

SECOND EDITION

Solaris[™] Internals

SOLARIS 10 AND OPENSOLARIS
KERNEL ARCHITECTURE

The Solaris logo features a stylized sun icon above the word "solaris" in a lowercase, sans-serif font.The Sun Microsystems logo consists of a diamond-shaped icon with a grid pattern, followed by the word "Sun" in a script font and "microsystems" in a smaller, lowercase sans-serif font below it.

Richard McDougall and Jim Mauro

Foreword by Bryan Cantrill

Solaris™ Internals: Solaris 10 and OpenSolaris Kernel Architecture Second Edition

Richard McDougall
Jim Mauro

Sun Microsystems Press



Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Sun Microsystems, Inc., has intellectual property rights relating to implementations of the technology described in this publication. In particular, and without limitation, these intellectual property rights may include one or more U.S. patents, foreign patents, or pending applications. Sun, Sun Microsystems, the Sun logo, J2ME, Solaris, Java, Javadoc, NetBeans, and all Sun and Java based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc., in the United States and other countries. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd.

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. THIS PUBLICATION COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THE PUBLICATION. SUN MICROSYSTEMS, INC., MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS PUBLICATION AT ANY TIME.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact: U.S. Corporate and Government Sales, (800) 382-3419, corpsales@pearsontechgroup.com.

For sales outside the U.S., please contact International Sales, international@pearsoned.com.

Visit us on the Web: www.prenhallprofessional.com

Library of Congress Cataloging-in-Publication Data

McDougall, Richard.

Solaris internals : solaris 10 and OpenSolaris kernel architecture /
Richard McDougall, Jim Mauro.—2nd ed.

p. cm.

Mauro's name appears first on the earlier edition.

Includes bibliographical references and index.

ISBN 0-13-148209-2 (hardback : alk. paper)

1. Operating systems (Computers) 2. Solaris (Computer file) I. Mauro,
Jim. II. Title.

QA76.76.O63M37195 2006

005.4'465—dc22

2006015114

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, write to:

Pearson Education, Inc.

Rights and Contracts Department

One Lake Street

Upper Saddle River, NJ 07458

Fax: (201) 236-3290

ISBN 0-13-148209-2

Text printed in the United States on recycled paper at Courier in Westford, Massachusetts.

First printing, July 2006

Dedication

*For Traci, Madi, and Boston—
for your love, encouragement, and support . . .
—Richard*

*Once again . . .
For Donna, Frank, and Dominick.
All my love, always . . .
—Jim*

