AUTH

ユーザーマニュアル

AH52126_UM01_03

2024/04/03

Axell

商標

- Adobe、Acrobat および Reader は、Adobe 社の米国ならびに他の国における商標または登録商標です。
- Arm®、CMSIS は、Arm Limited 登録商標または商標です。
- CentOS、Red Hat® Enterprise Linux®、Fedora®は Red Hat, Inc.またはその子会社の米国およびその他の国における商標または登録商標です。
- Facebook®は Meta Platforms, Inc.の登録商標です。
- Firefox は Mozilla Foundation の商標です。
- FIDO®は FIDO Alliance, Inc.の商標または登録商標です。
- GitHub® は GitHub, Inc.の登録商標です。
- Google Chrome™、Gmail™は Google LLC の登録商標です。
- Linux®は米国およびその他の国で登録された Linux Torvalds 氏の商標です。
- Mac、macOS、Safari、Xcode は、Apple Inc.の商標または商標です。
- Microsoft Edge、Visual Studio、Windows は Microsoft Corporation の米国およびその他の国における登録商標または商標です。
- Opera は Opera Software ASA の商標または登録商標です。
- RSA®は RSA Security LLC の登録商標です。
- SHALO は(株)アクセルの商標または登録商標です。
- Ubuntu は Canonical Ltd.の商標または登録商標です。

本製品で利用されているソフトウェアライセンスについて

第三者ソフトウェアについて

auth.shalo.jp からダウンロードできるソフトウェア（SHALO AUTH 専用ソフトウェア）とファームウェア更新データ、そして SHALO AUTH ドングルのファームウェアにはオープンソースソフトウェアが使用されています。これらのライセンスについて詳しくはライセンス条件をご参照ください。

ライセンス条件は次の方法で表示できます。

各 GUI ツールを起動し、[**ヘルプ**]＞[**バージョン情報**]＞[**ライセンス情報**]の順にクリックします。

お問い合わせ先

本製品に関するサポート・お問い合わせは電子メールアドレス shalo@axell.co.jp にお寄せください。

改訂履歴

文書番号	年月日	ページ	改訂内容
AH52126_UM01_1.00	2021/06/25		初版
AH52126_UM01_1.01	2022/05/10	p.34, 38, 42 p.39 p.94 p.95-96 p.163	ソフトウェアのアンインストール方法を追加（3 章） Linux への必要なライブラリのインストール方法を追加（3.5.2 節） Windows 向け Acrobat® 64-bit の説明を追加（7.2.1 節） Acrobat®からの PKCS #11 モジュールの削除方法を追加（7.2.2 節） 症状別トラブルシューティングに追加（11.5.5 節）
AH52126_UM01_02	2022/11/02	p.1	商標およびライセンスを更新 第三者ソフトウェアを追記
AH52126_UM01_03	2024/04/03	p.170-173	コーポレートロゴの変更 PKCS#11 モジュール ver.1.4 の仕様を反映（12 章）

本書の表記

書体・書式について

本書では通常の書体のほかに、特別な目的のために次の書体を使用します。

太字	文中では重要な情報、ターミナルではユーザーの入力文字を表します。
斜体	利用者の環境に合わせて変更される情報を表します。
mono	コマンド名やコマンド入力オプションを表します。

以下はファイル名とその内容を示します。各行の左に行番号が表示されます。

file.txt

```
1  This is a message.
```

以下は 1 つのコマンド行を示します。

```
command -p 環境固有文字列
```

以下はターミナル・コマンドプロンプトにおける入出力を示します。

```
$ command↵
Message
```

この中で使われる特殊文字の意味は次の通りです。

>	PowerShell のプロンプトを表します。
$	Bash のプロンプトを表します。Cygwin、Git for Windows、Linux、macOS 10.14 Mojave 以前で使われます。
%	Zsh のプロンプトを表します。macOS 10.15 Catalina 以降で使われます。
↵	エンターキーの入力を表します。

本書で使う記号

参考・補足事項を記載します。

重要な注意事項を記載します

本書の構成と読み進め方

本書は 12 章から構成されています。各章の概要は以下の通りです。

第 1 章　SHALO AUTH について簡単に紹介します。動作環境や概略仕様もこの章に含まれます。
第 2 章　SHALO AUTH を利用する上で知っておく必要のある情報を説明します。
第 3 章　Windows/macOS/Linux のオペレーティングシステム別での SHALO AUTH の導入方法と、専用ソフトウェアのインストール方法を説明します。
第 4 章　SHALO AUTH 専用ソフトウェアの 1 つ、鍵ツール SHALO Keyring の使い方を説明します。
第 5 章　SHALO AUTH 専用ソフトウェアの 1 つ、管理ツール SHALO Smith の使い方を説明します。
第 6 章　Google/Facebook/GitHub のウェブサービスで 2 段階認証に SHALO AUTH を使用する方法を説明します。
第 7 章　PDF ファイルのセキュリティで SHALO AUTH を使用する方法を説明します。
第 8 章　SSH 認証で SHALO AUTH を使用してユーザー認証する方法を説明します。
第 9 章　GitHub の SSH 認証で SHALO AUTH を使用する方法を説明します。
第 10 章　リモート PC から手元の PC に接続した SHALO AUTH を使う方法など、SHALO AUTH を便利に活用する方法を説明します。
第 11 章　SHALO AUTH を使っていて寄せられる疑問や問題への対処方法を回答します。
第 12 章　開発者向けに SHALO AUTH の PKCS #11 モジュールについての諸仕様を記載します。

本書は SHALO AUTH の利用目的によっては読まずに済む章を多く含みます。効率的に SHALO AUTH の使用方法を習得するために、目的別の読み進め方を以下に示します。

- **SHALO AUTH を FIDO U2F に使用する方**

  1 章から 2.2 節まで読み、Linux で使用する場合だけさらに 3.5.1 節を読みます。その後、ウェブサービスで 2 段階認証を設定する方法について必要に応じて 6 章を読みます。

- **SHALO AUTH を PDF で使用する方**

  2.2 節を除いて 1 章から 4 章まで読みます。その後、7 章を読みます。

- **SHALO AUTH を SSH 認証で使用する方**

  2.2 節を除いて 1 章から 4 章まで読みます。その後、8 章を読みます。

- **SHALO AUTH を Git の SSH 認証で使用する方**

  2.2 節を除いて 1 章から 4 章まで読みます。その後、8 章と 9 章を読みます。

目次

第 1 章
SHALO AUTH の紹介

この章では、SHALO AUTH について簡単に紹介します。

この章のトピック

1. SHALO AUTH とは？
2. 利用シーン
3. 動作環境
4. 概略仕様
5. 使用上の注意

1.1 SHALO AUTH とは？

SHALO AUTH（図 1）は USB 接続可能なセキュリティキーです。Windows・macOS・Linux に対応し、OS 標準搭載のデバイスドライバで利用できます。

図 1　SHALO AUTH 外観

SHALO AUTH は大きく分けて以下の 2 つの機能を持ちます。

- FIDO U2F セキュリティキー
- 汎用セキュリティキー

FIDO U2F セキュリティキー

SHALO AUTH は FIDO によりレベル 2 の U2F 認証器として認定されています。Google Chrome や Safari、Microsoft Edge、Firefox などの主要なウェブブラウザで二要素認証のセキュリティキーとして使用できます。

汎用セキュリティキー

SHALO AUTH は汎用セキュリティキーとして公開鍵暗号の RSA と ECDSA に対応し、安全な鍵の管理や証明書の管理、暗号化・復号、デジタル署名の発行・検証が可能です。

RSA	ECDSA
鍵長 1,024～4096 ビット	曲線 P-192/P-224/P-256/P-384/P-521/ secp192k1/secp224k1/secp256k1

リスト 1　サポートする公開鍵暗号方式

汎用セキュリティキーの機能は暗号トークン・インターフェースの業界標準規格 PKCS #11 API で公開されます。開発者は PKCS #11 API を使って SHALO AUTH で独自のハードウェア認証ソリューションを構築できます。

例として Adobe® Acrobat®／Adobe® Acrobat® Reader®による PDF ファイルのセキュリティが挙げられます。これらのソフトウェアは PKCS #11 API をサポートし、次のように PDF ファイルを運用できます。

- PDF ファイルを暗号化して SHALO AUTH を使ったときだけ閲覧できるようにする
- PDF ファイルに SHALO AUTH で電子署名を付与する

そのほか、多くの開発者にとって身近な SSH や Git のユーザー認証に SHALO AUTH を利用できます。SSH はリモート PC やクラウド上の仮想マシンとのセキュア通信に使われています。SSH のユーザー認証に SHALO AUTH を使用することで、ローカル PC に鍵を保存せずに安全に通信できるようになります。SSH はバージョン管理システムの Git などでセキュア通信のインフラとして使用されているため、それらでも SHALO AUTH を活用できます。

1.2 利用シーン

SHALO AUTH は主に次の用途に使用できます。

- Google/Facebook 等のウェブサービスの二要素認証
- 暗号化された PDF ファイルの閲覧
- GitHub 等の Git プラットフォームにおける二要素認証と SSH 認証
- PKCS #11 対応ソフトウェアを使用したユーザー認証・デジタル署名

ウェブサービスの二要素認証

二要素認証に SHALO AUTH を使うと、ウェブサービスの ID とパスワードを入力した後に SHALO AUTH のボタンを押すことで認証します。

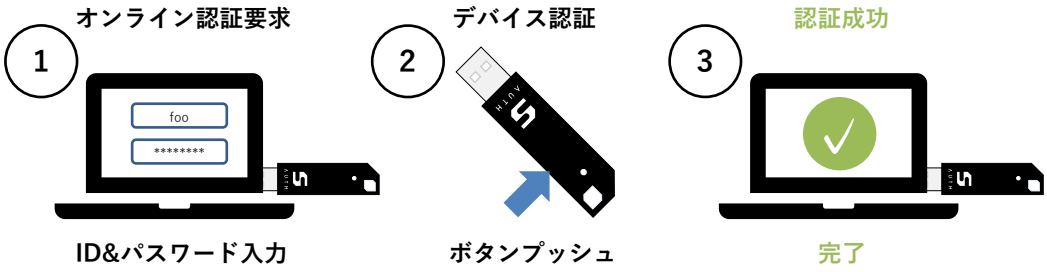


図 2　二要素認証の流れ

暗号化された PDF ファイルの閲覧

Adobe® Acrobat®のセキュリティ機能を使用して、特定の SHALO AUTH がない環境では閲覧できない PDF ファイルを作成できます。この PDF ファイルは SHALO AUTH 向けに暗号化され、Adobe® Acrobat®や Adobe® Acrobat® Reader®で PDF ファイルを閲覧する時に SHALO AUTH で暗号が解除されます。

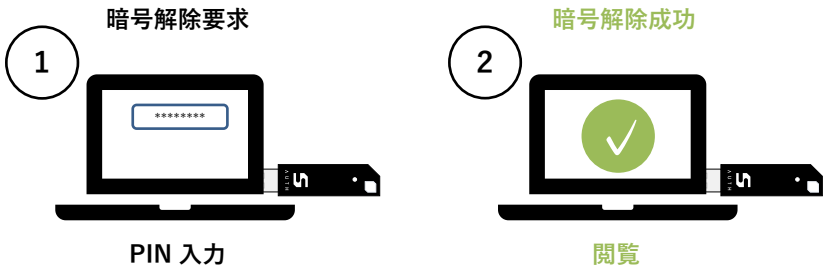


図 3　暗号化された PDF ファイルの閲覧の流れ

SSH の認証

SSH の認証に SHALO AUTH を使うと、リモート PC のパスワードの代わりに SHALO AUTH のユーザーPIN を入力することで認証します。

このユーザーPIN は SHALO AUTH に認証用のデジタル署名を生成させるためのものです。SHALO AUTH を持っていても正しいユーザーPIN を入力しないとデジタル署名は作られません。

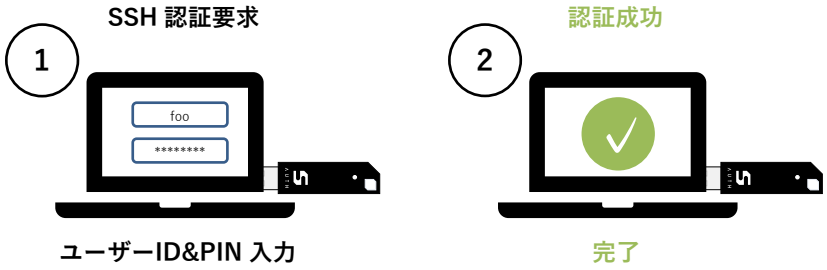


図 4　SSH の PKCS #11 認証の流れ

PKCS #11 対応ソフトウェアを使ったユーザー認証・デジタル署名

PKCS #11 対応ソフトウェアを使用して SHALO AUTH でユーザー認証やデジタル署名する場合、ソフトウェアに SHALO AUTH のユーザーPIN を入力して認証・署名します。

SHALO AUTH を持っていても正しいユーザーPIN を入力しないと処理されません。

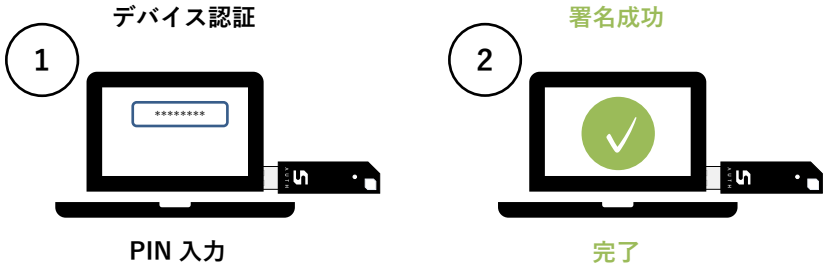


図 5　ユーザー認証・デジタル署名の流れ

1.3 動作環境

SHALO AUTH および SHALO AUTH 専用ソフトウェアは以下のオペレーティングシステムが動作する USB ポートを持つ PC で動作を確認しています。

オペレーティングシステム	バージョン
Windows	Windows 10 x86 ベースプロセッサ向け Windows 10 x64 ベースプロセッサ向け
macOS	macOS HighSierra（10.13）以降 インテルプロセッサおよび Apple Silicon プロセッサ向け
Linux	Red Hat Enterprise Linux 7 以降 CentOS 7 以降 Ubuntu 18.04 LTS 以降 Fedora 33 以降 ※いずれも x64 ベースプロセッサ向けのみ

SHALO AUTH は以下のウェブブラウザで U2F セキュリティキーとして利用できます。

ウェブブラウザ	バージョン
Google Chrome	バージョン 41 以降
Firefox	バージョン 67 以降
Microsoft Edge	バージョン 79 以降（Chromium 版のみ）
Safari	バージョン 13 以降

1.4 概略仕様

ハードウェア仕様

項目	説明
インタフェース	USB 2.0
適合コネクタ	USB Type-A
電源	USB バスパワー +5V±5%
外形寸法	68.6×16×8mm (キャップ含む)
動作保証環境	温度: -20～70°C 湿度: 20～80%(結露なきこと)
重量	7g
認証	VCCI(class-B)、FIDO U2F L2

FIDO U2F 機能

機能	説明
準拠規格	U2F v1.2
認証アルゴリズム	ECDSA P-256 with SHA-256
FIDO 認証鍵の生成数上限	1,000,000 個
ユーザー存在の確認方法	デバイスのボタン押下

PKCS #11 機能

機能	説明
準拠規格	PKCS #11 v2.40
SO PIN（管理 PIN）	長さ 4～256 バイトの UTF-8 文字列 連続 5 回の認証失敗で PIN ロックする保護機能付き
ユーザーPIN	長さ 4～256 バイトの UTF-8 文字列 連続 5 回の認証失敗で PIN ロックする保護機能付き
暗号処理	RSA: 暗号化・復号・署名・検証、ECDSA: 署名・検証、乱数生成、メッセージダイジェスト生成
データ管理	1 個あたり約 8KB のデータを最大 12 個格納可能 RSA プライベート鍵・RSA 公開鍵・ECDSA プライベート鍵・ECDSA 公開鍵・X.509 証明書のデータに対応 プライベート鍵の読み出し禁止に対応
メッセージダイジェスト	SHA-1, SHA-256, SHA-384, SHA-512
乱数生成器	NIST SP 800-90A 準拠 CTR-DRBG（AES-256 ベース）
RSA	PKCS #1 による RSA 暗号 1,024 ビット～4,096 ビットの鍵長に対応
ECDSA	FIPS 186-4 による以下の楕円曲線署名 secp192k1, secp192r1 (P-192), secp224k1, secp224r1 (P-224), secp256k1, secp256r1 (P-256), secp384r1 (P-384), secp521r1 (P-521)

1.5 使用上の注意

SHALO AUTH が PC に認識されない場合、一度 SHALO AUTH を取り外してから再度装着してください。

SHALO AUTH を自己給電（セルフパワー）USB ハブに接続する場合、PC をシャットダウン・スタンバイするか USB ハブを PC から取り外した際に、USB ハブから SHALO AUTH を取り外してください。PC 起動後に SHALO AUTH が認識されない場合があります。

第 2 章

SHALO AUTH の準備をする

この章では、SHALO AUTH をご利用になる前にお読みいただきたい内容について説明します。

この章のトピック

1. SHALO AUTH の外観と機能
2. U2F を理解する
3. PKCS #11 を理解する
4. SHALO AUTH 専用ソフトウェアの紹介

2.1 SHALO AUTH の外観と機能

SHALO AUTH は正面にライト 1 個、側面にボタン 1 個持ちます。USB プラグはキャップで保護されています。使用する際は SHALO AUTH のキャップを外して USB ポートに装着します。

図 6　SHALO AUTH の外観説明

白色ライト

通常、白色ライトは消灯しています。SHALO AUTH の状態をユーザーに知らせる際に点灯します。白色ライトが知らせる内容と点灯パターンの対応は次の通りです。

タイミング	点灯パターン	解説
PC に装着後	点灯	セルフテストを実施中です。
	1 秒あたり 1～3 回の点滅	セルフテストで異常が検出されました。
ソフトウェア操作中	点灯	データを書き込み中です。 PC から取り外さないでください。
	1 秒あたり 5 回の点滅	SHALO Keyring/SHALO Smith が起動している場合、それらの選択・操作対象となっていることを示します。 そうでない場合は U2F でユーザーの承認を待っています。承認するにはボタンを押します。
約 30 秒ボタン長押し	1 秒あたり 10 回の点滅	管理 PIN を省略して購入状態に戻せる状態です。 10 秒間続きます。

ボタン

ボタンは主にユーザーが SHALO AUTH の動作を承認する場合に使われます。次節で説明します。

2.2 U2F を理解する

SHALO AUTH は U2F セキュリティキーとして使用できます。U2F とはウェブサービスで本人確認するための仕組みで、FIDO アライアンスにより策定されています。

U2F の本人確認の仕組みは次の 2 要素を組み合わせたものです。

知識　ID やパスワードといった「本人が知っていること」
所有　USB トークンやスマートフォンなど「本人が持っているもの」

このように 2 つの要素を使用して認証する本人確認方法を**二要素認証（2FA: Tow-Factor Authentication）**といいます。

U2F セキュリティキーに本人確認の処理を行わせる際、ユーザーは物理的な動きで U2F セキュリティキーに許可を与えます。SHALO AUTH ではライト点滅でユーザーに許可を求め、ユーザーは SHALO AUTH 側面のボタンを押して許可を与えます。

類似の用語に**二段階認証（Two-step verification）**があります。これは ID とパスワードによる認証（一段階目）を行った後に、別の認証（二段階目）を行って本人確認する方法です。一段階目と二段階目で異なる要素を採用しているかどうかは考慮されません。

SHALO AUTH の使用を**許可しない場合はボタンを押してはいけません。**ユーザーの操作なしでライトが点滅した場合、悪意を持つソフトウェアが密かに SHALO AUTH を利用しようとしている恐れがあります。

U2F 利用の流れ

U2F に対応したウェブサービスの本人確認で SHALO AUTH の U2F を利用するには、U2F に対応したウェブブラウザと USB を持つ PC が必要です。

U2F は次の 3 通りの場面で使われます。

1. セキュリティキーの登録
2. 本人確認
3. セキュリティキーの登録解除

ウェブサービスでセキュリティキーを登録解除すると、セキュリティキーを廃棄・譲渡した後でセキュリティキー取得者による成り済ましを防げます。

本節でそれぞれの場面について順に説明します。

セキュリティキー登録の仕組み

ウェブサービスのユーザー設定でセキュリティキーを登録します。この登録はウェブブラウザで簡単にできます。具体的には次の 3 つのステップを踏みます。

ステップ 1　SHALO AUTH をウェブサービスに登録する作業中、SHALO AUTH のライトが点滅してユーザーに SHALO AUTH の使用許可を求めます。

ステップ 2　SHALO AUTH のボタンを押して許可すると、SHALO AUTH はこのウェブサービス専用の FIDO 認証鍵（プライベート鍵と公開鍵のペア）を生成します。

ステップ 3　SHALO AUTH の情報と生成した公開鍵をウェブサーバーに登録します。

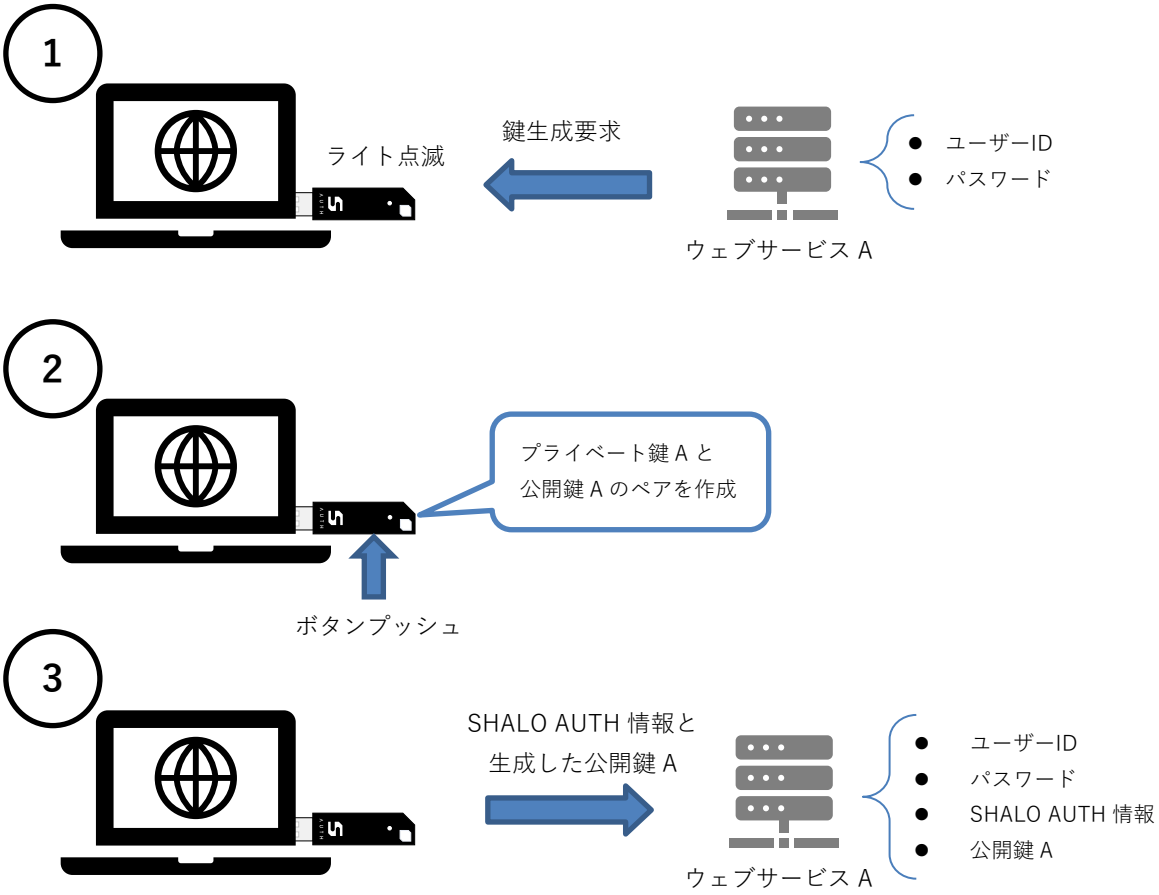


図 7　U2F セキュリティキー登録

多くのウェブサービスでは二要素認証を利用する場合、**他の認証方法または復旧方法と併用することを強く推奨しています。**これはセキュリティキーの破損・紛失により本人確認ができなくなることを防ぐためです。

本人確認の仕組み

U2F を利用した本人確認では、まずユーザーID とパスワードを入力します。それから次の 3 つのステップを踏みます。

ステップ 1　ウェブサービスから認証要求を受け取ると SHALO AUTH のライトが点滅し、ユーザーに SHALO AUTH の使用許可を求めます。

ステップ 2　SHALO AUTH のボタンを押して許可すると、SHALO AUTH はこのウェブサービス専用のプライベート鍵を使ってデジタル署名を生成します。

ステップ 3　SHALO AUTH の生成したデジタル署名をウェブサービスに送ります。ウェブサービスは SHALO AUTH が生成したデジタル署名をユーザー情報に登録された公開鍵で検証することで本人確認します。

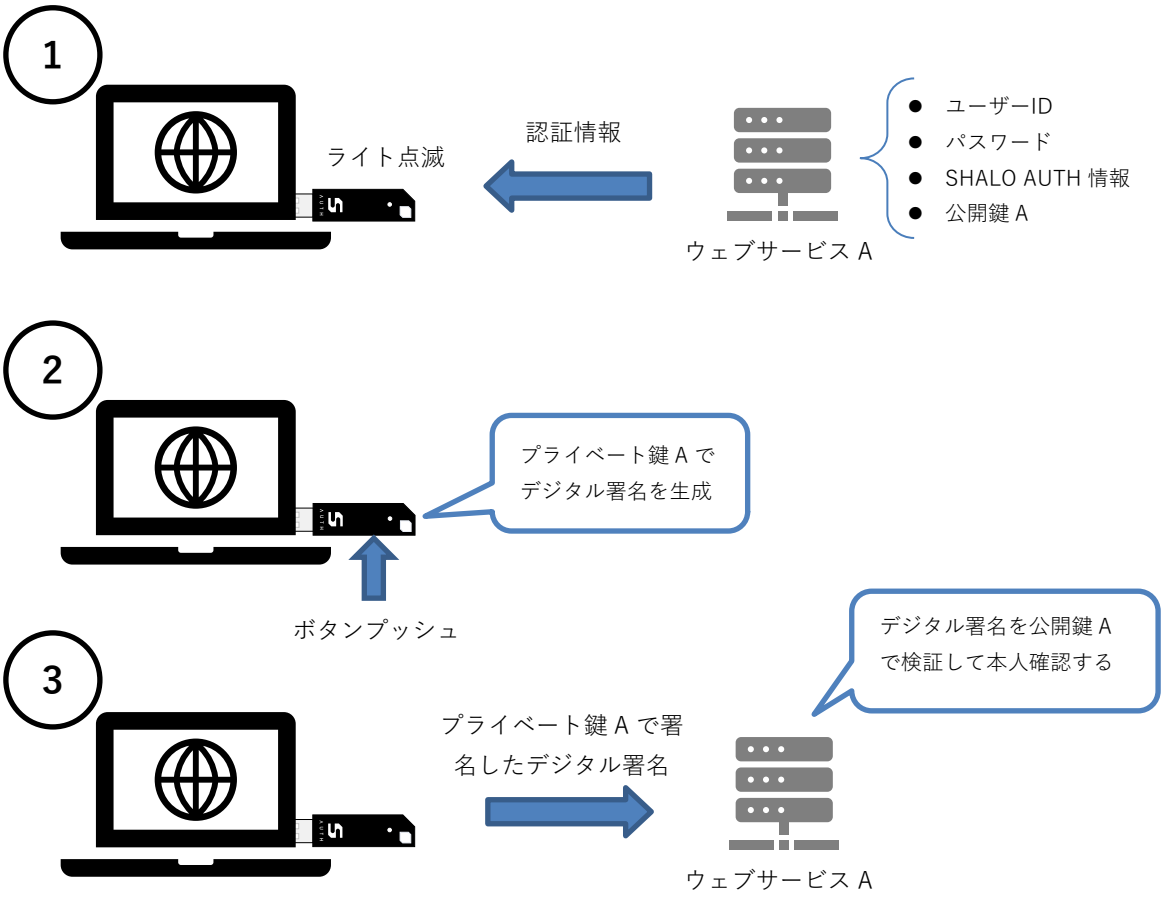


図 8　U2F の本人確認

セキュリティキーの登録解除の仕組み

ウェブサービスでセキュリティキーの登録を解除するには、ウェブサービスのユーザー設定で登録済みの U2F セキュリティキーを削除します。

この処理はウェブサービスが保持していた SHALO AUTH 情報と公開鍵を削除します。ウェブサービスごとに異なる FIDO 認証鍵を登録しているため、１つのウェブサービスで登録解除しても他のウェブサービスには影響しません。

SHALO AUTH は譲渡・廃棄される場合にそなえて、これまで生成した**すべての FIDO 認証鍵を無効化**させることができます。これは SHALO AUTH から U2F で使用される情報をすべて削除し、新しい SHALO AUTH として扱われるようにします。これによりウェブサービスで登録解除漏れがあっても次の SHALO AUTH 取得者による成り済ましを防ぐことができます。

2.3 PKCS #11 を理解する

2.3.1 PKCS #11 とは？

PKCS #11 は暗号トークンをソフトウェアから扱う API で、暗号トークンを使ってデジタル署名やユーザー認証するアプリケーションに広く使われています。

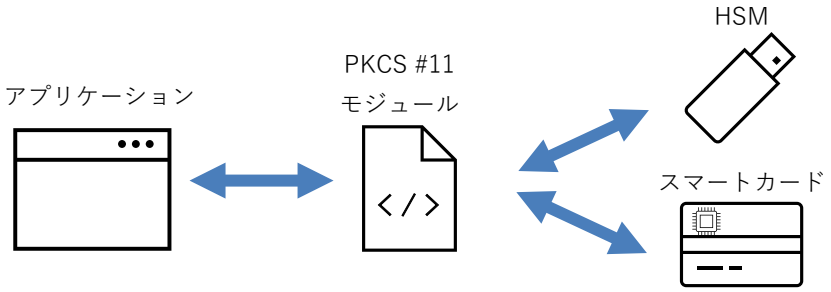


図 9　PKCS #11 の位置づけ

暗号トークン

暗号トークンとは HSM (Hardware Security Module)やスマートカードといった暗号装置を指します。HSM とは、暗号鍵の安全に保管しその暗号鍵を使用して暗号処理を行う装置です。スマートカードは IC チップを内蔵した IC カードを指し、その働きは HSM と同様です。スマートカードの身近な例として、**PIN（暗証番号）**を使う以下のカードが挙げられます。

- クレジットカード
- キャッシュカード
- マイナンバーカード

PKCS #11 の機能

PKCS #11 は大きく分けて 3 つの機能を暗号トークンに提供します。これを次の表にまとめます。

機能	概要
PIN 認証	PIN（暗証番号）による認証で以下の利用者を区別できます。 ● セキュリティ管理者（SO: Security Officer） ● 使用者（User） ● 公開利用者（Public） 一定回数の PIN 間違いで PIN をロックして暗号トークンを保護できます。
データ管理	暗号鍵や証明書などを暗号トークンに安全に管理できます。 これらのデータは暗号処理に使用されます。 データごとに所有者以外の使用を制限することができ、また暗号トークンからの取り出しも永久的に禁止できます。
暗号処理	保存された鍵でデータの暗号化・復号、デジタル署名を作成・検証できます。 メッセージダイジェストや高セキュリティの乱数を生成できます。

2.3.2 PIN 認証

PKCS #11 は PIN を入力して次の 2 種類の役割を認証できます。

セキュリティ管理者　暗号トークンの発行業務と PIN 管理を行います
使用者　暗号トークンの持つ秘密情報を使って暗号処理を行います

使用者は暗号トークンの所持者です。暗号トークンの公開情報利用に PIN 認証は不要です。SHALO AUTH ではセキュリティ管理者の PIN を**管理 PIN**、使用者の PIN を**ユーザーPIN** と呼称します。各 PIN には 4〜256 文字の英数記号を指定できます。

SHALO AUTH を個人で購入した場合、購入者はセキュリティ管理者と使用者の両方の役割を持ちます。両者の PIN に同じ PIN を設定することもできます。

機能と PIN の関係

各機能と実行に必要な PIN の対応を次の表にまとめます。

機能		必要な PIN
管理	暗号トークンの初期設定	管理 PIN（初回は不要）
	管理 PIN の変更	管理 PIN
	ユーザーPIN の設定・ロック解除	管理 PIN
通常利用	ユーザーPIN の変更	ユーザーPIN
	暗号トークンの保護されたデータの作成・読み出し・削除	ユーザーPIN
	暗号トークンの保護されたデータの鍵を使う暗号処理	ユーザーPIN
	暗号トークンの公開データの作成・読み出し・削除	不要
	暗号トークンの公開データの鍵を使う暗号処理	不要
	暗号トークンのデータを使わない暗号処理	不要

PIN のロック

PIN 認証に**連続で 5 回失敗すると PIN がロックされます。**その後、ロックが解除されるまで PIN 認証を禁止します。各 PIN のロック解除方法は以下の通りです。

PIN の種類	ロックの解除方法
ユーザーPIN	セキュリティ管理者としてユーザーPIN を再設定します。
管理 PIN	暗号トークンを購入時の状態に戻します。 **すべての情報と FIDO 認証鍵が削除されます。**

SHALO AUTH を PKCS #11 向けに初期設定しても FIDO 認証鍵は削除されません。
購入時の状態に戻した場合はすべての FIDO 認証鍵が削除されます。

2.3.3 データ管理

データ容量

SHALO AUTH は PKCS #11 で定義される以下の 3 種のデータを全部で最大 12 個保存できます。

- 公開鍵暗号（RSA または ECDSA）のプライベート鍵
- 公開鍵暗号（RSA または ECDSA）の公開鍵
- X.509 証明書

SHALO AUTH 専用ソフトウェアは 1 つの鍵を SHALO AUTH に保存する際にこの 3 種類をセットで保存します。このソフトウェアを使用すると 4 セットの鍵を保存できます。

X.509 証明書は公開鍵の情報を含むため、X.509 証明書か公開鍵のどちらかに統一すれば SHALO AUTH は最大 6 セットの鍵ペアを扱えます。
その場合はデータ管理に他の PKCS #11 アプリケーションを使用してください。

データセットの識別

PKCS #11 では複数のデータ間で関連性を識別する **CKA_ID 属性**と呼ばれる情報をデータに付与します。同じ CKA_ID 属性を持つデータは同じセットとみなされます。

ある鍵ペアを構成するプライベート鍵と公開鍵は同じ CKA_ID 属性を持ちます。その公開鍵に対して発行される X.509 証明書も同じ CKA_ID 属性を持ちます。

データの保護

SHALO AUTH は各データに対するデータ保護方法として以下を指定できます。

- データ変更の禁止
- データ削除の禁止
- データのアクセス・利用にユーザーPIN 認証を必要とする
- データの外部への読み出し禁止（プライベート鍵のみ）

SHALO AUTH 専用ソフトウェアでデータを保存した場合、各データは以下のように管理されます。この条件以外で運用する場合は、他の PKCS #11 アプリケーションを使用してください。

データ種別	変更	削除	ユーザーPIN 認証保護	外部への読み出し
公開鍵暗号のプライベート鍵	可	可	あり	不可
公開鍵暗号の公開鍵	可	可	なし	可
X.509 証明書	可	可	なし	可

2.4　SHALO AUTH 専用ソフトウェアの紹介

SHALO AUTH は汎用セキュリティキー向けに次の 2 つのソフトウェアを提供しています。

SHALO Keyring　SHALO AUTH に鍵データを保存するソフトウェア
SHALO Smith　SHALO AUTH を管理するソフトウェア

SHALO Smith は SHALO AUTH を廃棄・譲渡する際にすべての FIDO 認証鍵を無効化させる用途にも使われます。

SHALO Keyring

SHALO Keyring は汎用セキュリティキーで扱う暗号鍵を SHALO AUTH に格納するためのソフトウェアです。

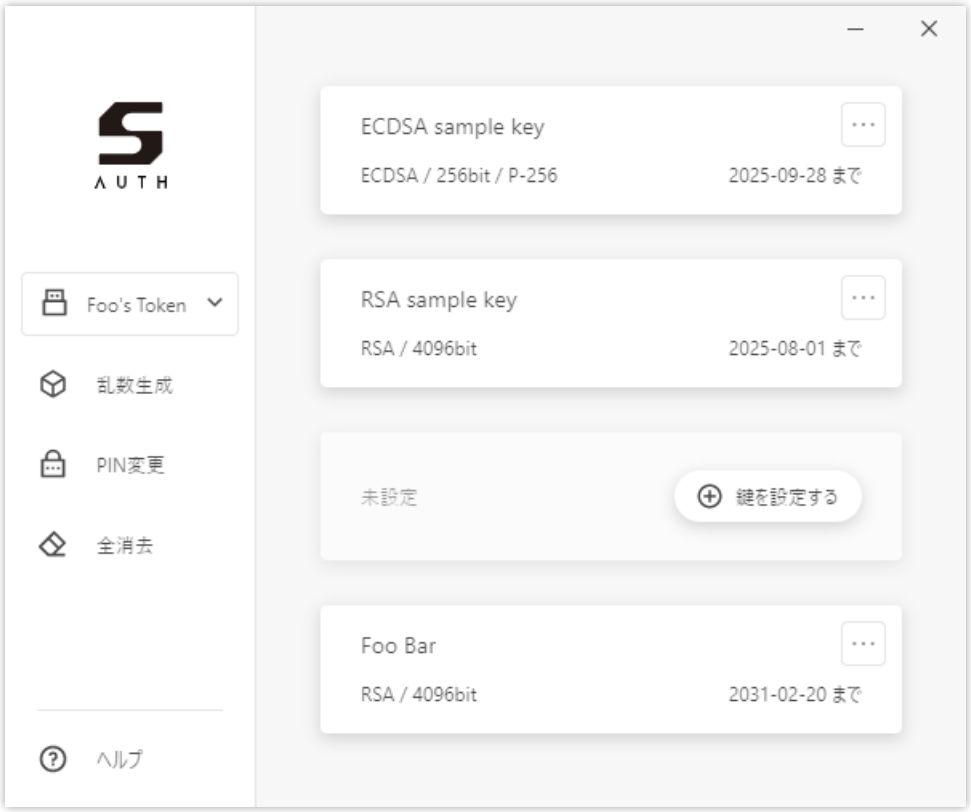


図 10　SHALO Keyring のウィンドウ

SHALO Keyring は以下の機能を提供します。これらの操作に PKCS #11 の管理 PIN は不要です。

- SHALO AUTH のセットアップ
- 暗号鍵の追加・削除
- ユーザーPIN の変更
- パスワードや乱数列の生成

SHALO Keyring の使用法は第 4 章で解説します。

SHALO Smith

SHALO Smith は SHALO AUTH を管理するためのソフトウェアです。

図 11　SHALO Smith のウィンドウ

SHALO Smith は以下の機能を提供します。これらの操作には管理 PIN が必要です。

- SHALO AUTH のセットアップ
- 管理 PIN の変更
- ユーザーPIN の再設定・ロック解除
- 購入時の状態への復帰とすべての FIDO 認証鍵の削除

SHALO Smith の使用法は第 5 章で解説します。

第 3 章
インストール

この章では、SHALO AUTH 専用ソフトウェアのインストール方法とアンインストール方法を説明します。

SHALO AUTH 専用ソフトウェアは以下の OS に対応しています。

- Windows
- macOS
- Linux

この章のトピック

1. Windows にインストールする
2. Windows からアンインストールする
3. macOS にインストールする
4. macOS からアンインストールする
5. Linux にインストールする
6. Linux からアンインストールする

3.1 Windows にインストールする

SHALO AUTH は Windows に含まれる標準ドライバで動作します。SHALO AUTH を初めて PC の USB ポートに接続すると自動的にセットアップが始まります。セットアップが完了すると下図のようにデスクトップに通知が表示されます。

図 12　SHALO AUTH のセットアップ完了通知

SHALO AUTH を U2F セキュリティキーとしてだけ使う場合、この節の残りの説明は不要です。

3.1.1 SHALO Keyring のインストール

https://auth.shalo.jp より Windows 向けの SHALO Keyring をダウンロードできます。

インストールするにはダウンロードしたファイル shalo_keyring_*x.y.z*_windows.exe（x.y.z はバージョン番号）を実行します。インストールは 3 ステップで完了します。

