Solaris Smart Cards Administration Guide

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, , CA 94303-4900
U.S.A. 650-960-1300

Part Number 806-1646
February 2000, Revision A

Contents

Preface

The Solaris Smart Cards feature enables users to log in securely to the Solaris™ 8 desktop environment or other application through use of a smart card. The *Solaris Smart Cards Administration Guide* explains how to set up hosts and smart cards for this form of authentication. It also explains how to use cards after they have been configured.

Who Should Use This Book

This document addresses the needs of two different audiences:

- Network administrators experienced with the Solaris operating environment or other form of the UNIX® operating system
- Users who need to log in to their Solaris based machine with a smart card

If You Are a Network Administrator

The first four chapters of *Solaris Smart Cards Administration Guide* assume that you are a network administrator, security administrator, or system administrator. The text assumes that you have a solid knowledge of authentication and related network security concepts. If you need an introduction to authentication, refer to the *Authentication Management Infrastructure Administration Guide.*

The first chapter of the *Solaris Smart Cards Administration Guide* contains introductory and conceptual information regarding the smart card software. The next three chapters describe the tasks involved in administering smart cards from the

command line.These sections contain procedures for accomplishing the main tasks described in the chapter.

If You Are a Smart Card User

If your goal is to use your smart card to securely log in to a machine, refer to Chapter 5. You might also want to read “What Happens During a Smart Card Login” on page 1-2 as an introduction to smart cards concepts.

Before You Read This Book

The procedures in this book assume that you have installed the Solaris 8 operating environment on all hosts in your domain to be used with smart cards.

How This Book Is Organized

The information in this book is organized as follows:

Chapter 1 introduces the smart-card authentication technology and explains how smart cards work.

Chapter 2 describes the tasks involved in setting up a host to support smart-card authentication.

Chapter 3 describes the tasks involved in setting up a smart card for a user.

Chapter 4 describes the maintenance tasks for hosts, readers, and smart cards.

Chapter 5 explains how to use smart cards.

Note - This chapter is written for the smart-card user. Chapters 1 – 4 are written for the system or security administrator.

Related Books

TABLE P–1 Related Documentation

Application	Title or Product	Part Number/Location
Authentication Management Infrastructure	*Authentication Management Infrastructure Administration Guide*	805-7739 `http://docs.sun.com`
Authentication Management Infrastructure	*Authentication Management Infrastructure Programming Guide*	805-3079-10, `http://docs.sun.com`

Ordering Sun Documents

Fatbrain.com, an Internet professional bookstore, stocks select product documentation from Sun Microsystems, Inc.

For a list of documents and how to order them, visit the Sun Documentation Center on Fatbrain.com at `http://www.fatbrain.com/documentation/sun`.

Accessing Sun Documentation Online

The `docs.sun.com`SM web site enables you to access Sun technical documentation online. You can browse the `docs.sun.com` archive or search for a specific book title or subject. The URL is `http://docs.sun.com`

Using UNIX Commands

Solaris Smart Cards can be used in conjunction with any Solaris administration tools or UNIX commands and procedures. However, this document may not contain information on basic UNIX commands and procedures such as shutting down the system, booting the system, and configuring devices.

Refer to one or more of the following for this information:

- AnswerBook™ online documentation for the Solaris 7 or 8 software environment

- Other software documentation that you received with your system

Typographic Conventions

TABLE P–2 Typographic Conventions

Typeface or Symbol	Meaning	Examples
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output.	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail.`
`AaBbCc123`	What you type, when contrasted with on-screen computer output.	`% su` `Password:`
AaBbCc123	Book titles, new words or terms, words to be emphasized. Command-line variable; replace with a real name or value.	Read Chapter 6 in the *User's Guide*. These are called *class* options. You *must* be `root` to do this. To delete a file, type `rm` *filename*.

Shell Prompts

TABLE P–3 Shell Prompts

Shell	Prompt
C shell	*machine_name*%
C shell superuser	*machine_name*#

TABLE P–3 Shell Prompts *(continued)*

Shell	Prompt
Bourne shell and Korn shell	$
Bourne shell and Korn shell superuser	#

CHAPTER 1

Introducing Solaris Smart Cards

Solaris Smart Cards enables secure login to the Solaris desktop environment or other application through use of a smart card. Information on the smart card verifies the identity of the user during login. Users who cannot provide the same login information that is on the smart card are denied access to the application.

The smart-card software is an intrigue part of the Solaris 8 operating environment, and thus is automatically installed with the operating system software. After the machine running the Solaris 8 software boots, the smart-card daemon `ocfserv` runs automatically.

This chapter discusses the following topics:

- Main features of Solaris Smart Cards
- Configurations supported
- How the smart-card feature works
- Task map for administering Solaris Smart Cards

Main Features of Solaris Smart Cards

The Solaris Smart Cards software has the following features:

- Implements the open card framework (OCF) 1.1 standard for smart cards
- Supports a variety of card readers
- Supports three widely used smart cards
- Enables configuration from the Solaris Smart Cards Graphical User Interface (GUI) or UNIX command line

- Protects login to the desktop environment or application, through use of the password, PIN, and challenge-response authentication methods
- Enables the storage of a user's security credentials directly onto the card (Java cards only)

Supported Cards and Readers

Solaris Smart Cards supports the following smart cards and card readers.

TABLE 1–1 Types of Cards Supported

Card Type	Description	Reader Used
iButton™	Java "button" type smart card	iButton reader
Cyberflex	Java card	Sun Smart Card Reader I
Payflex	Non-Java smart card	Sun Smart Card Reader I

What Happens During a Smart Card Login

Smart cards enable users to log in to a secure desktop environment or sensitive application that otherwise would be closed to them. The next sequence explains what happens when someone logs in to a host protected by the default Solaris Smart Cards configuration:

1. The user inserts the card into the card reader attached to the host.
2. The user attempts to run a protected application, typically the Solaris desktop, but other applications can be protected by smart cards, as well.
3. The application tries to authenticate the user by reading authentication information configured on the card—by default, the user's personal identification number (PIN) and user account password.

