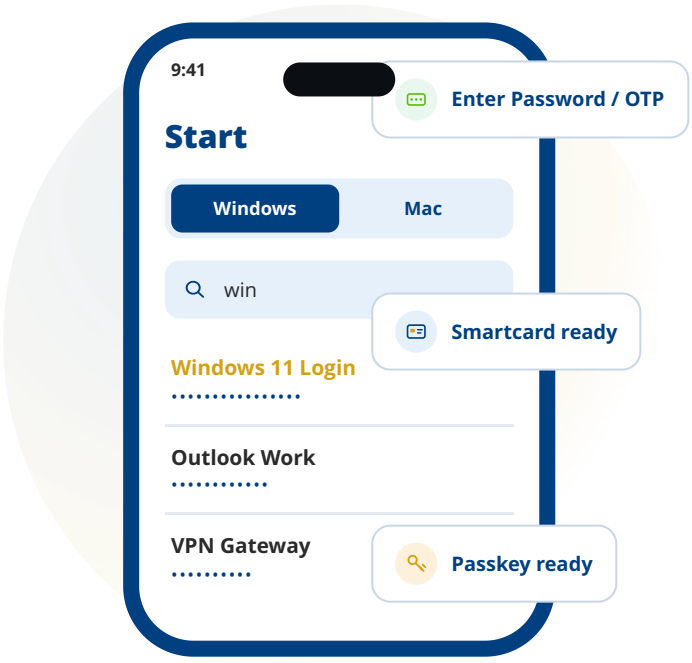
TECHNISCHER ÜBERBLICK

Phrase-Lock

Ihr Smartphone als Universalschlüssel – für jedes Konto und jede Softwareanwendung.

Passwort, Smartcard und Passkey – auf einem Gerät.



phraselock.com

github.com/phraselock

Version 1.1

Inhalt

Phrase-Lock®

Teil 1 – Überblick 3

Drei Wege der Anmeldung – ein Gerät, das Smartphone 3

Eigene Infrastruktur statt fremder Cloud 3

Architektur 3

Was Phrase-Lock auf dem Computer kann 4

Zwei Zertifikatswelten (PKI) 4

Sicherheitsmodell auf einen Blick 4

Teil 2 – Im Detail 5

Komplexe Passwörter sicher eingeben 5

Smartcard (PIV) 5

FIDO2 / Passkeys 6

Smartphone-Apps 6

Aufbau je Plattform 7

Windows 7

Linux 7

macOS 8

Verbindung zwischen Smartphone und Computer 8

Middleware und Betrieb 9

Sicherheit in Schichten 9

Standards und Schnittstellen 10

Was liegt wo 10

In zwei Schritten testen 10

Teil 1 – Überblick

Phrase-Lock bringt jede Form der Anmeldung auf ein einziges Gerät: das Smartphone. Komplexe Passwörter, Einmalpasswörter, Smartcard und FIDO2 – für Windows, Linux und macOS, bis zum Anmeldebildschirm.

Drei Wege der Anmeldung – ein Gerät, das Smartphone

Komplexe Passwörter

Das Smartphone tippt lange, zufällige Passwörter – in jedes Eingabefeld, auch am Anmeldebildschirm.

Niemand muss sie kennen.

Smartcard (PIV)

Zertifikatsanmeldung an der Domäne, Remote Desktop, Signatur, SSH.

Das Smartphone ist die Karte.

FIDO2 / Passkeys

Phishing-resistente Anmeldung im Browser, in Web-Anwendungen und an Windows (Entra ID).

Das Smartphone ist der Schlüssel.

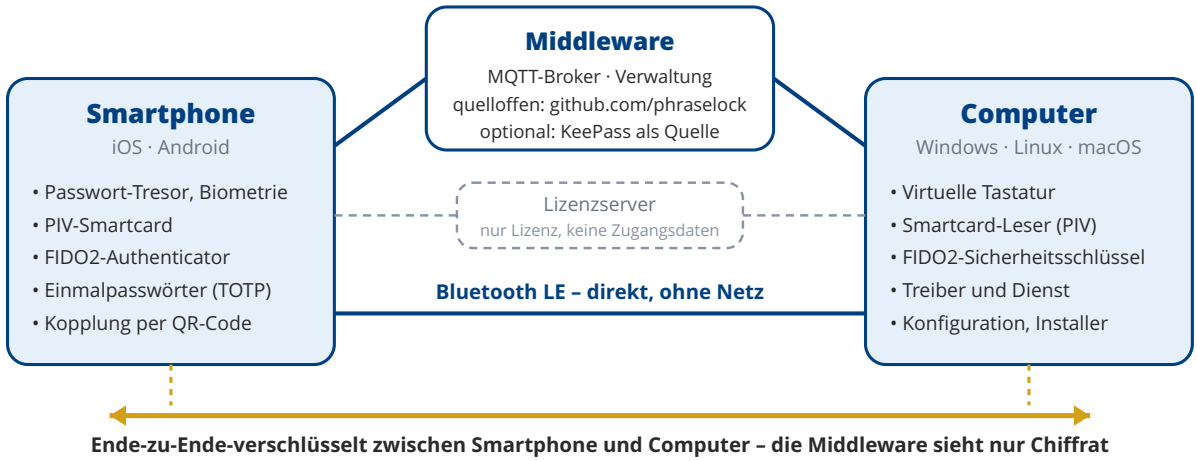
Eigene Infrastruktur statt fremder Cloud

