Trustner

Security Whitepaper

July 2020

Table of Content

1. Background

Trustner is a simple and secure communication service.

In the modern, intertwined 21st century it is understood that there is no absolute security on the Internet. We at Trustner take all sorts of risks into consideration; which led us to develop a client suited for an environment that can only be as safe as the corresponding device it is running on.

Data theft is nowadays considered one of the biggest security risks when navigating in today's digital society. Trustner is a major improvement in electronic communication as it routinely encrypts data in a way that is considered state of the art. The overall risk of data infringement can be reduced considerably through simple tasks like regularly updating antivirus software. If in extreme cases the infrastructure of Trustner is compromised by an outside force, the attacker cannot easily gain access to user content. The private keys used for encryption and decryption are only stored on registered user devices.

Trustner is a German based company where the legislation on data protection is well developed. The location is seen a competitive advantage.

It is important to understand that Trustner is not suited for anonymous communication. Primary use-cases include the electronic exchange among professionals in healthcare, the legal sector, banking and other sectors where customer confidentiality is required. Some Trustner services require audit-proof authentication of users.

In privacy-sensitive areas like healthcare, German law requires so called Data Protection Concepts (DPCs) that are to be presented to the privacy protection authorities before software can be put into use. Trustner has prepared a DPC for the "Trustner Medical Apps" which can be used by doctors and physicians who want to communicate with colleagues, patients, their families and other medical staff.

The Trustner DPC has been presented to data protection authorities in several Bundesländer (federal states), where Trustner has been piloted.

This document mainly describes the encryption concept of Trustner, which is an important part of the DPC. The encryption concept itself has been developed in cooperation with a senior encryption expert from the Host Görtz Institut für IT-Sicherheit at the Ruhr-Universität Bochum, one of the leading German research institutions for data security.

For further detailed information, do not hesitate to contact us under support@trustner.com

2. The Trustner Platform

2.1. Some Definitions

2.1.1. The API

The Trustner API provides a RESTful interface for the Trustner platform. The platform stores encrypted content and relationships between content and user, i.e. the link between a post, a room or a chat.

2.1.2. Apps

While using Trustner, the user can choose a variety of different communication apps (or “room types”). Some Apps allow data uploads; other Apps delete content automatically after some time. There are Apps for doctors which can only be used with personal identification. Apps run on a web server and use Trustner platform services. Trustner Apps can also be hosted and operated by competent third parties.

2.1.3. Client

We offer Trustner clients for various devices and operating systems (Windows, OSX, iOS, Android). The encryption component of the client is called KeyBox.

2.1.4. Core

The Trustner core consists the basic user-interface outside the App interface. The responsibility lies with Trustner.

2.1.5. Instance, “Room” or Chat

A Trustner app can have an unlimited number of instances that are “rooms”. A room is virtual meeting place where users communicate or share data. Every room can have an account-specific user with an own user profile. A chat is a basic room for direct communication between selected contacts.

2.1.6. Store

The Trustner Store is a place where the different Apps can be activated. Some Apps are free, while other include payment features.

2.1.7. Platform

The Trustner Platform manages user authentication, invitation mechanisms, app administration, storage and payment methods as services. It also stores and distributes the public keys of the registered users.

2.2. The Concept

Trustner is built as a platform. It enables developers to create communications apps for users with different privacy and contact requirements. Each App can run independently from another.

All apps, from internal as well as external developers, use the same API. However, the initial version has an API that is open only to in-house programmers. As the platform progresses, third party developer accounts become available.

All apps, regardless of the developer are accessible via the Trustner Store, which is a part of the core.

The main function of the platform is, in conjunction with the client, the provision of our encryption functionality.

An additional passphrase is encrypted and stored on the user device's hard disc only. This passphrase is used for the encryption and decryption of the private key. It is provided by the Trustner core after user registration and authentication.

Each device has an asymmetric key pair consisting of a private and public key.

The so-called resource key is a symmetric key for encryption and decryption of data within the Trustner app. This key is known by all members of an app instance. There is no limit to the number of resource keys in an app.

