Solaris スマートカードの管理

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303
U.S.A. 650-960-1300

Part Number 806-3010-10
2000 年 3 月

原典: *Solaris Smart Cards Administration Guide*

Part No: 806-1646-10

Revision A

Adobe PostScript

目次

はじめに

Solaris™ スマートカードの機能を利用すると、スマートカードを使って Solaris 8 デスクトップ環境などのアプリケーションに安全にログインできます。このマニュアルでは、スマートカードを使って認証を行うためのホストとスマートカードの設定方法について説明します。また、設定後のカードの使い方についても説明します。

対象読者

このマニュアルは、次の 2 種類の読者を対象としています。

- Solaris オペレーティング環境またはその他の形式の UNIX オペレーティングシステムに関する知識を持っているネットワーク管理者
- スマートカードを使って Solaris マシンにログインする必要があるユーザー

ネットワーク管理者向けの情報

このマニュアルの最初の 4 章は、ネットワーク管理者、セキュリティ管理者、システム管理者を対象としています。ここでの説明を理解するためには、認証やネットワークセキュリティの概念について十分に理解している必要があります。

第 1 章では、スマートカードソフトウェアの概要と、その概念を示します。第 2 ～ 4 章では、コマンド行からスマートカードを管理するときに必要な作業と、その主な手順について説明します。

スマートカードユーザー向けの情報

スマートカードを使ってマシンに安全にログインする方法については、第 5 章を参照してください。スマートカードの概念を簡単に把握したい場合は、14ページの「スマートカードによるログイン」を参照してください。

お読みになる前に

このマニュアルの手順を実行する前に、スマートカードを使用するドメイン内のすべてのホストに Solaris 8 オペレーティングシステムをインストールしておく必要があります。

内容の紹介

このマニュアルの内容は次のとおりです。

第 1 章では、スマートカード認証テクノロジを紹介し、スマートカードの動作について説明します。

第 2 章では、スマートカードによる認証をサポートするためのホストの設定作業について説明します。

第 3 章では、ユーザー用のスマートカードの設定に必要な作業について説明します。

第 4 章では、ホスト、リーダー、スマートカードの保守作業について説明します。

第 5 章では、スマートカードの使用方法について説明します。

注 - 第 5 章は、スマートカードのユーザーを対象としています。第 1 章～第 4 章は、システムまたは管理者とセキュリティ管理者を対象としています。

Sun のマニュアルの注文方法

専門書を扱うインターネットの書店 Fatbrain.com から、米国 Sun Microsystems™, Inc. (以降、Sun™ とします) のマニュアルをご注文いただけます。

マニュアルのリストと注文方法については、http://www1.fatbrain.com/documentation/sun の Sun Documentation Center をご覧ください。

Sun のオンラインマニュアル

http://docs.sun.com では、Sun が提供しているオンラインマニュアルを参照することができます。マニュアルのタイトルや特定の主題などをキーワードとして、検索をおこなうこともできます。

UNIX コマンドの使用方法

Solaris スマートカードは、他の Solaris 管理ツールや UNIX コマンド、プロシージャなどと合わせて使用できます。ただし、このマニュアルでは、システムの停止・起動、デバイスの設定などを行う基本的な UNIX コマンドおよびプロシージャについては説明していません。

これらの情報については、以下のいずれかを参照してください。

- Solaris 7 または 8 のソフトウェア環境に関する AnswerBook2™ オンラインマニュアル
- システムに付属するその他のソフトウェアのマニュアル

表記上の規則

このマニュアルでは、次のような字体や記号を特別な意味を持つものとして使用します。

表 **P–1** 表記上の規則

字体または記号	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	.login ファイルを編集します。 ls -a を使用してすべてのファイルを表示します。 system%
AaBbCc123	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	system% **su** password:
AaBbCc123	変数を示します。実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、rm *filename* と入力します。
『 』	参照する書名を示します。	『コードマネージャ・ユーザーズガイド』を参照してください。
「 」	参照する章、節、ボタンやメニュー名、強調する単語を示します。	第 5 章「衝突の回避」を参照してください。 この操作ができるのは、「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。	sun% **grep '^#define ** **XV_VERSION_STRING'**

ただし AnswerBook2™ では、ユーザーが入力する文字と画面上のコンピュータ出力は区別して表示されません。

コード例は次のように表示されます。

- C シェルプロンプト

```
system% command y|n [filename]
```

- Bourne シェルおよび Korn シェルのプロンプト

```
system$ command y|n [filename]
```

- スーパーユーザーのプロンプト

```
system# command y|n [filename]
```

[ ] は省略可能な項目を示します。上記の例は、*filename* は省略してもよいことを示しています。

| は区切り文字 (セパレータ) です。この文字で分割されている引数のうち 1 つだけを指定します。

キーボードのキー名は英文で、頭文字を大文字で示します (例: Shift キーを押します)。ただし、キーボードによっては Enter キーが Return キーの動作をします。

ダッシュ (-) は 2 つのキーを同時に押すことを示します。たとえば、Ctrl-D は Control キーを押したまま D キーを押すことを意味します。

一般規則

- このマニュアルでは、「IA」という用語は、Intel 32 ビットのプロセッサアーキテクチャを意味します。これには、Pentium、Pentium Pro、Pentium II、Pentium II Xeon、Celeron、Pentium III、Pentium III Xeon の各プロセッサ、および AMD、Cyrix が提供する互換マイクロプロセッサチップが含まれます。

第 1 章

Solaris スマートカードの紹介

Solaris スマートカードを使用すると、Solaris デスクトップ環境などのアプリケーションに、スマートカードを使って安全にログインできます。これは、スマートカード上の情報を使って、ログイン時にユーザーの ID を確認できるからです。スマートカード上のログイン情報と同じ情報を提供できないユーザーは、アプリケーションへのアクセスを拒否されます。

スマートカードソフトウェアは、Solaris 8 オペレーティング環境の重要な部分なので、オペレーティングシステムとともに自動的にインストールされます。Solaris 8 を実行しているマシンを起動すると、スマートカードデーモン `ocfserv` が自動的に実行されます。

この章では、次の内容について説明します。

- Solaris スマートカードの主な機能
- サポートされている設定
- スマートカードの機能
- Solaris スマートカードを管理するための作業マップ

Solaris スマートカードの主な機能

Solaris スマートカードソフトウェアには、次の機能があります。

- スマートカード用のオープンカードフレームワーク (OCF) 1.1 規格の実装

- さまざまなカードリーダーのサポート
- 一般的な 3 種類のスマートカードのサポート
- Solaris スマートカードのグラフィカルユーザーインタフェース (GUI) を使った設定方法と、UNIX のコマンド行からの設定方法
- パスワード、PIN、challenge-response 認証方式により、デスクトップ環境やアプリケーションへのログインを保護
- ユーザーのセキュリティ資格情報をカード上に直接格納 (Java カードのみ)

サポートされているカードとリーダー

Solaris スマートカードは、次のスマートカードとカードリーダーをサポートしています。

表 1–1 サポートしているカードのタイプ

カードタイプ	説明	使用されるリーダー
iButton	Java™ の iButton タイプのスマートカード	iButton リーダー
Cyberflex	Java のカード	Sun™ Smart Card Reader I
Payflex	非 Java のスマートカード	Sun Smart Card Reader I

スマートカードによるログイン

スマートカードを使用すると、これまでログインできなかった、セキュリティ保護されているデスクトップ環境や機密情報を扱うアプリケーションにログインできます。ここでは、Solaris スマートカード (デフォルト設定) を使ってセキュリティ保護されているホストにログインするときの手続きについて説明します。

1. ホストに接続されたカードリーダーにカードを挿入します。

2. この時点では、セキュリティ保護されたアプリケーション (Solaris デスクトップなど) は実行できません。Solaris デスクトップ以外のアプリケーションもスマートカードで保護できます。
3. アプリケーションは、カード上に設定された認証情報を読み取ってユーザーを認証しようとします。認証情報とは、デフォルトではユーザーの暗証番号 (PIN) とユーザーアカウントのパスワードです。
4. アプリケーションは、ユーザーに PIN の入力を要求します。ユーザーが PIN を入力すると、この PIN とカードに格納されている PIN が照合されます。
5. 入力された PIN とカード上の PIN の一致を確認できた場合、アプリケーションは、ホストの `/etc/nsswitch.conf` ファイル (NIS、NIS+、またはローカルファイル) に指定されているパスワードデータベースの中からカード上のパスワードと同一のものを検索します。
6. カードのパスワードと同じパスワードが見つかった場合、アプリケーションはユーザーが認証されたものとみなし、ログインを許可します。

