprise Server is a software application for Ubuntu systems. It is installed on a server system in an organization to facilitate communication between Wickr Enterprise Client peers. The Wickr Enterprise Server interacts with the Wickr Enterprise Clients that are used for end user communication.

The focus of this evaluation is on the TOE functionality supporting the claims of version 1.4 of the Protection Profile for Application Software [App PP]. The only capabilities covered by the evaluation are those specified in the aforementioned Protection Profile; no additional security functional claims are made by this Security Target. The security functionality specified in [App PP] includes protection of security-relevant data at rest and in transit, any cryptographic functionality used to achieve this, and security of the interactions between the application(s) and their underlying platform(s). Where appropriate and permitted by the [App PP], this evaluation will identify areas where the TOE's underlying platform is used to support the TOE's implementation of its claimed security functionality.

The Security Target contains the following additional sections:

- Product and TOE Description (Section 2)
- Security Problem Definition (Section 3)
- Security Objectives (Section 4)
- IT Security Requirements (Section 5)
- TOE Summary Specification (Section 6)
- Protection Profile Claims (Section 7)
- Rationale (Section 8)

1.1 Security Target, TOE and CC Identification

ST Title – Wickr Enterprise Server 1.30.0 Security Target

ST Version – Version 1.0

ST Date – 28 March 2023

TOE Identification – Wickr Enterprise Server 1.30.0. The platform-specific versions of the TOE include:

1. Wickr Enterprise Server for Linux 1.30.0.

   Evaluated on Ubuntu 18.04.

The TOE runs on the platform OS as a containerized application in Docker (runtime engine 20.10) with an Amazon Linux 2 container image.

TOE Developer – Wickr, Inc.

Evaluation Sponsor – Wickr, Inc.

CC Identification – *Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 5, April 2017*

1.2 Conformance Claims

This ST and the TOE it describes are conformant to the following CC specifications: This ST is conformant to:

- *Protection Profile for Application Software, Version 1.4, 07 October 2021*
- The following NIAP Technical Decisions apply to this PP and Functional Package and have been accounted for in the ST development and the conduct of the evaluation, or were considered to be non-applicable:

Table 1 Technical Decisions

TD #	TD Title	Applicability to Evaluation
0719	ECD for PP APP V1.3 and 1.4	Not applicable; this TD updates the App PP to include a formal ECD which is needed for the PP itself to conform to CC Part 3. This does not change the ST or how the evaluation of the TOE is conducted.
0717	Format changes for PP_APP_V1.4	Applicable
0709	Number of elements for iterations of FCS_HTTPS_EXT.1	Not applicable; the TOE does not claim this SFR.
0669	FIA_X509_EXT.1 Test 4 Interpretation	Not applicable; the TOE does not claim this SFR.
0664	Testing activity for FPT_TUD_EXT.2.2	Applicable
0655	Mutual authentication in FTP_DIT_EXT.1 for SW App	Applicable
0650	Conformance claim sections updated to allow for MOD_VPNC_V2.3 and 2.4	Not applicable: the TOE does not claim VPN client functionality
0628	Addition of Container Image to Package Format	Applicable
0624	Addition of DataStore for Storing and Setting Configuration Options	Applicable

- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 5, April 2017.
  - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1 Revision 5, April 2017.
  - Part 3 Extended

1.3 Conventions

The following conventions have been applied in this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
  - Iteration: allows a component to be used more than once with varying operations. This ST does not iterate any SFRs so the convention is not used.
  - Assignment: allows the specification of an identified parameter. Assignments are indicated using italics and are surrounded by brackets (e.g., [*assignment item*]). Note that an assignment within a selection would be identified in both italics and underline, with the brackets themselves underlined since they are explicitly part of the selection text, unlike the brackets around the selection itself (e.g., [selection item, [*assignment item inside selection*]]).
  - Selection: allows the specification of one or more elements from a list. Selections are indicated using underlines and are surrounded by brackets (e.g., [selection item]).
  - Refinement: allows the addition of details and non-technical changes to grammar and formatting. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… ~~some~~ **big** things …"). Note that minor grammatical changes that do not involve the addition or removal of entire words (e.g., for consistency of quantity such as changing "meets" to "meet") do not have formatting applied.

- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.
- The ST does not highlight operations that have been completed by the PP authors, though it does preserve brackets to show where operations have been made.

1.3.1 Terminology

Room	A chat available to users on connected clients

1.3.2 Acronyms

API	Application Programming Interface
ASLR	Address Space Layout Randomization
AES	Advanced Encryption Standard
CAVP	Cryptographic Algorithm Validation Program
CC	Common Criteria for Information Technology Security Evaluation
CEM	Common Evaluation Methodology for Information Technology Security
CRL	Certificate Revocation List
FIPS	Federal Information Processing Standard
HMAC	Hashed Message Authentication Code
HTTP(S)	Hypertext Transfer Protocol (Secure)
IP	Internet Protocol
NIAP	National Information Assurance Partnership
NIST	National Institute of Standards and Technology
OCSP	Online Certificate Status Protocol
PII	Publicly Identifiable Information
PP	Protection Profile
RSA	Rivest, Shamir and Adleman (algorithm for public-key cryptography)
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
SSL	Secure Socket Layer Protocol
ST	Security Target
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functions
TSS	TOE Security Specification

2. Product and TOE Description

The TOE is the Wickr Enterprise Server 1.30.0 product. This section provides an overview of the capabilities of the product and then proceeds to describe the TOE itself in terms of its evaluated components and functional claims.

2.1 Introduction

Wickr Enterprise Server is an on-premise application providing communication with Wickr Enterprise Clients.