まず、最初の画面でインストールオプションを選びます。管理者権限のないユーザーは[**現在のユーザーのみにインストールする**]を選択してください。

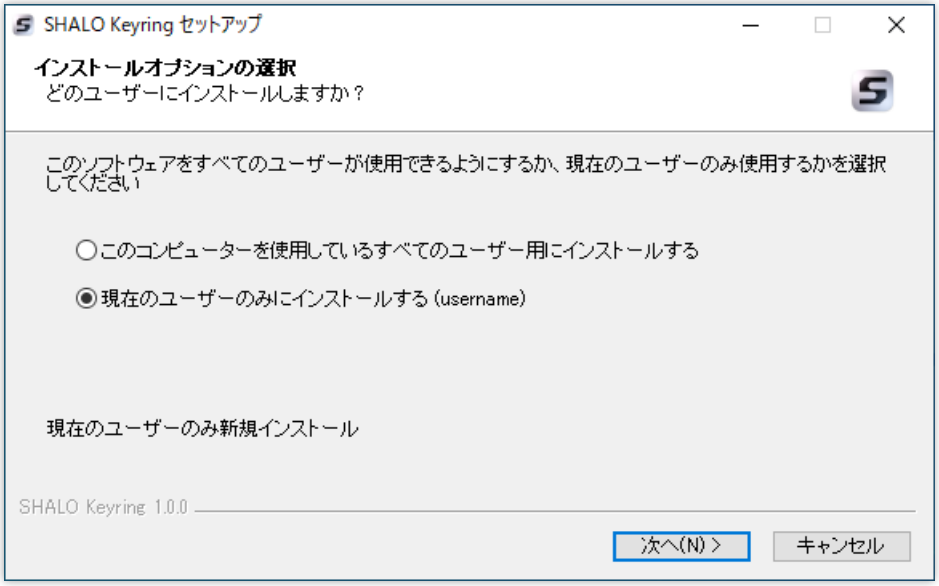


図 13　SHALO Keyring のインストールオプション

次にインストール先を指定できます。問題がなければ[**インストール**]をクリックします。

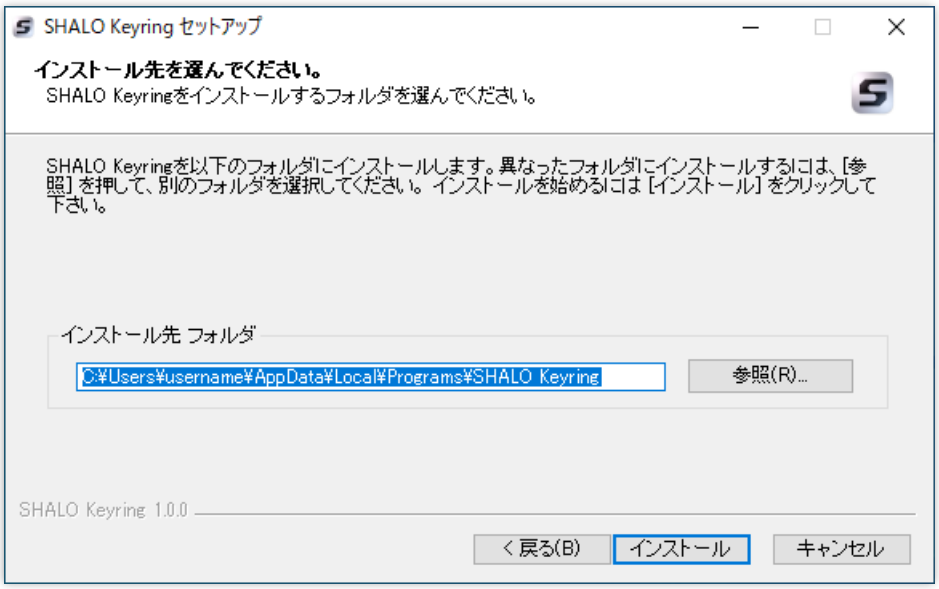


図 14　SHALO Keyring のインストール先の指定

インストールが完了すると以下のように表示されます。[**完了**]をクリックして終了します。

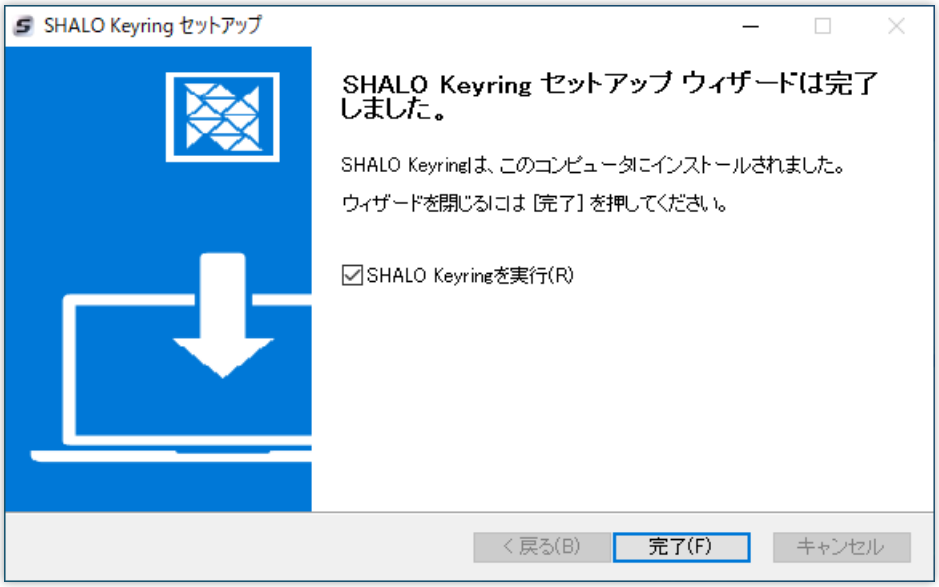


図 15　SHALO Keyring のインストール完了

SHALO Keyring は、デスクトップやスタートメニューに作成されたショートカットから起動できます。

3.1.2　SHALO Smith のインストール

https://auth.shalo.jp より Windows 向けの SHALO Smith をダウンロードできます。

インストールするにはダウンロードしたファイル shalo_smith_*x.y.z*_windows.exe（x.y.z はバージョン番号）を実行します。インストールは 3 ステップで完了します。

まず、最初の画面でインストールオプションを選びます。管理者権限のないユーザーは[**現在のユーザーのみにインストールする**]を選択してください。

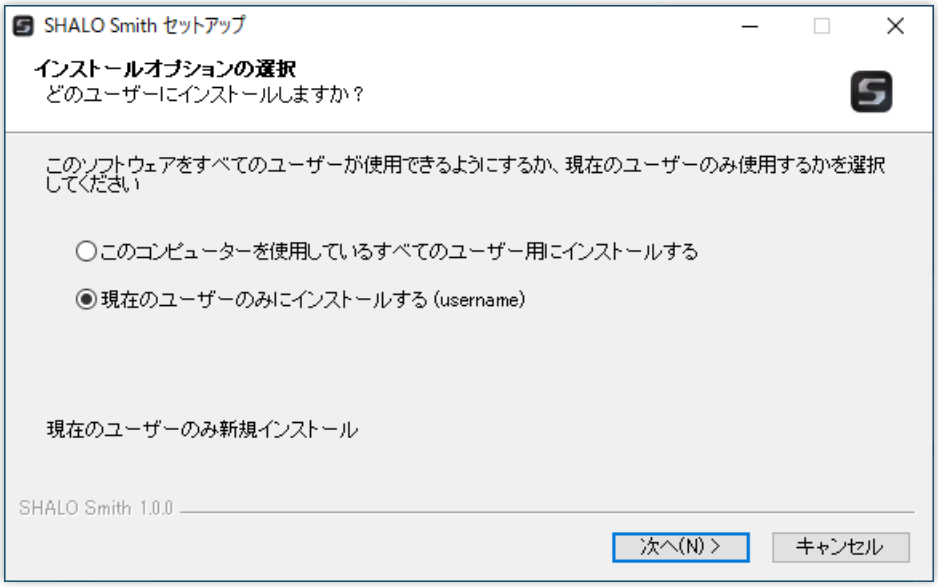


図 16　SHALO Smith のインストールオプション

次にインストール先を指定できます。問題がなければ[**インストール**]をクリックします。

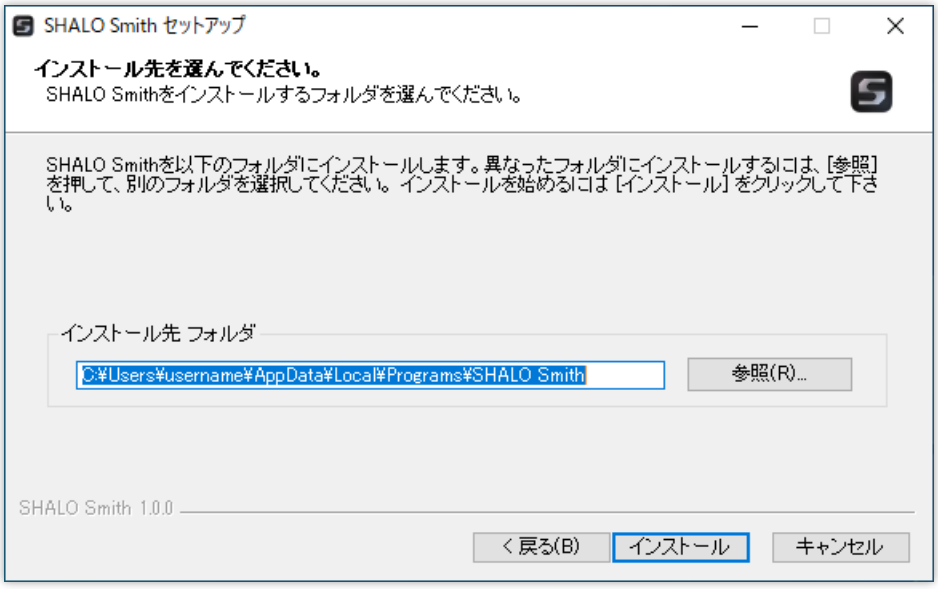


図 17　SHALO Smith のインストール先の指定

インストールが完了すると以下のように表示されます。[**完了**]をクリックして終了します。

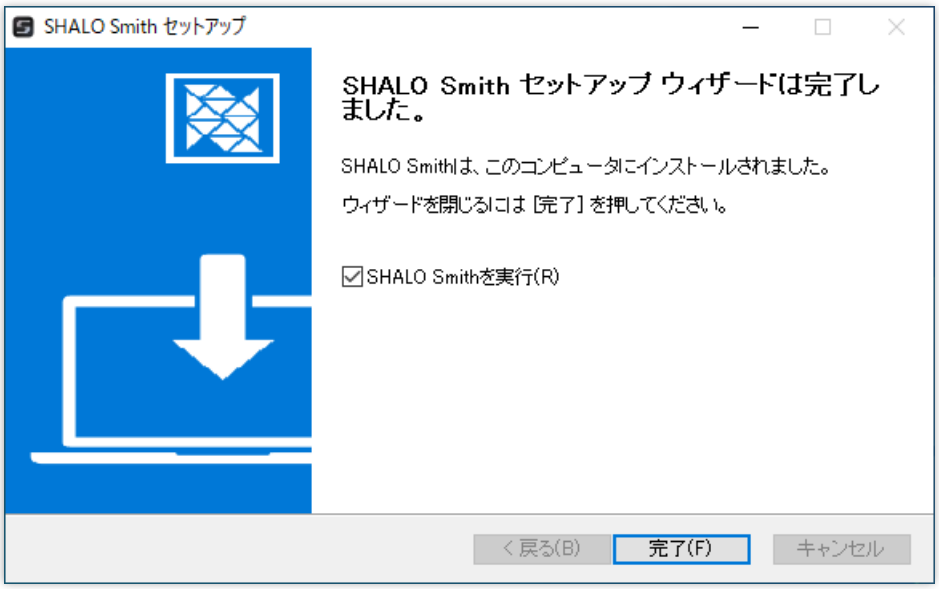


図 18　SHALO Smith のインストール完了

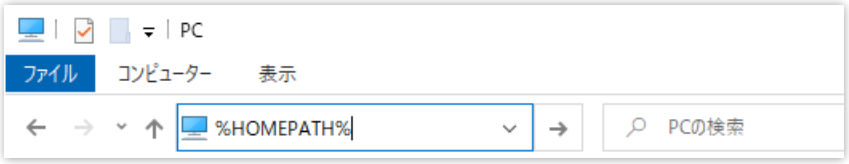
SHALO Smith は、デスクトップやスタートメニューに作成されたショートカットから起動できます。

3.1.3　PKCS #11 モジュールのインストール

https://auth.shalo.jp より Windows 向けの PKCS #11 モジュールをダウンロードできます。

PKCS #11 モジュールをインストールするには、ホームディレクトリの shalo_pkcs11 フォルダにダウンロードした ZIP ファイルを展開します。次の手順で行います。

1. エクスプローラーでホームディレクトリに移動します。
   下のようにエクスプローラーのアドレスバーで「`%HOMEPATH%`」と入力してエンターキーを押すと移動できます。



2. 「`shalo_pkcs11`」という名前のフォルダを作ります。
3. ダウンロードした shalo_pkcs11_*x.y.z* _windows.zip（x.y.z はバージョン番号）を右クリックして、メニューで[**すべて展開...**]を選択します。
4. ファイルの展開先フォルダに 2.で作成したフォルダを指定します。

インストール後の shalo_pkcs11 フォルダは次のようになります。

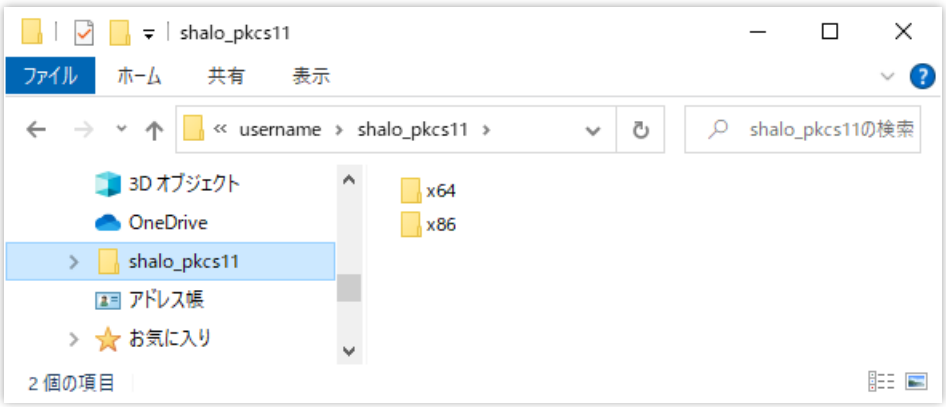


図 19　ホームディレクトリの shalo_pkcs11 フォルダ

システムドライブが C:の場合、作成した shalo_pkcs11 フォルダの絶対パスは次のとおりです。*username* は自身のユーザー名に読み替えてください。

```
C:¥Users¥username¥shalo_pkcs11
```

インストールされた PKCS #11 モジュールのアプリケーション用途別のパスは次の通りです。

モジュールの用途	モード	ホームディレクトリからの相対パス
Windows アプリケーション	32 bit	`shalo_pkcs11¥x86¥slpkcs11-vc.dll`
	64 bit	`shalo_pkcs11¥x64¥slpkcs11-vc.dll`
Windows に移植したアプリケーション（MinGW, Cygwin, Git for Windows）	32 bit	`shalo_pkcs11¥x86¥slpkcs11-mingw32.dll`
	64 bit	`shalo_pkcs11¥x64¥slpkcs11-mingw64.dll`

3.2 Windows からアンインストールする

3.2.1 SHALO Keyring のアンインストール

SHALO Keyring をアンインストールするには、次の手順で行います。

1. Windows のスタートボタンを右クリックし、[**アプリと機能**]をクリックします。
2. アプリの一覧で「SHALO Keyring」をクリックして、[**アンインストール**]ボタンをクリックします。SHALO Keyring アンインストールウィンドウが表示されます。
3. アンインストールウィンドウで[**次に**]をクリックします。
4. 最後に[**完了**]ボタンをクリックしてアンインストールウィンドウを終了します。

3.2.2 SHALO Smith のアンインストール

SHALO Smith をアンインストールするには、次の手順で行います。

1. Windows のスタートボタンを右クリックし、[**アプリと機能**]をクリックします。
2. アプリの一覧で「SHALO Smith」をクリックして、[**アンインストール**]ボタンをクリックします。SHALO Smith アンインストールウィンドウが表示されます。
3. アンインストールウィンドウで[**次に**]をクリックします。
4. 最後に[**完了**]ボタンをクリックしてアンインストールウィンドウを終了します。

3.2.3 PKCS #11 モジュールのアンインストール

PKCS #11 モジュールをアンインストールするには、エクスプローラーで PKCS #11 モジュールのインストール先フォルダを削除します。これは次の手順で行います。

1. PKCS #11 モジュールを使用しているソフトウェアを終了させます。必要であればそれらから PKCS #11 モジュールの登録を削除します。
2. エクスプローラーでホームディレクトリに移動します。
3. shalo_pkcs11 フォルダを削除します。

PKCS #11 モジュールを Acrobat® に登録している場合は、7.2.2 節に従って Acrobat®から PKCS #11 モジュールを削除してください。
認証エージェントを自動起動させている場合は、認証エージェントから SHALO AUTH を削除してください。

3.3 macOS にインストールする

SHALO AUTH は macOS に含まれる標準ドライバで動作します。SHALO AUTH を U2F セキュリティキーとしてだけ使う場合、この節の説明は不要です。

3.3.1 SHALO Keyring のインストール

https://auth.shalo.jp より macOS 向けの SHALO Keyring をダウンロードできます。

インストールするにはダウンロードしたファイル shalo_keyring_*x.y.z*_macos.dmg（x.y.z はバージョン番号）をダブルクリックして開きます。

下図のウィンドウが表示されるので、左の SHALO Keyring アイコンを右の Application フォルダにドラッグ&ドロップするとインストールは完了します。

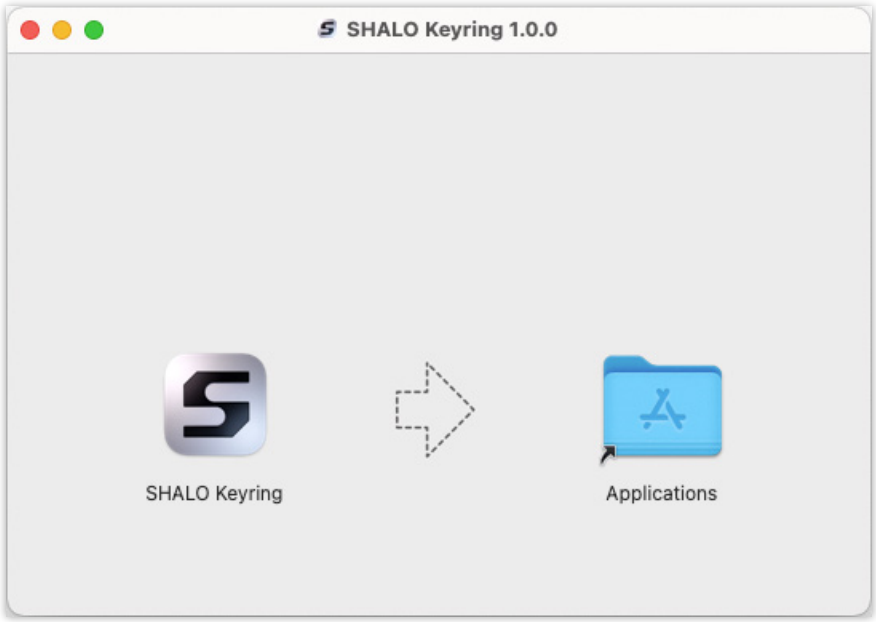


図 20　SHALO Keyring のインストール

SHALO Keyring は Launchpad またはアプリケーションフォルダから起動できます。

3.3.2 SHALO Smith のインストール

https://auth.shalo.jp より macOS 向けの SHALO Smith をダウンロードできます。

インストールするにはダウンロードしたファイル shalo_smith_*x.y.z*_macos.dmg（x.y.z はバージョン番号）をダブルクリックして開きます。

下図のウィンドウが表示されるので、左の SHALO Smith アイコンを右の Application フォルダにドラッグ&ドロップするとインストールは完了します。

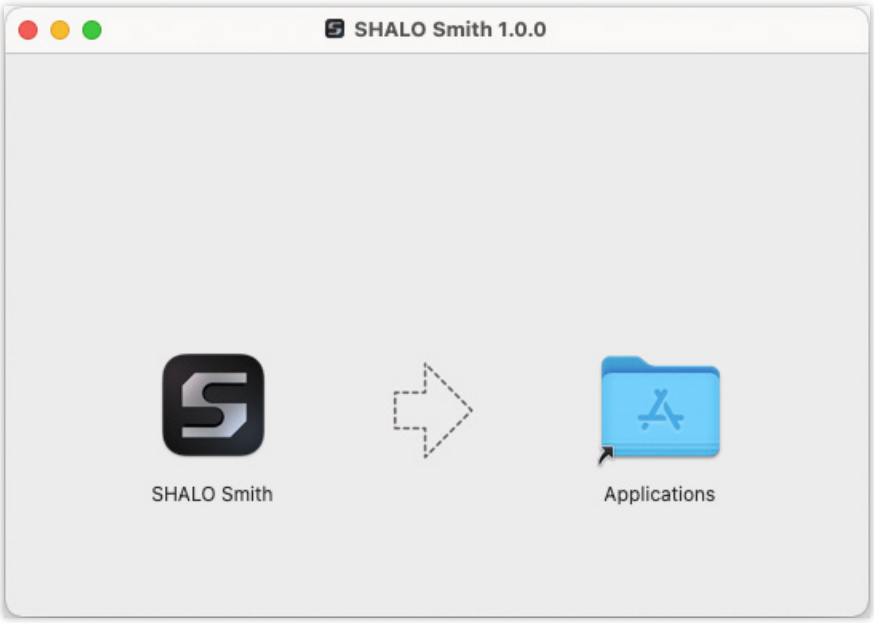


図 21　SHALO Smith のインストール

SHALO Smith は Launchpad またはアプリケーションフォルダから起動できます。

3.3.3 PKCS #11 モジュールのインストール

https://auth.shalo.jp より macOS 向けの PKCS #11 モジュールをダウンロードできます。

ダウンロードしたファイル shalo_pkcs11_*x.y.z*_macos.zip（x.y.z はバージョン番号）を Finder でダブルクリックすると、ZIP ファイルが展開されて PKCS #11 モジュールのファイル libslpkcs11.dylib が作成されます。

インストールするにはこの libslpkcs11.dylib を`/usr/local/lib` に **root 権限でコピー**します。

`/usr/local/lib` は ssh-agent の既定のホワイトリストです。

ダウンロードフォルダで ZIP ファイルを展開した場合は、ターミナルを開いて次のコマンドを実行します。

```
% sudo cp ~/Downloads/libslpkcs11.dylib /usr/local/lib/
```

異なるフォルダで `ZIP` ファイルを展開した場合は`~/Downloads` をそのフォルダパスに変更してください。

macOS の環境によっては以下のメッセージが表示される場合があります。

```
cp: directory /usr/local/lib does not exist
```

その場合は次のようにして root 権限でディレクトリを作成してからファイルをコピーします。

```
% sudo mkdir -p /usr/local/lib
% sudo cp ~/Downloads/libslpkcs11.dylib /usr/local/lib/
```

3.4 macOS からアンインストールする

3.4.1 SHALO Keyring のアンインストール

SHALO Keyring をアンインストールするには、次の手順で行います。

1. Finder でアプリケーションフォルダを開きます。
2. アプリケーションフォルダの SHALO Keyring をゴミ箱にドロップするか、SHALO Keyring を選択して Command-Delete を押します。
3. ターミナルを開いて次のコマンドを実行し、設定ファイルを削除します。

```
% rm -r ~/Library/Application¥ Support/shalo-keyring↵
```

3.4.2 SHALO Smith のアンインストール

SHALO Smith をアンインストールするには、次の手順で行います。

1. Finder でアプリケーションフォルダを開きます。
2. アプリケーションフォルダの SHALO Smith をゴミ箱にドロップするか、SHALO Smith を選択して Command-Delete を押します。
3. ターミナルを開いて次のコマンドを実行し、設定ファイルを削除します。

```
% rm -r ~/Library/Application¥ Support/shalo-smith↵
```

3.4.3 PKCS #11 モジュールのアンインストール

PKCS #11 モジュールをアンインストールするには、**/usr/local/lib** から libslpkcs11.dylib を **root 権限で削除**します。ターミナルを開いて次のコマンドを実行します。

```
% sudo rm /usr/local/lib/libslpkcs11.dylib↵
```

PKCS #11 モジュールを Acrobat®に登録している場合は、7.2.2 節に従って Acrobat®から PKCS #11 モジュールを削除してください。
認証エージェントを自動起動させている場合は、認証エージェントから SHALO AUTH を削除してください。

3.5 Linux にインストールする

SHALO AUTH は Linux に含まれる標準ドライバで動作します。ただし root 権限なしで使用するには 3.5.1 節に従った操作が必要です。これは SHALO AUTH を U2F セキュリティキーとして使う場合も同様です。

3.5.1 udev ルールファイルのインストール

root 権限を持たないユーザーが SHALO AUTH を使用するには、SHALO AUTH の udev ルールファイルをインストールする必要があります。

SHALO AUTH 向けの udev ルールファイルは次の Linux 向けダウンロードソフトウェアに 60-usb-shalo-auth.rules というファイル名で含まれています。

- SHALO Keyring
- SHALO Smith
- PKCS #11 モジュール

インストールするには、ダウンロードしたファイルに含まれる 60-usb-shalo-auth.rules を `/etc/udev/rules.d` へ **root 権限でコピー**し、`udevadm` コマンドを実行して新しいルールを即時適用します。

これはターミナルを開いて次のようにします。

```
$ tar xvzf ダウンロードしたファイル
$ sudo cp 60-usb-shalo-auth.rules /etc/udev/rules.d/
$ sudo udevadm control --reload-rules
```

udev ルールはインストール時にすでに装着されている SHALO AUTH には適用されません。適用し直すには SHALO AUTH を装着し直してください。

3.5.2 必要なライブラリのインストール

SHALO Keyring と SHALO Smith を利用するには、Linux に libfuse2 をインストールしておく必要があります。Ubuntu 22.04 LTS 以降ではターミナルを開いて次のようにしてインストールします。

```
$ sudo apt update
$ sudo apt -y install libfuse2
```

Ubuntu 22.04 LTS よりも前の Ubuntu や他のディストリビューションには通常 libfuse2 が含まれるため、この手順は不要です。

3.5.3 SHALO Keyring のインストール

https://auth.shalo.jp より Linux 向けの SHALO Keyring をダウンロードできます。

ダウンロードしたファイル shalo_keyring_*x.y.z*_linux.tar.gz（x.y.z はバージョン番号）を展開するには、ターミナルで次のコマンドを実行します。

```
$ tar xvzf shalo_keyring_x.y.z_linux.tar.gz
shaloKeyring.appimage
60-usb-shalo-auth.rules
```

これによって以下のファイルが作成されます。

shaloKeyring.appimage	Linux 向け SHALO Keyring
60-usb-shalo-auth.rules	SHALO AUTH 向け udev ルールファイル

udev ルールファイル（60-usb-shalo-auth.rules）をインストールしていない場合は、3.5.1 節を参照してインストールしてください。
libfuse2 をインストールしていない場合は、3.5.2 節を参照してインストールしてください。

shaloKeyring.appimage に決められたインストール先はありません。管理しやすい場所に配置してください。SHALO Keyring を起動するには shaloKeyring.appimage を実行します。

3.5.4 SHALO Smith のインストール

https://auth.shalo.jp より Linux 向けの SHALO Smith をダウンロードできます。

ダウンロードしたファイル shalo_smith_*x.y.z*_linux.tar.gz（x.y.z はバージョン番号）を展開するには、ターミナルで次のコマンドを実行します。

```
$ tar xvzf shalo_smith_x.y.z_linux.tar.gz
shaloSmith.appimage
60-usb-shalo-auth.rules
```

これによって以下のファイルが作成されます。

shaloSmith.appimage	Linux 向け SHALO Smith
60-usb-shalo-auth.rules	SHALO AUTH 向け udev ルールファイル

udev ルールファイル（60-usb-shalo-auth.rules）をインストールしていない場合は、3.5.1 節を参照してインストールしてください。
libfuse2 をインストールしていない場合は、3.5.2 節を参照してインストールしてください。

shaloSmith.appimage に決められたインストール先はありません。管理しやすい場所に配置してください。SHALO Smith を起動するには shaloSmith.appimage を実行します。

3.5.5 PKCS #11 モジュールのインストール

https://auth.shalo.jp より Linux 向けの PKCS #11 モジュールをダウンロードできます。

ダウンロードしたファイル shalo_pkcs11_*x.y.z*_linux.tar.gz（x.y.z はバージョン番号）を展開するには、ターミナルで次のコマンドを実行します。

```
$ tar xvzf shalo_pkcs11_x.y.z_linux.tar.gz
libslpkcs11.so
60-usb-shalo-auth.rules
```

これによって以下のファイルが作成されます。

libslpkcs11.so	Linux 向け PKCS #11 モジュール
60-usb-shalo-auth.rules	SHALO AUTH 向け udev ルールファイル

udev ルールファイル（60-usb-shalo-auth.rules）をインストールしていない場合は、3.5.1 節を参照してインストールしてください。

インストールするには PKCS #11 モジュール libslpkcs11.so を`/usr/lib` に **root 権限でコピー**します。ターミナルを開いて次のようにします。

```
$ sudo cp libslpkcs11.so /usr/lib/
```

`/usr/lib` と`/usr/local/lib` は ssh-agent のホワイトリストに登録されているディレクトリです。
多くのデスクトップ向け Linux ディストリビューションは GUI ログイン時に ssh-agent を自動実行します。このとき起動される ssh-agent へのホワイトリスト追加は容易ではありません。
本書では既定のホワイトリストの 1 つの`/usr/lib` に PKCS #11 モジュールを配置して、起動オプションの変更を回避します。

3.6 Linux からアンインストールする

3.6.1 udev ルールファイルのアンインストール

udev ルールファイルをアンインストールするには、`/etc/udev/rules.d` から 60-usb-shalo-auth.rules を **root 権限で削除**します。これはターミナルを開いて次のようにします。

```
$ sudo rm /etc/udev/rules.d/60-usb-shalo-auth.rules
```

3.6.2 SHALO Keyring のアンインストール

SHALO Keyring をアンインストールするには、配置した shaloKeyring.appimage を削除します。そして、ターミナルで次のコマンドを実行して SHALO Keyring の設定ファイルを削除します。

```
$ rm -r ~/.config/shalo-keyring
```

3.6.3 SHALO Smith のアンインストール

SHALO Smith をアンインストールするには、配置した shaloSmith.appimage を削除します。そして、ターミナルで次のコマンドを実行して SHALO Smith の設定ファイルを削除します。

```
$ rm -r ~/.config/shalo-smith
```

3.6.4 PKCS #11 モジュールのアンインストール

PKCS #11 モジュールをアンインストールするには、`/usr/lib` から libslpkcs11.so を **root 権限で削除**します。ターミナルを開いて次のコマンドを実行します。

```
$ sudo rm /usr/lib/libslpkcs11.so
```

認証エージェントを自動起動させている場合は、認証エージェントから SHALO AUTH を削除してください。

第 4 章

鍵ツール SHALO Keyring を使う

この章では鍵ツール SHALO Keyring を説明します。SHALO Keyring は汎用セキュリティキー向けの暗号鍵を SHALO AUTH に設定するためのソフトウェアです。

SHALO AUTH を U2F セキュリティキーとしてだけ使用する場合、この章の説明は不要です。

この章のトピック

1. SHALO AUTH をセットアップする
2. SHALO AUTH の状態を確認する
3. 新しい鍵を生成する
4. 既存の鍵を取り込む
5. 鍵を削除する
6. 公開鍵を取得する
7. ユーザーPIN を変更する
8. パスワードや乱数列を生成する
9. 鍵データの CKA_ID 属性

4.1 SHALO AUTH をセットアップする

SHALO Keyring は新しい SHALO AUTH を検出すると図 22 のウィンドウを表示します。ここで[**セットアップを始める**]をクリックするとセットアップを開始できます。

図 22　SHALO Keyring による SHALO AUTH セットアップ

セットアップでは汎用セキュリティキー機能向けのデータ領域を初期化し、次の管理情報を設定します。

デバイスラベル　複数の SHALO AUTH を区別するために使われる個体名です。
ユーザーPIN　利用時のパスワードです。保護された暗号鍵の使用を許可します。
管理 PIN　管理用のパスワードです。ユーザーPIN を再設定する場合や、SHALO AUTH を購入時の状態に戻す際に使います。

このセットアップは U2F セキュリティキーの機能に影響を与えません。
セットアップ以前に U2F セキュリティキーとして SHALO AUTH を登録したウェブサービスはセットアップした SHALO AUTH で引き続き利用できます。

セットアップ済みの SHALO AUTH を再度セットアップするには SHALO Smith を使って SHALO AUTH を購入時の状態に戻す必要があります。購入時の状態に戻すと U2F セキュリティキーの情報も削除されます。

SHALO AUTH のセットアップは、デバイスラベル・ユーザーPIN・管理 PIN の順に設定します。

デバイスラベルの設定

デバイスラベルには、英字・数字・記号、日本語やその他言語の文字を使用できます。デバイスラベルの最大文字数は文字の種類に依存します。長すぎる場合は警告が表示されます。

図 23　デバイスラベルの設定

ユーザーPIN の設定

ユーザーPIN には英字・数字・記号を使用できます。4 文字以上 256 文字以下の長さでユーザーPIN を指定してください。確認のためにユーザーPIN を 2 回入力します。

図 24　ユーザーPIN の設定

管理 PIN の設定

管理 PIN には英字・数字・記号を使用できます。4 文字以上 256 文字以下の長さで管理 PIN を指定してください。確認のために管理 PIN を 2 回入力します。

図 25　管理 PIN の設定

セットアップが完了すると下のように表示されます。最後に[**アプリを始める**]をクリックします。

図 26　セットアップ完了した SHALO Keyring ウィンドウ

4.2 SHALO AUTH の状態を確認する

SHALO Keyring は SHALO AUTH の状態に適したウィンドウを表示します。SHALO AUTH がセットアップ完了直後の場合は図 27、SHALO AUTH に 1 つ以上の鍵が設定されている場合は図 28 のようなウィンドウが表示されます。

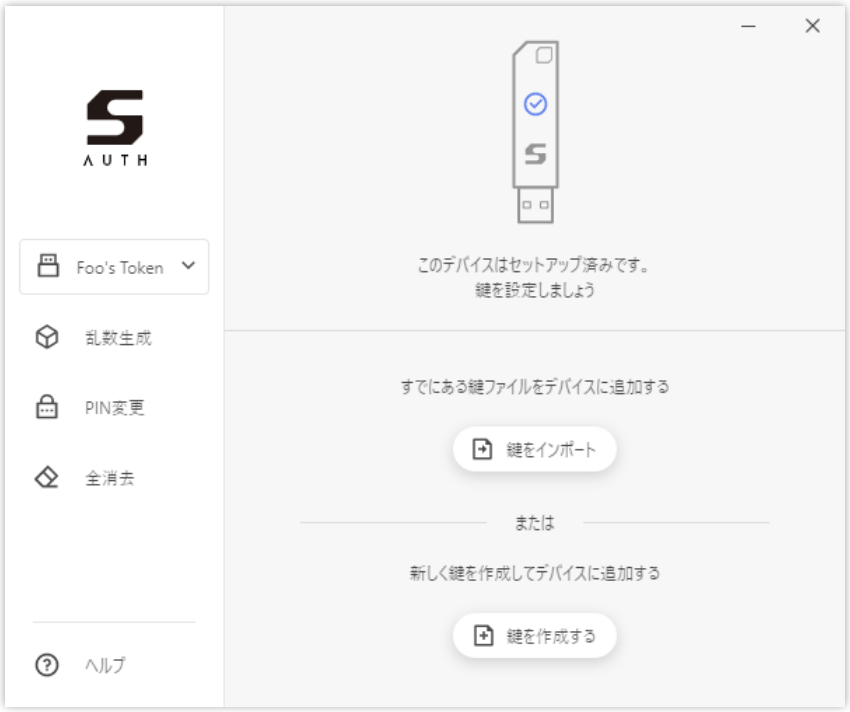
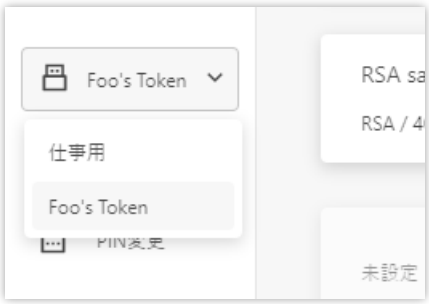


図 27　セットアップ直後の SHALO AUTH を PC に装着した場合

図 28　鍵が追加されている SHALO AUTH を PC に装着した場合

デバイスラベル

ここには情報表示・操作対象の SHALO AUTH のデバイスラベルが表示されます。



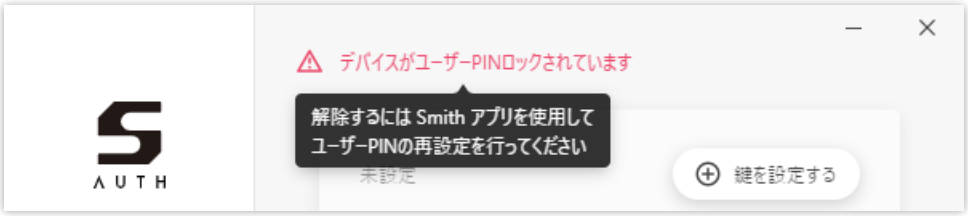
クリックすると PC に装着されている SHALO AUTH の一覧が表示されます。一覧からデバイスラベルを選択すると、ウィンドウの表示内容が選択された SHALO AUTH の情報に切り替わります。また[**PIN 変更**]や[**全消去**]などのデバイス操作は選択された SHALO AUTH に対して適用されます。

SHALO Keyring は SHALO AUTH を同時に最大 8 台まで取り扱うことができます。

選択されている SHALO AUTH はライトが点滅します。複数の SHALO AUTH を PC に接続している場合、ライト点滅の有無で対象デバイスを識別できます。

デバイスの状態

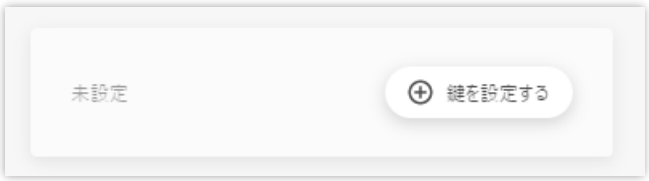
デバイスが異常な状態にある場合は、ウィンドウ上部に赤字で警告が表示されます。警告にマウスカーソルを合わせると対処方法がツールチップで表示されます。



鍵スロット

SHALO Keyring は SHALO AUTH に鍵データを 4 セット格納できます。この格納領域を**鍵スロット**と呼び、鍵スロット 1～4 の情報を縦に並べて表示します。

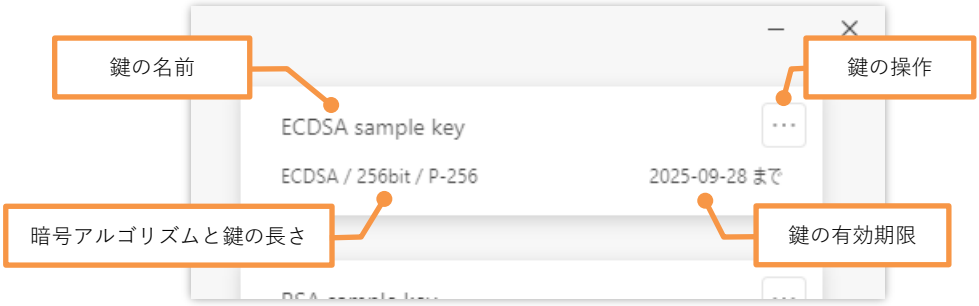
鍵スロットに鍵が設定されていない場合、下図のように表示されます。



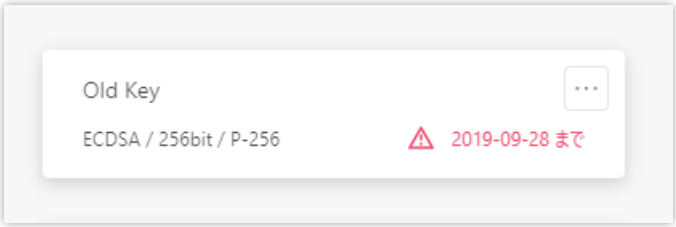
鍵スロットに鍵が設定されている場合、次の 3 つの情報が表示されます。

1. 鍵の名前
2. 鍵の暗号アルゴリズムと鍵の長さ
3. 鍵の有効期限

これらは下図のように配置されます。



鍵の有効期限が過ぎている場合は次のように赤字で表示されます。



鍵の操作は［…］をクリックして表示される下図のメニューから実行できます。

4.3　新しい鍵を生成する

SHALO Keyring は鍵を作成する機能を持ち、X.509 証明書とセットで SHALO AUTH に格納できます。これには SHALO Keyring で[**鍵を作成する**]をクリックします。

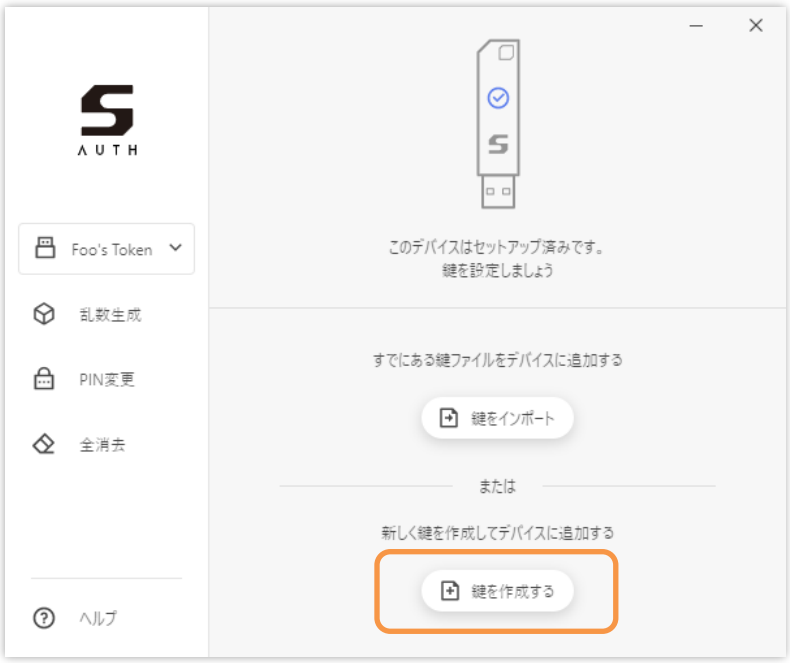


図 29　セットアップ直後の SHALO AUTH に鍵を作成

もし SHALO Keyring が下図のようになっている場合、未設定の欄にある[**鍵を設定する**]をクリックし、[**鍵を新規作成**]をクリックします。

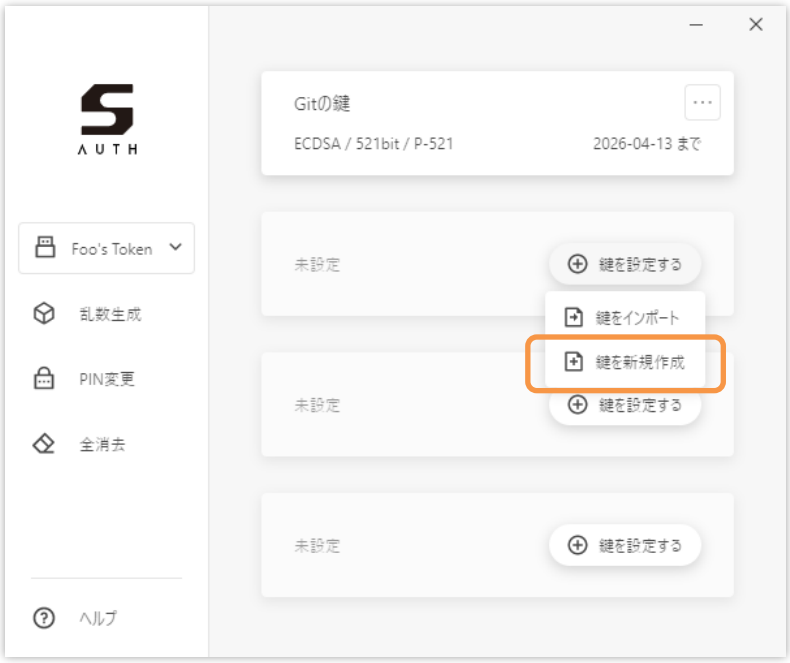


図 30　格納先を指定して鍵を作成

作成する鍵の情報を下図の鍵作成ウィンドウで指定します。指定した後、[**作成する**]をクリックするとユーザーPIN の入力が求められるので、ユーザーPIN を入力します。ユーザーPIN の認証が成功すれば鍵が作成されます。

図 31　鍵作成ウィンドウ

鍵の名前

作成する鍵を区別する名前です。SHALO Keyring で表示されるほか、X.509 証明書でサブジェクトとしても使用されます。

有効期限

鍵の有効期限を指定します。西暦の年-月-日の並びで入力するか、右のアイコンをクリックして表示されるカレンダーで日付を選択します。

この有効期限は公開鍵の X.509 証明書の有効期限として扱われます。
鍵の有効期限は X.509 証明書に対応したアプリケーションでのみ効力を持ちます。

鍵の種類

作成する鍵の暗号アルゴリズムを指定します。通常は ECDSA P-521 を選択してください。ECDSA は P-521 > P-384 > P-256 の順に暗号が弱くなります。

