man pages section 9: DDI and DKI Overview

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

Part No: 835-8008
December 2000

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Contents

Preface

Overview

A man page is provided for both the naive user and the sophisticated user who is familiar with the Trusted Solaris operating environment and is in need of online information. A man page is intended to answer concisely the question “What does it do?” The man pages in general comprise a reference manual. They are not intended to be a tutorial.

Trusted Solaris Reference Manual

In the AnswerBook2™ and online man command forms of the man pages, all man pages are available:

- Trusted Solaris man pages that are unique for the Trusted Solaris environment
- SunOS 5.8 man pages that have been changed in the Trusted Solaris environment
- SunOS 5.8 man pages that remain unchanged.

The printed manual, the *Trusted Solaris 8 Reference Manual* contains:

- Man pages that have been added to the SunOS operating system by the Trusted Solaris environment
- Man pages that originated in SunOS 5.8, but have been modified in the Trusted Solaris environment to handle security requirements.

Users of printed manuals need both manuals in order to have a full set of man pages, since the *SunOS 5.8 Reference Manual* contains the common man pages that are not modified in the Trusted Solaris environment.

Man Page Sections

The following contains a brief description of each section in the man pages and the information it references:

- Section 1 describes, in alphabetical order, commands available with the operating system.
- Section 1M describes, in alphabetical order, commands that are used chiefly for system maintenance and administration purposes.
- Section 2 describes all of the system calls. Most of these calls have one or more error returns. An error condition is indicated by an otherwise impossible returned value.
- Section 3 describes functions found in various libraries, other than those functions that directly invoke UNIX system primitives, which are described in Section 2 of this volume.
- Section 4 outlines the formats of various files. The C structure declarations for the file formats are given where applicable.
- Section 5 contains miscellaneous documentation such as character set tables.
- Section 6 contains available games and demos.
- Section 7 describes various special files that refer to specific hardware peripherals, and device drivers. STREAMS software drivers, modules and the STREAMS-generic set of system calls are also described.
- Section 9 provides reference information needed to write device drivers in the kernel operating systems environment. It describes two device driver interface specifications: the Device Driver Interface (DDI) and the Driver/Kernel Interface (DKI).
- Section 9E describes the DDI/DKI, DDI-only, and DKI-only entry-point routines a developer may include in a device driver.
- Section 9F describes the kernel functions available for use by device drivers.
- Section 9S describes the data structures used by drivers to share information between the driver and the kernel.

Below is a generic format for man pages. The man pages of each manual section generally follow this order, but include only needed headings. For example, if there are no bugs to report, there is no BUGS section. See the `intro` pages for more information and detail about each section, and `man`(1) for more information about man pages in general.

NAME

This section gives the names of the commands or functions documented, followed by a brief description of what they do.

SYNOPSIS

This section shows the syntax of commands or functions. When a command or file does not exist in the standard path, its full pathname is shown. Options and arguments are alphabetized, with single letter arguments first, and options with arguments next, unless a different argument order is required.

The following special characters are used in this section:

[ ] The option or argument enclosed in these brackets is optional. If the brackets are omitted, the argument must be specified.

. . . Ellipses. Several values may be provided for the previous argument, or the previous argument can be specified multiple times, for example, ' "`filename`. . ." .

| Separator. Only one of the arguments separated by this character can be specified at a time.

{ } Braces. The options and/or arguments enclosed within braces are interdependent, such that everything enclosed must be treated as a unit.

PROTOCOL

This section occurs only in subsection 3R to indicate the protocol description file.

DESCRIPTION

This section defines the functionality and behavior of the service. Thus it describes concisely what the command does. It does not discuss OPTIONS or cite EXAMPLES. Interactive commands, subcommands, requests, macros, functions and such, are described under USAGE.

IOCTL

This section appears on pages in Section 7 only. Only the device class which supplies appropriate parameters to the ioctl (2) system call is called `ioctl` and generates its own heading. `ioctl` calls for a specific device are listed alphabetically (on the man page for that specific device). `ioctl` calls are used for a particular class of devices all of which have an `io` ending, such as `mtio`(7I)

OPTIONS

This secton lists the command options with a concise summary of what each option does. The options are listed literally and in the order they appear in the SYNOPSIS section. Possible arguments to options are discussed under the option, and where appropriate, default values are supplied.

OPERANDS

This section lists the command operands and describes how they affect the actions of the command.

OUTPUT

This section describes the output – standard output, standard error, or output files – generated by the command.

