# HP OpenVMS
# V8.3 新機能説明書

**注文番号**: BA322-90059

2006 年 10 月

本書では，OpenVMS Alpha および OpenVMS I64 Version 8.3 オペレーティング・システムの新機能と，このソフトウェアのドキュメントの概要について説明します。

**改訂/更新情報**: 新規マニュアルです。

**ソフトウェア・バージョン**: OpenVMS I64 Version 8.3
OpenVMS Alpha Version 8.3

**日本ヒューレット・パッカード株式会社**

原典： HP OpenVMS Version 8.3 New Features and Documentation Overview


本書は，日本語 VAX DOCUMENT V 2.1を用いて作成しています。

# 目次

## 図



## 表

# まえがき

## 対象読者

本書は， HP OpenVMS オペレーティング・システムを使用する一般ユーザ，システム管理者，プログラマを対象としています。

本書では OpenVMS Version 8.3 の新機能を説明します。新機能が使用中のシステムに与える影響について， V8.3 をインストール，アップグレード，使用する前にリリース・ノートをお読みください。

## 本書の構成

本書の構成は以下のとおりです。

- 第 1 部， OpenVMS Version 8.3 の新機能
  - 第 1 章， OpenVMS の新機能について概要を説明しています。
  - 第 2 章， OpenVMS オペレーティング・システムの一般ユーザ向けの新機能について説明しています。
  - 第 3 章，システム管理作業に関する新機能について説明しています。
  - 第 4 章， COPY/RECORDABLE_MEDIA (CDDVD) ユーティリティについて説明しています。
  - 第 5 章，プログラミングの新機能について説明しています。
  - 第 6 章， InfoServer ユーティリティについて説明しています。このユーティリティは， Integrity サーバ対応の OpenVMS だけでなく OpenVMS Alpha でもサポートされるようになりました。
  - 第 7 章，重要なレイヤード・プロダクトの新機能について説明しています。
- 第 2 部， OpenVMS のドキュメント
  - 第 8 章， OpenVMS ドキュメントの変更点について説明しています。
  - 第 9 章，ドキュメントの入手方法について説明しています。
  - 第 10 章， OpenVMS ドキュメント・セットに含まれている各ドキュメントについて説明しています。

# 第1部
# OpenVMS Version 8.3の新機能

# 1

# HP OpenVMS Version 8.3 の新機能の概要

OpenVMS Version 8.3 は，1 年 365 日 24 時間体制 (24 × 365) で休みなく稼働する環境にとって不可欠な，現時点で最高レベルの可用性，拡張性，柔軟性，性能，セキュリティを提供します。OpenVMS では，基本オペレーティング・システムと OpenVMS Cluster に新技術を採用するとともに，新しい HP Integrity サーバをサポートすることで，可用性と性能が向上しています。

## 1.1 新機能のまとめ

OpenVMS Version 8.3 では，OpenVMS Version 8.2 および Version 8.2–1 のすべての機能に加えて，表 1–1 に示す新機能が追加されています。表 1–1 では，OpenVMS Version 8.3 に用意されている各機能の概要を機能要素 (一般ユーザ，システム管理，プログラミング，および関連製品) 別に示します。

**表 1–1 OpenVMS Version 8.3 ソフトウェアの機能概要**

| 一般ユーザ機能 | 説明 |
|---|---|
| 新しい Integrity サーバ・システムのサポート | 新しいエントリ・レベル，ミッドレンジ，ハイエンドの Integrity サーバがサポートされました。 |
| バッチ・キューの上限を拡大 | ジョブ・リミットが 65535 に拡大されました。 |
| 遠隔プロセスに対する Ctrl/T のサポート | 新しいシンボル DCL$CTRLT_PID を定義し，遠隔プロセス ID を指定できるようになりました。 |
| Ctrl/T の出力のカスタマイズ | 新しいシンボル DCL$CTRLT が定義できるようになりました。 |
| DCL コマンドとレキシカル関数 | DCL コマンド，修飾子，レキシカル関数が追加および変更されました。 |
| DCL パーマネント・シンボル | 2 つのシンボル$FACILITY と$IDENT が追加されました。 |
| DCL プロンプト・サイズの拡大 | 最大プロンプト・サイズが 32 文字から 64 文字に拡大されました。 |
| /SINCE 修飾子へのキーワードの追加 | JOB_LOGIN キーワードが追加されました。 |
| ハイパースレッド機能 (I64 のみ) | 第 2 の仮想コアを作成することで，処理の効率がさらに高まります。 |
| HP Instant Capacity (iCAP) および HP Temporary Instant Capacity (TiCAP) (I64 のみ) | セル・ベースの Integrity サーバ向けの，新しい HP プライシング・ソリューション製品です。 |
| I/O サイズ・リミットの拡大 | COPY コマンドでのブロック・サイズの上限が拡大されました。 |
| LMF の拡張 | 準拠レポートのあて先を変更できるようになりました。 |
| HP nPartition Provider (I64 のみ) | OpenVMS Version 8.3 に移植されました。 |

(次ページに続く)

**表 1–1 (続き) OpenVMS Version 8.3 ソフトウェアの機能概要**

| 一般ユーザ機能 | 説明 |
|---|---|
| HP Pay Per Use (PPU) (I64 のみ) | セル・ベースの Integrity サーバ向けの， HP On Demand Solutions 製品です。 |
| HP Superdome サーバのサポート (I64 のみ) | OpenVMS は， PA-RISC の nPartitions と Intel® Itanium® 2 の nPartitions の両方を持つ HP sx1000 チップセット・ベースの構成をサポートすることができます。 |
| Web-Based Enterprise Management Services for OpenVMS (WBEM) | 業界標準のエンタープライズ管理フレームワークです。 |

| システム管理機能 | 説明 |
|---|---|
| BACKUP Ctrl/T メッセージ | 会話型のバックアップ操作中に，より多くの情報を表示するように拡張されました。 |
| BACKUP /IO_LOAD 修飾子 | システム上での同時読み込み I/O の回数を増加または減少させることができるようになりました。 |
| BACKUP /PROGRESS_REPORT 修飾子 | バックアップ情報を現在の出力デバイスに送ります。 |
| BACKUP セーブ・セットの暗号化 | ENCRYPT コマンドおよび DECRYPT コマンドの追加により， AES 暗号化アルゴリズムとユーザ指定の鍵を使用して， BACKUP セーブ・セットの暗号化および復号化を行います。 |
| 光媒体 CD および DVD 用のツール | 光媒体 CD および DVD に記録するためのツールが追加されました。 |
| クラスタ・インターコネクト | データの圧縮と数ギガビットの回線速度を提供します。 |
| クラスタ・サテライト・ブート | クラスタ・サテライト・ブートが， OpenVMS Alpha だけでなく，OpenVMS I64 でもサポートされるようになりました。 |
| ロックの動的な再マスタリング | ロック・ツリーの再マスタリングを判断する方法が更新されました。 |
| 動的ボリューム拡張 (DVE) のサポート | BACKUP ユーティリティに追加されました。 |
| 暗号化 | OpenVMS レイヤード・プロダクトの暗号化機能が，オペレーティング・システムに統合されました。 |
| InfoServer ユーティリティ | OpenVMS I64 だけでなく OpenVMS Alpha でもサポートされるようになりました。 OpenVMS が動作する AlphaServer および Integrity サーバ上で， InfoServer ハードウェア製品の機能を提供します。 InfoServer ユーティリティは， AlphaServer や Integrity サーバで OpenVMS オペレーティング・システムをインストールまたはアップグレードするために使用できます。 |
| MONITOR ALIGN コマンド (I64 のみ) | Monitor ユーティリティに新しいコマンドが追加されました。 |
| MONITOR の PROCESSES クラス向けのクラス名修飾子 | プロセスごとのモードの使用量を監視するために使用されます。 |
| マルチパスの機能拡張 | EVA コントローラおよび MSA コントローラの，新しいアクティブ - アクティブ機能が追加されました。 |
| パッチ関連のメニュー・オプション | OpenVMS オペレーティング・システムの配布メディアのメイン・メニューに，パッチ関連の操作を行うオプションが新たに追加されました。 |
| PCSI ユーティリティの機能拡張 | 手動検証，自動検証， ODS-5 の完全なサポート，署名済み製品キットの検証が追加されました。 |