PDF ファイルを暗号化・署名する鍵、また ECDSA を利用できない環境で使用する鍵には RSA を選択してください。

4.4　既存の鍵を取り込む

SHALO Keyring はファイルから鍵を読み取り、SHALO AUTH に取り込むことができます。その際に鍵の X.509 証明書を作成し、鍵と共に SHALO AUTH に格納します。

SHALO Keyring が対応している鍵のデータ形式は次の 3 種類です。

データ形式	拡張子	説明
PEM	.pem	**RSA 鍵の場合:** 「-----BEGIN RSA PRIVATE KEY-----」で始まり、 「-----END RSA PRIVATE KEY-----」で終わるテキストです。 **ECDSA 鍵の場合:** 「-----BEGIN EC PRIVATE KEY-----」で始まり、 「-----END EC PRIVATE KEY-----」で終わるテキストです。
OpenSSH	.pem	「-----BEGIN OPENSSH PRIVATE KEY-----」で始まり、 「-----END OPENSSH PRIVATE KEY-----」で終わるテキストです。
PuTTY	.ppk	PuTTY 付属の puttygen で生成した鍵のファイルです。

上記以外の鍵ファイルを取り込みたい場合、11.3 節を参照して PEM 形式に変換してください。

SHALO AUTH にファイルから鍵を取り込むには、SHALO Keyring で[**鍵をインポート**]をクリックします。

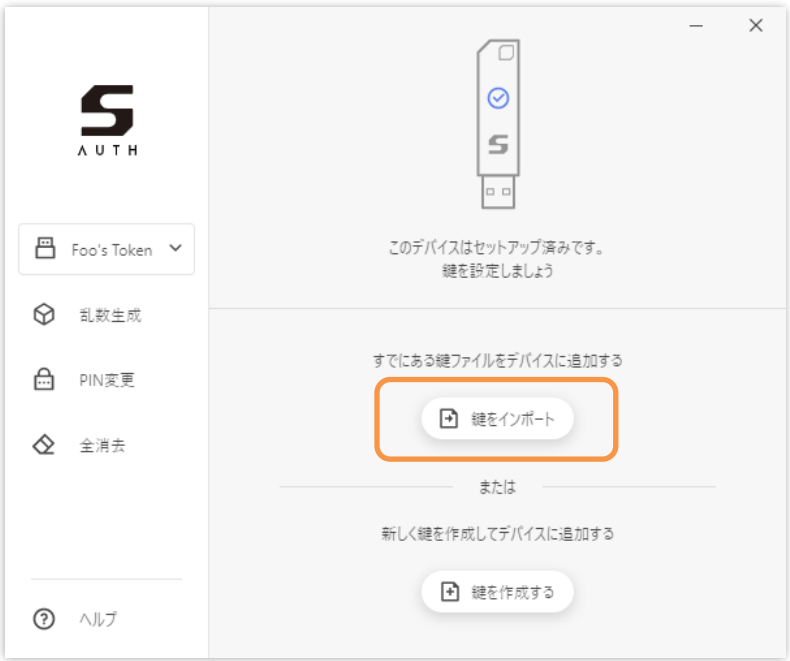


図 32　セットアップ直後の SHALO AUTH に鍵をインポート

もし SHALO Keyring が下図のようになっている場合、未設定の欄にある[**鍵を設定する**]をクリックし、[**鍵をインポート**]をクリックします。

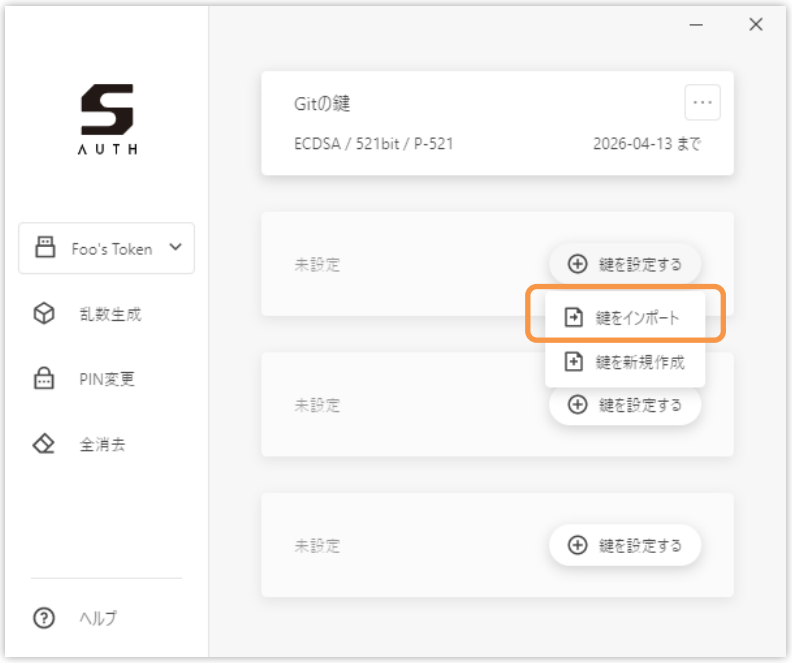


図 33　格納先を指定して鍵をインポート

どちらの方法でも下図のウィンドウが表示されます。枠内に鍵ファイルをドロップするか、[**ファイルを開く**]をクリックして鍵ファイルを選択します。

図 34　鍵ファイルのインポート

指定した鍵ファイルがパスフレーズで暗号化されている場合、パスフレーズの入力が求められます。

SHALO AUTH に鍵をインポートする際に付与する情報を下図のウィンドウで指定します。指定した後、[**インポート**]をクリックしてユーザーPIN を入力すると鍵が SHALO AUTH に取り込まれます。

図 35　インポートする鍵の付加情報

鍵の名前

インポートする鍵を区別する名前です。SHALO Keyring で表示されるほか、X.509 証明書でサブジェクトとしても使用されます。

有効期限

鍵の有効期限を指定します。西暦の年-月-日の並びで入力するか、右のアイコンをクリックして表示されるカレンダーで日付を選択します。

この有効期限は公開鍵の X.509 証明書の有効期限として扱われます。
鍵の有効期限は X.509 証明書に対応したアプリケーションでのみ効力を持ちます。

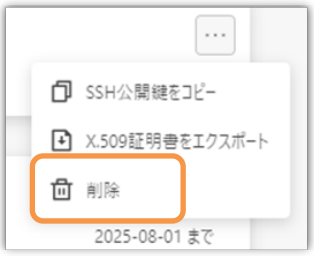
4.5　鍵を削除する

SHALO Keyring で鍵を削除する方法は 2 種類あります。

- 鍵スロットを指定して鍵を削除する
- すべてのデータを消す

鍵スロットを指定して鍵を削除する

空にする鍵スロットの[…]をクリックし、メニューから[**削除**]を選択します。ユーザーPIN の入力が求められるので、ユーザーPIN を入力します。ユーザーPIN の認証が成功すれば鍵が削除されます。



すべてのデータを消す

ウィンドウ左の[**全消去**]をクリックします。この処理では SHALO Keyring 以外のアプリケーションで保存された PKCS #11 オブジェクトも削除されます。

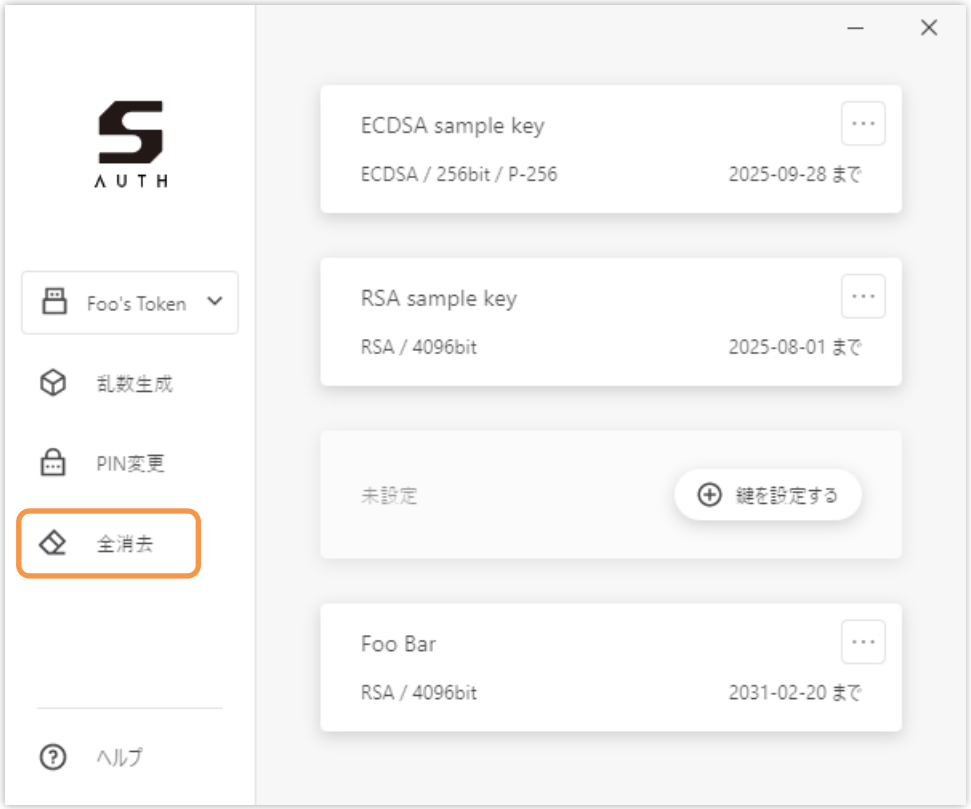


図 36　全消去の実行

ラベル・管理 PIN・ユーザーPIN と FIDO U2F で使用する鍵には影響しません。

以下の警告を読み、問題が無ければ[**すべての鍵を消去する**]をクリックします。ユーザーPIN の入力が求められるので、ユーザーPIN を入力します。ユーザーPIN の認証が成功すれば全消去されます。

図 37　全消去の警告

4.6 公開鍵を取得する

SHALO Keyring は次の 2 つ方法で公開鍵を取得できます。

- SSH で使われるデータ形式の公開鍵
- X.509 証明書

鍵の暗号アルゴリズムが、RSA または P-521・P-384・P-256 のいずれかの場合のみ SSH 公開鍵を取得できます。

いずれの場合も鍵スロットの[…]をクリックして下図のメニューを開きます。

SSH 公開鍵

SSH 公開鍵は以下のいずれかで始まるテキストデータです。

- `ssh-rsa`
- `rsa-sha2-256`
- `rsa-sha2-512`
- `ecdsa-sha2-nistp256`
- `ecdsa-sha2-nistp384`
- `ecdsa-sha2-nistp521`

メニューから[**SSH 公開鍵をコピー**]を選択するとクリップボードに SSH 公開鍵がコピーされます。他のソフトウェアに貼り付けて使用してください。

X.509 証明書

X.509 証明書を PEM 形式でファイルに保存できます。

メニューから［**X.509 証明書をエクスポート**]をクリックし、保存先ファイル名を指定してファイルに保存します。

このファイル形式は、「`-----BEGIN CERTIFICATE-----`」で始まり「`-----END CERTIFICATE-----`」で終わるテキストファイルです。

4.7　ユーザーPIN を変更する

ユーザーPIN を変更するにはウィンドウ左の[**PIN 変更**]をクリックします。

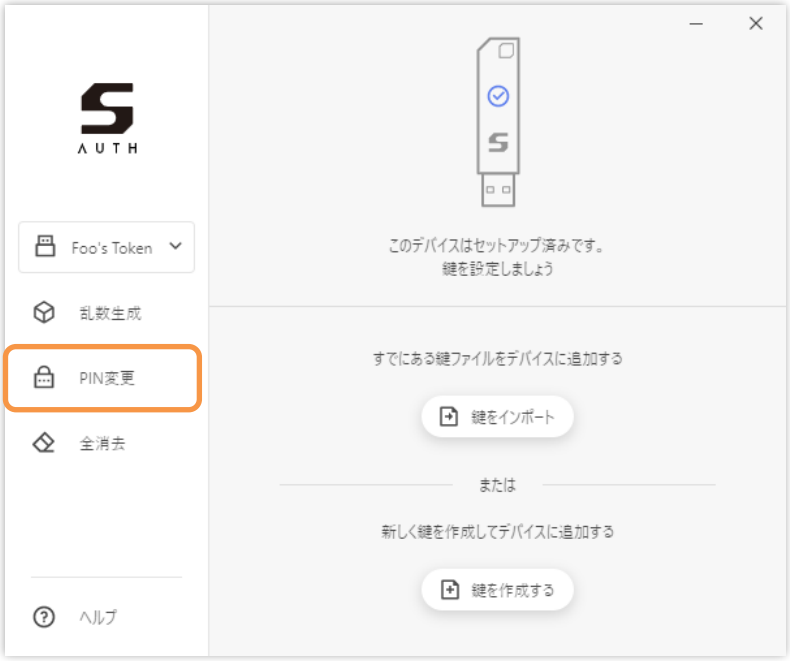


図 38　PIN 変更の実行

下図のウィンドウで現在のユーザーPIN と新しいユーザーPIN を入力した後、[**変更する**]をクリックします。

図 39　ユーザーPIN の変更ウィンドウ

ユーザーPIN がロックされている場合、ユーザーPIN の変更ではロックを解除できません。SHALO Smith を使用してユーザーPIN を再設定（5.4 節）してください。

4.8　パスワードや乱数列を生成する

SHALO Keyring は SHALO AUTH のハードウェア乱数生成器を使ってパスワードや乱数列を生成できます。用途別の乱数生成・出力条件は以下の通りです。

用途	指定可能な条件	生成個数	区切り文字
パスワード	長さ 1～64 文字 大文字の有無 小文字の有無 数字の有無 記号の有無	最大 8 個	改行
整数値	最小値: -32768～+32767 最大値: -32768～+32767	表形式で 32 行 32 列まで	区切りなし カンマ スペース タブ文字
16 進数文字列	ビット長 1～64 bit 先頭の"0x"の有無	表形式で 16 行 16 列まで	区切りなし カンマ スペース タブ文字

パスワードで記号を有効にすると以下の文字を使用します。
`~ ! @ # $ % ^ & * ( ) _ + - = { } [ ] ¥ | : ; " ' < > , . ? /`

実行方法

乱数を生成するにはウィンドウ左の[**乱数生成**]をクリックします。

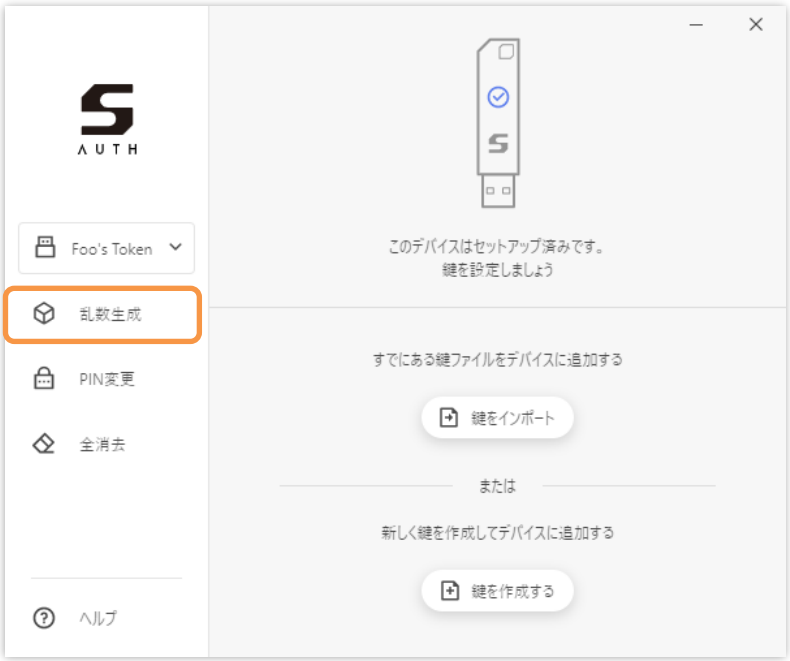


図 40　乱数生成の実行

図 41 のウィンドウで乱数の目的と生成条件を指定します。[**生成する**]をクリックするとウィンドウ内に生成された乱数が表示されます。[**コピー**]をクリックすると表示されている乱数全体をクリップボードにコピーします。

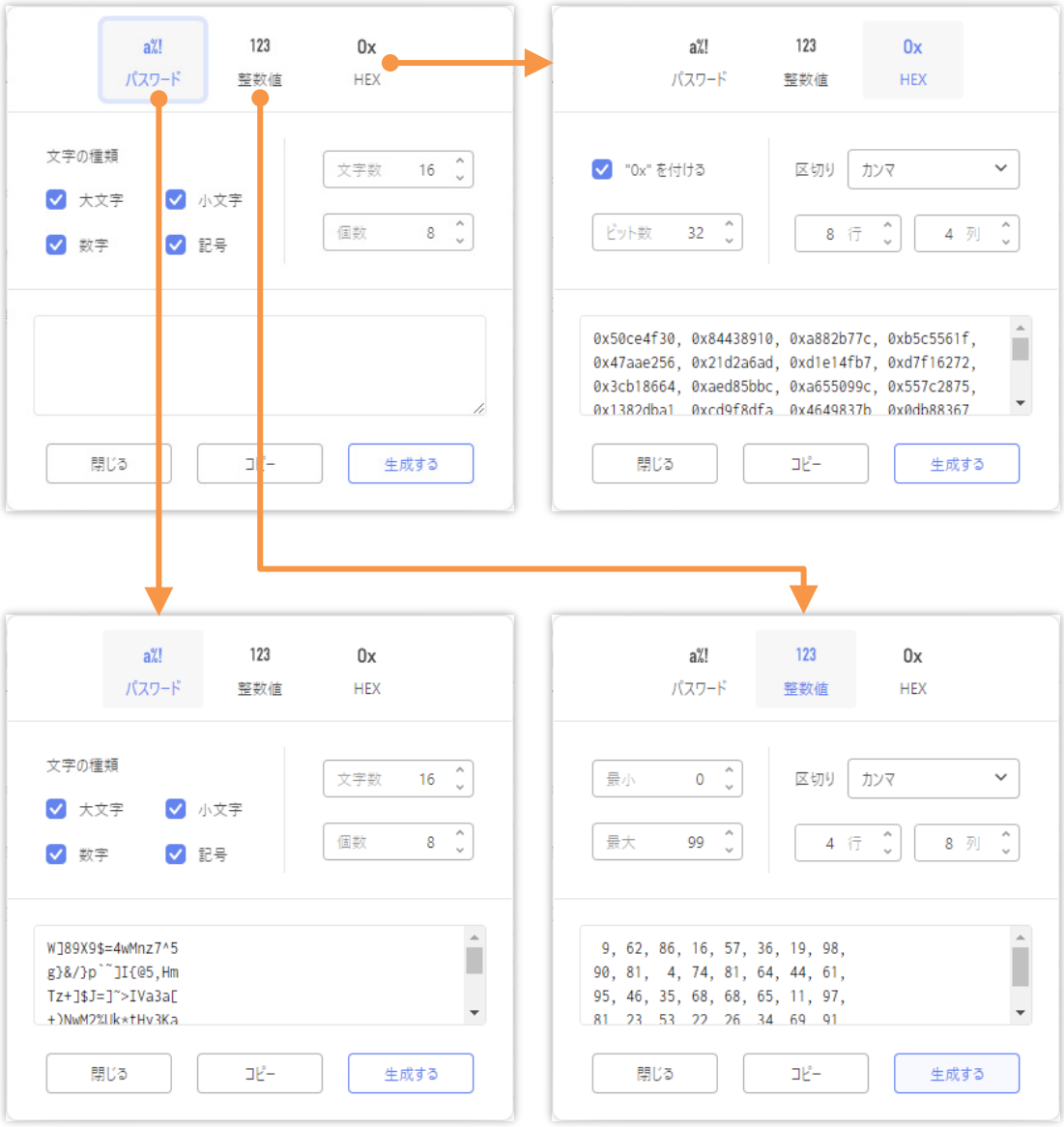


図 41　乱数生成ウィンドウと乱数生成例

4.9 鍵データの CKA_ID 属性

SHALO Keyring は鍵スロットのために以下の 4 個の CKA_ID 属性を予約しています。SHALO Keyring が SHALO AUTH に格納するプライベート鍵・公開鍵・X.509 証明書には、格納先の鍵スロットに対応する CKA_ID 属性が付与されます。

鍵スロット	CKA_ID 属性（16 進数）	CKA_ID 属性（文字列）
鍵スロット 1	41 58 54 4F 4F 4C 4B 45 59 23 31	AXTOOLKEY#1
鍵スロット 2	41 58 54 4F 4F 4C 4B 45 59 23 32	AXTOOLKEY#2
鍵スロット 3	41 58 54 4F 4F 4C 4B 45 59 23 33	AXTOOLKEY#3
鍵スロット 4	41 58 54 4F 4F 4C 4B 45 59 23 34	AXTOOLKEY#4

他の PKCS #11 対応ソフトウェアで独自にデータを管理する場合は予約された CKA_ID 属性を使用しないでください。
予約された CKA_ID 属性を使用すると SHALO Keyring でデータ操作されるほか、SHALO Keyring から正常にデータ操作できなくなる場合があります。

第 5 章

管理ツール SHALO Smith を使う

この章では管理ツール SHALO Smith を説明します。SHALO Smith は SHALO AUTH の発行・管理業務に特化したソフトウェアです。

SHALO AUTH を譲渡・廃棄する際には SHALO Smith を使って SHALO AUTH を購入時の状態に戻します。

この章のトピック

1. SHALO AUTH の状態を確認する
2. SHALO AUTH をセットアップする
3. SHALO AUTH を購入時の状態に戻す
4. ユーザーPIN を再設定する
5. 管理 PIN を変更する

5.1 SHALO AUTH の状態を確認する

SHALO Smith は PC に装着している SHALO AUTH を最大 8 台まで表示・管理操作できます。

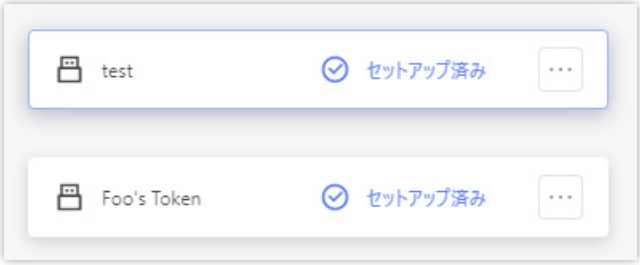
図 42 は SHALO Smith のウィンドウです。この例では 1 台の新しい SHALO AUTH と 3 台のセットアップ済みの SHALO AUTH が PC に装着されています。

図 42　4 台の SHALO AUTH を表示する SHALO Smith

縦に並んだ白い長方形 1 つ 1 つが 1 台の SHALO AUTH を示します。下図のように長方形内部の左側にデバイスラベルが表示され、右側にデバイスの状態が表示されます。

SHALO AUTH の識別

マウスでクリックされた長方形は下図のように薄く色が付き、それに対応する SHALO AUTH のライトが点滅します。



SHALO AUTH の状態

SHALO AUTH の状態は以下の通りです。

状態表示	説明
[セットアップ]ボタン	セットアップされていません。ボタンを押すとセットアップ（5.2 節）できます。
セットアップ済み	セットアップ済みで正常な状態です。
ユーザーPIN ロック	セットアップ済みですが、ユーザーPIN がロックされています。 復旧するにはユーザーPIN の再設定（5.4 節）が必要です。
管理 PIN ロック	セットアップ済みですが、管理 PIN がロックされています。 購入時の状態に戻す（5.3 節）以外に復旧方法はありません。

SHALO AUTH へのアクション

新しい SHALO AUTH は[**セットアップ**]をクリックするとセットアップできます。

右端の[…]をクリックすると、右図のようにメニューが表示され、SHALO AUTH の管理に関連したアクションを実行できます。

新しい SHALO AUTH ではファクトリーリセットのみ実行できます。
これは U2F で使用するすべての FIDO 認証鍵を削除します。

5.2 SHALO AUTH をセットアップする

セットアップでは汎用セキュリティキー機能向けのデータ領域を初期化し、次の管理情報を設定します。SHALO Keyring でもセットアップできます。

デバイスラベル　複数の SHALO AUTH を区別するために使われる個体名です。
ユーザーPIN　利用時のパスワードです。保護された暗号鍵の使用を許可します。
管理 PIN　管理用のパスワードです。ユーザーPIN を再設定する場合や、SHALO AUTH を購入時の状態に戻す際に使います。

このセットアップは U2F セキュリティキーの機能に影響を与えません。
セットアップ以前に U2F セキュリティキーとして SHALO AUTH を登録したウェブサービスはセットアップした SHALO AUTH で引き続き利用できます。

セットアップ済みの SHALO AUTH を再度セットアップするには購入時の状態に戻す必要があります。購入時の状態に戻すと U2F セキュリティキーの情報も削除されます。

実行手順

[**セットアップ**]をクリックすると SHALO AUTH のセットアップを開始します。セットアップは、デバイスラベル・ユーザーPIN・管理 PIN の順に設定します。

デバイスラベルの設定

デバイスラベルには、英字・数字・記号、日本語やその他言語の文字を使用できます。デバイスラベルの最大文字数は文字の種類に依存します。長すぎる場合は警告が表示されます。

図 43　デバイスラベルの設定

ユーザーPIN の設定

ユーザーPIN には英字・数字・記号を使用できます。4 文字以上 256 文字以下の長さでユーザーPIN を指定してください。確認のためにユーザーPIN を 2 回入力します。

図 44　ユーザーPIN の設定

管理 PIN の設定

管理 PIN には英字・数字・記号を使用できます。4 文字以上 256 文字以下の長さで管理 PIN を指定してください。確認のために管理 PIN を 2 回入力します。

図 45　管理 PIN の設定

5.3 SHALO AUTH を購入時の状態に戻す

SHALO AUTH を購入時の状態に戻すと以下の**すべての情報を削除します。**

- 管理 PIN
- ユーザーPIN
- デバイスラベル
- すべての PKCS #11 データ
- すべての FIDO 認証鍵

購入時の状態に戻すと U2F で使用するすべての FIDO 認証鍵も削除されます。そのため、以前登録したウェブサービスであっても未登録デバイスとして扱われます。

SHALO AUTH を譲渡・廃棄する際には SHALO AUTH を購入時の状態に戻すことを**強く推奨**します。二要素認証として登録済みのサービスに対して譲渡した SHALO AUTH による不正認証を防ぐことができます。

実行手順

1. 対象デバイスの右にある[…]をクリックします。
2. メニューで[**ファクトリーリセット**]を選択します。
3. 図 46 で[**ファクトリーリセットする**]をクリックします。
4. 管理 PIN を入力し、[**認証する**]をクリックします。

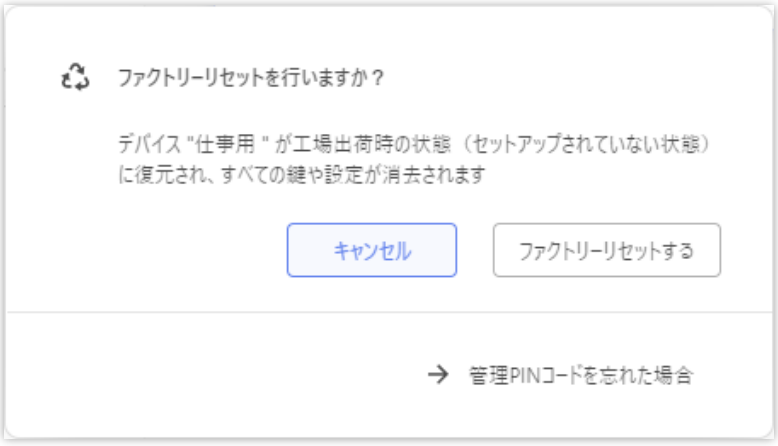


図 46　ファクトリーリセットウィンドウ

管理 PIN がわからない場合の方法

以下の手順に従って操作すると管理 PIN なしでファクトリーリセットできます。

1. 図 46 で[**管理 PIN コードを忘れた場合**]をクリックします。
2. 図 47 のウィンドウが表示された後、SHALO AUTH のライトが素早く点滅するまで SHALO AUTH 側面のボタンを押し続けます。これは約 30 秒かかります。
3. ライト点滅中に[**リセット**]をクリックします。

図 47　管理 PIN を入力しないファクトリーリセット

管理 PIN がロックされている場合でも、この方法で SHALO AUTH を購入時の状態に戻すことができます。

ライトが点滅していない状態で[**リセット**]をクリックすると、管理 PIN の認証失敗とみなされて PIN 認証試行残が 1 つ減ります。これを繰り返すと管理 PIN がロックされます。

5.4　ユーザーPIN を再設定する

ユーザーPIN の再設定を行うと以下を行います。

- ユーザーPIN を新しいものに変更します。
- ユーザーPIN のロックを解除します。
- ロックされるまでの認証試行残を 5 回に戻します。

ユーザーPIN の再設定では現在のユーザーPIN を入力するは必要ありませんが、現在の管理 PIN を入力する必要があります。

実行手順

1. 対象デバイスの右にある[…]をクリックします。
2. メニューで[**ユーザーPIN を再設定する**]を選択します。
3. 図 48 のウィンドウでそれぞれの PIN を入力した後、[**再設定する**]をクリックします。

図 48　ユーザーPIN の再設定

5.5　管理 PIN を変更する

管理 PIN を変更するには以下の手順を行います。

1. 対象デバイスの右にある[…]をクリックします。
2. メニューで[**管理 PIN を変更する**]を選択します。
3. 現在の管理 PIN と新しい管理 PIN を入力した後、[**変更する**]をクリックします。

図 49　管理 PIN の変更

第 6 章

ウェブサービスで U2F を使う

この章ではウェブサービスで 2 段階認証に SHALO AUTH の U2F を使用する方法を説明します。

SHALO AUTH の U2F を 2 段階認証に使う場合、紛失・破損に備えて他のログイン方法を追加することを**強く推奨**します。そのため、この章はあらかじめ各ウェブサービスのアカウントで 2 段階認証プロセスを有効にしていることを前提に説明します。

U2F の概要については 2.2 節を参照してください。

この章のトピック

1. Google の U2F 設定
2. Facebook の U2F 設定
3. GitHub の U2F 設定

6.1 Google の U2F 設定

6.1.1 SHALO AUTH を登録する

2 段階認証プロセスがすでにオンになっている場合、以下の手順で Google アカウントに SHALO AUTH を追加できます。SHALO AUTH は PC から取り外しておきます。

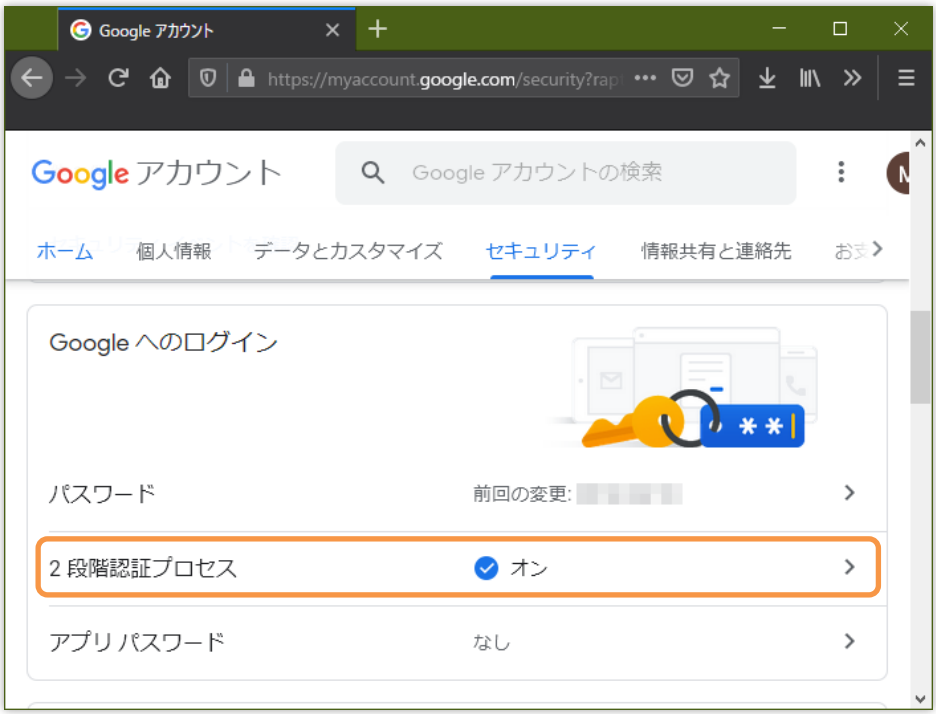
1. https://myaccount.google.com/ をウェブブラウザで開き、ログインします。
2. [**セキュリティ**]を選択します。
3. [**Google へのログイン**]の[**2 段階認証プロセス**]をクリックします。
4. 2 段階認証プロセスのページで[**セキュリティキーを追加**]をクリックします。
5. [**次へ**]をクリックします。
6. SHALO AUTH を PC に接続し、SHALO AUTH のライトが点滅したらボタンを押します。
7. セキュリティキーの名前を入力して、[**完了**]をクリックします。
8. ログアウトして、SHALO AUTH でログインできることを確認します。

この手順をスクリーンショットとともに説明します。

本節の説明はマニュアル作成時の情報に基づいています。
ウェブサイトの構成が説明と異なる場合があることに注意してください。

手順 1～3

下図で示すように、[**2 段階認証プロセス**]をクリックします。



手順 4

下図のようにページを下にスクロールして、[**セキュリティキー**]にある[**セキュリティキーを追加**]をクリックします。

手順 5

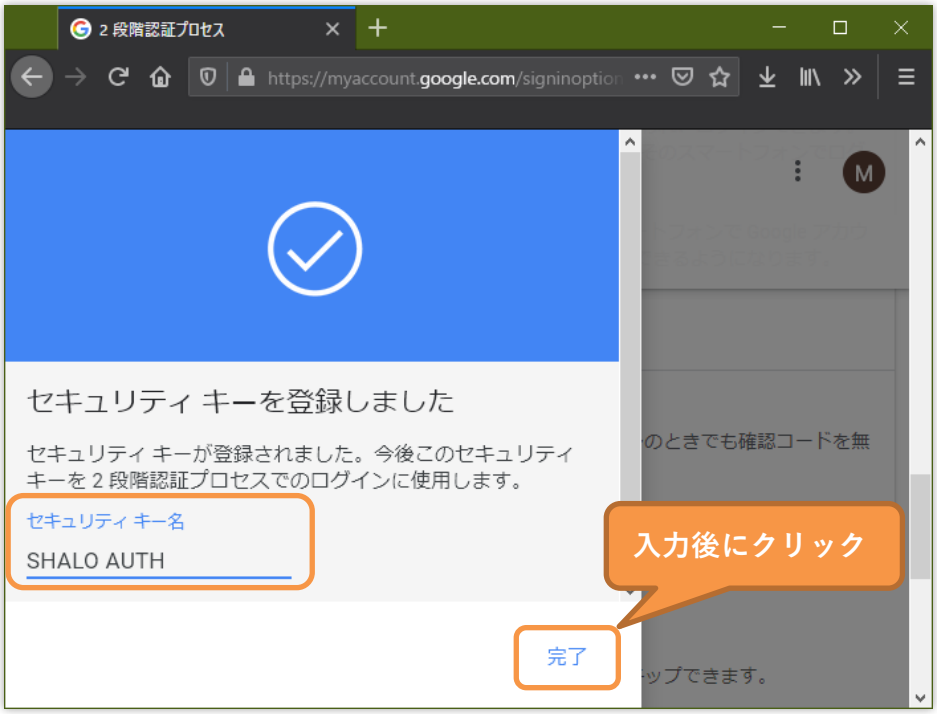
下図のように表示されます。[**次へ**]をクリックします。

手順 6

スクリーンに従って SHALO AUTH を PC に接続します。SHALO AUTH のライトが点滅したら SHALO AUTH のボタンを押します。

手順 7

セキュリティキーに名前を付けます。これは Google アカウントで登録したセキュリティキーを区別するためのものです。SHALO AUTH 本体には影響しません。名前を入力して、最後に[**完了**]をクリックします。



手順 8

正しくログインできることを確かめるためにログアウトしてからログインします。 ログイン情報を入力後、SHALO AUTH のライトが点滅してから SHALO AUTH のボタンを押します。

6.1.2　SHALO AUTH の登録を解除する

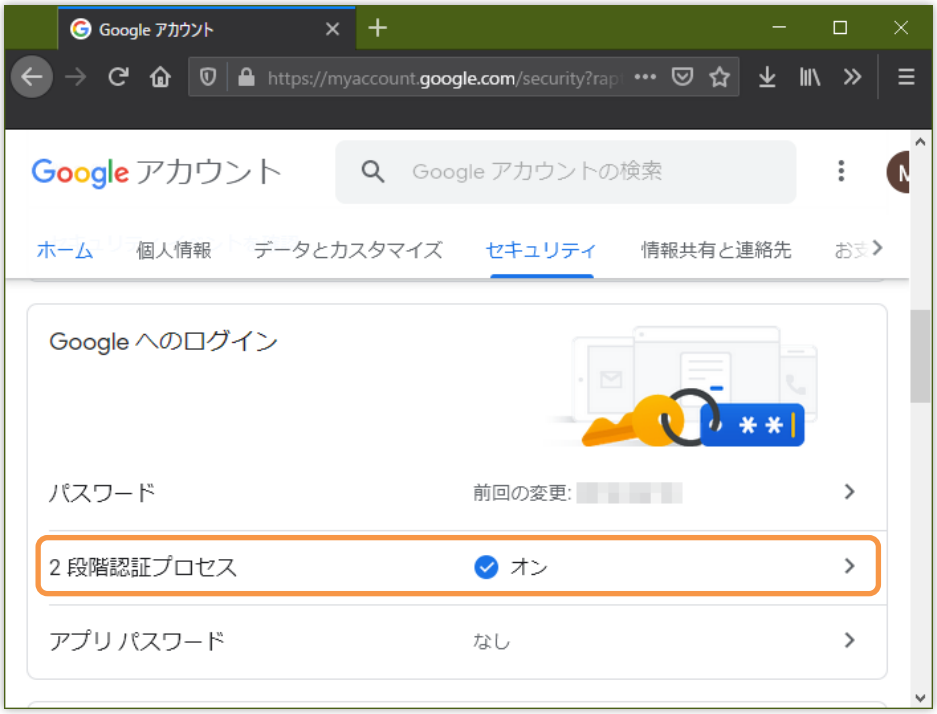
以下の手順で Google アカウントから SHALO AUTH の登録を削除できます。

1. https://myaccount.google.com/ をウェブブラウザで開き、ログインします。
2. [**セキュリティ**]を選択します。
3. [**Google へのログイン**]の[**2 段階認証プロセス**]をクリックします。
4. 登録削除するセキュリティ キーの横にある編集アイコンをクリックします。
5. [**このキーを取り消す**]をクリックします。

この手順をスクリーンショットとともに説明します。

手順 1～3

下図で示すように、[**2 段階認証プロセス**]をクリックします。

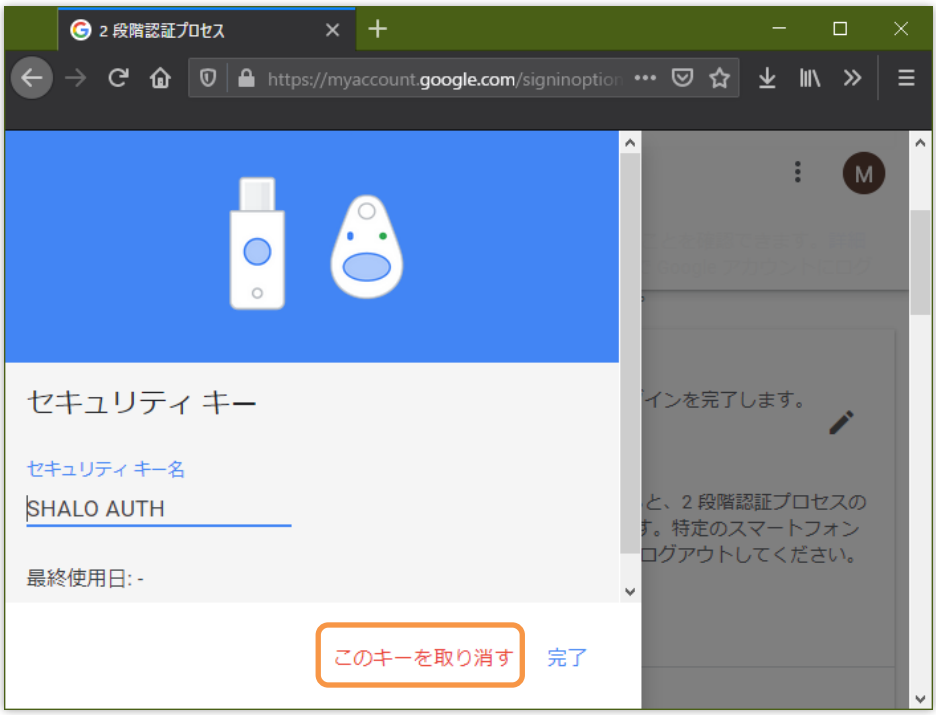


手順 4

下図で示すように、削除対象のセキュリティキーの横にある編集アイコンをクリックします。

手順 5

下図で示すように、[**このキーを取り消す**]をクリックします。



6.2 Facebook の U2F 設定

2 段階認証の設定は、Facebook アカウントの設定で[**セキュリティとログオン**] > [**二段階認証**]のページで行います。

6.2.1 SHALO AUTH を登録する

以下の手順で Facebook アカウントの 2 段階認証に SHALO AUTH を登録できます。SHALO AUTH は PC から取り外しておきます。

1. ウェブ版の Facebook にログインします。
2. [**アカウント**]アイコンをクリックし、[**設定とプライバシー**]-[**設定**]をクリックします。
3. 設定ページで[**セキュリティとログイン**]をクリックします。
4. [**二段階認証を使用**]の[**編集**]をクリックします。
5. [**バックアップ方法の追加**]でセキュリティキーの[**設定**]をクリックします。
6. SHALO AUTH を PC に接続し、SHALO AUTH のライトが点滅してから SHALO AUTH のボタンを押します。
7. セキュリティキーの名前を入力して[**Save**]をクリックします。
8. [**OK**]をクリックします。
9. ログアウトして、SHALO AUTH でログインできることを確認します。

