TECHNICAL OVERVIEW

Phrase-Lock

Your smartphone as a universal key –
for every account and every software application.

Password, smart card and passkey – on one device.



phraselock.com

github.com/phraselock

Version 1.1

Contents

Phrase-Lock®

Part 1 – Overview 3

Three ways to sign in – one device, the smartphone 3

Your own infrastructure instead of someone else's cloud 3

Architecture 3

What Phrase-Lock does on the computer 4

Two certificate worlds (PKI) 4

Security model at a glance 4

Part 2 – In Detail 5

Entering complex passwords securely 5

Smart card (PIV) 5

FIDO2 / passkeys 6

Smartphone apps 6

Structure per platform 7

Windows 7

Linux 7

macOS 8

Connection between smartphone and computer 8

Middleware and operation 9

Security in layers 9

Standards and interfaces 10

What is stored where 10

Try it in two steps 10

Part 1 – Overview

Phrase-Lock brings every kind of sign-in to a single device: the smartphone. Complex passwords, one-time passwords, smart card and FIDO2 – for Windows, Linux and macOS, right up to the sign-in screen.

Three ways to sign in – one device, the smartphone

Complex passwords

The smartphone types long, random passwords – into any input field, even on the sign-in screen.

Nobody needs to know them.

Smart card (PIV)

Certificate sign-in to the domain, Remote Desktop, signing, SSH.

The smartphone is the card.

FIDO2 / passkeys

Phishing-resistant sign-in in browsers, in web applications and to Windows (Entra ID).

The smartphone is the key.

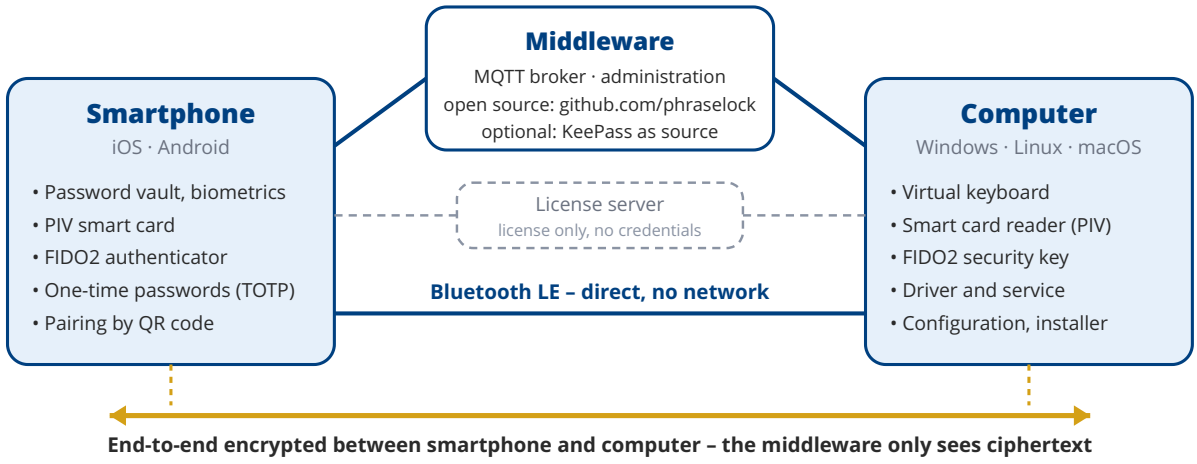
Your own infrastructure instead of someone else's cloud

- **Control over the data** lies solely with the user.
- Everything between smartphone and computer is **open source** (github.com/phraselock) and runs on the user's own systems – **no third party, no cloud**.
- **Adaptable:** e.g. a data source other than KeePass.
- Or **no server at all:** directly via Bluetooth.

Architecture

Phrase-Lock connects two endpoints: the **smartphone** and the **computer**. Everything in between is infrastructure the user owns.