Wickr Enterprise Server is part of a client-server distribution. The TOE is the server portion of this distribution. It interacts with the Wickr Enterprise Client application in its operational environment. Collectively, they make up the Wickr Enterprise solution.

2.2 Product Overview

This sub-section describes capabilities of Wickr Enterprise. The scope of the evaluation is covered in the subsequent sub-sections that provide the TOE overview and describe the TOE architecture and physical and logical boundaries.

Wickr Enterprise is an end-to-end encrypted service that provides communication services for client devices in a closed-loop, zero-trust environment.

Wickr Enterprise is a client-server implementation. Users join the Wickr Enterprise Network by downloading a free copy of their platform's Wickr Client application. Users subscribe to the Enterprise Network by paying a monthly subscription fee. Upon establishing a subscription, the User's Wickr Client receives configuration information from a Wickr Server, enabling connection to the Wickr Enterprise Network. Wickr Servers are part of the infrastructure supporting the Wickr Enterprise Network. They exchange user information and routing information to enable messages to traverse the network from Server to Server until the final Client is reached.

All Wickr Clients communicate through Wickr Servers for client-to-client communication. The 'base' configuration of the Wickr Server runs as a Messaging Server. Wickr Client to Client communication is actually Wickr User/Wickr Client to Wickr User/Wickr Client communication. A registered Wickr User may be registered on multiple platforms (Client instances) and a Wickr Server may have multiple Wickr Users registered on its system. The Wickr Messaging Server is responsible for authenticating Wickr Users and discovering routing information. On receipt of a connection request, once authenticated, the Messaging Server discovers information about the recipient(s) by configuration information.

A data message will be sent to each instance of the registered User. Additionally, a copy of the data message will be sent to the sender's additional User accounts to enable the sender to have a copy of the message as "sent" on each client. So, if a sending User that is registered on two different platforms attempts to connect to a User that is also registered on two different platforms, the Messaging Server will create the message to be routed to three different User/Client destinations. If a recipient client's system is powered off, data messages are held for that User/Client until that instance of the User powers on. If a recipient client's system is powered on but the User is not logged onto the Enterprise Network, the Messaging Server will provide a push notification to notify the User a data message is available.

Administration of the Wickr Enterprise Network is by Web access to the Messaging Server. The Messaging Server sends configuration information to Wickr Clients. Additionally, Wickr Users may manage parameters about their Wickr Client accounts.

2.3 TOE Overview

The Target of Evaluation (TOE) comprises the Wickr Enterprise Server 1.30.0 software application (hereinafter referred to as Wickr Server). The TOE is deployed within a Docker container on a Linux platform.

The focus of this evaluation is on the TOE functionality supporting the claims in the *Protection Profile for Application Software, Version 1.4*. Specifically, the following capabilities are within the scope of the evaluation:

- Trusted communications between Wickr Client and Wickr Servers, as well as with remote peers.
- The extent to which the TSF relies on platform-provided and third-party capabilities to perform its functionality.

- The extent to which data used to determine the behavior of the TSF is secured while at rest and in transit.
- The ability for the TOE to function on host platforms that are configured for secure operation.
- The ability of the TOE to interface with the low-level components of its host platforms in such a manner that the TOE cannot be used as an attack vector to exploit the host platforms.
- The ability of the organization deploying the TOE to perform timely and trusted security updates to it.

The TSF includes all security data and configuration settings needed to support this behavior. Not all configuration settings are security-relevant.

The Wickr Client uses the Wickr Server to broker communications with other Wickr Clients. The Wickr Server relies on platform-provided cryptography for the establishment of trusted channels.

The TOE is administered via a remote web interface. Administrator accounts are locally defined. This interface uses platform-provided TLS/HTTPS.

The TOE boundary includes nginx for routing and load balancing for network traffic.

2.4 TOE Architecture

The Wickr Server TOE is a containerized software application that runs on Ubuntu 18.04.

Note that Ubuntu 18.04 reaches End of Standard Support on May 31, 2023. To ensure that the security posture of the operational environment can be maintained past this date, it is necessary for the organization deploying the TOE to have an Ubuntu Pro subscription, which includes Expanded Security Maintenance (ESM) for Ubuntu 18.04 until April 2028.

The product architecture is shown in the following figure. The Wickr Server application (the TOE) is indicated by the red box. The application runs in a containerized Docker format, which is indicated by the yellow box. The Docker infrastructure then runs on the Linux server platform.

The environmental Wickr Clients are used for messaging communications.