4. The application prompts the user to type the PIN, and then compares the typed PIN with the PIN stored on the card.
5. If the typed PIN and the PIN on the card match, the application then searches the password database specified in the host's `/etc/nsswitch.conf` file (NIS, NIS+, or local files) for the same password as is on the card.
6. If the application finds the card's password in the host's password database, it considers the user authenticated and logs in the user.

Determining the Best Smart Card Site Configuration for Your Organization

Before purchasing smart cards and card readers, consider your organization's need for authenticated communications. Your organization's reason for using smart cards might be:

- To keep machines in a particular department or domain secure from unauthorized access
- To limit access to a sensitive application only to authorized users

Your organization's needs can best be served by two types of smart-cards site configurations, the security domain or badge office. A combination of both site configurations might suit some large organizations.

Smart-Cards Security Domain Configuration

If a discrete number of users in your organization require smart-card authentication, consider setting up a *security domain* to support them. For example, suppose the 20-person corporate finance department wants to use smart cards to protect logging in to their group's desktops. In this scenario, each host in corporate finance needs a smart-card reader attached, and each user needs one or more smart cards.

Your tasks as administrator of a security domain involve:

1. Ensuring that each host in the domain has the Solaris 8 environment installed
2. Ensuring that the smart cards and Authentication Management Infrastructure (AMI) daemons are running on the hosts
3. Installing card readers on the hosts
4. Obtaining the correct UNIX user account names for all people in the security domain

5. Initializing the users' smart cards

Note - In a large organization with both badge offices and security domain, the administrator of a security domain might not be responsible for initializing the users' smart cards.

Badge Office Configuration

Large organizations might require every employee to use smart cards. As the network or security administrator, you might not have access to every user's host or know every application that requires an authenticated login. Moreover, in a big organization, users might need to use their smart cards to log in to several hosts.

In this scenario, consider setting up a badge office in which you initialize smart cards for employees on a particular machine or group of machines dedicated for that purpose.

Your tasks as administrator of a badge office would involve:

1. Ensuring that each host in the badge office used for smart card initialization has the Solaris 8 software installed
2. Installing card readers on the hosts
3. Ensuring that the smart cards and AMI daemons are running on these hosts
4. Making sure that an up-to-date supply of smart cards is always on hand
5. Setting up an online or paper request form, by which any users requiring smart cards can supply their existing account names, passwords, and names of secure applications they need to access. A manager's signature or other type of approval of the form vouches that the users are who they say they are. Managerial approval is particularly important if a user is a system administrator requiring root access to a set of machines.

Note - If you are a badge office administrator, you might not be responsible for setting up smart cards on the users' machines. System administrators in your organization have this responsibility.

Administering Solaris Smart Cards

You can administer Solaris Smart Cards through the Solaris Smart Cards graphical user interface (GUI), or the UNIX command line, or a combination of the two.

Using the Solaris Smart Cards GUI

When you install Solaris 8, you automatically install the Solaris Smart Cards GUI. You can run the smart cards GUI either by accessing it from your Solaris desktop or from the UNIX command line.

▼ To Start Up the Solaris Smart Cards GUI From the Desktop

Use this procedure on each machine where you want to configure smart cards.

1. **Log in as root to the Common Desktop Environment (CDE).**
   If you are currently running CDE under your UNIX user name, exit CDE and log in as root.
2. **Click the arrow for the Applications Manager on the desktop menu bar to display the Applications menu.**
3. **Choose Applications to access the Application Manager.**
4. **Double-click the System_Admin icon to access the Solaris Management Console.**
5. **Click the Smart Card icon to start the Solaris Smart Cards GUI.**

▼ To Start Up the Solaris Smart Cards GUI From the Command Line

1. **Log in as root on the UNIX command line.**
2. **Start the GUI by typing:**

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

▼ To Restart the GUI if You Are Unsuccessful

You might receive connection errors similar to the following when trying to start the Solaris Smart Cards GUI.

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

1. **Log in to a terminal window with your UNIX user name (not root).**

2. **Clear the client authorization problem by typing:**

```
% xhost +
```

3. **Restart the GUI, as described in "To Start Up the Solaris Smart Cards GUI From the Desktop" on page 1–5.**

Getting Instructions for Using the Smart Cards GUI

The help system included with the Solaris Smart Cards GUI has:

- Procedures for the major smart cards tasks for you to follow as you use the GUI. Pull down on the Help menu bar and choose Help.
- Descriptions of each dialog box in the GUI, available by pressing the Help button at the bottom of the box.
- Spot help for the Smart Cards Console, automatically displayed in the Information pane.

The Solaris Smart Cards GUI help system does not contain conceptual information about smart cards. Instead, refer to this *Solaris Smart Cards System Administration Guide* for descriptions of how smart cards work, details about authentication types, and other conceptual information.

Administering Solaris Smart Cards From the Command Line

The remainder of *Solaris Smart Cards Administration Guide* contains tasks for administering Solaris Smart Cards from the UNIX command line. The guide also contains conceptual information to supplement both command line and GUI tasks. If you need detailed information about the smart cards commands, refer to `smartcard` (1M) and `ocfserv` (1M) man pages.

Tasks for Setting Up Smart Cards

The next task map lists the most common tasks used in Solaris Smart Cards operations and links to the text where they are described.

TABLE 1–2 Task Map for Smart-Cards Administration

Task Description	Where to Find it
Install the Solaris 8 environment on all machines to use smart cards.	Solaris 8 installation documentation
Physically attach card readers to all hosts to use smart cards.	Documentation that comes with card readers
Monitor the status of the `ocfserv` and `amiserv` daemons.	“Checking the Status of the Server Daemons” on page 2-3
Configure smart cards from the Solaris Smart Cards graphical user interface.	“Using the Solaris Smart Cards GUI” on page 1-5 and the Help system built into the GUI.
Configure card readers on each host.	“Configuring the Card Reader ” on page 2-5
Set up server and client properties on the host.	“Configuring Smart Card Properties ” on page 2-8
Enable smart-card operations on the host.	“Enabling Smart Card Operations on the Host” on page 2-18
Create the necessary infrastructure on the smart card.	“Loading the Smart Card Infrastructure” on page 3-1
Load user-specific information on to the smart card.	“Initializing a Smart Card for a User” on page 3-2
Distribute smart cards to users, and, if needed, key files to users’ hosts.	“Preparing the Smart Card for Use on Multiple Hosts” on page 3-10

CHAPTER 2

Setting Up Hosts for Smart Card Support

This chapter describes the tasks necessary for preparing hosts to support smart cards. Topics covered include:

- Gathering information about your prospective smart card site configuration
- Checking the status of daemons on the host
- Changing permissions on the serial port
- Configuring the card reader
- Configuring the smart card properties on the host
- Turning on smart cards operations

Note - The tasks in this chapter assume that you have identified how smart cards will be implemented at your site—security domain or badge office. See “Determining the Best Smart Card Site Configuration for Your Organization” on page 1-3 for more information.