- **Smartphone (iOS, Android):** vault for passwords and keys, release by biometrics, PIV smart card, FIDO2 authenticator, one-time passwords.
- **Computer (Windows, Linux, macOS):** driver and service provide the operating system with a keyboard, a smart card reader and a FIDO2 security key.
- **Connection:** directly via Bluetooth LE – without a network – or via the middleware over MQTT with mutual certificate authentication (mTLS).
- **Pairing:** smartphone and computer are paired by QR code – also many-to-many (n:m): one smartphone with many computers, one computer with many smartphones.
- **Middleware:** MQTT broker and administration, open source, run by the user; optionally KeePass as the source of credentials.
- **End-to-end encrypted** between smartphone and computer – the middleware only sees ciphertext.
- **License server:** only checks the licenses of smartphone and computer – it never sees credentials.



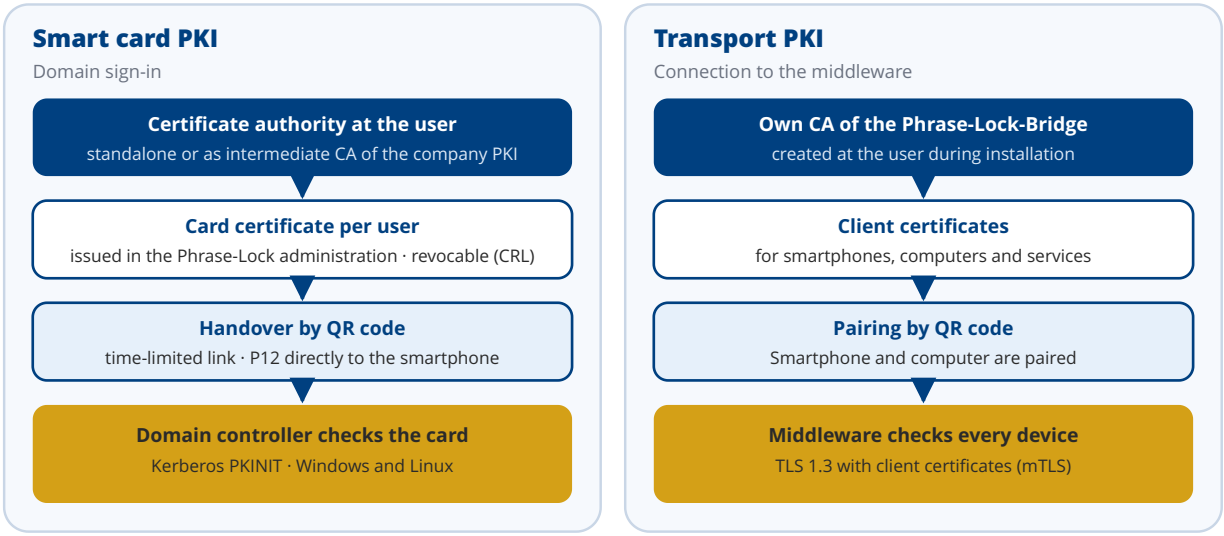
What Phrase-Lock does on the computer

Function	Windows	Linux	macOS
Enter complex passwords securely (virtual keyboard)	✓	✓	✓
Enter user names and one-time passwords (TOTP)	✓	✓	✓
Sign-in and lock screen: sign in with password	✓	✓	–
Smart card (PIV): smartphone as card in a virtual reader	✓	✓	–
Sign-in and lock screen: sign in with smart card	✓	✓	–
Domain sign-in with smart card (Active Directory, Kerberos PKINIT)	✓	✓	–
Remote Desktop with the smartphone's smart card	✓	–	–
FIDO2 / passkeys: smartphone as security key	✓	✓	✓
Windows sign-in with FIDO2 (Entra ID)	✓	–	–

Connection to the smartphone

Connection	Windows	Linux	macOS
Bluetooth LE – direct, no network	✓	✓	✓
MQTT over mTLS – via your own middleware	✓	✓	✓

Two certificate worlds (PKI)



Distribution: the certificates of both worlds are distributed by the user's own infrastructure (Phrase-Lock administration) – no certificate passes through a third party. The smartphone only scans a QR code with limited validity. It contains no certificate, just a reference the smartphone uses to fetch its certificate directly from the user's own infrastructure.