RETURN VALUES

If the man page documents functions that return values, this section lists these values and describes the conditions under which they are returned. If a function can return only constant values, such as 0 or –1, these values are listed in tagged paragraphs. Otherwise, a single paragraph describes the return values of each function. Functions declared void do not return values, so they are not discussed in RETURN VALUES.

ERRORS

On failure, most functions place an error code in the global variable `errno` indicating why they failed. This section lists alphabetically all error codes a function can generate and describes the conditions that cause each error. When more than one condition can cause the same error, each condition is described in a separate paragraph under the error code.

USAGE

This section lists special rules, features, and commands that require in-depth explanations. The subsections listed here are used to explain built-in functionality:

- Commands
- Modifiers
- Variables
- Expressions
- Input Grammar

EXAMPLES

This section provides examples of usage or of how to use a command or function. Wherever possible a complete example including command-line entry and machine response is shown. Whenever an example is given, the prompt is shown as `example%`, or if the user must be root, `example#`. Examples are followed by explanations, variable substitution rules, or returned values. Most examples illustrate concepts from the SYNOPSIS, DESCRIPTION, OPTIONS, and USAGE sections.

ENVIRONMENT VARIABLES

This section lists any environment variables that the command or function affects, followed by a brief description of the effect.

EXIT STATUS

This section lists the values the command returns to the calling program or shell and the conditions that cause these values to be returned. Usually, zero is returned for successful completion, and values other than zero for various error conditions.

FILES

This section lists all file names referred to by the man page, files of interest, and files created or required by commands. Each is followed by a descriptive summary or explanation.

ATTRIBUTES

This section lists characteristics of commands, utilities, and device drivers by defining the attribute type and its corresponding value. See `attributes`(5) for more information.

SUMMARY OF TRUSTED SOLARIS CHANGES

This section describes changes to a Solaris item by Trusted Solaris software. It is present in man pages that have been modified from Solaris software.

SEE ALSO

This section lists references to other man pages, in-house documentation and outside publications. The references are divided into two sections, so that users of printed manuals can easily locate a man page in its appropriate printed manual.

DIAGNOSTICS

This section lists diagnostic messages with a brief explanation of the condition causing the error.

WARNINGS

This section lists warnings about special conditions which could seriously affect your working conditions. This is not a list of diagnostics.

NOTES

This section lists additional information that does not belong anywhere else on the page. It takes the form of an aside to the user, covering points of special interest. Critical information is never covered here.

BUGS

This section describes known bugs and, wherever possible, suggests workarounds.

CHAPTER

Device Driver Interfaces

10

NAME

Intro – Introduction to device driver interfaces

DESCRIPTION

Section 9 provides reference information needed to write device drivers for Solaris 2, Solaris 7, Solaris 8, and Trusted Solaris versions 2.5.1, 7, and 8. It describes the interfaces provided by the Device Driver Interface Driver-Kernel Interface (DDI/DKI).

This reference information can be:

- Information that is unique to and originates in the Trusted Solaris environment, such as `tsol_linkb`(9F). Each new Trusted Solaris–specific device driver interface is identified by the `tsol_` prefix on its name.
- `SunOS` 5.8 device drivers that have been modified to work within Trusted Solaris security policy, such as `copyb`(9F). Man pages for modified device drivers have been rewritten to remove information that is not accurate for how the system call behaves within the Trusted Solaris environment. Modified man pages also have added descriptions for Trusted Solaris security features.
- `SunOS` 5.8 device drivers that remain unchanged from the Solaris 8 release.

The printed *Trusted Solaris 8 Reference Manual* includes only those man pages that have been modified or originate in the Trusted Solaris environment. Printed versions of unchanged `SunOS` 5.8 man pages are found in the *SunOS 5.8 Reference Manual.*

Trusted Solaris drivers can call any of the functions described in 9E, 9F, or 9S.

Porting

Software is usually considered portable if it can be adapted to run in a different environment more cheaply than it can be rewritten. The new environment may include a different processor, operating system, and even the language in which the program is written, if a language translator is available. Likewise the new environment might include multiple processors. More often, however, software is ported between environments that share an operating system, processor, and source language. The source code is modified to accommodate the differences in compilers or processors or releases of the operating system.

In the past, device drivers did not port easily for one or more of the following reasons:

- To enhance functionality, members had been added to kernel data structures accessed by drivers, or the sizes of existing members had been redefined.
- The calling or return syntax of kernel functions had changed.
- Driver developers did not use existing kernel functions where available, or relied on undocumented side effects that were not maintained in the next release.