(次ページに続く)

**表** 1–1 (**続き**) OpenVMS Version 8.3 **ソフトウェアの機能概要**

| **システム管理機能** | **説明** |
|---|---|
| SANCP ユーティリティ | 特定の Fibre Channel ストレージ・ポート上の論理ユニット番号 (LUN) へのすべてのパスにわたって，ホストあたりのアクティブな入出力の数を制限することができます。 |
| SAS ユーティリティ (I64 のみ) | HP 8 Internal Port Serial Attached SCSI Host Bus Adapter (SAS コントローラ) の IR (Integrated RAID) 機能を構成します。 |
| SCACP のデータ圧縮 | SET VC/COMPRESSION (または/NOCOMPRESSION) コマンドで，指定した仮想サーキット (VC) ごとに圧縮したデータの送信を有効または無効にすることができます。 |
| SCACP の数ギガビットのスケーリング | /WINDOW=RECEIVE_SIZE 修飾子と/WINDOW=TRANSMIT_SIZE 修飾子を使用して，自動的に計算された送受信ウィンドウ・サイズを指定変更することができます。 |
| Integrity サーバにおける HP OpenVMS I64 Serial Multiplexer (MUX) のサポート (I64 のみ) | USB を使用して Integrity サーバにシリアル回線を追加することができます。 |
| Spinlock Trace ユーティリティ | 表示が改良されました。 |
| HP System Analysis Tools ユーティリティ | System Dump Debugger が Integrity サーバ対応の OpenVMS で利用可能になりました。いくつかの新しい System Dump Analyzer コマンドと修飾子が追加されました。 |
| システム・パラメータ | 本リリースでは，以下に示すシステム・パラメータが新たに追加されています。<br>• EXECSTACKPAGES<br>• GB_CACHEALLMAX<br>• GB_DEFPERCENT<br>• IO_PRCPU_BITMAP<br>• LOCKRMWT<br>• SCH_HARD_OFFLD<br>• SCH_SOFT_OFFLD<br>• SCHED_FLAGS<br>• SMP_CPU_BITMAP<br>• SMP_PRCPU_BITMAP<br>• VCC_PAGESIZE<br>• VCC_RSVD |

(**次ページに続く**)

**表 1–1 (続き) OpenVMS Version 8.3 ソフトウェアの機能概要**

| システム管理機能 | 説明 |
|---|---|
| システム・サービス・ロギング | SSLOG が改良されました。 |
| タイム・ゾーン | 新しいタイム・ゾーンがデータベースに追加され，サポートされるようになりました。 |
| デバイス管理のための UCM の変更 | システムに差し込んだデバイスが自動的に設定されるようになりました。 |
| OpenVMS での仮想 LAN (VLAN) のサポート | OpenVMS に IEEE 802.1Q (VLAN) のサポートが追加されました。 |
| HP Volume Shadowing for OpenVMS の機能拡張 | 以下の新機能が追加されています。<br>• ボリューム処理時に自動的にビットマップを作成<br>• SET SHADOW の新しい修飾子/RESET |

| プログラミング機能 | 説明 |
|---|---|
| CDSA (Common Data Security Architecture) | Secure Delivery と HRS (Human Recognition Service Standard) のサポートが追加されています。 |
| HP C ランタイム・ライブラリ (CRTL) の機能拡張 | 以下の機能が追加されています。<br>• シンボリック・リンクと POSIX 準拠のパス名<br>• バイト単位のロック<br>• 新しい関数<br>• TCP/IP のヘッダ・ファイルの更新 |
| デッドロック待ち | 新しい項目コード PPROP$C_DEADLOCK_WAIT により，ロック・マネージャで 1 秒以下のデッドロック待ちが可能になりました。 |
| 項目コード | 以下の項目コードが新たに追加されました。<br>• DEVICE_MAX_IO_SIZE<br>• VOLUME_RETAIN_MAX<br>• VOLUME_RETAIN_MIN<br>• MAILBOX_INITIAL_QUOTE<br>• MAILBOX_BUFFER_QUOTA |
| HP Kerberos for OpenVMS の機能拡張 | Kerberos for OpenVMS に新機能が追加されました。 Kerberos Version 3.0 for OpenVMS は， MIT Kerberos V5 Release 1.4.1 を基にしています。 |
| ライブラリ・ユーティリティの機能拡張 (I64 のみ) | マングル化された名前とデマングル化された名前の一覧を表示できるようになりました。 |

(次ページに続く)

**表 1–1 (続き) OpenVMS Version 8.3 ソフトウェアの機能概要**

| プログラミング機能 | 説明 |
|---|---|
| リンカ・ユーティリティの機能拡張 | /DNI 修飾子と，/FULL 修飾子の DEMANGLED_SYMBOLS キーワード。 |
| HP MACRO Compiler for OpenVMS Alpha システムの機能拡張 | Alpha システム向けの最新の GEM バックエンドを使用するようにアップグレードされ， DCL 修飾子/ARCHITECTURE が新たに追加されるなど，機能拡張が行われました。 |
| RMS (Record Management System) の機能拡張 | いくつかの機能拡張が行われました。 |
| HP Secure Sockets Layer (SSL) for OpenVMS の機能拡張 | SSL に新機能が追加されました。 HP SSL Version 1.3 は， OpenSSL 0.9.7e を基にしています。 |
| USB 汎用ドライバ | ユーザが USB デバイスのサポートを追加するための，$QIO ベースの API が提供されました。 |

| 関連製品の機能 | 説明 |
|---|---|
| HP Distributed NetBeans for OpenVMS | デスクトップ・システム上で NetBeans IDE が実行できるようになりました。 |
| HP OpenVMS Management Station Version 3.3 | OpenVMS Alpha Version 8.3 では OpenVMS Management Station Version 3.3 が提供されます。 |
| HP Secure Web Browser for OpenVMS | OpenVMS Version 8.3 では HP Secure Web Browser for OpenVMS Version 1.7-13 が提供されます。 |
| HP Secure Web Server for OpenVMS | OpenVMS Version 8.3 では HP Secure Web Server for OpenVMS Version 2.1 が提供されます。 |
| HP TCP/IP Services for OpenVMS | OpenVMS Version 8.3 では， HP TCP/IP Services for OpenVMS Version 5.6 が提供されます。 |
| HP Web Services Integration Toolkit for OpenVMS | 一連のツールが提供されます。 |

OpenVMS Version 8.3 の使用を開始する前に，『HP OpenVMS V8.3 リリース・ノート[翻訳版]』および『HP OpenVMS V8.3 インストレーション・ガイド[翻訳版]』をお読みください。

# 2

# 一般ユーザ機能

この章では，OpenVMS Alpha オペレーティング・システムと OpenVMS I64 オペレーティング・システムのすべてのユーザに関連する新機能について説明します。

## 2.1 新しい Integrity サーバのサポート

OpenVMS Version 8.3 は新しいエントリ・レベル，ミッドレンジ，ハイエンドの Integrity サーバ・システムをサポートします。また，OpenVMS Version 8.3 は，OpenVMS Version 8.2 および Version 8.2–1 でサポートされていた，すべての Integrity サーバとオプションを引き続きサポートします。

## 2.2 バッチ・ジョブ・キューの上限を拡大

OpenVMS Version 8.3 では，バッチ・キューのジョブの上限が 255 から 65535 に拡大されています。

## 2.3 DCL コマンドとレキシカル関数

表 2–1 と表 2–2 に，OpenVMS Version 8.3 で新たに追加または変更された DCL コマンド，修飾子，およびレキシカル関数の要約を示します。DCL の使用方法に関するいくつかの新機能を以下の項で説明します。詳細は，オンライン・ヘルプまたは『OpenVMS DCL ディクショナリ』を参照してください。