The Trustner client automatically manages the exchange of symmetric keys among the contacts of a room or chat

3. The Trustner Client

The encryption of the Trustner is client based and uses standard open source libraries.

The Trustner client organizes the encryption and decryption of shared content. It also handles the exchange of cryptographic keys between the Trustner apps. The resource keys are encrypted and stored in the client's local database.

The client looks and feels like a web browser and uses web technologies. It has exclusive access to Trustner and the Trustner apps. External URLs are handled by the web browsers.

3.1. Cryptography

Communication between Trustner and Trustner apps and the communication between Trustner apps and the terminal devices as well as vice versa, is encrypted by using TLS 1.2.

There is no master key for recovery. An optional procedure to provide a second key through a fully automated process with no access for the provider can be implemented at a later stage.

The symmetric encryption uses AES 256 (OFB mode) with OpenSSL Version 1.0.2g.

RSA with a key length of 2048 Bit is used for asymmetric encryption of the resource key. Elliptic curves are considered a viable alternative to RSA.

Encryption algorithms are versioned to allow an easy replacement if necessary.

3.2. Operating Mode

The download of a Trustner client (with the KeyBox as encryption component) initiates the registration procedure. A repeated login after a successful authentication opens a user-specific client session. A session-token is generated on the server, which is then relayed to the KeyBox by loading the site by using a JavaScript invocation.

The session token establishes an association between the user account and the KeyBox.

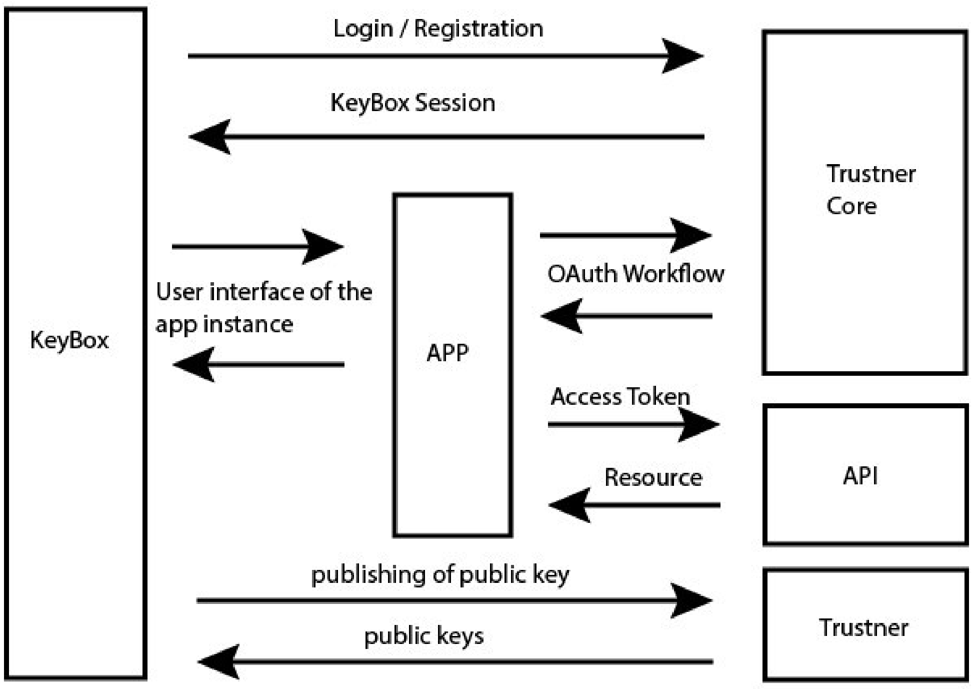

Figure 3.1.: Exchange between App, Core, API, KeyBox and Trustner

Figure 3.1. shows the exchange including the OAuth workflow for third party apps, which has not been implemented.

3.2.1. Case 1: A public / private key pair does not yet exist

On the user's device, a public/private key pair is generated in the KeyBox. The default setting is the RSA 2048 key pair.

The required time for key generation depends on the entropy-pool of the user's computer.

After successful generation, the public key is rolled out, the private key remains in the client encrypted with the secret.

