Openloop

Openloop セキュリティキー ガイド
PIV/PKCS#11 & FIDO2/CTAP2 パスキー

バージョン v0.91.0 | 最終更新 2026年4月2日
株式会社ハウディ・クリプト　代表取締役社長　浅田 一憲

目次

1. はじめに

Openloopは暗号資産ハードウェアウォレットとしてだけでなく、**セキュリティキー**としても機能します。1台のデバイスで以下の2つのセキュリティ機能を提供します。

機能	プロトコル	主な用途
パスキー	FIDO2/CTAP2	Webサイトのパスワードレス認証、二要素認証
PIV/PKCS#11	PIV (NIST SP 800-73)	SSH認証、TLSクライアント証明書、コード署名

この2つの機能は用途が重複せず、相互補完の関係にあります。

ユースケース	パスキー (CTAP2)	PIV/PKCS#11
ブラウザWebAuthn認証	✓	—
SSH公開鍵認証	—	✓
TLSクライアント認証	—	✓
コード署名 / git署名	—	✓

前提条件

- **パスキー**: ソフトウェアのインストールは不要。USB接続のみで動作します。
- **PIV/PKCS#11**: **Openloop Connect**（デスクトップアプリ）の起動が必要です。PKCS#11ライブラリはConnectに同梱されています。

---

2. パスキー（FIDO2/CTAP2）

2.1 概要

OpenloopはFIDO2/CTAP2準拠のセキュリティキーとして動作します。パスワードに代わる安全な認証手段として、対応するWebサイトやサービスで利用できます。

項目	仕様
プロトコル	CTAP2 (FIDO_2_0)
後方互換	U2F (FIDO U2F V2)
通信	USB HID (FIDO Alliance Usage Page 0xF1D0)
署名アルゴリズム	ES256 (P-256) / EdDSA (Ed25519)
Discoverable Credential	✓ サポート（Resident Key）
User Verification	✓ デバイスPIN + 物理タッチ
最大クレデンシャル数	50

2.2 Openloopでの有効化

パスキー機能はデフォルトで無効になっています。使用前に以下の手順で有効化してください。

1. Openloopデバイスの **設定** > **USB設定** を開く
2. **USB HID** を **ON** に設定
3. **設定** > **パスキー** を開く
4. **パスキー** を **ON** に設定

> ⚠ USB HIDとパスキーの両方を有効にする必要があります。

2.3 Resident Key と Non-Resident Key

パスキーのクレデンシャルには2種類あります。

種類	説明	特徴
Resident Key（Discoverable Credential）	クレデンシャル情報をデバイス内に保存。ユーザー名の入力なしで認証可能。	デバイスの保存領域を使用。Openloopでは**最大50個**まで保存可能。
Non-Resident Key	クレデンシャルIDにデバイス固有の暗号で包んだ情報を埋め込む。デバイス内に状態を保持しない。	保存数の制限なし。ただし認証時にサーバーからcredentialIdの提示が必要。

Webサイトがどちらを要求するかは、登録時の `residentKey` パラメータで決まります。近年のパスキー対応サービス（Google、Microsoft、GitHub等）はResident Keyを要求するのが一般的です。

> ℹ デバイスに保存されたResident Keyの一覧は、**設定** > **パスキー** > **クレデンシャル一覧** で確認・削除できます。Non-Resident Keyはデバイスに状態を持たないため一覧には表示されません。

2.4 パスキーの登録（Registration）

対応するWebサイトやサービスでパスキーを登録します。

1. Webサイトのセキュリティ設定で「セキュリティキーを追加」等を選択
2. ブラウザがセキュリティキーの挿入を求めるダイアログを表示
3. OpenloopをUSBで接続（または既に接続済み）
4. Openloopのデバイス画面に登録リクエストが表示される
5. 内容を確認し、**承認**をタッチ
6. デバイスPINの入力を求められる場合があります（User Verification設定による）

