Wickr Enterprise Client 6.10
Security Target

Version 1.0
07 March 2023

Prepared for:

Wickr LLC
W 31st Street
New York, NY 10001

Prepared by:

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

Contents

LIST OF FIGURES AND TABLES

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 Client 6.10.

Wickr Enterprise Client includes stand-alone executable clients for Windows, macOS, iOS, and Android systems. These clients are installed on endpoint systems in an organization to facilitate peer communications. The Wickr Enterprise Client interacts with an environmental Wickr Enterprise Server that is used for administration of the Wickr Enterprise Client and communication control.

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 Client 6.10 Security Target

ST Version – Version 1.0

ST Date – 07 March 2023

TOE Identification – Wickr Enterprise Client 6.10. The platform-specific versions of the TOE include:

1. Wickr Enterprise Client for Windows 6.10.2

   Evaluated on Microsoft Windows 10.

2. Wickr Enterprise Client for macOS 6.10.2

   Evaluated on macOS 12.4 Monterey.

3. Wickr Enterprise Client for iOS 6.10.0

   Evaluated on iOS 15.5.

4. Wickr Enterprise Client for Android 6.10.0

   Evaluated on Android 12.

TOE Developer – Wickr LLC

Evaluation Sponsor – Wickr LLC

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 claim exact conformance to the following CC specifications:

- *Protection Profile for Application Software*, Version 1.4, 07 October 2021 with the following optional, selection-based, and objective SFRs:
  - FCS_CKM.1/SK
  - FCS_COP.1/SKC
  - FCS_RBG_EXT.2
  - FIA_X509_EXT.1
  - FIA_X509_EXT.2
  - FPT_TUD_EXT.2

- The following NIAP Technical Decisions apply to this PP 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	Applicable
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. An iterated SFR that is defined in [App PP] identifies this using a slash and additional characters placed at end of the component, generally intended to be descriptive of the purpose of each iteration. For example, FCS_CKM.1/AK and FCS_CKM.1/SK indicate that the PP includes two iterations of the FCS_CKM.1

requirement: /AK (asymmetric keys) and /SK (symmetric keys). Depending on the specific claims made, the ST may include some or all of the iterated components. For iterations defined by the ST author, the same convention is used but with a sequential number following the slash.

- 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 Acronyms

AA	Assurance Activity
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
UUID	Universally Unique IDentifier

2. Product and TOE Description

The TOE is the Wickr Enterprise Client 6.10 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 Client is an on-premise application providing communication with remote peers.

Wickr Enterprise Client is part of a client-server distribution. The TOE is the client portion of this distribution. It interacts with the Wickr Enterprise Server 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 Client software application (hereinafter referred to as Wickr Client). The TOE may be deployed on Windows, Android, iOS or macOS.

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 Server.
- 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 a host platform that is configured for secure operation.
- The ability of the TOE to interface with the low-level components of its host platform in such a manner that the TOE cannot be used as an attack vector to exploit the host platform.
- 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 for messaging rooms. The Wickr Client uses platform-provided TLS for security of data in transit.

2.4 TOE Architecture

The Wickr Client TOE is a software application that runs on several different platforms:

- Windows
- macOS
- iOS
- Android

The product architecture is shown in the following figure. The Wickr Client application (the TOE) is indicated by the red box. The other items are implemented on the Wickr Servers and other systems that are part of the TOE's Operational Environment.

The figure depicts initial communication setup for messaging. The direction of the arrows indicates which system initiates communication. Communication is bidirectional once a connection is established. The Wickr Client relies on the cryptographic functions of its host platform for data in transit.