この手順をスクリーンショットとともに説明します。

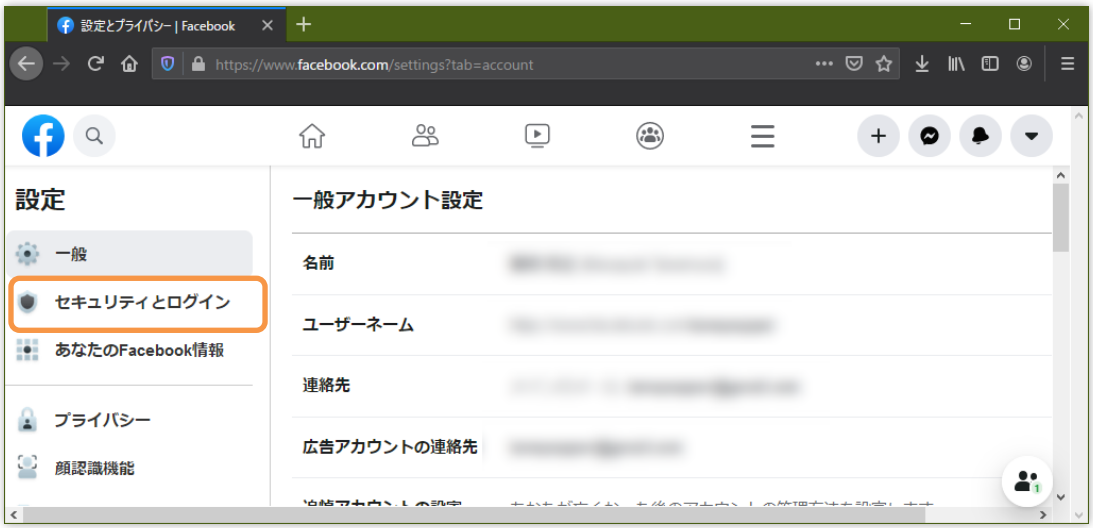
本節の説明はマニュアル作成時の情報に基づいています。
ウェブサイトの構成が説明と異なる場合があることに注意してください。

手順 1～2

下図のように、Facebook のアイコンとメニューを順番にクリックしていきます。

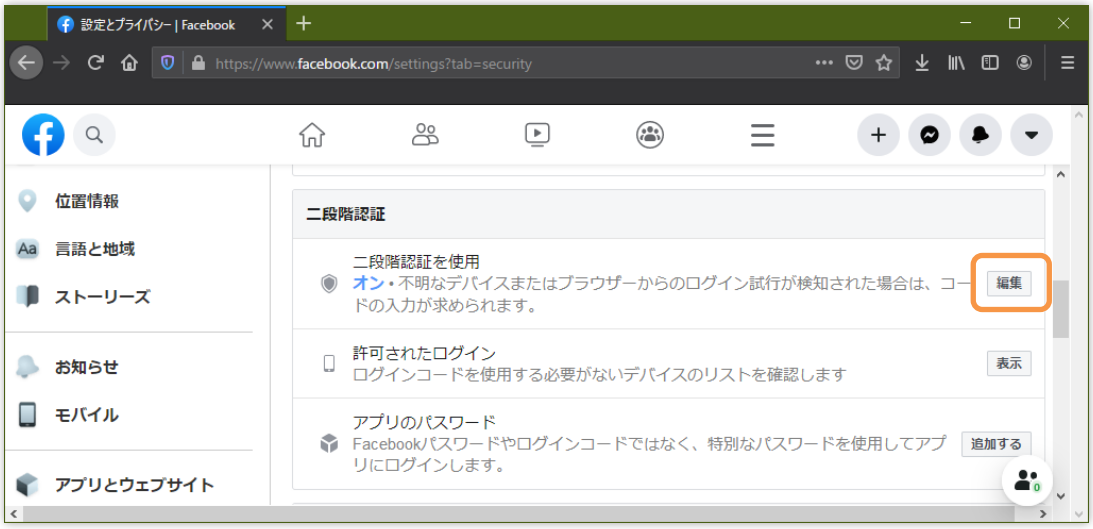
手順 3

下図で示すように表示されます。[**セキュリティとログオン**]をクリックします。



手順 4

ページを下にスクロールし、[**二段階認証を使用**]の横の[**設定**]をクリックします。



手順 5

ページを下にスクロールし、下図で示すように[**バックアップ方法を追加**]で[**セキュリティキー**]の[**設定**]をクリックします。

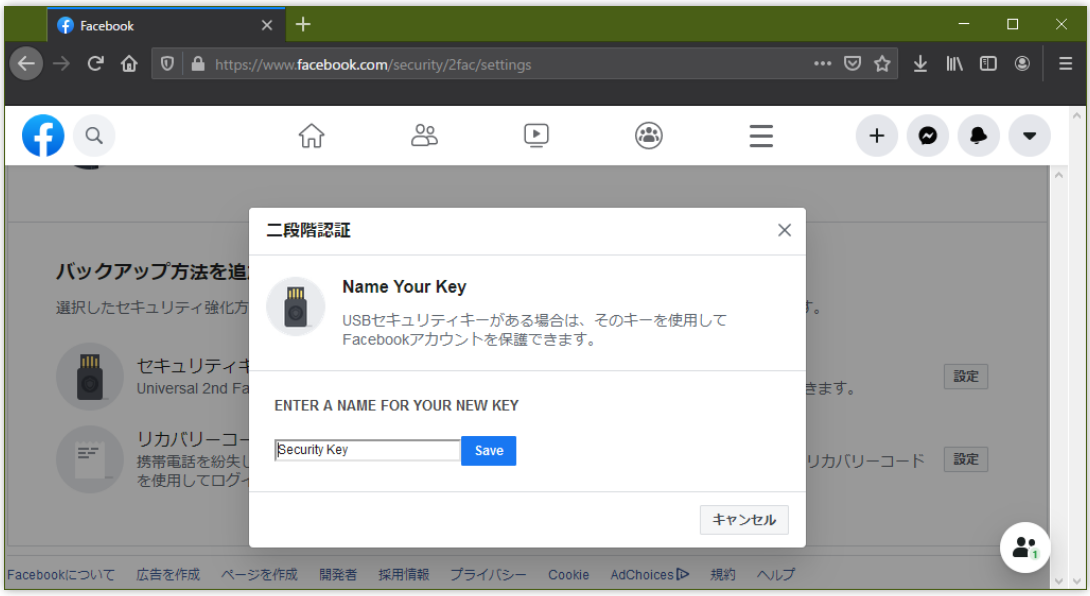
手順 6

下図のように表示されるので、SHALO AUTH を PC に接続します。そして SHALO AUTH のライトが点滅してから SHALO AUTH のボタンを押します。

手順 7

SHALO AUTH が正常に登録されると下図のように表示されます。ここでセキュリティキーに名前を付けます。この名前は Facebook アカウントで登録したセキュリティキーを区別するためのものです。SHALO AUTH 本体には影響しません。

セキュリティキーの名前を入力して[**Save**]をクリックします。



手順 8

すべてが完了すると下図のように表示されます。[**OK**]ボタンを押してウィンドウを閉じます。

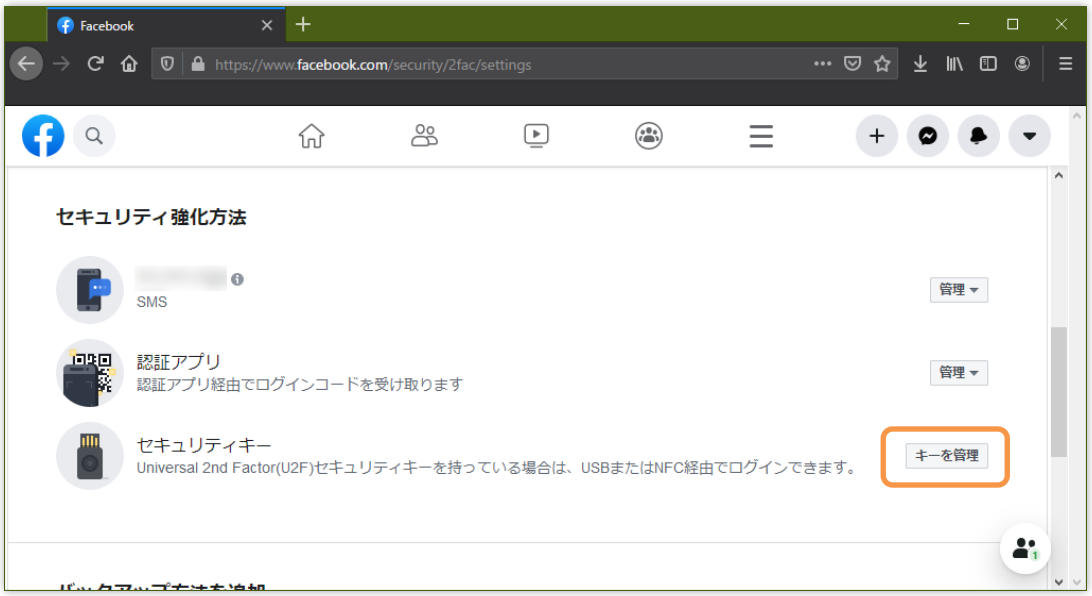
手順 8

正しくログインできることを確かめるためにログアウトしてからログインします。下図のように表示され、SHALO AUTH のライトが点滅していることを確認してから SHALO AUTH のボタンを押します。

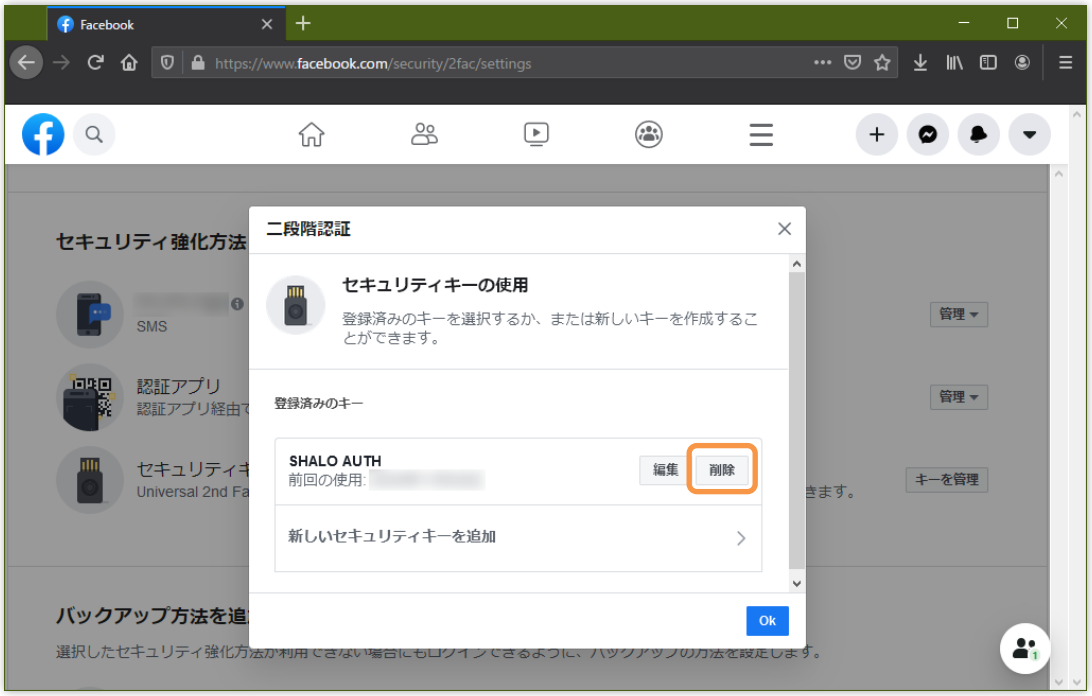
6.2.2　SHALO AUTH の登録を解除する

前節の手順 1～4 で開いた[**セキュリティとログオン**] > [**二段階認証**] のページから、SHALO AUTH の登録を解除できます。

まず[**セキュリティキー**]の横の[**キーを管理**]をクリックします。



登録済みのセキュリティキーが表示されます。削除するセキュリティキーの横の[**削除**]をクリックすると、登録の解除が完了します。



6.3 GitHub の U2F 設定

6.3.1 SHALO AUTH を登録する

あらかじめ TOTP モバイル App または SMS で二要素認証を有効にしておきます。このとき取得されるリカバリーコードを安全な場所に保存してください。

以下の手順で GitHub の二要素認証に SHALO AUTH を登録できます。SHALO AUTH は PC から取り外しておきます。

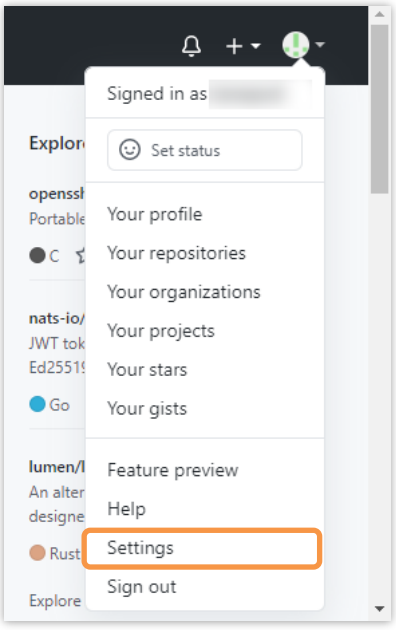
1. https://www.github.com をウェブブラウザで開いてログインします。
2. 右上のプロフィール画像をクリックし、続いて[**Settings**]をクリックします。
3. 左のサイドバーで[**Account Security**]をクリックします。
4. [**Security Keys**]の横にある[**Add**]をクリックします。
5. [**Security Keys**]で、[**Register new security key**]をクリックします。
6. セキュリティのニックネームを入力して、[**Add**]をクリックします。
7. SHALO AUTH を PC に接続し、SHALO AUTH のライトが点滅してから SHALO AUTH のボタンを押します。
8. サインアウトして、SHALO AUTH でサインインできることを確認します。

この手順をスクリーンショットとともに説明します。

本節の説明はマニュアル作成時の情報に基づいています。
ウェブサイトの構成が説明と異なる場合があることに注意してください。

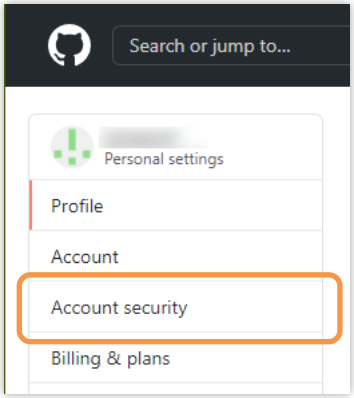
手順 1～2

GitHub にログインし、右上のプロフィール画像をクリックし、[**Settings**]をクリックします。



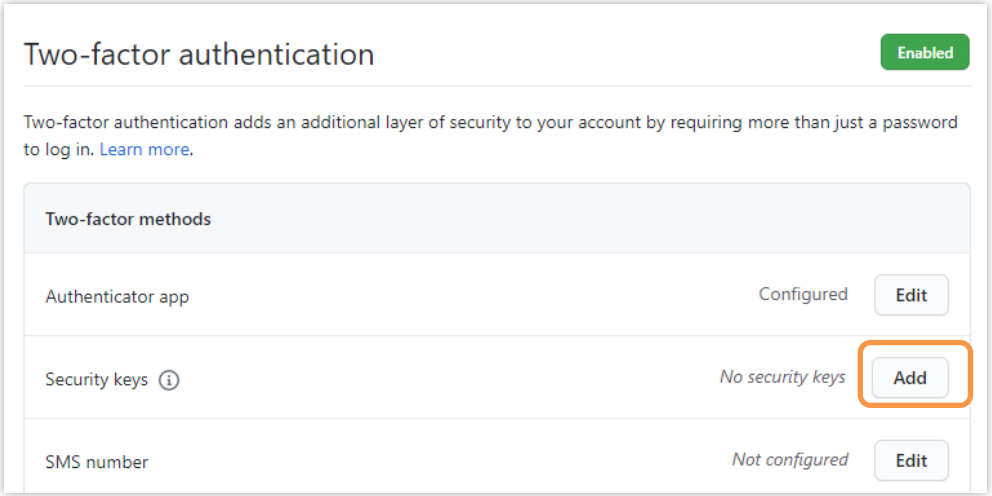
手順 3

左のサイドバーで[**Account Security**]をクリックします。



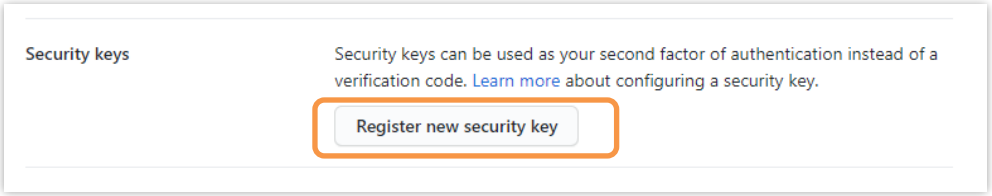
手順 4

[**Security Keys**]の横にある[**Add**]をクリックします。



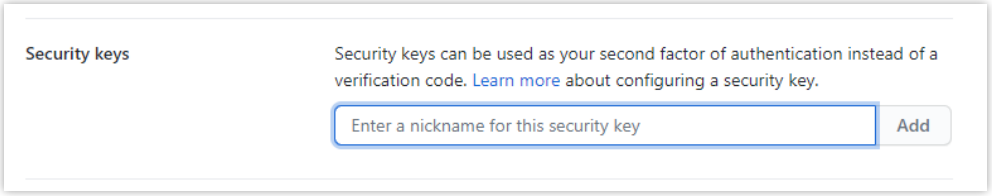
手順 5

[**Security Keys**]で、[**Register new security key**]をクリックします。



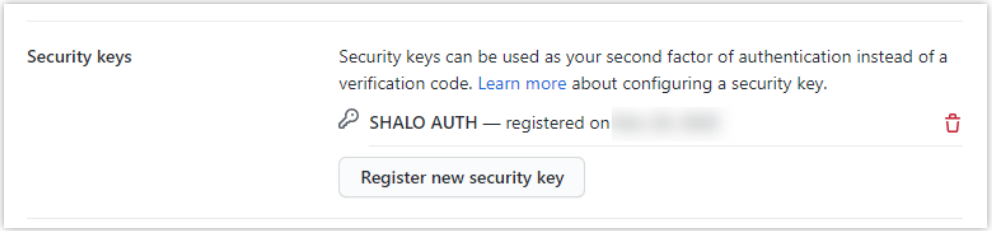
手順 6

セキュリティのニックネームを入力して、[**Add**]をクリックします。



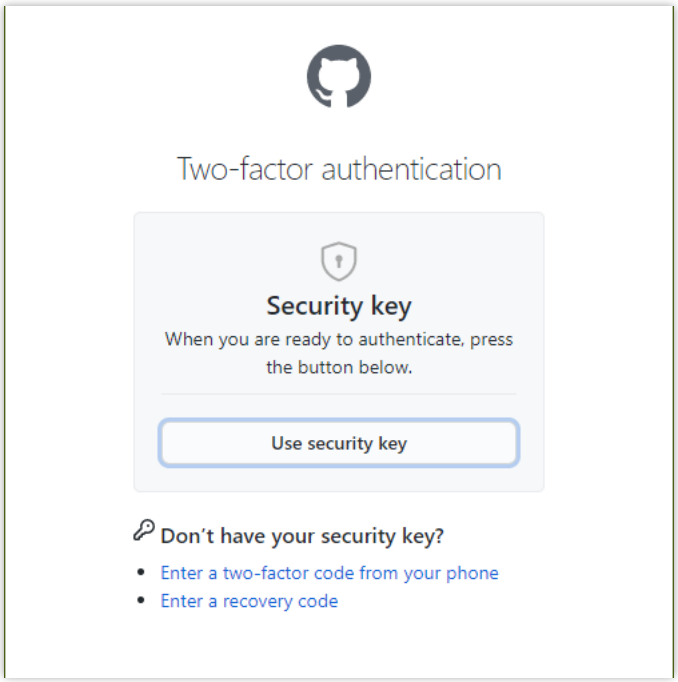
手順 7

SHALO AUTH を PC に接続し、SHALO AUTH のライトが点滅してから SHALO AUTH のボタンを押します。正しく登録されると以下のように自分の指定したニックネームが表示されます。



手順 8

正しくサインインできることを確かめるためにサインアウトしてからサインインします。 [**Use security key**]をクリックし、SHALO AUTH のライトが点滅してから SHALO AUTH のボタンを押します。



6.3.2 SHALO AUTH の登録を解除する

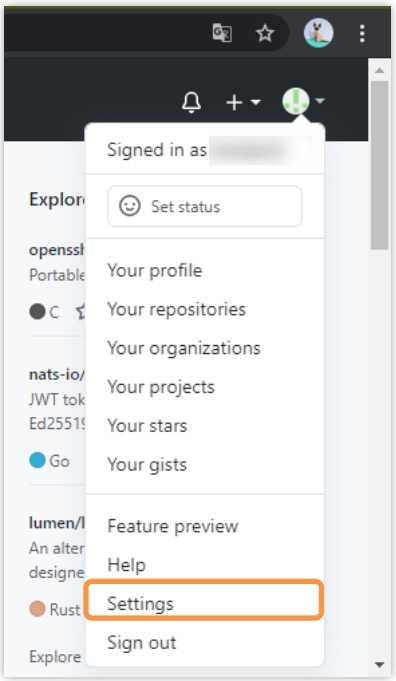
以下の手順で GitHub から SHALO AUTH の登録を削除できます。

1. https://www.github.com をウェブブラウザで開いてログインします。
2. 右上のプロフィール画像をクリックし、続いて[**Settings**]をクリックします。
3. 左のサイドバーで[**Account Security**]をクリックします。
4. [**Security Keys**]の横にある[**Edit**]をクリックします。
5. [**Security Keys**]で削除するセキュリティキーのニックネームの横のアイコンをクリックします。

この手順をスクリーンショットとともに説明します。

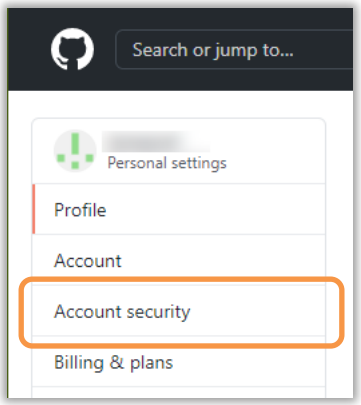
手順 1～2

GitHub にログインし、右上のプロフィール画像をクリックし、[**Settings**]をクリックします。



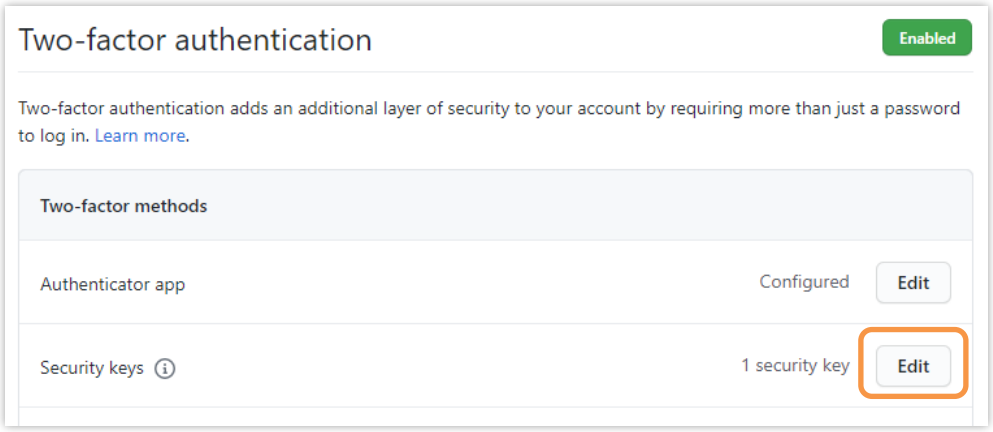
手順 3

左のサイドバーで[**Account Security**]をクリックします。



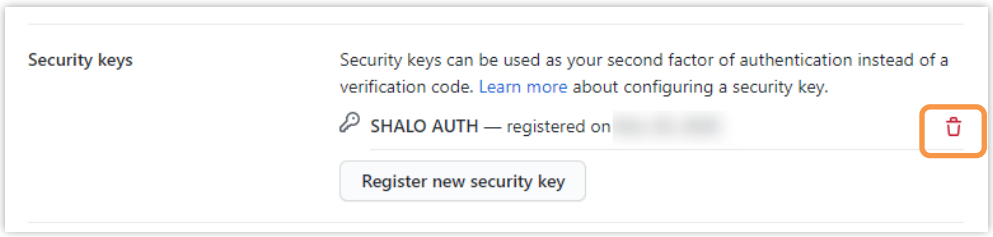
手順 4

[**Security Keys**]の横にある[**Edit**]をクリックします。



手順 5

[**Security Keys**]で削除するセキュリティキーのニックネームの横のアイコンをクリックします。



第 7 章
PDF ファイルで使う

Windows/macOS 向けの Adobe® Acrobat®と Adobe® Acrobat® Reader®は PKCS #11 モジュール経由で SHALO AUTH を利用できます。そして SHALO AUTH に格納されている鍵を Acrobat®のデジタル ID として利用できます。

この章では PDF ファイルのセキュリティに SHALO AUTH を活用する方法を説明します。

この章のトピック

1. PDF ファイルのセキュリティを理解する
2. Acrobat®の設定
3. SHALO AUTH からデジタル ID を取り込む
4. デジタル ID の証明書を他の人に渡す
5. デジタル ID で PDF ファイルを暗号化する
6. 暗号化された PDF ファイルを閲覧する
7. デジタル ID で PDF ファイルに電子署名を付ける

7.1 PDF ファイルのセキュリティを理解する

Windows/macOS 向けの Adobe® Acrobat®と Adobe® Acrobat® Reader®（以降、Acrobat®）は PKCS #11 モジュール経由で SHALO AUTH を利用できます。

PDF ファイルのセキュリティと SHALO AUTH を組み合わせると次のように運用できます。

- PDF ファイルを暗号化して特定の SHALO AUTH の所有者だけが閲覧できるようにする
- PDF ファイルに SHALO AUTH で電子署名を付与する

これらは Acrobat®で**デジタル ID** と呼んでいる本人識別情報を使用します。この節ではデジタル ID を説明してから PDF の暗号化と電子署名を説明します。そして次節以降で Acrobat®から SHALO AUTH を利用する方法を説明します。

デジタル ID

デジタル ID は本人を識別するための情報で、次の 2 つから構成されます。

- 公開鍵暗号のプライベート鍵
- 証明書（公開鍵暗号の公開鍵と本人情報）

これらは 2.3.3 節で説明した SHALO AUTH が管理するデータと同じです。Acrobat®は PKCS #11 API に対応しているので、SHALO AUTH に格納されている鍵をデジタル ID として使用できます。

デジタル ID として RSA 鍵を使用してください。
Acrobat®は PKCS #11 経由で ECDSA 鍵を使用できません。

PDF ファイルの暗号化

PDF ファイルを暗号化して不特定多数による閲覧を防ぐことができます。PDF ファイルの暗号化は次の方法があります。

- パスワードによる保護
- 証明書による保護

パスワードによる保護は、**パスワードを知る人だけが閲覧できる**ように暗号化する方法です。作成者と閲覧者は共通のパスワードを使います。

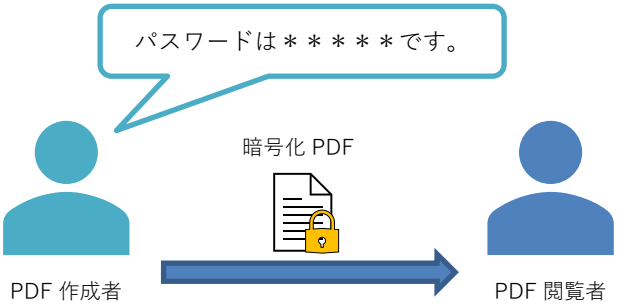


図 50　パスワードで保護された PDF を提供する

一方、証明書による保護は**証明書の対象者だけが閲覧できる**ように暗号化する方法です。これは閲覧者から提供されたデジタル ID の証明書に含まれる公開鍵で PDF ファイルを暗号化します。**閲覧にはデジタル ID が必要です。**

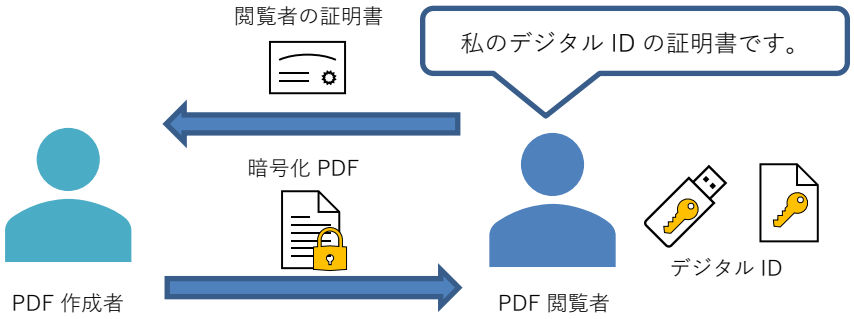


図 51　閲覧者のデジタル ID に合わせて暗号化した PDF を提供する

閲覧用デジタル ID の複製を防ぐには、作成者が閲覧用デジタル ID を用意します。そしてデジタル ID が格納された SHALO AUTH を閲覧者に提供します。

この運用では PDF ファイル毎にデジタル ID を作成する必要はありません。作成者がデジタル ID の証明書を管理していれば、それを使用して他の PDF ファイルも同じ閲覧者向けに暗号化できます。

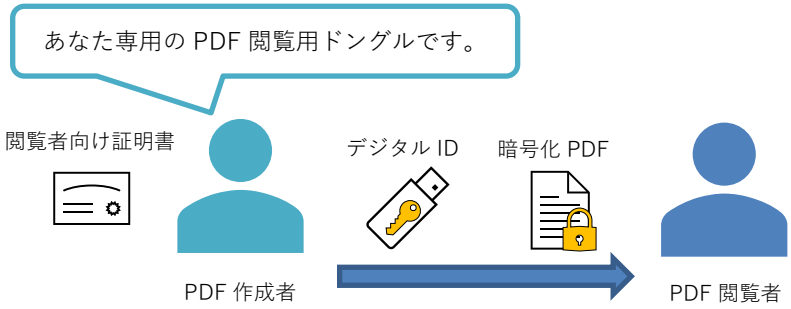


図 52　作成者が用意した閲覧用ドングルと共に暗号化した PDF の提供する

電子署名

PDF ファイルにデジタル ID で電子署名を付与すると、閲覧者は電子署名から次のことがわかります。これには作成者のデジタル ID の証明書が必要です。

- 本当に作成者本人が作成したか
- 内容が改ざんされていないか

図 53　電子署名で PDF ファイルを検証する

7.2 Acrobat®の設定

7.2.1 Acrobat®に PKCS #11 モジュールを登録する

Acrobat®で SHALO AUTH を使用するには、Acrobat®に SHALO AUTH の PKCS #11 モジュールを登録します。

Acrobat®での SHALO AUTH 利用上の注意

Acrobat®を実行中に SHALO AUTH を PC に装着した場合でも、SHALO AUTH を利用できます。

Acrobat®で SHALO AUTH のユーザーPIN を入力すると、明示的にログアウトを行うか Acrobat®を終了するまでユーザーPIN の入力は不要です。

Windows 版 Acrobat®で SHALO AUTH を利用するには、保護モードを無効にする必要があります。

PC に装着されている SHALO AUTH の汎用セキュリティキー機能は Acrobat®が任意のタイミングで使用開始し、他のソフトウェアから使用できなくなります。他のソフトウェアから SHALO AUTH を使用できない場合は Acrobat®を終了します。

Acrobat®起動中に SHALO AUTH を取り外さないでください。SHALO AUTH を装着し直した場合でも SHALO AUTH を使った処理ができなくなります。
SHALO AUTH を取り外して Acrobat®が「PKCS #11 エラー」を表示した場合には Acrobat®を終了してください。

登録手順

以下の手順で PKCS #11 モジュールを Acrobat®に登録します。

1. Windows: メニューから[**編集**] > [**環境設定**]をクリックします。
   macOS: メニューから[**Acrobat Reader**] > [**環境設定**]または[**Acrobat Pro DC**] > [**環境設定**]をクリックします。
2. Windows のみ: [**セキュリティ（拡張）**]をクリックし、[**サンドボックスによる保護**]領域で[**起動時に保護モードを有効にする**]のチェックを外し、Acrobat®を再起動します。
3. [**署名**]をクリックし、[**ID と信頼済み証明書**]領域の[**詳細**]をクリックします。
4. [**PKCS#11 モジュールおよびトークン**]を選択し、[**モジュールを追加**]をクリックします。
5. SHALO AUTH の PKCS #11 モジュールファイルを選択します。

この手順をスクリーンショットとともに説明します。

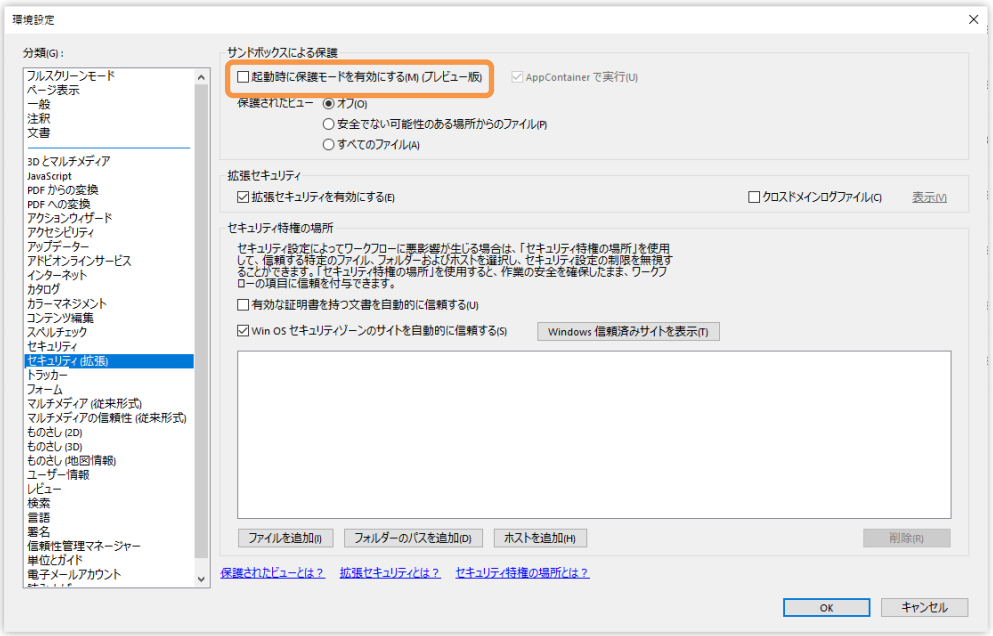
手順 1

Windows の場合:　メニューから[**編集**] > [**環境設定**]をクリックします。

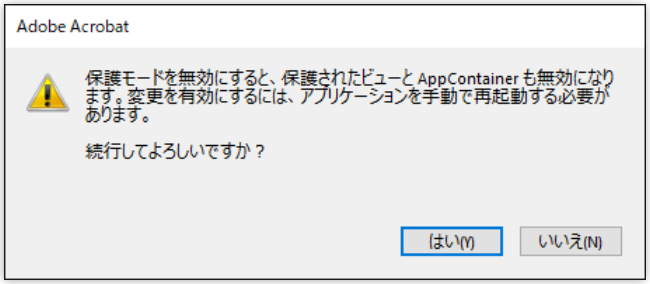
macOS の場合:　メニューから[**Acrobat Reader**] > [**環境設定**]または[**Acrobat Pro DC**] > [**環境設定**]をクリックします。
最初のメニュー項目はアプリケーションの品種によって細部が異なります。

手順 2（Windows のみ）

分類で[**セキュリティ（拡張）**]をクリックし、[**サンドボックスによる保護**]領域で[**起動時に保護モードを有効にする**]のチェックが外れていることを確認します。

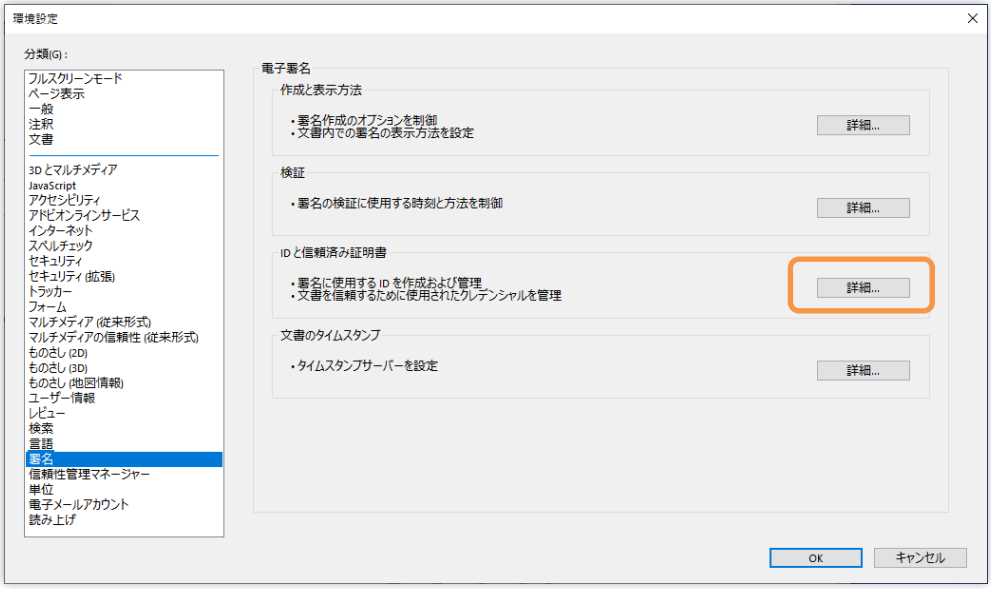


チェックがついていた場合はチェックを外します。このとき以下のウィンドウが表示されます。[**はい**]をクリックした後で Acrobat®を終了し、再度起動してから手順 1 を実行します。



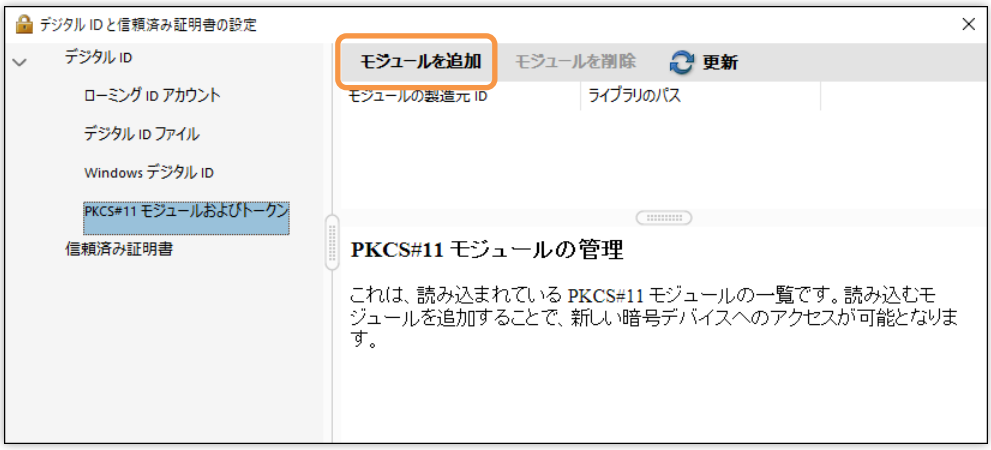
手順 3

分類で[**署名**]をクリックし、[**ID と信頼済み証明書**]領域の[**詳細**]をクリックします。



手順 4

下図のウィンドウで[**PKCS#11 モジュールおよびトークン**]をクリックし、[**モジュールを追加**]をクリックします。

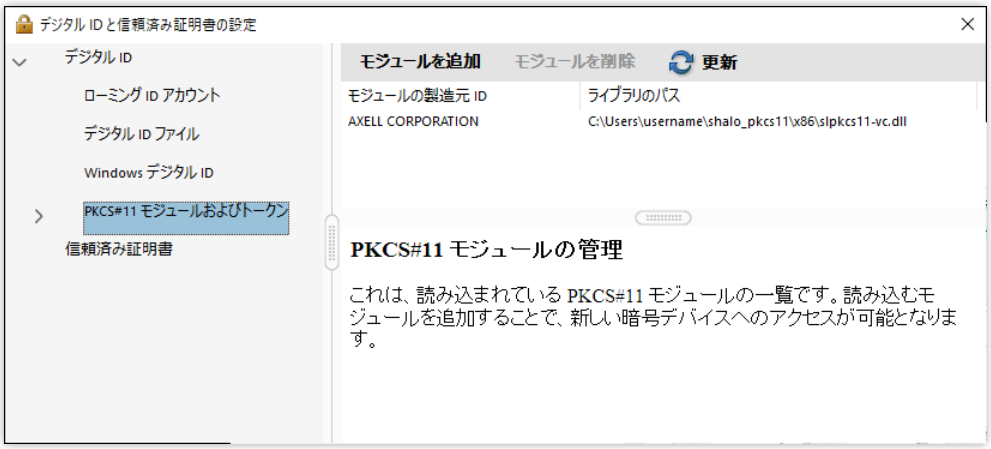


手順 5

SHALO AUTH の PKCS #11 モジュールを選択します。環境に応じて次のファイルを選択します。

Windows (Acrobat 32-bit)	C:¥ユーザー¥*ユーザー名*¥shalo_pkcs11¥x86¥slpkcs11-vc.dll
Windows (Acrobat 64-bit)	C:¥ユーザー¥*ユーザー名*¥shalo_pkcs11¥x64¥slpkcs11-vc.dll
macOS	/usr/local/lib/libslpkcs11.dylib

ファイルが正常に読み込まれると、下図のようにモジュールの一覧に登録されます。



[**PKCS#11 モジュールおよびトークン**]の下の階層に[**AXELL PKCS#11 library**]が追加されます。

[**AXELL PKCS#11 library**]の下の階層には PC に接続されている SHALO AUTH のデバイスラベルが表示されます。

7.2.2 Acrobat®から PKCS #11 モジュールを削除する

以下の手順で PKCS #11 モジュールを Acrobat®から削除します。

1. Windows: メニューから[**編集**] > [**環境設定**]をクリックします。
   macOS: メニューから[**Acrobat Reader**] > [**環境設定**]または[**Acrobat Pro DC**] > [**環境設定**]をクリックします。
2. [**署名**]をクリックし、[**ID と信頼済み証明書**]領域の[**詳細**]をクリックします。
3. [**PKCS#11 モジュールおよびトークン**]を選択し、リストから SHALO AUTH の PKCS #11 モジュールファイルを選択します。
4. [**モジュールを削除**]をクリックします。

この手順をスクリーンショットとともに説明します。

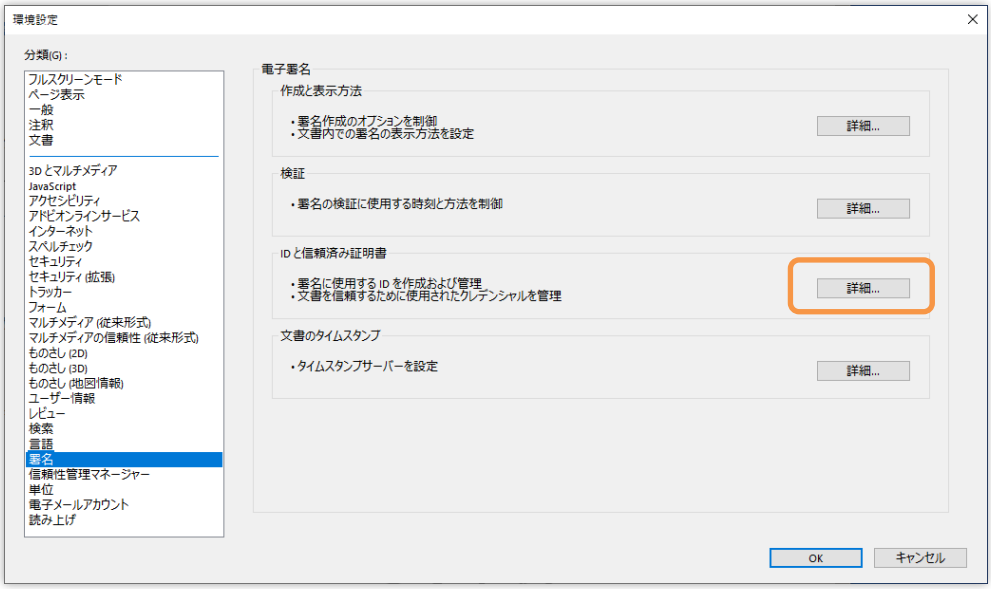
手順 1

Windows の場合: メニューから[**編集**] > [**環境設定**]をクリックします。

macOS の場合: メニューから[**Acrobat Reader**] > [**環境設定**]または[**Acrobat Pro DC**] > [**環境設定**]をクリックします。
最初のメニュー項目はアプリケーションの品種によって細部が異なります。