- Architecture-specific code had been scattered throughout the driver when it could have been isolated.

Operating systems are periodically reissued to customers as a way to improve performance, fix bugs, and add new features. This is probably the most common threat to compatibility encountered by developers responsible for maintaining software. Another common problem is upgrading hardware. As new hardware is developed, customers occasionally decide to upgrade to faster, more capable computers of the same family. Although they may run the same operating system as those being replaced, architecture-specific code may prevent the software from porting.

Scope of Interfaces

Although application programs have all of the porting problems mentioned, developers attempting to port device drivers have special challenges. Before describing the DDI/DKI, it is necessary to understand the position of device drivers in operating systems.

Device drivers are kernel modules that control data transferred to and received from peripheral devices but are developed independently from the rest of the kernel. If the goal of achieving complete freedom in modifying the kernel is to be reconciled with the goal of binary compatibility with existing drivers, the interaction between drivers and the kernel must be rigorously regulated. This driver/kernel service interface is the most important of the three distinguishable interfaces for a driver, summarized as follows:

- Driver–Kernel. I/O System calls result in calls to driver entry point routines. These make up the kernel-to-driver part of the service interface, described in Section 9E. Drivers may call any of the functions described in Section 9F. These are the driver-to-kernel part of the interface.
- Driver–Hardware. All drivers (except software drivers) must include code for interrupt handling, and may also perform direct memory access (DMA). These and other hardware-specific interactions make up the driver/hardware interface.
- Driver–Boot/Configuration Software. The interaction between the driver and the boot and configuration software is the third interface affecting drivers.

Scope of the DDI/DKI

The primary goal of the DDI/DKI is to facilitate both source and binary portability across successive releases of the operating systems on a particular machine. In addition, it promotes source portability across implementations of UNIX on different machines, and applies only to implementations based on System V Release 4. The DDI/DKI consists of several sections:

- DDI/DKI Architecture Independent - These interfaces are supported on all implementations of System V Release 4.

- DKI-only - These interfaces are part of System V Release 4, and may not be supported in future releases of System V. There are only two interfaces in this class, `segmap`(9E) and `hat_getkpfnum`(9F)
- Solaris DDI - These interfaces specific to Solaris.
- Solaris SPARC specific DDI - These interfaces are specific to the SPARC processor, and may not be available on other processors supported by Solaris.
- Solaris IA specific DDI - These interfaces are specific to the IA processor, and may not be available on other processors supported by Solaris.

To achieve the goal of source and binary compatibility, the functions, routines, and structures specified in the DDI / DKI must be used according to these rules.

- Drivers cannot access system state structures (for example, `u` and `sysinfo`) directly.
- For structures external to the driver that may be accessed directly, only the utility functions provided in Section 9F should be used. More generally, these functions should be used wherever possible.
- The headers `<sys/ddi.h>` and `<sys/sunddi.h>` must be the last header files included by the driver.

Audience

Section 9 is for software engineers responsible for creating, modifying, or maintaining drivers that run on this operating system and beyond. It assumes that the reader is familiar with system internals and the C Programming Language.

PCMCIA Standard

The *PC Card 95 Standard* is listed under the `SEE ALSO` heading in some Section 9 reference pages. This refers to documentation published by the Personal Computer Memory Card International Association (PCMCIA) and the Japan Electronic Industry Development Association (JEIDA).

How to Use Section 9

Section 9 is divided into three subsections:

`9E` Driver Entry Points – contains reference pages for all driver entry point routines.

9E contains no routines modified for the Trusted Solaris environment.

`9F` Kernel Functions – contains reference pages for all driver support routines.

The Trusted Solaris environment modifies some routines and adds its own.

`9S` Data Structures – contains reference pages for driver-related structures.

9S contains no routines modified for the Trusted Solaris environment.

SEE ALSO

Trusted Solaris 8 Reference Manual

`intro`(9F)

SunOS 5.8 Reference Manual

`intro`(9E), `intro`(9S)

NOTES

Solaris' implementation of the DDI/DKI was designed to provide binary compatibility for third-party device drivers across currently supported hardware platforms across minor releases of the operating system.

However, unforeseen technical issues may force changes to the binary interface of the DDI/DKI. We cannot therefore promise or in any way assure that DDI/DKI-compliant device drivers will continue to operate correctly on future releases.

Name Description

Index