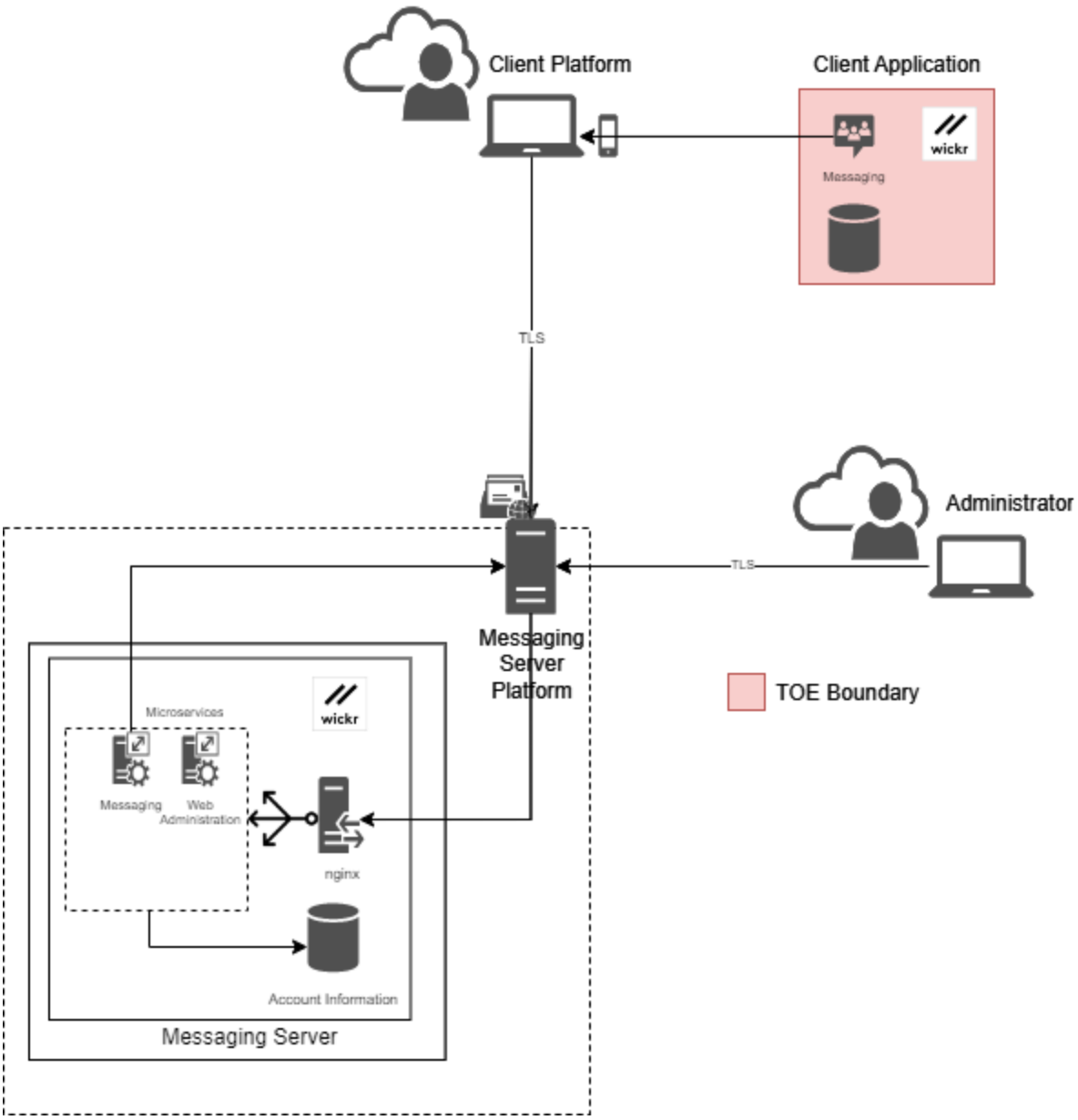

Figure 1: Wickr Architecture

2.4.1 Physical Boundary

The TOE consists of the Wickr Client application. For this evaluation, the TOE is evaluated on the following specific platforms:

- Windows:
  - Microsoft Surface Laptop 4
  - Intel i5-1145G7 (Tiger Lake) processor
  - Windows 10 64-bit OS
- macOS:
  - MacBook Pro
  - Intel i7-9750H (Coffee Lake) processor
  - macOS Monterey 12.4 OS
- iOS:
  - iPhone 12

  - Apple A14 Bionic (ARMv8) processor
  - iOS 15.5 OS
- Android:
  - Samsung Galaxy S20 FE
  - Qualcomm Snapdragon 865 (ARMv8) processor
  - Android 12 OS

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

- A Wickr Server installed for messaging
- One or more remote Wickr Client instances to establish connections with
- A Workstation with a browser to access the Wickr Server's Admin Console. (Wickr Clients receive configuration information from a Wickr Server)
- An 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
- Identification and Authentication
- 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 for this are supplied by the host platform. All platform versions of the TOE also implement their own NIST-validated cryptographic algorithms through OpenSSL to support the protection of credential data at rest. The TOE relies on platform-provided entropy for random number generator seeding.

The TOE uses cryptographic functionality to protect stored credential data. This is done through a combination of TSF-provided cryptography and platform cryptography for all platform versions.

2.4.2.2 User Data Protection

The TOE provides cryptographic functionality and also leverages functionality provided by its underlying OS platforms to secure sensitive data at rest. The TOE uses network resources provided by the underlying platforms. All platform services are invoked at the direction of the user.

The TOE uses network connectivity to interact with a Wickr Server to establish connections with other Wickr Clients. The TOE or its platform, depending on platform version, check for updates from an update server.

2.4.2.3 Identification and Authentication

The TOE relies on platform-provided functionality to validate X.509 certificates used to authenticate TLS servers when establishing trusted communications except in the case where the desktop platform versions of the TOE (macOS, Windows) are responsible for validating the crlsign bit on any certificate used to sign a CRL. Certificate validation is performed in accordance with RFC 5280 and CRLs are used for revocation checking in all cases except for iOS, which uses OCSP.

2.4.2.4 Security Management

Wickr Client configuration data is stored locally using mechanisms that are recommended by the respective platform vendors. The TOE is not installed with default credentials. The Wickr Client applies configuration settings it obtains from the Wickr Server.

2.4.2.5 Privacy

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

2.4.2.6 Protection of the TSF

The TOE includes measures to integrate securely with its underlying OS 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 configured in a secure manner. All platform versions of the TOE are compiled with stack overflow protection.

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 in a format that is appropriate for the TOE's platform. Updates to the TOE are digitally signed, and the signature is validated prior to installation. The TOE does not modify its own code. Removal of the application removes all executable code associated with the TOE.

2.4.2.7 Trusted Path/Channels

The TOE uses trusted channels to secure data in transit between itself and external entities. The TOE communicates with the Wickr Server for messaging services and authentication using platform provided TLS.

2.5 TOE Documentation

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

- Wickr Enterprise Administrator Guide, v426151b
- Wickr Enterprise Installation and Maintenance v1.30.0
- Wickr Enterprise Desktop User Guide v6.10
- Wickr Enterprise Client Common Criteria Evaluated Configuration Guide (CCECG), Version 1.0

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 because the TOE 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 (iterated by ST author)
- FCS_RBG_EXT.2: Random Bit Generation from Application
- FCS_STO_EXT.1: Storage of Credentials (iterated by ST author)
- FDP_DAR_EXT.1: Encryption of Sensitive Application Data
- FDP_DEC_EXT.1: Access to Platform Resources
- FDP_NET_EXT.1: Network Communications
- FIA_X509_EXT.1: X.509 Certificate Validation (iterated by ST author)
- FIA_X509_EXT.2: X.509 Certificate Authentication
- 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 (iterated by ST author)
- 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_CKM.1/SK Cryptographic Symmetric Key Generation
	FCS_COP.1/SKC Cryptographic Operation – Encryption/Decryption
	FCS_RBG_EXT.1 Random Bit Generation Services
	FCS_RBG_EXT.2 Random Bit Generation from Application
	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
FIA: Identification and authentication	FIA_X509_EXT.1/1 X.509 Certificate Validation (macOS, Windows)
	FIA_X509_EXT.1/2 X.509 Certificate Validation (Android)
	FIA_X509_EXT.1/3 X.509 Certificate Validation (iOS)
	FIA_X509_EXT.2 X.509 Certificate Authentication
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.1/1 Integrity for Installation and Update (macOS, Windows)
	FPT_TUD_EXT.1/2 Integrity for Installation and Update (Android, iOS)
	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]

[1] This SFR is modified by TD0717

FCS_CKM_EXT.1.1 The application shall [

- Generate no asymmetric cryptographic keys

].