- Die **Hoheit über die Daten** liegt ausschließlich beim Anwender.
- Alles zwischen Smartphone und Computer ist **quelloffen** (github.com/phraselock) und läuft beim Anwender selbst – **kein Dritter, keine Cloud**.
- **Anpassbar:** z. B. eine andere Datenbasis als KeePass.
- Oder **ganz ohne Server:** direkt per Bluetooth.

Architektur

Phrase-Lock verbindet zwei Endpunkte: das **Smartphone** und den **Computer**. Alles dazwischen ist Infrastruktur, die der Anwender selbst besitzt.

- **Smartphone (iOS, Android):** Tresor für Passwörter und Schlüssel, Freigabe per Biometrie, PIV-Smartcard, FIDO2-Authenticator, Einmalpasswörter.
- **Computer (Windows, Linux, macOS):** Treiber und Dienst stellen dem Betriebssystem eine Tastatur, einen Smartcard-Leser und einen FIDO2-Sicherheitsschlüssel bereit.
- **Verbindung:** direkt per Bluetooth LE – ohne Netz – oder über die Middleware per MQTT mit gegenseitiger Zertifikatsprüfung (mTLS).
- **Kopplung:** Smartphone und Computer werden per QR-Code einander zugeordnet – auch mehrfach (n:m): ein Smartphone mit vielen Computern, ein Computer mit vielen Smartphones.
- **Middleware:** MQTT-Broker und Verwaltung, quelloffen, beim Anwender selbst; optional KeePass als Quelle der Zugangsdaten.
- **Ende-zu-Ende-verschlüsselt** zwischen Smartphone und Computer – die Middleware sieht nur Chiffrat.
- **Lizenzserver:** prüft nur die Lizenzen von Smartphone und Computer – er sieht keine Zugangsdaten.



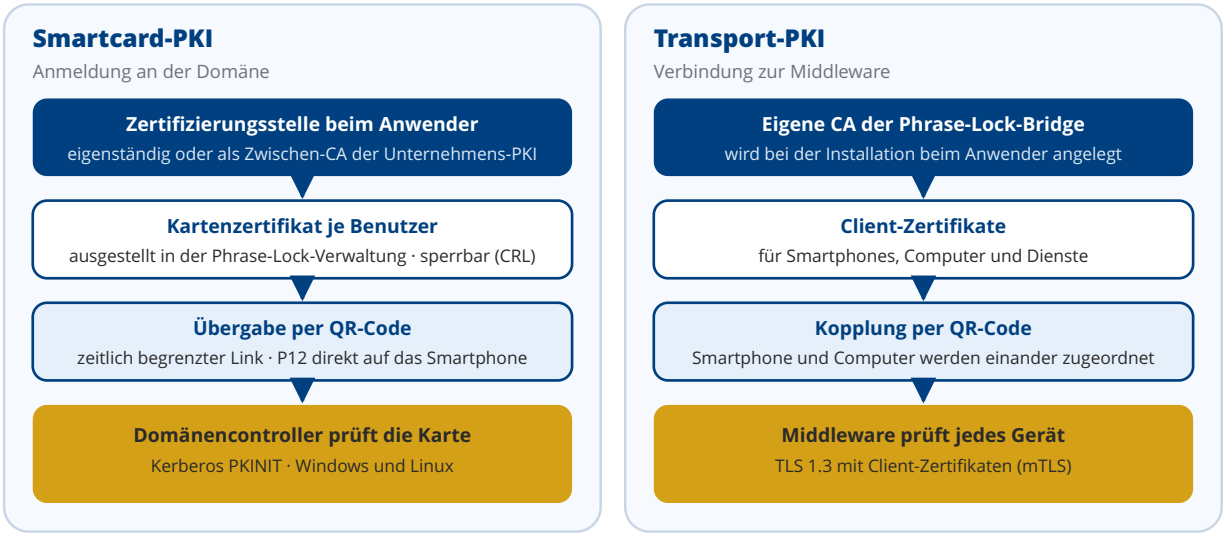
Was Phrase-Lock auf dem Computer kann

Funktion	Windows	Linux	macOS
Komplexe Passwörter sicher eingeben (virtuelle Tastatur)	✓	✓	✓
Benutzernamen und Einmalpasswörter (TOTP) eingeben	✓	✓	✓
Anmelde- und Sperrbildschirm: Anmeldung mit Passwort	✓	✓	–
Smartcard (PIV): Smartphone als Karte im virtuellen Leser	✓	✓	–
Anmelde- und Sperrbildschirm: Anmeldung mit Smartcard	✓	✓	–
Domänenanmeldung mit Smartcard (Active Directory, Kerberos PKINIT)	✓	✓	–
Remote Desktop mit der Smartcard des Smartphones	✓	–	–
FIDO2 / Passkeys: Smartphone als Sicherheitsschlüssel	✓	✓	✓
Windows-Anmeldung mit FIDO2 (Entra ID)	✓	–	–