Before Configuring a Host for Smart Card Use

Before using the procedures for configuring a host for smart cards, you need several preparatory tasks.

Gathering Information About Your Smart Card Site Configuration

This section contains checklists to help you prepare your site. Here is a checklist of tasks that you need to do or items that you need to procure for a security domain.

TABLE 2–1 Security Domain Pre-Configuration Checklist

Task Description	Done (?)
Determine the types of card readers and smart cards your organization will use. See “Supported Cards and Readers” on page 1-2 for more information.	
Identify the hosts that need secure login through smart cards.	
Install the Solaris 8 operating environment on all machines to use smart cards.	
Physically assemble card readers on all machines to use smart cards.	
Identify the applications that must be protected by smart card authentication.	
Obtain the user account names and passwords of users who need smart cards. If your site requires it, you might need to verify the users’ identities with their managers.	
Depending on your site’s security policies, create a default PIN for the user to change later.	

Here is a checklist of tasks that you need to do or items that you need to procure for a badge office.

TABLE 2–2 Badge Office Pre-configuration Checklist

Task Description	Done (?)
Install the Solaris 8 operating environment on all machines in the badge office to be used for initializing smart cards.	
Physically assemble card readers on all badge office machines to use smart cards.	

TABLE 2–2 Badge Office Pre-configuration Checklist *(continued)*

Task Description	Done (?)
Identify the applications at your site, in addition to desktops, that must be protected by smart card authentication. Create a form for users at your site to fill out with authentication information. This form should list the individual's: ■ UNIX login name and password ■ Applications in addition to the desktop that require a secure login ■ Manager's name and signature, if required by your site's security policies Create a default PIN for the user to change later.	

Checking the Status of the Server Daemons

Smart card operation relies on two daemons, the smart card server `ocfserv` and the AMI server `amiserv`. When a machine boots after Solaris 8 installation, `ocfserv` and `amiserv` should run automatically. Before you configure the card reader, check the status of these daemons on each host to be used with smart cards.

▼ To Start the `ocfserv` and `amiserv` Daemons

1. **Log in as root on each host requiring a card reader.**

2. **Verify that `ocfserv` is running by typing:**

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

   `grep` should print a response similar to the following:

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

   If you do not get this response, you need to restart the `ocfserv` daemon.

3. **Verify that `amiserv` is running by typing:**

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

`grep` should print a response similar to the following:

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

If you do not get this response, you need to restart the `amiserv` daemon.

4. **Start the daemons by typing**:

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

▼ To Halt the `ocfserv` and `amiserv` Daemons

You can cleanly stop the `ocfserv` and `amiserv` daemons from running, if necessary.

1. **Stop the daemons by typing**:

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

Note - For more information about the `ocfserv` and `amiserv` daemons, refer to the `ocfserv`(1M) and `amiserv`(1M) man pages.

Setting Up Card Readers

Solaris Smart Cards support two external card readers, the iButton and Sun Smart Card Reader I.

The next table shows the two drivers associated with the readers, and the corresponding values you need to supply to configure them: Java Class name and Reader Model name.

TABLE 2–3 Card Readers Supported

Reader Type	Card Terminal Factory Name	Model Name
Smart Card Reader I	com.sun.opencard.terminal.scm. SCMStc.SCMStcCardTerminalFactory	SunSCRI
iButton	com.ibutton.oc.terminal.jib.iButtonCardTerminalFactory	DS1402

You can install as many card readers on a host as there are serial ports into which you can plug the readers.

Attaching the Card Reader

Before physically attaching a smart card reader, you need to change the default permission on the port.

▼ To Change Ownership on a Serial Port

1. **Log in as root on the host where you are attaching the card reader.**

2. **Change port ownership from uucp to root; change the group ownership to sys.**

   For example, to change ownership of serial port a, you type:

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

3. **Physically attach the smart card reader to the serial port, following instructions in the documentation provided with the card reader.**

Configuring the Card Reader

You configure a card reader using the `smartcard -c admin` command with the following syntax:

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

In this syntax:

- `-t terminal` indicates that you are about to configure a card reader provider.
- `-j` *card_terminal_factory_name* defines the Java card terminal factory name of the card reader type, as shown in Table 2–3.
- `-x add` indicates that you want to add a card reader.
- `-d` *device_filename* specifies the UNIX device name of the port where you have plugged in the card reader.
- `-r` *user_friendly_name* specifies the unique name that you want to give to the card reader.
- `-n` *card_reader_model* designates the name of the card reader model, as listed in Table 2–3.

Refer to the `smartcard` (1M) man page for more information.

▼ To Configure the iButton Reader

1. Configure the iButton reader by typing the next command all on one line:

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

where:

- `-t terminal` – Indicates that you are about to configure a card reader.
- `-j com.ibutton.oc.terminal.jib.iButtonCardTerminalFactory` – Is the Java card terminal factory name of the iButton reader.

Note - Be careful to type the Java card terminal factory name following option `-j` exactly as shown in Table 2–3, with no spaces or returns between characters.

- `-x add` – Tells `smartcard -c admin` that you want to perform an add operation.
- *device_filename* – Defines the port where the card reader is attached, for example, the `/dev/cua/b` serial port.
- *user_friendly_reader_name* – Indicates the name that you want for the iButton reader, for example, `MyButtonReader`.
- DS1402 – Defines the model name for the iButton card reader.