最適なスマートカードサイト構成の決定

スマートカードとカードリーダーを購入する前に、通信の認証が必要かどうかを検討します。サイトでスマートカードを使用する理由は次のいずれかです。

- 特定の部署やドメインにあるマシンを承認されていないアクセスから保護する
- 機密情報を扱うアプリケーションへのアクセスを、承認されたユーザーに限定する

スマートカードサイトの構成には、セキュリティドメインまたはバッチオフィスの 2 種類があり、これらを使って組織のニーズを満たすことができます。大規模な組織には、両方のサイト構成の組み合わせが適している場合もあります。

スマートカードのセキュリティドメインの構成

組織内のさまざまな人数のユーザーに対してスマートカードによる認証が必要な場合は、セキュリティドメインを設定してサポートすることを検討してください。たとえば、20 名のスタッフを抱える法人金融部門があるとします。このグループのデスクトップへのログインをスマートカードで保護したい場合は、部門の各ホストに

スマートカードリーダーを接続し、各ユーザーに 1 枚以上のスマートカードを支給する必要があります。

セキュリティドメインの管理者としての作業には次のものがあります。

1. ドメイン内の各ホストに Solaris 8 環境がインストールされていることを確認する
2. スマートカードと認証管理インフラストラクチャ (AMI) デーモンがホスト上で実行されていることを確認する
3. カードリーダーをホストに設置する
4. セキュリティドメイン内のすべてのユーザーの UNIX ユーザーアカウント名を取得する
5. ユーザーのスマートカードを初期化する

注 - バッチオフィス構成とセキュリティドメイン構成を組み合わせた大規模な組織では、ユーザーのスマートカードを初期化する作業をセキュリティドメインの管理者以外が担当することがあります。

バッチオフィスの構成

大規模な組織では、各従業員がスマートカードを使用する必要があります。また、ユーザーが自分のスマートカードを使って複数のホストにログインしなければならない場合もあります。ネットワーク管理者やセキュリティ管理者であっても、すべてのユーザーのホストに対するアクセス権を持っているわけではなく、ログイン時に認証を必要とするアプリケーションをすべて把握しているわけでもありません。

ここでは、従業員のスマートカードを初期化するための専用マシン (複数の場合もあります) を使用するバッチオフィスの設定について考えます。

バッチオフィスの管理者としての作業には次のものがあります。

1. スマートカードの初期化に使用するバッチオフィス内のすべてのホストに Solaris 8 ソフトウェアがインストールされていることを確認する
2. カードリーダーをホストに設置する
3. これらのホスト上でスマートカードと AMI デーモンが実行されていることを確認する
4. 新しいスマートカードが常に手元に供給されていることを確認する

5. スマートカードを必要とするユーザーが、既存の自分のアカウント名、パスワード、アクセスしたいセキュリティ保護されたアプリケーションを記入する要求フォーム (オンラインフォームまたは専用の用紙) を用意する

   ユーザーの身元は、この要求フォームを承認するマネージャーの署名によって保証されます。複数のマシンにスーパーユーザーとしてアクセスできるシステム管理者の身元を明らかにするには、マネージャーの承認が特に重要です。

注 - ユーザーのマシン上でスマートカードの設定を行う担当者が、バッチオフィスの管理者ではなく組織のシステム管理者になることもあります。

Solaris スマートカードの管理

Solaris スマートカードは、Solaris スマートカードグラフィカルユーザーインタフェース (GUI) や UNIX コマンド行を使って管理できます。GUI と UNIX コマンド行を組み合わせて使用することもできます。

Solaris スマートカード GUI の使用

Solaris 8 をインストールすると、Solaris スマートカード GUI が自動的にインストールされます。スマートカード GUI には、Solaris デスクトップまたは UNIX のコマンド行からアクセスできます。

▼ デスクトップから Solaris スマートカード GUI を起動するには

スマートカードを設定しようとしているすべてのマシンで、次の作業を行います。

1. **共通デスクトップ環境 (CDE) にスーパーユーザーとしてログインします。**

   すでに自分の UNIX ユーザー名で CDE を実行している場合は、CDE を終了して、スーパーユーザーとしてログインします。

2. デスクトップのメニューバーにある矢印をクリックして、アプリケーションメニューを表示させます。

3. 「アプリケーション」を選択して、アプリケーションマネージャにアクセスします。

4. 「システム管理」アイコンをダブルクリックして、システム管理コンソールにアクセスします。

5. スマートカードアイコンをクリックして、**Solaris** スマートカード **GUI** を起動します。

▼ コマンド行から Solaris スマートカード GUI を起動するには

1. **UNIX** のコマンド行にスーパーユーザーとしてログインします。

2. 次のように入力して、**GUI** を起動します。

```
# /usr/dt/bin/sdtsmartcardadmin
```

▼ うまくいかない場合に GUI を再起動するには

Solaris スマートカード GUI を起動できない場合、次のような接続エラーが返されることがあります。

```
Xlib: connection to ":0.0" refused by server
Xlib: Client is not authorized to connect to Server
```

1. スーパーユーザーではなく、自身の **UNIX** ユーザー名で端末ウィンドウにログインします。

2. 次のように入力して、クライアントの認証に関する問題を解決します。

```
% xhost +
```

3. 17ページの「デスクトップから Solaris スマートカード GUI を起動するには」の手順に従って、**GUI** を再起動します。

スマートカード **GUI** の使用に関する説明

Solaris スマートカード GUI のヘルプシステムは、次の情報を提供します。

- スマートカード GUI による主な作業の手順

  ヘルプメニューバーをプルダウンしてヘルプを選択します。

- GUI にある各ダイアログボックスの説明

  ダイアログボックスの下部にある「ヘルプ」ボタンを押すと利用できます。

- スマートカードコンソールに関するスポットヘルプ

  「情報」区画に自動的に表示されます。

Solaris スマートカード GUI のヘルプシステムは、スマートカードの概念的な情報は提供しません。スマートカードの動作や認証タイプの詳細などについては、このマニュアルを参照してください。

コマンド行からの **Solaris** スマートカードの管理

ここからは、UNIX のコマンド行を使った Solaris スマートカードの管理作業について説明していきます。また、コマンド行での作業と GUI での作業を支援する概念的な情報も提供します。スマートカードのコマンドの詳細については、`smartcard`(1M) と `ocfserv`(1M) のマニュアルページを参照してください。

スマートカードの設定に関する作業

次の作業マップに、Solaris スマートカードの操作に必要な共通作業と、その説明の記載場所を示します。

表 1–2 スマートカードの管理に関する作業マップ

作業	説明の場所
スマートカードを使用するすべてのマシンに Solaris 8 環境をインストールする	Solaris 8 のインストール関連マニュアル
スマートカードを使用するすべてのホストにカードリーダーを物理的に接続する	カードリーダーの付属マニュアル
`ocfserv` と `amiserv` のデーモンの状態を監視する	23ページの「サーバーデーモンの状態のチェック」
Solaris スマートカード GUI からスマートカードを設定する	17ページの「Solaris スマートカード GUI の使用」と、GUI に組み込まれたヘルプシステム
各ホストでカードリーダーを設定する	26ページの「カードリーダーの設定」
ホスト上でサーバーとクライアントの属性を設定する	28ページの「スマートカードの属性の設定」
ホスト上でスマートカードを操作できるようにする	41ページの「ホスト上でスマートカードを有効化するには」
スマートカード上に必要なインフラストラクチャを作成する	43ページの「スマートカードのインフラストラクチャの読み込み」
スマートカード上にユーザー固有の情報を読み込む	44ページの「スマートカードをユーザー用に初期化」
スマートカードをユーザーに配布し、必要に応じてキーファイルをユーザーのホストに配布する	54ページの「複数のホストでのスマートカードの使用準備」

第 2 章

スマートカードをサポートするホストの設定

この章では、スマートカードをサポートするホストの設定作業について説明します。扱う主題は次のとおりです。

- スマートカードサイトの設定に関する情報の収集
- ホスト上のデーモンの状態のチェック
- シリアルポートのアクセス権の変更
- カードリーダーの設定
- ホスト上のスマートカードの属性の設定
- スマートカードの操作の有効化

注 - この章は、セキュリティドメイン構成またはバッチオフィス構成のサイトでスマートカードを実装する方法を理解しているユーザーを対象としています。詳細については、15ページの「最適なスマートカードサイト構成の決定」を参照してください。