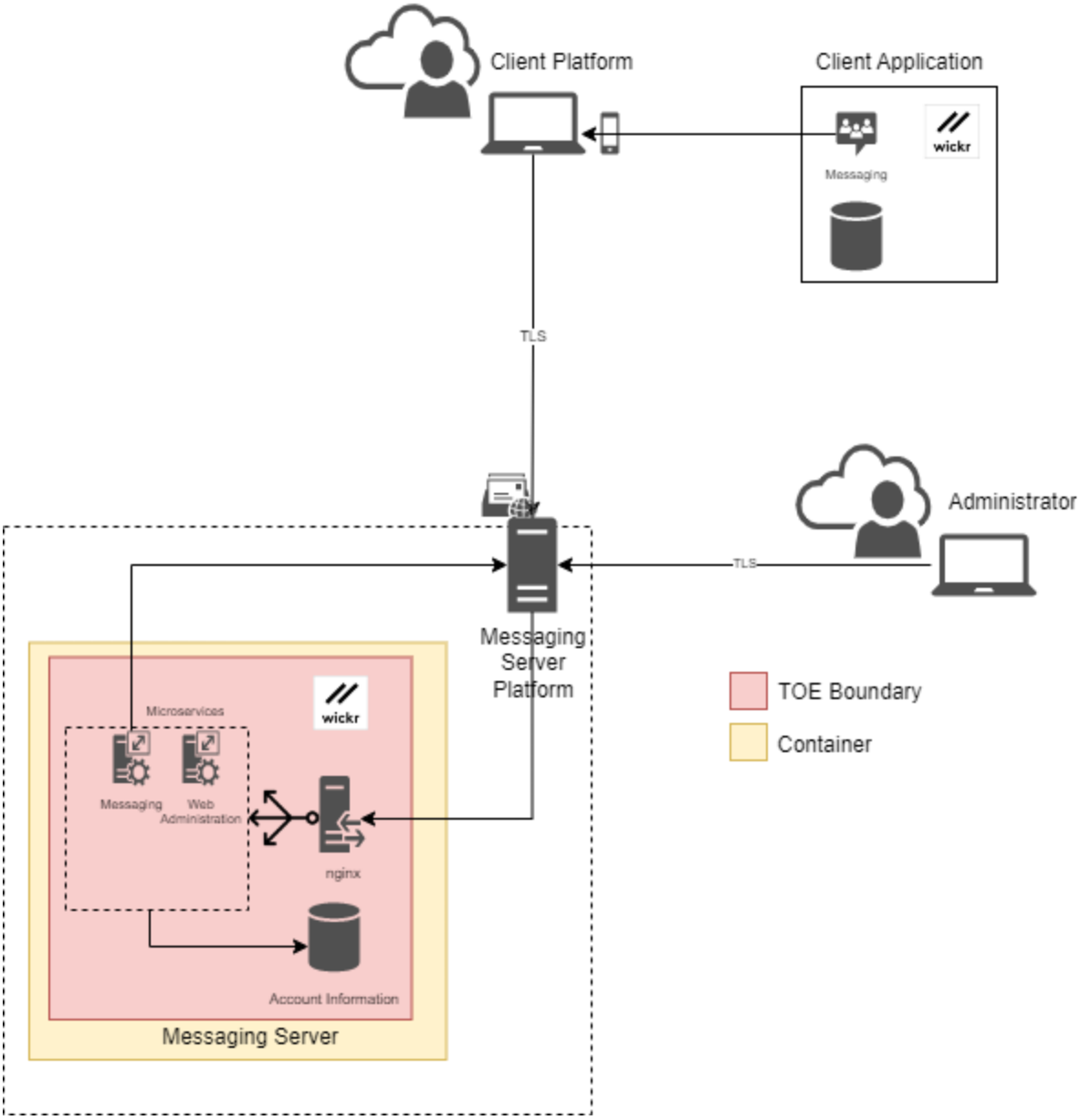

Figure 1: Wickr Architecture

2.4.1 Physical Boundary

The TOE includes the Wickr Server in a base deployment that provides data messaging. The TOE includes an nginx service which functions as a reverse proxy for routing inbound network connections. This service is considered non-interfering with respect to security as it does not enforce any of the security functionality claimed by the TSF. The TOE runs on Linux platforms. For this evaluation, the TOE is evaluated on the following specific platform:

- Linux
  - Intel Xeon E5-2620v3 (Haswell) processor
  - Ubuntu 18.04 LTS 64-bit OS

In addition to the platforms identified above, the TOE's operational environment includes the following:

- Two or more remote Wickr Client instances to establish connections

- A workstation with a browser to access the TOE's administrator interface
- Docker is required to run the TOE
- Update server (public download site)

2.4.2 Logical Boundary

This section summarizes the security functions provided by the TOE:

- Cryptographic Support
- User Data Protection
- Security Management
- Privacy
- Protection of the TSF
- Trusted Path/Channels

2.4.2.1 Cryptographic Support

The TOE uses NIST-validated cryptographic algorithms to secure messaging data in transit. The cryptographic functions are supplied by the host platform. Credential data is protected by a platform-provided mechanism.

2.4.2.2 User Data Protection

The TOE leverages platform functionality to secure sensitive data at rest. The TOE uses network resources provided by the underlying platforms.

The TOE uses network connectivity to interact with Wickr Clients and for administrator sessions.

2.4.2.3 Security Management

The TOE provides management capability for environmental components via a web interface. Administrator accounts are defined locally. Wickr Server configuration data is stored locally but is not managed through the TOE.

2.4.2.4 Privacy

The TOE does not process any PII. No transmission of PII occurs that is not in direct response to user activity.

2.4.2.5 Protection of the TSF

The TOE includes measures to integrate securely with its Linux platform. The TOE does not perform explicit memory mapping, nor does it allocate any memory region with both write and execute permissions. Similarly, the TOE does not write user-modifiable data to directories that contain executable files. The TOE is compatible with its supported host OS platform when it is configured in a secure manner. The TOE includes C code that is compiled to enforce ASLR and to protect against stack overflow as well as interpreted code that enforces ASLR through its runtime environment and is not susceptible to stack-based buffer overflow attacks.

The TOE uses a well-defined set of platform APIs and third-party libraries.

The TOE provides the ability for a user to check its version. The TOE platform is used to apply updates. Updates are delivered as a container image. Updates to the TOE are digitally signed, and the signature is validated by the platform prior to installation. The TOE does not modify its own code. Removal of the application removes all executable code associated with the TOE.

2.4.2.6 Trusted Path/Channels

The TOE uses trusted paths to secure data in transit between itself and external entities using platform-provided mechanisms. The TOE uses platform provided TLS and HTTPS for service requests, data communication, and web administration.