FCS_CKM.1/SK	Cryptographic Symmetric Key Generation

FCS_CKM.1.1/SK The application shall generate symmetric cryptographic keys using a Random Bit Generator as specified in FCS_RBG_EXT.1 and specified cryptographic key sizes [

- 256 bit

].

FCS_COP.1/SKC	Cryptographic Operation – Encryption/Decryption[2]

FCS_COP.1.1/SKC The application shall perform [*encryption/decryption*] in accordance with a specified cryptographic algorithm [

- AES-GCM (as defined in NIST SP 800-38D) mode

] and cryptographic key sizes [256-bit].

FCS_RBG_EXT.1	Random Bit Generation Services

FCS_RBG_EXT.1.1 The application shall [

- implement DRBG functionality

] for its cryptographic functions.

FCS_RBG_EXT.2	Random Bit Generation from Application

FCS_RBG_EXT.2.1 The application shall perform all deterministic random bit generation (DRBG) services in accordance with NIST Special Publication 800-90A using [CTR_DRBG (AES)].

FCS_RBG_EXT.2.2 The deterministic RBG shall be seeded by an entropy source that accumulates entropy from a platform-based DRBG and [no other noise source] with a minimum of [256 bits] of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

FCS_STO_EXT.1	Storage of Credentials

FCS_STO_EXT.1.1 The application shall [

- invoke the functionality provided by the platform to securely store [*client API key*],
- implement functionality to securely store [*database key*] according to [FCS_COP.1/SKC]

] to non-volatile memory.

[2] This SFR is modified by TD0717

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, • location services].
FDP_DEC_EXT.1.2	The application shall restrict its access to [• address book, • [*photos*]].

FDP_NET_EXT.1	Network Communications
FDP_NET_EXT.1.1	The application shall restrict network communication to [• user-initiated communication for [*connection to Wickr Server*], • respond to [*push notifications for incoming peer connections (iOS, Android)*] • [*application-initiated checking for updates (Android, iOS, macOS, and Windows)*]].

5.2.3 Identification and Authentication (FIA)

FIA_X509_EXT.1/1	X.509 Certificate Validation (macOS, Windows)

FIA_X509_EXT.1.1/1	The **macOS and Windows** application shall [invoke platform-provided functionality, implement functionality] to validate certificates in accordance with the following rules:
	• RFC 5280 certificate validation and certificate path validation.
	• The certificate path must terminate with a trusted CA certificate.
	• The application shall validate a certificate path by ensuring the presence of the basicConstraints extension, that the CA flag is set to TRUE for all CA certificates, and that any path constraints are met.
	• The application shall validate that any CA certificate includes caSigning purpose in the key usage field
	• The application shall validate the revocation status of the certificate using [CRL as specified in RFC 5280 Section 6.3]
	• The application shall validate the extendedKeyUsage field according to the following rules:
	  ○ Certificates used for trusted updates and executable code integrity verification shall have the Code Signing Purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
	  ○ Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the EKU field.

- Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the EKU field.
- S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the EKU field.
- OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the EKU field.
- Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the EKU field.

FIA_X509_EXT.1.2/1 The application shall treat a certificate as a CA certificate only if the basicConstraints extension is present and the CA flag is set to TRUE.

Application Note: *The macOS and Windows platform versions of the TOE rely on the platform for X.509 validation for all functions except for determining whether a CRL is valid based on the presence of the crlsign key usage bit of the certificate used to sign the CRL.*

FIA_X509_EXT.1/2	**X.509 Certificate Validation (Android)**

FIA_X509_EXT.1.1/2 The **Android** application shall [invoke platform-provided functionality] to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The application shall validate a certificate path by ensuring the presence of the basicConstraints extension, that the CA flag is set to TRUE for all CA certificates, and that any path constraints are met.
- The application shall validate that any CA certificate includes caSigning purpose in the key usage field
- The application shall validate the revocation status of the certificate using [CRL as specified in RFC 5280 Section 6.3]
- The application shall validate the extendedKeyUsage field according to the following rules:
  - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing Purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
  - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the EKU field.
  - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the EKU field.
  - S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the EKU field.
  - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the EKU field.
  - Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the EKU field.

FIA_X509_EXT.1.2/2 The application shall treat a certificate as a CA certificate only if the basicConstraints extension is present and the CA flag is set to TRUE.

Application Note: *The Android platform version of the TOE relies fully on the platform for X.509 validation and uses CRL for revocation checking.*

FIA_X509_EXT.1/3	**X.509 Certificate Validation (iOS)**

FIA_X509_EXT.1.1/3 The **iOS** application shall [invoke platform-provided functionality] to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.

- The certificate path must terminate with a trusted CA certificate.
- The application shall validate a certificate path by ensuring the presence of the basicConstraints extension, that the CA flag is set to TRUE for all CA certificates, and that any path constraints are met.
- The application shall validate that any CA certificate includes caSigning purpose in the key usage field
- The application shall validate the revocation status of the certificate using [OCSP as specified in RFC 6960]
- The application shall validate the extendedKeyUsage field according to the following rules:
    - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing Purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
    - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the EKU field.
    - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the EKU field.
    - S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the EKU field.
    - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the EKU field.
    - Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the EKU field.

FIA_X509_EXT.1.2/3 The application shall treat a certificate as a CA certificate only if the basicConstraints extension is present and the CA flag is set to TRUE.

Application Note: *The iOS platform version of the TOE relies fully on the platform for X.509 validation and uses OCSP for revocation checking.*

FIA_X509_EXT.2	**X.509 Certificate Authentication**

FIA_X509_EXT.2.1 The application shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [TLS].

FIA_X509_EXT.2.2 When the application cannot establish a connection to determine the validity of a certificate, the application shall [accept the certificate].