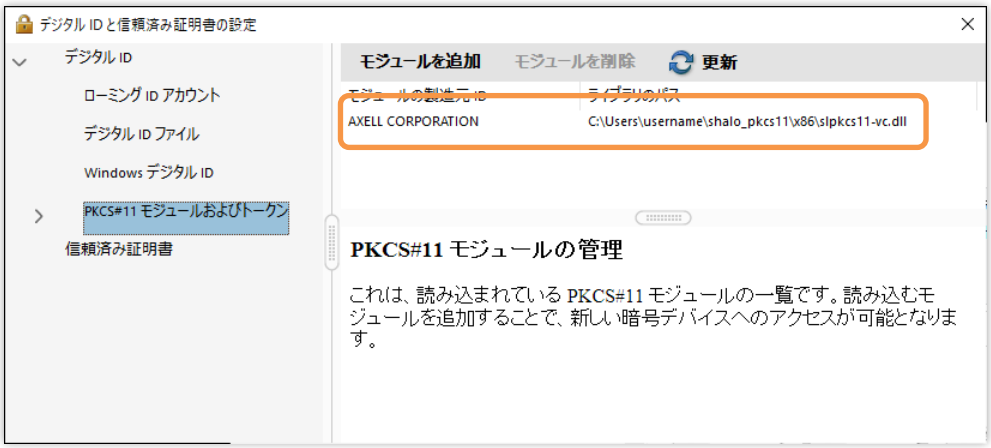
手順 2

分類で[**署名**]をクリックし、[**ID と信頼済み証明書**]領域の[**詳細**]をクリックします。



手順 3

下図のウィンドウで[**PKCS#11 モジュールおよびトークン**]をクリックし、リストから SHALO AUTH の PKCS #11 モジュールファイルを選択します。



手順 4

[**モジュールを削除**]をクリックします。

7.3 SHALO AUTH からデジタル ID を取り込む

Acrobat®で初めて SHALO AUTH を使う場合や SHALO AUTH の鍵を変更した場合は、SHALO AUTH の証明書をデジタル ID として Acrobat®に取り込みます。

これには SHALO AUTH を PC に接続して Acrobat®で次の手順を踏みます。

1. Windows: メニューから[**編集**] > [**環境設定**]をクリックします。
   macOS: メニューから[**Acrobat Reader**] > [**環境設定**]または[**Acrobat Pro DC**] > [**環境設定**]をクリックします。
2. [**署名**]をクリックし、[**ID と信頼済み証明書**]領域の[**詳細**]をクリックします。
3. [**デジタル ID**]をクリックし、SHALO AUTH の鍵情報が読み込まれているか確認します。
4. （鍵情報が読み込まれていない場合）[**PKCS#11 モジュールおよびトークン**] > [**AXELL PKCS#11 library**]をクリックし、トークンラベルの一覧から使用する SHALO AUTH のデバイスラベルを選択して[**ログイン**]をクリックします。そしてパスワードとしてユーザーPIN を入力します。

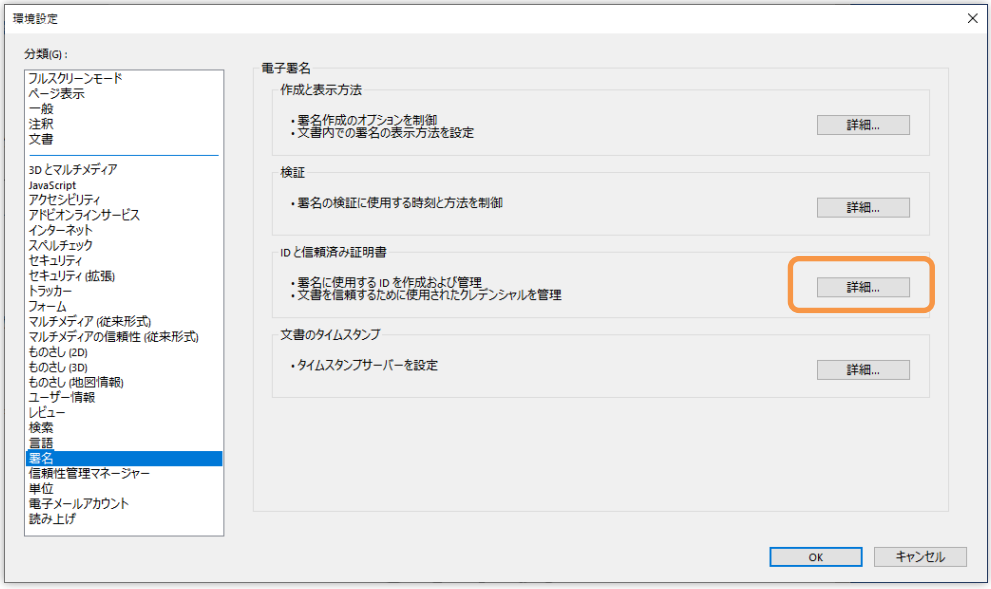
この手順をスクリーンショットとともに説明します。

手順 1～2

Windows の場合: メニューから[**編集**] > [**環境設定**]をクリックします。

macOS の場合: メニューから[**Acrobat Reader**] > [**環境設定**]または[**Acrobat Pro DC**] > [**環境設定**]をクリックします。
最初のメニュー項目はアプリケーションの品種によって細部が異なります。

表示される環境設定ウィンドウで下図のように分類で[**署名**]をクリックし、[**ID と信頼済み証明書**]の[**詳細**]をクリックします。



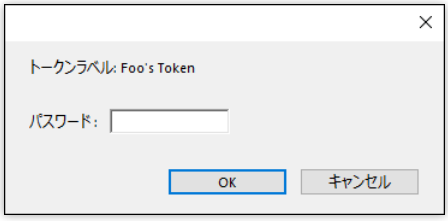
手順 3

[**デジタル ID**]をクリックし、SHALO AUTH の鍵情報が読み込まれているか確認します。格納メカニズムで「PKCS#11 暗号トークン」となっている項目が SHALO AUTH の証明書です。

手順 4（証明書が読み込まれていない場合）

[**PKCS#11 モジュールおよびトークン**] > [**AXELL PKCS#11 library**]をクリックし、トークンラベルの一覧から使用する SHALO AUTH のデバイスラベルを選択して[**ログイン**]をクリックします。

以下のウィンドウでユーザーPIN を入力します。



[**AXELL PKCS#11 library**]の子要素をクリックし、鍵情報が正しく読み込まれているか確認します。

7.4 デジタル ID の証明書を他の人に渡す

他の人にデジタル ID の証明書を提供する場合、X.509 証明書ではなく Acrobat®専用のファイル形式で証明書を出力する必要があります。

これには SHALO AUTH を PC に接続し、Acrobat ®で次の手順を踏みます。

1. Windows: メニューから[**編集**] > [**環境設定**]をクリックします。
   macOS: メニューから[**Acrobat Reader**] > [**環境設定**]または[**Acrobat Pro DC**] > [**環境設定**]をクリックします。
2. [**署名**]をクリックし、[**ID と信頼済み証明書**]の[**詳細**]をクリックします。
3. [**PKCS#11 モジュールおよびトークン**]を展開し、対象の SHALO AUTH を選択します。
4. 証明書を選択し、[**書き出し**]をクリックします。
5. 書き出しオプションを指定して書き出します。

この手順をスクリーンショットとともに説明します。

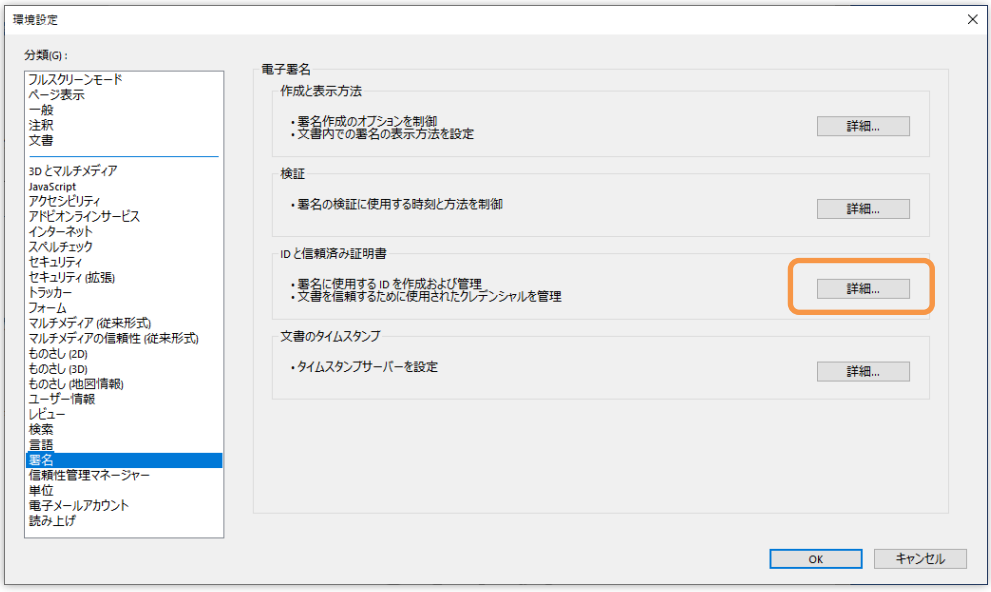
ECDSA 鍵の証明書を提供しないように注意してください。
ECDSA 鍵で暗号化された PDF ファイルは SHALO AUTH で閲覧できません。

手順 1～2

Windows の場合: メニューから[**編集**] > [**環境設定**]をクリックします。

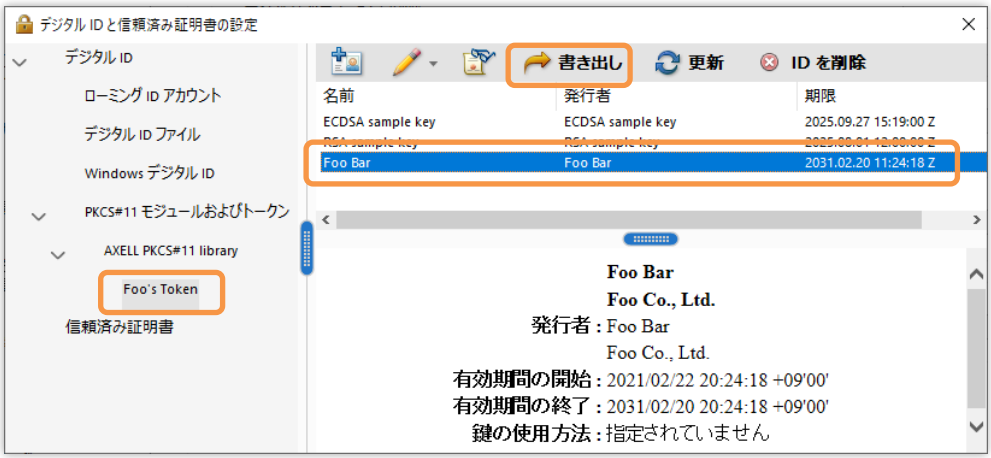
macOS の場合: メニューから[**Acrobat Reader**] > [**環境設定**]または[**Acrobat Pro DC**] > [**環境設定**]をクリックします。
最初のメニュー項目はアプリケーションの品種によって細部が異なります。

表示される環境設定ウィンドウで下図のように分類で[**署名**]をクリックし、[**ID と信頼済み証明書**]の[**詳細**]をクリックします。



手順 3, 4

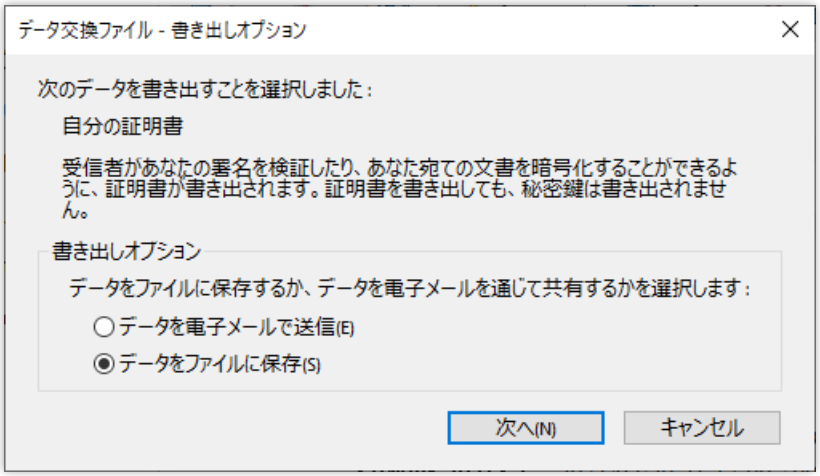
下図のウィンドウで[**PKCS#11 モジュールおよびトークン**]を展開し、対象の SHALO AUTH のデバイスラベルを選択します。証明書を選択して、書き出しをクリックします。



[**デジタル ID**]をクリックして、そこから証明書を選択することもできます。
複数の SHALO AUTH を接続している場合はデバイスラベルから選択すると証明書を簡単に探すことができます。

手順 5

下図のウィンドウで書き出しオプションを選択し、[**次へ**]をクリックします。その後はそれぞれのウィンドウに従って操作します。



7.5 デジタル ID で PDF ファイルを暗号化する

証明書による保護で PDF ファイルを暗号化する方法を説明します。PDF ファイルの暗号化ではデジタル ID の証明書が必要です。暗号化される PDF ファイルには複数の閲覧者を指定でき、次の 2 種類の方法で指定できます。

- 所有しているデジタル ID（SHALO AUTH のデジタル ID）で指定する
- デジタル ID の証明書ファイルで指定する

SHALO AUTH のデジタル ID を指定した場合でもユーザーPIN を入力する必要はありません。

ECDSA 鍵の証明書で PDF ファイルを暗号化しないように注意してください。
ECDSA 鍵で暗号化された PDF ファイルは SHALO AUTH で閲覧できません。

PDF ファイルの暗号化には Adobe® Acrobat®が必要です。
Adobe® Acrobat® Reader®では PDF ファイルを暗号化できません。

PDF ファイルを暗号化するには Adobe® Acrobat®で次の手順を踏みます。

1. PDF ファイルを開きます。
2. メニューから[**ファイル**] > [**プロパティ**]をクリックします。
3. [**セキュリティ**]タブを開き、セキュリティ方法で[**証明書によるセキュリティ**]を選びます。
4. 暗号化する文書コンポーネントと暗号化アルゴリズムを選択し、[**次へ**]をクリックします。
5. PC に接続されている SHALO AUTH で閲覧できるようにするにはデジタル ID を選択し、[**OK**]をクリックします。そうでない場合は[**キャンセル**]をクリックし、 [**続行**]をクリックします。
6. デジタル ID の証明書で閲覧者を指定する場合は、[**参照**]をクリックしてデジタル ID の証明書ファイルを選択します。
7. 閲覧者のデジタル ID をクリックして[**権限**]をクリックします。
8. 運用方針に従って適切に設定して[**OK**]をクリックします。
9. [**次へ**]をクリックし、[**完了**]をクリックします。
10. 文書のプロパティを閉じ、PDF ファイルを保存します。

PDF ファイルを暗号化するには閲覧者に適切な権限を設定する必要があります。
不適切な権限を許可すると閲覧制限を回避する PDF ファイルを生成できます。

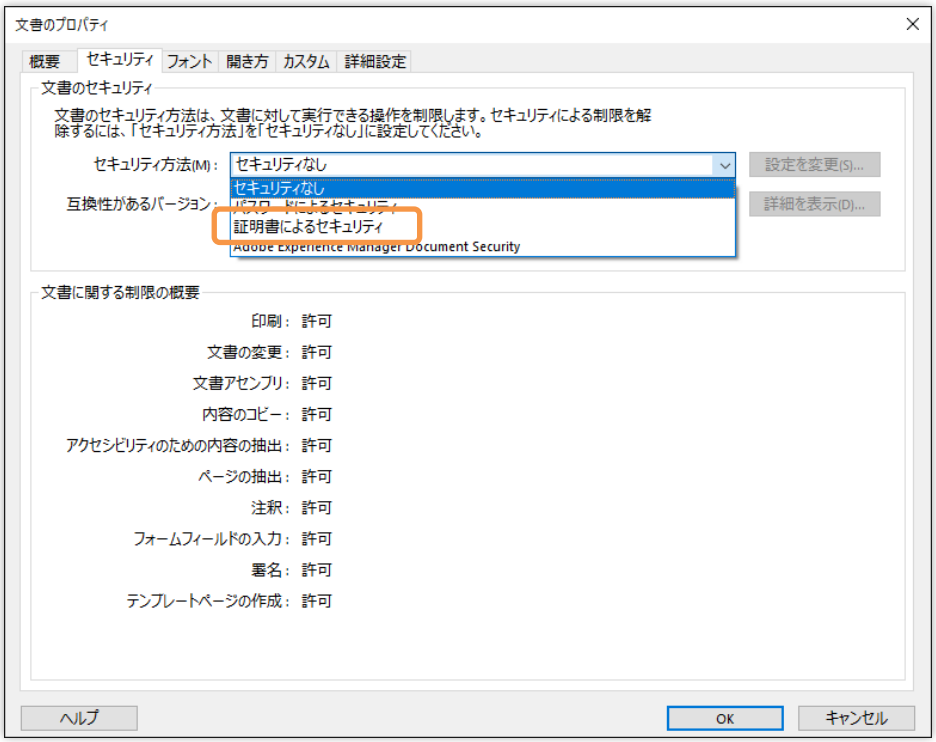
この手順をスクリーンショットとともに説明します。

手順 1～2

暗号化したい PDF ファイルを開き、メニューから[**ファイル**]＞[**プロパティ**]をクリックします。

手順 3

文書のプロパティウィンドウで、下図のように[**セキュリティ**]タブを開き、セキュリティ方法で[**証明書によるセキュリティ**]を選びます。

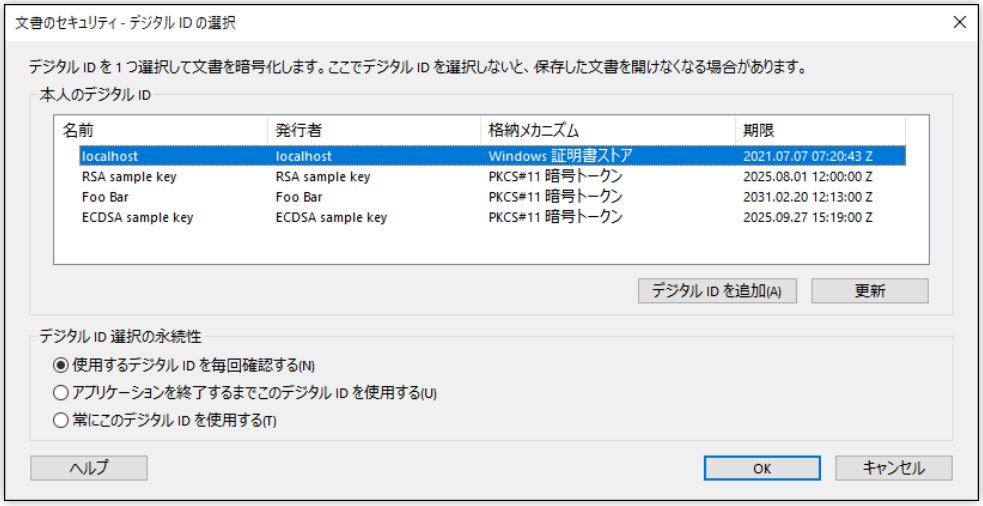


手順 4

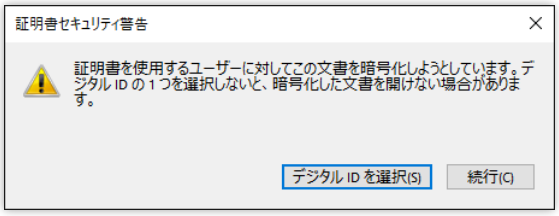
暗号化する文書コンポーネントと暗号化アルゴリズムを選択し、[**次へ**]をクリックします。

手順 5

[**本人のデジタル ID**]の欄には利用可能なデジタル ID が表示されます。PC に接続されている SHALO AUTH で閲覧できるようにするにはデジタル ID を選択し、[**OK**]をクリックします。

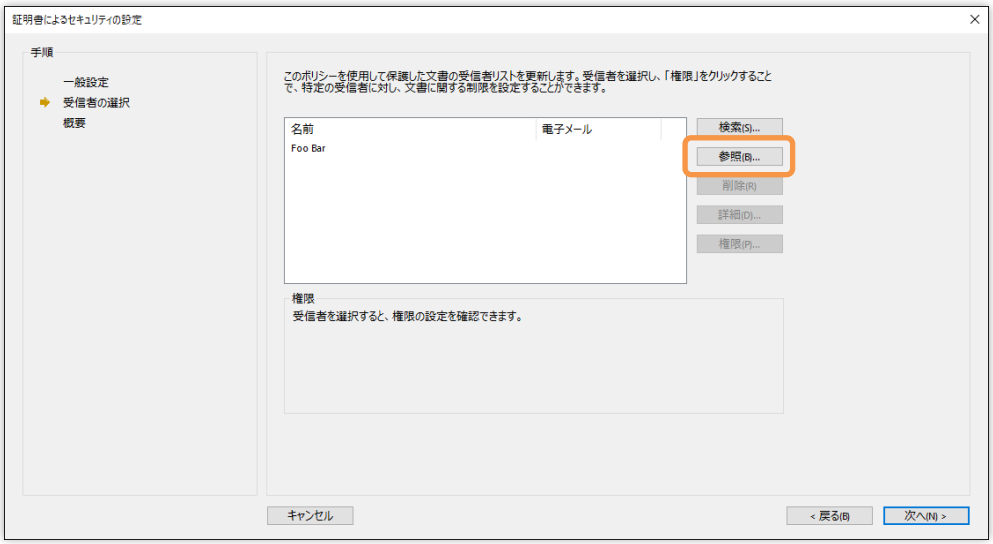


そうでない場合は[**キャンセル**]をクリックし、以下のウィンドウで[**続行**]をクリックします。



手順 6

デジタル ID の証明書で閲覧者を指定する場合は、[**参照**]をクリックしてデジタル ID の証明書ファイルを選択します。

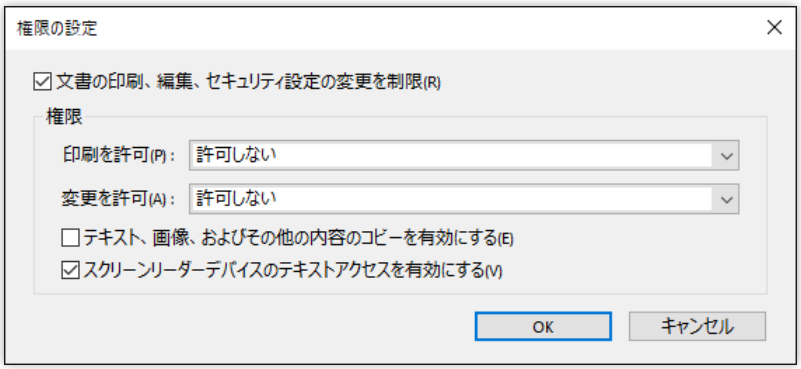


手順 7

閲覧者のデジタル ID をクリックして[**権限**]をクリックします。

手順 8

以下のウィンドウが表示されます。運用方針に従って適切に設定して[**OK**]をクリックします。



適切な権限を設定しないと閲覧制限を回避するデータを生成できてしまいます。
たとえば印刷を許可した場合、仮想プリンタに出力することで暗号化されていないデータを作成できます。

手順 9

[**次へ**]をクリックすると下のように表示されます。[**完了**]をクリックします。

手順 10

文書のプロパティを閉じ、PDF ファイルを保存します。

7.6 暗号化された PDF ファイルを閲覧する

暗号化された PDF ファイルを閲覧するには、SHALO AUTH を PC に装着してから Acrobat®で PDF ファイルを開きます。

ほかのソフトウェアが SHALO AUTH の汎用セキュリティキー機能を使用している場合、そのソフトウェアで SHALO AUTH の使用を止める必要があります。
FIDO U2F セキュリティ機能は使用中でも影響ありません。

Acrobat®から SHALO AUTH の使うと Acrobat®を終了するまでほかのソフトウェアから SHALO AUTH を使用できません。他のソフトウェアで SHALO AUTH の汎用セキュリティ機能を使う場合は一度 Acrobat®を終了してください。

通常、暗号化された PDF ファイルを開くと以下のいずれかのウィンドウが表示されます。Acrobat®でユーザーPIN を入力してある場合は PDF ファイルの内容が表示されます。

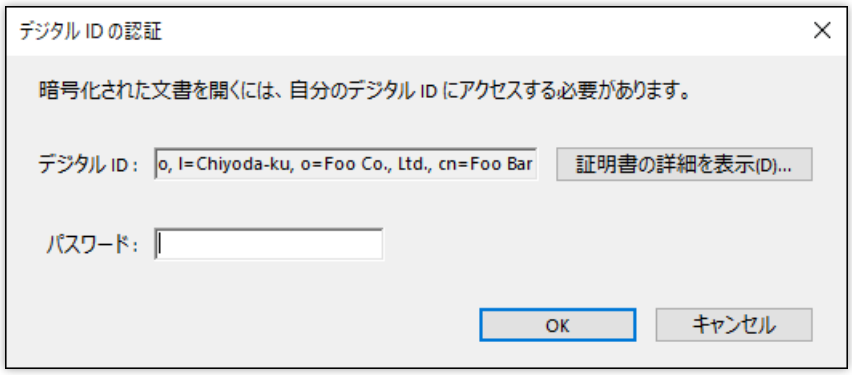


図 54　閲覧に必要なデジタル ID が Acrobat®に登録されている場合

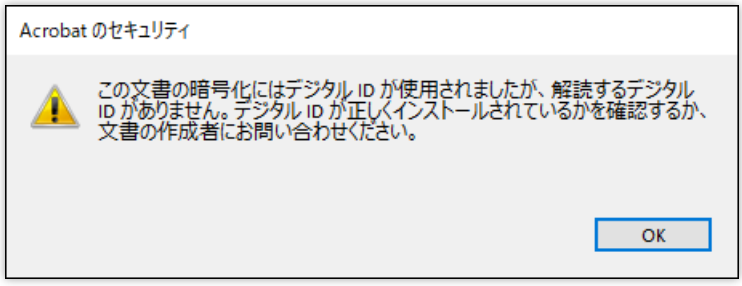


図 55　閲覧に必要なデジタル ID が Acrobat®に登録されていない場合

図 54 の場合、パスワードのフィールドに SHALO AUTH のユーザーPIN を入力して[**OK**]をクリックします。PIN 認証に成功し、正しく暗号を解除できると PDF ファイルを閲覧できます。

図 55 の場合、PDF ファイルの閲覧に必要なデジタル ID が Acrobat®に取り込まれていません。7.2 節と 7.3 節を参照して、SHALO AUTH からデジタル ID を Acrobat®取り込んでください。そして再び PDF ファイルを開きます。

7.7 デジタル ID で PDF ファイルに電子署名を付ける

PDF ファイルに SHALO AUTH で電子署名するには次の手順を踏みます。Adobe® Acrobat®と Adobe® Acrobat® Reader®で操作が少し異なります。

1. Acrobat®で PDF ファイルを開きます。
2. [**ツール**]を選択し、[**証明書**]をクリックします。
3. [**電子署名**]をクリックします。
4. PDF で電子署名の表示領域を、マウスをドラッグして指定します。
5. 署名に使用するデジタル ID を選択します。
6. SHALO AUTH のユーザーPIN を入力し（PIN 入力が求められた場合）、[**レビュー**]をクリックして（Adobe® Acrobat®のみ）、[**署名**]をクリックします。
7. PDF の保存先ファイルを指定します。

ほかのソフトウェアが SHALO AUTH の汎用セキュリティキー機能を使用している場合、そのソフトウェアで SHALO AUTH の使用を止める必要があります。
FIDO U2F セキュリティ機能は使用中でも影響ありません。

Acrobat®から SHALO AUTH の使うと Acrobat®を終了するまでほかのソフトウェアから SHALO AUTH を使用できません。他のソフトウェアで SHALO AUTH の汎用セキュリティ機能を使う場合は一度 Acrobat®を終了してください。

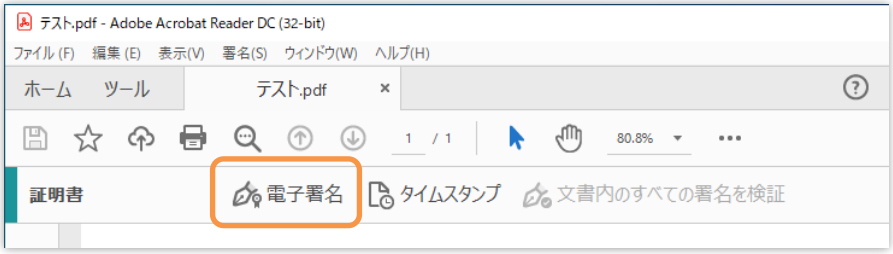
この手順をスクリーンショットとともに説明します。

手順 2

下図のように[**ツール**]を選択し、[**証明書**]をクリックします。

手順 3

ツールが下図のように証明書に切り替わってから[**電子署名**]をクリックします。



手順 4

PDF で電子署名を表示させる領域を、マウスをドラッグして指定します。

手順 5

下図のように SHALO AUTH に格納されている証明書の一覧が表示されます。この中から署名に使用するデジタル ID を選択し、[**続行**]をクリックします。

手順 6 (Adobe® Acrobat®)

下のウィンドウのように PIN 入力を求められた場合は SHALO AUTH のユーザーPIN を入力します。 [**レビュー**]をクリックしてから[**署名**]をクリックします。

手順 6 (Adobe® Acrobat® Reader®)

下のウィンドウのように PIN 入力を求められた場合は SHALO AUTH のユーザーPIN を入力します。そして[**署名**]をクリックします。

手順 7

ファイル保存ダイアログが表示されるので、電子署名を付与した PDF の保存先ファイルを指定します。

第 8 章

SSH 認証で使う

PKCS #11 をサポートする SSH クライアントは SHALO AUTH を使ってユーザー認証できます。また PKCS #11 をサポートしていないソフトウェアでも、PKCS #11 をサポートする認証エージェントに対応するものは SHALO AUTH を利用できます。

この章では SSH ソフトウェアで代表的な OpenSSH と PuTTY で SHALO AUTH を使ってユーザー認証する方法を説明します。またこれらが提供する認証エージェントの使用法を説明します。

この章のトピック

1. SSH とは？
2. SSH の鍵を準備する
3. 認証エージェントを準備する（Windows – OpenSSH）
4. 認証エージェントを準備する（Windows – PuTTY-CAC）
5. 認証エージェントを準備する（macOS）
6. 認証エージェントを準備する（Linux）
7. SSH クライアントを使う

8.1 SSH とは？

SSH（Secure Shell）はリモートホストと安全に通信するための通信規約（プロトコル）です。主な用途は次の 2 点です。

- リモートホストへのログイン
- ファイル転送

SSH を利用するにはリモートホストで **SSH サーバー**を動作させている必要があります。ユーザーはローカル PC で **SSH クライアント**を実行してリモートホストの SSH サーバーに接続します。

8.1.1 SSH クライアント

SSH 向けに次のアプリケーションが広く使用されています。

- OpenSSH　主要な OS で標準的に利用される SSH サーバー・クライアントです。
- PuTTY-CAC　暗号トークンに対応した Windows 向け SSH クライアントです。
- TeraTerm　様々な制御端末に対応した Windows 向け端末ソフトです。
- WinSCP　SSH を使用した Windows 向けのファイル転送ソフトです。

これらは SHALO AUTH を使ってパスワードレスでユーザー認証できます。OpenSSH と PuTTY-CAC は PKCS #11 モジュール経由で SHALO AUTH を使用します。TeraTerm と WinSCP は他のソフトが提供する**認証エージェント**と呼ばれる仕組みを利用して SHALO AUTH を間接的に使用します。

上に挙げた 4 つのソフトウェアで OpenSSH と PuTTY-CAC が認証エージェントを提供します。TeraTerm と WinSCP は PuTTY-CAC の認証エージェントを使用します。本章ではこの 4 つのソフトウェアを説明します。

OpenSSH の注意

ECDSA を使用するには、OpenSSH 8.0p1 以降を使用してください。
バージョンを確認するには、ターミナルで `ssh -V` を実行します。

環境別の OpenSSH の制限は 11.4 節を参照してください。

PuTTY-CAC の注意

PuTTY-CAC Release 0.70 Update 7 以降を使用してください。

8.1.2 認証エージェント

認証エージェントは秘密鍵や暗号トークンを使った認証処理を一括して扱う常駐プログラムです。認証エージェントとアプリケーションの関係は以下のようになっています。

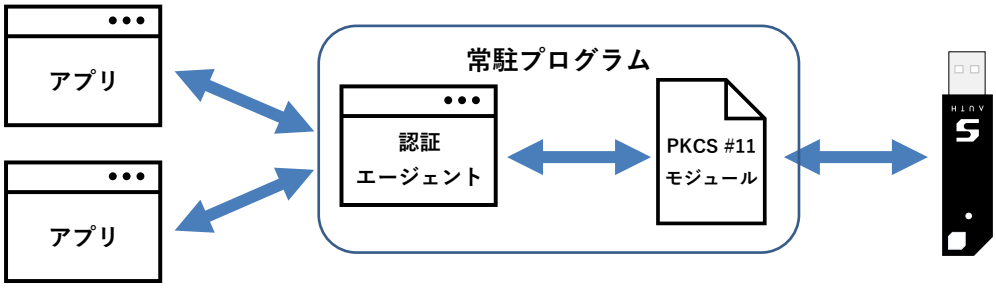


図 56　認証エージェントを経由した SHALO AUTH の利用

アプリケーションは秘密鍵や暗号トークンを使う処理を認証エージェントに委託します。認証エージェントを使えばアプリケーションごとに秘密鍵や暗号トークンを設定する必要がありません。

認証エージェントは暗号トークンにも有用です。認証エージェントは起動したままなので、一度ユーザーPIN 入力すれば再入力は不要です。

認証エージェント利用の流れ

SHALO AUTH を認証エージェントで利用するには次の手順で運用します。

1. 認証エージェントが起動していない場合はこれを起動します。
2. SHALO AUTH を PC に装着します。
3. 認証エージェントに PKCS #11 モジュールをロードします。

この手順の後に実行される SSH クライアントは常駐する認証エージェント経由でユーザー認証します。

SHALO AUTH を PC から取り外す場合や、他のアプリケーションで直接 SHALO AUTH を使用する場合は、認証エージェントから PKCS #11 モジュールをアンロードします。

認証エージェントで SHALO AUTH の利用を再開する場合は、認証エージェントに再度 PKCS #11 モジュールをロードします。

8.2 SSH 鍵を準備する

SSH でユーザー認証するには SSH 鍵を用意して以下を行います。

- SSH プライベート鍵を SHALO AUTH に登録する
- SSH 公開鍵をリモートホストに登録する

8.2.1 SSH 鍵を SHALO AUTH に登録する

SHALO Keyring を使って SHALO AUTH に SSH 鍵を登録します。SSH 鍵として利用可能な暗号方式は以下の通りです。

- RSA: 鍵長 2,048～4,096 ビット
- ECDSA: P-256 / P-384 / P-521

SHALO Keyring を使用した鍵の生成手順は 4.3 節を、すでに存在する鍵を SHALO AUTH に登録する方法は 4.4 節を参照してください。

8.2.2 SSH 公開鍵をリモートホストに登録する

リモートホストの~/.ssh/authorized_key ファイルに SSH 公開鍵を追加します。ホームディレクトリのファイル *key.pub* に SSH 公開鍵が保存されている場合、以下のようにします。

```
$ cat ~/key.pub >> ~/.ssh/authorized_keys
```

SSH 公開鍵は SHALO AUTH から取得できます。
SHALO Keyring を使用する方法は 4.6 節を参照してください。SHALO Keyring を使わない方法は 11.1 節を参照してください。

8.3 認証エージェントを準備する（Windows – OpenSSH）

Windows では Git for Windows や Cygwin で OpenSSH の認証エージェント ssh-agent を利用できます。しかし Windows10 に標準搭載される OpenSSH の認証エージェントでは SHALO AUTH を使用できません。

8.3.1 自動起動させる

Git Bash や Cygwin を実行した時に ssh-agent が自動的に起動されるように、それぞれの環境の ~/.bashrc に以下の内容を追加します。

~/.bashrc への追加内容

```
1   export SLPKCS11FILE=pkcs11file
2   ssh-add -l > /dev/null 2>&1
3   if [ "$?" == 2 ] ; then
4     SSH_AGENT_FILE=~/.ssh-agent
5     test -f $SSH_AGENT_FILE && source $SSH_AGENT_FILE > /dev/null
6     ssh-add -l > /dev/null 2>&1
7     if [ "$?" == 2 ] ; then
8       (umask 066; ssh-agent -P "/usr/lib/*,/usr/local/lib/*,$SLPKCS11FI
    LE" > $SSH_AGENT_FILE)
9       source $SSH_AGENT_FILE > /dev/null
10      setx SSH_AUTH_SOCK "$SSH_AUTH_SOCK" > /dev/null
11      setx SSH_AGENT_PID "$SSH_AGENT_PID" > /dev/null
12    fi
13  fi
14
15  alias shalo-add='ssh-add -s $SLPKCS11FILE'
16  alias shalo-remove='ssh-add -e $SLPKCS11FILE'
```

1 行目の「*pkcs11file*」に指定する PKCS #11 モジュールのパスは以下の通りです。

環境	PKCS #11 モジュールのファイルパス
Git for Windows 64bit	/c/Users/*ユーザー名*/shalo_pkcs11/x64/slpkcs11-mingw64.dll
Git for Windows 32bit	/c/Users/*ユーザー名*/shalo_pkcs11/x86/slpkcs11-mingw32.dll
Cygwin 64bit	/cygdrive/c/Users/*ユーザー名*/shalo_pkcs11/x64/slpkcs11-mingw64.dll
Cygwin 32bit	/cygdrive/c/Users/*ユーザー名*/shalo_pkcs11/x86/slpkcs11-mingw32.dll

~/.bashrc の Windows から見たファイルパスは以下の通りです。

環境	Windows におけるファイルパス
Git for Windows	C:¥Users¥*ユーザー名*¥.bashrc
Cygwin	*Cygwin インストール先ディレクトリ*¥home¥*ユーザー名*¥.bashrc

8.3.2 SHALO AUTH を登録・削除する

前節の設定によって、Git Bash または Cygwin で以下のエイリアスが利用できます。

shalo-add	PKCS #11 モジュールを ssh-agent にロードします。
shalo-remove	PKCS #11 モジュールを ssh-agent からアンロードします。

認証エージェントに SHALO AUTH を登録する

SHALO AUTH を PC に接続した後に `shalo-add` を実行します。

```
$ shalo-add↵
Enter passphrase for PKCS#11: ユーザーPIN を入力↵
Card added: /c/Users/ユーザー名/shalo_pkcs11/x64/slpkcs11-mingw64.dll
```

認証エージェントで SHALO AUTH の利用をやめる

`shalo-remove` を実行します。SHALO AUTH を取り外す場合も同様です。

```
$ shalo-remove↵
Card removed: /c/Users/ユーザー名/shalo_pkcs11/x64/slpkcs11-mingw64.dll
```

8.4 認証エージェントを準備する（Windows – PuTTY-CAC）

Windows では PuTTY-CAC の認証エージェント Pageant を利用できます。

8.4.1 起動・終了方法

Pageant を起動するには PuTTY-CAC のファイルの 1 つ pageant.exe を実行します。pageant.exe を実行してもウィンドウは表示されません。その代わり図 57 のように通知領域に常駐アイコンが追加されます。

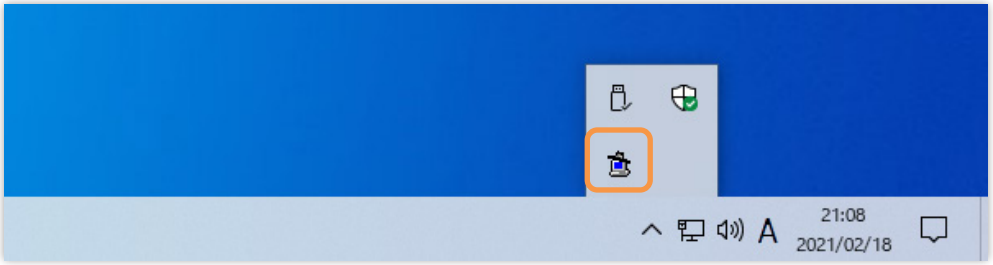


図 57 Pageant の常駐アイコン

常駐アイコンを右クリックすると Pageant のコンテキストメニューを表示できます。

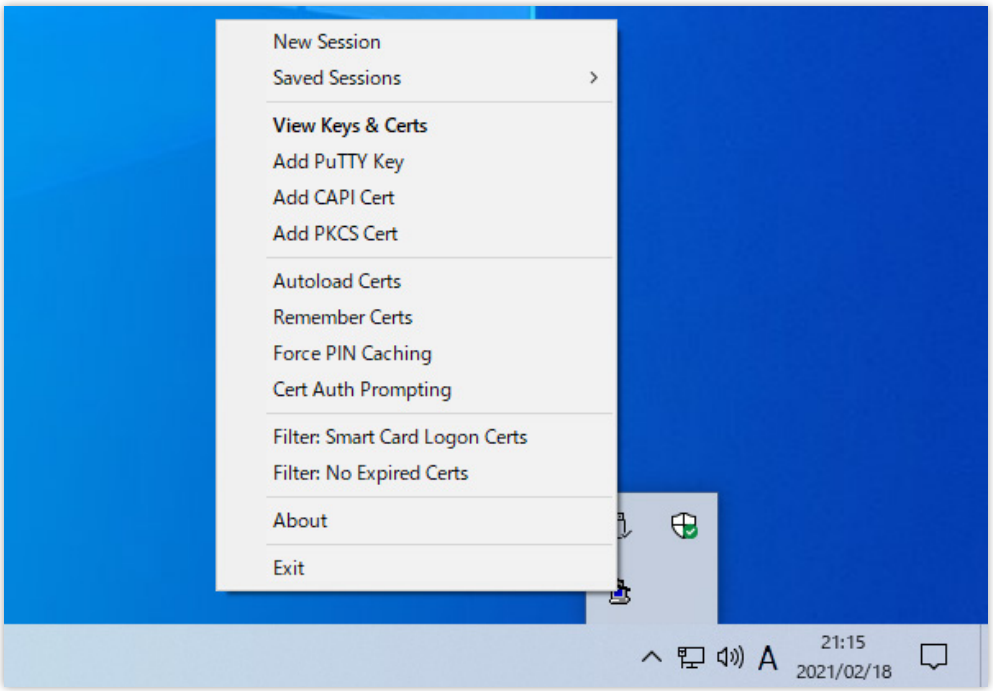


図 58 Pageant のコンテキストメニュー

Pageant を終了する際には、このコンテキストメニューで[**Exit**]をクリックします。

8.4.2 鍵を登録する

SHALO AUTH を Pageant に登録するには、SHALO AUTH の持つ SSH 鍵を 1 つずつ登録する必要があります。SHALO AUTH の持つすべての鍵を一括で登録することは出来ません。