2.5 TOE Documentation

Wickr provides the following product documentation in support of the installation and secure use of the TOE:

- Wickr Enterprise NIAP Version Installation and Maintenance v1.30.0
- Wickr Enterprise Administrator Guide Version 426151b

3. Security Problem Definition

This Security Target includes by reference the Security Problem Definition, composed of threats and assumptions, from the [App PP]. The Common Criteria also provides for organizational security policies to be part of a security problem definition, but no such policies are defined in the [App PP].

In general, the threat model of the [App PP] is designed to protect against the following:

- Disclosure of sensitive data at rest or in transit that the user has a reasonable expectation of security for
- Excessive or poorly-implemented interfaces with the underlying platform that allow an application to be used as an intrusion point to a system

This threat model is applicable to the TOE because it processes information that may contain sensitive data that a user expects will not be disclosed to anyone other than the remote peer, and because the TOE runs on general purpose operating systems that may contain other data, applications, or network services that enforce their security in part through the assumption that the underlying operating system is trusted.

4. Security Objectives

Like the Security Problem Definition, this Security Target includes by reference the security objectives defined in [App PP]. This includes security objectives for the TOE (used to mitigate threats) and for its operational environment (used to satisfy assumptions).

5. IT Security Requirements

This section defines the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) that serve to represent the security functional claims for the Target of Evaluation (TOE) and to scope the evaluation effort.

The SFRs have all been drawn from the following Protection Profile (PP):

- *Protection Profile for Application Software*, *version 1.4, 07 October 2021* [App PP]

As a result, any selection/assignment/refinement operations already performed by that PP on the claimed SFRs are not identified here (i.e., they are not formatted in accordance with the conventions specified in section 1.3 of this Security Target). Formatting conventions are only applied on SFR text that was chosen at the ST author's discretion.

5.1 Extended Requirements

All of the extended requirements in this ST have been drawn from the [App PP]. The PP defines the following extended SAR and SFRs; since they have not been redefined in this ST, the [App PP] should be consulted for more information regarding these extensions to CC Parts 2 and 3.

- ALC_TSU_EXT.1: Timely Security Updates
- FCS_CKM_EXT.1: Cryptographic Key Generation
- FCS_RBG_EXT.1: Random Bit Generation Services
- FCS_STO_EXT.1: Storage of Credentials
- FDP_DAR_EXT.1: Encryption of Sensitive Application Data
- FDP_DEC_EXT.1: Access to Platform Resources
- FDP_NET_EXT.1: Network Communications
- FMT_CFG_EXT.1: Secure by Default Configuration
- FMT_MEC_EXT.1: Supported Configuration Mechanism
- FPR_ANO_EXT.1: User Consent for Transmission of Personally Identifiable Information
- FPT_AEX_EXT.1: Anti-Exploitation Capabilities
- FPT_API_EXT.1: Use of Supported Services and APIs
- FPT_IDV_EXT.1: Software Identification and Versions
- FPT_LIB_EXT.1: Use of Third Party Libraries
- FPT_TUD_EXT.1: Integrity for Installation and Update
- FPT_TUD_EXT.2: Integrity for Installation and Update
- FTP_DIT_EXT.1: Protection of Data in Transit

5.2 TOE Security Functional Requirements

The following table identifies the SFRs that are satisfied by the TOE.

Table 2 TOE Security Functional Components

Requirement Class	Requirement Component
FCS: Cryptographic Support	FCS_CKM_EXT.1 Cryptographic Key Generation Services
	FCS_RBG_EXT.1 Random Bit Generation Services
	FCS_STO_EXT.1 Storage of Credentials
FDP: User Data Protection	FDP_DAR_EXT.1 Encryption of Sensitive Application Data
	FDP_DEC_EXT.1 Access to Platform Resources
	FDP_NET_EXT.1 Network Communications
FMT: Security Management	FMT_CFG_EXT.1 Secure by Default Configuration
	FMT_MEC_EXT.1 Supported Configuration Mechanism
	FMT_SMF.1 Specification of Management Functions
FPR: Privacy	FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information
FPT: Protection of the TSF	FPT_AEX_EXT.1 Anti-Exploitation Capabilities
	FPT_API_EXT.1 Use of Supported Services and APIs
	FPT_IDV_EXT.1 Software Identification and Versions
	FPT_LIB_EXT.1 Use of Third Party Libraries
	FPT_TUD_EXT.1 Integrity for Installation and Update
	FPT_TUD_EXT.2 Integrity for Installation and Update
FTP: Trusted Path/Channels	FTP_DIT_EXT.1 Protection of Data in Transit

5.2.1 Cryptographic Support (FCS)

FCS_CKM_EXT.1	**Cryptographic Key Generation Services**[1]
FCS_CKM_EXT.1.1	The application shall [• generate no asymmetric cryptographic keys].

FCS_RBG_EXT.1	**Random Bit Generation Services**
FCS_RBG_EXT.1.1	The application shall [use no DRBG functionality] for its cryptographic functions.

FCS_STO_EXT.1	**Storage of Credentials**
FCS_STO_EXT.1.1	The application shall [

[1] Modified by TD0717

- invoke the functionality provided by the platform to securely store [*administrator credentials, server private key*]

] to non-volatile memory.

5.2.2 User Data Protection (FDP)

FDP_DAR_EXT.1	**Encryption of Sensitive Application Data**
FDP_DAR_EXT.1.1	The application shall [protect sensitive data in accordance with FCS_STO_EXT.1] in non-volatile memory.

FDP_DEC_EXT.1	**Access to Platform Resources**
FDP_DEC_EXT.1.1	The application shall restrict its access to [network connectivity].
FDP_DEC_EXT.1.2	The application shall restrict its access to [system logs].