▼ To Configure the Sun Smart Card Reader I

1. **Configure the Sun Smart Card Reader I by typing the next command all on one line:**

```
# 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
```

where:

- `-t terminal` – Indicates that you are about to configure a card reader.
- `-j com.sun.terminal.scm.SCMstcCardTerminalFactory` – Is the Java card terminal factory name of the Sun Smart Card Reader I.

Note - Be careful to type the card terminal factory name following option `-j` exactly as shown in Table 2–3, with no spaces or returns between characters.

- `-x add` – Tells `smartcard -c admin` that you want to perform an add operation.
- *device_filename* – Defines the port where the card reader is attached, for example, the `/dev/cua/b` serial port.
- `-r` *user_friendly_reader_name* – Indicates the name that you want for the Sun Smart Card Reader I.
- `Sun SCRI` – Defines the model name for the Sun Smart Card Reader I.

Configuring Smart Card Properties

Solaris Smart Cards include a group of properties that you must set for each host to define how the `ocfserv` server and client applications should operate. After configuring the card reader, review the default smart cards properties set when you install the Solaris 8 software on each host. You need to change these properties if:

- They do not entirely support the security needs of your site.
- Your smart card or card reader manufacturer updates their product, and changes the product's ID numbers, Java Class names, or other information.
- Developers at your site create custom applications that require you to add security properties.

▼ To View Smart Card Properties

1. **Log in as root on the host that you want to configure.**

2. **Display the configurable properties by typing:**

```
# smartcard -c admin
```

Your screen should resemble the following.

```
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
```

(continued)

(Continuation)

```
                    3B691100000057920201010000100
 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               = 008F0E000000000000000000000004000034909000
 cardservicelocations      = com.sun.opencard.service.common
 CyberFlex.ATR             = 3B169481100601810F 3B169481100601811F
 country                   = US
 debugging.filename        = /tmp/ocf_debugfile
 language                  = en
 debugging                 = 0
```

Server Properties

Server properties define operations of the smart cards server `ocfserv` on each host. This section explains each server property listed by `smartcard -c admin` and tells how to change the default value of the property, if applicable.

You can change the default server property by using the following syntax of the `smartcard -c admin` command.

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

where:

- `-x modify` – Indicates that you want to perform a modify operation.
- "*property_name=property_value*" – Represents the property to be modified and the value you want to assign to it.

Server Authentication Mechanism Property

The `authmechanism` property defines the mechanism `ocfserv` uses to authenticate the user, as shown in:

```
 authmechanism  = Pin Password
```

The possible authentication mechanisms are:

- `Pin` – Indicates that the host must prompt the user for a PIN during login as its mechanism for authenticating the user. The typed PIN must match the PIN embedded on the smart card.
- `Password` – Indicates that the host must authenticate the user by reading the password on the smart card. and verifying that this password is in the host's password database (NIS, NIS+, or local files). If `ocfserv` does not find a password stored on the card, it prompts the user to type a password, which it then compares against the passwords in the host's password database.
- `ChallengeResponse` – Indicates that the host must initiate a challenge-response conversation between itself and the smart card as its means of authenticating the user.

The default authentication mechanism for Solaris Smart Cards is Pin Password, as explained in "What Happens During a Smart Card Login" on page 1-2.

▼ To Change the Default Authentication Mechanism

1. **Log in as root on the host where you want to change the authentication mechanism.**

2. **Change the authentication mechanism to ChallengeResponse, for example, by typing:**

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

If you type **`smartcard -c admin`**, you can verify on the resulting display that the challenge-response mechanism has been added.

```
authmechanism  = ChallengeResponse
```

How Challenge-Response Authentication Works

Unless you explicitly configure challenge-response as the server authentication mechanism, `ocfserv` uses PIN Password by default.The *challenge-response* is the conversation that occurs between the host and the smart card. When you use `smartcard -c init` to add a user to the card, the host automatically generates a symmetric key (DES key) and puts a copy of the key on the card. The host then

stores the symmetric key in the `/etc/smartcard/.keys` file. (See “Preparing the Smart Card for Use on Multiple Hosts” on page 3-10.)

When you configure `ocfserv` to use challenge-response authentication, the host initiates the challenge by generating a random number and sending it to the card inserted in the reader. The card uses its symmetric key to generate its response to the host, so that the host can verify if the response is correct.

Supported Card Readers Property

The `OpenCard.terminals` property defines the card readers configured for the host. For example, for a host with a Sun Smart Card Reader I, the value for `OpenCard.terminals` is:

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

Here `OpenCard.terminals` defines the Sun Smart Card Reader I as the currently configured reader. You see the `OpenCard.terminals` property displayed by `smartcard -c admin` only after you have added a card reader. For instructions on adding a card reader, see “Setting Up Card Readers” on page 2-4.

Server Protocol Property

The `ocfserv.protocol` property defines the TCP/IP protocol used by `ocfserv`:

```
ocfserv.protocol          = rpc
```

`ocfserv` uses the remote procedure call (RPC) protocol; do not change this value.

Smart Card Answer to Reset Properties

The answer to reset (ATR) properties contain numeric values supplied by the smart card manufacturers. The following ATR properties are defined for the cards supported by Solaris Smart Cards:

```
PayFlex.ATR    = 3B690000579202010100010A9
             3B691100000057920201010001 00
IButton.ATR      = 008F0E000000000000000000000004000034909000
CyberFlex.ATR  =3B169481100601810F 3B169481100601811F
```

You do not need to change the ATR properties unless the card manufacturer issues a new card type with a different ATR. For instructions, see “To Change the ATR Property” on page 4–3.

`authservicelocations` Property

This property defines the location of the Java Class directory containing the authentication module:

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

Do not change this value.

Card Services Properties

This is a Java Class directory where the card-specific modules are located. Each smart card type has the following modules defined:

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

It is unlikely that you will have to change these values. If you do, refer to the `smartcard`(1M) man page for complete details.

`initializerlocations` Property

This property defines the location of the Java Class directory containing the applet initializer

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

:

Do not change this value.

Card Service Location Property

This property defines the location of the Java Class directory where the card service module is located.

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

Do not change this value.

Locale-Specific Properties

You can define two locale-specific properties to `ocfserv`:

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

▼ To Change the Default Locale of a Smart Card Host

1. **Log in as root on each host where you want to change locale-specific information.**

2. **Specify the appropriate country for this host by typing:**

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

   where *country_code* is the two-character country code appropriate for this host.