2.5 パスキーによる認証（Authentication）

登録済みのWebサイトにログインする際に使用します。

1. Webサイトのログイン画面で「セキュリティキーでログイン」等を選択
2. OpenloopをUSBで接続
3. Openloopのデバイス画面に認証リクエストが表示される
4. **承認**をタッチ

2.6 クレデンシャルの管理

登録したパスキーはOpenloopデバイス上で管理できます。

- **一覧表示**: 設定 > パスキー > クレデンシャル一覧
- **個別削除**: クレデンシャルを選択して削除
- **全削除**: 設定 > パスキー > パスキーリセット

> ⚠ パスキーリセットを実行すると、すべてのクレデンシャルが削除されます。各Webサイトで再登録が必要になります。

2.7 対応ブラウザ・プラットフォーム

OS	Chrome	Edge	Safari	Firefox
Windows	✓	✓	-	✓
macOS	✓	-	△	△

- **Windows**: 全ブラウザでWebAuthn登録・認証が正常動作（OS標準webauthn.dll経由）
- **macOS Chrome**: 登録・認証は正常動作
- **macOS Safari/Firefox**: 登録・認証は正常動作。ユーザーがデバイス側で承認しなかった場合、ブラウザが応答待ちを続ける場合があります（プラットフォーム側の制限）。ブラウザの「キャンセル」ボタンで復帰可能。

2.8 対応サービスの例

パスキーに対応する主要なWebサイト・サービス:

- **Google** アカウント
- **Microsoft** アカウント
- **GitHub**
- **Cloudflare**

- **AWS** (IAM)
- **1Password**
- その他、FIDO2/WebAuthn対応の多数のサービス

> ℹ 対応サービスの最新リストは passkeys.directory で確認できます。

---

3. PIV/PKCS#11

3.1 概要

OpenloopはPIV（Personal Identity Verification, NIST SP 800-73）スマートカードインターフェースを実装しています。PKCS#11共有ライブラリを通じて、SSH認証やTLSクライアント認証など、パスキーではカバーできないユースケースに対応します。

3.2 アーキテクチャ

```
SSH / Firefox / pkcs11-tool
  ↓ PKCS#11 C API
libopenloop-pkcs11 (.dylib / .dll / .so)
  ↓ PIV APDU → WebSocket (ws://127.0.0.1:21320)
Openloop Connect (APDUの透過転送)
  ↓ USB HID
Openloop デバイス (PIVハンドラ → SE050で署名)
```

PKCS#11ライブラリはOpenloop Connect経由でデバイスと通信するため、PIV/PKCS#11機能を使用する際は**Openloop Connectが起動している**必要があります。

3.3 PIV/PKCS#11の有効化

設定 > パスキー・PIV > PIV ON でPIVを有効化します。OFFにするとPKCS#11のスロットが非表示になります。

また、以下の設定も確認してください:

1. **Openloop Connect**をインストール・起動する
2. Openloopデバイスの **設定** > **USB設定** > **USB HID** を **ON** に設定
3. OpenloopをUSBで接続し、ConnectがデバイスをDevice Connectedと表示することを確認

3.4 PIV PIN設定

PIV PINは**オプション**です。PINを設定するとUSB経由のPIN認証が有効になり、PIN送信時は確認画面なしで署名されます（ブラインド署名）。PINを設定しない場合は、従来通りデバイス画面での確認操作が必要です。

デュアルモード

OpenloopのPIV実装は、PIN送信の有無に応じて2つの動作モードを提供します。

モード	動作	用途
PIN送信あり	ブラインド署名（確認画面なし、自動署名）	自動化・CI/CD・スクリプト向け
PIN未送信	従来の確認画面フロー（デバイス画面で承認）	対話的な利用

> 1つのデバイスで両方のモードを使い分けることができます。PINを送信するかどうかはクライアント側（SSH設定等）で制御します。

PIN設定方法

```
# PINの初期設定
pkcs11-tool --module "$MODULE" --login --init-pin --new-pin 123456

# PINの変更
pkcs11-tool --module "$MODULE" --login --change-pin --pin 123456 --new-pin 654321
```

PINの削除

PINを削除するには、以下のいずれかの方法を使用します:

- **デバイスUI**: 設定 > パスキー・PIV > PIV PIN の🗑ボタンをタッチ
- **USB APDU**: RESET RETRY COUNTERコマンドを送信