5.2.4 Security Management (FMT)

FMT_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 logging,*

- *delete stored logs,*
- *secure shredder,*
- *configuration of authentication lockout*]

].

5.2.5 Privacy (FPR)

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

5.2.6 Protection of 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 third-party libraries 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.1/1	**Integrity for Installation and Update (macOS, Windows)**
FPT_TUD_EXT.1.1/1	The application shall [provide the ability] to check for updates and patches to the application software.
Application Note:	*The macOS and Windows platform versions of the TOE use their own mechanism to check for the availability of software updates.*
FPT_TUD_EXT.1.2/1	The application shall [provide the ability] to query the current version of the application software.
FPT_TUD_EXT.1.3/1	The application shall not download, modify, replace or update its own binary code.

FPT_TUD_EXT.1.4/1	Application updates shall be digitally signed such that the application platform can cryptographically verify them prior to installation.
FPT_TUD_EXT.1.5/1	The application is distributed [as an additional software package to the platform OS].

FPT_TUD_EXT.1/2	**Integrity for Installation and Update (Android, iOS)**
FPT_TUD_EXT.1.1/2	The application shall [leverage the platform] to check for updates and patches to the application software.
Application Note:	*The Android and iOS platform versions of the TOE use platform mechanisms to check for the availability of software updates.*
FPT_TUD_EXT.1.2/2	The application shall [provide the ability] to query the current version of the application software.
FPT_TUD_EXT.1.3/2	The application shall not download, modify, replace or update its own binary code.
FPT_TUD_EXT.1.4/2	Application updates shall be digitally signed such that the application platform can cryptographically verify them prior to installation.
FPT_TUD_EXT.1.5/2	The application is distributed [as an additional software package to the platform OS].

FPT_TUD_EXT.2	**Integrity for Installation and Update**[3]
FPT_TUD_EXT.2.1	The application shall be distributed using [the format of the platform-supported package manager].
FPT_TUD_EXT.2.2	The application shall be packaged such that its removal results in the deletion of all traces of the application, with the exception of configuration settings, output files, and audit/log events.
FPT_TUD_EXT.2.3	The application installation package shall be digitally signed such that its platform can cryptographically verify them prior to installation.

5.2.7 Trusted Path/Channels (FTP)

FTP_DIT_EXT.1	**Protection of Data in Transit** [4]
FTP_DIT_EXT.1.1	The application shall [• invoke platform-provided functionality to encrypt all transmitted data with [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

[3] This SFR is modified by TD0628
[4] This SFR is modified by TD0655

	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.

6. TOE Summary Specification

This chapter describes the security functions of the TOE:

- Cryptographic Support
- User Data Protection
- Identification and Authentication
- 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 invokes platform-provided cryptography to secure data in transit. All cryptographic services used for data in transit protection are implemented by the platform cryptographic library for the respective platforms.

The TOE implements its own cryptography (Wickr OpenSSL version 2.0.16) for use in credential storage protection. Credential storage is accomplished through a combination of platform cryptography and TSF cryptography.

The following table identifies the cryptographic functions implemented and invoked by the TOE and the NIST algorithm certificates that demonstrate the validity of these functions.

Table 4 Cryptographic Functions

Platform	Function	Standard	Library	Certificate
Windows	ECC key generation (P-384, P-521)	FIPS PUB 186-4	Windows 10 SymCrypt	#A2677
	ECC based key establishment	NIST SP 800-56A	Windows 10 SymCrypt	#A2677
	SHA-256, SHA-384	FIPS PUB 180-4	Windows 10 SymCrypt	#A2677
	HMAC-SHA-256, HMAC-SHA-384	FIPS PUB 198-1, FIPS PUB 180-4	Windows 10 SymCrypt	#A2677
	RSA (2048-bit or greater) ECDSA (P-256, P-384)	FIPS PUB 186-4, Section 4; FIPS PUB 186-4, Section 5	Windows 10 SymCrypt	#A2677
	AES-CBC (256 bits), AES-GCM (128 bits, 256 bits)	NIST SP 800-38	Windows 10 SymCrypt	#A2677
	AES-GCM (256 bits)		Wickr OpenSSL Version 2.0.16	#A3372
	CTR_DRBG(AES)	NIST SP 800-90A	Windows 10 SymCrypt	#A2677
			Wickr OpenSSL Version 2.0.16	#A3372
macOS	ECC key generation (P-384, P-521)	FIPS PUB 186-4	Apple coreCrypto Module [User]	#A2820
	ECC based key establishment	NIST SP 800-56A	Apple coreCrypto Module [User]	#A2820
	SHA-256, SHA-384	FIPS PUB 180-4	Apple coreCrypto Module [User]	#A2820
	HMAC-SHA-256, HMAC-SHA-384	FIPS PUB 198-1, FIPS PUB 180-4	Apple coreCrypto Module [User]	#A2820
	RSA (2048-bit or greater) ECDSA (P-256, P-384)	FIPS PUB 186-4, Section 4; FIPS PUB 186-4, Section 5	Apple coreCrypto Module [User]	#A2820
	AES-CBC (256 bits), AES-GCM (128 bits, 256 bits)	NIST SP 800-38	Apple coreCrypto Module [User]	#A2820