Contents

Foreword

Preface

About the Authors

Acknowledgments

PART ONE Introduction to Solaris Internals

Chapter 1 Introduction

1.1 Key Features of Solaris 10, Solaris 9, and Solaris 8

1.1.1 Solaris 10

1.1.2 Solaris 9

1.1.3 Solaris 8

1.2 Key Differentiators

1.3 Kernel Overview

1.3.1 Solaris Kernel Architecture

1.3.2 Modular Implementation

1.4 Processes, Threads, and Scheduling

1.4.1 A New Threads Model

1.4.2 Global Process Priorities and Scheduling

1.5 Interprocess Communication

1.5.1 Traditional UNIX IPC

1.5.2 System V IPC

1.5.3 POSIX IPC

1.5.4 Solaris Doors: Advanced Solaris IPC

1.6 Signals

1.7 Memory Management

1.7.1 Global Memory Allocation

1.7.2 The Cyclic Page Cache

1.7.3 Kernel Memory Management

1.8 Files and File Systems

1.9 Resource Management

1.9.1 Processor Controls and Domains

1.9.2 Solaris Resource Management

1.9.3 Internet Protocol Quality of Service

1.9.4 Resource Management and Observability

PART TWO The Process Model

Chapter 2 The Solaris Process Model

- 2.1 Components of a Process
 - 2.1.1 Thread Objects
 - 2.1.2 Core Process Components
- 2.2 Process Model Evolution
 - 2.2.1 Thread Model Evolution
 - 2.2.2 Unified Process Model
- 2.3 Executable Objects
- 2.4 Process Structures
 - 2.4.1 The `proc` Structure
 - 2.4.2 User Area
 - 2.4.3 Lightweight Processes (LWPs)
 - 2.4.4 Kernel Threads
- 2.5 Kernel Process Table
 - 2.5.1 Process Limits
 - 2.5.2 Thread Limits
- 2.6 Process Resource Attributes
- 2.7 Process Creation
- 2.8 System Calls
 - 2.8.1 System Calls on SPARC Architectures
 - 2.8.2 A Tour through a System Call
- 2.9 Process Termination
 - 2.9.1 LWP and Kernel Thread Exit
 - 2.9.2 Deathrow List
- 2.10 The Process File System
 - 2.10.1 Procfs Implementation
 - 2.10.2 Process Resource Usage
 - 2.10.3 Microstate Accounting
- 2.11 Signals
 - 2.11.1 Signals Implementation
 - 2.11.2 Observing Signal Activity
 - 2.11.3 Summary
- 2.12 Sessions and Process Groups
- 2.13 MDB Reference