次の手順で 1 つの SSH 鍵を登録できます。

1. Pageant のコンテキストメニューで[**Add PKCS Cert**]を選択します。
2. ファイル選択ダイアログで SHALO AUTH の PKCS #11 モジュールを選択します。
3. SHALO AUTH が持つすべての証明書から Pageant に登録するものを 1 つ選択します。

手順 2 では PuTTY-CAC の 32bit 版と 64bit 版で選択する PKCS #11 モジュールは次のように異なります。利用環境にあったファイルを選択してください。

ソフトウェア	PKCS #11 モジュールのファイルパス
PuTTY-CAC 32bit 版	C:¥Users¥*ユーザー名*¥shalo_pkcs11¥x86¥slpkcs11-vc.dll
PuTTY-CAC 64bit 版	C:¥Users¥*ユーザー名*¥shalo_pkcs11¥x64¥slpkcs11-vc.dll

手順 3 では下図（左）の証明書選択ダイアログが表示されます。[**その他**]をクリックすると、下図（右）のように使用可能な鍵の証明書がすべて表示されます。その中から SSH 認証に使用する鍵の証明書を 1 つ選んで[**OK**]をクリックします。

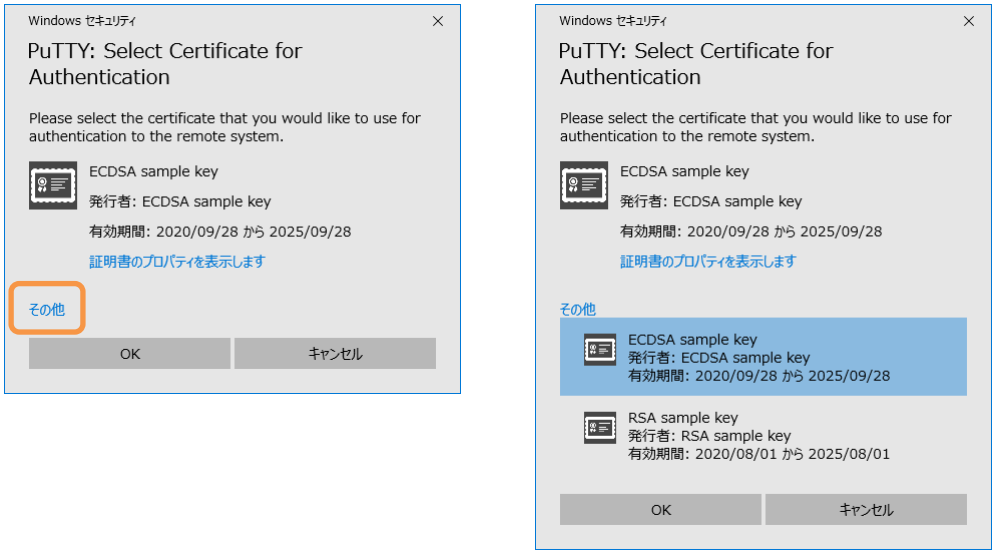


図 59　証明書選択ダイアログで鍵を選ぶ

8.4.3 登録済みの鍵を確認・削除する

登録されている鍵の一覧は Pageant Key List ウィンドウで確認できます。このウィンドウを開くには、Pageant のコンテキストメニューで [**View Keys & Certs**]をクリックします。

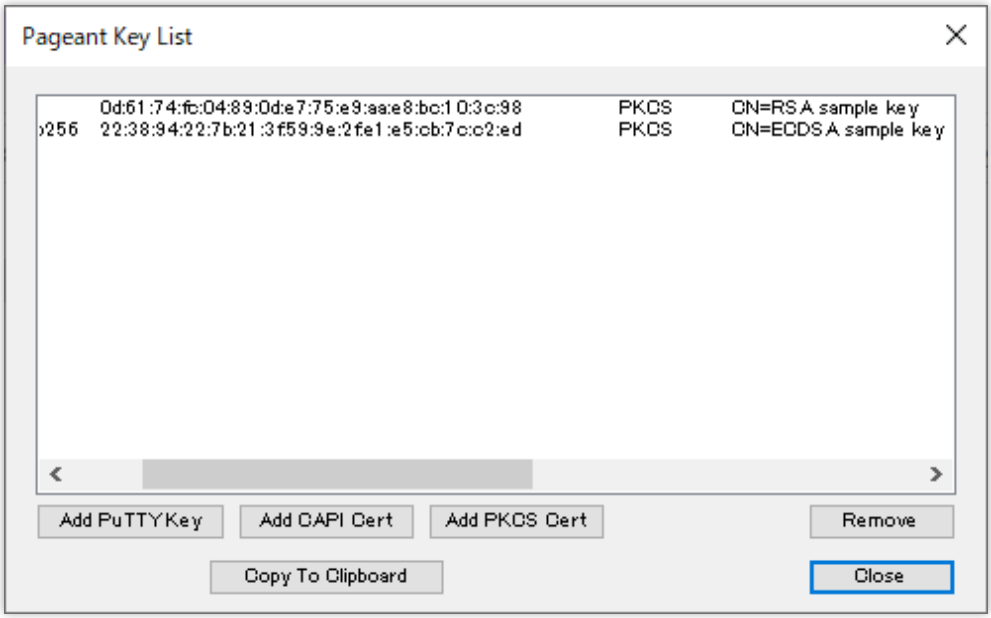


図 60　Pageant に登録された鍵の一覧

登録した鍵の削除

登録した鍵を削除するには、ウィンドウで鍵を選択して[**Remove**]をクリックします。

8.4.4 鍵を自動的に読み込ませる

Pageant 起動時に、Pageant に登録された PKCS #11 モジュールと SSH 鍵を自動的に読み込ませることができます。

この機能を有効にするには、Pageant のコンテキストメニューで [**Remember Certs**]にチェックを付けます。

SHALO AUTH を使う場合はこの機能を有効にしておき、次のように運用すると便利です。

- SHALO AUTH を PC に接続した後に Pageant を起動する
- SHALO AUTH を取り外す前に Pageant を終了する

Pageant を起動したときに SHALO AUTH が装着されていない場合、登録されている鍵は登録が解除されてしまいます。その場合はもう一度登録し直してください。

8.5 認証エージェントを準備する（macOS）

macOS では OpenSSH の認証エージェント ssh-agent を利用できます。

macOS に標準搭載されている OpenSSH は ssh-add コマンドを実行したときに自動的に ssh-agent を起動するように構成されています。
環境変数 SSH_AUTH_SOCK は launchd によって ssh-agent 向けのソケットがセットされています。macOS 標準搭載の ssh-agent サービスの停止は推奨しません。

エイリアスの追加

シェルの設定ファイルに定義を追加します。追加先の設定ファイルは次の通りです。

シェル種別	設定ファイル名
Bash（macOS 10.14 Mojave 以前のデフォルトシェル）	~/.bashrc
Zsh（macOS 10.15 Catalina 以降のデフォルトシェル）	~/.zshrc

設定ファイルへの追加内容

```
1  export SLPKCS11FILE=/usr/local/lib/libslpkcs11.dylib
2
3  alias shalo-add='ssh-add -s $SLPKCS11FILE'
4  alias shalo-remove='ssh-add -e $SLPKCS11FILE'
```

この追加によって、ターミナルで以下のエイリアスが利用できます。

shalo-add	PKCS #11 モジュールを ssh-agent にロードします。
shalo-remove	PKCS #11 モジュールを ssh-agent からアンロードします。

認証エージェントに SHALO AUTH を登録する

SHALO AUTH を mac に接続した後に **shalo-add** を 1 回実行します。

```
$ shalo-add↵
Enter passphrase for PKCS#11: ユーザーPIN を入力↵
Card added: /usr/local/lib/libslpkcs11.dylib
```

認証エージェントで SHALO AUTH の利用をやめる

shalo-remove を実行します。SHALO AUTH を取り外す場合も同様です。

```
$ shalo-remove↵
Card removed: /usr/local/lib/libslpkcs11.dylib
```

8.6 認証エージェントを準備する（Linux）

Linux では OpenSSH の認証エージェント ssh-agent を利用できます。

8.6.1 自動起動させる

~/.bashrc に以下を追加して、Linux にログインしたときに ssh-agent が適切に自動起動されるようにします。

~/.bashrc の追加データ

```
export SLPKCS11FILE=/usr/lib/libslpkcs11.so

ssh-add -l > /dev/null 2>&1
if [ "$?" == 2 ] ; then
  SSH_AGENT_FILE=~/.ssh-agent
  test -f $SSH_AGENT_FILE && source $SSH_AGENT_FILE > /dev/null

  ssh-add -l > /dev/null 2>&1
  if [ "$?" == 2 ] ; then
    (umask 066; ssh-agent > $SSH_AGENT_FILE)
    source $SSH_AGENT_FILE > /dev/null
  fi
fi

alias shalo-add='ssh-add -s $SLPKCS11FILE'
alias shalo-remove='ssh-add -e $SLPKCS11FILE'
```

8.6.2 SHALO AUTH を登録・削除する

前節の設定によって、以下のエイリアスが利用できます。

shalo-add	PKCS #11 モジュールを ssh-agent にロードします。
shalo-remove	PKCS #11 モジュールを ssh-agent からアンロードします。

認証エージェントに SHALO AUTH を登録する

SHALO AUTH を PC に接続した後に shalo-add を実行します。

```
$ shalo-add
Enter passphrase for PKCS#11: ユーザーPIN を入力
Card added: /usr/lib/libslpkcs11.so
```

認証エージェントで SHALO AUTH の利用をやめる

shalo-remove を実行します。SHALO AUTH を取り外す場合も同様です。

```
$ shalo-remove
Card removed: /usr/lib/libslpkcs11.so
```

8.7 SSH クライアントを使う

この節では 8.2 節で説明した次の項目を完了していることを前提にしています。

- SHALO AUTH に SSH プライベート鍵を登録します。
- リモートホストに SSH 公開鍵を登録します。

8.7.1 ssh を使う

ssh は OpensSSH のクライアントプログラムです。ssh-agent が起動している場合、**ssh** は自動的に ssh-agent を使用してユーザー認証します。ssh-agent を使用せずに SHALO AUTH を使う方法は、10.1 節を参照してください。

ssh の使用法は次の通りです。

```
ssh ユーザー名@ホスト名
```

ホスト名が hostname のリモートホストにユーザー名 username で接続する例を示します。

```
$ ssh username@hostname↵
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-58-generic x86_64)

～略～
username@ubuntu:~$
```

パスワードとユーザーPIN のいずれの入力も不要です。
パスワードの入力を求められた場合、ssh-agent に SHALO AUTH を正しく登録できていないか、リモートホストに SSH 公開鍵を正しく登録できていません。

初回接続時の警告

ローカル PC から **ssh** でリモートホストに初めて接続した場合、**ssh** は以下のメッセージを出力します。これは **ssh** が過去に接続したリモートホストの公開鍵を記録していて、未知のリモートホストへの接続やホスト名の成り済ましを警告するためです。

```
The authenticity of host 'hostname (IP アドレス)' can't be established.
ECDSA key fingerprint is SHA256:リモートホストのフィンガープリント.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

リモートホストのフィンガープリントを知っている場合、表示されたものと一致するか確認してください。接続先が本物であれば **yes** と入力してエンターキーを押します。そうすると **ssh** はこのリモートホストとフィンガープリントを~/.ssh/known_hosts ファイルに保存し、ユーザー認証を開始します。

8.7.2 plink を使う

plink は PuTTY のコマンドライン接続ツールです。Pageant が起動している場合、**plink** は自動的に Pageant を使用してユーザー認証します。

plink を実行するには、環境変数 PATH に PuTTY-CAC のディレクトリを追加するか、PuTTY-CAC のディレクトリで次のように実行します。

```
plink ユーザー名@ホスト名
```

以下は PowerShell から **plink** を使用して、ホスト名が hostname のリモートホストにユーザー名 username で接続する例です。

```
PS C:¥PuTTY-CAC>plink username@hostname↵
Using username "username".
```

ユーザーPIN を Pageant に入力していなければ下図のように PuTTY の認証ウィンドウが表示されます。 [**パスワード**]に SHALO AUTH のユーザーPIN を入力して[**OK**]をクリックします。

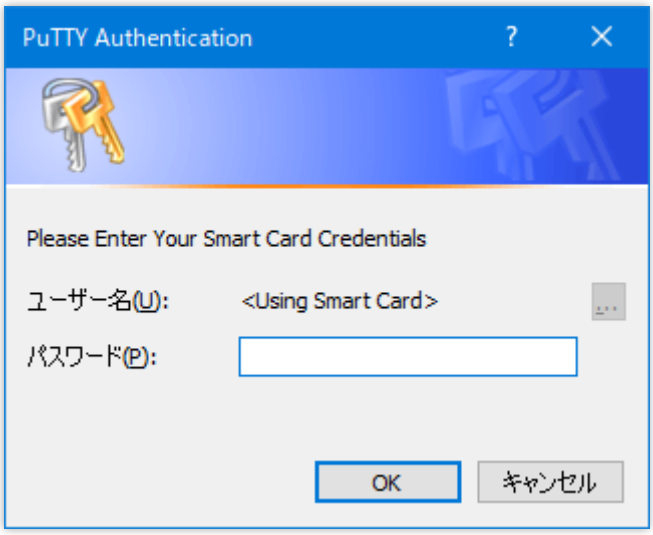


図 61　PuTTY の認証ウィンドウ

認証が成功すると次のメッセージが表示されます。エンターキーを押すとリモートホストとの接続が確立します。

```
Access granted. Press Return to begin session. ↵
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-58-generic x86_64)

～略～
username@ubuntu:~$
```

初回接続時の警告

ローカル PC からリモートホストに PuTTY で初めて接続した場合、**plink** は以下のメッセージを出力します。これは PuTTY が過去に接続したリモートホストの公開鍵を記録していて、未知のリモートホストへの接続やホスト名の成り済ましを警告するためです。

```
WARNING - POTENTIAL SECURITY BREACH!
The server's host key does not match the one PuTTY has
cached in the registry. This means that either the
server administrator has changed the host key, or you
have actually connected to another computer pretending
to be the server.
The new ssh-ed25519 key fingerprint is:
ssh-ed25519 255 リモートホスト固有データ
If you were expecting this change and trust the new key,
enter "y" to update PuTTY's cache and continue connecting.
If you want to carry on connecting but without updating
the cache, enter "n".
If you want to abandon the connection completely, press
Return to cancel. Pressing Return is the ONLY guaranteed
safe choice.
Update cached key? (y/n, Return cancels connection)
```

リモートホストのフィンガープリントを知っている場合、表示されたものと一致するか確認してください。接続先が本物であれば **y** と入力してエンターキーを押します。そうすると **plink** はこのリモートホストとフィンガープリントをレジストリに保存し、ユーザー認証を開始します。

8.7.3 putty を使う

putty は PuTTY の GUI 接続ツールです。Pageant が起動している場合、putty は自動的に Pageant を使用してユーザー認証します。putty を使って SSH 接続する手順を 3 ステップで説明します。説明は PuTTY Release 0.74 に基づいています。

はじめに、putty を起動して表示される以下のウィンドウで[**Host Name**]に接続先ホスト名を入力して[**Open**]をクリックします。ここでは例としてホスト名が hostname のリモートホストにユーザー名 username で接続します。

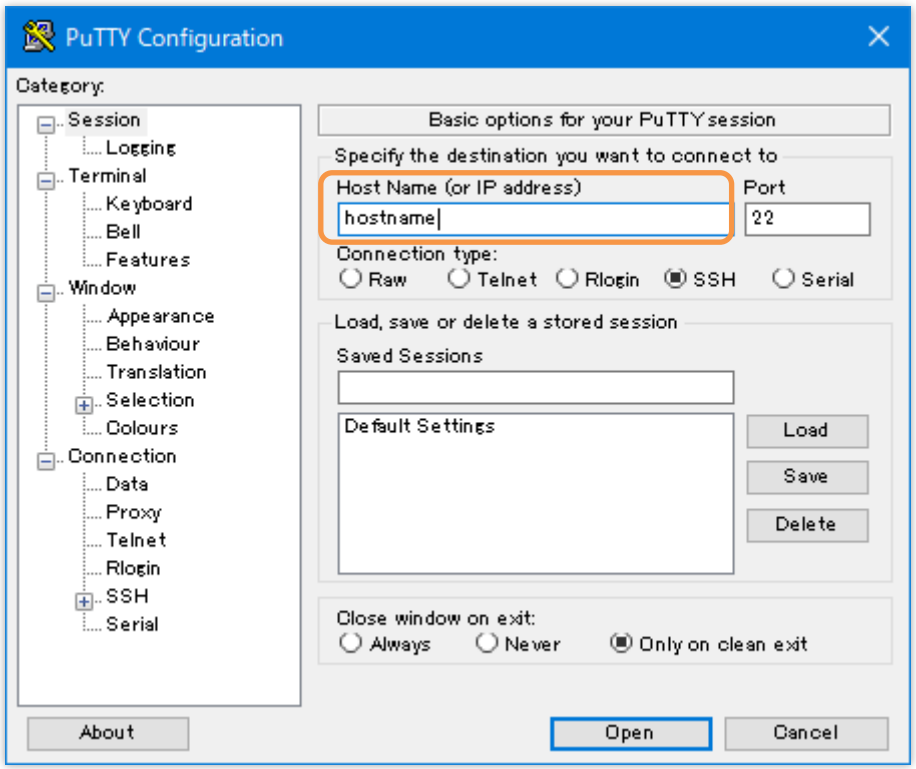


図 62　PuTTY 設定ウィンドウの接続先ホスト入力

すると PuTTY のターミナルウィンドウが表示され、下図のようにログインするユーザー名の入力が求められます。ここでユーザー名を入力してエンターキーを押します。

ターミナルウィンドウでパスワードの入力を求められた場合、Pageant に SHALO AUTH の鍵を正しく登録できていないか、リモートホストに SSH 公開鍵を正しく登録できていません。

最後に、ユーザーPIN を Pageant に入力していなければ下図のように PuTTY の認証ウィンドウが表示されます。 [**パスワード**]に SHALO AUTH のユーザーPIN を入力して[**OK**]をクリックします。リモートホストの認証に成功すると PuTTY のターミナルウィンドウにリモートホストのメッセージが表示されます。

初回接続時の警告

ローカル PC からリモートホストに初めて PuTTY で接続した場合、PuTTY は以下のメッセージを出力します。これは PuTTY が過去に接続したリモートホストの公開鍵を記録していて、未知のリモートホストへの接続やホスト名の成り済ましを警告するためです。

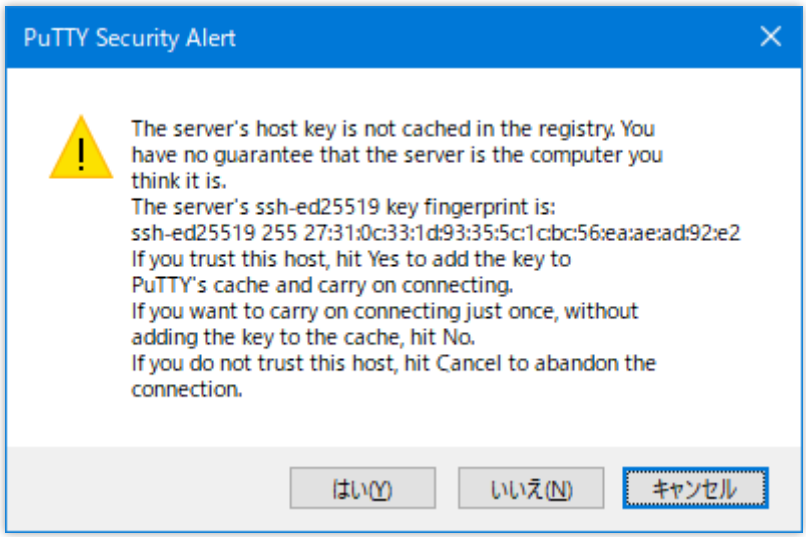


図 63　PuTTY で初めて接続したサーバーに対する警告

リモートホストのフィンガープリントを知っている場合、表示されたものと一致するか確認してください。接続先が本物であれば[**はい**]をクリックします。そうすると `putty` はこのリモートホストとフィンガープリントをレジストリに保存し、ユーザー認証を開始します。

8.7.4 TeraTerm を使う

TeraTerm は Pageant を使用すると SHALO AUTH でユーザー認証できます。TeraTerm で SSH 接続する手順を 3 ステップで説明します。説明は TeraTerm バージョン 4.105 に基づいています

はじめに、TeraTerm を起動して表示される下図のウィンドウで[**ホスト**]に接続先ホスト名を入力し、[**OK**]をクリックします。ここでは例としてホスト名 hostname にユーザー名 username で接続します。

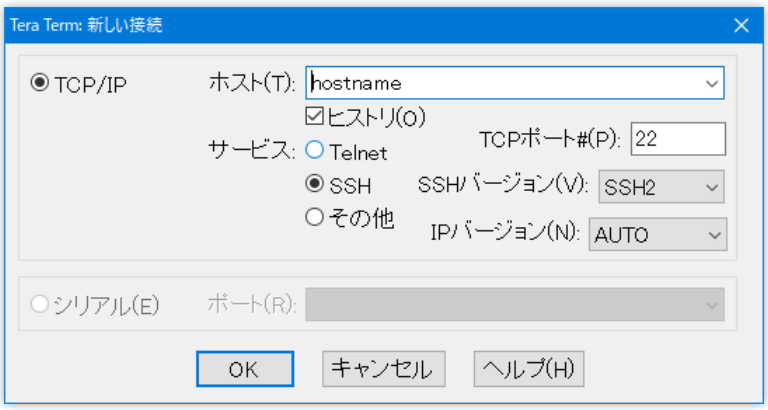


図 64　TeraTerm の接続先入力

次に、下図の SSH 認証ウィンドウで[**Pageant を使う**]を選択し、[**ユーザー名**]にログインするユーザー名を入力して[**OK**]をクリックします。

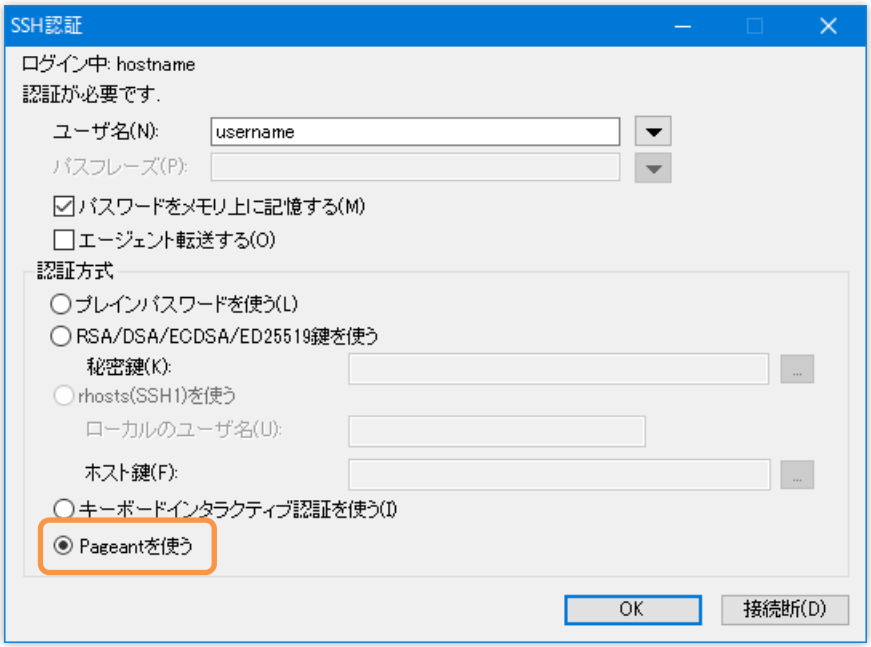


図 65　TeraTerm の SSH 設定

最後に、ユーザーPIN を Pageant に入力していなければ次の図のように PuTTY の認証ウィンドウが表示されます。 [**パスワード**]に SHALO AUTH のユーザーPIN を入力して[**OK**]をクリック

します。リモートホストの認証に成功すると TeraTerm のターミナルウィンドウにリモートホストのメッセージが表示されます。

初回接続時の警告

ローカル PC からリモートホストに初めて TeraTerm で接続した場合、TeraTerm は以下のメッセージを出力します。これは TeraTerm が過去に接続したリモートホストの公開鍵を記録していて、未知のリモートホストへの接続やホスト名の成り済ましを警告するためです。

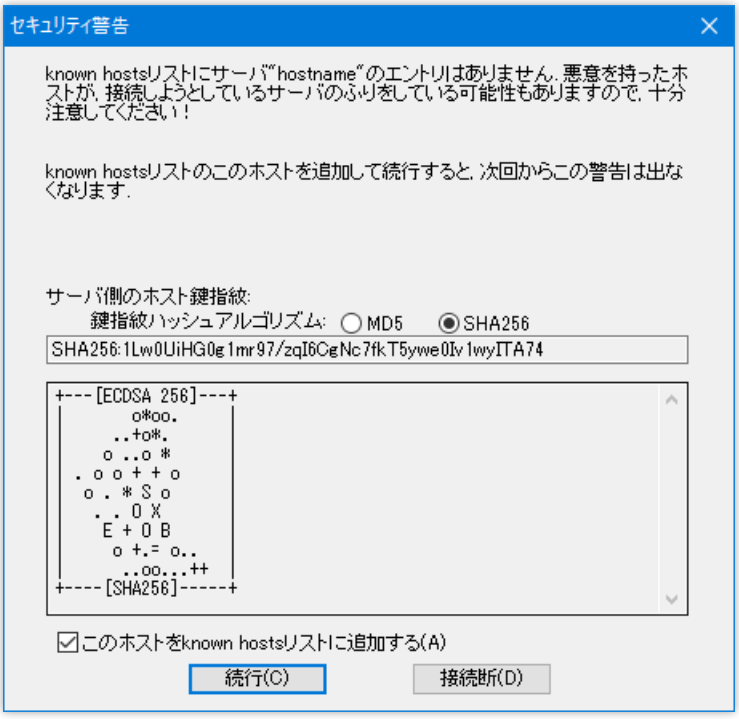


図 66　TeraTerm で初めて接続したサーバーに対する警告

リモートホストのフィンガープリントを知っている場合、表示されたものと一致するか確認してください。接続先が本物であれば[**はい**]をクリックします。そうすると TeraTerm はこのリモートホストとフィンガープリントをファイルに保存し、ユーザー認証を開始します。

8.7.5 WinSCP を使う

SCP/SFTP によるファイル転送ソフト WinSCP は SSH を使用します。WinSCP は Pageant を使用して SHALO AUTH でユーザー認証できます。説明は WinSCP バージョン 5.15.10 に基づいています

WinSCP はデフォルトで Pageant を使うように設定されています。以下のウィンドウで、[**ホスト名**]と[**ユーザー名**]だけを入力して[**ログイン**]をクリックします。ここでは例としてホスト名 hostname にユーザー名 username で接続します。

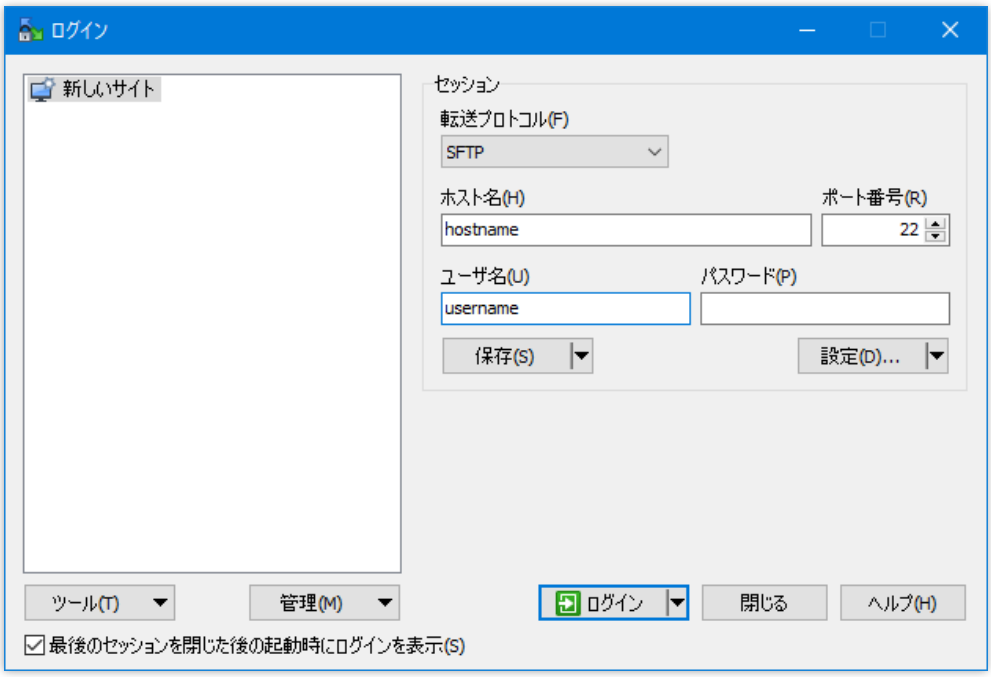


図 67　WinSCP の接続先入力

PuTTY の認証ウィンドウが表示された場合、[**パスワード**]に SHALO AUTH のユーザーPIN を入力して[**OK**]をクリックします。リモートホストの認証に成功するとリモートホストの認証に成功すると WinSCP のウィンドウにリモートホストのホームディレクトリが表示されます。

初回接続時の警告

ローカル PC からリモートホストに初めて WinSCP で接続した場合、WinSCP は以下のメッセージを出力します。これは WinSCP が過去に接続したリモートホストの公開鍵を記録していて、未知のリモートホストへの接続やホスト名の成り済ましを警告するためです。

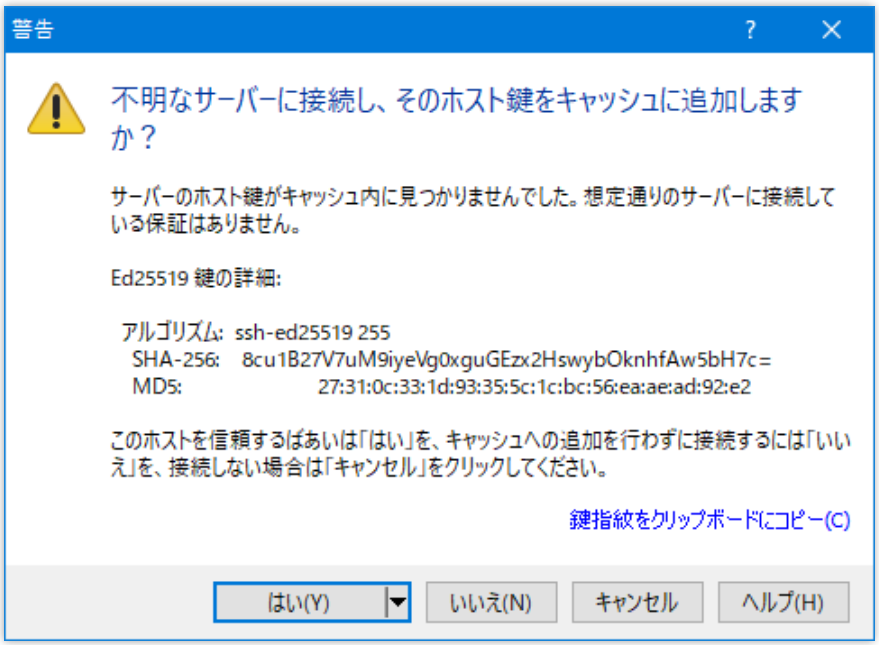


図 68　WinSCP で初めて接続したサーバーに対する警告

第 9 章

Git の SSH 認証で使う

Git はプログラムソースコードの変更履歴を記録・管理する分散型バージョン管理システムです。そのセキュア通信には SSH プロトコルが使われています。

この章では Git プラットフォームに GitHub を取り上げ、Git クライアントから GitHub への SSH 認証で SHALO AUTH を使用する方法を説明します。

この章のトピック

1. Git と SSH 認証
2. SSH 公開鍵を GitHub に登録する
3. SSH 接続をテストする
4. Git クライアントの互換性情報
5. Git クライアントの設定

9.1 Git と SSH 認証

Git はデータ転送用のプロトコルとして主に以下の 2 種類を使います。

HTTP プロトコル　ユーザー名とパスワードを使って認証します。
SSH プロトコル　SSH 鍵で認証します。

HTTP(HTTPS)プロトコルは以下のように `https://`でリポジトリを指定します。

```
https://server/user/project.git
```

SSH プロトコルは以下のように `ssh://`でリポジトリを指定します。

```
ssh://user@server/project.git
```

また、SSH プロトコルは SCP コマンドのような省略形でもリポジトリを指定できます。

```
user@server:project.git
```

SSH プロトコルを使うには

リモートリポジトリをクローンする際に SSH プロトコルの形式でリポジトリを指定します。クローンした後も、そのリポジトリでリモートリポジトリとデータ転送する際には常に SSH 認証が使われます。

HTTP プロトコルを使うリポジトリを SSH プロトコルに変える

リポジトリのリモート URL を変更する機能で HTTP プロトコルを使うリポジトリを SSH プロトコルに変更できます。

現在のリモート URL を確認するには、ターミナルでリポジトリに移動して `git remote -v` を実行します。GitHub の場合は以下のようになっています。

```
$ git remote -v
origin  https://github.com/ユーザー名/リポジトリ.git (fetch)
origin  https://github.com/ユーザー名/リポジトリ.git (push)
```

リモート URL を変更するには、`git remote set-url` で SSH プロトコル形式の URL を origin に指定します。

```
git remote set-url origin git@github.com:ユーザー名/リポジトリ.git
```

GitHub 以外のホスティングサーバーでは URL のパス構成が上記と異なる点に注意してください。

9.2 SSH 公開鍵を GitHub に登録する

SHALO AUTH は RSA か ECDSA の P-256/P-384/P-521 を SSH 鍵として使用できます。GitHub に SSH 公開鍵を登録する方法は、通常の SSH リモートホストへの登録方法とは異なり、ウェブブラウザで行います。

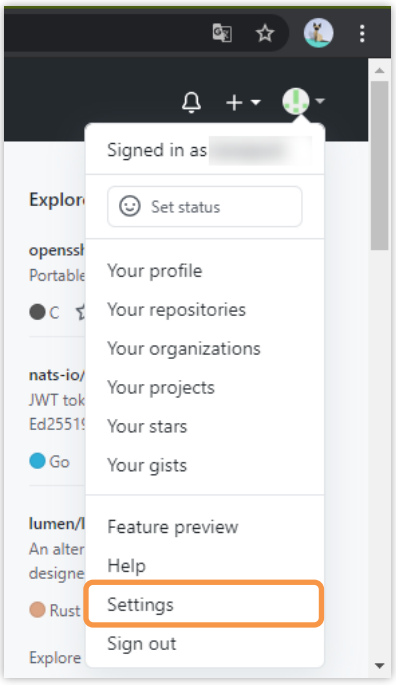
以下の方法で GitHub に SSH 公開鍵を登録できます。

1. https://www.github.com をウェブブラウザで開いてログインします。
2. 右上のプロフィール画像をクリックし、続いて[**Settings**]をクリックします。
3. 左のサイドバーで[**SSH and GPG keys**]をクリックします。
4. [**New SSH Key**]をクリックします。
5. [**Title**]に鍵の名前を入力し、[**Key**]に SSH 鍵を入力してから[**Add SSH key**]をクリックします。

この手順をスクリーンショットとともに説明します。

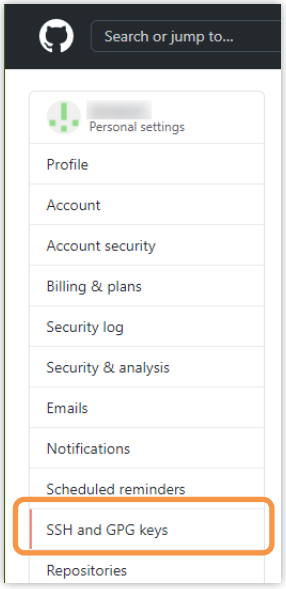
手順 1～2

GitHub にログインします。そして右上のプロフィール画像をクリックし、[**Settings**]をクリックします。



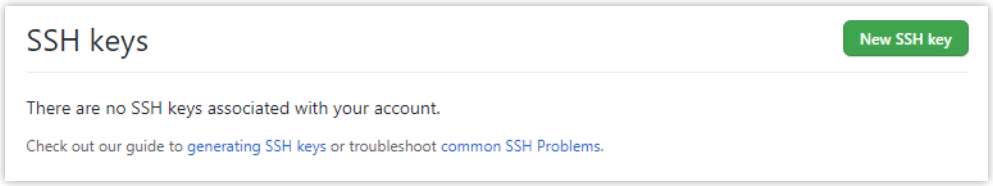
手順 3

左のサイドバーで[**SSH and GPG keys**]をクリックします。



手順 4

[**New SSH Key**]をクリックします。



手順 5

[**Title**]に鍵の名前を入力し、[**Key**]に SSH 公開鍵を入力します。最後に[**Add SSH key**]をクリックします。SSH 公開鍵については 4.6 節を参照してください。

9.3 SSH 接続をテストする

GitHub との SSH 接続をテストするには次の条件で SSH 接続します。

ホスト名 github.com
ユーザー名 git

この節で説明する GitHub への **SSH 接続テストを必ず実施**してください。
SSH 接続テストをしない場合、Git クライアントが SSH サーバー初回接続時の警告を受信してしまい、正常に動作しません。

9.3.1 認証エージェントに ssh-agent を使う場合

ssh-agent に SHALO AUTH を登録してから以下のコマンドを実行します。

```
ssh -T git@github.com
```

初めて GitHub に ssh コマンドで接続した場合、次のようなメッセージが出力されます。

```
The authenticity of host 'github.com (52.69.186.44)' can't be
established.
RSA key fingerprint is
SHA256:nThbg6kXUpJWGl7E1IGOCspRomTxdCARLviKw6E5SY8.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

これは接続先サーバーの成り済ましを防ぐためにサーバーの公開鍵の確認を求めています。GitHub の公開鍵と一致していることを確認してから **yes** と入力してエンターキーを押してください。

GitHub の公開鍵は次の URL で公開されています。
https://docs.github.com/ja/github/authenticating-to-github/githubs-ssh-key-fingerprints

認証に成功すると以下のように*ユーザー名*の箇所に GitHub アカウントの名前を出力し、SSH 接続が終了します。

```
Hi ユーザー名! You've successfully authenticated, but GitHub does not
provide shell access.
```

SSH 認証が正しく行えない場合、次のようなメッセージが出力されます。

```
git@github.com: Permission denied (publickey).
```

その場合は 11.5.9 節を参考にして、GitHub に登録した SSH 公開鍵や SHALO AUTH の環境を確認してください。

9.3.2 認証エージェントに Pageant を使う場合

Pageant に SHALO AUTH の鍵を登録してから以下のコマンドを実行します。

```
plink -T git@github.com
```

GUI 接続ツールの putty ではウィンドウがすぐに閉じられるためメッセージを確認できません。

初めて GitHub に plink で接続した場合、次のようなメッセージが出力されます。

```
The server's host key is not cached in the registry. You
have no guarantee that the server is the computer you
think it is.
The server's rsa2 key fingerprint is:
ssh-rsa 2048 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48
If you trust this host, enter "y" to add the key to
PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without
adding the key to the cache, enter "n".
If you do not trust this host, press Return to abandon the
connection.
Store key in cache? (y/n)
```

これは接続先サーバーの成り済ましを防ぐためにサーバーの公開鍵の確認を求めています。これに **y** と入力してエンターキーを押してください。

plink はサーバー公開鍵を SHA256 で提示しないため、GitHub の以下の URL で公開される公開鍵と一致するか確認できません。
https://docs.github.com/ja/github/authenticating-to-github/githubs-ssh-key-fingerprints

認証に成功すると以下のように*ユーザー名*の箇所に GitHub アカウントの名前を出力し、SSH 接続が終了します。

```
Hi ユーザー名! You've successfully authenticated, but GitHub does not
provide shell access.
```

SSH 認証が正しく行えない場合、次のようなメッセージが出力されます。

```
FATAL ERROR: No supported authentication methods available (server sent:
publickey)
```

その場合は GitHub に登録した SSH 公開鍵や SHALO AUTH の環境を確認してください。

9.4 Git クライアントの互換性情報

ここでは Git クライアント機能を持つ主要なソフトウェアについて、各 OS と認証エージェントを利用する場合の SHALO AUTH 動作リストを掲載します。

本節はマニュアル作成時の情報に基づいています。
これらソフトウェアの動作を必ずしも保証するものではありません。

動作環境は第 8 章で説明した認証エージェントで SHALO AUTH を利用する構成です。表中で Windows（ssh-agent）となっている列は Git for Windows の OpenSSH を使用します。

ソフトウェア	Windows (ssh-agent)	Windows (Pageant)	macOS (ssh-agent)	Linux (ssh-agent)
git コマンド 2.30.1	✔	✔※1	✔	✔
GitHub Desktop 2.6.0	✔	✔※1	✔	—
GitKraken 7.4.1	✘	✔※2	✔※2	✔※2
Source Tree 3.3.9	✘	✔※3	✔	—
TortoiseGit 2.11.0.0	✔	✔	—	—
Visual Studio 2017	✘	✘	—	—
Visual Studio 2019	✔	✔※1	—	—
Visual Studio 2019 for Mac	—	—	✔	—
Visual Studio Code	✔	✔※1	✔	✔
Xcode 12	—	—	✘	—

✔ 利用可能です
✘ 利用できません
— ソフトウェアは OS をサポートしません
※1 GIT_SSH 環境変数に PuTTY の plink.exe の絶対パスを登録します（9.5.1 節）
※2 GitKraken の[**Preferences**] > [**SSH**]で[**Use local SSH agent**]にチェックします（9.5.2 節）
※3 SourceTree の[**オプション**] > [**全般**]で[**SourceTree 起動時に SSH エージェントを起動します**]のチェックを外します（9.5.3 節）

認証エージェントに Pageant を使う場合は plink を使用して GitHub の SSH 接続をテストします。ssh-agent を使う場合は ssh を使用して SSH 接続をテストします。

9.5 Git クライアントの設定

この節では 9.4 節で※1～※3 のついた項目の設定方法を説明します。それらのついていない Git クライアントでは設定不要です。

9.5.1 GIT_SSH 環境変数（Windows で Pageant を使う場合のみ）

git コマンドは内部で ssh を起動します。GIT_SSH 環境変数が定義されている場合、そこで指定されたプログラムを起動して ssh の代わりに使用します。そのため GIT_SSH 環境変数に plink.exe を指定すれば Git の認証に PuTTY-CAC の Pageant を利用できます。