**表 2–1 DCL コマンドと DCL のドキュメントの更新**

| DCL コマンド | ドキュメントの更新 |
|---|---|
| DEASSIGN | 新しい/[NO]LOG 修飾子 |
| DECRYPT | 新しいコマンド |
| DIFFERENCES | /IGNORE に対する新しい WHITE_SPACE キーワード |
| ENCRYPT | 新しいコマンド |
| READ | 新しい/WAIT 修飾子と，/MATCH に対する新しいキーワード LT および LE |
| SEARCH | /WILDCARD 修飾子に対する新しいキーワード |

(次ページに続く)

**表 2–1 (続き) DCL コマンドと DCL のドキュメントの更新**

| DCL コマンド | ドキュメントの更新 |
|---|---|
| SEARCH | /STATISTICS 修飾子を指定すると，統計情報が設定されたいくつかの DCL シンボルが定義されるようになりました。 |
| SET | 新しい/RMS_RELATED_CONTEXT 修飾子 |
| SET FILE | /ATTRIBUTES 修飾子テーブルに，新たに 7 つの*DATE キーワードが追加されました。 RMS に，新たに 5 つのグローバル・バッファ・オプションが追加されました。 |
| SHOW DEVICES | /FULL を指定した場合に，マルチパス・デバイスの最後のパス切り替え時刻が表示されるようになりました。 LAN デバイスの場合，/FULL 修飾子が更新され，デフォルトの MAC アドレス情報と現在の MAC アドレス情報，使用している LAN プロトコル (該当する場合)，データ・リンクの速度，オートネゴシエーション，二重モード，ジャンボ・フレームなど，有効になっているその他さまざまな属性が表示されるようになりました。 |
| SHOW LICENSE | OE データベース内のすべてのライセンスが 1 つのコマンドで表示されるようにコマンドが更新されました。 |
| SHOW PROCESS | Q キー・オプションが/CONTINUOUS 修飾子に追加されました。 |
| SYNCHRONIZE/TIME_OUT=n | SYNCH コマンドを終了する前に待つ秒数をユーザが指定できるようにするための新しいコマンド。 |

**表 2–2 DCL レキシカル関数とそのドキュメントに対する更新**

| DCL レキシカル関数 | ドキュメントの更新 |
|---|---|
| F$CUNITS | 新しいレキシカル関数 |
| F$FILE_ATTRIBUTES | 2 つの項目コード GBC32 および GBCFLAGS を追加 |
| F$GETDVI | 新しい項目コードを追加 |
| F$LICENSE | ライセンス作成元を指定する新しいオプション引数を追加 |
| F$MATCH_WILD | 新しいレキシカル関数 |

### 2.3.1 遠隔プロセスに対する Ctrl/T のサポート

遠隔プロセス ID を指す新しいシンボル DCL$CTRLT_PID を定義することができます。必要な特権を持っている場合は，そのプロセスの Ctrl/T 情報を表示することができます。遠隔プロセスは，同じシステム上の異なるプロセス，またはクラスタ内の別のシステム上のプロセスです。次の例は， NODE1 上の特権を持つユーザが DCL$CTRLT_PID を定義し， NODE2 上のユーザ JSMITH のプロセス情報を表示する方法を示します。

```
$ <Ctrl/T>
NODE1::SYSTEM 17:40:55   (DCL)   CPU=00:00:00.16 PF=212 IO=98 MEM=146
$
$ DCL$CTRLT_PID="23800436"  !Define symbol to point to remote process ID
$
$ <Ctrl/T>
NODE2::JSMITH 17:41:12 LOOPER    CPU=01:28:05.17 PF=2700 IO=594 MEM=322
$
```

### 2.3.2 DCL パーマネント・シンボル

イメージの停止時に，DCL は$SEVERITY シンボルと$STATUS シンボルに情報を設定します。Version 8.3 では，2 つのシンボル$FACILITY と$IDENT が新たに追加されました。これらのシンボルには，機能番号とメッセージ番号が格納されます。

```
$ EXIT  %X10911A02
$ SHOW SYMBOL $STATUS
  $STATUS == "%X10911A02"
$ SHOW SYMBOL $FACILITY
  $FACILITY == "%X00000091"
$ SHOW SYMBOL $IDENT
  $IDENT == "%X00000340"
$ SHOW SYMBOL $SEVERITY
  $SEVERITY == "2"
```

### 2.3.3 Ctrl/T の出力のカスタマイズ

新しいシンボル DCL$CTRLT を定義することで，従来の Ctrl/T の出力に DCL$CTRLT で定義されたテキストを追加することができます。ユーザ・アプリケーションの進行状況を表示したり，デバッグ目的でこの手法を使用することができます。以下の例は，コマンド・プロシージャ内での DCL$CTRLT の使用方法を示したものです。コマンド・プロシージャはループを実行し，それまでに実行したループの繰り返し回数を示すために，シンボル DCL$CTRLT を更新します。

```
$ TYPE CTRLT_LOOP.COM
$ inner=0
$ outer=0
$ loop:
$ loop1:
$ if inner .gt. 20000 then goto end_loop1
$ inner=inner+1
$ dcl$ctrlt=F$FAO("Inner loop count is !SL !/ -
_$ Outer loop count is !SL",inner,outer)
$ goto loop1
$ end_loop1:
$ inner=0
$ outer=outer+1
$ goto loop
$
$ @CTRLT_LOOP
NODE1::JSMITH 10:46:37   (DCL)   CPU=00:03:42.68 PF=13453 IO=6743 MEM=187
Inner loop count is 12306
 Outer loop count is 0
NODE1::JSMITH 10:46:43   (DCL)   CPU=00:03:49.19 PF=13455 IO=6744 MEM=187
Inner loop count is 19200
 Outer loop count is 2
 .
 .
 .
```

### 2.3.4 /SINCE 修飾子への JOB_LOGIN キーワードの追加

/SINCE 修飾子が指定できるコマンド (SHOW LICENSE を除く) では，/SINCE=JOB_LOGIN が指定できるようになりました。 JOB_LOGIN は，ジョブのマスタープロセスのログイン時刻を示します。なお， PIPE はパイプ・セグメントごとにサブプロセスを作成するため，パイプ中で/SINCE=LOGIN を使用しても効果がありません。

```
$ PIPE DIRECTORY/SINCE=LOGIN | SEARCH SYS$INPUT TEST
%SEARCH-I-NOMATCHES, no strings matched
$
$ PIPE DIRECTORY/SINCE=JOB_LOGIN | SEARCH SYS$INPUT TEST
TEST.TXT;1
```

### 2.3.5 COPY コマンドにおける I/O サイズ上限の拡大

COPY コマンドでサポートされる I/O あたりの最大ブロック数が， 2**31 –1 に拡大されました。ブロック数は，必要に応じて， I/O を実行しているデバイス・ドライバがサポートしている最大数に減らされる点に注意してください。たとえば， SCSI Fibre Channel ドライバがサポートしている最大 I/O サイズは，現時点で 256 ブロックです。

### 2.3.6 最大プロンプト・サイズの拡大

最大 DCL プロンプト・サイズが 32 文字から 64 文字に拡大され，一部のユーザで必要となる特別なプロンプトやエスケープ・シーケンスが設定できるようになりました。

## 2.4 ハイパースレッド機能 (I64 のみ)

OpenVMS for Integrity Servers Version 8.3 は，デュアルコアの Intel Itanium 2 プロセッサ上の nPartitions でハイパースレッド機能をサポートします。ハイパースレッド機能を使用すると，単一のコア上に第 2 の論理 CPU を作成し，処理の効率を高めることができます。たとえば，ハイパースレッド機能が有効なデュアルコア・プロセッサでは，各コア上に 2 つ，合計 4 つの論理 CPU を使用できます。