Chapter 3 Scheduling Classes and the Dispatcher

- 3.1 Fundamentals
- 3.2 Processor Abstractions

3.2.1 Processor Observability

3.3 Dispatcher Queues, Structures, and Variables

3.3.1 Dispatcher Structures

3.3.2 Dispatcher Structure Linkage

3.3.3 Examining Dispatcher Structures

3.4 Dispatcher Locks

3.4.1 Dispatcher Lock Functions

3.4.2 Thread Locks

3.4.3 Thread Lock Functions

3.4.4 Lock Statistics

3.5 Dispatcher Initialization

3.6 Scheduling Classes

3.6.1 Scheduling Class Data

3.6.2 Scheduling Class Functions

3.6.3 Scheduling Class Dispatcher Tables

3.7 Thread Priorities

3.7.1 Global Priorities

3.7.2 User Priorities

3.7.3 Setting Thread Priorities

3.8 Dispatcher Functions

3.8.1 Dispatcher Queue Management

3.8.2 The Heart of the Dispatcher: `swtch()`

3.9 Preemption

3.10 The Kernel Sleep/Wakeup Facility

3.10.1 Condition Variables

3.10.2 Sleep Queues

3.10.3 The Sleep Process

3.10.4 The Wakeup Mechanism

3.11 Interrupts

3.11.1 Interrupt Priorities

3.11.2 Interrupts as Threads

3.11.3 Interrupt Thread Priorities

3.11.4 High-Priority Interrupts

3.11.5 Interrupt Management

3.11.6 Interrupt Monitoring

3.11.7 Interprocessor Interrupts and Cross-Calls

3.12 Summary

3.13 MDB Reference

Chapter 4 Interprocess Communication

4.1 The System V IPC Framework

4.1.1 IPC Objects

4.1.2 IPC Framework Design

4.1.3 Locking

4.1.4 Module Creation

4.2 System V IPC Resource Controls

4.2.1 The Solution

4.3 Configuring IPC Tuneables on Solaris 10

4.4 System V Shared Memory

4.4.1 Shared Memory Kernel Implementation

4.4.2 Intimate Shared Memory (ISM)

4.4.3 Dynamic ISM Shared Memory

4.5 System V Semaphores

4.5.1 Semaphore Kernel Resources

4.5.2 Kernel Implementation of System V Semaphores

4.5.3 Semaphore Operations

4.6 System V Message Queues

4.6.1 Kernel Resources for Message Queues

4.6.2 Kernel Implementation of Message Queues

4.7 POSIX IPC

4.7.1 POSIX Shared Memory

4.7.2 POSIX Semaphores

4.7.3 POSIX Message Queues

4.8 Solaris Doors

4.8.1 Doors Overview

4.8.2 Doors Implementation

4.9 MDB Reference

Chapter 5 Process Rights Management

5.1 Then and Now

5.2 Least Privilege in Solaris

5.3 Process Privilege Models

5.3.1 The Traditional Solaris Superuser Model

5.3.2 Extending Solaris with Process Privileges

5.3.3 How the Solaris 10 Least Privilege Model Was Chosen

5.3.4 Other UNIX Implementations

5.4 Privilege Awareness: The Details

5.4.1 Per-Process State

5.4.2 Privilege Awareness State Transitions

5.4.3 Privilege State Manipulation

5.4.4 Privilege Escalation Prevention

5.4.5 The Trouble with uid 0

5.4.6 Basic Privileges

5.4.7 Privileges and the Runtime Environment

5.4.8 Privileges and NFS

5.4.9 Privileges and Third-Party File Systems

5.5 Least Privilege Interfaces

5.5.1 The Conspiracy of Bit Sets and Constants

5.5.2 Privilege Names and Constants

5.5.3 Kernel Data Structures

5.5.4 Kernel Interfaces

5.5.5 System Call Interfaces

5.5.6 Library Interfaces

5.5.7 Using Privileges with Role-Based Access Control

5.5.8 Using Privileges with Role-Based Access Control

5.5.9 Using DTrace for Tracking Privileges

5.5.10 Enhancements to `proc(4)` and Core Dumps

5.5.11 Privilege Debugging

5.5.12 Privilege Auditing

5.5.13 Device Protection

PART THREE Resource Management

Chapter 6 Zones

6.1 Introduction

6.1.1 Zone Basics

6.1.2 Zone Principles

6.2 Zone Runtime

6.2.1 Zone State Model

6.2.2 Zone Names and Numeric IDs

6.2.3 Zone Runtime Support

6.2.4 Listing Zone Information

6.3 Booting Zones

6.4 Security

6.4.1 Credential Handling

6.4.2 Fine-Grained Privileges