Verbindung zum Smartphone

Verbindung	Windows	Linux	macOS
Bluetooth LE – direkt, ohne Netz	✓	✓	✓
MQTT über mTLS – über die eigene Middleware	✓	✓	✓

Zwei Zertifikatswelten (PKI)



Verteilung: Die Zertifikate beider Welten verteilt die eigene Infrastruktur des Anwenders (Phrase-Lock-Verwaltung) – kein Zertifikat läuft über Dritte. Das Smartphone liest dafür nur einen QR-Code mit zeitlich begrenzter Gültigkeit ein. Er enthält kein Zertifikat, sondern eine Referenz, über die es sein Zertifikat direkt von der eigenen Infrastruktur abholt.

Sicherheitsmodell auf einen Blick

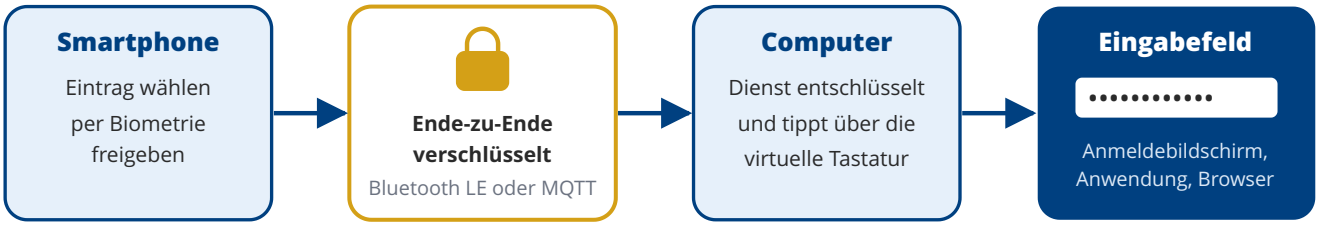
- **Geheimnisse bleiben auf dem Smartphone.** Jede Verwendung wird dort per Biometrie freigegeben.
- **Ende-zu-Ende-Verschlüsselung** zwischen Smartphone und Computer.
- **Gegenseitige Authentisierung** an der Middleware mit Client-Zertifikaten (mTLS).
- **Autonom** über Bluetooth LE – ohne Netz, Server oder Cloud.
- **Eigene Infrastruktur:** Middleware quelloffen, beim Anwender.
- **Standards statt Eigenbau:** Für das Betriebssystem ist Phrase-Lock eine Tastatur, eine Smartcard oder ein FIDO2-Schlüssel – Anwendungen müssen nicht angepasst werden.

> **Quelloffen:** Middleware und Installer für die eigene Infrastruktur liegen öffentlich auf **github.com/phraselock**.

Teil 2 – Im Detail

Komplexe Passwörter sicher eingeben

Starke Passwörter scheitern in der Praxis am Abtippen: Sie werden kurz gewählt, notiert oder wiederverwendet. Phrase-Lock nimmt dem Menschen das Eintippen ab.