Security model at a glance

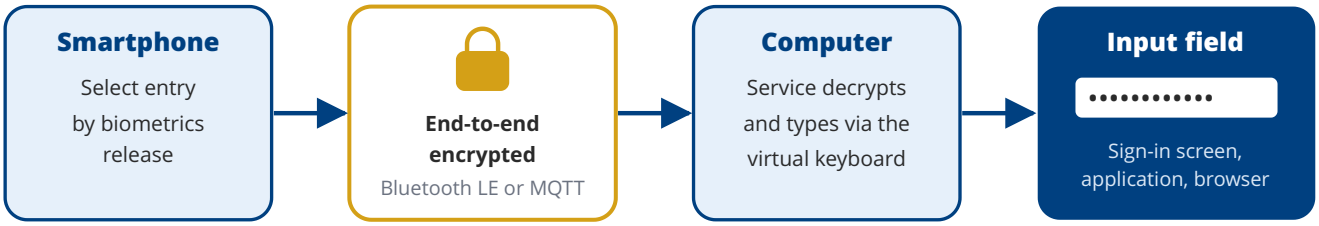
- **Secrets stay on the smartphone.** Every use is released there by biometrics.
- **End-to-end encryption** between smartphone and computer.
- **Mutual authentication** at the middleware with client certificates (mTLS).
- **Autonomous** via Bluetooth LE – no network, server or cloud.
- **Own infrastructure:** middleware open source, run by the user.
- **Standards instead of custom builds:** to the operating system, Phrase-Lock is a keyboard, a smart card or a FIDO2 key – applications need no changes.

> **Open source:** middleware and installers for your own infrastructure are publicly available at **github.com/phraselock**.

Part 2 – In Detail

Entering complex passwords securely

In practice, strong passwords fail at typing: they are chosen short, written down or reused. Phrase-Lock takes the typing off the user's hands.

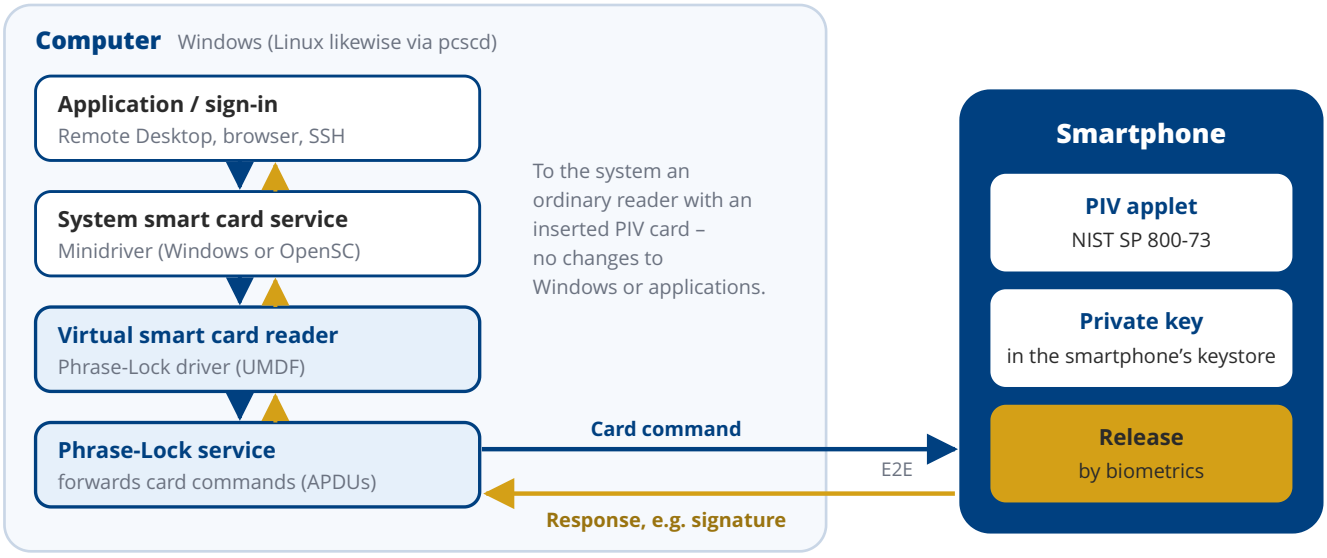


- **Any length, any complexity** – nobody has to know, remember or type the password.
- **Wherever a keyboard works** – without changes to the application, even before sign-in.
- **No sticky note, no clipboard** – the password leaves the vault only encrypted.