6.4.3 Role-Based Access Control

6.5 Process Model

6.5.1 Signals and Process Control

6.5.2 Global Zone Visibility and Access

6.5.3 /proc

6.5.4 Core Files

6.6 File Systems

6.6.1 Configuration

6.6.2 Size Restrictions

6.6.3 File System-Specific Issues

6.6.4 File System Traversal Issues

6.7 Networking

6.7.1 Partitioning

6.7.2 Interfaces

6.7.3 IPv6

6.7.4 IPsec

6.7.5 Raw IP Socket Access

6.7.6 DLPI Access

6.7.7 Routing

6.7.8 TCP Connection Teardown

6.8 Devices

6.8.1 Device Categories

6.8.2 /dev and /devices Namespace

6.8.3 Device Management: Zone Configuration

6.8.4 Device Management: Zone Runtime

6.8.5 Zone Console Design

6.8.6 ftpd

6.9 Interprocess Communication

6.9.1 Pipes, STREAMS, and Sockets

6.9.2 Doors

6.9.3 Loopback Transport Providers

6.9.4 System V IPC

6.9.5 POSIX IPC

6.10 Resource Management and Observability

6.10.1 Performance

6.10.2 Solaris Resource Management Interactions

6.10.3 Kstats

6.11 MDB Reference

Chapter 7 Projects, Tasks, and Resource Controls

7.1 Projects and Tasks Framework

7.1.1 Introduction

7.1.2 Projects

7.1.3 Tasks

7.1.4 Why We Added Tasks to Solaris

7.2 The Project Database

7.3 Project and Task APIs

7.3.1 Interfaces for Projects and Tasks

7.4 Kernel Infrastructure for Projects and Tasks

7.4.1 System Call Interaction with Projects

7.4.2 `proc(4)`

7.4.3 In-Kernel Project Data Structures

7.5 Resource Controls

7.5.1 Introduction to Resource Controls

7.5.2 What Is an `rctl`?

7.5.3 Numeric Values of Resource Controls

7.5.4 Resource Control Definitions

7.5.5 Policy

7.5.6 Consequences of Exceeding an `rctl`

7.5.7 Signal and `siginfo` Semantics for Exceeded Controls

7.5.8 Generalizing Hard and Soft Limits

7.5.9 Resource Controls and the Task

7.5.10 Visibility through `/proc`; Privileges and Ownership

7.6 Interfaces for Resource Controls

7.6.1 Project Name-Service Attributes

7.6.2 Attributes Originating within Solaris

7.6.3 Grammar for Attributes

7.6.4 Interpretation of `rctl` Attributes

7.6.5 An Example `/etc/project`

7.6.6 System Calls and Private Kernel Interfaces

7.6.7 Library Functions

7.7 Kernel Interfaces for Resource Controls

7.7.1 Data Structures

7.7.2 Operations Vector

7.7.3 Interface Overview

7.7.4 Interface Definitions

7.7.5 An Example Resource Control

Chapter 8 Introduction to Solaris Memory

- 8.1 Virtual Memory Primer
- 8.2 Two Levels of Memory
- 8.3 Memory Sharing and Protection
- 8.4 Pages: Basic Units of Physical Memory
- 8.5 Virtual-to-Physical Translation
- 8.6 Physical Memory Management: Paging and Swapping
- 8.7 Virtual Memory as a File System Cache
- 8.8 New Features of the Virtual Memory Implementation

Chapter 9 Virtual Memory

- 9.1 Design Overview
- 9.2 Virtual Address Spaces
 - 9.2.1 Sharing Executables and Libraries
 - 9.2.2 Address Spaces on SPARC Systems
 - 9.2.3 x86 and x64 Address Space Layout
 - 9.2.4 Growing the Heap
 - 9.2.5 The Stack
 - 9.2.6 Using `pmap` to Look at Mappings
- 9.3 Tracing the VM System
- 9.4 Virtual Address Space Management
 - 9.4.1 Address Space Management
 - 9.4.2 Address Space Callbacks
 - 9.4.3 Virtual Memory Protection Modes
 - 9.4.4 Page Faults in Address Spaces
- 9.5 Segment Drivers
 - 9.5.1 The vnode Segment: `seg_vn`
 - 9.5.2 Copy-on-Write
 - 9.5.3 Page Protection and Advice
- 9.6 Anonymous Memory
- 9.7 The Anonymous Memory Layer
- 9.8 The `swapfs` Layer
 - 9.8.1 `swapfs` Implementation
- 9.9 Virtual Memory Watchpoints
- 9.10 Changes to Support Large Pages
 - 9.10.1 System View of a Large Page
 - 9.10.2 Free List Organization
 - 9.10.3 Large-Page Faulting