スマートカード用にホストを設定する前に

スマートカード用のホストを設定する前に、いくつかの準備を行う必要があります。

スマートカードサイトの設定に関する情報の収集

この節では、サイトを準備するときに役立つチェックリストを提供します。

以下は、セキュリティドメインの設定前に実行する作業と、調達する品目のチェックリストです。

表 2–1 セキュリティドメインの構成前チェックリスト

作業	実行済み **(?)**
使用するカードリーダーとスマートカードタイプを決定する 詳細については、14ページの「サポートされているカードとリーダー」を参照してください。	
スマートカードを使った安全なログインが必要なホストを特定する	
スマートカードを使用するすべてのマシンに Solaris 8 操作環境をインストールする	
スマートカードを使用するすべてのマシンにカードリーダーを物理的に接続する	
スマートカードによる認証で保護する必要があるアプリケーションを特定する	
スマートカードが必要なユーザーのユーザーアカウント名とパスワードを取得する ユーザーの ID をマネージャに確認しなければならない場合があります。	
サイトのセキュリティポリシーに応じて、ユーザーが後で変更できるデフォルトの PIN を作成	

以下は、バッチオフィスの設定前に実行する作業と、調達する品目のチェックリストです。

表 2–2 バッチオフィスの構成前チェックリスト

作業	実行済み **(?)**
スマートカードの初期化に使用するバッチオフィス内のすべてのマシンに Solaris 8 オペレーティング環境をインストールする	
スマートカードを使用するすべてのバッチオフィスマシンにカードリーダーを物理的に接続する	
デスクトップの他に、スマートカードによる認証で保護する必要があるサイトアプリケーションを特定する	
ユーザー用の認証情報記入フォームをサイトに作成する このフォームには、次の個人情報を記載します。 ■ UNIX のログイン名とパスワード ■ デスクトップの他に、安全なログインが必要なアプリケーション ■ サイトのセキュリティポリシー上必要な場合は、マネージャーの名前と署名	
ユーザーが後で変更できるデフォルトの PIN を作成する	

サーバーデーモンの状態のチェック

スマートカードは、スマートカードサーバー `ocfserv` と AMI サーバー `amiserv` という 2 種類のデーモンを利用して機能しています。Solaris 8 のインストール後にマシンを起動すると、`ocfserv` と `amiserv` が自動的に実行されます。カードリーダーを設定する前に、スマートカードを使用するすべてのホスト上でこれらのデーモンの状態をチェックしてください。

▼ ocfserv と amiserv のデーモンを起動するには

1. カードリーダーが必要な各ホストにスーパーユーザーとしてログインします。
2. 次のように入力して、`ocfserv` が実行中であることを確認します。

```
# ps -ef | grep ocfserv
```

`grep` により、次のような出力が得られます。

```
root   228     1  0 16:31:17 ?        0:00 /usr/sbin/ocfserv
-p com.sun.opencard.utils.OCFPropertyFileLoader
```

このような出力が得られない場合は、ocfserv デーモンを再起動する必要があります。

3. 次のように入力して、 amiserv が実行中であることを確認します。

```
# ps -ef | grep amiserv
```

grep により、次のような出力が得られます。

```
root   229     1  0 16:16:47 ?        0:00 /usr/lib/security/amiserv
```

このような出力が得られない場合は、amiserv デーモンを再起動する必要があります。

4. 次のように入力して、これらのデーモンを起動します。

```
# /etc/init.d/ocfserv start
# /etc/init.d/amiserv start
```

▼ ocfserv と amiserv のデーモンを停止するには

必要に応じて、実行中の ocfserv と amiserv のデーモンを完全に停止できます。

1. 次のように入力して、デーモンを停止させます。

```
# /etc/init.d/ocfserv stop
# /etc/init.d/amiserv stop
```

注 - `ocfserv` と `amiserv` のデーモンの詳細については、`ocfserv`(1M) のマニュアルページを参照してください。

カードリーダーの設定

Solaris スマートカードは、iButton と Sun Smart Card Reader I という 2 種類の外付けカードリーダーをサポートしています。

次の表では、これらのリーダー用の 2 種類のドライバと、その設定時に指定する値 (Java クラス名とリーダーモデル名) を示します。

表 2–3 サポートされているカードリーダー

リーダー	カード端末の出荷時の名前	モデル名
Sun Smart Card Reader I	`com.sun.opencard.terminal.scm.SCMStc.SCMStcCardTerminalFactory`	SunSCRI
iButton	`com.ibutton.oc.terminal.jib.iButtonCardTerminalFactory`	DS1402

ホストにインストールできるカードリーダー数は、リーダーを接続できるシリアルポートの数と同じです。

カードリーダーの接続

スマートカードリーダーを物理的に接続する前に、ポートのデフォルトのアクセス権を変更する必要があります。

▼ シリアルポートの所有権を変更するには

1. カードリーダーを接続しようとしているホストにスーパーユーザーとしてログインします。

2. ポートの所有権を **uucp** からスーパーユーザー **(root)** に変更し、グループの所有権を **sys** に変更します。

   たとえば、シリアルポートの所有権を変更する場合は次のように入力します。

```
# chown root:sys /dev/cua/a
```

3. カードリーダーの付属マニュアルの指示に従って、スマートカードリーダーをシリアルポートに物理的に接続します。

カードリーダーの設定

カードリーダーを設定するには、`smartcard -c admin` コマンドを次の構文で実行します。

```
# smartcard -c admin -t terminal -j card_terminal_factory_name \
-x add -d device_pathname -r user_friendly_reader_name \
 -n card_reader_model
```

- `-t terminal` – カードリーダーを設定します。
- `-j` *card_terminal_factory_name* – カードリーダーの Java カード端末の出荷時の名前を定義します (表 2–3 参照)。
- `-x add` – カードリーダーを追加します。
- `-d` *device_filename* – カードリーダーを差し込んであるポートの UNIX デバイス名を指定します。
- `-r` *user_friendly_reader_name* – カードリーダーの一意な名前を指定します。
- `-n` *card_reader_model* – カードリーダーのモデル名を指定します (表 2–3 参照)。

詳細については、`smartcard`(1M) のマニュアルページを参照してください。

▼ iButton リーダーを設定するには

1. 次のコマンドを **1** 行にすべて入力して、**iButton** リーダーを設定します。

```
# smartcard -c admin -t terminal \
-j com.ibutton.oc.terminal.jib.iButtonCardTerminalFactory  \
-x add -d device_filename -r user_friendly_reader_name -n DS1402
```

- `-t terminal` – カードリーダーを設定します。
- `-j com.ibutton.oc.terminal.jib.iButtonCardTerminalFactory` – iButton リーダーの Java カード端末の出荷時の名前です。

注 - オプション `-j` に続く Java カード端末の出荷時の名前は、正確に入力してください (表 2–3 参照)。文字の間に空白文字や復帰改行を挿入しないでください。

- `-x add` – 追加操作を実行しようとしていることを `smartcard -c admin` に知らせています。
- *device_filename* – `/dev/cua/b` のように、カードリーダーを接続するシリアルポートを定義します。
- *user_friendly reader_name* – `MyButtonReader` のように、iButton リーダーに付ける名前を指定します。
- `DS1402` – iButton カードリーダーのモデル名を定義します。

▼ Sun Smart Card Reader I を設定するには

1. 次のコマンドを **1** 行にすべて入力して、**Sun Smart Card Reader I** を設定します。

```
# smartcard -c admin -t terminal \
-j com.sun.opencard.terminal.scm.SCMStc.SCMStcCardTerminalFactory \
```

```
-x add -d device_filename -r user_friendly_reader_name -n SunSCRI
```

- -t terminal – カードリーダーを設定します。
- -j com.sun.terminal.scm.SCMstcCardTerminalFactory – Sun Smart Card Reader I の Java カード端末の出荷時の名前です。

注 - オプション -j に続くカード端末の出荷時の名前は、注意して正確に入力してください (表 2–3 参照)。文字の間に空白文字や復帰改行を挿入しないでください。

- -x add – 追加操作を実行しようとしていることを smartcard -c admin に知らせています。
- *device_filename* – /dev/cua/b のように、カードリーダーを接続するシリアルポートを定義します。
- -r *user_friendly reader_name* – Sun Smart Card Reader I に付ける名前を指定します。
- Sun SCRI – Sun Smart Card Reader I のモデル名を定義します。

スマートカードの属性の設定

Solaris スマートカードを使用する際は、ocfserv サーバーとクライアントアプリケーションの機能を定義する目的で、ホストごとに属性を設定する必要があります。カードリーダーの設定後、各ホストに Solaris 8 ソフトウェアをインストールするときに、デフォルトのスマートカードの属性のセットを再検討してください。次のような場合は、属性を変更する必要があります。

- サイトのセキュリティ要件を完全にはサポートしていない属性がある場合
- スマートカードやカードリーダーのメーカーが製品を更新した結果、製品番号、Java クラス名などが変更された場合

- サイト内の開発者が、セキュリティ属性を必要とするカスタムアプリケーションを作成した場合

▼ スマートカードの属性を確認するには

1. 設定しようとしているホストにスーパーユーザーとしてログインします。
2. 次のように入力して、設定可能な属性を表示させます。

```
# smartcard -c admin
```