PINロック時の対処

PINを**8回連続で間違える**とロックされます。ロック状態になった場合は、デバイスUIの🗑ボタンでPINを削除し、再設定してください。

SSH使用例（PIN付き）

```
# PIN付きでSSH接続
ssh -o "PKCS11Provider=$MODULE" -o "PKCS11Pin=123456" user@host
```

> ⚠ PINをコマンドラインに直接記述すると、シェル履歴やプロセス一覧に残る可能性があります。セキュリティが重要な環境では、ssh-agentの利用やPIN未送信モード（デバイス確認画面フロー）を検討してください。

---

4. PKCS#11ライブラリパス

PKCS#11ライブラリはOpenloop Connectに同梱されています。パスはOSによって異なります。

OS	パス
macOS	`/Applications/Openloop Connect.app/Contents/Resources/pkcs11/libopenloop-pkcs11.dylib`
Windows	`C:\Program Files\Openloop Connect\resources\pkcs11\libopenloop-pkcs11.dll`
Linux	`/opt/Openloop Connect/resources/pkcs11/libopenloop-pkcs11.so`

以降の例では、macOSのパスをシェル変数に設定して使用します:

```
MODULE="/Applications/Openloop Connect.app/Contents/Resources/pkcs11/libopenloop-pkcs11.dylib"
```

---

5. SSH公開鍵認証

OpenloopをPKCS#11経由でSSHハードウェアキーとして使用します。秘密鍵はデバイスのSE050セキュアエレメント内に保管され、外部に取り出すことはできません。

5.1 鍵の準備

PIVスロット（デフォルト: 9A Authentication）に鍵ペアを生成します。鍵の生成方法は以下のいずれかです:

- **デモページ**: PIV Demo のWebインターフェースから生成
- **pkcs11-tool**: コマンドラインから生成（第7章参照）

5.2 公開鍵の取得

```
# デバイスからSSH公開鍵を取得
ssh-keygen -D "$MODULE"

# ファイルに保存
ssh-keygen -D "$MODULE" > ~/openloop_key.pub
```

取得した公開鍵をリモートサーバーの `~/.ssh/authorized_keys` に追加してください。

5.3 SSH接続

```
# 単発の接続
ssh -I "$MODULE" user@hostname
```

5.4 永続的な設定（~/.ssh/config）

毎回 `-I` オプションを指定する代わりに、`~/.ssh/config` に設定を記述できます。

```
# 特定のホストに適用
Host myserver
    HostName 192.168.1.100
    User ubuntu
    PKCS11Provider /Applications/Openloop Connect.app/Contents/Resources/pkcs11/libopenloop-pkcs

# すべてのホストに適用
Host *
    PKCS11Provider /Applications/Openloop Connect.app/Contents/Resources/pkcs11/libopenloop-pkcs
```

5.5 SSH実装の互換性とOpenSSH推奨

OpenloopのPIV/PKCS#11はP-256、Ed25519、RSA-2048の3つのアルゴリズムをサポートしています（RSA-2048はSE050C1/C2バリアントのみ）。**SSH実装によってPKCS#11経由で利用できるアルゴリズムが異なります。**

SSH実装	RSA-2048	P-256 (ECDSA)	Ed25519 (EdDSA)	備考
OpenSSH (8.5以降)	✓	✓	✓	推奨。全アルゴリズムをフルサポート。
macOS　システムSSH (`/usr/bin/ssh`)	✓	△	✗	PKCS#11経由ではRSAのみ安定。

SSH実装	RSA-2048	P-256 (ECDSA)	Ed25519 (EdDSA)	備考
PuTTY (Windows)	✓	△	✗	PKCS#11対応が限定的。RSA中心。
Dropbear	✗	✗	✗	PKCS#11非対応。