ここでは GIT_SSH 環境変数を定義する方法を 3 ステップで説明します。

はじめに、Windows の検索ボックスで下図のように『環境変数』と入力し、[**環境変数を編集**]をクリックします。

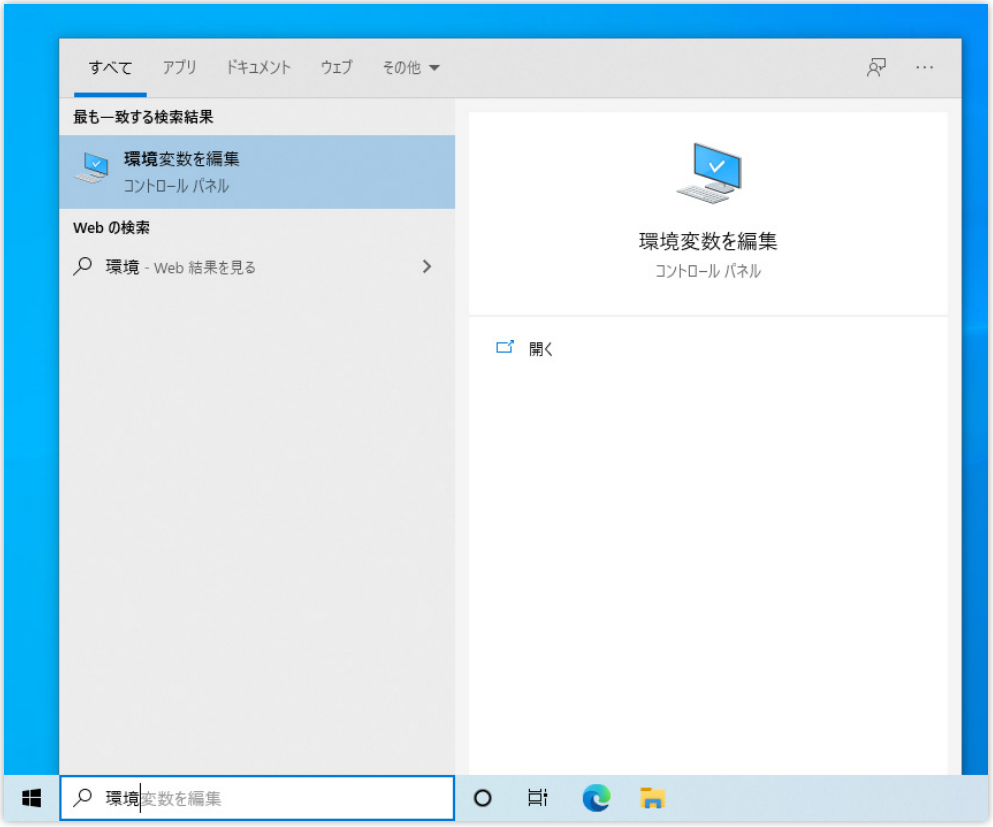


図 69　検索ボックスで「環境変数」を探す

次に、環境変数ウィンドウでユーザー環境変数の[**新規...**]をクリックします。

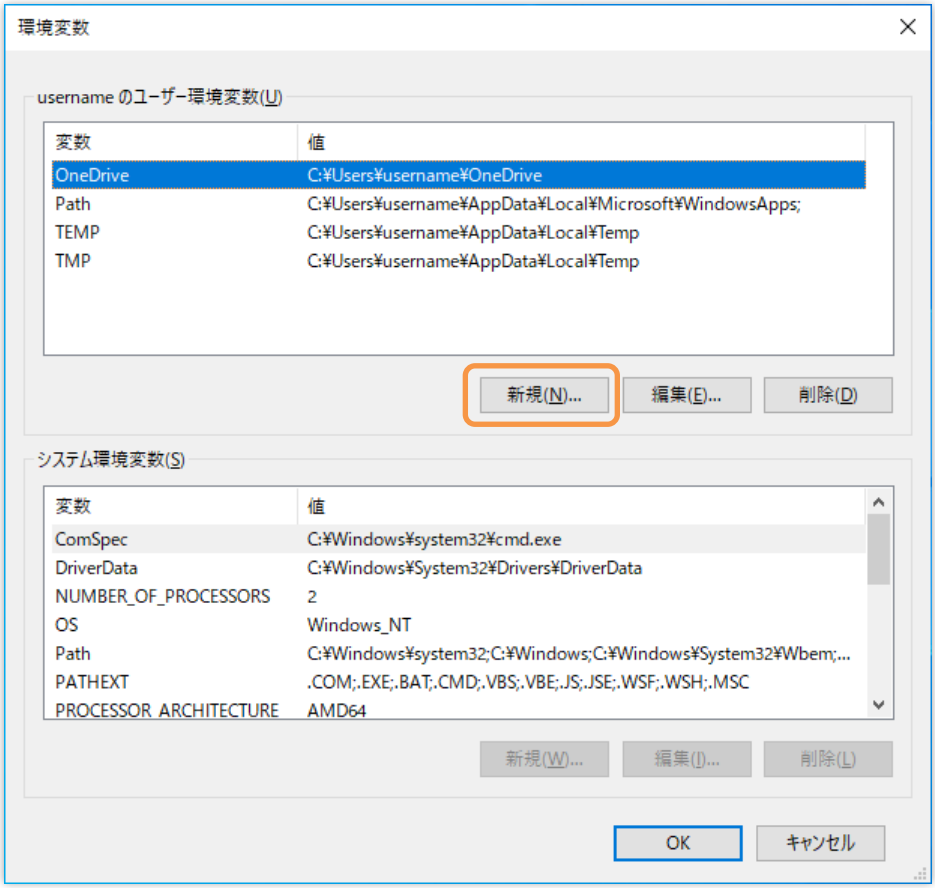


図 70　環境変数の追加

最後に、[**変数名**]に **GIT_SSH** を入力し、[**ファイルの参照**]をクリックして PuTTY-CAC の plink.exe を選択します。最後に[**OK**]をクリックします。

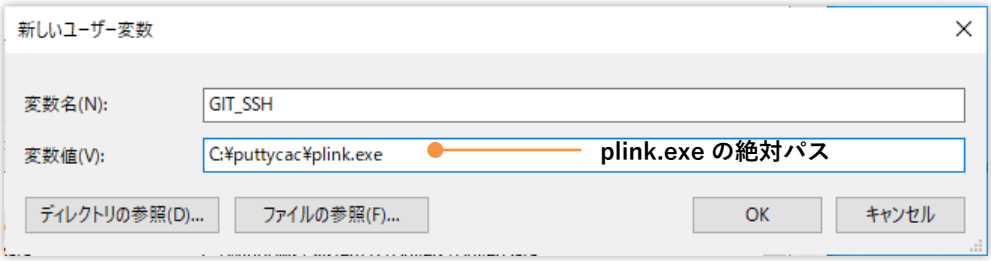


図 71　GIT_SSH 環境変数の作成

9.5.2　GitKraken

この節は GitKraken 7.4.1 を基にしています。他のバージョンでは画面構成・動作が異なる場合があります。

GitKraken は Windows 版で Pageant を、macOS・Linux 版では ssh-agent を使用できます。ただし初期設定で無効化されています。

GitKraken にこれらの認証エージェントを使用させるには、次の手順を踏みます。

1. GitKraken のメニューの[**File**] > [**Preferences...**] をクリックします。
2. 下図のウィンドウの[**Preferences**] > [**SSH**]を選択します。
3. [**Use local SSH agent**]にチェックを入れます。

GitKraken のウィンドウで設定箇所は下図の通りです。

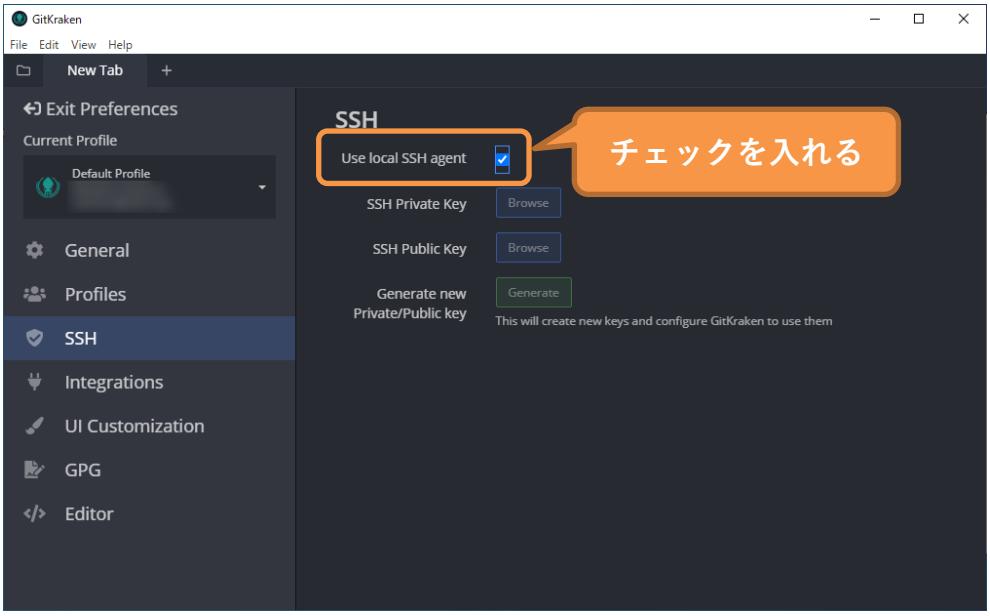


図 72　GitKraken の SSH 設定

9.5.3 Source Tree (Windows のみ)

この節は Source Tree 3.9.1 を基にしています。他のバージョンでは画面構成・動作が異なる場合があります。

Windows 版 Source Tree は PKCS #11 に対応していない PuTTY を搭載し、Source Tree 起動時にその Pageant を起動します。Windows 版 Source Tree では内蔵の Pageant を起動させないようにします。

Source Tree の起動時に Pageant を起動させないようにするには次の手順を踏みます。

1. Source Tree のメニューから[**ツール**] > [**オプション**]をクリックします。
2. 下図のオプションウィンドウの[**全般**]タブを選択します。
3. [**SourceTree 起動時に SSH エージェントを起動します**]のチェックを外します。

Source Tree の設定箇所は下図の通りです。

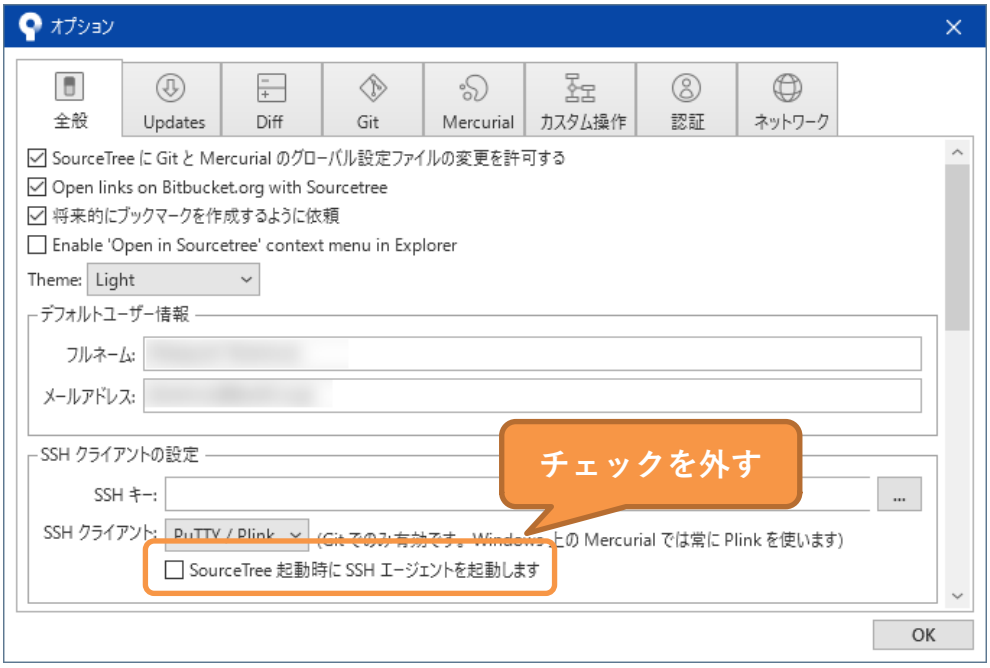


図 73 Source Tree の SSH 設定

第 10 章
より便利に使う

この章では SHALO AUTH をより活用していただくための情報を紹介します。

この章のトピック

1. 認証エージェントなしで OpenSSH から SHALO AUTH を使う
2. SSH 接続先で SHALO AUTH を使う
3. リモートデスクトップの接続先で SHALO AUTH を使う

10.1 認証エージェントなしで OpenSSH から SHALO AUTH を使う

この節では認証エージェントを使わずに OpenSSH で SHALO AUTH を使う方法を説明します。同時に複数の ssh から SHALO AUTH を利用できませんが、認証エージェントが使えない制限された環境では有用です。

方法は次の 2 種類があります。

- ssh の-I オプションを使う
- ssh の設定ファイル（~/.ssh/config）に登録する

ssh の-I オプション

ssh 実行時に-I オプションで PKCS #11 モジュールを指定できます。書式は以下の通りです。

```
ssh -I pkcs11file ユーザー名@ホスト名
```

コマンドを実行すると『Enter PIN for '*SHALO AUTH のラベル*'』と表示されます。そこで SHALO AUTH のユーザーPIN を入力してエンターキーを押します。

実行例を以下に示します。この例では SHALO AUTH のラベルは"Foo's Token"で、ホスト名 hostname のリモートホストにユーザー名 username で接続します。

```
$ ssh -I pkcs11file username@hostname↵
Enter PIN for 'Foo's Token': ユーザーPIN を入力↵
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)
～略～
```

pkcs11file には、下の表の中から環境に適したファイルパスを指定します。

環境		PKCS #11 モジュールのファイルパス
Windows	Gir for Windows 32 bit	/c/Users/*ユーザー名*/shalo_pkcs11/x86/slpkcs11-mingw32.dll
	Gir for Windows 64 bit	/c/Users/*ユーザー名*/shalo_pkcs11/x64/slpkcs11-mingw64.dll
	Cygwin 32 bit	/cygdrive/c/Users/*ユーザー名*/shalo_pkcs11/x86/slpkcs11-mingw32.dll
	Cygwin 64 bit	/cygdrive/c/Users/*ユーザー名*/shalo_pkcs11/x64/slpkcs11-mingw64.dll
macOS		/usr/local/lib/libslpkcs11.dylib
Linux		/usr/lib/libslpkcs11.so

8.3 節, 8.5 節, 8.6 節でシェルの設定ファイルに SLPKCS11FILE を登録している場合は、*pkcs11file* に$SLPKCS11FILE を指定できます。

~/.ssh/config

~/.ssh/config は `ssh` の設定ファイルです。`ssh` の`-I` オプション相当を設定ファイルに記載することで `ssh` 実行時のオプション指定を省略できます。ただし ssh 実行時にユーザーPIN を入力する点に変わりはありません。

認証エージェントを使わない場合でもこの方法を採用することで git コマンドから SHALO AUTH を使用できます。しかし Git LFS を使うリポジトリは利用できません。ユーザーPIN の入力ができない GUI の Git クライアントも同様です。

~/.ssh/config には接続先リモートホスト毎の設定を定義します。PKCS #11 モジュールを使用させる最小限の記述を以下に示します。斜体の文字列を環境にあわせて変更します。

```
Host 名称
     Hostname IP アドレスまたはリモートホストアドレス
     PKCS11Provider  PKCS#11 モジュールの絶対パス
```

PKCS#11 モジュールの絶対パスには、下の表の中から環境に適したファイルパスを指定します。

環境		PKCS #11 モジュールのファイルパス
Windows	Gir for Windows 32 bit	/c/Users/*ユーザー名*/shalo_pkcs11/x86/slpkcs11-mingw32.dll
	Gir for Windows 64 bit	/c/Users/*ユーザー名*/shalo_pkcs11/x64/slpkcs11-mingw64.dll
	Cygwin 32 bit	/cygdrive/c/Users/*ユーザー名*/shalo_pkcs11/x86/slpkcs11-mingw32.dll
	Cygwin 64 bit	/cygdrive/c/Users/*ユーザー名*/shalo_pkcs11/x64/slpkcs11-mingw64.dll
macOS		/usr/local/lib/libslpkcs11.dylib
Linux		/usr/lib/libslpkcs11.so

10.2 SSH 接続先で SHALO AUTH を使う

認証エージェントを使用してリモートホストに SSH 接続したときに、リモートホストでもローカル PC の認証エージェントを使用させることができます。これを **ssh agent forwarding** といいます。

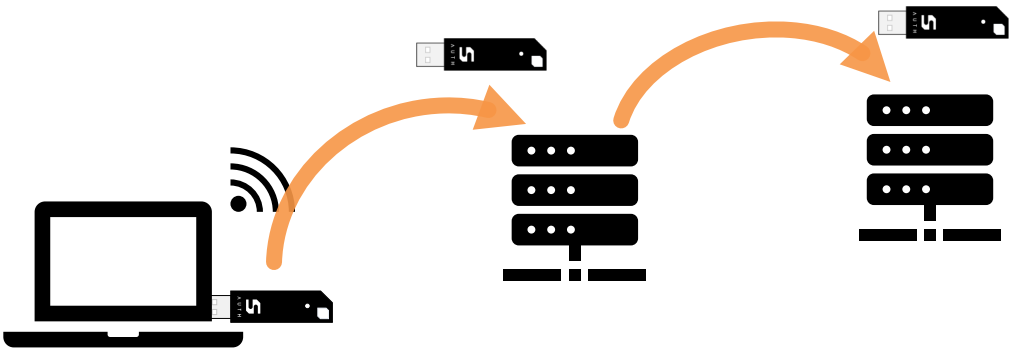


図 74　ローカル PC の認証エージェントをリモートホストに持ち込む

このためにはリモートホストで ssh agent forwarding を許可するように設定します。SSH クライアントは `ssh`, `plink`, `putty` ともに対応しています。

リモートホストに接続中でも、接続元 PC では SHALO AUTH を使用できます。

SSH 接続先リモートホストでは認証エージェントの提供する機能だけ使用できます。U2F セキュリティキーの機能は使用できません。

リモートホストの SSH サーバーを設定する

リモートホストで SSH サーバーの設定ファイルを変更します。設定ファイル sshd_config で次のように AllowAgentForwarding を有効にします。

sshd_config

```
AllowAgentForwarding yes
```

ssh で接続する

ssh コマンドで ssh agent forwarding するには、-A オプションを追加します。

```
ssh -A ユーザー名@ホスト名
```

例としてリモートホストに接続した後に、リモートホストと異なる SSH 鍵を使用する GitHub アカウントに SSH 接続をテストする使用法を以下に示します。

```
$ ssh -A username@hostname
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.8.0-43-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 updates can be installed immediately.
0 of these updates are security updates.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Mon Feb 22 19:06:07 2021 from 192.168.1.1
username@hostname:~$ ssh -T git@github.com
Hi username! You've successfully authenticated, but GitHub does not provide
shell access.
```

plink で接続する

plink も、ssh コマンドと同様に-A オプションで ssh agent forwarding が有効になります。

```
plink -A ユーザー名@ホスト名
```

putty で接続する

putty では、リモートホストに接続する前に[**Connection**] > [**SSH**] > [**Auth**]をクリックし、[**Allow agent forwarding**]にチェックを付けます。

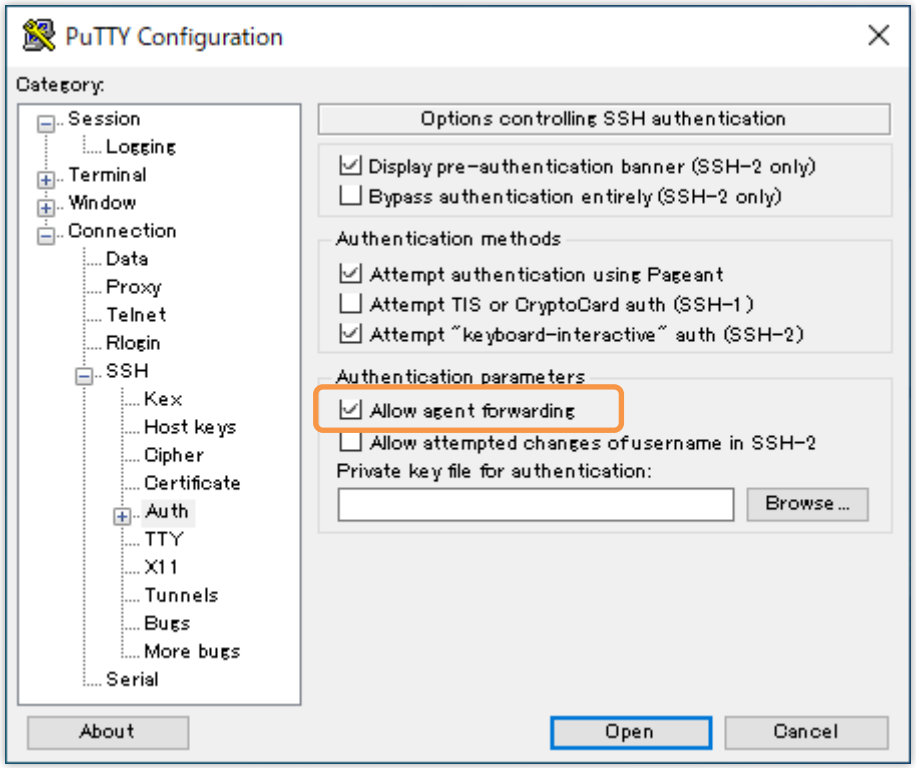


図 75　PuTTY で ssh agent forwarding を有効化

10.3　リモートデスクトップの接続先で SHALO AUTH を使う

Windows のリモートデスクトップで、接続元 PC に装着した SHALO AUTH をリモートデスクトップ接続先の PC で使用することができます。これには **RemoteFX USB デバイスリダイレクト**という機能を使用します。

SHALO AUTH をリモートデスクトップ接続先にリダイレクトすると、接続元 PC から SHALO AUTH を取り外して接続先 PC にそれを装着した場合と同じ状態になります。接続先 PC で SHALO AUTH 専用ソフトウェアや PKCS #11 モジュールのインストールが必要です。

図 76　ローカル PC の SHALO AUTH 取り外してリモート PC に接続する

SHALO AUTH をリモートデスクトップ接続先 PC にリダイレクトしている間は、接続元 PC で SHALO AUTH を使用できません。

リモートデスクトップ接続先では汎用セキュリティキー機能（PKCS #11）のみ使用できます。U2F セキュリティキーの機能は使用できません。

RemoteFX USB デバイスリダイレクトの利用要件は次の通りです。

接続先 PC	Windows 10 Pro または Windows 10 Enterprise
接続元 PC	Windows 10 Pro または Windows 10 Enterprise

Windows 10 Home エディションや macOS では使用できません。

この節では接続先 PC と接続元 PC の環境設定を解説し、その後で SHALO AUTH のリダイレクト方法を説明します。

10.3.1 接続先 PC を設定する

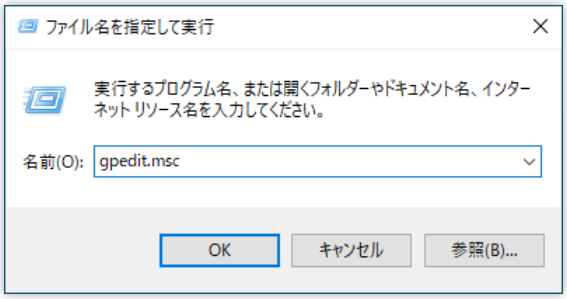
接続先 PC で次の手順を踏みます。

1. ローカルグループポリシーエディターを起動します。
2. 左のペインで以下をクリックして開きます。
   [**コンピューターの構成**] > [**管理用テンプレート**] > [**Windows コンポーネント**] > [**リモート デスクトップ サービス**] > [**リモートデスクトップ セッション ホスト**] > [**デバイスとリソースのリダイレクト**]
3. [**サポートされているプラグ アンド プレイ デバイスのリダイレクトを許可しない**]をダブルクリックします。
4. [**無効**]にチェックを付け、[**OK**]をクリックします。

この手順をスクリーンショットとともに説明します。

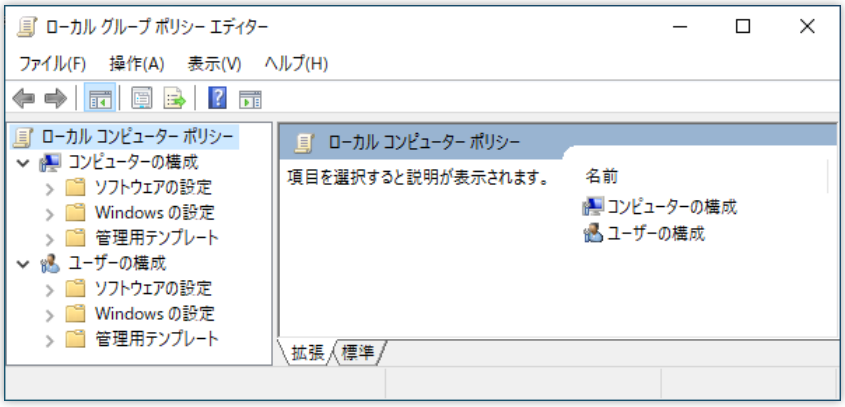
手順 1

スタートボタンを右クリックし（または Windows キー+X を押します）、[**ファイルを指定して実行**]を選択します。以下のウィンドウで gpedit.msc と入力して[**OK**]をクリックします。



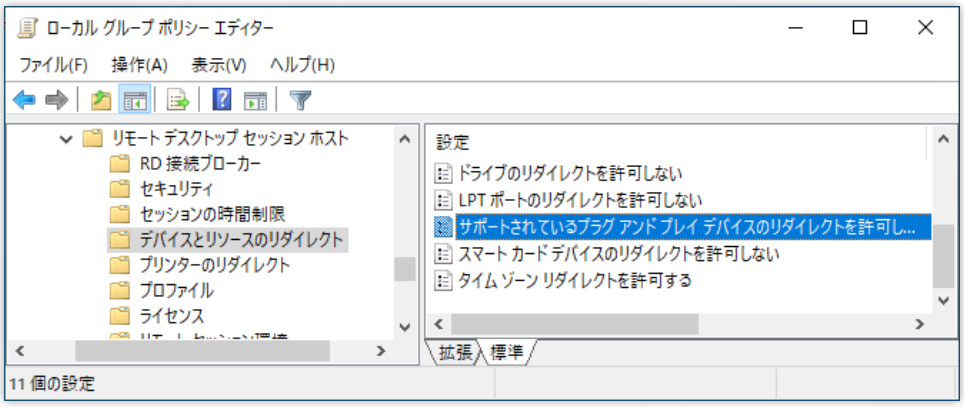
手順 2

下図のようなローカルグループポリシーエディターが表示されます。ウィンドウの左ペインで次の順にクリックして開きます。[**コンピューターの構成**] > [**管理用テンプレート**] > [**Windows コンポーネント**] > [**リモート デスクトップ サービス**] > [**リモートデスクトップ セッション ホスト**] > [**デバイスとリソースのリダイレクト**]



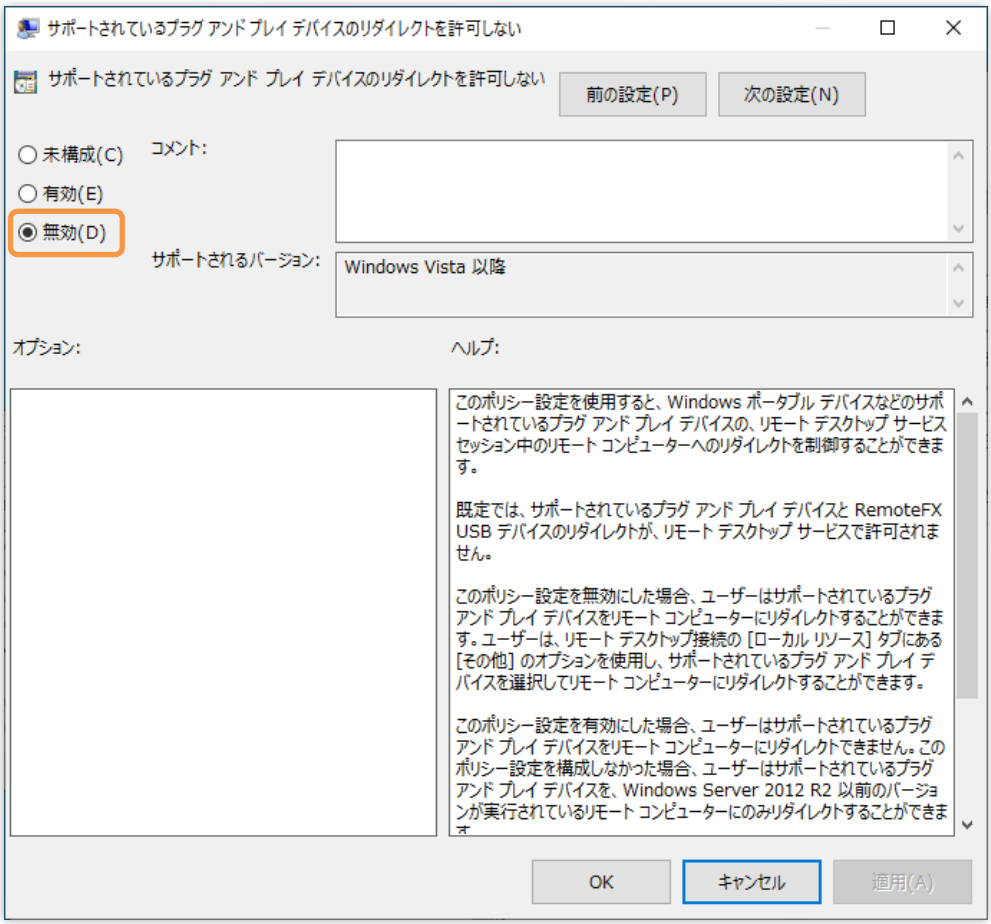
手順 3

下図のウィンドウで[**サポートされているプラグ アンド プレイ デバイスのリダイレクトを許可しない**]をダブルクリックします。



手順 4

次のウィンドウが表示されます。ここで[**無効**]にチェックを入れ、[**OK**]をクリックします。



10.3.2 接続元 PC を設定する

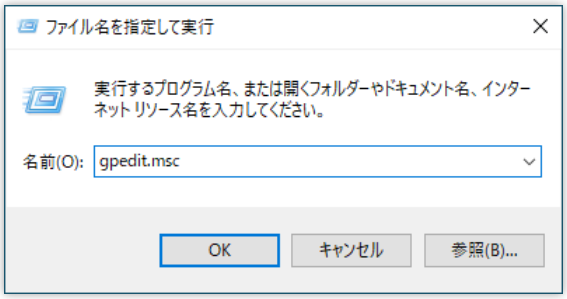
接続元 PC で次の手順を踏みます。

1. ローカルグループポリシーエディターを起動します。
2. 左のペインで以下をクリックして開きます。
   [**コンピューターの構成**] > [**管理用テンプレート**] > [**Windows コンポーネント**] > [**リモート デスクトップ サービス**] > [**リモートデスクトップ接続のクライアント**] > [**RemoteFX USB デバイス リダイレクト**]
3. [**サポートされている他の RemoteFX USB デバイスの、このコンピューターからの RDP リダイレクトを許可する**]をダブルクリックします。
4. [**有効**]にチェックを付け、RemoteFX USB リダイレクトアクセス権に[**管理者とユーザー**]を選択し、[**OK**]をクリックします。
5. Windows を再起動します。

この手順をスクリーンショットとともに説明します。

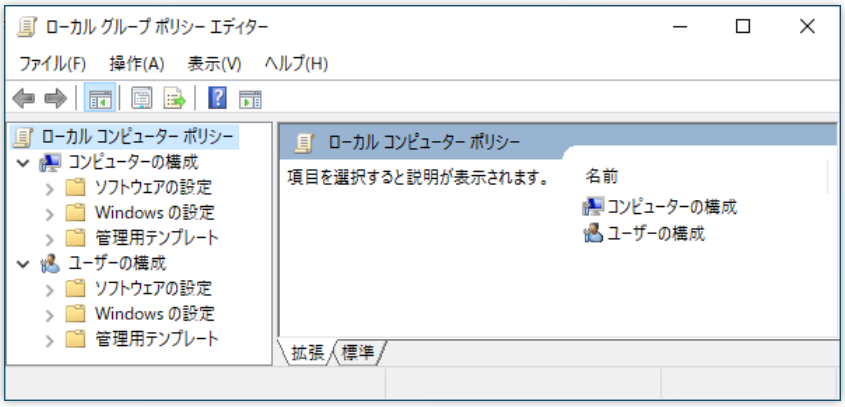
手順 1

スタートボタンを右クリックし（または Windows キー+X を押します）、[**ファイルを指定して実行**]を選択します。以下のウィンドウで gpedit.msc と入力して[**OK**]をクリックします。



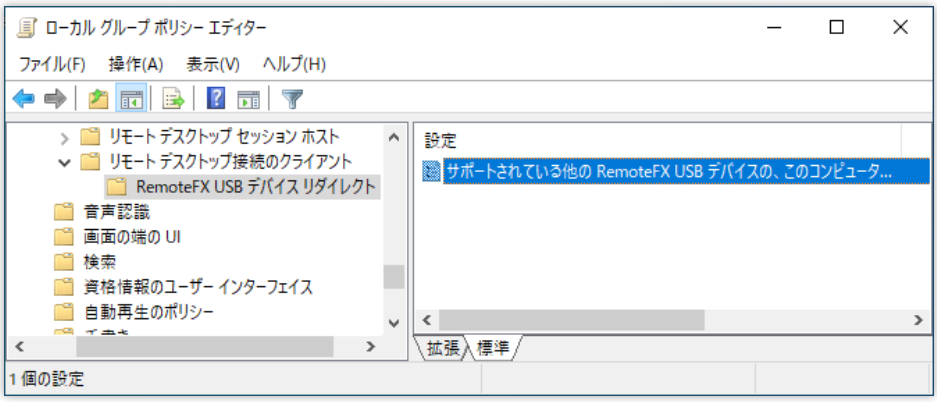
手順 2

ウィンドウの左ペインで次の順にクリックして開きます。[**コンピューターの構成**] > [**管理用テンプレート**] > [**Windows コンポーネント**] > [**リモート デスクトップ サービス**] > [**リモートデスクトップ接続のクライアント**] > [**RemoteFX USB デバイス リダイレクト**]



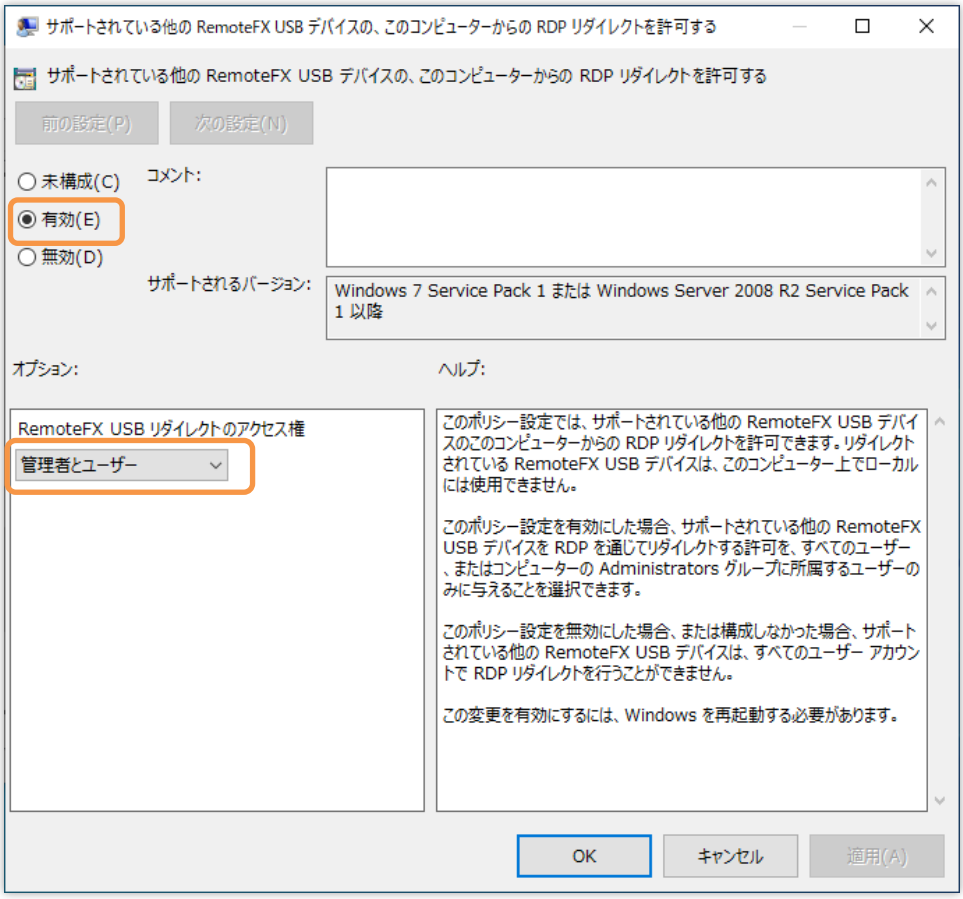
手順 3

次のようなウィンドウになります。ここで[**サポートされている他の RemoteFX USB デバイスの、このコンピューターからの RDP リダイレクトを許可する**]をダブルクリックします。



手順 4

次のウィンドウが表示されます。ここで[**有効**]にチェックを付け、RemoteFX USB リダイレクトアクセス権に[**管理者とユーザー**]を選択し、[**OK**]をクリックします。



手順 5

Windows を再起動します。

10.3.3 SHALO AUTH のリダイレクトと解除

リモートデスクトップでリモート PC に接続してから全画面モードにします。画面の上部中央に表示される接続バーで、下に示すアイコンをクリックします。

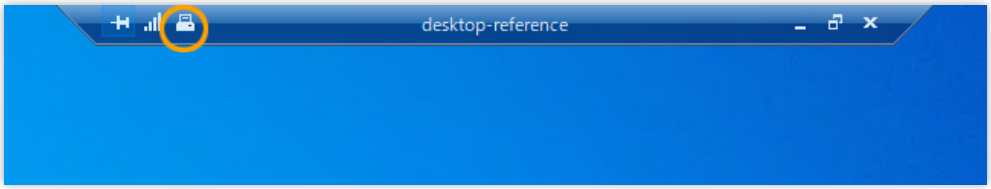


図 77　リモートデスクトップの接続バー

接続バーが表示されない場合は、マウスを上部中央に移動すると表示されます。接続バーのピンをクリックすると接続バーが常に表示されるようになります。

以下のウィンドウが表示されます。リモートデスクトップ接続先で SHALO AUTH を使うには[**SHALO AUTH**]にチェックを付け、[**OK**]をクリックします。リモートデスクトップ接続先から SHALO AUTH を取り外すにはチェックを外します。

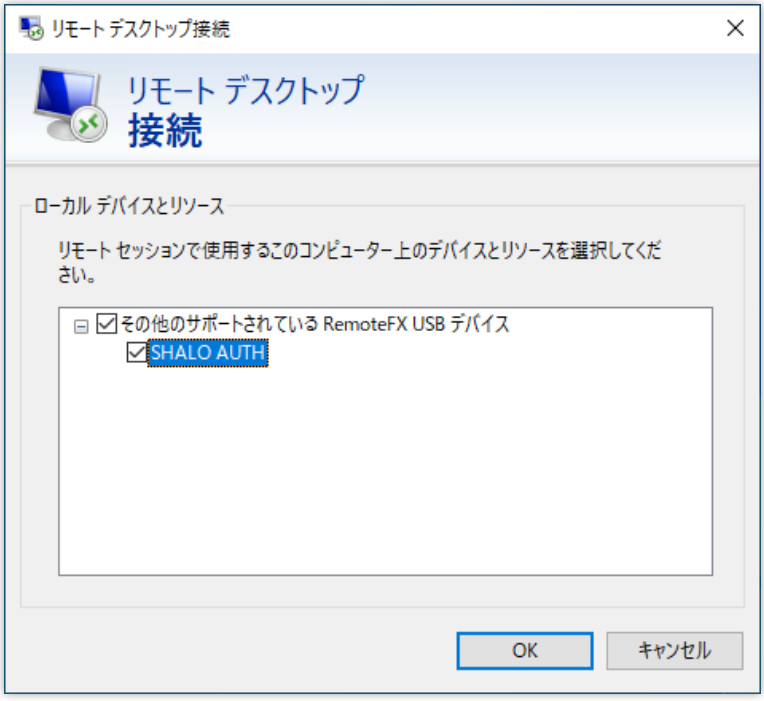


図 78　リモート PC にリダイレクトするデバイス

第 11 章
よくある質問

この章では、SHALO AUTH を使っていて寄せられる質問に答えます。

この章のトピック

1. SHALO Keyring を使わずに SSH 公開鍵を読み出すには？
2. SHALO Keyring を使わずに鍵を作るには？
3. .pfx/.p12/DER 形式の鍵を取り込むには？
4. SHALO AUTH の利用に制限のある OpenSSH は？
5. 症状別トラブルシューティング

11.1 SHALO Keyring を使わずに SSH 公開鍵を読み出すには？

OpenSSH で SHALO AUTH の SSH 公開鍵を読み取ることができます。方法は次の 2 通りです。

- PKCS #11 モジュールを使う
- 認証エージェントを使う

PKCS #11 モジュールを使う場合

ssh-keygen に-D オプションで PKCS #11 モジュールのファイルパスを指定します。

```
ssh-keygen -D pkcs11file
```

PKCS #11 モジュールの名前は環境ごとに異なります。モジュールのファイル名は第 3 章で確認してください。

PKCS #11 モジュールを他のアプリケーションで使用しているか、認証エージェントに登録している場合、ssh-keygen コマンドは失敗します。

以下の例では、SHALO AUTH にあるすべての公開鍵を出力し、次に testkey2 の公開鍵だけファイル key.pub に保存します。

```
$ ssh-keygen -D $SLPKCS11FILE↵
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBA
3/YCyF+KOni2K0nLT625u5teJ8hAubFhr+2LYkBGbADxcNQm4fgpHi+U4nqIddJ1OVl+asi5u
I0BZAK6Nq+qI= testkey1
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBF
8QnCuzFzd2lyn3AEmfLbLjnJZLxdlNdw9F3GZyEK9XROEUL/m6FAY1W4WPnDbWVnOtoBj3DEE
zb1774UHuBEg= testkey2

$ ssh-keygen -D $SLPKCS11FILE | grep testkey2 > key.pub↵
```

認証エージェントを使う場合

shalo-add で SHALO AUTH を OpenSSH の認証エージェントに登録している場合、ssh-add コマンドに-L オプションを付けると認証エージェントが管理する SSH 公開鍵を読み出せます。

```
$ ssh-add -L↵
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBA
3/YCyF+KOni2K0nLT625u5teJ8hAubFhr+2LYkBGbADxcNQm4fgpHi+U4nqIddJ1OVl+asi5u
I0BZAK6Nq+qI= testkey1
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBF
8QnCuzFzd2lyn3AEmfLbLjnJZLxdlNdw9F3GZyEK9XROEUL/m6FAY1W4WPnDbWVnOtoBj3DEE
zb1774UHuBEg= testkey2
```