FDP_NET_EXT.1	**Network Communications**
FDP_NET_EXT.1.1	The application shall restrict network communication to [• respond to [*communication requests from Wickr Clients, communication request for web administration, user lookups, bot enterprise service, installation UI, web socket messaging, OIDC services, Replicated services, push device service*], • *application-initiated communication of push notifications to mobile clients*].

5.2.3 Security Management (FMT)

FMT_CFG_EXT.1	**Secure by Default Configuration**
FMT_CFG_EXT.1.1	The application shall only provide enough functionality to set new credentials when configured with default credentials or no credentials.
FMT_CFG_EXT.1.2	The application shall be configured by default with file permissions which protect the application's binaries and data files from modification by normal unprivileged users.

FMT_MEC_EXT.1	**Supported Configuration Mechanism**
FMT_MEC_EXT.1.1	The application shall [invoke the mechanisms recommended by the platform vendor for storing and setting configuration options].

FMT_SMF.1	**Specification of Management Functions**
FMT_SMF.1.1	The TSF shall be capable of performing the following management functions [[• *Configure the messaging proxy* • *Manage users* • *Configure certificates* • *Configure room management* • *Configure event logging*]].

5.2.4 Privacy (FPR)

FPR_ANO_EXT.1	User Consent for Transmission of Personally Identifiable Information
FPR_ANO_EXT.1.1	The application shall [not transmit PII over a network].

5.2.5 Protection of the TSF (FPT)

FPT_AEX_EXT.1	Anti-Exploitation Capabilities
FPT_AEX_EXT.1.1	The application shall not request to map memory at an explicit address except for [*no exceptions*].
FPT_AEX_EXT.1.2	The application shall [not allocate any memory region with both write and execute permissions].
FPT_AEX_EXT.1.3	The application shall be compatible with security features provided by the platform vendor.
FPT_AEX_EXT.1.4	The application shall not write user-modifiable files to directories that contain executable files unless explicitly directed by the user to do so.
FPT_AEX_EXT.1.5	The application shall be compiled with stack-based buffer overflow protection enabled.

FPT_API_EXT.1	Use of Supported Services and APIs
FPT_API_EXT.1.1	The application shall only use documented platform APIs.
Application Note:	*The list of supported platform APIs has been provided in Appendix A.1 for readability purposes.*

FPT_IDV_EXT.1	Software Identification and Versions
FPT_IDV_EXT.1.1	The application shall be versioned with [[*major.minor.release*]].

FPT_LIB_EXT.1	Use of Third Party Libraries
FPT_LIB_EXT.1.1	The application shall be packaged with only [*the items listed in Section A.2*].
Application Note:	*The list of supported third-party libraries has been provided in Appendix A.2 for readability purposes.*

FPT_TUD_EXT.1	Integrity for Installation and Update
FPT_TUD_EXT.1.1	The application shall [leverage the platform] to check for updates and patches to the application software.
FPT_TUD_EXT.1.2	The application shall [provide the ability] to query the current version of the application software.
FPT_TUD_EXT.1.3	The application shall not download, modify, replace or update its own binary code.
FPT_TUD_EXT.1.4	Application updates shall be digitally signed such that the application platform can cryptographically verify them prior to installation.
FPT_TUD_EXT.1.5	The application is distributed [as an additional software package to the platform OS].

FPT_TUD_EXT.2	Integrity for Installation and Update[2]
FPT_TUD_EXT.2.1	The application shall be distributed using [a container image].
FPT_TUD_EXT.2.2	The application shall be packaged such that its removal results in the deletion of all traces of the application, with the exception of configuration settings, output files, and audit/log events.
FPT_TUD_EXT.2.3	The application installation package shall be digitally signed such that its platform can cryptographically verify them prior to installation.

5.2.6 Trusted Path/Channels (FTP)

FTP_DIT_EXT.1	Protection of Data in Transit [3]
FTP_DIT_EXT.1.1	The application shall [• invoke platform-provided functionality to encrypt all transmitted data with [HTTPS, TLS]] between itself and another trusted IT product.

5.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are included by reference to [App PP].

Table 3 Assurance Components

Requirement Class	Requirement Component
ADV: Development	ADV_FSP.1 Basic Functional Specification
AGD: Guidance Documents	AGD_OPE.1: Operational User Guidance
	AGD_PRE.1: Preparative Procedures
ALC: Life-Cycle Support	ALC_CMC.1: Labelling of the TOE
	ALC_CMS.1: TOE CM coverage
	ALC_TSU_EXT.1: Timely Security Updates
ATE: Tests	ATE_IND.1 Independent Testing – Conformance
AVA: Vulnerability Assessment	AVA_VAN.1 Vulnerability Survey

Consequently, the evaluation activities specified in the [App PP] apply to the TOE evaluation, including any changes made to them by subsequent NIAP Technical Decisions as summarized in section 1.2 above.

[2] This SFR has been modified as per NIAP TD0628

[3] This SFR is modified by TD0655

6. TOE Summary Specification

This chapter describes the security functions of the TOE:

- Cryptographic Support
- User Data Protection
- Security Management
- Privacy
- Protection of the TSF
- Trusted Path/Channels

It also describes the process put in place by the TOE vendor to provide timely security updates to the TOE as per the ALC_TSU_EXT.1 requirements of the [App PP].