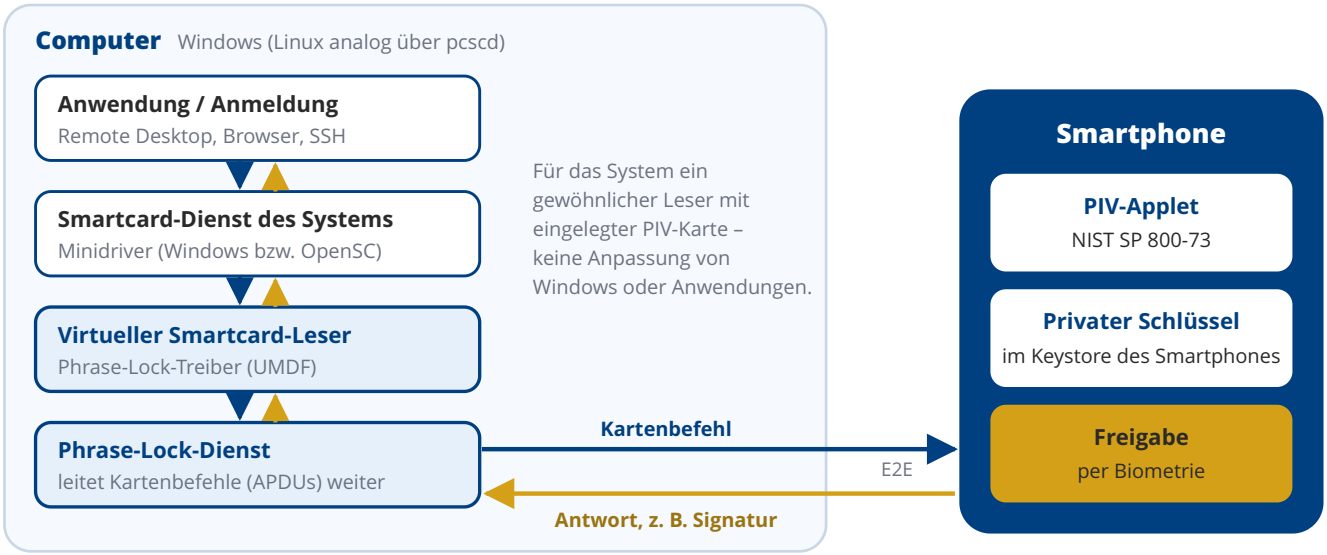


- **Jede Länge, jede Komplexität** – das Passwort muss niemand kennen, merken oder abtippen.
- **Überall, wo eine Tastatur funktioniert** – ohne Anpassung der Anwendung, auch vor der Anmeldung.
- **Kein Zettel, keine Zwischenablage** – das Passwort verlässt den Tresor nur verschlüsselt.

- Das Passwort liegt im **Tresor der App** und wird am Smartphone ausgewählt und per **Biometrie** freigegeben.
- Es geht **Ende-zu-Ende-verschlüsselt** an den Computer; der Dienst gibt es über die **virtuelle Tastatur** ein.
- Für das Betriebssystem ist das eine gewöhnliche Tastatur: Es funktioniert in **jedem Eingabefeld** – Anmeldebildschirm, Anwendungen, Browser, Konsole – ohne Anpassung.
- Da der Dienst schon **vor der Anmeldung** läuft, funktioniert das auch am Windows-Anmeldebildschirm.
- Benutzername, Passwort und Einmalpasswort (TOTP) lassen sich nacheinander eingeben.
- Optional dient eine bestehende **KeePass-Datenbank** auf dem eigenen Server als gemeinsame Quelle; jedes Smartphone fordert die ihm zugeordneten Einträge an.

Smartcard (PIV)

Das Smartphone wird zur Smartcard nach **PIV** (NIST SP 800-73). Der Computer erhält dafür einen **virtuellen Kartenleser**, in dem die Karte des Smartphones steckt.

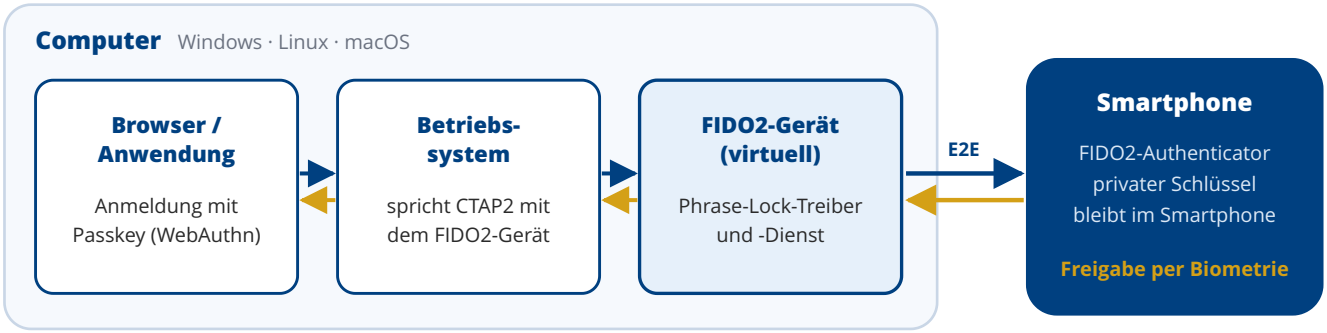


- Für Windows und Linux ist es ein **gewöhnlicher Leser mit PIV-Karte** – Anmeldung, Remote Desktop, Signatur, SSH und Browser nutzen ihn ohne Anpassung.

- Der **private Schlüssel** entsteht in der eigenen Verwaltung, weil iOS für RSA keine Zertifikatsanfrage erzeugen kann. Er wird einmalig mTLS- und Ende-zu-Ende-verschlüsselt übertragen und im Keystore des Smartphones abgelegt; jede Verwendung wird dort per Biometrie freigegeben.
- **Domänenanmeldung** an Active Directory per Kerberos PKINIT mit Zertifikaten der Unternehmens-CA.
- **Schlüssel:** RSA 2048 mit dem eingebauten Windows-Minidriver; RSA 3072 und ECC P-256 mit dem OpenSC-Minidriver.
- **Remote Desktop:** Die Karte am Client meldet den Benutzer am entfernten Rechner an – auch von außerhalb des Firmennetzes über einen Tunnel ohne offenen Port.