画面には次のように表示されます。

```
Client Properties:
  ClientName.PropertyName     Value
  -----------------------     -----
  default.validcards        =  CyberFlex IButton PayFlex
  default.authmechanism     = Pin=UserPin
  default.defaultaid        = A000000062030400
Server Properties:
  PropertyName                Value
  ------------                -----
 authmechanism                    = Pin Password
OpenCard.terminals                  = com.sun.terminal.scm.
             SCMstcCardTerminalFactory|MySCM|SerialDrive|
                  /dev/cua/b
  ocfserv.protocol          = rpc
  PayFlex.ATR               = 3B690000579202010100010A9

                     3B691100000057920201010001 00
  authservicelocations      =  com.sun.opencard.service.auth
  OpenCard.services         = com.sun.opencard.service.
            cyberflex.CyberFlexServiceFactory
             com.sun.opencard.service.ibutton.
            IButtonServiceFactory
            com.sun.opencard.service.payflex.
            PayFlexServiceFactory
            abc.class com.sun.services.scm.
            SCMstcCardTerminaFactory
 initializerlocations =          com.sun.opencard.cmd.IButtonInit
  IButton.ATR               = 008F0E0000000000000000000004000034909000
  cardservicelocations      = com.sun.opencard.service.common

  CyberFlex.ATR             = 3B169481100601810F 3B169481100601811F
```

(続く)

```
country                   = US
debugging.filename        = /tmp/ocf_debugfile
language                  = en
debugging                 = 0
```

サーバー属性

サーバー属性は、各ホスト上のスマートカードサーバー ocfserv の動作を定義します。この節では、smartcard -c admin によって一覧表示されるサーバーの属性について説明し、属性のデフォルト値を変更する方法を説明します。

smartcard -c admin コマンドを使用して、デフォルトのサーバー属性を変更できます。次のように入力して、ください。

```
# smartcard -c admin -x modify "property_name=property_value"
```

- -x modify – 変更操作を実行しようとしていることを示します。
- *"property_name=property_value"* – 変更する属性とその属性に割り当てる値を表します。

サーバー認証メカニズム属性

authmechanism 属性によって、ocfserv がユーザーを認証するために使用するメカニズムを定義します。

```
authmechanism  = Pin Password
```

可能な認証メカニズムは次の通りです。

- Pin

  ユーザー認証のメカニズムとして、ホストへのログイン時に PIN の入力を求めます。ユーザーは、スマートカードに組み込まれた PIN と同じ PIN を入力する必要があります。

- Password

この認証メカニズムでは、ホストがスマートカード上のパスワードを読み込み、このパスワードが自身のパスワードデータベース (NIS、NIS+、またはローカルファイル) に存在することを確認することによってユーザーを認証します。カード上にパスワードが存在しない場合、ocfserv はユーザーにパスワードの入力を要求し、入力されたパスワードをホストのパスワードデータベースにあるパスワードと照合します。

- ChallengeResponse

  ユーザー認証の手段として、ホストとスマートカードが challenge-response 方式の対話を行います。

Solaris スマートカードのデフォルトの認証メカニズムは、Pin パスワード方式です(14ページの「スマートカードによるログイン」を参照)。

▼ デフォルトの認証メカニズムを変更するには

1. 認証メカニズムを変更しようとしているホストにスーパーユーザーとしてログインします。

2. たとえば次のように入力して、認証メカニズムを **ChallengeResponse** に変更します。

```
# smartcard -c admin -x modify authmechanism=ChallengeResponse
```

   smartcard -c admin と入力すると、結果が表示されます。この場合は、challenge-response メカニズムが追加されていることを確認できます。

```
authmechanism  = ChallengeResponse
```

challenge-response 認証の機能

サーバーの認証メカニズムは、明示的に challenge-response に設定しない限り、PIN パスワード方式になります (デフォルト)。challenge-response とは、ホストとスマートカードの間で発生する対話のことです。smartcard -c init を使ってカードにユーザー情報を追加すると、ホストは自動的に対象鍵 (DES キー) を生成し、その

コピーをカードに追加します。さらに、この対象鍵を /etc/smartcard/.keys ファイルに保管します。54ページの「複数のホストでのスマートカードの使用準備」を参照してください。

ocfserv で challenge-response 方式の認証を使用するように設定すると、ホストは乱数を生成し、リーダーに挿入されているカードにその乱数を送って質問 (challenge) を開始します。これに対するカードの応答 (response) は対象鍵を使って生成されるため、ホストはそれが正しいかどうかを判断できます。

サポートされているカードリーダー属性

OpenCard.terminals 属性は、そのホスト用に設定されたカードリーダーを定義します。たとえば、Sun Smart Card Reader I が接続されたホストの OpenCard.terminals の値は、次のようになります。

```
OpenCard.terminals      = com.sun.terminal.scm.
              CMstcCardTerminalFactory|MySCM|SunSCRI
                dev/cua/b
```

ここで OpenCard.terminals は、Sun Smart Card Reader I を現在設定されているリーダーとして定義しています。カードリーダーを追加すると、smartcard -c admin で表示される OpenCard.terminals 属性が表示されます。カードリーダーの追加については、25ページの「カードリーダーの設定」を参照してください。

サーバープロトコルの属性

ocfserv.protocol 属性は、ocfserv が使用する TCP/IP プロトコルを定義します。

```
ocfserv.protocol            = rpc
```

ocfserv は、遠隔手続き呼び出し (RPC) プロトコルを使用します。この値は変更しないでください。

スマートカードのリセットに対する回答属性

リセットに対する回答 (Answer To Reset: ATR) 属性には、スマートカードのメーカーによって定義された数値が含まれています。Solaris スマートカードがサポートするカードには、次の ATR 属性が定義されています。

```
PayFlex.ATR    = 3B690000579202010100010 0A9
         3B691100000057920201010001 00
IButton.ATR      = 008F0E0000000000000000000000004000034909000
CyberFlex.ATR  =3B169481100601810F 3B169481100601811F
```

ATR の異なる新しいカードタイプがカードメーカーから発売されない限り、ATR 属性を変更する必要はありません。これについては、58ページの「ATR 属性を変更するには」を参照してください。

authservicelocations 属性

この属性は、認証モジュールが入った Java クラスディレクトリの位置を定義します。

```
authservicelocations      = com.sun.opencard.service.auth
```

この値は変更しないでください。

カードサービス属性

これは、カード固有のモジュールが配置される Java クラスディレクトリです。スマートカードタイプごとに、次のように定義されたモジュールが含まれています。

```
OpenCard.services    = com.sun.opencard.service.
         cyberflex.CyberFlexServiceFactory
       `  com.sun.opencard.service.ibutton.
         IButtonServiceFactory
         com.sun.opencard.service.payflex.
         PayFlexServiceFactory
```

これらの値を変更することはめったにありません。値を変更しなければならない場合は、smartcard(1M) のマニュアルページで詳しい説明を確認してください。

initializerlocations 属性

この属性は、アプレット初期設定機能が入った Java クラスディレクトリの位置を定義します。

```
initializerlocations      = com.sun.opencard.cmd.IButtonInit
```

この値は変更しないでください。

カードサービス位置属性

この属性は、カードサービスモジュールが配置されている Java クラスディレクトリの位置を定義します。

```
cardservicelocations      = com.sun.opencard.service.common
```

この値は変更しないでください。

ロケール固有属性

ocfserv には、次のような 2 種類のロケール固有属性を定義できます。

```
country                   = US
language                  = en
```

▼ スマートカードホストのデフォルトのロケールを変更するには

1. ロケール固有の情報を変更しようとしているホストにスーパーユーザーとしてログインします。

2. 次のように入力して、このホストに適する国を指定します。

```
# smartcard -c admin -x modify country=country_code
```

country_code には、このホストに適する 2 文字の国コードが入ります。

3. 次のように入力して、このホストに適する言語を指定します。

```
# smartcard -c admin -x modify language=language_code
```

language_code には、このホストのロケールに適する 2 文字の言語コードが入ります。

デバッグ属性

このデバッグ属性を設定すると、ホスト上でスマートカードの動作をデバッグできます。指定によって、Solaris スマートカードは、標準的なデバッグ機能と詳細なトレース機能を提供します。デフォルトでは、ocfserv には次のデバッグ属性が定義されています。

```
debugging.filename        = /tmp/ocf_debugfile
debugging                 = 0
OpenCard.trace
```

- /tmp/ocf_debugfile はデバッグ情報が入ったファイルの名前です。
- デフォルトのデバッグレベル 0 は、デバッグがオフになっていることを示しています。レベル 1 はデバッグがオンになっていることを示しています。