- The password is kept in the **app's vault**, selected on the smartphone and released by **biometrics**.
- It travels **end-to-end encrypted** to the computer; the service enters it via the **virtual keyboard**.
- To the operating system this is an ordinary keyboard: it works in **every input field** – sign-in screen, applications, browser, console – without changes.
- Because the service runs **before sign-in**, this also works on the Windows sign-in screen.
- User name, password and one-time password (TOTP) can be entered one after another.
- Optionally, an existing **KeePass database** on your own server serves as a shared source; each smartphone requests the entries assigned to it.

Smart card (PIV)

The smartphone becomes a smart card according to **PIV** (NIST SP 800-73). The computer gets a **virtual card reader** holding the smartphone's card.

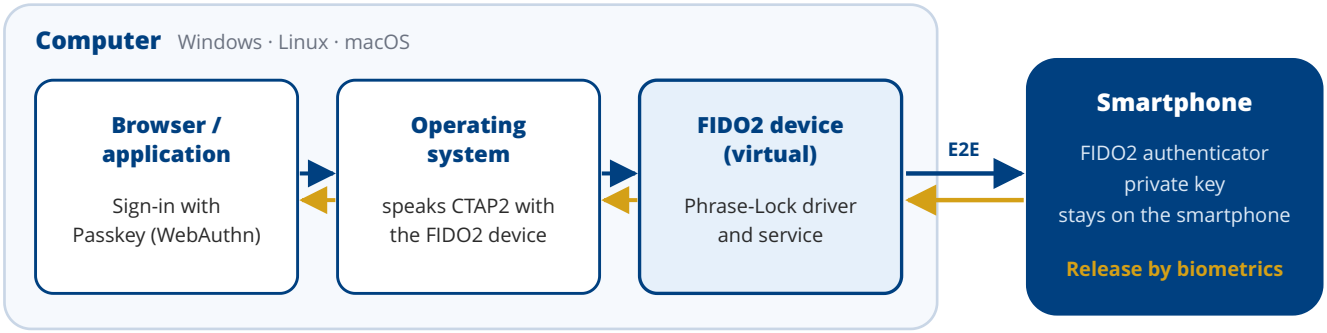


- To Windows and Linux it is an **ordinary reader with a PIV card** – sign-in, Remote Desktop, signing, SSH and browsers use it without changes.

- The **private key** is created in the user's own administration, because iOS cannot create a certificate signing request for RSA. It is transferred once, protected by mTLS and end-to-end encryption, and stored in the smartphone's keystore; every use is released there by biometrics.
- **Domain sign-in** to Active Directory via Kerberos PKINIT with certificates from the company CA.
- **Keys:** RSA 2048 with the built-in Windows minidriver; RSA 3072 and ECC P-256 with the OpenSC minidriver.
- **Remote Desktop:** the card on the client signs the user in on the remote computer – also from outside the company network through a tunnel without an open port.

FIDO2 / passkeys

The smartphone becomes a **FIDO2 security key** (CTAP2). The computer sees a FIDO2 device, just like a hardware key – with biometrics and the smartphone's vault.



- **Browsers and web applications:** passkeys and security keys according to WebAuthn – phishing-resistant.
- **Windows sign-in with Entra ID:** the smartphone serves as the FIDO2 security key for signing in to Windows.
- **No changes needed:** to the system it is an ordinary FIDO2 security key over USB HID; browsers, web applications and tools such as libfido2 use it like a hardware key.
- **Private keys** stay on the smartphone; every use is released there by biometrics.