FIDO2 / Passkeys

Das Smartphone wird zum **FIDO2-Sicherheitsschlüssel** (CTAP2). Der Computer sieht ein FIDO2-Gerät, wie bei einem Hardware-Schlüssel – mit Biometrie und dem Tresor des Smartphones.



- **Browser und Web-Anwendungen:** Passkeys und Sicherheitsschlüssel nach WebAuthn – phishing-resistent.
- **Windows-Anmeldung mit Entra ID:** Das Smartphone dient als FIDO2-Sicherheitsschlüssel für die Anmeldung an Windows.
- **Ohne Anpassung:** Für das System ist es ein gewöhnlicher FIDO2-Sicherheitsschlüssel über USB-HID; Browser, Web-Anwendungen und Werkzeuge wie libfido2 nutzen ihn wie einen Hardware-Schlüssel.
- **Private Schlüssel** bleiben auf dem Smartphone; jede Verwendung wird dort per Biometrie freigegeben.

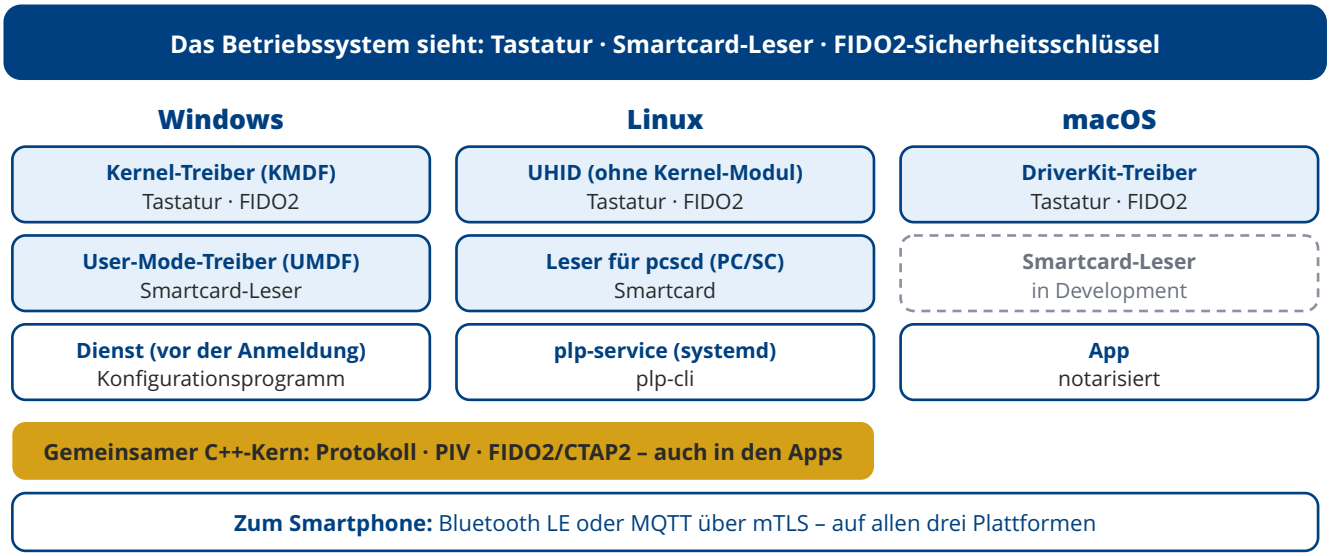
Smartphone-Apps

- Native Apps für **iOS** und **Android**.
- **Tresor** für Passwörter, Einmalpasswörter (TOTP) und Schlüssel; Freigabe jeder Verwendung per **Biometrie**.
- **PIV-Smartcard** und **FIDO2-Authenticator** (CTAP2) laufen in der App; private Schlüssel verlassen das Smartphone nicht.
- **Kopplung** mit Computer und Server per QR-Code.
- Verbindung zum Computer per **Bluetooth LE** oder **MQTT**.