EFI Shell のcpuconfigコマンドを使用すると，プロセッサがハイパースレッド機能をサポートしていれば， nPartitions に対してハイパースレッド機能の有効/無効を切り替えることができます。 nPartitions コマンドと Partition Manager の最近のリリースでも，ハイパースレッド機能がサポートされています。

ハイパースレッド機能による性能への影響は，動作するアプリケーションの構成によって大きく変わります。最初はハイパースレッド機能を無効にし，後で有効にしてみることをお勧めします。ハイパースレッド機能を有効にした場合にコアを共有する

2 つの CPU は，**コスレッド**と呼ばれます。 SHOW CPU/BRIEF コマンドと SHOW CPU/FULL コマンドでは，コスレッドに関する情報も表示されるようになりました。以下に例を示します。

```
$ show cpu/brief 3

System: XYZZY, HP rx4640

CPU 3    State: RUN                CPUDB: 820DB480     Handle: 000060A0
         Owner: 000004CB8        Current: 000004C8     Partition 0 (XYZZY)
         COTHd:          1
```

この例で，COTHd: 1は CPU 3 と CPU 1 が同じコアを共有していることが分かります。

cpuconfigコマンドの出力例は次のようになります。

```
Shell> cpuconfig

PROCESSOR MODULE INFORMATION

       # of              L3      L4      Family/
CPU    Logical           Cache   Cache     Model          Processor
Module    CPUs   Speed   Size    Size     (hex.)    Rev   State
------  -------  ------  ------  ------  ---------  ---   -----
 0          4    1.4 GHz   6 MB   None    20/00      C0    Active

CPU threads are turned on.
```

cpuconfigコマンドについての詳細は，『HP OpenVMS V8.3 インストレーション・ガイド[翻訳版]』を参照してください。ハイパースレッド機能と nPartitions については，『HP **システムパーティションガイド**: nPartitions **の管理作業**』を参照してください。

## 2.5 HP iCAP (Instant Capacity) と HP TiCAP (Temporary Instant Capacity) (I64 のみ)

OpenVMS Version 8.3 で iCAP がサポートされるようになりました。 iCAP は，セル・ベースの Integrity サーバ向けの HP Utility Pricing Solutions 製品で，コンポーネント (プロセッサ，セル・ボード，およびメモリ) の購入に基づくプライシング・モデルが採用されています。 Instant Capacity では，最初に指定した数の使用開始コンポーネントを購入し，指定された数の使用停止コンポーネントに関しては CWUR (Component without Usage Rights) の費用を支払います。使用停止コンポーネントの使用を開始するには，コンポーネント価格の残額を支払い，セキュリティ保護された iCAP Web ポータルから取得したコードワードを適用することでライセンスを有効にします。それらのコンポーネントは，リブートなしですぐに使用開始することができます。

OpenVMS Version 8.3 では， TiCAP もサポートされています。この HP 製品では，プロセッサを一定の (一時的な) 期間使用する権利を前払いで購入することができます。一時的なキャパシティは， 20 日間や 30 日間といった単位で販売されています。1 日は「1 つのコアについて 24 時間」を意味します。

iCAP と TiCAP の詳細は，次の Web サイトにある『HP Instant Capacity (iCAP) ユーザーガイド』を参照してください。

http://docs.hp.com/ja/hpuxos11iv2

## 2.6 LMF (License Management Facility) の変更と機能拡張

ここでは， License Management Facility に対する変更と機能拡張について説明します。

### 2.6.1 LMF 準拠レポート

LMF 準拠レポートのあて先アカウントを変更できるようになりました。 LMF は，準拠レポートを論理名 LMF$COMPLIANCE_CONTACT_ACCOUNT で定義されたアカウントに送りますが，デフォルトのアカウントは引き続き SYSTEM アカウントです。

### 2.6.2 ライセンス用語の変更 (I64 のみ)

OpenVMS for Integrity Servers のライセンス方式と用語は， PPL (Per Processor License) から PCL (Per Core License) に変更されました。デュアルコア Intel Itanium 2 プロセッサの導入により，「プロセッサ」という用語の意味は，これまでと変わりました。デュアルコア Intel Itanium 2 システムでは，各プロセッサに 2 つのコアが含まれており，アクティブなコアの数でライセンスされます。デュアルコア Intel Itanium 2 システム以外のシステムでは，コアはプロセッサと同じ意味となります。

PCL では， OpenVMS for Integrity Servers のライセンス方式を定めています。PCL モデルでは， Alpha システムと VAX システムで使用されてきた静的な評価方式ではなく，システム上のアクティブなプロセッサ・コアの数に応じて製品がライセンスされます。それぞれのアクティブなプロセッサ・コアに対して， 1 つの PCL 単位が必要となります。システム上のアクティブなプロセッサ・コアの数を増やしたり減らしたりすると， PCL ライセンスに対する要件も変化します。

オペレーティング環境，個別に購入した OE 製品 (クラスタリングなど)， OpenVMS I64 上の多数のスタンドアロン製品を実行するために， PCL ライセンスが必要です。

PCL ライセンスでは，必要な数だけライセンスを購入でき，ライセンスを他のプロセッサに移動させることができるため，柔軟性があります。プロセッサ・コアを追加してシステムをアップグレードまたは再構成する場合は，追加の PCL ライセンスを購入します。

LMF は， PCL ライセンスの数とアクティブなプロセッサ・コアの数を定期的に確認し，準拠モデルを適用します。システムに対するあらゆる変更に対して準拠状況が確認されます。

PCL を実現するために， Hardware_ID オプションのCPU_SOCKETS=nがSOCKETS=nに変更されました。また， SHOW LICENSE/CHARGE コマンドは，システム上のアクティブなコアの数を表示するように更新されています。

本リリースでは，『OpenVMS License Management Utility Manual』は，新しい用語を使用するように更新されていません。マニュアルを読む際には，以下の用語が変っていることに注意してください。

- Per Processor License は Per Core License となりました。
- PPL は PCL となりました。
- CPU はプロセッサ・コアとなりました。
- CPU_SOCKETS=nは SOCKETS=nとなりました。

以下の定義に注意してください。

- プロセッサ — プロセッサ・ソケットに挿入するコンポーネントです。プロセッサには複数のプロセッサ・コアが含まれることがあります。
- プロセッサ・モジュール — 1 つ以上のプロセッサを，システム・バス上の 1 つのソケットに接続するためにパッケージ化したものです。
- コア — セル・ベースのプロセッサ内にある実際のデータ処理エンジンです。 1 つのプロセッサに複数のコアが含まれることがあります。
- プロセッサ・ソケット — プロセッサを取り付けるための，システム基板上のソケットです。

**注意**

**以前の PPL ライセンスも引き続きサポートされるため，変更は必要ありません。 PCL ライセンスと既存の PPL ライセンスを組み合わせることもできます。**

## 2.7 HP nPartition Provider for OpenVMS (I64 **のみ**)

HP nPartition Provider は，セル・ベースの Integrity サーバ上で Instant Capacity (iCAP) 機能をサポートするために， OpenVMS Version 8.3 に移植されました。

OpenVMS Version 8.3 の nPartition Provider は， OpenVMS Version 8.3 上で動作している nPartition Provider に対するリモート WBEM 接続など，ローカルまたはリモートでの nPartitions の管理作業をサポートしていません。

## 2.8 HP Pay Per Use (PPU) (I64 **のみ**)

OpenVMS Version 8.3 は， HP Finance からリースしているセル・ベースの Integrity サーバ・システムでの Pay Per Use をサポートしています。 PPU を使用すると，実際に消費した処理能力に対してだけ支払うようにすることができます。 PPU には次の 2 種類があります。