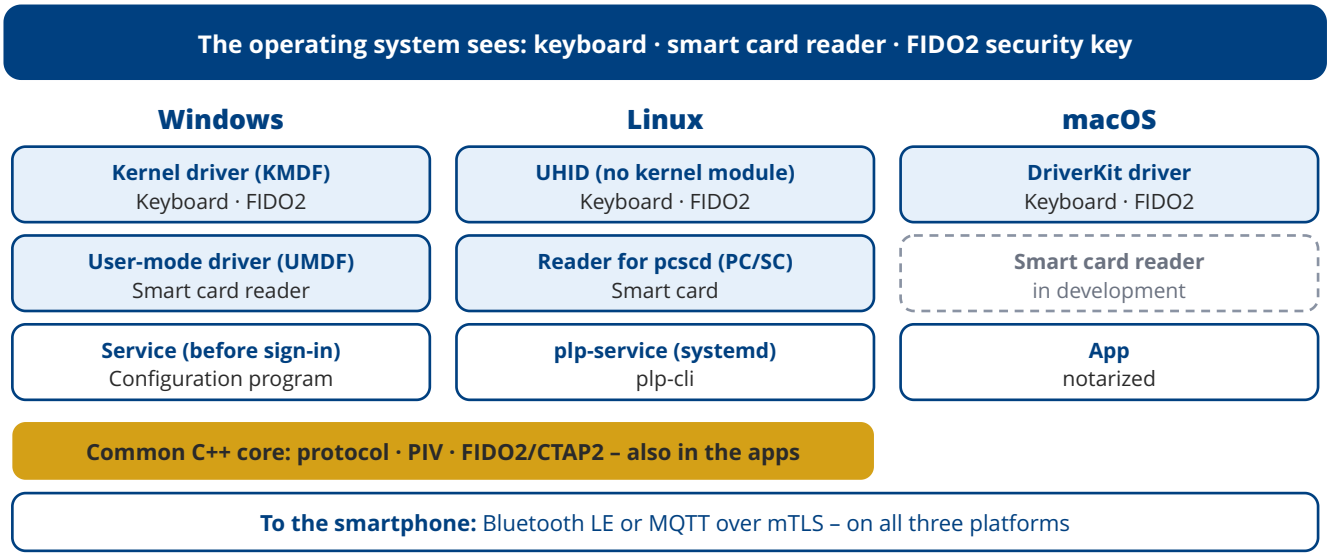
Smartphone apps

- Native apps for **iOS** and **Android**.
- **Vault** for passwords, one-time passwords (TOTP) and keys; every use released by **biometrics**.
- **PIV smart card** and **FIDO2 authenticator** (CTAP2) run in the app; private keys never leave the smartphone.
- **Pairing** with computer and server by QR code.
- Connection to the computer via **Bluetooth LE** or **MQTT**.

Further features

- **Autofill support**
- **Custom encryption**
- **Biometric protection**
- **Peer-to-peer data exchange**
- **Unlimited smart cards** on the smartphone
- **Unlimited FIDO2 credentials**
- **Backup and restore** – including FIDO2 credentials

Structure per platform



Common core and protocol

- A portable **C++ core** is used on Windows, Linux and the smartphones: Phrase-Lock protocol, PIV applet, FIDO2/CTAP2 incl. CBOR.
- The **PIV applet** is plain C++17 without platform dependencies: APDU in, response out. Cryptography, user interaction and storage are provided by each platform.
- **Protocol and encryption are independent of the transport:** Bluetooth LE and MQTT carry the same messages.
- **Device binding:** the license is bound to the respective device.

Windows

Drivers

- **Kernel driver (KMDF):** virtual keyboard and virtual FIDO2 device (HID).
- **User-mode driver (UMDF):** virtual smart card reader. Windows sees an ordinary reader with a PIV card; card commands go through the service to the smartphone. An error in the user-mode driver only ends the driver process, not the system.
- Both drivers are **signed by Microsoft** (attestation), for **x64 and ARM64**.

Service and configuration

- The **service** runs before sign-in (session 0) and keeps the connection to the smartphone: keyboard, smart card, FIDO2, Bluetooth LE and MQTT.
- A **configuration program** for pairing and settings.
- **Installer** (setup or MSI) for drivers, service and configuration.