> ▲ **OpenSSH（Homebrew版）の使用を強く推奨します。** macOSのシステムSSH（Apple版）はPKCS#11経由でのECDSA/EdDSAサポートが不完全です。RSA-2048鍵を使用すれば互換性の問題を回避できます（SE050C1/C2バリアントのみ）。

```
# Homebrew OpenSSHのインストール
brew install openssh

# Homebrew版を使用（フルパス指定）
/opt/homebrew/bin/ssh -I "$MODULE" user@hostname

# バージョン確認（8.5以上であることを確認）
/opt/homebrew/bin/ssh -V

# ~/.ssh/config で IgnoreUnknown UseKeychain を追加
#  （Homebrew SSHが認識しない macOS 独自オプションの警告を抑制）
```

5.6 デバッグ

```
# PKCS#11ライブラリのデバッグ出力を有効化
OPENLOOP_PKCS11_DEBUG=1 ssh-keygen -D "$MODULE"

# SSH接続の詳細ログ
ssh -vvv -I "$MODULE" user@hostname
```

---

6. Firefox TLSクライアント認証

なぜFirefoxか

主要ブラウザの中で、**Firefoxは唯一、外部PKCS#11モジュールの読み込みをサポート**しているブラウザです。ChromeやEdge、SafariはOS標準のキーチェーン/証明書ストアのみを使用し、サードパーティのPKCS#11ライブラリを直接読み込む機能を提供していません。そのため、OpenloopのPKCS#11ライブラリを使ったTLSクライアント認証はFirefoxで利用します。

PKCS#11モジュールをFirefoxに登録して、TLSクライアント証明書認証（mTLS）を利用します。

6.1 PKCS#11モジュールの登録

1. Firefoxの **設定** を開く
2. **プライバシーとセキュリティ** に移動
3. **証明書** セクションの **セキュリティデバイス…** をクリック
4. **読み込む** をクリック
5. **モジュール名**: `Openloop` と入力
6. **モジュールファイル名**: 上記のライブラリパスを参照して選択
7. **OK** をクリック

6.2 確認方法

1. セキュリティデバイスダイアログで **Openloop** モジュールを展開
2. トークン情報が表示されるスロットを確認
3. **証明書を表示**（設定 > 証明書 > 証明書を表示）をクリック
4. **あなたの証明書** タブにOpenloopの証明書が表示されます

6.3 mTLS認証フロー

クライアント証明書を要求するWebサイトにアクセスすると、Firefoxが自動的にOpenloopデバイスから証明書を選択するよう促します。署名操作にはUser　Presence（デバイスへの物理的なタッチ）が必要です。

6.4 自動登録（macOS）

macOSでは、Openloop　Connectが以下のパスにマニフェストを配置することで、PKCS#11モジュールをFirefoxに自動登録できます:

```
~/Library/Application Support/Mozilla/PKCS11Modules/openloop_pkcs11.json
```

自動登録が有効な場合、手動での登録は不要です。

---

7. pkcs11-tool リファレンス

OpenSCの `pkcs11-tool` コマンドを使って、デバイスの鍵やオブジェクトを操作できます。

インストール

```
# macOS
brew install opensc

# Ubuntu/Debian
sudo apt install opensc

# Windows
# OpenSCインストーラーからインストール: https://github.com/OpenSC/OpenSC/releases
```

スロット・トークン一覧

```
pkcs11-tool --module "$MODULE" -T
```

オブジェクト一覧

```
pkcs11-tool --module "$MODULE" -O
```

鍵ペア生成

```
# P-256 をスロット 9A（Authentication）に生成
pkcs11-tool --module "$MODULE" --keypairgen \
  --key-type EC:prime256v1 \
  --id 01 --label "PIV AUTH"

# Ed25519 をスロット 9A に生成
pkcs11-tool --module "$MODULE" --keypairgen \
  --key-type EC:edwards25519 \
  --id 01 --label "PIV AUTH"
```

署名テスト

```
# テストデータを作成し、スロット 9A の鍵で署名
echo "test data" | openssl dgst -sha256 -binary > /tmp/hash.bin
pkcs11-tool --module "$MODULE" --sign \
  --mechanism ECDSA \
  --id 01 \
  --input-file /tmp/hash.bin \
  --output-file /tmp/sig.bin
```

鍵の削除

```
pkcs11-tool --module "$MODULE" --delete-object \
  --type privkey --id 01
```