- パーセント CPU

  システム内のすべての CPU の使用率を定期的に監視します。

- アクティブ CPU

  システム内のアクティブ CPU の数を定期的に数えます。

システム管理者は，負荷の増加に対処するために，使用停止状態の CPU をすぐに使用開始することができます。パーセント CPU でもアクティブ CPU でも，使用状況のデータが個別の Utility Meter に送信され，さらにセキュリティ保護された HP PPU Web ポータルに送信され，その結果， 48 時間以内にさまざまな使用レポートが参照可能になります。 (いずれのセル・ベースの Integrity サーバでも， Instant Capacity と Pay Per Use は同時に使用できません。)

PPU についての詳細は，次の Web サイトにある『HP Pay Per Use (PPU) **ユーザーガイド**』を参照してください。

`http://docs.hp.com/ja/hpuxos11iv2`

## 2.9 HP Superdome **ハイブリッド・サーバのサポート** (I64 **のみ**)

HP sx1000 チップセットを搭載した HP Superdome サーバでは，同じサーバ内で PA-RISC nPartitions と Intel Itanium 2 nPartitions を使用する構成をサポートすることができます。

Superdome ハイブリッド・サーバで PA-RISC nPartitions と Intel Itanium 2 nPartitions の両方を使用するためには，特定のハードウェア，ファームウェア，オペレーティング・システム，管理ツールが必要です。

HP OpenVMS for Integrity Servers Version 8.3 は，Superdome ハイブリッド・サーバ上で 9 MB のキャッシュを搭載した Intel Itanium 2 シングルコア・プロセッサを使用した nPartitions 内で動作します。

詳細と要件については，次の Web サイトにある『HP Superdome Hybrid Servers: Intel Itanium 2 and PA-RISC nPartition Mixing』を参照してください。

http://docs.hp.com

このマニュアルは，「Systems Hardware」セクションの，「HP Integrity Superdome Server」および「HP 9000 Superdome Server」という見出しの下にあります。

## 2.10 HP Web-Based Enterprise Management Services for OpenVMS (WBEM)

WBEM は，Integrity サーバ・システム向けの HP OpenVMS 上で使用可能なオプション製品で，業界標準のエンタープライズ管理フレームワークとリソース記述を提供します。WBEM 構造化フレームワークは拡張可能であり，各種インターネット標準を使用しています。管理アプリケーションの開発者は，リソース情報と動作状況の表示に関する開発済みのソフトウェアを利用することができます。たとえば，特定のプラットフォームまたはアプリケーション向けに作成されたコードを入手し，それを同じ目的で WBEM で使用することができます。

# 3
# システム管理機能

この章では，システム管理者向けの新機能，変更，機能拡張について説明します。

## 3.1 BACKUP ユーティリティの機能拡張

OpenVMS Version 8.3 では， Backup ユーティリティに対して以下の機能拡張が行われています。

- DVE (動的ボリューム拡張: Dynamic volume expansion)
- セーブ・セットの暗号化
- 会話型バックアップ操作の際の，より詳細な Ctrl/T メッセージの表示
- 拡張された BACKUP メッセージを現在の出力デバイスに送る新しい /PROGRESS_REPORT 修飾子
- システム上での同時 READ I/O の数を制御する新しい/IO_LOAD 修飾子

これらの機能拡張の詳細については，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』を参照してください。

### 3.1.1 OpenVMS Backup ユーティリティでの DVE (Dynamic Volume Expansion) のサポート

OpenVMS Version 8.3 から，イメージをセーブ・セットにバックアップするときに，ボリューム拡張サイズがセーブ・セット・ヘッダに記録されるようになりました。以前は， BACKUP が出力ボリュームを初期化する際に， DVE 情報を持っていませんでした。そのため，ディスク同士でのバックアップを行う場合やセーブ・セットを復元する場合に，ボリューム拡張サイズや論理ボリューム・サイズが保持されませんでした。

#### 3.1.1.1 ボリューム拡張サイズ

BACKUP/LIST を実行すると，セーブ・セット内にボリューム拡張サイズが格納されていればそれが表示されるようになりました。セーブ・セットを復元すると (またはディスク間のバックアップを行うと)，復元先のデバイスはボリューム拡張の上限をセーブ・セットから引き継ぎます。セーブ・セットに拡張サイズが含まれていない場合は， BITMAP.SYS のサイズは以前のバージョンの OpenVMS と同様に決定されます。

/IGNORE=LIMIT **オプション**
新しいオプション/IGNORE=LIMIT を指定すると，対象デバイスで拡張の上限が継承されなくなります。

/LIMIT **修飾子**
新しい修飾子/LIMIT では，セーブ・セット・ヘッダに格納されている値にかかわらず，復元操作や保存操作の際に，拡張サイズの上限を指定することができます。これは， INITIALIZE ユーティリティの/LIMIT 修飾子の機能と同じです。

#### 3.1.1.2 論理ボリューム・サイズ

特に指定しなければ，論理ボリューム・サイズは保持されません。その理由としては，たとえば 2GB のセーブ・セットを 4GB のディスクに復元する場合，使用可能なディスク領域が 2GB だけになってしまうためです。

/SIZE **修飾子**
論理ボリューム・サイズを保持するには，/SIZE 修飾子を使用します。/SIZE を指定すると，復元先デバイスのジオメトリは，$GETDVI を呼び出してデバイスの物理的な上限から決まるのではなく，論理サイズによって決まります。

/SIZE 修飾子では，ターゲット・デバイスの新しい論理サイズとして，オプションの値を指定できます。この新しい値はセーブ・セット内の既存の値より優先されます。これは， INITIALIZE ユーティリティの/SIZE 修飾子の動作と同じです。

/NOINITIALIZE **修飾子**
BACKUP/NOINITIALIZE を使用する場合の制限事項は，このコマンドが出力デバイスの DVE 特性を保持しないことです。その理由としては，対象デバイスがフォーリン・マウントされるため， OpenVMS は拡張サイズと論理サイズを取得することができないためです。この制限に対処するには，/LIMIT 修飾子と/SIZE 修飾子を使用します。

『OpenVMS システム管理者マニュアル』の「記憶媒体の管理」の章には， DVE について詳しく説明している項があります。

### 3.1.2 BACKUP セーブ・セットの暗号化

OpenVMS Backup ユーティリティでは，機能的に同等なバックアップ・コピーを作成することによって，ファイルやボリュームが破壊されないように保護することができます。 BACKUP は BACKUP 形式で書き込まれたセーブ・セットを作成するため，セーブ・セットのデータを解釈できるのは BACKUP だけです。セーブ・セットを作成する際には，暗号化することで保護を強化することができます。

OpenVMS Version 8.3 には，以下の暗号化機能が新たに追加されています。

- AES (Advanced Encryption Standard) 暗号化アルゴリズムのサポート
- ALGORITHM オプションに対する以下の AES キーワードのサポート

```
AESCBC          AESCFB
AESECB          AESOFB
```

各モードでは，3 種類の長さ (128，192，および 256 ビット) のユーザ定義の秘密鍵が使用でき，合計 12 通りの鍵の組み合わせがあります。ALGORITHM オプションの後に=AES を指定すると，省略時の指定として AESCBC128 になります。

- ユーザが指定した AES 暗号化アルゴリズムを使用した，セーブ・セット内のデータの暗号化

**注意**

**スタンドアロン BACKUP は，OpenVMS オペレーティング・システムのサポートなしで動作する Backup ユーティリティで，/ENCRYPT 修飾子をサポートしていません。**

### 3.1.3 追加の CTRL/T メッセージ

BACKUP を使用して会話型でデータをバックアップまたは復元する場合，Ctrl/T を押して操作の進行状況を表示することができます。OpenVMS Version 8.3 では，以下の状況のいずれかでこの情報が増加します。

- ディスクからセーブ・セットを復元する場合
- セーブ・セットを使用してテープまたはディスクにイメージをバックアップする場合

表示されるようになった追加の情報は以下のとおりです。

- 完了した操作の割合
- 操作を完了するために必要な時間の予測

新しい/PROGRESS_REPORT 修飾子を使用すると，拡張されたメッセージを現在の出力デバイスに送ることができます。

### 3.1.4 新しい/PROGRESS_REPORT 修飾子

第 3.1.3 項で説明している BACKUP 操作を実行する際に新しい/PROGRESS_REPORT 修飾子を指定すると，BACKUP 操作の進行状況を示すメッセージが出力デバイスに送られます。