Weitere Funktionen

- **Autofill-Unterstützung**
- **Benutzerdefinierte Verschlüsselung**
- **Biometrischer Schutz**
- **Peer-to-Peer-Datenaustausch**
- **Unbegrenzt viele Smartcards** auf dem Smartphone
- **Unbegrenzt viele FIDO2-Credentials**
- **Backup und Restore** – auch der FIDO2-Credentials

Aufbau je Plattform



Gemeinsamer Kern und Protokoll

- Ein portabler **C++-Kern** wird auf Windows, Linux und den Smartphones verwendet: Phrase-Lock-Protokoll, PIV-Applet, FIDO2/CTAP2 inkl. CBOR.
- Das **PIV-Applet** ist reines C++17 ohne Plattformabhängigkeit: APDU hinein, Antwort heraus. Kryptografie, Benutzerinteraktion und Speicher stellt jede Plattform selbst bereit.
- **Protokoll und Verschlüsselung sind unabhängig vom Transport:** Bluetooth LE und MQTT tragen dieselben Nachrichten.
- **Gerätebindung:** Die Lizenz ist an das jeweilige Gerät gebunden.

Windows

Treiber

- **Kernel-Treiber (KMDF):** virtuelle Tastatur und virtuelles FIDO2-Gerät (HID).
- **User-Mode-Treiber (UMDF):** virtueller Smartcard-Leser. Windows sieht einen gewöhnlichen Leser mit PIV-Karte; die Kartenbefehle gehen über den Dienst an das Smartphone. Ein Fehler im User-Mode-Treiber beendet nur den Treiberprozess, nicht das System.
- Beide Treiber sind **von Microsoft signiert** (Attestation), für **x64 und ARM64**.

Dienst und Konfiguration

- Der **Dienst** läuft schon vor der Anmeldung (Session 0) und hält die Verbindung zum Smartphone: Tastatur, Smartcard, FIDO2, Bluetooth LE und MQTT.
- Ein **Konfigurationsprogramm** für Kopplung und Einstellungen.
- **Installer** (Setup bzw. MSI) für Treiber, Dienst und Konfiguration.

Linux

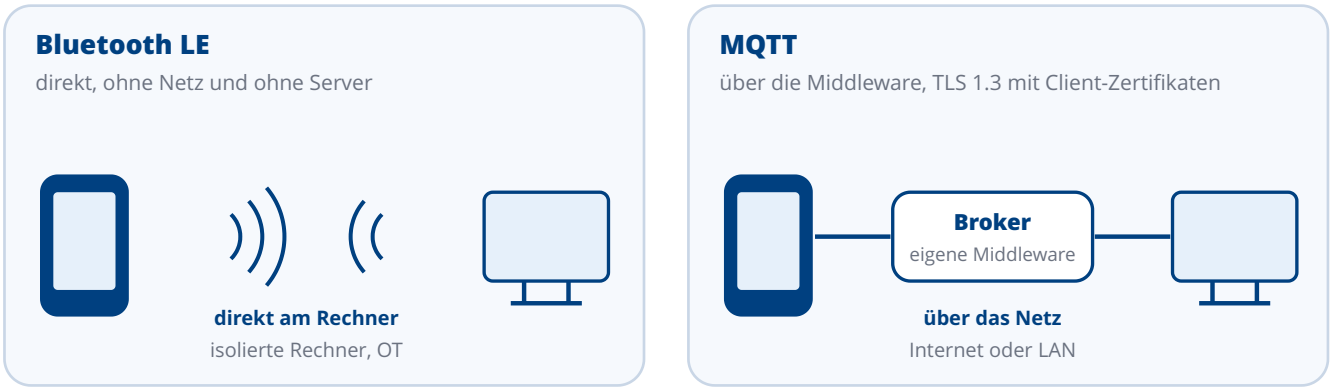
- **Kein eigenes Kernel-Modul:** Tastatur und FIDO2-Gerät entstehen im Userspace über **UHID**. Für Browser, libfido2 und PAM sind sie nicht von echter Hardware zu unterscheiden.
- **Ein Prozess:** `plp-service` (systemd) vereint Geräte und Dienstlogik; er startet vor dem Anmeldebildschirm.
- **Smartcard:** eigener Leser für **pcscd** (PC/SC). Darüber funktionieren OpenSC, SSH, Browser und die **Domänenanmeldung** am Anmeldebildschirm (GDM, Kerberos PKINIT).
- **Bedienung:** Kommandozeilenwerkzeug `plp-cli` (Kopplung per QR-Code, Status, Einstellungen) statt einer grafischen Oberfläche.
- **Kryptografie:** OpenSSL 3.

- **Pakete:** `.deb` und `.tar.gz`.

macOS

- **DriverKit-Treiber:** virtuelle Tastatur und virtuelles FIDO2-Gerät (HID), ohne Kernel-Erweiterung.
- **App** für Kopplung, Einstellungen und Verbindung zum Smartphone.
- **Notarisiert** von Apple.
- **Smartcard:** in Development (virtueller USB-Kartenleser); Voraussetzung ist eine Berechtigung von Apple.