6.1 Timely Security Updates

Wickr normally provides releases on a quarterly basis. Bugs may result in additional releases on accelerated schedules. The releases include bug fixes and security updates for all platform versions of the TOE. Additionally, when updates are made to bundled third-party capabilities, they are obtained by Wickr and included in releases. Wickr support personnel contact the POCs for affected customers. The only mechanism to deploy security updates is through maintenance releases. Upon discovery of a vulnerability, the impact will be assessed for priority based on the severity of the bug. The target timeline for releases ranges from 48 hours for critical bugs to 90 days for low severity bugs. Security reports are communicated from customers to Customer Support through an HTTPS form on the HackerOne platform.

6.2 Cryptographic Support

The TOE relies on platform-provided cryptographic functions. The following table identifies the cryptographic libraries implemented and invoked by the TSF and the NIST algorithm certificates that demonstrate that the claimed conformance has been met.

Table 4 Cryptographic Functions

Platform	Function	Standard	Library	Certificate
Linux	ECC key generation (P-384)	FIPS PUB 186-4	Amazon Linux 2 OpenSSL Crypto Module	#A3455
	ECC based key establishment	NIST SP 800-56A	Amazon Linux 2 OpenSSL Crypto Module	#A3455
	SHA-256, SHA-384	FIPS PUB 180-4	Amazon Linux 2 OpenSSL Crypto Module	#A3455
	HMAC-SHA-256, HMAC-SHA-384	FIPS PUB 198-1, FIPS PUB 180-4	Amazon Linux 2 OpenSSL Crypto Module	#A3455
	RSA (4096-bit) ECDSA (P-256, P-384)	FIPS PUB 186-4, Section 4; FIPS PUB 186-4, Section 5	Amazon Linux 2 OpenSSL Crypto Module	#A3455
	AES-GCM (128 bits, 256 bits)	NIST SP 800-38D	Amazon Linux 2 OpenSSL Crypto Module	#A3455

	CTR_DRBG(AES)	NIST SP 800-90A	Amazon Linux 2 OpenSSL Crypto Module	#A3455

The proprietary Entropy Analysis Report (EAR) describes how the platform cryptographic library extracts random data from the platform pseudorandom number generator (PRNG) to ensure that an amount of entropy that is at least equal to the strength of the generated keys is present when seeding the DRBG for key generation purposes. The TOE relies on the OS platform as the entropy source. Specifically, random numbers are obtained from the accumulation of non-deterministic system events aggregated by the Linux PRNG into the /dev/random interface.

The credentials maintained by the TOE are administrator account information (username and salted hashed password) and the private key of the certificate used by the platform TLS server. Credential data is stored in an encrypted volume using Linux Unified Key Setup (LUKS), with an encryption key derived using platform-provided PBKDF2.

The Cryptographic Support security function is designed to satisfy the following security functional requirements:

- FCS_CKM_EXT.1 – The TOE relies on platform-provided cryptographic functionality. As such, the TOE does not implement asymmetric key generation.
- FCS_RBG_EXT.1 – The TOE does not directly use a DRBG to support its security function; the DRBG is called by the underlying platform cryptography that implements TLS functionality.
- FCS_STO_EXT.1 – The TOE uses platform-provided functionality to protect stored credential data.

6.3 User Data Protection

The [App PP] defines 'sensitive data' as follows: "Sensitive data may include all user or enterprise data or may be specific application data such as emails, messaging, documents, calendar items, and contacts. Sensitive data must minimally include PII, credentials, and keys. Sensitive data shall be identified in the application's TSS by the ST author." In the evaluated configuration, the TOE is deployed on a system that uses platform full disk encryption to protect data at rest when the system is not in use.

The table below lists the data that is considered to be 'sensitive data' for this TOE along with where that data resides.

Table 5 Sensitive Data

Sensitive Data	Stored On	Exchange	Protection At Rest	Protection In Transit
User Credentials	App data directory	Not sent to remote systems	Volume encryption using platform-provided PBKDF2	N/A
TLS Server Private Key	File system	Not sent to remote systems	Volume encryption using platform-provided PBKDF2	N/A

Note that peer data in encrypted form passes through the TOE, but it is never decrypted by the TOE. The keys for this encrypted data are only retained by the Wickr Clients.

The underlying platform functionality that the TOE interacts with is the system's network connectivity and system logs. The TOE restricts network connectivity to the following uses only:

- Remotely-initiated: connections to the Wickr Server web GUI (TCP/443), incoming messaging connections from Wickr Clients (TCP/443), user lookups (TCP/3000), bot enterprise service (TCP/4001), installation UI (TCP/8800), web socket messaging (TCP/9000), OIDC service (TCP/9001), Replicated services (TCP/9870-9881), and push device service for client configuration (TCP/9999).
- TSF-initiated: Apple push notifications (TCP/2195), Google push notifications (TCP/443)

The User Data Protection security function is designed to satisfy the following security functional requirements:

- FDP_DAR_EXT.1 – Sensitive data at rest is protected by volume encryption.
- FDP_DEC_EXT.1 – The TOE's use of platform services is well understood by users prior to authorizing the TOE activity.

- FDP_NET_EXT.1 – The TOE communicates over the network for well-defined purposes. Depending on the function, the use of network resources is remotely initiated or initiated by the TOE itself.

6.4 Security Management

The TOE is protected from unauthorized access via the host platform's file system. By default, application binaries and data files are stored in directories where they are not able to be modified by any unauthorized users.

The TOE, when installed, is configured with a single default administrator account with a default password. This password must be changed on first login. Additional user accounts may be configured by authorized administrators.

Configuration data related to the TOE's initial configuration is stored in /etc/replicated. This is used for initial deployment of the TOE and is not managed through the administrative GUI or modified through other operation of the TOE.