3. **Specify the appropriate language for the host by typing:**

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

   where *language_code* is the two-character language code appropriate for this host's locale.

Debugging Properties

You can debug smart card operations on a host by setting the debugging properties. Solaris Smart Cards offers standard debugging and a detailed trace of your operations, if specified. By default, the following debugging properties are defined for `ocfserv`:

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

where:

- `/tmp/ocf_debugfile` is the name of the file to contain debugging information.
- The default debugging level 0 indicates that debugging is turned off; level 1 indicates that debugging is turned on.

▼ To Enable `ocfserv` Debugging

1. **Log in as root on the host where you want to debug `ocfserv`.**

2. **Turn on debugging by typing:**

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

3. **(Optional) Change the location of the `ocfserv` debugging file by typing:**

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

   where *filename* is the fully qualified file name for the debugging file.

4. **(Optional) Start the opencard trace of `ocfserv` activities by typing:**

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

   where *debug_level* is a value from 0 – 9.

Note - For complete information about debugging Solaris Smart Cards, refer to the `smartcard`(1M) man page.

Client Properties

Client properties define how the host should handle security requirements for application programs, such as `dtlogin`.

Default Card Types for Client Applications Properties

Two card properties designate which smart card types must be used to log in to a particular client application, or all client applications on the host: `defaultcard` and `validcards`.

The `validcards` property specifies all smart card types that are valid for a particular application. By contrast, the `defaultcard` tells the application to wait until the card defined as the *defaultcard* is loaded into the reader.

For example, suppose you specify iButton, Cyberflex, and CardA as the `validcards` properties for Application B. Then you specify Cyberflex as the `defaultcard` property. Now, if Application B accepts only its default card and the user tries to log in to Application B with CardA, then the host displays the message:

```
Waiting for Default Card
```

Log in to Application B is blocked until the user inserts a Cyberflex card into the reader.

When you run `smartcard -c admin`, these values are displayed:

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

▼ To Change the Valid Smart Cards for an Application

1. **Log in as root.**

2. **Change the default valid cards by typing**:

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

   where: *IButton*|*CyberFlex* |*PayFlex* indicates any one of these values or combination of values.

For example, to define the valid smart card types for all applications as CyberFlex and Payflex, you type:

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

▼ To Assign a Default Smart Card to an Application

1. **Log in as root on the host with the client properties you want to modify.**

2. **Assign a default smart card type to an application by typing:**

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

where:

- *application_name* is the application you want to define a default smart card for.
- *card_name* is the card type that must be used to log in to this application, either `CyberFlex`, `PayFlex`, or `IButton`.

For example, to define iButton as the default card type for a host's desktop, you type:

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

Thereafter, when you run `smartcard -c admin`, you see the following client properties:

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

Default Authentication Mechanism for Client Applications Properties

Solaris Smart Cards provides the authmechanism property to define the authentication mechanism to be used by a client application program. The `default.authmechanism` property specifies the default authentication mechanisms for all client applications. By default this mechanism is Pin=UserPin. You also can use authmechanism to define the authentication mechanism to be used for a specific client application.

▼ To Change the Default Authentication Mechanism for All Client Programs

1. **Log in as root on the host with the properties you want to modify.**

2. **Change the default authentication mechanism by typing all on one line:**

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

where *Pin* | *Password* | *ChallengeResponse* can be any one of these values or a combination of them.

For example, if you want the default authentication mechanism for client programs to be both Pin and Password, you type:

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

Thereafter, when you type `smartcard -c admin`, you see the following default authentication mechanisms:

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

Note - If a discrepancy exists between the assigned client authentication mechanisms and those assigned to `ocf.server.authmechanism`, the client authentication mechanisms have preference over those assigned to `ocfserv`.

▼ To Assign an Authentication Mechanism to a Particular Client Application

The *application_name*`.authmechanism` property enables you to assign a specific authentication mechanism for a particular application.

1. **Log in as root on the host.**

2. **Assign the authentication mechanism for the client program by typing:**

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

where:

- *application_name* is the name of the application requiring a specific authentication mechanism.
- *mechanism* is the mechanism to use: Pin, Password, ChallengeResponse, or any combination of these three.

  For example, you use the following to have the desktop require a challenge-response conversation with the smart card before the user can log in.

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

When you next run `smartcard -c admin`, you can see the new property:

```
dtlogin.authmechanism     = ChallengeResponse
```

Default Applet Identification Property

The default applet identification (AID) property is an ID number assigned to the default smart card applet to run for every application. The default ID number shown by `smartcard -c admin` is:

```
default.defaultaid        = A000000062030400
```

This value is the AID for SolarisAuthApplet, the applet run by default by Solaris Smart Cards.

You should not need to change the `defaultaid` property unless you need to replace it with an applet custom-built for your site. If this is the case, refer to the `smartcard`(1M) man page for assistance.

Enabling Smart Card Operations on the Host

The final step in host preparation is to turn on smart card operations. To do this, you need to make some changes to the desktop environment.

▼ To Enable Smart Card Operations on a Host

1. **Log in as root on each host to be used in smart card operations.**

2. **Stop the desktop by typing:**

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

3. **Turn on smart card operations by typing:**

```
# smartcard -c enable
```

4. **Restart the desktop by typing:**

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

5. **(Optional) Reboot the host.**

   The next time some one logs in to this host, the user has to use the accepted smart card for the host and possibly type a PIN. See Chapter 5 for information about logging in with a smart card.

CHAPTER 3

Initializing Smart Cards

This chapter explains how to *initialize a smart card*, that is, prepare the card for use in your security domain. Solaris Smart Cards support three types of smart cards:

- Cyberflex
- Payflex
- iButton

This chapter contains tasks for:

- Creating the necessary infrastructure on the smart card
- Setting up the properties to personalize a smart card for the user

Loading the Smart Card Infrastructure

The first step in setting up a smart card involves downloading the necessary infrastructure onto the card. If your site configuration is a security domain, you can prepare smart cards for users on their individual hosts or on your machine. For a badge office, you prepare all smart cards on the reader attached to a host specifically designated for that purpose.