	AES-GCM (256 bits)		Wickr OpenSSL Version 2.0.16	#A3372
	CTR_DRBG(AES)	NIST SP 800-90A	Apple coreCrypto Module [User]	#A2820
			Wickr OpenSSL Version 2.0.16	#A3372
iOS	ECC key generation (P-384, P-521)	FIPS PUB 186-4	Apple coreCrypto Module [User]	#A2788
	ECC based key establishment	NIST SP 800-56A	Apple coreCrypto Module [User]	#A2786
	SHA-256, SHA-384	FIPS PUB 180-4	Apple coreCrypto Module [User]	#A2789 (SHA-256), #A2788 (SHA-384)
	HMAC-SHA-256, HMAC-SHA-384	FIPS PUB 198-1, FIPS PUB 180-4	Apple coreCrypto Module [User]	#A2789 (HMAC-SHA-256), #A2788 (HMAC-SHA-384)
	RSA (2048-bit or greater) ECDSA (P-256, P-384)	FIPS PUB 186-4, Section 4; FIPS PUB 186-4, Section 5	Apple coreCrypto Module [User]	#A2788
	AES-GCM (128 bits, 256 bits)	NIST SP 800-38	Apple coreCrypto Module [User]	#A2787
	AES-GCM (256 bits)		Wickr OpenSSL Version 2.0.16	#A3372
	CTR_DRBG(AES)	NIST SP 800-90A	Apple coreCrypto Module [User]	#A2787
			Wickr OpenSSL Version 2.0.16	#A3372
Android	ECC key generation (P-384, P-521)	FIPS PUB 186-4	Samsung BoringSSL Android	#A2351
	ECC based key establishment	NIST SP 800-56A	Samsung BoringSSL Android	#A2351
	SHA-256, SHA-384	FIPS PUB 180-4	Samsung BoringSSL Android	#A2351

	HMAC-SHA-256, HMAC-SHA-384	FIPS PUB 198-1, FIPS PUB 180-4	Samsung BoringSSL Android	#A2351
	RSA (2048-bit or greater) ECDSA (P-256, P-384)	FIPS PUB 186-4, Section 4; FIPS PUB 186-4, Section 5	Samsung BoringSSL Android	#A2351
	AES-CBC (256 bits), AES-GCM (128 bits, 256 bits)	NIST SP 800-38	Samsung BoringSSL Android	#A2351
	AES-GCM (256 bits)		Wickr OpenSSL Version 2.0.16	#A3372
	CTR_DRBG(AES)	NIST SP 800-90A	Samsung BoringSSL Android	#A2351
			Wickr OpenSSL Version 2.0.16	#A3372

To ensure sufficient strength for the keys generated for credential storage, all versions of the TOE implement DRBG functionality for key generation, using the AES-256 CTR mode. For all key generation functionality, whether performed by the TOE or invoked from the operational environment, the proprietary Entropy Analysis Report (EAR) describes how the platform extracts random data from its 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 following platform APIs, depending on the platform used:

- Windows: ProcessPRNG
- Android, macOS, iOS: /dev/random

It is assumed that these platform DRBGs provide at least 256 bits of entropy.

The TOE maintains a unique API key that is used as a credential for server communications. This API key, along with sensitive data regarding messaging history and contacts, is stored in an encrypted SQL database. All platform versions of the TOE use platform-provided AES to encrypt the database; iOS uses Coredata, which uses Apple CoreCrypto to encrypt the database using AES-256-GCM, and all other platforms use SQLCipher which call the respective platform cryptographic libraries to encrypt the database using AES-256-CBC. The DEK that encrypts the database is generated randomly. The DEK is encrypted by a KEK, which is not persistently stored, using AES-256-GCM. The generation of the DEK/KEK and encryption of the KEK are performed by the TSF.

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_CKM.1/SK – The TOE generates symmetric keys for protection of locally-stored credential data.
- FCS_COP.1/SKC – The TOE uses a NIST-validated implementation to perform AES encryption and decryption in support of storage of credential data.
- FCS_RBG_EXT.1 – The TOE implements a random bit generation function for the purpose of symmetric key generation.
- FCS_RBG_EXT.2 – The TOE uses a NIST-validated implementation to generate pseudo-random bits and this implementation is seeded with sufficiently strong entropy collected from the platform DRBG.

- FCS_STO_EXT.1 – The TOE uses a combination of platform-provided functionality and its own functionality to securely store credential data at rest.

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."

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
Client API key	Encrypted client database	Issued from TOE to Wickr server at user direction	Encrypted by FCS_STO_EXT.1 (see DEK reference in section 6.2)	TLS
Database Key	OS file system, encrypted	n/a	Encrypted by FCS_STO_EXT.1 (see KEK reference in section 6.2)	n/a
Message history	Encrypted client database	n/a	Encrypted by FCS_STO_EXT.1 (see DEK reference in section 6.2)	n/a
Contacts	Encrypted client database	n/a	Encrypted by FCS_STO_EXT.1 (see DEK reference in section 6.2)	n/a

The underlying platform functionality that the TOE interacts with is the system's network connectivity. Network usage of the TOE is authorized implicitly through user guidance; it does not make any specific requests on its own to use network services once installed. The TOE restricts network connectivity to the following uses only:

- User-initiated: connections to the Wickr Server
  - Note that a Wickr Client receives an inbound message/call through the connection it initially established to a Wickr Server.
- Remotely-initiated: push notifications for incoming connections (iOS and Android)
- Application-initiated: checking for updates (Android, iOS, macOS, and Windows)

In support of peer connections, the TOE may access the following platform hardware or other resources: location services, photos, and address book.

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 the mechanisms claimed in the iterations of FCS_STO_EXT.1.
- 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 user-initiated or initiated by the TOE itself.

6.4 Identification and Authentication

The TOE relies on platform-provided functionality for validation of X.509 certificates for authenticating the Wickr Server in almost all cases. The sole exception to this is that desktop versions of the application (macOS and Windows) are responsible for checking that the certificate used to sign a CRL is valid by the presence or absence of the crlsign key usage bit on the signing certificate.

Certificates are validated in the following manner:

- Certificate validation and certificate path validation are performed in accordance with RFC 5280.
- The certificate path is checked to ensure that it terminates with a trusted CA certificate.
- The certificate path is validated by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.
- iOS uses OCSP (RFC 6960) for revocation checking. All other platforms use CRL (RFC 5280).
- The application shall validate the extendedKeyUsage field according to the following rules:
  - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
  - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
  - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
  - S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the extendedKeyUsage field.

Certificate revocation status is checked using CRLs. In the event that the revocation status of a certificate cannot be verified, the certificate will be accepted.

Because the purpose of certificate validation function is to validate the authenticity of a TLS server, the TOE platform chooses what certificates to use based on what is presented by the server as part of establishing the TLS session.