▼ ocfserv デバッグを有効化するには

1. ocfserv をデバッグしようとしているホストにスーパーユーザーとしてログインします。

2. 次のように入力して、デバッグを有効にします。

```
# smartcard -c admin -x modify debugging=1
```

3. **(オプション)** 次のように入力して、ocfserv のデバッグファイルの位置を変更します。

```
# smartcard -c admin -x modify debugging.filename=filename
```

この場合の *filename* は、デバッグファイルの完全修飾名です。

4. **(オプション)** 次のように入力して、ocfserv によるオープンカードシステムのトレースを開始します。

```
# smartcard -t debug debug_level
```

この場合の *debug_level* は、0 ～ 9 の値になります。

注 - Solaris スマートカードのデバッグに関する完全な情報については、smartcard(1M) のマニュアルページを参照してください。

クライアント属性

クライアント属性では、dtlogin などのアプリケーションプログラムのセキュリティ条件がホストで処理されるときの方式を定義します。

クライアントアプリケーション属性に関するデフォルトのカードタイプ

defaultcard と validcards という 2 種類のカード属性によって、ホスト上のクライアントアプリケーションへのログインに使用するスマートカードタイプを指定します。

validcards 属性では、特定のアプリケーションに、有効なすべてのスマートカードタイプを指定します。一方、defaultcard では、*defaultcard* で定義されたカードの読み込みが完了するまで、アプリケーションが待機するように指定できます。

たとえば、アプリケーション B の validcards 属性に、iButton、Cyberflex、CardA を指定した場合を想定してください。さらに、defaultcard 属性には Cyberflex を指定します。ここで、アプリケーション B がデフォルトのカードだけ

を受け付けており、ユーザーがカード A を使ってアプリケーション B にログインしようとする場合、ホストは次のメッセージを表示します。

```
Waiting for Default Card
```

アプリケーション B へのログインは、ユーザーがリーダーに Cyberflex を挿入するまでブロックされます。

`smartcard -c admin` を実行すると、次の値が表示されます。

```
default.validcards        = CyberFlex IButton PayFlex
```

▼ アプリケーションの有効なスマートカードを変更するには

1. スーパーユーザーとしてログインします。
2. 次のように入力して、デフォルトの有効なカードを変更します。

```
# smartcard -c admin -a default -x modify validcards= \
--- ``IButton|CyberFlex  |PayFlex''
```

この場合の *IButton*|*CyberFlex*|*PayFlex* は、これらの値のいずれか 1 つ、またはその組み合わせを示しています。

たとえば、すべてのアプリケーションに対して有効なスマートカードとして、CyberFlex と PayFlex を定義するには、次のように入力します。

```
# smartcard -c admin -a default -x modify validcards= \
''CyberFlex Payflex''
```

▼ アプリケーションにデフォルトのスマートカードを割り当てるには

1. クライアント属性を変更しようとしているホストにスーパーユーザーとしてログインします。

2. 次のように入力して、デフォルトのスマートカードタイプをアプリケーションに割り当てます。

```
# smartcard -c admin -a application_name -x add defaultcard=card_name
```

   - *application_name* は、デフォルトのスマートカードを定義しようとしているアプリケーションです。
   - *card_name* は、このアプリケーションにログインするために使用するカードタイプです。CyberFlex、PayFlex、または IButton のいずれかになります。

   たとえば、iButton をホストのデスクトップのデフォルトカードタイプとして定義するには、次のように入力します。

```
# smartcard -c admin -a dtlogin -x add defaultcard=IButton
```

   その後、smartcard -c admin を実行すると、次のクライアント属性が表示されます。

```
dtlogin.defaultcard       = IButton
default.validcards        = CyberFlex PayFlex
```

クライアントアプリケーション属性のデフォルトの認証メカニズム

Solaris スマートカードには、クライアントアプリケーションプログラムが使用する認証メカニズムを定義する authmechanism 属性が用意されています。この default.authmechanism 属性は、すべてのクライアントアプリケーションのデフォルトの認証メカニズムを指定します。デフォルトでは、このメカニズムは

Pin=UserPin です。authmechanism 属性では、特定のクライアントアプリケーションで使用する認証メカニズムも定義できます。

▼ すべてのクライアントプログラムのデフォルトの認証メカニズムを変更するには

1. 変更しようとしている属性を持つホストにスーパーユーザーとしてログインします。

2. 次のように **1** 行に入力して、デフォルトの認証メカニズムを変更します。

```
# smartcard -c admin -a default -x modify \
authmechanism=''Pin | Password | ChallengeResponse''
```

この場合、*Pin* | *Password* | *ChallengeResponse* は、これらの値のいずれか 1 つ、またはその組み合わせとなります。

たとえば、クライアントプログラムのデフォルトの認証メカニズムとして Pin と Password の両方を指定したい場合、次のように入力します。

```
# smartcard -c admin -a default -x modify ``authmechanism=Pin Password''
```

その後、`smartcard -c admin` と入力すると、次のデフォルト認証メカニズムが表示されます。

```
default.authmechanism      = Pin Password
```

注 - 割り当てられたクライアント認証メカニズム
と、`ocf.server.authmechanism` に割り当てられた認証メカニズムが一致しない場合、クライアント認証メカニズムの方が優先されます。

▼ 認証メカニズムを特定のクライアントアプリケーションに割り当てるには

application_name.authmechanism 属性によって、特定のアプリケーションに固有の認証メカニズムを割り当てることができます。

1. ホストにスーパーユーザーとしてログインします。
2. 次のように入力して、クライアントプログラムに認証メカニズムを割り当てます。

```
# smartcard -c admin -a application_name -x modify authmechanism=mechanism
```

- *application_name* は、固有の認証メカニズムを必要とするアプリケーションです。
- *mechanism* は、使用するメカニズムです。Pin、Password、ChallengeResponse またはこれら 3 種の組み合わせです。

  たとえば、次のように入力すると、ユーザーがログインする前に、デスクトップがスマートカードを使った challenge-response の会話を要求するように設定できます。

```
# smartcard -c admin -a dtlogin -x modify authmechanism=''ChallengeResponse''
```

以後、smartcard -c admin を実行すると、次の属性が表示されます。

```
dtlogin.authmechanism      = ChallengeResponse
```

デフォルトアプレット ID 属性

デフォルトアプレット ID (AID) 属性とは、すべてのアプリケーションを対象に実行されるデフォルトのスマートカードアプレットに割り当てられた ID 番号です。smartcard -c admin によって表示されるデフォルト ID 番号は、次のとおりです。

```
default.defaultaid         = A000000062030400
```

この値は、Solaris スマートカードがデフォルトで実行するアプレット、SolarisAuthApplet の AID です。

サイト専用にカスタマイズされたアプレットに置き換える必要がない場合は、この `defaultaid` 属性を変更する必要はありません。変更する場合は、参考のために `smartcard`(1M) のマニュアルページを参照してください。

ホスト上でスマートカードを有効化するには

ホストの準備における最終ステップは、スマートカードの動作を有効にすることです。このためには、デスクトップ環境をいくつか変更する必要があります。

▼ ホスト上でのスマートカードの動作の有効化

1. スマートカードオペレーションで使用するホストにスーパーユーザーとしてログインします。
2. 次のように入力して、デスクトップを停止します。

```
# /usr/dt/bin/dtlogin -daemon
```

3. 次のように入力して、スマートカードオペレーションを有効にします。

```
# smartcard -c enable
```

4. 次のように入力して、デスクトップを再起動します。

```
# /usr/dt/bin/dtlogin -daemon
```

5. **(**オプション**)** ホストを再起動します。

以後、このホストにログインするユーザーは、許可されたスマートカードを使う必要があり、場合によっては PIN を入力する必要があります。スマートカードを使ったログインについては、第 5 章を参照してください。

第 3 章

スマートカードの初期化

この章では、スマートカードを初期化する方法、つまりカードをセキュリティドメインで使えるようにする方法について説明します。Solaris スマートカードは、3種類のスマートカードをサポートしています。

- Cyberflex
- Payflex
- iButton

この章では、次の作業について説明しています。

- 必要なインフラストラクチャをスマートカードに作成する作業
- スマートカードをユーザー用に専用化する属性を設定する作業

スマートカードのインフラストラクチャの読み込み

スマートカードを設定する最初の手順では、必要なインフラストラクチャをカードにダウンロードする必要があります。セキュリティドメイン構成のサイトでは、個々のホスト上またはユーザーのマシン上でスマートカードを準備することができます。バッチオフィス構成のサイトでは、専用ホストに接続されたリーダーで、すべてのスマートカードを用意します。