An indicator at the lower margin of the client-window notifies the user, when encryption is available.

3.2.2. Case 2: There is already a public / private key pair for the user

The private key is encrypted and secured. The session token enables the KeyBox to decrypt the private key through a passphrase. Private keys remain exclusively on the user's devices.

There are two options envisioned for the generation and storage of passphrases:

- In normal mode, the passphrase is generated by the client and stored on the server. This distinct separation of access-relevant data protects the encrypted private key from offline-attacks; the passphrase as counterpart cannot be retrieved. The login procedure (password input) retrieves the passphrase from the server. A high-level password quality is enforced during the account registration as additional safety feature.

- In advanced mode, the user can choose the passphrase individually. The passphrase is always located with the user in this scenario. A second password exists for the decryption of the private key in addition to the standard password used for login.

Resource keys can only be decrypted by using the (also decrypted) private key. Room content can therefore only be displayed by using the decrypted combination of private and resource keys. The resource key is a symmetric key that in turn also encrypts content. Resource keys are encrypted with the public keys of all recipients.

In case a KeyBox is missing a resource key, it can issue a key exchange request to the other KeyBoxes associated with the app instance.

3.3. Trustner Client API

The KeyBox API is accessible for apps via a JavaScript bridge. It is accessible only through a Trustner client view and can only be used through the Trustner apps and core. The API includes methods for encryption, decryption and key exchange.

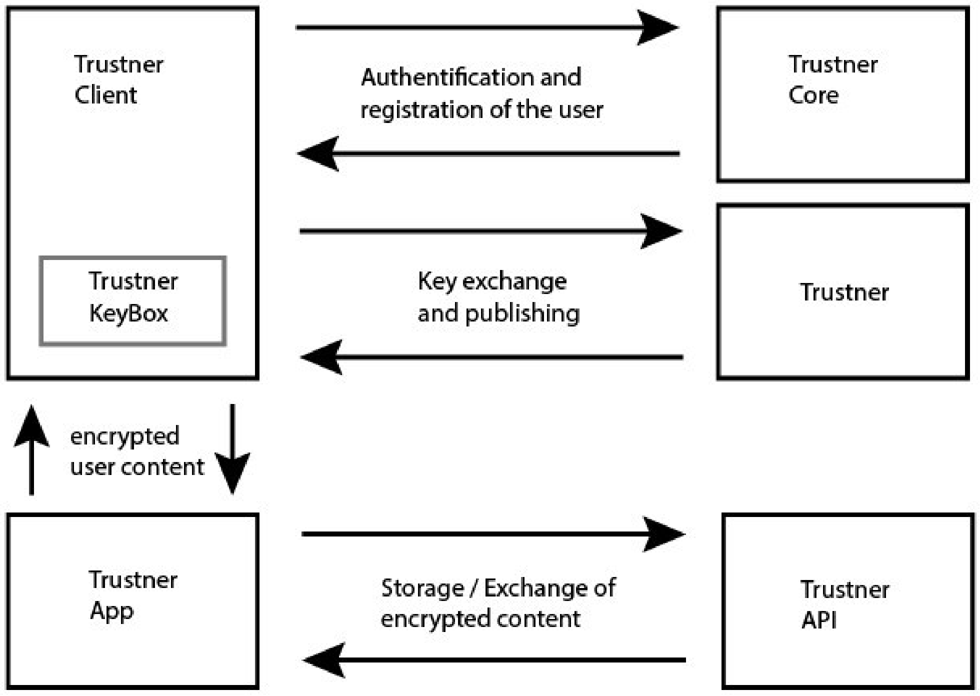
Trustner Client
Trustner KeyBox
Authentification and registration of the user
Trustner Core
Key exchange and publishing
Trustner
encrypted user content
Trustner App
Storage / Exchange of encrypted content
Trustner API

Figure 3.21: Operating Mode of Trustner KeyBox

3.4. Trustner Client and Authentication Protocol

The Trustner KeyBox is associated with a user account after either the login or registration process. The KeyBox relates to the account through unique identifier: the KeyBox ID. Users can have several KeyBoxes and every KeyBox ID additionally includes a public-private-key pair.