Note - The host machine you use for initializing a smart card must already be configured for smart cards operations. If you have not done so yet, refer to Chapter 2.

▼ To Load the Card Infrastructure

Use this command to load infrastructure onto all card types supported by Solaris Smart Cards.

1. **Insert the card into the reader.**
2. **Log in as root on the machine to be used for initializing smart cards.**
3. **Load the Solaris Smart Card capx file, and set up card infrastructure by typing:**

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

When `smartcard -c load` finishes creating the infrastructure, it displays the message:

```
Operation successful.
```

Initializing a Smart Card for a User

After the card infrastructure has been loaded, you initialize the smart card for an individual. Solaris Smart Cards include a set of properties that you can configure on the smart card for a particular user.

▼ To List Properties You Can Configure on the Smart Card

1. **Insert the smart card into the card reader.**
2. **Log in on the host.**
3. **List the configurable properties by typing:**

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

where the `-A` option tells `smartcard -c init` to list the configurable properties available from the applet with the AID A000000062030400. This AID

specifies the default SolarisAuthApplet capx file introduced in “To Load the Card Infrastructure” on page 3–2.

Use `smartcard -c init` to display the following list of properties:

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

Defining Properties on the Smart Card

You need to set properties on the individual smart cards based on the user’s requirements, your site’s security policies, and the limitations of the type of smart card used. In Chapter 2, you defined properties for the `ocfserv` server and the client applications running on the host. Using the `smartcard -c init` command, you define corresponding properties for the individual smart cards. The client and server programs on the host read the properties on the smart card to determine whether to give the user access to a particular application.

TABLE 3–1 Properties That Can Be Initialized on a Smart Card

Property	Card Type Supported	Description
PIN	All	Personal ID number. See “PIN Property” on page 3-4.
Password	All	User’s password, as it exists on the host or domain’s password database. See “User and Password Properties” on page 3-5.
User	All	User’s account name, as it exists on the host or domain’s password database. See “User and Password Properties” on page 3-5.
Application	All	Application that requires the user to log in with the information on this smart card. See “Application Property” on page 3-6.

TABLE 3–1 Properties That Can Be Initialized on a Smart Card *(continued)*

Property	Card Type Supported	Description
Private Key	Cyberflex iButton	Private key to be used when signing files. See “Private-Key Property” on page 3-8.
Certificate	Cyberflex iButton,	Configurable property, but Solaris Smart Cards has no interface defined for certificates. For more information, refer to the `smartcard`(1M) man page.

Note - These properties apply only to cards initialized with the SolarisAuthApplet provided with Solaris Smart Cards. If your site uses a different smart-card applet, the available properties might be different. Refer to the `smartcard`(1M) man page for more information.

PIN Property

The PIN property is an authentication property on the smart card that defines a personal ID number (PIN) for the card. The `smartcard -c load` command creates the default PIN `$$$$java` on the card. You or the user can change `$$$$java` to a personalized PIN. Consider giving the users cards with the default PIN name or similar sequence, for example, changeme, in use at your site. Users can later change this PIN, as described in “To Change the PIN on Your Smart Card” on page 5–3.

Note - Although you can define more than one user name and password combination on a smart card, the card can only have one PIN.

▼ To Initialize a PIN for a Smart Card

This procedure is appropriate for all cards supported by Solaris Smart Cards. You can initialize the PIN property for more than one smart card at a time by repeating the procedure.

1. **Log in as root on the host where you are initializing smart cards.**
2. **Ensure that the smart card you want to initialize is in the reader.**
3. **Set the PIN for the card by typing:**

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

where *PIN_number* represents the PIN number you want to set for the card, needed for verifying the identity of the user.

How the PIN Property Works

The default authentication mechanism for `ocfserv` and individual applications is PIN Password. In this scenario, the user tries to log in to an application, such as the desktop. The application requests that the user type a PIN.

The `ocfserv` server verifies the authenticity of the user by comparing the PIN typed by the user to the PIN on the smart card. If the PINs match, the user is either given access to the application, or `ocfserv` reads additional authentication properties on the card.

User and Password Properties

The user and password properties are additional authentication properties on the card. They identify the user and associate the user with the smart card's PIN. Before setting these properties, you must obtain the user accounts and their associated passwords for everyone at your site who will use smart cards.

▼ To Initialize a Password on the Smart Card

This procedure is appropriate for all smart-cards devices supported by Solaris Smart Cards. You can initialize the user and password properties for more than one smart card at a time by repeating the procedure.

1. **Log in as root on the host where you are initializing smart cards.**

2. **Ensure that the smart card you want to initialize is in the reader.**

3. **Set the user name and password for the card by typing on one line:**

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

where:

- *PIN_number* – Represents the PIN assigned to the card, which is needed to verify the identity of the user.
- *user_name* – Is the individual's UNIX login name.
- *user_password* – Is the password associated with *user_name*. This password must be in the password database defined by a host's `/etc/nsswitch.conf` file (NIS, NIS+, or local files).
- `application=` *application_name* – Designates the application that requires the smart-card login with this PIN and password.

How the User and Password Properties Work

On hosts using the default authentication mechanism of PIN Password, `ocfserv` verifies the authenticity of the PIN as explained in "How the PIN Property Works" on page 3-5. In addition, `ocfserv` reads the user and password properties on the card. If the password on the smart card matches a password in the host's password database, the user is given access to the application.

You can define more than one user and password combination on the smart card. For example, the owner of the smart card might need different user names and passwords to access different applications. Moreover, for a system administrator, you can define a standard user and password, and password for root on the card. However, the smart card must have only one PIN.

Application Property

Use the application property to designate the specific applications that the user needs to log in to with a particular user name and password. For example, to require a smart card-based login to the desktop, you must specify `dtlogin` as the application associated with the user name and password on the card. You can also require smart-card-based logins for an application specific to your site, such as a financial package or personnel database, by specifying their names as the application property.

Before you initialize an application on the card, find out which applications a user needs to access through smart card authentication. This is particularly important when preparing a smart card for a system administrator or other individual who might need to log in to an application as root or other restricted user name.