Linux

- **No kernel module of its own:** keyboard and FIDO2 device are created in user space via **UHID**. Browsers, libfido2 and PAM cannot tell them apart from real hardware.
- **One process:** `plp-service` (systemd) combines devices and service logic; it starts before the sign-in screen.
- **Smart card:** own reader for **pcscd** (PC/SC). Through it, OpenSC, SSH, browsers and **domain sign-in** at the sign-in screen work (GDM, Kerberos PKINIT).
- **Operation:** command-line tool `plp-cli` (pairing by QR code, status, settings) instead of a graphical interface.
- **Cryptography:** OpenSSL 3.
- **Packages:** `.deb` and `.tar.gz`.

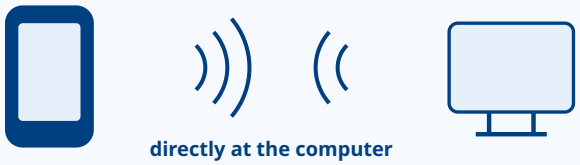
macOS

- **DriverKit driver:** virtual keyboard and virtual FIDO2 device (HID), without a kernel extension.
- **App** for pairing, settings and connection to the smartphone.
- **Notarized** by Apple.
- **Smart card:** in development (virtual USB card reader); requires an entitlement from Apple.

Connection between smartphone and computer

Bluetooth LE

direct, no network and no server

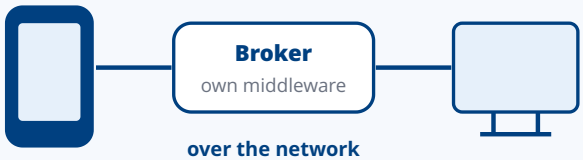


directly at the computer

isolated computers, OT

MQTT

via the middleware, TLS 1.3 with client certificates



over the network

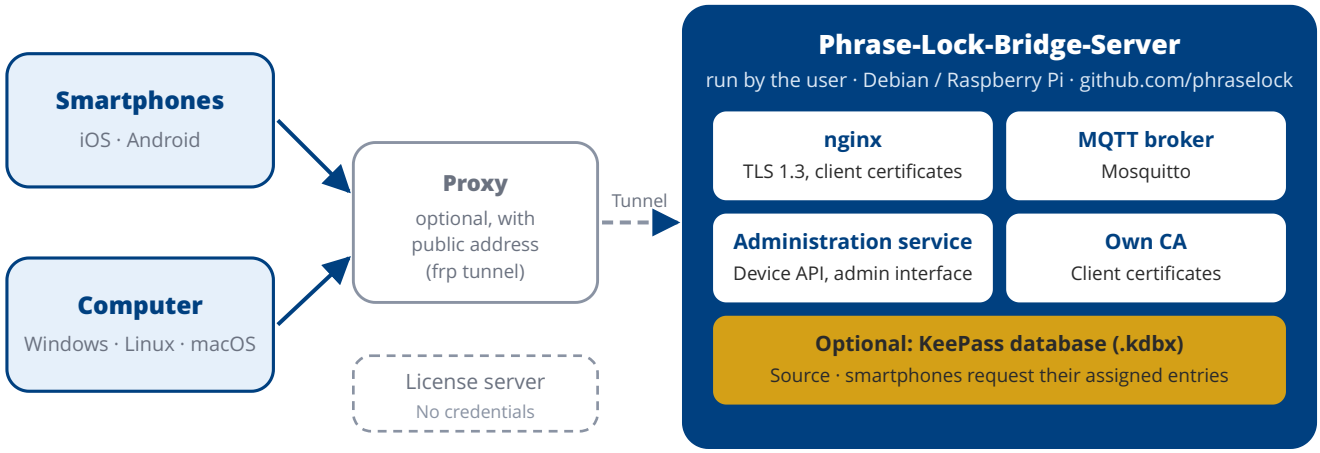
internet or LAN

	Bluetooth LE	MQTT
Network required	no	yes (internet or LAN)
Server required	no	MQTT broker (middleware)
Protection	end-to-end encryption of the messages	TLS 1.3 with client certificates + end-to-end encryption
Typical use	isolated computers, no network	computers on a network (internet or LAN)

Bluetooth LE in detail: the computer acts as a Bluetooth peripheral (GATT server), the smartphone connects to it. The built-in Bluetooth adapter or a standard USB Bluetooth dongle is enough. No pairing at operating-system level is needed – security comes from Phrase-Lock's end-to-end encryption.