A KeyBox installation can be shared by many users. Using the Trustner KeyBox on public computers, like internet cafes, is discouraged.

The private key is protected by the passphrase as mentioned above.

After login the Trustner KeyBox issues a Trustner API access token.

3.5. Key Exchange

Symmetric resource keys must be made available in order to communicate with two or more users in an app instance. These resource keys cannot be intercepted by either a third party or a Trustner administrator.

This security feature is ensured by encrypting the resource key with the public key of all room members inside an app instance. The KeyBox on the terminal device performs the encryption. It is crucial to note that the KeyBox needs to know the public keys of every instance user in order to encrypt the symmetric resource key. The public key of each KeyBox is transferred to the Trustner server after being generated.

KeyBoxes can request public keys; this method applies when a new member enters a room / app instance or when a new device is added to an existing account. The new KeyBox needs the public keys in order to exchange resource keys.

The AES 256 key is stored as a 64-character long hexadecimal representation. The key is generated by using the devices random-number-generator system. Apple, Android and Windows devices use the OpenSSL random number generation without the dysfunctional Dual Elliptic Curve Deterministic Random Bit Generator (DUAL EC DRBG).

A newly generated resource key is encrypted for local storage by using the public key of the users KeyBox. The resource key is encrypted for each instance user (with their corresponding public keys) before being distributed. The permission to deposit an encrypted resource key is granted only to the members of the app instance. A permission check is carried out by the Trustner platform.

Every key exchange is logged in the user’s account. Advanced users can check these logs in order to see who issued a key to whom. Advanced users may also be able to individually permit a key exchange at a later stage of the client.

The decryption of deposited keys is only possible by using the private key of the KeyBox.

3.6. Additional Clients

Users can use several devices to access Trustner apps. Each device requires a Trustner client download. Access is possible from all registered devices through the login credentials.