▼ To Initialize an Application on the Smart Card

This procedure is appropriate for all smart card devices supported by Solaris Smart Cards. You can initialize the application property for more than one smart card at a time by repeating the procedure.

1. Log in as root on the machine where you initialize smart cards.

2. **Ensure that the smart card you want to initialize is in the reader.**

3. **Initialize the application on the smart card by typing on one line:**

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

where:

- `-P` *PIN* – Is the PIN assigned to this card.
- `user=`*user_name* – Defines the user name to be used when logging in to this application.
- `password=`*password* – Defines the password to be used when logging in to this application.
- `application=` *application_name* – Designates the application that requires the smart card login with this PIN and password.

How the Application Property Works

The application property on the card works in tandem with the three authentication properties. For example, suppose you initialized a smart card for user Frank with the following information:

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

where:

- `-A A000000062030400` – Is the SolarisAuthApplet.
- `-P '$$$$java'` – Is the PIN for this card, in this case the default PIN, which user Frank can change later.
- `application=dtlogin` – Is the application to require the smart card login.
- `user=frank` is the name Frank must give to log in to the desktop (`dtlogin` application).
- `password=changeme` – Is the password user Frank must type to log in to the desktop.

When Frank inserts his card into the reader and tries to log in to the host (`dtlogin`), the `ocfserv` server reads the card to check if any authentication properties are

associated with `dtlogin`. `ocfserv` finds that the user and password properties are associated with `dtlogin`.

`ocfserv` prompts Frank for his PIN, and then compares the typed PIN with the PIN property on the smart card assigned to the `dtlogin` application. `ocfserv` also uses the user name and password on Frank's card, along with the passwords in the host's password database, to verify that Frank is whom he claims to be. If these properties compare successfully, Frank is logged in to the desktop.

Private-Key Property

To use this feature of Solaris Smart Cards, you must have a public-key infrastructure (PKI) such as Authentication Management Infrastructure (AMI) set up at your site. AMI is a feature of the Solaris 8 operating environment. Refer to the *Authentication Management Infrastructure Administration Guide* if you want to use AMI to create public keys for your site.

Note - You can only store one private key on a smart card.

▼ To Initialize a Private Key on the Smart Card

This procedure is appropriate for the Java-based iButton and Cyberflex smart cards. However, you cannot store a private key on the Payflex card.

1. **Create a public/private-key pair for the user using the appropriate commands for your PKI.**

2. **Export the private-key part of the key pair into a separate file.**

   Record the fully qualified path name of the file because you have to specify it when setting up the private-key property.

3. **Log in as root on the machine used for initializing smart cards.**

4. **Insert a card into the smart card reader.**

5. **Access the Java security directory by typing:**

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

6. **Edit the** `java.security` **file.**

7. **Locate the** `security.provider` **definition in the file:**

```
 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. **Ensure that there is a comment sign (#) before** `security.provider.<n>=<className>`**.**

9. **Add the following text so that the file looks like:**

```
# 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. **Restart the servers by typing:**

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

11. **Initialize the card by typing on one line:**

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

where:

- *PIN_number* – Represents the PIN assigned to the card.

- *keyfile_name* – Is the full path name of the file containing the user's private key.

Note - The certificate property is not fully implemented by the SolarisAuthApplet.

How the Private-Key Property Works

After authenticating the PIN and password on the card, the `ocfserv` server copies the file specified in *keyfile_name* to the smart card. Thereafter, the private key is available on the key for signing data as an additional form of authentication. When the user runs a command for signing data, such as `amisign` from AMI, the command uses the private key on the user's smart card to create the signed data.

Depending on your site's policies, you might want to delete the user's private-key file from the host where it is stored.Thereafter, the private key exists only on the user's smart card.

Preparing the Smart Card for Use on Multiple Hosts

When you run the `smartcard -c init` command to initialize a user's smart card, you create a symmetric key on the host and on the smart card. `ocfserv` creates a file called `/etc/smartcard/.keys` that contains information about all secret keys configured on a host. If the user needs to access hosts other than the host where the smart card was created, you need to export the `/etc/smartcard/.keys` file to all hosts the user must access.

▼ To Export the Keys File

Use this procedure for exporting the `/etc/smartcard/.keys` from the host where the card was created.

1. **Log in as root on the host where the card was created.**

2. ***For a badge office site,* create a separate key file for this user, containing only that user's keys as shown in `/etc/smartcard/.keys`.**

3. **Export the `/etc/smartcard/.keys` by typing:**

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

where *key_file_name* represents the name of the file containing the individual user's symmetric key, either `/etc/smartcard/.keys` or another file specifically for that user.

▼ To Import the Keys

Use this procedure to import the user's symmetric key onto a different host than the host where the user's card was created.

1. **Log in to the host as root.**

2. **Import the key file to the new host by typing:**

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

   where *key_file_name* is either `/etc/smartcard/.keys` or other file that you created for the user.

3. **Repeat the first two steps on every host that the user must access through the smart card.**

CHAPTER 4

Maintaining Smart Cards

This chapter describes maintenance tasks for Solaris Smart Cards that you have to perform periodically to keep your security domain or badge office running smoothly. Topics covered include:

- Reinstalling packages
- Updating properties on the host or smart card
- Clearing the contents of a smart card
- Removing a card reader
- Disabling a smart card login to the desktop

Removing or Reinstalling Smart Card Packages

Table 4–1 lists the Solaris Smart Cards packages added during Solaris 8 installation.

TABLE 4–1 Solaris Smart Card Packages

Package Name	Description
SUNWjcom	Java Communications API for smart card support - Java code and Native code
SUNWjcomx	Java Communications API for smart card support - Native code (64-bit)

TABLE 4–1 Solaris Smart Card Packages *(continued)*

Package Name	Description
SUNWjib	Dallas Semiconductor serial iButton OCF Card Terminal Driver
SUNWocf	Open Card Framework - core libraries and utilities
SUNWocfr	Open Card Framework - configuration files
SUNWocfx	Open Card Framework - 64 bit core libraries
SUNWscgui	Solaris Smart Cards graphical user interface
SUNWpamsc	Pluggable Authentication Module for smart card authentication
SUNWpamsx	Pluggable Authentication Module for smart card authentication
SUNWscmsc	Sun External Smart Card Reader I OCF card terminal drive
SUNWocfh	Open Card Framework header files
SUNWscmos	Pluggable Authentication Module for smart card authentication

Should you need to remove a package, use the standard UNIX `pkgrm` command. Reinstall the package using the `pkgadd` command.