Verbindung zwischen Smartphone und Computer

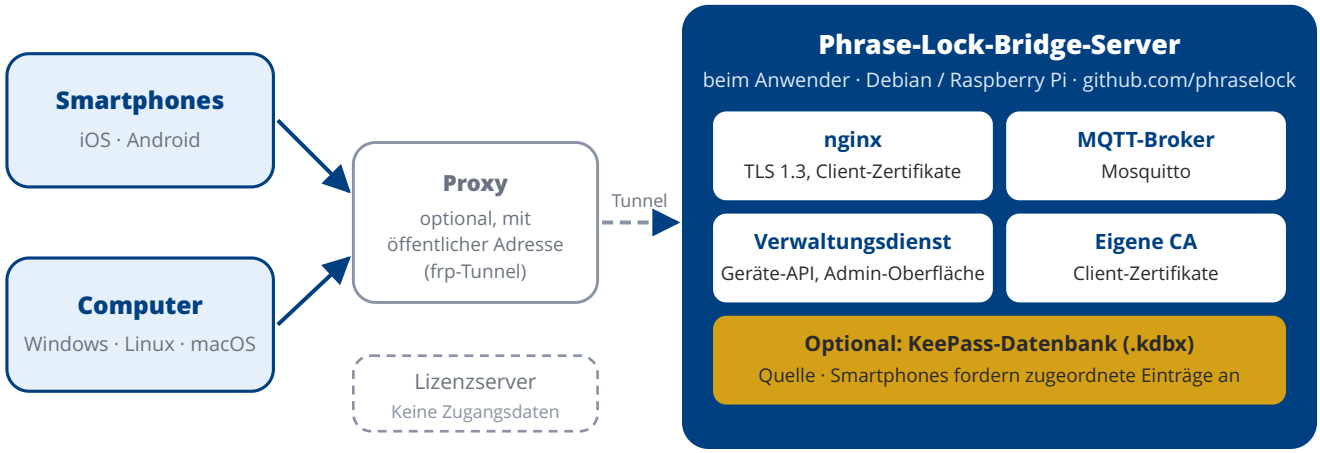


	Bluetooth LE	MQTT
Netz nötig	nein	ja (Internet oder LAN)
Server nötig	nein	MQTT-Broker (Middleware)
Absicherung	Ende-zu-Ende-Verschlüsselung der Nachrichten	TLS 1.3 mit Client-Zertifikaten + Ende-zu-Ende-Verschlüsselung
Typischer Einsatz	isolierte Rechner, kein Netz	Rechner im Netz (Internet oder LAN)

Bluetooth LE im Detail: Der Computer tritt als Bluetooth-Peripheriegerät (GATT-Server) auf, das Smartphone verbindet sich mit ihm. Dafür genügt der eingebaute Bluetooth-Adapter oder ein handelsüblicher USB-Bluetooth-Dongle. Eine Kopplung auf Ebene des Betriebssystems ist nicht nötig – die Sicherheit kommt aus der Ende-zu-Ende-Verschlüsselung von Phrase-Lock.

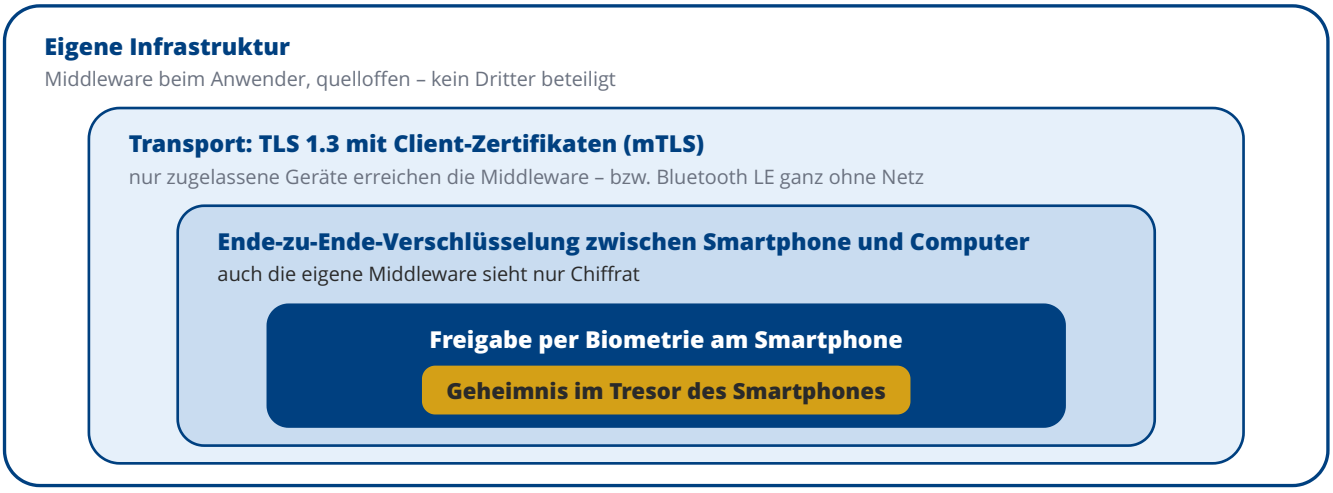
MQTT im Detail: MQTT ist ein schlankes Publish/Subscribe-Protokoll mit minimalem Overhead – sparsam bei Akku, Bandbreite und Rechenleistung und für instabile Mobilfunkverbindungen ausgelegt. Smartphone und Computer bauen die Verbindung nur **ausgehend** zum Broker auf: Am Computer muss kein Port geöffnet werden, es funktioniert hinter NAT und Firewall. Getrennte Verbindungen werden automatisch wieder aufgebaut. Es kann **jeder beliebige MQTT-Broker** verwendet werden; standardmäßig kommt **Mosquitto** zum Einsatz – quelloffen, extrem schlank und performant, er läuft sogar auf einem Raspberry Pi. MQTT ist als ISO-Standard (ISO/IEC 20922) in der Industrieautomatisierung etabliert.