The following management capabilities are provided by the TOE:

- Configure the Messaging proxy
- Manage users
- Configure certificates
- Configure room management – default configuration for chat rooms, including room title, room description, whether messages expire after a certain time, and whether messages are automatically deleted after all participants in the chat have read them.
- Configure event logging – allows for configuration of how verbose the log messages are

The Security Management security function is designed to satisfy the following security functional requirements:

- FMT_CFG_EXT.1 – The TOE is protected against direct modification by untrusted users via the host OS platform.
- FMT_MEC_EXT.1 – Locally-modifiable configuration settings for the TOE are stored in appropriate locations.
- FMT_SMF.1 – The TOE is managed remotely through a GUI.

6.5 Privacy

The TOE never transmits known PII over a network. Encrypted PII data may be passed through the TOE at the request of Wickr Client users, but this data is never decrypted on the TOE.

The Privacy security function is designed to satisfy the following security functional requirements:

- FPR_ANO_EXT.1 – the TOE prevents the unnoticed/unauthorized transmission of PII across a network by ensuring that any such transmission is the result of explicit user action.

6.6 Protection of the TSF

The list of APIs used by the TOE is provided in Section A.1 and the list of third-party libraries is provided in Section A.2.

The TOE implements several mechanisms to protect against exploitation. Address space layout randomization (ASLR) is enforced by a combination of the TOE's C code being compiled to use it and through enforcement by the host platform on the Docker engine that the TOE runs on, and the TOE relies fully on the platform to perform memory mapping. There is no situation where the TSF maps memory to an explicit address. There is no allocation of memory with both write and execute permissions. The TOE does not invoke mprotect with the PROT_EXEC permission. The TOE is compatible with AppArmor.

The TOE only writes user-modifiable files to directories that contain executable files when explicitly directed to do so by the user.

The TOE is written in a combination of compiled code (C) for nginx and interpreted code (Node, Java). C code is compiled to use -wl,dynamicbase to enforce ASLR; interpreted code is not subject to stack-based overflow attacks.

The TOE is distributed as a Docker container image. The TOE platform provides the means to check for, apply and verify software updates, while the TOE can display its current version (in major.minor.release format).

The TOE never downloads, modifies, replaces or updates its own binary code. Updates to the TOE are always performed by the platform.

When an installation package is downloaded by the platform, it is verified prior to installation. Updates are digitally signed using a 4096-bit RSA key.

The Protection of the TSF security function is designed to satisfy the following security functional requirements:

- FPT_AEX_EXT.1 – The TOE interacts with its host OS platform in a manner that does not expose the system to memory-related exploitation.
- FPT_API_EXT.1 – The TOE uses documented platform APIs.
- FPT_IDV_EXT.1 – Each TOE distribution is versioned as major.minor.release.
- FPT_LIB_EXT.1 – The set of third-party libraries used by the TOE is well-defined.
- FPT_TUD_EXT.1 – The TOE can be updated through installation packages. Updates are signed by the vendor and validated by the host OS platform prior to installation.
- FPT_TUD_EXT.2 – Updates to the TOE are packaged using container images and removal of the TOE does not preserve any executable code on the platform.

6.7 Trusted Path/Channels

The TOE uses TLS and HTTPS to secure data in transit over trusted channels. The TOE uses platform provided TLS and HTTPS for service requests, data communication, and web administration. The secure protocols are supported by platform-provided functionality. Refer to section 6.2 for the specific NIST validation information.

The TOE uses nginx which makes calls to the platform OpenSSL library to implement trusted communications. The platform-provided TLS implementation uses the following cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

The Trusted Path/Channels security function is designed to satisfy the following security functional requirements:

- FTP_DIT_EXT.1 – The TOE invokes platform-provided functionality to protect sensitive data in transit using TLS and HTTPS.

7. Protection Profile Claims

This ST claims exact conformance to the *Protection Profile for Application Software*, Version 1.4, 07 October 2021 [App PP] along with all applicable errata and interpretations from the certificate issuing scheme.

As explained in section 3, Security Problem Definition, the Security Problem Definition of [App PP] has been included by reference into this ST.

As explained in section 4, Security Objectives, the Security Objectives of [App PP] has been included by reference into this ST.

All claimed SFRs are defined in [App PP]. All mandatory SFRs are claimed. A subset of the optional and objective SFRs are claimed. Selection-based SFR claims are consistent with the selections made in the mandatory SFRs that prompt their inclusion.

8. Rationale

This Security Target includes by reference the [App PP] Security Problem Definition, Security Objectives, and Security Assurance Requirements. The Security Target does not add, remove, or modify any of these items. Security Functional Requirements have been reproduced with the Protection Profile operations completed. All selections, assignments, and refinements made on the claimed Security Functional Requirements have been performed in a manner that is consistent with what is permitted by the [App PP]. The proper set of selection-based requirements have been claimed based on the selections made in the mandatory requirements. Consequently, the claims made by this Security Target are sufficient to address the TOE's security problem. Rationale for the sufficiency of the TOE Summary Specification is provided below.

8.1 TOE Summary Specification Rationale

Each subsection in Section 6, the TOE Summary Specification, describes a security function of the TOE. Each description is followed with rationale that indicates which requirements are satisfied by aspects of the corresponding security function. The set of security functions work together to satisfy all of the security functions and assurance requirements. Furthermore, all of the security functions are necessary in order for the TSF to provide the required security functionality.

This section in conjunction with Section 6, the TOE Summary Specification, provides evidence that the security functions are suitable to meet the TOE security requirements. The collection of security functions work together to provide all of the security requirements. The security functions described in the TOE summary specification are all necessary for the required security functionality in the TSF. The table below demonstrates the relationship between security requirements and security functions.