注 - スマートカードの初期化に使用するホストマシンを、あらかじめスマートカード用に設定しておく必要があります。まだそのように設定していない場合は、第 2 章を参照してください。

▼ カードのインフラストラクチャを読み込むには

コマンドを使って、Solaris スマートカードがサポートするすべてのタイプのカードにインフラストラクチャを読み込ませることができます。

1. カードをリーダーに挿入します。
2. スマートカードの初期化に使用するマシンにスーパーユーザーとしてログインします。
3. **Solaris** スマートカードの **capx** ファイルを読み込んでから、次のように入力し、カードのインフラストラクチャを設定します。

```
# smartcard -c load -i /usr/share/lib/smartcard/SolarisAuthApplet.capx
```

`smartcard -c load` がインフラストラクチャの作成を完了すると、次のメッセージが表示されます。

```
処理が成功しました。
```

スマートカードをユーザー用に初期化

カードのインフラストラクチャが読み込まれたら、個人用にスマートカードを初期化します。Solaris スマートカードには、スマートカード上で特定ユーザー用に設定できる属性のセットが入っています。

▼ スマートカードで設定できる属性を一覧表示するには

1. スマートカードをカードリーダーに挿入します。
2. ホストにログインします。
3. 次のように入力して、設定可能な属性を一覧表示します。

```
# smartcard -c init -A A000000062030400 -L
```

`-A` オプションより、`smartcard -c init` は、AID A000000062030400 を使ったアプレットから設定可能な属性を一覧表示します。この AID は、44ページの「カードのインフラストラクチャを読み込むには」で紹介した、デフォルトの SolarisAuthApplet.capx ファイルを指定しています。

`smartcard -c init` を使って、次の属性を一覧表示できます。

```
pin: enter pin
application: enter application name
user: enter user name
password: enter password
privatekey: please enter file path
certificate: please enter file path
```

スマートカードの属性の定義

個々のスマートカードに属性を設定するときは、ユーザーの要求、サイトのセキュリティポリシー、使用するスマートカードのタイプによる制限を考慮する必要があります。第 2 章では、ホストで実行されている `ocfserv` サーバーとクライアントアプリケーションに属性を定義しました。`smartcard -c init` コマンドを使って、対応する属性を個別のスマートカードに定義します。ホスト上のクライアントとサーバープログラムは、スマートカード上の属性を読み込んで、特定のアプリケーションに対するアクセス権をユーザーに許可するかどうかを判定します。

表 3–1　スマートカードで初期化できる属性

属性	サポートするカードタイプ	説明
PIN	すべて	暗証番号。46ページの「PIN 属性」を参照
パスワード	すべて	ユーザーのパスワード (ホストまたはドメインのパスワードデータベースにあるパスワードと同じ)。48ページの「ユーザーとパスワードの属性」を参照
ユーザー	すべて	ユーザーのアカウント名 (ホストまたはドメインのパスワードデータベースにあるアカウント名と同じ)。48ページの「ユーザーとパスワードの属性」を参照
アプリケーション	すべて	このスマートカード上の情報を使ってログインするようにユーザーに要求するアプリケーション。49ページの「アプリケーション属性」を参照
非公開鍵	Cyberflex iButton	ファイルに署名する際に使用する非公開鍵。51ページの「非公開鍵属性」を参照
証明書	Cyberflex iButton	設定可能な属性であるが、Solaris スマートカードには証明書用に定義されたインタフェースは用意されていない。詳細については、`smartcard`(1M) のマニュアルページを参照

注 - これらの属性は、Solaris スマートカードの SolarisAuthApplet を使って初期化されたカードにのみ適用されます。サイトが別のスマートカードアプレットを使っている場合、利用可能な属性が異なる場合があります。詳細については、`smartcard`(1M) のマニュアルページを参照してください。

PIN 属性

PIN 属性は、そのカードの暗証番号 (PIN) を定義する、スマートカード上の認証属性です。`smartcard -c load` コマンドによって、デフォルトの PIN `$$$$java` がカードに作成されます。管理者やユーザーは、`$$$$java` を個人用 PIN に変更できます。サイトで使用する際は、changeme のようなデフォルトの PIN 名または同様のシーケンスを持つカードをユーザーに配布してください。66ページの「スマートカード上の PIN を変更するには」で説明されているように、ユーザーは後でこの PIN を変更できます。

注 - スマートカード上には、ユーザー名とパスワードの組み合わせを複数定義できます。ただし、PIN は 1 つしか定義できません。

▼ スマートカードの PIN を初期化するには

この手順は、Solaris スマートカードがサポートするすべてのカードに当てはまります。この手順を繰り返すことで、複数のスマートカードの PIN 属性を同時に初期化できます。

1. スマートカードの初期化を行うホストにスーパーユーザーとしてログインします。
2. 初期化しようとしているスマートカードがリーダーに挿入されていることを確認します。
3. 次のように入力して、そのカードの **PIN** を設定します。

```
# smartcard -c init -A A000000062030400 -P 'PIN_number'
```

この場合、*PIN_number* は、ユーザーの ID 確認に必要な、カードに設定しようとしている PIN 番号を表しています。

PIN 属性の機能

ocfserv と個々のアプリケーションのデフォルトの認証メカニズムは、PIN パスワード方式です。この場合、ユーザーがデスクトップのようなアプリケーションにログインしようとすると、アプリケーションはユーザーに PIN の入力を要求します。

ocfserv サーバーは、ユーザーが入力した PIN とスマートカードの PIN を照合して、ユーザーが本人であることを確認します。これらの PIN が一致すると、ユーザーにアプリケーションへのアクセスが許可されます。許可されない場合は、ocfserv がカードに補助認証属性を読み込みます。

ユーザーとパスワードの属性

ユーザーとパスワードの属性は、カードにある補助認証属性です。これらの属性はユーザーを特定し、そのユーザーとスマートカードの PIN を関連付けます。これらの属性を設定する前に、サイトでスマートカードを使用するすべてのユーザーを対象に、ユーザーアカウントとアカウントに関連付けられたパスワードを取得する必要があります。

▼ スマートカードのパスワードを初期化するには

この手順は、Solaris スマートカードがサポートするすべてのスマートカードデバイスに当てはまります。この手順を繰り返すことで、複数のスマートカードのユーザーとパスワードの属性を同時に初期化できます。

1. スマートカードを初期化するホストにスーパーユーザーとしてログインします。
2. 初期化しようとしているスマートカードがリーダーに挿入されていることを確認します。
3. 次のように 1 行に入力して、そのカードのユーザー名とパスワードを設定します。

```
# smartcard -c init -A A000000062030400 -P 'PIN_number' user=user_name \
password=user_password application=application_name
```

- *PIN_number* – カードに割り当てられた PIN を表しています。ユーザーの ID を確認するために必要です。
- *user_name* – 個人の UNIX ログイン名です。
- *user_password* – *user_name* に関連付けられたパスワードです。このパスワードは、ホストの `/etc/nsswitch.conf` ファイル (NIS、NIS+、またはローカルファイル) によって定義されたパスワードデータベースの中に存在する必要があります。

- `application=` *application_name* – この PIN とパスワードを使ったスマートカードによるログインを要求するアプリケーションを指定します。

ユーザーとパスワードの属性の機能

PIN パスワード方式 (デフォルトの認証メカニズム) を使用しているホストでは、`ocfserv` は、47ページの「PIN 属性の機能」で説明されているように、PIN が本物であることを確認します。さらに、`ocfserv` は、カードのユーザーとパスワードの属性を読み込みます。スマートカードのパスワードが、ホストのパスワードデータベースにあるパスワードと一致すると、ユーザーにアプリケーションへのアクセスが許可されます。

1 つのスマートカードにユーザーとパスワードの組み合わせを複数定義することができます。たとえば、スマートカードの所有者が、別のアプリケーションにアクセスするために、異なるユーザー名とパスワードを必要とする場合があります。さらに、システム管理者については、標準のユーザー名とパスワードの他に、カードのスーパーユーザーとしてのパスワードを定義できます。ただし、スマートカードには、1 つの PIN しか定義できません。

アプリケーション属性

アプリケーション属性を使うと、特定のユーザー名とパスワードを使ってログインする必要があるアプリケーションを指定できます。たとえば、スマートカードを使ってデスクトップにログインする必要がある場合、カードのユーザー名とパスワードに関連付けられたアプリケーションとして、`dtlogin` を指定する必要があります。また、財務パッケージや人事データベースのようなサイト固有のアプリケーションの名前をアプリケーション属性として指定することによって、このアプリケーションへのスマートカードベースのログインを要求することもできます。