Middleware und Betrieb



- **Phrase-Lock-Bridge:** Installer für die eigene Infrastruktur (Debian, Raspberry Pi): MQTT-Broker (Mosquitto), nginx, eigene Zertifizierungsstelle für Client-Zertifikate, Verwaltungsdienst. Quelloffen auf GitHub.
- **Verwaltungsdienst:** API für die Geräte (nur mit Client-Zertifikat) und Admin-Oberfläche.
- **KeePass als Quelle:** Eine bestehende KeePass-Datenbank kann als gemeinsame Quelle der Zugangsdaten dienen. Jedes Smartphone fordert die ihm zugeordneten Einträge an – kein automatischer Abgleich, keine Kopie der ganzen Datenbank auf dem Smartphone.
- **Ohne feste IP-Adresse:** optionaler Proxy (Tunnel) für Standorte ohne öffentliche Adresse.
- **Lizenzserver:** prüft nur die Lizenzen von Smartphones und Computern – er sieht keine Zugangsdaten.

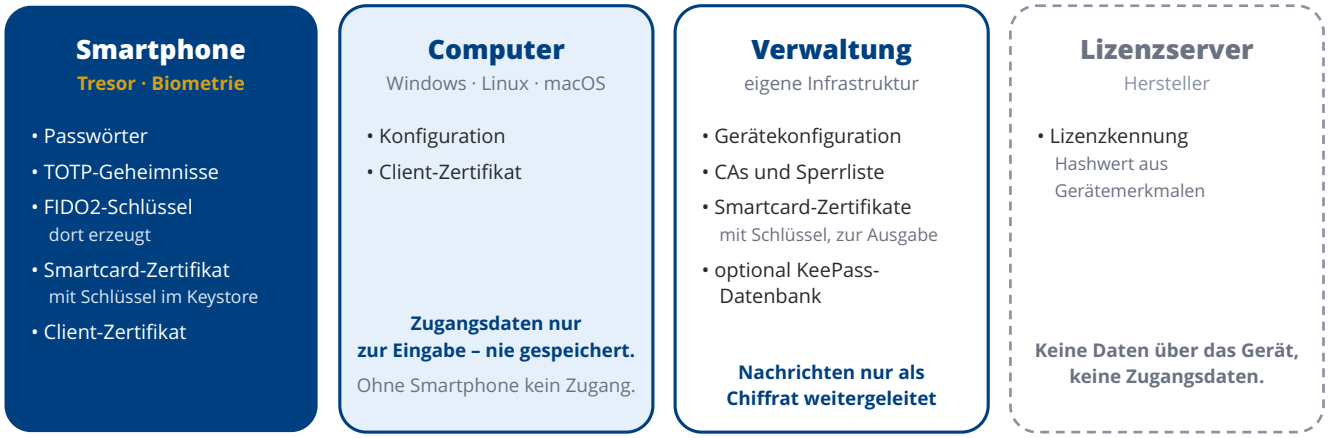
Sicherheit in Schichten



Standards und Schnittstellen

Bereich	Standard
Passworteingabe	USB-HID-Tastatur (virtuell)
Smartcard	PIV nach NIST SP 800-73; RSA 2048/3072, ECC P-256
Kartenschnittstelle	Windows Smart Card (SCardSvr, Minidriver), PC/SC unter Linux
Domäne	Active Directory, Kerberos PKINIT
FIDO2 / WebAuthn	CTAP2-Authenticator über USB-HID – Browser, Web-Anwendungen, Windows-Anmeldung (Entra ID), libfido2
Transport	MQTT über TLS mit Client-Zertifikaten; Bluetooth Low Energy (GATT)
Kryptografie	RSA und ECC (PIV), ECDSA P-256 (FIDO2), TLS 1.3

Was liegt wo



In zwei Schritten testen



Windows, Entra ID, macOS, iOS, Android, Bluetooth und alle weiteren genannten Marken sind Eigentum ihrer jeweiligen Inhaber.