Table 6 Security Functions vs. Requirements Mapping

	Cryptographic Support	User Data Protection	Security Management	Privacy	Protection of the TSF	Trusted Path/Channels
FCS_CKM_EXT.1	X					
FCS_RBG_EXT.1	X					
FCS_STO_EXT.1	X					
FDP_DAR_EXT.1		X				
FDP_DEC_EXT.1		X				
FDP_NET_EXT.1		X				
FMT_CFG_EXT.1			X			
FMT_MEC_EXT.1			X			
FMT_SMF.1			X			
FPR_ANO_EXT.1				X		
FPT_AEX_EXT.1					X	
FPT_API_EXT.1					X	
FPT_IDV_EXT.1					X	

	Cryptographic Support	User Data Protection	Security Management	Privacy	Protection of the TSF	Trusted Path/Channels
FPT_LIB_EXT.1					X	
FPT_TUD_EXT.1					X	
FPT_TUD_EXT.2					X	
FTP_DIT_EXT.1						X

Appendix A: TOE Usage of Third-Party Components

This Appendix lists the platform APIs and third-party libraries that are used by the application.

A.1 Platform APIs

Listed below are the platform APIs used by the application:

openssl

A.2 Third-Party Libraries

Listed below are the third-party libraries present in the application:

Library	Version
amqp-connection-manager	4.1.3
amqplib	0.7.1
axios-retry	3.2.4
bcrypt	5.0.1
bluebird	3.5.1
body-parser-xml	2.0.3
body-parser	1.19.0
boolean	3.0.2
bson	4.6.1
busboy	1.0.0
bytes	3.1.2
chai	4.2.0
clone	2.1.2
command-line-args	5.1.1
command-line-usage	6.1.0
cookie-parser	1.4.5
crypto-random-int	1.0.1
crypto-random-string	3.3.1
csv-parse	4.15.4
csv-parser	3.0.0
csv-stringify	5.6.2
date-arithmetic	3.1.0
date-fns	2.21.3
debug	4.3.1
deep-equal	2.0.5
deepmerge	4.2.2
dompurify	3.0.1
dotenv	8.2.0
ejs	3.1.6
express-basic-auth	1.2.0
express-fileupload	1.2.1
express-handlebars	5.3.5
express-list-endpoints	5.0.0
express-pino-logger	5.0.0
express-promise-router	3.0.3
express-rate-limit	5.1.3
express	4.17.1
fastq	1.6.0
file-type	16.5.3

flat	5.0.2
foo-foo-mq	5.1.0
generate-json-webpack-plugin	2.0.0
get-value	3.0.1
google-protobuf	3.19.4
handlebars	4.7.7
helmet	4.4.1
hi-base32	0.5.1
hot-shots	5.7.0
ioredis	4.19.1
ipaddr.js	2.0.1
iplocation	7.0.0
is-uuid	1.0.2
js-base64	3.6.0
jsonwebtoken	9.0.0
jwks-rsa	1.12.2
jwt-decode	3.1.2
koa-bodyparser	4.3.0
koa-router	7.4.0
koa	2.13.1
ldapjs	2.2.4
lodash	4.17.19
microtime	3.0.0
minimatch	3.1.2
moment	2.27.0
multer	1.4.4
mysql	5.6
mysql2	1.5.3
netmask	2.0.1
nginx	1.22.1
nodemailer	6.7.2
node-redis	3.1.2
node-retry	0.12.0
open	8.4.0
otplib	12.0.1
parse-duration	0.1.1
pino	5.12.2
pkce-challenge	2.1.0
postgrator	4.0.1
pretty-ms	7.0.1
process	0.11.10
prop-types	15.7.2
qrcode	1.4.4
qs	6.10.1
query-string	7.1.0
random-number-csprng	1.0.2
rate-limit-redis	2.0.0
react-cookie	4.0.3
react-device-detect	1.17.0
react-dom	17.0.2
react-dropzone	11.3.2
react-json-view	1.21.3
react-router-dom	5.2.0

react-select	4.3.1
react-sortable-hoc	2.0.0
react	17.0.2
recurly-js	3.2.0
redis-stream	0.1.0
redoc-cli	0.13.16
request-promise	4.2.6
s3-streams	0.4.0
secure-random	1.1.2
shortid	2.2.16
sinon-chai	3.5.0
statuses	1.5.0
stream-json	1.7.4
styled-components	5.3.0
superagent	5.2.2
supertest	6.1.3
swagger-ui-express	4.1.6
throttle	1.0.3
ts-node	10.2.0
twilio	4.7.2
twin-bcrypt	2.0.1
uuid-validate	0.0.3
uuid	3.4.0
uuidv4	6.2.7
validator	13.9.0
xregexp	3.2.0
y18n	4.0.1
zenko cloudserver	7.5.1
bunny/bunny	0.5.0
christian-riesen/base32	1.6.0
dolondro/google-authenticator	2.2.0
guzzlehttp/guzzle	6.5.8
guzzlehttp/promises	1.4.1
guzzlehttp/psr7	1.9.0
paragonie/random_compat	9.99.100
psr/cache	1.0.1
psr/http-message	1.0.1
psr/log	1.1.4
ralouphie/getallheaders	3.0.3
react/event-loop	1.2.0
react/promise	2.8.0
swiftmailer/swiftmailer	5.4.12
symfony/deprecation-contracts	2.4.0
symfony/polyfill-ctype	1.23.0
symfony/polyfill-intl-idn	1.27.0
symfony/polyfill-intl-normalizer	1.27.0
symfony/polyfill-php72	1.27.0s
symfony/yaml	5.3.6
twilio/sdk	5.42.2