11.2 SHALO Keyring を使わずに鍵を作るには？

プライベート鍵をファイルとして保管する場合は SHALO Keyring を使わずに鍵を作る必要があります。この節では以下の 3 つのソフトウェアを使った方法を紹介します。

- OpenSSH
- PuTTY
- OpenSSL

11.2.1 OpenSSH を使う

OpenSSH の `ssh-keygen` コマンドは、SSH 向けに公開鍵暗号 RSA や ECDSA のプライベート鍵・公開鍵の鍵ペアを作成できます。

`ssh-keygen` コマンドの主なオプションは以下の通りです。

オプション	説明
`-t` *暗号種別*	暗号種別には `rsa` または `ecdsa` を指定します。
`-b` *ビット長*	ビット長は、RSA の場合は鍵長です。ECDSA の場合は楕円曲線名の P-に続く数字（256 / 384 / 521）です。
`-C` *コメント*	SSH 公開鍵に付けるコメント文字列です。

各暗号方式の鍵を作るためのコマンドは以下の通りです。

作成する鍵	コマンド	備考
RSA 鍵	`ssh-keygen -t rsa -b` *長さ*	*長さ*は 2048～4096 を指定します。
P-256 の ECDSA 鍵	`ssh-keygen -t ecdsa -b 256`	
P-384 の ECDSA 鍵	`ssh-keygen -t ecdsa -b 384`	
P-521 の ECDSA 鍵	`ssh-keygen -t ecdsa -b 521`	

作成手順は以下の通りです。

1. ターミナル（Windows の場合は CMD・Git Bash・Cygwin など）を開きます。
2. 作成する鍵に応じたオプションを指定して `ssh-keygen` コマンドを実行します。
3. “Enter file in which to save the key”と表示されるので、鍵のファイル名を指定してエンターキーを押します。
4. “Enter passphrase”と表示されるので、鍵ファイルを暗号化して保護するためのパスフレーズを入力してエンターキーを押します。
5. “Enter same passphrase again”と表示されるので、パスフレーズをもう一度入力してエンターキーを押します。

パスフレーズはプライベート鍵のファイル暗号化に使用されます。プライベート鍵を SHALO AUTH にインポートする際、パスフレーズを再度入力する必要があるため、忘れないようにしてください。

次ページに RSA で 4,096 ビットの鍵ペアを作る例を示します。コメントは「test_comment」としています。

```
$ ssh-keygen -t rsa -b 4096 -C test_comment↵
Generating public/private rsa key pair.
Enter file in which to save the key (/home/foo/.ssh/id_rsa): shalo↵
Enter passphrase (empty for no passphrase): ↵
Enter same passphrase again: ↵
Your identification has been saved in shalo.
Your public key has been saved in shalo.pub.
The key fingerprint is:
SHA256:ss3DI0VU54cWl4Hp8eOw41r0zeOsyQrCT3vrOdWvhz4 test_comment
The key's randomart image is:
+---[RSA 4096]----+
|        ... oooo |
|       .   o++.  |
|        .  .+o.  |
|       .   .o.o  |
|      . S   .+.. |
|      .B   .oo.+ |
|      oo*o .o..o+|
|       .+oo++ E +|
|         o+*+=+* |
+----[SHA256]-----+
```

入力した名前を持つファイル（例では shalo）にプライベート鍵が保存されます。公開鍵は入力文字列に“.pub”を付けたファイル名（例では shalo.pub）で保存されます。

11.2.2 PuTTY を使う

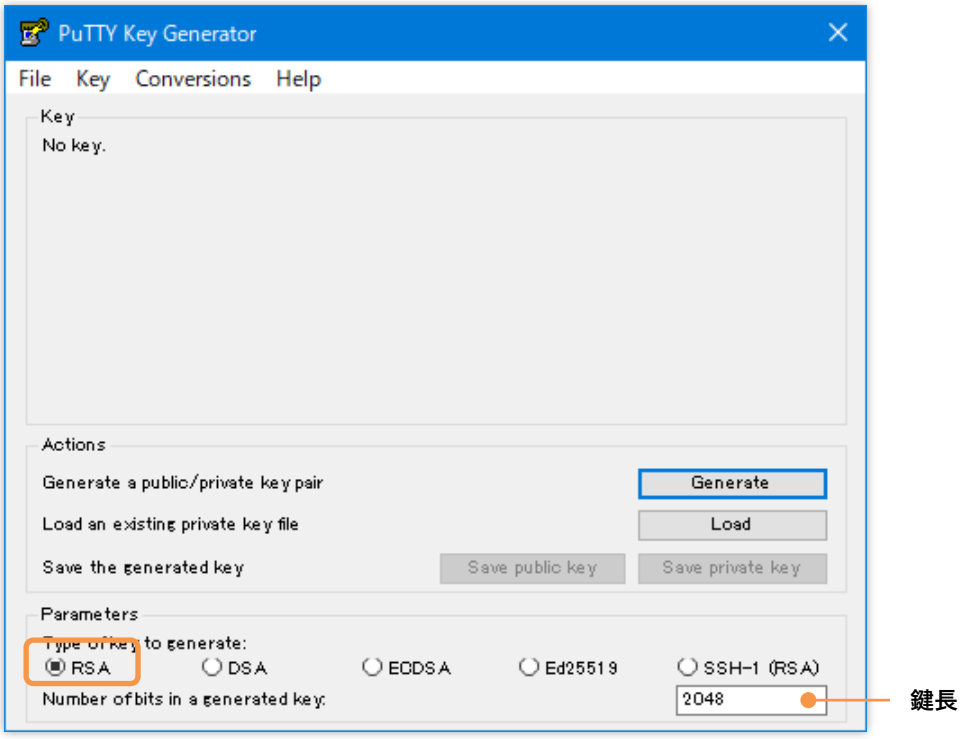
PuTTY 付属の puttygen は GUI 操作で SSH 向けの鍵を作成できます。鍵の作成は以下の手順で行います。

1. puttygen を起動します。
2. 生成する鍵を指定します。
3. [**Generate**]をクリックします。
4. プログレスバーが右端に達するまで PuTTY Key Generator のウィンドウ内でマウスカーソルを動かします。
5. [**Key comment**]にコメントを入力し、[**Key passphrase**]と[**Confirm passphrase**]にパスフレーズを入力します。
6. [**Save private key**]をクリックしてプライベート鍵をファイルに保存します。
7. [**Save public key**]をクリックして公開鍵をファイルに保存します。

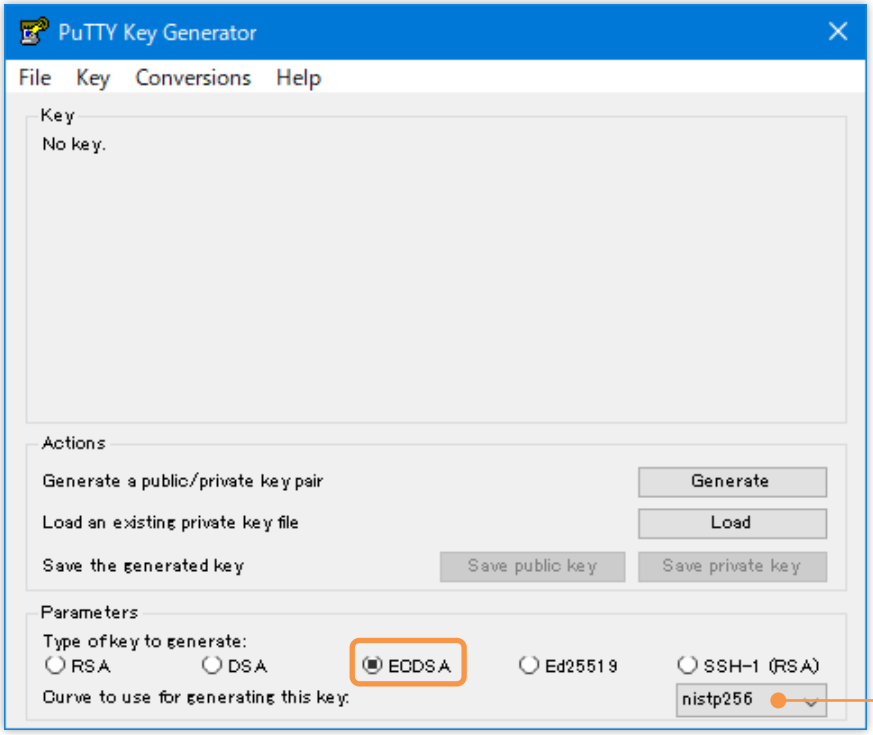
この手順をスクリーンショットとともに説明します。

手順 1～2

puttygen を起動すると以下のウィンドウが表示されます。RSA 鍵を生成する場合は、[**RSA**]にチェックを入れ、ウィンドウ下部の入力フィールドに鍵長を入力します。ECDSA 鍵を生成する場合は、[**ECDSA**]にチェックを入れます。



[**ECDSA**]にチェックを入れた場合、次の図のように曲線名を指定できるようになります。曲線名で[**nistp256**]は P-256、[**nistp384**]は P-384、[**nistp521**]は P-521 を意味します。

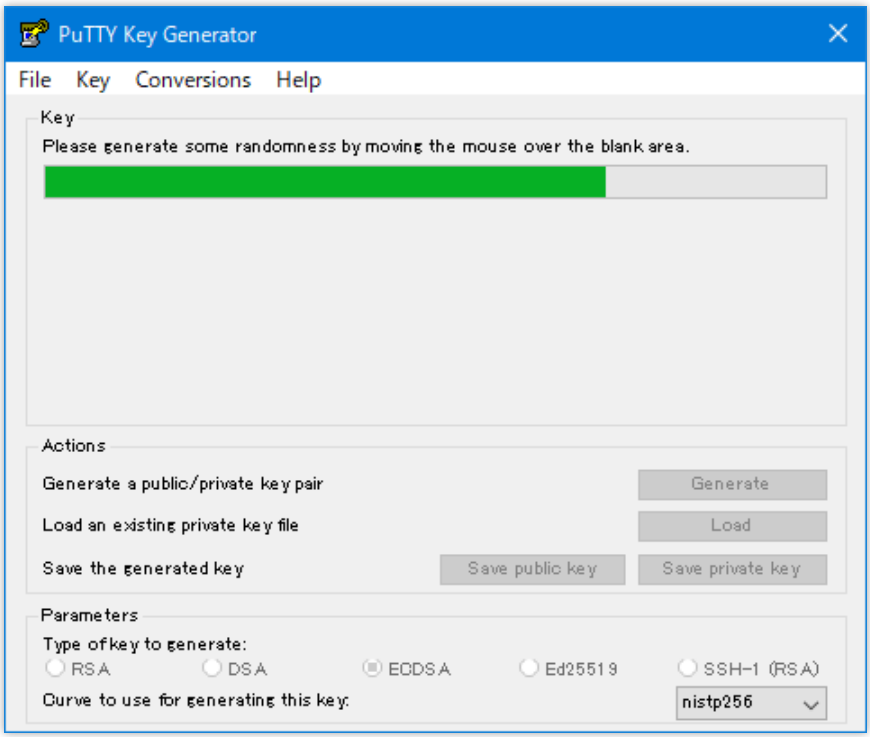


手順 3

[**Generate**]をクリックします。

手順 4

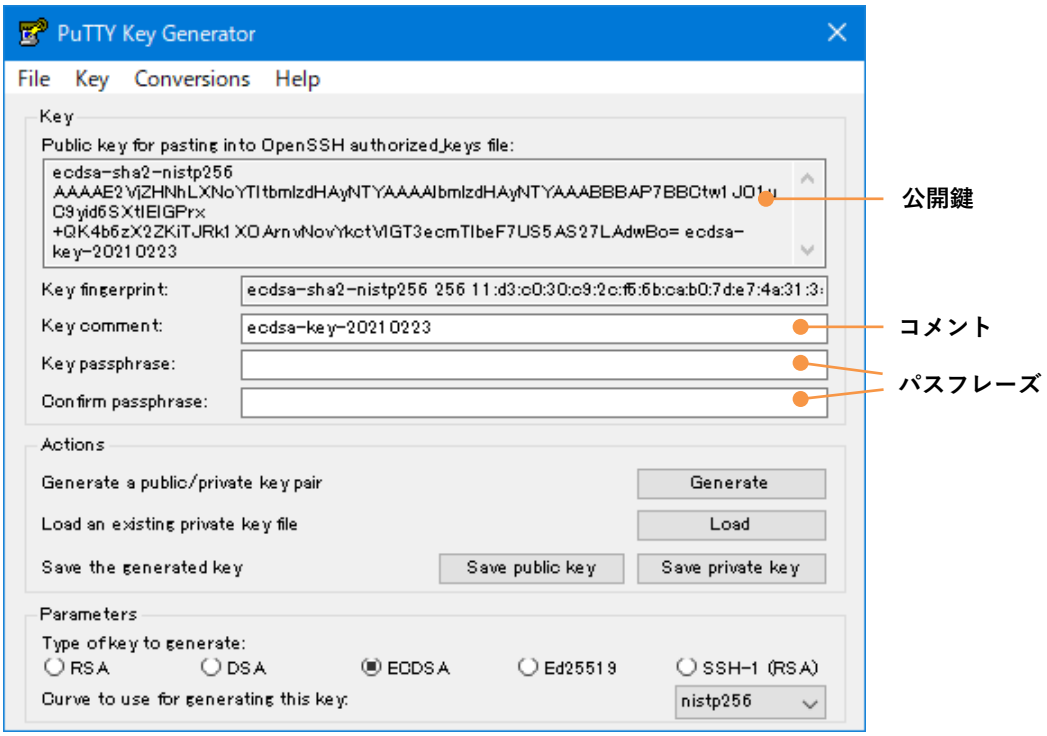
以下のようにプログレスバーが右端に達するまでウィンドウ内でマウスカーソルを動かします。



手順 5

鍵が生成されるとウィンドウに以下のように表示されます。[**Key comment**]にコメントを入力し、[**Key passphrase**]と[**Confirm passphrase**]にパスフレーズを入力します。

パスフレーズはプライベート鍵のファイル暗号化に使用されます。プライベート鍵を SHALO AUTH に取り込む際にパスフレーズを入力するため、忘れないようにしてください。



手順 6

[**Save private key**]をクリックしてプライベート鍵をファイルに保存します。

手順 7

[**Save public key**]をクリックして公開鍵をファイルに保存します。

公開鍵はウィンドウ上部のエディットフィールドにも表示されています。これをコピーして使用することもできます。

11.2.3 OpenSSL を使う

OpenSSL は、SSH 向けに限らず様々なプライベート鍵・公開鍵の鍵ペアを作成できます。鍵は PEM 形式で出力されます。

RSA の場合

`openssl genrsa` コマンドを使用して RSA 鍵を作成します。書式は次の通りです。

```
openssl genrsa -out 出力ファイル 鍵のビット長
```

鍵長 4,096 ビットの鍵を作ってファイル rsakey.pem に保存する例は次の通りです。

```
$ openssl genrsa -out rsakey.pem 4096↵
Generating RSA private key, 4096 bit long modulus (2 primes)
.....................................................................
.....................................++++
.............................................++++
e is 65537 (0x010001)
```

ECDSA の場合

`openssl ecparam` コマンドを使用して ECDSA 鍵を作成します。書式は次の通りです。

```
openssl ecparam -genkey -name 曲線名 -out 出力ファイル
```

OpenSSL が対応する曲線名は次のコマンドで確認できます。環境によっては一部の曲線がサポートされていない場合があります。

```
openssl ecparam -list_curves
```

SHALO AUTH がサポートする次の曲線は OpenSSL で異なる名称を使っています。

- secp192r1 (P-192) **prime192v1**
- secp256r1 (P-256) **prime256v1**

secp256r1 (P-256)の鍵を作ってファイル eckey.pem に保存する例は次の通りです。

```
$ openssl ecparam -genkey -name prime256v1 -out eckey.pem↵
```

11.3 .pfx/.p12/DER 形式の鍵を取り込むには？

SHALO Keyring は PKCS #12 形式（拡張子が pfx または p12）や DER 形式の鍵に対応していません。OpenSSL を使ってこれらを PEM 形式に変換してから SHALO Keyring で取り込みます。

この節では、OpenSSL を使用して PEM 形式に変換する方法を説明します。

pfx または p12 形式

プライベート鍵を PEM 形式で保存するには次のようにします。PEM ファイルを暗号化しない場合は、さらに-nodes オプションを付けます。

```
openssl pkcs12 -in 入力ファイル -nocerts -out 出力ファイル
```

server.pfx ファイルのプライベート鍵を key.pem として保存するには次のようにします。

```
$ openssl pkcs12 -in server.pfx -nocerts -out key.pem↵
Enter Import Password: 入力ファイルのパスワードを入力↵
Enter PEM pass phrase: 出力ファイルのパスワードを入力↵
Verifying - Enter PEM pass phrase: 出力ファイルのパスワードをもう一度入力↵
```

DER 形式の RSA プライベート鍵

RSA プライベート鍵を DER 形式から PEM 形式にするには次のようにします。

```
openssl rsa -inform DER -in 入力ファイル -outform PEM -out 出力ファイル
```

DER 形式の ECDSA プライベート鍵

ECDSA プライベート鍵を DER 形式から PEM 形式にするには次のようにします。

```
openssl ecparam -inform DER -in 入力ファイル -outform PEM -out 出力ファイル
```

11.4 SHALO AUTH の利用に制限のある OpenSSH は？

OpenSSH の構成は次のようにして確認できます。

```
$ ssh -V↵
OpenSSH_8.5p1, OpenSSL 1.1.1k  25 Mar 2021
```

この例では OpenSSH のバージョンが 8.5p1 で、暗号ライブラリに OpenSSL 1.1.1k を使用していることを示します。

SHALO AUTH の RSA 鍵と ECDSA 鍵を使用できる OpenSSH の構成は以下の通りです。

OpenSSH の構成	RSA 鍵	ECDSA 鍵
OpenSSH 5.2p1 以前	✗	✗
OpenSSH 5.3p1～OpenSSH 7.9p1	✓	✗
OpenSSH 8.0p1 以降＋OpenSSH 1.0	✓	✗
OpenSSH 8.0p1 以降＋OpenSSH 1.1	✓	✓
OpenSSH 8.0p1 以降＋LibreSSL 2.9 以前	✓	✗
OpenSSH 8.0p1 以降＋LibreSSL 3.0 以降	✓	✓

OpenSSH は ECDSA 鍵で P-256 / P-384 / P-521 のみサポートします。

環境別 OpenSSH のバージョン

各環境で標準パッケージとなっている OpenSSH のバージョンは以下の通りです。背景がオレンジ色の環境では SHALO AUTH の ECDSA 鍵を使用できません。

環境	構成（ssh -V による出力結果）
Git for Windows 2.21.0 以前	OpenSSH_7.9p1, OpenSSL 1.1.1a 以前の組み合わせ
Git for Windows 2.22.0 以降	OpenSSH_8.0p1, OpenSSL 1.1.1c 以降の組み合わせ
Git for Windows 2.31.1	OpenSSH_8.5p1, OpenSSL 1.1.1k　25 Mar 2021
Cygwin 3.2.0	OpenSSH_8.5p1, OpenSSL 1.1.1f　31 Mar 2020
macOS BigSur	OpenSSH_8.1p1, LibreSSL 2.7.3
Ubuntu 18.04.5 LTS	OpenSSH_7.6p1 Ubuntu-4ubuntu0.3, OpenSSL 1.0.2n　7 Dec 2017
Ubuntu 20.04.2 LTS	OpenSSH_8.2p1 Ubuntu-4ubuntu0.2, OpenSSL 1.1.1f　31 Mar 2020
CentOS 7.9-2009	OpenSSH_7.4p1, OpenSSL 1.0.2k-fips　26 Jan 2017
CentOS 8.3.2011	OpenSSH_8.0p1, OpenSSL 1.1.1g FIPS　21 Apr 2020
Fedora 33-1.2	OpenSSH_8.4p1, OpenSSL 1.1.1g FIPS　21 Apr 2020
Fedora 34-1.2	OpenSSH_8.5p1, OpenSSL 1.1.1k　FIPS 25 Mar 2021

11.5 症状別トラブルシューティング

11.5.1 ユーザーPIN がロックされた

ユーザーPIN は続けて 5 回間違えるとロックされます。SHALO Smith を使ってユーザーPIN を再設定（5.4 節）してください。

11.5.2 管理 PIN がロックされた

管理 PIN は続けて 5 回間違えるとロックされます。管理 PIN だけを復旧する方法はありません。

管理 PIN がロックされても SHALO AUTH 内の鍵は削除されません。
ユーザーPIN がロックされていなければ引き続き鍵を使用できます。

管理 PIN を復旧するには SHALO AUTH を購入時の状態に戻し（5.3 節）、再度セットアップします。

SHALO AUTH を購入時の状態に戻すと、SHALO AUTH が持つすべてのデータは削除され、また U2F セキュリティキーとして以前行った登録も無効化されます。

11.5.3 SHALO AUTH を PC に接続するとライトが点滅し続ける

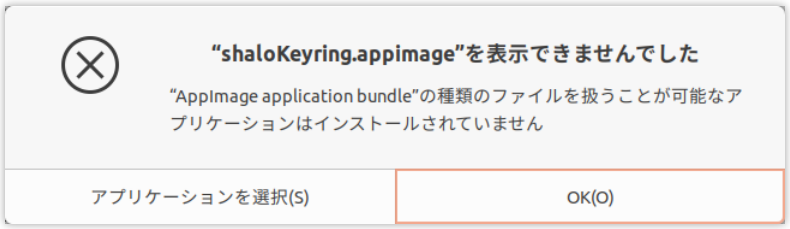
SHALO AUTH が修復不能な異常を検知すると、１秒間に 1～3 回の頻度でライトを点滅し続けます。このようになった SHALO AUTH を使用することはできません。以下の手続きに従って認証情報を廃棄してください。

- SHALO AUTH を U2F セキュリティキーとして使用しているウェブサービスで、SHALO AUTH の登録を解除します。
- SHALO AUTH の公開鍵をサーバーに登録している場合は、サーバーから削除します。

11.5.4 Linux で shaloKeyring.appimage/shaloSmith.appimage が起動しない（1）

症状

Linux の GUI から shaloKeyring.appimage か shaloSmith.appimage を起動しようとすると、以下のウィンドウが表示されます。



原因

.appimage ファイルの実行可能権限が不足しています。

対処方法

.appimage ファイルに実行可能権限を追加します。問題のあった.appimage ファイルを右クリックします。コンテキストメニューから［**プロパティ**］を選択し、以下のウィンドウで［**プログラムとして実行可能**］にチェックを入れます。

11.5.5 Linux で shaloKeyring.appimage/shaloSmith.appimage が起動しない（2）

症状

Linux で shaloKeyring.appimage か shaloSmith.appimage を起動しようとしても、ウィンドウ表示されません。

原因

環境が前提条件を満たしていません。

対処方法

shaloKeyring.appimage / shaloSmith.appimage はターミナルから起動されると、エラー発生時にターミナルにエラーメッセージを出力します。このエラーメッセージから問題の原因を特定できます。

例は次の通りです。

```
$ ./shaloKeyring.appimage
dlopen(): error loading libfuse.so.2

AppImages require FUSE to run.
You might still be able to extract the contents of this AppImage
if you run it with the --appimage-extract option.
See https://github.com/AppImage/AppImageKit/wiki/FUSE
for more information
```

上の例では libfuse2 ライブラリのファイル libfuse.so.2 が不足しています。これは 3.5.2 節を参照して libfuse2 をインストールすると問題を解決できます。

11.5.6 SHALO Keyring/Smith が SHALO AUTH を認識しない

症状

SHALO AUTH を PC に接続しているにもかかわらず、SHALO Keyring/Smith からデバイスが見つからない。

原因

SHALO Keyring/Smith が SHALO AUTH にアクセスする際に、他のソフトウェアが SHALO AUTH を使用中です。

対処方法

1 つの SHALO AUTH を複数のソフトウェアから同時に使用しないようにしてください。以下のソフトウェアが SHALO AUTH を使用します。

1. 認証エージェント
2. SHALO Keyring/Smith
3. Adobe® Acrobat®/Adobe® Acrobat® Reader®（SHALO AUTH にアクセスした場合）

SHALO Keyring と SHALO Smith は同時に起動しないようにします。

11.5.7 ssh -I が「C_GetTokenInfo ～ failed: ??」で失敗する

症状

ssh -I で PKCS#11 モジュールを指定して SSH サーバーに接続しようとすると以下のように失敗する。

```
$ ssh -I pkcs11file username@hostname
C_GetTokenInfo for provider pkcs11file slot 0 failed: 48
username@hostname: Permission denied (publickey).
```

原因

PKCS#11 モジュールが SHALO AUTH にアクセスする際に、他のソフトウェアが SHALO AUTH を使用しています。

対処方法

1 つの SHALO AUTH を複数のソフトウェアから同時に使用しないようにしてください。以下のソフトウェアが SHALO AUTH を使用します。

1. 認証エージェント
2. SHALO Keyring/Smith
3. Adobe® Acrobat®/Adobe® Acrobat® Reader®（SHALO AUTH にアクセスした場合）

11.5.8 ssh -I が「C_GetAttributeValue failed: 18」と出力する

症状

ssh -I で PKCS#11 モジュールを指定して SSH サーバーに接続しようとすると以下のように出力します。

```
$ ssh -I pkcs11file username@hostname
C_GetAttributeValue failed: 18
username@hostname: Permission denied (publickey).
```

原因

「C_GetAttributeValue failed: 18」は OpenSSH が対応していない鍵を検出すると出力されます。SSH サーバーに接続成功した場合でも出力される場合があります。

対処方法

このメッセージを出力して SSH サーバーに接続に失敗する場合は、利用環境の OpenSSH が対応する鍵を使うようにします。OpenSSH がサポートする鍵の種別は 11.4 節を参照してください。

11.5.9 ssh-agent を使って SSH サーバーにログインできない

症状

ssh-agent に `shalo-add` で SHALO AUTH を登録した後に、SSH サーバーに接続しようとすると以下のように失敗します。

```
$ ssh username@hostname↵
username@hostname: Permission denied (publickey).
```

原因

ssh サーバーに登録されている公開鍵が ssh-agent に読み込まれていません。ssh-agent に読み込まれている公開鍵は `ssh-add -L` で確認できます。

```
$ ssh-add -L
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBA
3/YCyF+KOni2K0nLT625u5teJ8hAubFhr+2LYkBGbADxcNQm4fgpHi+U4nqIddJ1OVl+asi5u
I0BZAK6Nq+qI= testkey1
```

対処方法

SHALO AUTH の目的の鍵が ssh-agent に読み込まれていない場合は、SHALO Keyring で鍵の種別を確認し、それが OpenSSH でサポートされているものかを確認してください。OpenSSH がサポートする鍵の種別は 11.4 節を参照してください。

ssh-agent に正しく読み込まれている場合は、SSH サーバーに公開鍵が登録されているか確認してください。

11.5.10 ssh-agent に SHALO AUTH を登録できない

症状

`shalo-add` を実行してユーザーPIN を入力すると、以下のように表示されます。

```
Could not add card "PKCS#11 ライブラリのパス": agent refused operation
```

原因

以下の原因が考えられます。

- すでに PKCS#11 モジュールが登録されている。
- 他のアプリケーションで PKCS #11 モジュールを使っている。
- ユーザーPIN が違う。
- ユーザーPIN がロックされている。
- SHALO AUTH に鍵が入っていない、または格納されている鍵に対応していない。
- PKCS #11 モジュールのパスが ssh-agent に許可されていない。

対処方法

原因を特定するために以下を実行します。

1. `shalo-remove` を実行してから再度 `shalo-add` を実行します。
2. PKCS #11 モジュールを使用しているアプリケーションを終了します。Acrobat®に PKCS #11 モジュールを登録している場合は Acrobat®を終了します。
3. SHALO Keyring か SHALO Smith で SHALO AUTH の状態を確認します。（4.2 節、5.1 節）
4. `ssh` の`-I` オプションを使ってサーバーに接続してエラー出力を確認します。（10.1 節）

原因が見つからない場合、第 3 章の通りに PKCS #11 モジュールを正しいディレクトリにインストールしているか確認します。

OpenSSH 7.9p1 以降では ssh-agent の動作状態をコンソールに出力させてエラー内容を知ることができます。これにはターミナルが 2 つ必要です。

1 つ目のターミナルで以下のようにデバッグモードで ssh-agent を起動します。ssh-add の実行に応じてこのターミナルに動作内容が出力されます。

```
$ ssh-agent -d > ~/agenttmp↵
```

もう 1 つのターミナルで以下のように SHALO AUTH を登録します。

```
$ source ~/agenttmp > /dev/null↵
$ ssh-add -v -s $SLPKCS11FILE↵
```

PKCS#11 モジュールがホワイトリストに入っていない場合は、次のように出力されます。

```
refusing PKCS#11 add of "PKCS#11 モジュールのファイルパス": provider not whit
elisted
```

SHALO AUTH が見つからない場合は、最後に「returned no slots」と出力されます。これは多くの場合、他のソフトウェアが SHALO AUTH を使用しているために起きます。

```
debug1: provider /usr/lib/libslpkcs11.so: manufacturerID <AXELL CORPORATIO
N> cryptokiVersion 2.40 libraryDescription <AXELL PKCS#11 library> library
Version 1.3
debug1: pkcs11_register_provider: provider /usr/lib/libslpkcs11.so returne
d no slots
```

SHALO AUTH の中に鍵が見つからない場合は、最後に「returned no keys」と出力されます。

```
debug1: provider /usr/lib/libslpkcs11.so: manufacturerID <AXELL CORPORATIO
N> cryptokiVersion 2.40 libraryDescription <AXELL PKCS#11 library> library
Version 1.3
debug1: provider /usr/lib/libslpkcs11.so slot 0: label <デバイスラベル> man
ufacturerID <AXELL CORPORATION> model <SHALO AUTH> serial <> flags 0x40d
debug1: pkcs11_provider_finalize: 0x55fba2e1ddc0 refcount 1 valid 1
debug1: pkcs11_provider_unref: 0x55fba2e1ddc0 refcount 1
debug1: pkcs11_add_provider: provider /usr/lib/libslpkcs11.so returned no
keys
```

ユーザーPIN に関するエラーの場合は場合、途中に「C_Login failed」と出力されます。ユーザーPIN を間違えたか、ユーザーPIN がロックされていることが考えられます。

```
debug1: provider /usr/lib/libslpkcs11.so: manufacturerID <AXELL CORPORATIO
N> cryptokiVersion 2.40 libraryDescription <AXELL PKCS#11 library> library
Version 1.3
debug1: provider /usr/lib/libslpkcs11.so slot 0: label <デバイスラベル> man
ufacturerID <AXELL CORPORATION> model <SHALO AUTH> serial <> flags 0x5040d
C_Login failed: 164
debug1: pkcs11_provider_finalize: 0x55fba2e0fc10 refcount 1 valid 1
debug1: pkcs11_provider_unref: 0x55fba2e0fc10 refcount 1
debug1: pkcs11_add_provider: provider /usr/lib/libslpkcs11.so returned no
keys
```

第 12 章

PKCS #11 モジュール情報

この章では、SHALO AUTH 向け PKCS #11 モジュールの諸仕様を記載します。

この章のトピック

1. サポートされている API
2. サポートされているキータイプ
3. サポートされているメカニズム
4. サポートされている属性

12.1 サポートされている API

鍵生成と鍵 Wrap、オブジェクトのコピー機能はサポートされません。API のサポート状況は以下の通りです。

サポートされている API		サポートされていない API
C_Initialize	C_FindObjectsFinal	C_GetOperationState
C_Finalize	C_EncryptInit	C_SetOperationState
C_GetInfo	C_Encrypt	C_CopyObject
C_GetFunctionList	C_EncryptUpdate	C_GetObjectSize
C_GetSlotList	C_EncryptFinal	C_DigestKey
C_GetSlotInfo	C_DecryptInit	C_SignRecoverInit
C_GetTokenInfo	C_Decrypt	C_SignRecover
C_GetMechanismList	C_DecryptUpdate	C_VerifyRecoverInit
C_GetMechanismInfo	C_DecryptFinal	C_VerifyRecover
C_InitToken	C_DigestInit	C_DigestEncryptUpdate
C_InitPIN	C_Digest	C_DecryptDigestUpdate
C_SetPIN	C_DigestUpdate	C_SignEncryptUpdate
C_OpenSession	C_DigestFinal	C_DecryptVerifyUpdate
C_CloseSession	C_SignInit	C_GenerateKey
C_CloseAllSessions	C_Sign	C_GenerateKeyPair
C_GetSessionInfo	C_SignUpdate	C_WrapKey
C_Login	C_SignFinal	C_UnwrapKey
C_Logout	C_VerifyInit	C_DeriveKey
C_CreateObject	C_Verify	C_GetFunctionStatus
C_DestroyObject	C_VerifyUpdate	C_CancelFunction
C_GetAttributeValue	C_VerifyFinal	C_WaitForSlotEvent
C_SetAttributeValue	C_SeedRandom	
C_FindObjectsInit	C_GenerateRandom	
C_FindObjects		

12.2 サポートされているキータイプ

キータイプ	アルゴリズム	サポート範囲
CKK_RSA	RSA	1,024～4096 ビットの RSA キー
CKK_EC	ECDSA	以下の楕円曲線 secp192r1 (P-192) secp192k1 secp224r1 (P-224) secp224k1 secp256r1 (P-256) secp256k1 secp384r1 (P-384) secp521r1 (P-521)

12.3 サポートされているメカニズム

ダイジェストメカニズム

メカニズム	備考
CKM_SHA_1	Single part と multiple-part を両方サポートします。
CKM_SHA256	Single part と multiple-part を両方サポートします。
CKM_SHA384	Single part と multiple-part を両方サポートします。
CKM_SHA512	Single part と multiple-part を両方サポートします。

RSA メカニズム

メカニズム	Op	MinKey	MaxKey	Encrypt	Decrypt	Sign	Verify
CKM_RSA_X_509	Single	1024	4096	✓	✓	✓	✓
CKM_RSA_PKCS	Single	1024	4096	✓	✓	✓	✓
CKM_SHA1_RSA_PKCS	Both	1024	4096			✓	✓
CKM_SHA256_RSA_PKCS	Both	1024	4096			✓	✓
CKM_SHA384_RSA_PKCS	Both	1024	4096			✓	✓
CKM_SHA512_RSA_PKCS	Both	1024	4096			✓	✓
CKM_RSA_PKCS_OAEP	Single	1024	4096	✓	✓		
CKM_RSA_PKCS_PSS1	Single	1024	4096			✓	✓[1]
CKM_SHA1_RSA_PKCS_PSS	Both	1024	4096			✓	✓[1]
CKM_SHA256_RSA_PKCS_PSS	Both	1024	4096			✓	✓[1]
CKM_SHA384_RSA_PKCS_PSS	Both	1024	4096			✓	✓[1]
CKM_SHA512_RSA_PKCS_PSS	Both	1024	4096			✓	✓[1]

Op: Single は Single part オペレーションのみサポートします。

Op: Both は Single part と Multiple-part オペレーションを両方サポートします。

1: 鍵オブジェクトの CKA_VERIFY 属性と CKA_ENCRYPT 属性が CK_TRUE でなければなりません。

CK_RSA_PKCS_OAEP_PARAMS 構造体および CK_RSA_PKCS_PSS_PARAMS 構造体の mgf メンバは、CKG_MGF1_SHA1、CKG_MGF1_SHA256、CKG_MGF1_SHA384、CKG_MGF1_SHA512 を指定できます。hashAlg メンバは mgf メンバに影響を受けずに自由にダイジェストメカニズムを指定できます。

EC メカニズム

メカニズム	Op	MinKey	MaxKey	Encrypt	Decrypt	Sign	Verify
CKM_ECDSA	Both	192	521			✓	✓
CKM_ECDSA_SHA1	Both	192	521			✓	✓
CKM_ECDSA_SHA256	Both	192	521			✓	✓
CKM_ECDSA_SHA384	Both	192	521			✓	✓
CKM_ECDSA_SHA12	Both	192	521			✓	✓

Op: Both は Single part と Multiple-part オペレーションを両方サポートします。

12.4 サポートされている属性

すべてのオブジェクトが保持する属性

属性	デフォルト値	備考
CKA_TOKEN	False	ハードウェア機能でサポートされます
CKA_PRIVATE	False	ハードウェア機能でサポートされます
CKA_MODIFIABLE	True	ハードウェア機能でサポートされます
CKA_COPYABLE	True	C_CopyObject()はサポートされません
CKA_DESTROYABLE	True	ハードウェア機能でサポートされます

RSA プライベート鍵オブジェクトが追加でサポートする属性

属性	必須	備考
CKA_CLASS	✓	常に CKO_PRIVATE_KEY です
CKA_KEY_TYPE	✓	常に CKK_RSA です
CKA_LABEL		
CKA_ID		
CKA_ALLOWED_MECHANISMS		
CKA_SUBJECT		
CKA_MODULUS	✓	
CKA_PUBLIC_EXPONENT	✓	
CKA_PRIVATE_EXPONENT	✓	CKA_SENSITIVE による保護対象です
CKA_PRIME_1	✓	CKA_SENSITIVE による保護対象です
CKA_PRIME_2	✓	CKA_SENSITIVE による保護対象です
CKA_EXPONENT_1	✓	CKA_SENSITIVE による保護対象です
CKA_EXPONENT_2	✓	CKA_SENSITIVE による保護対象です
CKA_COEFFICIENT	✓	CKA_SENSITIVE による保護対象です
CKA_SENSITIVE		
CKA_DECRYPT		
CKA_SIGN		

RSA 公開鍵オブジェクトが追加でサポートする属性

属性	必須	備考
CKA_CLASS	✓	常に CKO_PUBLIC_KEY です
CKA_KEY_TYPE	✓	常に CKK_RSA です
CKA_LABEL		
CKA_ID		
CKA_ALLOWED_MECHANISMS		
CKA_SUBJECT		
CKA_MODULUS	✓	
CKA_PUBLIC_EXPONENT	✓	
CKA_ENCRYPT		
CKA_VERIFY		

EC プライベート鍵オブジェクトが追加でサポートする属性

属性	必須	備考
CKA_CLASS	✓	常に CKO_PRIVATE_KEY です
CKA_KEY_TYPE	✓	常に CKK_EC です
CKA_LABEL		
CKA_ID		
CKA_ALLOWED_MECHANISMS		
CKA_SUBJECT		
CKA_EC_PARAMS	✓	
CKA_VALUE	✓	CKA_SENSITIVE による保護対象です
CKA_SENSITIVE		
CKA_SIGN		

EC 公開鍵オブジェクトが追加でサポートする属性

属性	必須	備考
CKA_CLASS	✓	常に CKO_PRIVATE_KEY です
CKA_KEY_TYPE	✓	常に CKK_EC です
CKA_LABEL		
CKA_ID		
CKA_ALLOWED_MECHANISMS		
CKA_SUBJECT		
CKA_EC_PARAMS	✓	
CKA_EC_POINT	✓	
CKA_VERIFY		

公開鍵オブジェクトが追加でサポートする属性

属性	必須	備考
CKA_CLASS	✓	常に CKO_CERTIFICATE です
CKA_CERTIFICATE_TYPE	✓	常に CKC_X_509 です
CKA_LABEL		
CKA_ID		
CKA_SUBJECT		
CKA_VALUE	✓	
CKA_ISSUER		
CKA_SERIAL_NUMBER		

可変長データ型の属性の最大データ長

属性のデータ型が Byte array や string などの可変長でさらに長さが決められていない場合、1 つのオブジェクトとして 8KB に収まる限りデータ長に制限はありません。

ご注意

- 本資料及び本注意書の記載内容は2024年4月現在のものです。
- 本資料の一部又は全部を弊社の許可なく、転載・複写することを堅くお断りします。
- 本資料に記載されている製品（以下「本製品」といいます。）をご利用される際、本資料の内容を正しく守ってお使い下さい。
- 別途お客様と弊社との間で締結した書面による契約又は本製品の売買契約書の関連条項において定める場合を除き、弊社は、本製品及び技術情報に関して、お客様に生じた間接的、結果的、特別又は偶発的な損害（逸失利益、機会の喪失、業務の障害、データの喪失に基づく損害を含みますがこれらに限られません。）を負担いたしません。また弊社は、明示的にも黙示的にも、本製品及び技術情報に関して、一切の保証（機能動作の保証、商品性の保証、特定目的への合致の保証、情報の正確性の保証、第三者の権利の非侵害保証を含みますがこれらに限られません。）をしておりません。
- 本製品の異常や故障による機会損失、二次的損害又は最大絶対規格値を超えてご使用された場合の本製品の故障等に対しましては、弊社はその責を負いかねますのでご了承下さい。
- 記載されております応用例やその定数などの情報につきましては、本製品の標準的な動作や使い方を説明するものです。従いまして、量産設計をされる場合には、外部諸条件を考慮していただきますようお願い致します。また、本製品のお客様の設備等への組み込みは，お客様の責任にて行われますようお願い致します。
- 本資料に記載されております本製品に関する応用例、情報、諸データは、あくまで一例を示すものであり、これらに関する第三者の特許権又は著作権などの知的所有権及びその他の権利に対する弊社の保証を示すものではございません。従いまして(1)上記第三者の知的財産権の侵害の責任又は(2)本製品の使用により発生する責任につきましては、弊社はその責を負いかねますのでご了承下さい。
- 弊社は、日々本製品の品質等の向上に努めておりますが、本製品が故障する可能性を完全に取り除くことはできません。お客様におかれましては、本製品が故障しても、結果的に、人身事故、火災事故、社会的な損害を生じさせないよう、冗長設計、延焼対策設計、過電流防止対策設計、誤動作防止設計などの安全設計をお願い致します。
- 弊社は、本資料を完全なものとすべく努めておりますが、本資料は、あらゆる事象に対応できるものとはなっておりません。本資料を遵守することによって不具合等が発生するおそれがある場合にはあらかじめ弊社宛にご相談いただくなど、お客様自身で適宜な対応をお願い致します。またお客様が本注意書及び本資料を遵守されたとしても、必ずしも弊社が賠償・補償等の責任を負担するわけではないことについてもご理解頂けますようお願い致します。
- 本資料に記載されている製品は、AV機器、OA機器、通信機器、家電製品、アミューズメント機器などの一般的な電子機器への使用を意図しています。(a)直接生命に影響を及ぼす可能性のある機器（生命維持装置などを含みますがこれに限られません）及び(b)極めて高度な信頼性が要求され、その製品の故障や誤作動が多大な損害を発生させる可能性のある機器（輸送機器制御装置、原子力制御、軍事機器などを含みますがこれに限られません）への使用は意図しておらず、また使用することは出来ません。万一、上記機器へのご使用を検討される際は、事前に弊社営業窓口までご相談願います。
- 社名、製品名などは、一般に各社の商標または登録商標です。

株式会社アクセル
〒101-8973 東京都千代田区外神田4－14－1
秋葉原UDX 南ウイング10階
TEL 03-5298-1670 FAX 03-5298-1671
https://www.axell.co.jp/