カード上のアプリケーションを初期化する前に、スマートカードによる認証を使ってアクセスする必要があるアプリケーションを調べておきます。root (スーパーユーザー) など、一般のユーザーには使用が制限されているユーザー名でアプリケーションにログインする必要があるユーザー (システム管理者など) のスマートカードを用意する場合は、この作業が特に重要です。

▼ スマートカードのアプリケーションを初期化するには

この手順は、Solaris スマートカードがサポートするすべてのスマートカードデバイスに当てはまります。この手順を繰り返すことで、複数のスマートカードのアプリケーション属性を同時に初期化できます。

1. スマートカードを初期化するマシンにスーパーユーザーとしてログインします。
2. 初期化しようとしているスマートカードがリーダーに挿入されていることを確認します。
3. 次のように **1** 行に入力して、そのスマートカードのアプリケーションを初期化します。

```
# smartcard -c init -A A000000062030400 -P 'PIN_number' user=user_name \
password=user_password application=application_name
```

- `-P` *PIN* – このカードに割り当てられた PIN です。
- `user=`*user_name* – このアプリケーションにログインする際に使用するユーザー名を定義します。
- `password=`*password* – このアプリケーションにログインする際に使用するパスワードを定義します。
- `application=` *application_name* – この PIN とパスワードを使ったスマートカードによるログインが必要なアプリケーションを指定します。

アプリケーション属性の機能

カードのアプリケーション属性は、3 種類の認証属性とともに機能します。たとえば、Frank というユーザーのスマートカードに、次のような情報を付けて初期化する場合を想定してください。

```
# smartcard -c init -A A000000062030400 -P '$$$$java' application=dtlogin \
user=frank password=changeme
```

- -A A000000062030400 – SolarisAuthApplet です。
- -P '$$$$java' – このカードの PIN です。ここではデフォルトの PIN が指定されています。ユーザー Frank は、後でこの PIN を変更できます。
- application=dtlogin – スマートカードによるログインを要求するアプリケーションです。
- user=frank – Frank がデスクトップ (dtlogin アプリケーション) にログインするために入力する名前です。
- password=changeme – Frank がデスクトップにログインするために入力しなければならないパスワードです。

Frank が自分のカードをリーダーに挿入して、ホスト (dtlogin) にログインしようとすると、ocfserv サーバーによってカードの内容が読み込まれ、認証属性が dtlogin に関連付けられているかどうかのチェックが行われます。

ユーザーとパスワードの属性が dtlogin に関連付けられていることを検出した ocfserv は、Frank に PIN の入力を要求し、入力された PIN と、dtlogin アプリケーションに割り当てられたスマートカード上の PIN 属性を照合します。さらに、ホストのパスワードデータベースにあるパスワードと Frank のカード上の情報 (ユーザー名とパスワード) を照合して、Frank が本人であることを確認します。これらの属性が一致すると、Frank にデスクトップへのログインが許可されます。

非公開鍵属性

Solaris スマートカードのこの機能を使用するには、認証管理インフラストラクチャ (AMI) のような公開鍵インフラストラクチャ (PKI) を使ってサイトを設定する必要があります。

注 - 1 枚のスマートカードには、1 つの非公開鍵しか格納できません。

▼ スマートカードの非公開鍵を初期化するには

この手順は、Java ベースの iButton と Cyberflex のスマートカードに当てはまります。Payflex のカードには非公開鍵を格納できません。

1. **PKI** 用の適切なコマンドを使用して、ユーザー用に **1** 組の公開 / 非公開鍵を作成します。

2. 鍵の組み合わせのうち、非公開鍵の部分を別のファイルにエクスポートします。
   非公開鍵属性を設定する際にこのファイルの完全修飾パス名を指定する必要があるので、そのパス名を記録します。

3. スマートカードの初期化に使用するマシンにスーパーユーザーとしてログインします。

4. カードをスマートカードリーダーに挿入します。

5. 次のように入力して、**Java** セキュリティディレクトリにアクセスします。

```
# /usr/java1.2/jre/lib/security
```

6. java.security ファイルを編集します。

7. ファイル内で、security.provider 定義を探します。

```
 This is the "master security properties file".
#
.
.
# Each provider must implement a subclass of the Provider class.
# To register a provider in this master security properties file,
# specify the Provider subclass name and priority in the format
#
security.provider.<n>=<className>
```

8. コメント記号 **(#)** が security.provider.<n>=<className> の前に置かれていることを確認します。

9. 次のようにファイルを編集します。

```
# Each provider must implement a subclass of the Provider class.
# To register a provider in this master security properties file,
# specify the Provider subclass name and priority in the format
#
#    security.provider.<n>=<className>
security.provider.2=com.sun.ami.common.SunAMI
```

10. 次のように入力してサーバーを再起動します。

```
# /etc/init.d/ocfserv stop
# /etc/init.d/amiserv stop
# /etc/init.d/ocfserv start
# /etc/init.d/amiserv start
```

11. 次のように **1** 行に入力して、カードを初期化します。

```
# smartcard -c init -A A000000062030400 -P 'PIN_number' privatekey=keyfile_name
```

- *PIN_number* – カードに割り当てられた PIN を表します。
- *keyfile_name* – ユーザーの非公開鍵が入ったファイルのフルパス名です。

注 - SolarisAuthApplet では、証明書属性の完全な実装は提供されていません。

非公開鍵属性の機能

カードの PIN とパスワードを認証すると、`ocfserv` サーバーは *keyfile_name* で指定されたファイルをスマートカードにコピーします。以後、追加的な認証としてデータに署名するとき、この非公開鍵を利用できます。ユーザーがデータに署名するコマンド (AMI では `amisign`) を実行すると、非公開鍵により、スマートカード上に署名付きデータが作成されます。

サイトポリシーによっては、ユーザーの非公開鍵ファイルをホストから削除したい場合があります。削除すると、その非公開鍵はユーザーのスマートカード上にしか存在しないことになります。

複数のホストでのスマートカードの使用準備

`smartcard -c init` コマンドを実行してユーザーのスマートカードを初期化すると、ホスト上とスマートカード上に対称鍵が作成されます。`ocfserv` は、ホストに設定されているすべての秘密鍵に関する情報が入った `/etc/smartcard/.keys` というファイルを作成します。ユーザーが、スマートカードを作成したホスト以外のホストにアクセスしなければならない場合、アクセスする必要があるすべてのホストに、`/etc/smartcard/.keys` ファイルをエクスポートする必要があります。

▼ 鍵ファイルをエクスポートするには

カードが作成されたホストから `/etc/smartcard/.keys` ファイルをエクスポートするには、この手順を使います。

1. カードが作成されたホストにスーパーユーザーとしてログインします。
2. バッチオフィス構成のサイトでは、このユーザーの鍵 **(**`/etc/smartcard/.keys` で表示**)** だけが入った専用の鍵ファイルを作成します。
3. 次のように入力して、`/etc/smartcard/.keys` をエクスポートします。

```
# smartcard -c admin -k challenge_response -E -o key_file_name
```

この場合、*key_file_name* は、個別ユーザーの対称鍵が入ったファイルの名前を表しており、`/etc/smartcard/.keys` ファイルまたはそのユーザー固有の別ファイルとなります。

▼ 鍵をインポートするには

ユーザーのカードが作成されたホスト以外のホストへユーザーの対称鍵をインポートするには、この手順を使います。

1. スーパーユーザーとしてホストにログインします。

2. 次のように入力して、鍵ファイルを新しいホストにインポートします。

```
# smartcard -c admin -k challenge_response -I -i key_file_name
```

   この場合、*key_file_name* は、/etc/smartcard/.keys またはそのユーザー専用に作成した別のファイルです。

3. ユーザーがスマートカードを使ってアクセスする必要があるホストごとに、手順 **1** ～ **2** を繰り返します。

第 4 章

スマートカードの保守

スマートカードパッケージの削除または再インストール

表 4–1 では、Solaris 8 のインストール中に追加される Solaris スマートカードパッケージを一覧表示します。

表 4–1 Solaris スマートカードパッケージ

パッケージ名	説明
SUNWjcom	スマートカードをサポートする Java 通信 API – Java コードとネイティブコード
SUNWjcomx	スマートカードをサポートする Java 通信 API – ネイティブコード (64 ビット)
SUNWjib	Dallas Semiconductor シリアル iButton OCF カード端末ドライバ
SUNWocf	オープンカードフレームワーク – コアライブラリとユーティリティ
SUNWocfr	オープンカードフレームワーク – 設定ファイル
SUNWocfx	オープンカードフレームワーク – 64 ビットコアライブラリ
SUNWscgui	Solaris スマートカードグラフィカルユーザーインタフェース