### 3.1.5 新しい/IO_LOAD 修飾子

OpenVMS Version 8.3 では，新しいストレージ・コントローラでより効率的に動作するように BACKUP が最適化されています。/IO_LOAD 修飾子を使用すると，システム上で同時に実行する読み込み動作の数を増加または減少させることができます。

コマンドと修飾子の形式は次のとおりです。

```
BACKUP /IO_LOAD=n
```

ここで，nは， 1 からプロセスの AST リミットの間の整数です。省略時の値は 8 で，コマンド行で/IO_LOAD 修飾子を省略した場合に使用されます。

## 3.2 光媒体 CD および DVD の記録ツール

OpenVMS Version 8.3 では，光媒体 CD と DVD に記録するための新しいツールがサポートされています。これらのツールを利用することで，ローカルにマスタリングされたディスク・ボリューム・ファイルまたはディスク・イメージ・ファイルを，簡単に直接 CD-R， CD-RW， DVD+R， DVD+RW の光媒体記録デバイスに記録することができます。記録ツールで作成した光媒体データ・ディスクは，データ・アーカイブ操作，ソフトウェア配布のマスタ・ディスクの作成などの作業の一部として使用することができます。

COPY/RECORDABLE_MEDIA コマンドおよび関連するツールと診断機能は， CD 光媒体記録ツール SYS$MANAGER:CDRECORD.COM を補完し，最終的にはその代わりとして使用することができます。

COPY/RECORDABLE_MEDIA コマンドについては，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』を参照してください。サポート要件と機能については，本書の第 4 章を参照してください。

## 3.3 Integrity サーバ向け OpenVMS でのクラスタ・サテライトのサポート

OpenVMS Version 8.3 では， OpenVMS for Integrity Servers (I64) システムのクラスタ・サテライト・ブートがサポートされています。 I64 サテライト・システムでの動作や要件は， Alpha システムのものとは大きく異なります。クラスタに I64 システムを追加する前に，この項全体をお読みください。

### 3.3.1 Alpha のサテライトと I64 のサテライトの違い

表 3–1 に， Alpha のサテライトと Integrity サーバのサテライトの違いを示します。

表 3–1 Alpha のサテライトと Integrity サーバのサテライトの違い

| | Alpha | Integrity サーバ |
|---|---|---|
| ブート・プロトコル | MOP | PXE (BOOTP/DHCP /TFTP) |
| クラッシュ・ダンプ | DOSD (Dump Off the System Disk) によりリモート・システムのディスクまたはローカルのディスクにクラッシュ・ダンプを格納可能。 | DOSD が必要。リモート・ディスクへのクラッシュ・ダンプの格納は不可。 |
| エラー・ログ・バッファ | 常にリモート・システムのディスクに書き込まれる。 | エラー・ログ・バッファは，DOSD と同じディスクに書き込まれる。 |
| ファイル保護 | 標準のシステム・ディスクと全く同じ。 | すべてのロード可能 execlet が W:RE (省略時の設定) であり，特定のファイルで VMS$SATELLITE_ACCESS 識別子による ACL アクセスが可能であることが必要。 |

### サテライト

OpenVMS Version 8.3 システムまたはセル・ベース・システムの nPartitions は，サテライトとして使用できます。 nPartitions をサポートするためには，ファームウェアのアップグレードが必要です。

サテライト・ブートは，コア I/O LAN アダプタでのみサポートされます。すべてのサテライト・システムでは，クラッシュ・ダンプをサポートし，リブート後もエラー・ログ・バッファを保持するために，1 つ以上のローカル・ディスクが必要です。ディスクレス・システムでは，ソフトウェアが異常終了した場合にクラッシュ・ダンプを取得することができません。

### ブート・サーバ

OpenVMS Version 8.3 がサポートしているすべての Integrity サーバ・システムは，ブート・サーバとして使用することができます。現時点では， Integrity サーバ・サテライト・システムのクロス・アーキテクチャ・ブートはサポートされていないため， Integrity サーバ・サテライト・システムが含まれているクラスタでは， 1 台以上の Integrity サーバ・システムがブート・ノードの役目も果たすようにする必要があります。

### 必要なソフトウェア

- OpenVMS Version 8.3
- HP TCP/IP Services for OpenVMS Version 5.6 以降

他のサテライト・システムと同様に，システム・ソフトウェアは 1 台以上のノードによってクラスタにサービス提供されるディスクから読み込まれます。サテライト・システム・ディスクは，ブート・サーバのシステム・ディスクと同じでかまいませんが，必ずしも同じである必要はありません。 Alpha のサテライトでは，システム・ディスクをブート・サーバにマウントすることが推奨されているものの必須ではありま

せんが，I64 のサテライト・システムでは，システム・ディスクがブート・サーバにマウントされていることが必須です。

TCP/IP はブート・サーバのシステム・ディスクにインストールする必要があります。ブート・サーバのシステム・ディスクとサテライト・システム・ディスクが違う場合は，OpenVMS Version 8.3 はその両方にインストールする必要があります。

TCP/IP では，BOOTP と TFTP を設定し，1 つ以上のインタフェースを有効にしておく必要があります。少くとも 1 つの設定済みのインタフェースを，サテライト・システムからアクセス可能なセグメントに接続する必要があります。ブート・サーバとすべてのサテライト・システムには IP アドレスが必要です。TCP/IP Services for OpenVMS の構成についての詳細は，『TCP/IP Services for OpenVMS V5.6 インストレーション/コンフィギュレーション・ガイド』を参照してください。

### 3.3.2 サテライト・システムからの情報収集

サテライトに OpenVMS がインストールされているローカル・ディスクがある場合は，ログインします。そのようなディスクがない場合は，インストール DVD でブートして，オプション 8 (Execute DCL commands and procedures) を選択し，次のコマンドを実行します。

```
$ LANCP :== $LANCP
$ LANCP SHOW CONFIG

LAN Configuration:
Device Parent Medium/User Version Link Speed Duplex Size  MAC Address       Current Address   Type
------ ------ ----------- ------- ---- ----- ------ ----  ---------------   ---------------   ----
EIB0          Ethernet    X-16    Up   1000  Full   1500  00-13-21-5B-86-49 00-13-21-5B-86-49 UTP i82546
EIA0          Ethernet    X-16    Up   1000  Full   1500  00-13-21-5B-86-48 00-13-21-5B-86-48 UTP i82546
```

ブートに使用するアダプタの MAC アドレスを書き留めます。ブート・サーバに対してサテライト・システムを定義する際に必要になります。Current Address が MAC Address と違う場合は，MAC Address を使用します。

### 3.3.3 ブートとクラッシュ用にサテライト・システムを設定する

サテライトに，OpenVMS がインストールされたローカル・ディスクがある場合は，ログインします。そのようなディスクがない場合は，インストール DVD でブートし，オプション 8 (Execute DCL commands and procedures) を選択します。SYS$MANAGER:BOOT_OPTIONS.COM を使用して，ブートに使用するネットワーク・アダプタのブート・メニュー・オプションを追加します。手順の中で，このネットワーク・エントリがサテライト・ブート用かどうかが質問されます。その旨を指定すると，そのブート・メニュー・エントリに対して，メモリ・ディスク・ブート・オプション・フラグ (0x200000) が設定されます。メモリ・ディスク・フラグは，サテライト・ブートに必要です。