Maintaining Smart Cards

This section describes maintenance tasks for smart cards and hosts.

Updating to a New Smart Card Release

You need to change the answer to reset property (ATR) on a host if the manufacturer of the smart card used by your site issues a new card type with a different ATR. (See “Smart Card Answer to Reset Properties” on page 2-11 for an explanation of ATRs.) Change this property on every host that needs to accept the new card.

▼ To Change the ATR Property

1. **Refer to documentation from the card manufacturer to obtain the new ATR for your smart card.**

2. **Log in as root on each host to accept the new smart card type.**

3. **Change the smart card ATR by typing:**

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

   where:

   - *card_name* is either PayFlex, CyberFlex, or IButton.
   - *ATR_number* is the new ATR number assigned to the card.

Reusing a Smart Card

Smart cards are reusable. If you want a previously initialized smart card to support a different user or application, you can clear the contents of the card. The steps for clearing the card differ for the two supported external card readers.

Note - Make sure that you really want to clear the contents of the smart card before using the procedures in this section. All information on the card will be removed.

▼ To Clear the Contents of an iButton

1. **Log in as root on the machine with the iButton reader.**

2. **Make sure that the iButton you want to clear is inserted in the reader.**

3. **List the user-friendly name of the currently attached iButton reader by typing:**

```
# smartcard -c admin
```

4. **Search the resulting display for the line beginning with** `OpenCard.terminals`**.**

   The second value after the equal sign (=) is the user-friendly name. For example, in the line:

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

The value `iButtonAdapter` is the user-friendly name assigned to the attached iButton reader. You need the user-friendly reader name in the next step.

5. **Clear the iButton by typing:**

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

▼ To Clear the Contents of a Smart Card Using the Sun Smart Card Reader I

You cannot explicitly clear the data from a Cyberflex or Payflex smart card inserted in a Sun Smart Card Reader I. Instead, you clear the card by reloading the `SolarisAuthApplet.capx` file.

1. **Log in as root on the machine with the Sun Smart Card Reader I.**
2. **Make sure that the card you want to clear is inserted in the reader.**
3. **Clear the card by typing on one line:**

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

When completed, `ocfserv` displays the message:

```
Operation successful
```

Removing a Card Reader

You might need to remove a card reader from a host, for example, when a user no longer needs to use smart cards, or when you want to move the reader to another machine. Before you unplug the reader, you have to logically remove the reader, as well.

▼ To Remove a Card Reader

1. **Log in as root to the machine with the card reader to be disabled.**

2. **Logically remove the card reader by typing:**

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

3. **Unplug the card reader from the port.**

Disabling Smart Card Operations

You might need to disable smart card operations on a host if the user forgets the PIN for the smart card or if the user no longer requires a smart card login.

▼ To Disable Smart Cards

The following task assumes that the user has forgotten the smart card PIN and cannot log in to the desktop.

1. **Halt the machine, and then reboot in single-user mode.**

2. **Disable smart card operations by typing:**

```
# smartcard -c disable
```

3. **Exit single-user mode; have the machine resume its boot process and return to the desktop environment.**

CHAPTER 5

Using Smart Cards

This chapter describes how to use a smart card. It is written for anyone who needs to use a smart card to log in to the Solaris 8 operating environment.

Before Using Your Smart Card

Smart cards are resources that protect your Solaris desktop or individual applications in a more secure manner than the familiar UNIX login with a password. You use the smart card to *authenticate* yourself to the desktop or application, that is, to prove that you are who you claim to be.

What Is on the Smart Card

Solaris Smart Cards software supports three types of smart cards: the Cyberflex, iButton, and Payflex cards. Your system administrator will configure a smart card reader on your computer, and then provide you with the smart card used in your organization.

Your smart card contains:

- Your UNIX user name
- (Optional) Your password
- A personal ID number (PIN) for the smart card
- Names of the application programs that require you to log in with the PIN on your smart card
- (Optional) A private key to use for signing files

Information Required by Your Security Administrator

Depending on your site's policies, your security administrator initializes your smart card either on your machine or in a badge office. You need to give the administrator this information:

1. Your UNIX user name.
2. Your password.
3. Names of any applications in addition to the Solaris desktop that require you to log in with your password.
4. Possibly a preferred PIN, although this depends on your site's policies. Your administrator might give you a default PIN and ask you to change it.

Using Your Smart Card to Log in to the Desktop

When you receive your smart card from the security administrator, you can use it immediately.

▼ To Log in to the Solaris Desktop Using Your Smart Card

1. **Insert your smart card into the card reader, if you have not done so already.**

   The Solaris desktop environment prompts you for your PIN.

2. **Type the PIN provided by the security administrator, either your preferred PIN or a default PIN.**

   After you type the correct PIN, one of the following happens:

   - If the smart card contains your password, you automatically are logged in to the desktop.

- If your smart card does not contain your user name and password, the desktop prompts you for them, as in a standard UNIX login. After you type the correct user name and password, you are logged in to the desktop.

▼ To Log in to a Protected Application Using Your Smart Card

1. **Insert your smart card into the card reader, if you have not done so already.**

2. **Log in as described in "To Log in to the Solaris Desktop Using Your Smart Card" on page 5–2.**

3. **Run the protected application.**

   The application prompts you for your PIN.

4. **Type the PIN provided by the security administrator, either your preferred PIN or a default PIN.**

   After you type the correct PIN, one of the following occurs:

   - If the smart card contains your password, you automatically access the application.
   - If your smart card does not contain your user name and password, the application prompts you for them, as in a standard UNIX login. After you type the correct user name and password, you can access the application.

▼ To Change the PIN on Your Smart Card

1. **Insert your smart card in the card reader, if you have not done so already.**

2. **Change your PIN by typing**:

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

   where:

   - *old_PIN* is your current PIN.
   - *new_PIN* is your new PIN.

Index

D

E

G

H

I

J

K

L

N

O

P

R

S

T

U

V