---

8. PIVスロット・アルゴリズム一覧

8.1 PIVスロット

スロット	名前	用途
9A	PIV Authentication	SSH認証、一般的な認証。最も使用頻度が高い。
9C	Digital Signature	文書署名、S/MIME。常にPINが必要。
9D	Key Management	暗号化/復号化、鍵共有。
9E	Card Authentication	物理アクセス、非接触認証。PIN不要。

8.2 対応アルゴリズム

アルゴリズム	PIV ID	バリアント	備考
P-256 (secp256r1)	0x11	全機種	NIST P-256によるECDSA。広くサポートされている。
Ed25519	0x22	全機種	Curve25519によるEdDSA。SSHにはOpenSSH 8.5以上が必要。
RSA-2048	0x07	C1/C2のみ	PKCS#1 v1.5署名。SSH/GPG/PDF署名に対応。鍵生成に約10秒。

8.3 PKCS#11オブジェクトID マッピング

PIVスロット	PKCS#11 ID	CKA_LABEL	用途
9A	01	PIV AUTH	Authentication
9C	02	SIGN	Digital Signature
9D	03	KEY MGMT	Key Management
9E	04	CARD AUTH	Card Authentication

---

9. トラブルシューティング

パスキーが認識されない

- デバイスの **設定** > **USB設定** > **USB HID** が ON になっているか確認
- デバイスの **設定** > **パスキー** が ON になっているか確認
- USBケーブルがデータ通信対応か確認（充電専用ケーブルでは動作しません）
- デバイスを再接続してみてください

PKCS#11ライブラリが見つからない

- Openloop Connectがインストールされているか確認
- ライブラリパスが正しいか確認（第4章参照）

ssh-keygen -D で鍵が表示されない

- Openloop Connectが起動しているか確認
- デバイスがConnectに接続されているか確認（Device Connected表示）
- PIVスロットに鍵が生成されているか確認（pkcs11-tool -O で確認）
- macOSではHomebrew版OpenSSHを使用しているか確認

Firefox でモジュールが読み込めない

- Openloop Connectが起動しているか確認
- ライブラリファイルのパスが正しいか確認
- Firefoxを再起動してみてください

署名時にタイムアウトする

- デバイス画面に署名確認ダイアログが表示されていないか確認
- User Presence（物理タッチ）が必要な操作では、デバイスへのタッチが必要です

デバッグログの取得

PKCS#11ライブラリのデバッグ出力を有効にすることで、通信の詳細を確認できます:

```
OPENLOOP_PKCS11_DEBUG=1 ssh-keygen -D "$MODULE"
```

---

10. 参考資料

FIDO2 / WebAuthn

資料	URL
CTAP2仕様 (FIDO Alliance)	https://fidoalliance.org/specs/fido-v2.1-ps-20210615/fido-client-to-authenticator-protocol-v2.1-ps-20210615.html
WebAuthn仕様 (W3C)	https://www.w3.org/TR/webauthn-2/
U2F仕様 (FIDO Alliance)	https://fidoalliance.org/specs/fido-u2f-v1.2-ps-20170411/
CTAPHID — USBトランスポート	https://fidoalliance.org/specs/fido-v2.1-ps-20210615/fido-client-to-authenticator-protocol-v2.1-ps-20210615.html#usb
パスキー対応サービス一覧	https://passkeys.directory

PIV / PKCS#11

資料	URL
NIST SP 800-73 (PIV仕様)	https://csrc.nist.gov/pubs/sp/800/73/4/final
PKCS#11仕様 (OASIS)	https://docs.oasis-open.org/pkcs11/pkcs11-base/v3.0/pkcs11-base-v3.0.html
OpenSC (pkcs11-tool)	https://github.com/OpenSC/OpenSC
OpenSSH PKCS#11ドキュメント	https://man.openbsd.org/ssh-keygen#D

Openloop関連

資料	URL
Openloop PIV/PKCS#11 デモ	https://crypto.haudi.jp/openloop/demo/piv/

資料	URL
Openloop パスキー デモ	https://crypto.haudi.jp/openloop/demo/ctap2/
Openloop Connect	https://crypto.haudi.jp/openloop/

Copyright © 2026 Haudi Crypto, Inc. All rights reserved.