表 4–1　Solaris スマートカードパッケージ　続く

パッケージ名	説明
SUNWpamsc	スマートカード認証用の接続可能な認証モジュール
SUNWpamsx	スマートカード認証に関する接続可能な認証モジュール
SUNWscmsc	外付け Sun Smart Card Reader I OCF カード端末ドライブ
SUNWocfh	オープンカードフレームワークのヘッダーファイル
SUNWscmos	スマートカード認証用の接続可能な認証モジュール

パッケージを削除する必要がある場合は、標準 UNIX の pkgrm コマンドを使ってください。パッケージの再インストールには、 pkgadd コマンドを使ってください。

スマートカードの保守

この節では、スマートカードとホストの保守作業について説明します。

スマートカードの新しいリリースへの更新

サイトで使用しているスマートカードのメーカーが、別の ATR を使う新しいカードのタイプを発売した場合は、ホストの ATR 属性を変更する必要があります。ATR については、33ページの「スマートカードのリセットに対する回答属性」を参照してください。新しいカードを使用するすべてのホストでこの属性を変更します。

▼ ATR 属性を変更するには

1. カードのメーカーのマニュアルを参照し、スマートカードの新しい **ATR** を取得します。

2. 新しいスマートカードタイプを使用するホストにスーパーユーザーとしてログインします。

3. 次のように入力して、スマートカードの **ATR** を変更します。

```
# smartcard -c admin -x modify ''card_name.ATR=ATR_number''
```

   - *card_name* は、PayFlex、CyberFlex、IButton のいずれかです。
   - *ATR_number* は、このカードに割り当てられた新しい ATR 番号です。

スマートカードの再使用

スマートカードは再使用できます。以前に初期化したスマートカードで別のユーザーやアプリケーションをサポートしたい場合、このカードの内容を消去できます。カードの内容を消去する手順は、サポートされている 2 種類の外部カードリーダーのうちどちらを使用するかによって異なります。

注 - スマートカードの内容を消去する前に、この内容を本当に消去してもかまわないかどうか再確認してください。次の手順を実行すると、カード上のすべての情報が削除されます。

▼ iButton の内容を消去するには

1. **iButton** リーダーが接続されているマシンにスーパーユーザーとしてログインします。

2. 消去しようとしている **iButton** がリーダーに挿入されていることを確認します。

3. 次のように入力して、現在接続されている **iButton** リーダーの名前を一覧表示します。

```
# smartcard -c admin
```

4. 表示された結果の中から `OpenCard.terminals` で始まる行を探します。

   ユーザーが覚えやすいように割り当てた名前 (等号 (=) から 2 番目の値) がこの処理に必要な iButton リーダー名です。

```
OpenCard.terminals = com.ibutton.oc.terminal.jib.iButtonCardTerminalFactory|iButtonAdapter|
```

この例の場合、該当する名前は iButtonAdapter です。この名前は次の手順で使用します。

5. 次のように入力して、**iButton** を消去します。

```
# smartcard -c load -r user_friendly_name -u -A A000000062030400
```

▼ Sun Smart Card Reader I を使ってスマートカードの内容を消去するには

Sun Smart Card Reader I に挿入されている CyberFlex または PayFlex のスマートカードから明示的にデータを消去することはできません。代わりに、SolarisAuthApplet.capx ファイルを再度読み込むことでカードの内容を消去できます。

1. **Sun Smart Card Reader I** が接続されているマシンにスーパーユーザーとしてログインします。
2. 内容を消去しようとするカードがリーダーに挿入されていることを確認します。
3. **1** 行に次のように入力して、カードを消去します。

```
# smartcard -c load -i /usr/share/lib/smartcard/SolarisAuthApplet.capx -A A000000062030400
```

完了すると、ocfserv によって次のようなメッセージが表示されます。

```
処理に成功しました
```

カードリーダーの削除

スマートカードが不要になった場合や、リーダーを別のマシンに接続し直したい場合は、カードリーダーをホストから取り外します。カードリーダーは、物理的に取り外す前に、論理的にも削除する必要があります。

▼ カードリーダーを削除するには

1. 無効化しようとしているカードリーダーが接続されているマシンにスーパーユーザーとしてログインします。
2. 次のように入力して、カードリーダーを論理的に削除します。

```
# smartcard -c admin -t terminal -r user_friendly_name -x delete
```

3. ポートからカードリーダーを取り外します。

スマートカードのオペレーションの無効化

ユーザーがスマートカードの PIN を忘れてしまった場合や、スマートカードでのログインが不要になった場合は、ホスト上のスマートカードのオペレーションを無効化します。

▼ スマートカードを無効化するには

ユーザーがスマートカードの PIN を忘れてしまい、デスクトップにログインできなくなった場合は、次の作業を行います。

1. マシンを停止してから、シングルユーザーモードで再起動します。
2. 次のように入力して、スマートカードのオペレーションを無効化します。

```
# smartcard -c disable
```

3. シングルユーザーモードを終了します。マシンの起動プロセスを再開させ、デスクトップ環境に戻ります。

第 5 章

スマートカードの使用

この章では、スマートカードの使用方法を説明します。この章の対象読者は、スマートカードを使って Solaris 8 オペレーティング環境にログインする必要があるユーザーです。

スマートカードを使用する前に

スマートカードは、Solaris デスクトップや個々のアプリケーションを保護するリソースです。スマートカードを使用すると、パスワードによる一般的な UNIX ログインよりも高いセキュリティ効果を期待できます。これは、デスクトップやアプリケーションに対して自分自身を認証する (自分が本人であることを証明する) ことができるからです。

スマートカードの内容

Solaris スマートカードソフトウェアは、CyberFlex、iButton、PayFlex という 3 種類のスマートカードをサポートしています。ユーザーのコンピュータにスマートカードリーダーを設定し、組織で使用するスマートカードを各ユーザーに支給するのはシステム管理者です。

スマートカードには次の内容が含まれています。

- UNIX のユーザー名
- (オプション) パスワード

- スマートカード用の暗証番号 (PIN)
- スマートカードの PIN を使ったログインを要求するアプリケーションプログラムの名前
- (オプション) 署名付きファイルに使用される非公開鍵

セキュリティ管理者に提供する情報

サイトポリシーによっては、セキュリティ管理者が各ユーザーのマシンまたはバッチオフィスでスマートカードの初期化を行うことがあります。この場合は、セキュリティ管理者に次の情報を提供する必要があります。

1. UNIX のユーザー名
2. パスワード
3. パスワードを使ったログインを要求する Solaris デスクトップとその他のすべてのアプリケーション
4. 優先 PIN (サイトのポリシーによって異なる)

   セキュリティ管理者から受け取ったデフォルトの PIN を、各自で変更しなければならない場合もあります。

デスクトップへのスマートカードを使ったログイン

セキュリティ管理者から受け取ったスマートカードは、すぐに使用できます。

▼ スマートカードを使って Solaris デスクトップにログインするには

1. **まだスマートカードをカードリーダーに挿入していない場合は、カードを挿入します。**

Solaris デスクトップ環境が PIN の入力を求めてきます。

2. セキュリティ管理者から受け取った **PIN** を入力します。優先 **PIN** またはデフォルトの **PIN** です。

   適切な PIN を入力すると、次のいずれかの処理が行われます。

   - スマートカードにパスワードが入っている場合は、自動的にデスクトップにログインできます。
   - スマートカードにユーザー名とパスワードが入っていない場合は、標準的な UNIX ログインと同様にユーザー名とパスワードを入力しなければなりません。適切なユーザー名とパスワードを入力すると、デスクトップにログインできます。

▼ 保護されているアプリケーションにスマートカードを使ってログインするには

1. まだスマートカードをカードリーダーに挿入していない場合は、カードを挿入します。

2. 64ページの「スマートカードを使って Solaris デスクトップにログインするには」の方法に従ってログインします。

3. 保護されているアプリケーションを実行します。

   このアプリケーションが PIN の入力を求めてきます。

4. セキュリティ管理者から受け取った **PIN** を入力します。優先 **PIN** またはデフォルトの **PIN** です。

   適切な PIN を入力すると、次のいずれかの処理が行われます。

   - スマートカードにパスワードが入っている場合は、自動的にアプリケーションにアクセスできます。
   - スマートカードにユーザー名とパスワードが入っていない場合は、標準的な UNIX ログインと同様にユーザー名とパスワードを入力しなければなりませ

ん。適切なユーザー名とパスワードを入力すると、アプリケーションにアクセスできます。

▼ スマートカード上の PIN を変更するには

1. まだスマートカードをカードリーダーに挿入していない場合は、カードを挿入します。

2. 次のように入力して **PIN** を変更します。

```
% smartcard -c init -A A000000062030400 -P 'old_PIN' pin=new_PIN
```

   - *old_PIN* は現在の PIN です。
   - *new_PIN* は新しい PIN です。

索引