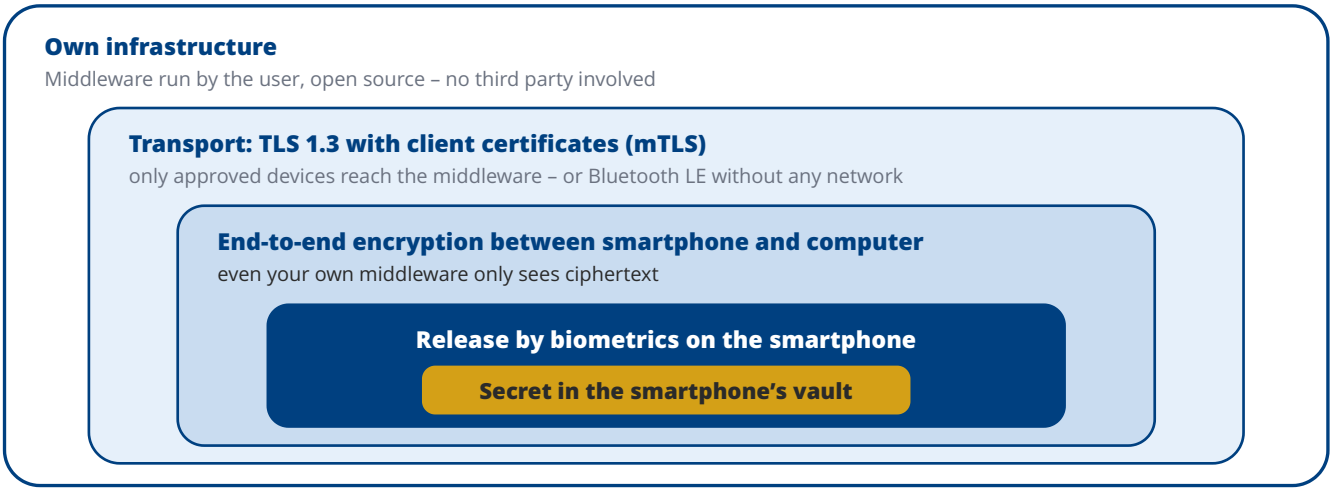
MQTT in detail: MQTT is a lean publish/subscribe protocol with minimal overhead – economical with battery, bandwidth and computing power, and designed for unstable mobile connections. Smartphone and computer only open **outgoing** connections to the broker: no port has to be opened on the computer, and it works behind NAT and firewalls. Dropped connections are re-established automatically. **Any MQTT broker** can be used; by default **Mosquitto** is used – open source, extremely lean and fast, it even runs on a Raspberry Pi. MQTT is an ISO standard (ISO/IEC 20922) established in industrial automation.

Middleware and operation



- **Phrase-Lock-Bridge:** installer for your own infrastructure (Debian, Raspberry Pi): MQTT broker (Mosquitto), nginx, own certificate authority for client certificates, administration service. Open source on GitHub.
- **Administration service:** API for the devices (client certificate only) and admin interface.
- **KeePass as source:** an existing KeePass database can serve as a shared source of credentials. Each smartphone requests the entries assigned to it – no automatic sync, no copy of the whole database on the smartphone.
- **No fixed IP address:** optional proxy (tunnel) for sites without a public address.
- **License server:** only checks the licenses of smartphones and computers – it never sees credentials.