システムを主にサテライト・ブートで使用する場合は，位置 1 にネットワーク・ブート・オプションを配置します。また，サテライト・システムでは，クラッシュ・ダンプ用と，書き込まれていないエラー・ログ・バッファをリブートやクラッシュ後も保持するために， DOSD (Dump Off the System Disk) が必要です。 BOOT_OPTIONS.COM は， DOSD デバイス・リストを管理するために使用することもできます。 DOSD デバイス・リストはこの時点で作成することをお勧めします。 DOSD デバイス・リストの設定方法については，『OpenVMS システム管理者マニュアル (下巻)』を参照してください。

### 3.3.4 ブート・サーバに対するサテライト・システムの定義

I64 サテライト・システムは， PXE プロトコルを通じてブートします。 OpenVMS では， PXE は TCP/IP 製品の BOOTP によって処理されます。クラスタで複数の I64 ブート・サーバを使用している場合は， BOOTP データベースを共通のディスクに格納してください。 TCP/IP コンポーネントの設定については， TCP/IP のドキュメントを参照してください。サテライト・システムを定義する前に， TCP/IP をインストール，設定し，動作させておく必要があります。

I64 ブート・サーバ上で，システム管理者または適切な特権を持つ他のアカウントでログインします。コマンド・プロシージャ SYS$MANAGER:CLUSTER_CONFIG_LAN.COM を実行します。 (DECnet を使用してサテライト・ノードを設定する CLUSTER_CONFIG.COM は， I64 システムをサポートしていません。ただし， I64 システムでは自動的に CLUSTER_CONFIG_LAN を起動します。) CLUSTER_CONFIG_LAN はメニュー方式のコマンド・プロシージャで，サテライト・システムの設定に役立つように設計されています。メニューはコンテキスト依存であり，アーキテクチャとインストールされている製品によって変わります。手順に慣れていない場合は，システム管理ドキュメントで CLUSTER_CONFIG_LAN のより幅広い概要を参照してください。

I64 サテライトを追加するために不可欠な情報としては，ノードの SCS ノード名， SCS システム ID，ハードウェア・アドレスがあります。また，サテライトの IP アドレス，ネットワーク・マスク，場合によってはゲートウェイ・アドレスも必要になります。これらの概念について詳しくない場合は， TCP/IP のドキュメントを参照してください。プロシージャは，サテライト用のシステム・ルートを作成します。

CLUSTER_CONFIG_LAN では，サテライト・システムをブート可能にするためのすべての手順が実行されます。ローカルなページング・ファイルとスワップ・ファイルを選択すると，ファイルを作成するために，サテライト・システムをクラスタ内でブートするように促されます。ローカルに作成しない場合は，ページング・ファイルとスワップ・ファイルはサービスされるシステム・ディスク上に作成され，都合の良いときにサテライトをブートすることができます。

### 3.3.5 サテライトのブート

ブート・メニューにオプションを追加した場合は，そのオプションを選択します。オプションを追加しなかった場合は，ハードウェアのドキュメントでネットワーク・アダプタからのブートに必要な手順を参照してください。環境変数 VMS_FLAGS は，必ずメモリ・ディスク・ブート・フラグ (0x200000) が含まれるように設定してください。システムは，ネットワークから VMS_LOADER を取得すると，詳細なブートの進行状況をシステム・メッセージの形で表示します。ブート・シーケンスを開始するためにダウンロードした各ファイルに対して，コンソール・デバイスにピリオドが 1 つ表示され，最後に IPB (一次ブートストラップ・イメージ) がロードされたことを示すメッセージが表示されます。

以下に例を示します。

```
Loading.: Satellite Boot EIA0 Mac(00-13-21-5b-86-48)
Running LoadFile()

CLIENT MAC ADDR: 00 13 21 5B 86 48
CLIENT IP: 16.116.43.79  MASK: 255.255.248.0  DHCP IP: 0.240.0.0

TSize.Running LoadFile()

Starting: Satellite Boot EIA0 Mac(00-13-21-5b-86-48)
Loading memory disk from IP 16.116.43.78
.....................................................................
Loading file: $13$DKA0:[SYS10.SYSCOMMON.SYSEXE]IPB.EXE from IP 16.116.43.78
%IPB-I-SATSYSDIS, Satellite boot from system device $13$DKA0:

    HP OpenVMS Industry Standard 64 Operating System, Version V8.3
    (c) Copyright 1976-2006 Hewlett-Packard Development Company, L.P.
```

最初のフル・ブート時には，サテライト・システムは AUTOGEN を実行してリブートします。

### 3.3.6 サテライト・システム上での追加作業

まだ DOSD 用のダンプ・ファイルを作成していない場合は，ここで作成します。SYS$STARTUP:SYCONFIG.COM ファイルを編集し， DOSD デバイスをマウントするコマンドを追加します。エラー・ログ・バッファが復旧されるようにするには，DOSD デバイスを SYCONFIG 内でマウントする必要があります。

## 3.4 ロックの動的な再マスタリング —LOCKRMWT

OpenVMS がロック・ツリーを再マスタリングするかどうかを判断する方法が，OpenVMS Version 8.3 で更新されました。 Version 8.3 よりも前のバージョンでは，システム・パラメータ LOCKDIRWT に基づいて決定されていました。ロック・ツリーは， LOCKDIRWT 値が大きなノードに移動されていました。複数のノードの LOCKDIRWT の値が同じ場合は，ロック・ツリーはアクティビティの高いノードに

移動されていました。アクティビティの量の判断では，ハードコードされた非常に小さなしきい値が使用されていたため，ノード間でのロック・ツリーの移動でスラッシングが発生することがよくありました。

Version 8.3 では，新しいシステム・パラメータであるロック・マスタの重み (LOCKRMWT) が実装されています。このパラメータは，ゼロ (0) から 10 の範囲の値を取り，デフォルト値は 5 です。このパラメータの値は，ノードがマスタ・ロック・ツリーを必要としている度合いを表します。値が大きいほどツリーがそのノードに移動する可能性が高まります。 0 と 10 は特別な値です。ゼロは，そのノードが，リソース・ツリー上のロックを持つ唯一のノードにならないかぎり，そのノードにマスタ・ツリーを移動させないことを示します。値がゼロのノード上のマスタ・ツリーは， LOCKRMWT の値が 0 より大きなノードがあれば，そのノードに移動されます。値 10 は，そのノードが常にマスタ・ロック・ツリーを必要とすることを示します。 LOCKRMWT の値が 10 より小さいノードにマスタ・ロック・ツリーがあり，値が 10 のノードが必要としている場合は，値が 10 のノードにロック・ツリーが再マスタリングされます。

その他の場合は，現在のマスタ・ノードとリモート・ノードの LOCKRMWT の差が計算されます。差が大きいほど，ツリーが再マスタリングされる可能性が高まります。ノードの LOCKRMWT の値が同じ場合は，リモート・ノード上の動作が約 13%以上多い場合にツリーが再マスタリングされます。リモート・ノードが 8 で現在のマスタが 5 の場合は，差は 3 となり，あと約 2%動作が多いとロック・ツリーはリモート・ノードに移動されます。リモート・ノードが 1 で現在のマスタが 9 の場合は，差は -8 となります。この場合，仮に現在のマスタよりも約 200%動作が多くても，ロック・ツリーはリモート・ノードに再マスタリングされます。

新しい LOCKRMWT パラメータは動的であるため，動作中のシステム上で SYSGEN を使用して変更することができます。また，ロックの再マスタリングでは，引き続き PE1 システム・パラメータが考慮されます。 PE1 の値よりも多くのロックを持つロック・ツリーは再マスタリングされません。 LOCKDIRWT パラメータは，ロックの動的な再マスタリングとは全く関係がなくなりました。このパラメータは，リソース・ディレクトリ・エントリを管理するノードの可能性だけを決定します。