9.10.4 Large-Page Freeing

9.10.5 Operations That Interfere with Large Pages

9.10.6 HAT Support

9.10.7 procfs Changes

9.11 MDB Reference

Chapter 10 Physical Memory

10.1 Physical Memory Allocation

10.1.1 The Allocation Cycle of Physical Memory

10.2 Pages: The Basic Unit of Solaris Memory

10.2.1 The Page Hash List

10.2.2 Page Structures

10.2.3 Free List and Cache List

10.2.4 Physical Page “memseg” Lists

10.2.5 The Page-Level Interfaces

10.2.6 The Page Throttle

10.2.7 Page Coloring

10.3 The Page Scanner

10.3.1 Page Scanner Operation

10.3.2 Page-Out Algorithm and Parameters

10.3.3 Shared Library Optimizations

10.3.4 Parameters That Limit Pages Paged Out

10.3.5 Page Scanner Implementation

10.3.6 The Memory Scheduler

10.4 MDB Reference

Chapter 11 Kernel Memory

11.1 Kernel Virtual Memory Layout

11.1.1 Kernel Address Space

11.1.2 Kernel Text and Data Segments

11.1.3 Virtual Memory Data Structures

11.1.4 UltraSPARC Kernel Nucleus

11.1.5 Loadable Kernel Module Text and Data

11.1.6 The Kernel Address Space and Segments

11.2 Kernel Memory Allocation

11.2.1 The Kernel Heap

11.2.2 The Kernel Memory Segment Driver

11.2.3 The Kernel Memory Slab Allocator

11.3 The Vmem Allocator

11.3.1 Background

11.3.2 Vmem Objectives

11.3.3 Interface Description

11.3.4 Vmem Implementation

11.3.5 Vmem Performance

11.3.6 Summary

11.4 Kernel Memory Allocator Tracing

11.4.1 Enabling KMA DEBUG Flags

11.4.2 Examining Kernel Memory Allocations with MDB

11.4.3 Detecting Memory Corruption

11.4.4 Checking a Freed Buffer: 0xdeadbeef

11.4.5 Debugging with the Redzone Indicator: 0xfeedface

11.4.6 Detecting Uninitialized Data: 0xbaddcafe

11.4.7 Associating Panic Messages with Failures

11.4.8 Memory Allocation Logging

11.4.9 Analyzing Memory with Advanced Techniques

11.4.10 Finding Corrupt Buffers with `::kmem_verify`

11.4.11 Using the Allocator Logging Facility

11.5 MDB Reference

Chapter 12 Hardware Address Translation

12.1 HAT Overview

12.2 The UltraSPARC HAT Layer

12.2.1 Introduction

12.2.2 `struct hat`

12.2.3 The Translation Table

12.2.4 The Translation Storage Buffer (TSB)

12.2.5 Intimate Shared Memory (ISM)

12.2.6 Synchronization in the HAT Layer

12.2.7 SPARC HAT Layer Kernel Tunables

12.2.8 SPARC Hat Layer `kstats`

12.3 The x64 HAT Layer

12.3.1 MMU Configuration

12.3.2 `struct mmu` Variable

12.3.3 Virtual Address Space Layout

12.3.4 64-Bit Address Space Layout

12.3.5 32-Bit Address Space Layout

12.3.6 HAT Implementation

12.4 MDB Reference

Chapter 13 Working with Multiple Page Sizes in Solaris

13.1 Determining When to Use Large Pages

13.2 Measuring Application Performance

13.2.1 Determination Allocated Page Sizes

13.2.2 Discovery of Supported Page Sizes

13.3 Configuring for Multiple Page Sizes

13.3.1 Enabling Large Pages

13.3.2 Advising Page-Size Preferences with `ppgsz(1M)`

13.3.3 Interposing Shared Libraries with `libmpss.so`

13.3.4 Request Larger Page Sizes with the Compiler

13.3.5 Interfaces to Request Larger Page Sizes

13.3.6 CPU Specific Large Page Support

PART FIVE File Systems

Chapter 14 File System Framework

14.1 File System Framework

14.2 Process-Level File Abstractions

14.2.1 File Descriptors

14.2.2 The `open` Code Path

14.2.3 Allocating and Deallocating File Descriptors

14.2.4 File Descriptor Limits

14.2.5 File Structures

14.3 Solaris File System Framework

14.3.1 Evolution of the File System Framework

14.3.2 The Solaris File System Interface

14.4 File System Modules

14.4.1 Interfaces for Mount Options

14.4.2 Module Initialization

14.5 The Virtual File System (`vfs`) Interface