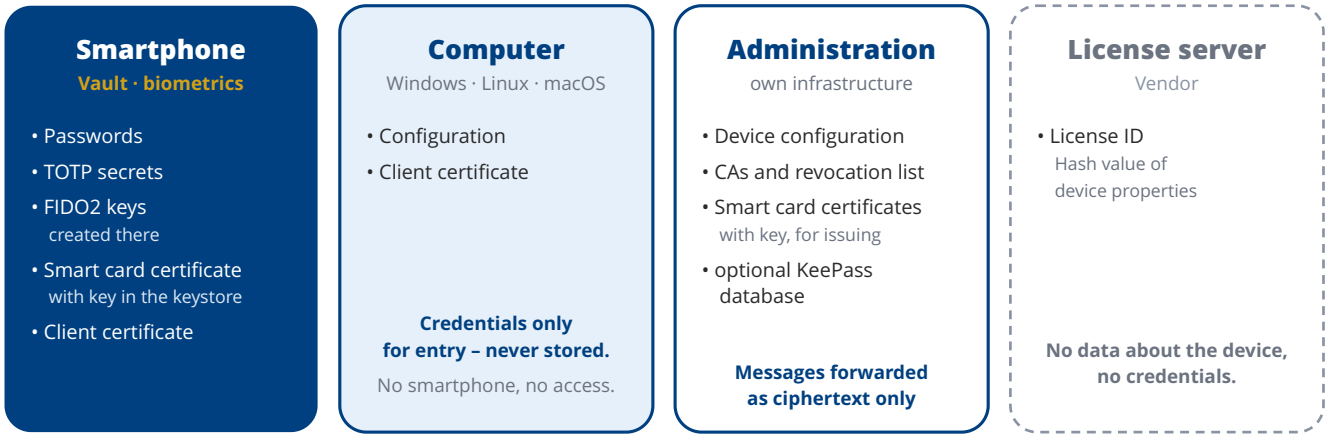
Security in layers



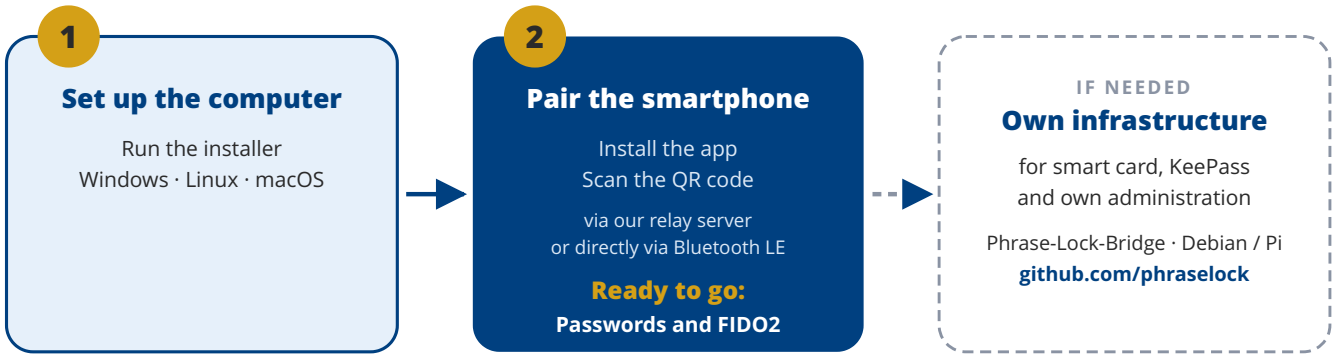
Standards and interfaces

Area	Standard
Password entry	USB HID keyboard (virtual)
Smart card	PIV according to NIST SP 800-73; RSA 2048/3072, ECC P-256
Card interface	Windows Smart Card (SCardSvr, minidriver), PC/SC on Linux
Domain	Active Directory, Kerberos PKINIT
FIDO2 / WebAuthn	CTAP2 authenticator over USB HID – browsers, web applications, Windows sign-in (Entra ID), libfido2
Transport	MQTT over TLS with client certificates; Bluetooth Low Energy (GATT)
Cryptography	RSA and ECC (PIV), ECDSA P-256 (FIDO2), TLS 1.3

What is stored where



Try it in two steps



Windows, Entra ID, macOS, iOS, Android, Bluetooth and all other trademarks mentioned are the property of their respective owners.