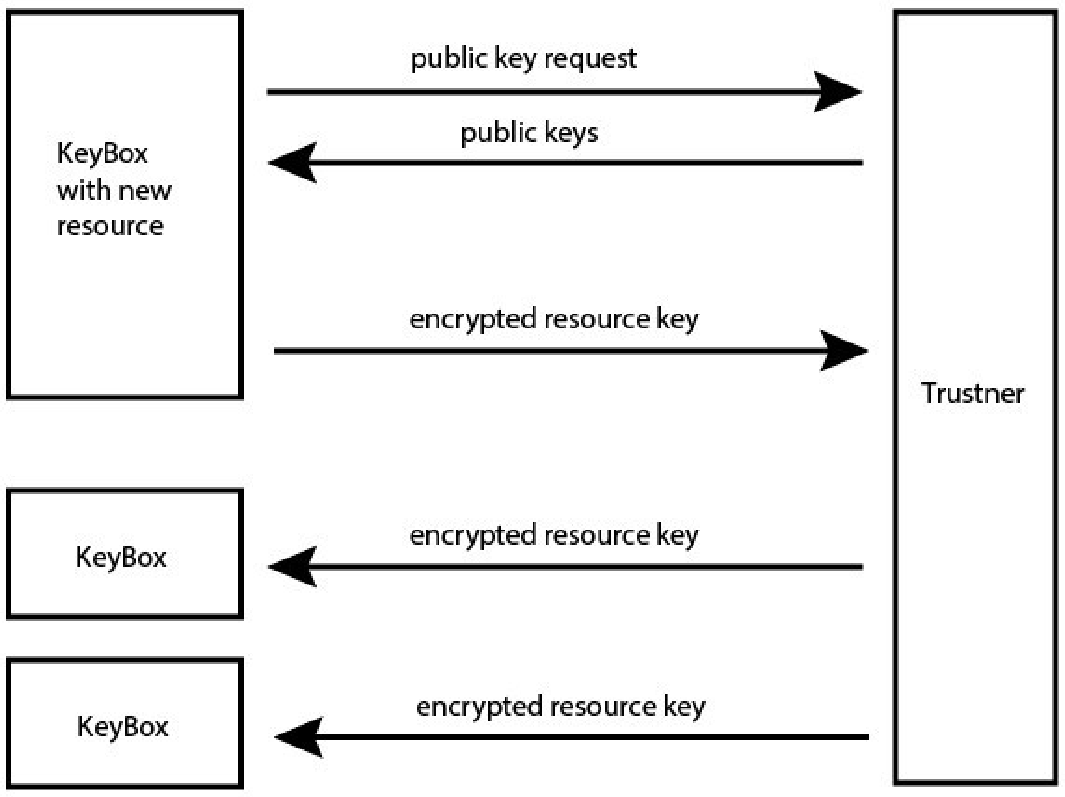

Figure 3.3: Exchange of Resource Key to new KeyBox

An additional device is treated like an additional contact in a room when it comes to key exchanges. Users can see and manage all registered devices once logged into their accounts. A protocol is generated for every newly registered device and stored on the server.

4. Risk Assessment

Trustner offers automatic end-to-end encryption for all communication content. It therefore stands for a high level of protection against unwanted insights of third parties. The actual security level, however, depends on several additional factors.

4.1. Weak Password

A weak password leaves the risk of losing confidential information and privacy within Trustner. To prevent this, certain password requirements must be meet when registering a new account or when changing the password over the account settings. Trustner passwords need to fulfill the *Bundesamt für Sicherheit in der Informationstechnik* (BSI) requirements: Passwords must have a minimum of eight lower- and upper-case characters, need to include special characters and numbers or have to be longer than twenty characters.

There is a time-out after three failed login attempts.

Passwords are stored with a one-way hash function (bcrypt) and are hardened with an additional individual Salt.

4.2. Password Theft

Even a strong password can be stolen when the device of the user is compromised.

The only defense is to create awareness at user level. A user should never login to Trustner on a computer he or she does not trust.

The computer of the user needs to be regularly updated. An up-to-date antivirus protection program will reduce the risk of password theft. Regular password changes are strongly recommended.

4.3. Compromised Client-Server Communication

The risk of a man-in-the-middle attack is highest in Wi-Fi zones and public internet access points, where others can interfere without being noticed, is high. In this case the attacker can compromise the client-server communication.

Trustner uses TLS 1.2 as a defense. However, given the situation with compromised certificates, a risk remains.

This risk is addressed through the option to check the fingerprint of the certificate.

4.4. Limited Randomness

Strong encryption relies on randomness for key generation. If the random key space is somehow limited, i.e. through a manipulated entropy pool, a private key may be brute forced.

The OpenSSL random generator promises a sufficient degree of randomness.

4.5. Bad Trustner Client

There is no security when the user installs a Trustner client that has been tampered with. This can happen when the user downloads the client from a fake website.

It is therefore important that the user goes to the official Trustner website each time he logs on to Trustner.

Trustner publishes the fingerprints of the binaries via HTTPS. An advanced user may check those to check the integrity of the binary.

4.6. Compromised Client Device

Trustner has no control over the devices where the client is downloaded. While professional organizations may have staff to secure infrastructure and hardware, private users usually do not have that protection.

A compromised client device is hard to detect. The best way to avoid this is to raise user awareness for regular software updates. Also, a current antivirus program can help.

4.7. Compromised Trustner Infrastructure

An unnoticed attack on the Trustner infrastructure could result in serious damage. It would enable the attacker to target individual user accounts and their contacts and thus potentially all Trustner users.

The attacker would have to gain access to the resource keys by adding KeyBoxes to compromised accounts. Since all key exchanges are logged on the client side, the user might eventually notice the addition of unauthorized KeyBoxes. A skillful attacker, however, may find a way to conceal the attack.

Limiting the number of users per room / app instance can prove to be a safeguard towards adding compromised clients. The option is offered as “close room or chat”.

5. Open Source Policy

The Trustner security concept is based on transparency and comprehensibility. Trustner uses only standard libraries and standard protocols. Later, the code of the key management process in the Trustner client will be published for public review.

6. Contact

Dr. Harald Sondhof

Trustner GmbH
Österbergstraße 9
72074 Tübingen
Germany

www.trustner.com