The Identification and Authentication security function is designed to satisfy the following security functional requirements:

- FIA_X509_EXT.1/1 – X.509 certificates are validated for desktop applications (Windows, macOS) by a combination of the TOE and the platform when establishing trusted communications. CRL is used for revocation checking.
- FIA_X509_EXT.1/2 – X.509 certificates are validated for Android by the platform when establishing trusted communications. CRL is used for revocation checking.
- FIA_X509_EXT.1/3 – X.509 certificates are validated for iOS by the platform when establishing trusted communications. OCSP is used for revocation checking.
- FIA_X509_EXT.2 – When revocation status of a certificate cannot be determined, the certificate is treated as invalid.

6.5 Security Management

All TOE components are protected from unauthorized access via the host platform's file system. No default credentials are present in the TOE. The platform administrator registers the user. By default, application binaries and data files are stored in directories where they are not able to be modified by any unauthorized users.

The Wickr Server connection is established as part of initial configuration of the TOE. Once the TOE is in an operational state and connected to the Wickr Server, the server can interface with the client to communicate configuration changes through a JSON API on the client. Specifically, the server can be used to configure the number of authentication failures that are allowed for a user to access the client.

The TOE allows local users to configure logging, which includes whether logging is enabled, whether enhanced logging is enabled, and where logs are stored. The user can also delete stored logs through the TOE. The TOE also has a 'secure shredder' function which will overwrite all memory and disk space used by files that are opened through the application.

Configuration settings that are not sensitive are stored by platform in the manner specified below. This includes user identification information (username/email), the current invalid password retry count, flags for whether the user is registered, needs to be shown a different onboarding screen, or (for iOS/Android) whether the user has granted permission for the application to use device features, and basic key/value data such as recently used emoji characters.

Android

The TOE stores all data other than user credentials in a custom SharedPreferences implementation that stores an encrypted blob (as opposed to plaintext key/value pairs) using the JNI library by providing the device UUID and

generated salt. The files are all in the protected/private app folder which would require root to access outside of the TOE.

iOS

Configuration settings are stored using the user defaults system. The app folder uses FileProtectionCompleteUntilFirstUserAuthentication to enable encryption. Registration configuration information is stored as a JSON-encoded file in the app document folder. This file is saved with the NSDataWritingFileProtectionCompleteUntilFirstUserAuthentication flag, so the file is only decrypted if the user has unlocked the phone.

macOS

The data path for the TOE is /Users/[USER]/Library/Application Support/Wickr, LLC/WickrEnterprise. Configuration settings are stored using the NSUserDefaults plist format.

Windows

Configuration settings are stored in the registry under HKEY_USERS\[USER ID]\Software\Wickr Enterprise

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 each TOE platform version are stored in appropriate locations for each host OS platform.
- FMT_SMF.1 – The TOE makes some configuration options available to the user through the application itself, while some configuration is pushed down to it from an environmental Wickr Server.

6.6 Privacy

The TOE never transmits known PII over a network. The user may direct the TOE to transmit data that contains PII.

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.

6.7 Protection of the TSF

For readability purposes, the list of APIs used on each platform 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. The TOE implements address space layout randomization (ASLR) using the dynamicbase flag and relies fully on its underlying host 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. All platform versions of the TOE are compiled with stack overflow protection. For Windows, the /GS flag is used. For macOS, stack_chk_fail and stack_chk_guard are used. For all other platforms, -fstack-protector-all is used.

The TOE is compatible with platform security features. In particular:

- The Android platform version of the TOE is compatible with Google Android OS
- The iOS platform version of the TOE can be run without disabling any iOS security features.
- The macOS platform version of the TOE can be run without disabling any macOS security features.
- The Windows platform version of the TOE can run successfully with Windows Defender Exploit Guard Exploit Protection configured with the following mitigations enabled: Control Flow Guard (CFG), Randomize memory allocations (Bottom-Up ASLR), Import address filtering (IAF), and Data Execution Prevention (DEP).

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 distributed as an additional software package to the platform. The TOE platform provides the means to apply and verify software updates, while the TOE can display its current version (in major.minor.release format). With respect to checking for updates, the implementation of this (TOE vs platform) is dependent on the platform version of the TOE. The macOS and Windows platform versions of the TOE periodically check for updates on the vendor's support site. The Android and iOS platform versions of the TOE periodically check for updates through a check by the TOE to their respective app stores. For these platforms, updates are checked hourly. Regardless of the platform version, 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 for the Android platform version of the TOE are digitally signed using a 4096-bit RSA key; updates for the iOS, macOS and Windows platform versions of the TOE are digitally signed using a 2048-bit RSA key.

Installation packages are in the standard format for updates on their platforms. In particular, Windows updates use MSI format, while macOS updates use DMG format. Android updates use Android App Bundle (AAB) format. iOS updates are distributed as an ipa file (iOS App Store Package). In all cases, uninstalling the TOE fully removes all traces of it from the platform.

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 – The TOE 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.1/1 – The TOE can be updated through installation packages whose availability is checked by the TOE. Updates are signed by the vendor and validated by the host OS platform prior to installation.
- FPT_TUD_EXT.1/2 – The TOE can be updated through installation packages whose availability is checked by the platform. Updates are signed by the vendor and validated by the host OS platform prior to installation.
- FPT_TUD_EXT.2 – The TOE is packaged using a standardized format for the supported OS platform and removal of the TOE does not preserve any executable code on the platform.

6.8 Trusted Path/Channels

The TOE relies on TLS to secure data in transit over trusted channels. The TOE uses platform-provided TLS for authentication and messaging communication. The protocol implementation is supported by NIST-validated cryptographic mechanisms provided by the platform. Refer to section 6.2 for the specific NIST validation information.

The macOS and Windows versions of the TOE invoke the following platform TLS interfaces through calls made by the Qt library included with the TOE:

- macOS: Apple Secure Transport
- Windows: Schannel

The iOS version of the TOE calls the URLSession API, which implements cryptographic functionality using the CoreCrypto library. The Android platform version of the TOE uses the okhttp third-party library which makes calls to the platform BoringSSL for protocol and cryptographic functionality.

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.