14.5.1 `vfs` Methods

14.5.2 `vfs` Support Functions

14.5.3 The `mount` Method

14.5.4 The `umount` Method

14.5.5 Root `vnode` Identification

14.5.6 `vfs` Information Available with `MDB`

14.6 The Vnode

14.6.1 Object Interface

14.6.2 `vnode` Types

14.6.3 `vnode` Method Registration

14.6.4 vnode Methods

14.6.5 Support Functions for Vnodes

14.6.6 The Life Cycle of a Vnode

14.6.7 vnode Creation and Destruction

14.6.8 The vnode Reference Count

14.6.9 Interfaces for Paging vnode Cache

14.6.10 Block I/O on vnode Pages

14.6.11 vnode Information Obtainable with mdb

14.6.12 DTrace Probes in the vnode Layer

14.7 File System I/O

14.7.1 Memory Mapped I/O

14.7.2 read() and write() System Calls

14.7.3 The seg_kpm Driver

14.7.4 The seg_map Driver

14.7.5 Interaction between segmap and segkpm

14.8 File Systems and Memory Allocation

14.8.1 Solaris 8—Cyclic Page Cache

14.8.2 The Old Allocation Algorithm

14.8.3 The New Allocation Algorithm

14.8.4 Putting It All Together: The Allocation Cycle

14.9 Path-Name Management

14.9.1 The lookupn() Method

14.9.2 The vop_lookup() Method

14.9.3 The vop_readdir() Method

14.9.4 Path-Name Traversal Functions

14.10 The Directory Name Lookup Cache

14.10.1 DNLC Operation

14.10.2 Primary DNLC Support Functions

14.10.3 DNLC Negative Cache

14.10.4 DNLC Directory Cache

14.10.5 DNLC Housekeeping Thread

14.10.6 DNLC Statistics

14.11 The File System Flush Daemon

14.12 File System Conversion to Solaris 10

14.13 MDB Reference

Chapter 15 The UFS File System

15.1 UFS Development History

15.2 UFS On-Disk Format

15.2.1 On-Disk UFS Inodes

15.2.2 UFS Directories

15.2.3 UFS Hard Links

15.2.4 Shadow Inodes

15.2.5 The Boot Block

15.2.6 The Superblock

15.2.7 The Cylinder Group

15.2.8 Summary of UFS Architecture

15.3 The UFS Inode

15.3.1 In-Core UFS Inodes

15.3.2 Inode Cache

15.3.3 Block Allocation

15.3.4 Methods to Read and Write UFS Files

15.4 Access Control in UFS

15.5 Extended Attributes in UFS

15.6 Locking in UFS

15.6.1 UFS Lock Descriptions

15.6.2 Inode Lock Ordering

15.6.3 UFS Lockfs Protocol

15.7 Logging

15.7.1 On-Disk Log Data Structures

15.7.2 In-Core Log Data Structures

15.7.3 Summary Information

15.7.4 Transactions

15.7.5 Rolling the Log

15.7.6 Redirecting Reads and Writes to the Log

15.7.7 Failure Recovery

15.8 MDB Reference

PART SIX Platform Specifics

Chapter 16 Support for NUMA and CMT Hardware

16.1 Memory Hierarchy Designs

16.1.1 What Is NUMA?

16.1.2 What Is CMT?

16.2 Memory Placement Optimization Framework

16.2.1 Latency Model

16.2.2 More Complex Models

16.3 Initial Thread Placement

16.4 Scheduling

16.5 Memory Allocation

16.6 Lgroup Implementation

16.6.1 Parameters Affecting MPO

16.7 MPO APIs

16.7.1 Informational

16.7.2 Verifying the Interface Version

16.7.3 Initialization of the Locality Group Interface

16.8 Locality Group Hierarchy

16.8.1 Locality Group Characteristics

16.8.2 Locality Groups and Thread and Memory Placement

16.9 MPO Statistics

16.10 MDB Reference

Chapter 17 Locking and Synchronization

17.1 Synchronization

17.2 Parallel Systems Architectures

17.3 Hardware Considerations for Locks and Synchronization

17.4 Introduction to Synchronization Objects

17.4.1 Synchronization Process

17.4.2 Synchronization Object Operations Vector

17.5 Mutex Locks

17.5.1 Overview

17.5.2 Solaris Mutex Lock Implementation

17.6 Reader/Writer Locks

17.6.1 Solaris Reader/Writer Locks

17.7 Turnstiles and Priority Inheritance

17.7.1 Turnstiles Implementation

17.8 Kernel Semaphores

17.9 DTrace Lockstat Provider

17.9.1 Overview

17.9.2 Adaptive Lock Probes

17.9.3 Spin Lock Probes

17.9.4 Thread Locks

17.9.5 Readers/Writer Lock Probes

PART SEVEN Networking

Chapter 18 The Solaris Network Stack

18.1 STREAMS and the Network Stack

18.1.1 The STREAMS Model

18.1.2 Network Stack as STREAMS Module

18.1.3 Issues with STREAMS-Based Stacks

18.2 Solaris 10 Stack: Design Goals

18.3 Solaris 10 Network Stack Framework

18.3.1 Vertical Perimeter

18.3.2 IP Classifier

18.3.3 Synchronization Mechanism

18.4 TCP as an Implementation of the New Framework

18.4.1 The Interface between TCP and IP

18.4.2 TCP Loopback

18.5 UDP

18.5.1 UDP Packet Drop within the Stack

18.5.2 UDP Module

18.5.3 UDP and Socket Interaction

18.6 Synchronous STREAMS

18.6.1 TCP Synchronous STREAMS

18.6.2 STREAMS Fallback

18.7 IP

18.7.1 Plumbing NICs

18.7.2 IP Network Multipathing

18.7.3 Multicast

18.8 Solaris Device Driver Framework

18.8.1 GLDv2 and DLPI Drivers (Solaris 9 and Prior)

18.8.2 A New Architecture: GLDv3

18.8.3 GLDv3 Link Aggregation Architecture

18.8.4 Checksum Offload

18.9 Interrupt Model and NIC Speeds

18.9.1 Solaris 9 and Earlier Releases

18.9.2 Dynamic Switch between Interrupt vs. Polling Mode

18.9.3 Interrupt Load Spreading

18.10 Summary

18.11 MDB Reference

PART EIGHT Kernel Services

Chapter 19 Clocks and Timers

19.1 The System Clock Thread

19.1.1 Thread Tick Processing

19.1.2 DTrace Providers for Tick Processing

19.2 Callouts and Callout Tables

19.3 System Time Facilities

19.3.1 High-Resolution Timer

19.3.2 Time-of-Day Clock

19.4 The Cyclic Subsystem

19.4.1 Cyclic Subsystem Interface Overview

19.4.2 Cyclic Subsystem Implementation Overview

19.4.3 Clients of the Cyclic Subsystem

19.4.4 Cyclic Kernel At-Large Interfaces

19.4.5 Cyclic Kernel Inter-Subsystem Interfaces

19.4.6 Cyclic Backend Interfaces

19.4.7 Cyclic Subsystem Backend-Supplied Interfaces

Chapter 20 Task Queues

20.1 Overview of Task Queues

20.2 Dynamic Task Queues

20.2.1 Why a *Dynamic* Task Queue?

20.2.2 Problems Addressed by Dynamic Task Queues

20.2.3 Task Pool Model

20.2.4 Interface Changes to Support Dynamic Task Queues

20.3 Task Queues Kernel Programming Interfaces

20.4 Device Driver Interface for Task Queues

20.5 Task Queue Observability

20.5.1 Kstat Counters

20.5.2 DTrace SDT Probes

20.6 Task Queue Implementation Notes

20.6.1 Use of Kmem Caches

20.6.2 Use of Vmem Arenas

20.6.3 Hashed Vmem Arenas

20.6.4 Cached List of Entries

20.6.5 Problems with Task Pool Implementation

20.6.6 Use of Dynamic Task Pools in STREAMS

Chapter 21 kldb Implementation

21.1 Introduction

21.1.1 MDB Components

21.1.2 Major kldb Design Decisions

21.1.3 The Structure of kldb

21.1.4 MDB Components and Their Implementation in kldb

APPENDICES

Appendix A Kernel Virtual Address Maps

Appendix B Adding a System Call to Solaris

Appendix C A Sample Procfs Utility

Bibliography

Index

- [download online The Trace here](#)
- [click Even Money](#)
- [The Idea of Continental Philosophy pdf](#)
- [read online Pelletier: The Forgotten Castaway of Cape York here](#)
- [click Stillness and Light: The Silent Eloquence of Shaker Architecture pdf, azw \(kindle\), epub, doc, mobi](#)
- **[The Complete Short Stories pdf, azw \(kindle\)](#)**

- <http://korplast.gr/lib/El-Croquis--Issue-158--John-Pawson--2006-2011--the-Voice-of-Matter.pdf>
- <http://www.1973vision.com/?library/Even-Money.pdf>
- <http://www.netc-bd.com/ebooks/Safe-C---How-to-avoid-common-mistakes.pdf>
- <http://korplast.gr/lib/Sex--Drugs--and-Cocoa-Puffs.pdf>
- <http://dadhoc.com/lib/Skyscraper.pdf>
- <http://conexdx.com/library/Heavy-Metal--A-Tank-Company-s-Battle-to-Baghdad.pdf>