7. Protection Profile Claims

This ST claims exact conformance to the *Protection Profile for Application Software*, Version 1.4, 7 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] have 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	Identification and Authentication	Security Management	Privacy	Protection of the TSF	Trusted Path/Channels
FCS_CKM_EXT.1	X						
FCS_CKM.1/SK	X						
FCS_COP.1/SKC	X						
FCS_RBG_EXT.1	X						
FCS_RBG_EXT.2	X						
FCS_STO_EXT.1	X						
FDP_DAR_EXT.1		X					
FDP_DEC_EXT.1		X					
FDP_NET_EXT.1		X					
FIA_X509_EXT.1/1			X				
FIA_X509_EXT.1/2			X				
FIA_X509_EXT.1/3			X				
FIA_X509_EXT.2			X				

	Cryptographic Support	User Data Protection	Identification and Authentication	Security Management	Privacy	Protection of the TSF	Trusted Path/Channels
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	
FPT_LIB_EXT.1						X	
FPT_TUD_EXT.1/1						X	
FPT_TUD_EXT.1/2						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 TOE:

A.1 Platform APIs

Listed below are the platform APIs used by each platform version.

Android

androidx.multidex:multidex:2.0.1

androidx.annotation:annotation:1.1.0

androidx.appcompat:appcompat:1.1.0

androidx.legacy:legacy-support-v4:1.0.0

androidx.legacy:legacy-support-v13:1.0.0

com.google.android.material:material:1.2.1

androidx.exifinterface:exifinterface:1.0.0

androidx.constraintlayout:constraintlayout:1.1.3

androidx.biometric:biometric:1.0.1

androidx.sqlite:sqlite:2.1.0

androidx.dynamicanimation:dynamicanimation:1.0.0

androidx.browser:browser:1.2.0

androidx.lifecycle:lifecycle-extensions:2.1.0

androidx.lifecycle:lifecycle-viewmodel:2.1.0

androidx.lifecycle:lifecycle-livedata:2.1.0

androidx.lifecycle:lifecycle-common-java8:2.1.0

com.google.android.gms:play-services-auth:17.0.0

com.google.android.gms:play-services-location:17.0.0

com.google.android.gms:play-services-maps:17.0.0

com.googlecode.libphonenumber:libphonenumber:8.9.16

com.google.firebase:firebase-iid:20.0.2

com.google.firebase:firebase-messaging:20.1.0

iOS

Accelerate

AddressBook

AddressBookUI

AssetsLibrary

AudioToolbox

AVFoundation

CFNetwork

CoreAudio

CoreGraphics

CoreData

CoreLocation

CoreMedia

CoreVideo

Contacts

ContactUI

CallKit

Foundation

ImageIO

MapKit

MediaPlayer

MessageUI

MobileCoreServices

OpenGLES

QuartzCore

SafariServices

Security

SystemConfiguration

UIKit

URLSession

PushKit

XCTest

macOS

/usr/lib/libobjc.A.dylib (compatibility version 1.0.0, current version 228.0.0)

/System/Library/Frameworks/QTKit.framework/Versions/A/QTKit (compatibility version 1.0.0, current version 1.0.0)

/System/Library/Frameworks/CoreVideo.framework/Versions/A/CoreVideo (compatibility version 1.2.0, current version 1.5.0)

/System/Library/Frameworks/CoreAudio.framework/Versions/A/CoreAudio (compatibility version 1.0.0, current version 1.0.0)

/System/Library/Frameworks/AudioToolbox.framework/Versions/A/AudioToolbox (compatibility version 1.0.0, current version 1000.0.0)

/System/Library/Frameworks/AudioUnit.framework/Versions/A/AudioUnit (compatibility version 1.0.0, current version 1.0.0)

/System/Library/Frameworks/AVFoundation.framework/Versions/A/AVFoundation (compatibility version 1.0.0, current version 2.0.0)

/System/Library/Frameworks/ApplicationServices.framework/Versions/A/ApplicationServices (compatibility version 1.0.0, current version 52.0.0)

/System/Library/Frameworks/VideoDecodeAcceleration.framework/Versions/A/VideoDecodeAcceleration (compatibility version 1.0.0, current version 1.0.0)

/System/Library/Frameworks/VideoToolbox.framework/Versions/A/VideoToolbox (compatibility version 1.0.0, current version 1.0.0)

/System/Library/Frameworks/CoreMedia.framework/Versions/A/CoreMedia (compatibility version 1.0.0, current version 1.0.0)

/System/Library/Frameworks/Carbon.framework/Versions/A/Carbon (compatibility version 2.0.0, current version 162.0.0)

/System/Library/Frameworks/AddressBook.framework/Versions/A/AddressBook (compatibility version 1.0.0, current version 2421.0.0)

/System/Library/Frameworks/SystemConfiguration.framework/Versions/A/SystemConfiguration (compatibility version 1.0.0, current version 1061.40.2)

/System/Library/Frameworks/CoreFoundation.framework/Versions/A/CoreFoundation (compatibility version 150.0.0, current version 1673.126.0)

/System/Library/Frameworks/CoreGraphics.framework/Versions/A/CoreGraphics (compatibility version 64.0.0, current version 1348.12.4)

/System/Library/Frameworks/CoreText.framework/Versions/A/CoreText (compatibility version 1.0.0, current version 1.0.0)

/System/Library/Frameworks/Foundation.framework/Versions/C/Foundation (compatibility version 300.0.0, current version 1673.126.0)

/System/Library/Frameworks/AppKit.framework/Versions/C/AppKit (compatibility version 45.0.0, current version 1894.10.126)

/System/Library/Frameworks/Security.framework/Versions/A/Security (compatibility version 1.0.0, current version 59306.41.2)

/usr/lib/libbsm.0.dylib (compatibility version 1.0.0, current version 1.0.0)

/System/Library/Frameworks/DiskArbitration.framework/Versions/A/DiskArbitration (compatibility version 1.0.0, current version 1.0.0)

/System/Library/Frameworks/IOKit.framework/Versions/A/IOKit (compatibility version 1.0.0, current version 275.0.0)

/System/Library/Frameworks/OpenGL.framework/Versions/A/OpenGL (compatibility version 1.0.0, current version 1.0.0)

/System/Library/Frameworks/AGL.framework/Versions/A/AGL (compatibility version 1.0.0, current version 1.0.0)

/usr/lib/libc++.1.dylib (compatibility version 1.0.0, current version 800.7.0)

/usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 1281.0.0)

Windows

ADVAPI32.dll

USER32.dll

dwmapi.dll

MSVCP140.dll

ole32.dll

KERNEL32.dll

VCRUNTIME140.dll

api-ms-win-crt-runtime-l1-1-0.dll

api-ms-win-crt-string-l1-1-0.dll

api-ms-win-crt-convert-l1-1-0.dll

api-ms-win-crt-stdio-l1-1-0.dll

api-ms-win-crt-heap-l1-1-0.dll

api-ms-win-crt-time-l1-1-0.dll

api-ms-win-crt-math-l1-1-0.dll

api-ms-win-crt-filesystem-l1-1-0.dll

api-ms-win-crt-locale-l1-1-0.dll

SHELL32.dll

A.2 Third-Party Libraries

Listed below are the third-party libraries used by the TOE.

All Platforms

openssl: OpenSSL License; Original SSLeay License

Android

Apache 2:

All official Google-provided Android libraries (Support libs, UI libs, FCM, etc)

Kotlin - https://github.com/JetBrains/kotlin/tree/master/license

Kotlin coroutines - https://github.com/Kotlin/kotlinx.coroutines/blob/master/LICENSE.txt

Anko - https://github.com/Kotlin/anko/blob/master/LICENSE

Leak Canary - https://github.com/square/leakcanary/blob/main/LICENSE.txt

Butterknife - https://github.com/JakeWharton/butterknife/blob/master/LICENSE.txt

Otto - https://github.com/square/otto/blob/master/LICENSE.txt

Timber - https://github.com/JakeWharton/timber/blob/master/LICENSE.txt

OkHttp - https://github.com/square/okhttp/blob/master/LICENSE.txt

Retrofit - https://github.com/square/retrofit/blob/master/LICENSE.txt

RxJava - https://github.com/ReactiveX/RxJava/blob/3.x/LICENSE

RxBindings - https://github.com/JakeWharton/RxBinding/blob/master/LICENSE.txt

AppAuth - https://github.com/openid/AppAuth-Android/blob/master/LICENSE

ZXing - https://github.com/zxing/zxing/blob/master/LICENSE

Process Phoenix - https://github.com/JakeWharton/ProcessPhoenix/blob/master/LICENSE.txt

Gson - https://github.com/google/gson/blob/master/LICENSE

CircleImageView - https://github.com/hdodenhof/CircleImageView/blob/master/LICENSE.txt

PhotoView - https://github.com/chrisbanes/PhotoView/blob/master/LICENSE

Slidr - https://github.com/r0adkll/Slidr/blob/master/LICENSE.md

AndroidPdfViewer - https://github.com/barteksc/AndroidPdfViewer/blob/master/LICENSE

SpyGlass - https://github.com/linkedin/Spyglass/blob/master/LICENSE

MjolnirRecyclerView - https://github.com/infinum/MjolnirRecyclerView/blob/master/LICENSE

Transitions Everywhere - https://github.com/andkulikov/Transitions-Everywhere/blob/master/LICENSE

Country Code Picker - https://github.com/hbb20/CountryCodePickerProject/blob/master/License.txt

GreenRobot EventBus - https://github.com/greenrobot/EventBus/blob/master/LICENSE

FlowLayout - https://github.com/ApmeM/android-flowlayout

nv-i18n - https://github.com/TakahikoKawasaki/nv-i18n/blob/master/LICENSE

CWAC Camera - Unavailable now (mirror here https://wkrnjpgit.wickrlan.net/android-dev/cwac-camera)

MIT:

BugSnag - https://github.com/bugsnag/bugsnag-android/blob/master/LICENSE

JSoup - https://github.com/jhy/jsoup/blob/master/LICENSE

Gif Drawable - https://github.com/koral--/android-gif-drawable/blob/dev/LICENSE

Countly - https://github.com/Countly/countly-sdk-android/blob/master/LICENSE

Microsoft AppCenter - https://github.com/microsoft/appcenter-sdk-android/blob/develop/license.txt

eBson - https://github.com/kohanyirobert/ebson/blob/master/LICENSE.txt

GPLv3:

Psiphon - https://github.com/Psiphon-Labs/psiphon-tunnel-core/blob/master/LICENSE

Other:

SQLCipher - https://www.zetetic.net/sqlcipher/license/

Protobuf - https://github.com/protocolbuffers/protobuf/blob/master/LICENSE (BSD according to Wikipedia)

iOS

AFNetworking

AppAuth

Almofire

Bugsnag

CocoaLumberjack

Countly

Differentiator

Kingfisher

MBProgressHUD

Moya

Nimble

NVActivityIndicatorView

PanModel

Quick

RxBiBinding

RxSwift

RxCocoa

RxDataSource

RxStartScream

StartScream

SSkeychain

SwiftTryCatch

SwiftyJSON

ProtocolBuffer

PopupDialog

PsiphoneTunnel

ZXingObjc

macOS, Windows:

sentry-crashpad: Apache-2.0 License

googlemaps: MIT License

Hunspell: LGPL 2.1

libbson: Apache-2.0 License

protobuf: https://github.com/protocolbuffers/protobuf/blob/master/LICENSE

Qt: LGPLv3 (commercial support)

QuaZip: LGPL 2.1

qzxing: Apache-2.0 License

sparkle: https://github.com/sparkle-project/Sparkle/blob/master/LICENSE

SQLCipher: https://github.com/sqlcipher/sqlcipher/blob/master/LICENSE

Twitter twemoji: MIT license

winsparkle: https://github.com/vslavik/winsparkle/blob/master/COPYING

YAJL: ISC License

zlib: https://zlib.net/zlib_license.html

zxing: Apache-2.0 License