複合バージョン・クラスタでは， Version 8.3 のノードと (LOCKRMWT システムパラメータを持たない) Version 8.3 よりも前のノードの間のやり取りには，LOCKDIRWT を使用した古い規則が使用されます。ロック・ツリーがクラスタ内で移動し続けるのを避けるために，上の規則には 1 つの例外があります。 Version 8.3 のマスタ・ノードは， LOCKDIRWT の値が大きなノードがあると， Version 8.3 よりも前のノードに，ロック・ツリーを再マスタリングしません。この例外が必要な理由としては， Version 8.3 よりも前のノードが，値が大きな LOCKDIRWT を持つノードにすぐに再マスタリングするためです。

SHOW CLUSTER ユーティリティは，クラスタ内のノードの LOCKRMWT システム・パラメータを表示するように拡張されています。 LOCKRMWT を表示するには，コマンド ADD RM_WT または ADD MEMBERS/ALL を使用します。 Version 8.3 よりも前のノードでは，このフィールドは****で表示され，これらのノードに LOCKRMWT システム・パラメータがないことを示します。

## 3.5 OpenVMS での暗号化

OpenVMS Version 8.3 では，以前の Encryption for OpenVMS ソフトウェア製品がオペレーティング・システムに統合されています。これにより，別途製品をインストールする必要がなく，製品ライセンスも不要になります。また， OpenVMS Version 8.3 には， AES (Advanced Encryption Standard) アルゴリズムのサポートが追加されており， OpenVMS ユーザ，システム管理者，セキュリティ管理者，プログラマが，ファイル，セーブ・セット，アプリケーション・データを AES 暗号化で保護することができます。

暗号化は，機密データや個人的なデータを，暗号文と呼ばれる理解不能な形式に変換することを指します。これは，データの機密を保護するために行います。復号化ではこの処理を逆に実行し，理解不能な暗号文を，平文と呼ばれる元の形式のデータに戻します。復号化は解読とも呼ばれます。

### 3.5.1 AES の機能

AES 暗号化は，以下の機能と互換性を備えています。

- 以前のデータ暗号化標準 (DES) アルゴリズムは，既存の DES データとそのアプリケーションで使用するために，引き続き提供されています。 DES で提供されていた機能については，同じレベルの DES のサポートが引き続き提供されます。
- AES 暗号化は BACKUP に統合されており， AES または DES を使用してセーブ・セットを暗号化および復号化することができます。
- コマンド行での AES 暗号化の使用方法は，修飾子が少し変更されていること以外は同じです。
- ENCRYPT$アプリケーション・プログラミング・インタフェース (API) の変更は最小限で， AES アルゴリズムを使用するに当たっては，テキスト・パラメータやフラグを変更するだけで済みます。
- AES 暗号化では， AES アルゴリズムで 4 つの暗号化モードがサポートされています。それぞれのモードで， 3 種類の長さ (128， 192， 256 ビット) の秘密鍵を指定することができ，合計 12 種類の暗号化と復号化操作が可能です。
  - AESCBC128！Cipher Block Chaining
  - AESCBC192！Cipher Block Chaining
  - AESCBC256！Cipher Block Chaining

- AESECB128 ! Electronic Code Book
- AESECB192 ! Electronic Code Book
- AESECB256 ! Electronic Code Book
- AESCFB128 ! Cipher Feedback
- AESCFB192 ! Cipher Feedback
- AESCFB256 ! Cipher Feedback
- AESOFB128 ! Output Feedback
- AESOFB192 ! Output Feedback
- AESOFB256 ! Output Feedback

• これらの追加の AES アルゴリズム，モード，鍵サイズは， API ENCRYPT$ENCRYPT_FILE( ) および ENCRYPT$INIT( ) のalgorithmパラメータで指定するか， ENCRYPT$GENERATE_KEY( ) API のalgorithm-nameパラメータで指定します。

### 3.5.2 /CREATE_KEY /AES コマンド修飾子

AES の鍵 (および DES の鍵) は， ENCRYPT のコマンド行修飾子/CREATE_KEY で作成します。ただし， AES の鍵の場合は，/AES 修飾子を追加する必要があります。

```
$ ENCRYPT /CREATE_KEY keyname "This is my secret key" /AES
```

これにより， 21 文字の AES 鍵が生成されます。最小の鍵の長さの要件を満たし，Encrypt の最大文字数 (約 240 文字) を超えない，任意の長さの鍵を指定することができます。

### 3.5.3 AES 鍵の長さに関する要件

AES 鍵の要件は，それぞれの AES モードで使用される実際のビット数です。これは，実際には，暗号化操作または復号化操作で必要な最小バイト数です。必要な最小の鍵サイズは以下のとおりです。

- 128 ビット・モード= 16 バイトの鍵
- 192 ビット・モード= 24 バイトの鍵
- 256 ビット・モード= 32 バイトの鍵

### 3.5.4 リテラル鍵の値と ASCII 圧縮

リテラル鍵の値は，通常は圧縮されずにそのまま暗号化アルゴリズムに渡されます。リテラル鍵の値は， ASCII， HEX，バイナリのいずれかです。ただし， ASCII DES 鍵の値は，記述子のデータ型が DSC$K_TYPE_T， DSC$K_TYPE_VT， DSC$K_TYPE_VT のいずれかの場合は圧縮されます。 AES 鍵は圧縮されません。その他の記述子データ型では， DES リテラル鍵の値を圧縮せずにそのまま渡すことが可能です。

リテラル鍵の値はコマンド行の/HEX 修飾子とともに指定するか，ENCRYPT$DEFINE _KEY()ルーチンに対する鍵フラグliteralとともに指定します。リテラル鍵の値は，key-type引数を使用し，鍵名の記述子で値を渡すことで，ENCRYPT$INIT()ルーチンに直接渡すことも可能です。コマンド行で指定した DES ASCII 鍵の値は，鍵が引用符で囲まれているかどうかにかかわらず圧縮されます。

### 3.5.5 XOR 鍵フラグ，または鍵の折りたたみ

Encrypt は，鍵の中の残りの文字に対して XOR 操作を行って，鍵自体に対して折りたたみを行い，アルゴリズムで使用する数のバイトで， AES (または DES) 用のサイズの鍵を作成します。そのため，鍵の値に対して最大バイト数 (240) のバイト・データを指定できますが，鍵の作成時には， AES 用に保存されるのは 32 バイト， DES 用に保存されるのは 8 バイトです。

この鍵を使用するときには，元の鍵が復元され，ストレージから復号化されます。しかし，暗号化操作に対して選択した AES 鍵のサイズに応じて， DES では 8 バイト (折りたたみ済み) の鍵だけが使用され， AES では 16 バイト， 24 バイト， 32 バイト (折りたたみ済み) の鍵だけが使用されます。

鍵サイズは，algorithm-nameパラメータの一部として指定します。たとえば，以下の API で，または ENCRYPT コマンドおよび DECRYPT コマンドのファイルに対する修飾子/DATA_ALGORITHM または/KEY_ALGORITHM でAESCBC256を指定します。

- ENCRYPT$ENCRYPT_FILE( )
- ENCRYPT$INIT( )
- ENCRYPT$GENERATE_KEY( )

**AES 鍵を作成する例**

次の例では， 32 バイトの AES 鍵を作成します。この鍵は AES (現在は AESCBC128) で暗号化され，特に指定しなければプロセスの論理名テーブルに ENCRYPT$KEY$MY_KEY という名前で格納されます。この鍵には， DES 鍵と区別するために AES 鍵のフラグが設定されます。

```
$  encrypt/create MY_KEY "This is a sample ASCII key value" /aes/log
%ENCRYPT-S-KEYDEF, key defined for key name = MY_KEY
```

### 3.5.6 ENCRYPT$DEFINE_KEY( ) API

AES 鍵と DES 鍵は， Encrypt のアプリケーション・プログラム・インタフェース (API) ENCRYPT$DEFINE_KEY( ) でも作成できます。鍵フラグは，鍵の種類 (名前またはリテラル鍵の値) と，鍵を格納する論理名テーブルを区別するために使用します。

AES 鍵フラグ・マスク ENCRYPT$M_KEY_AES と，値 ENCRYPT$V_KEY_AES も， AES 鍵を作成するために使用されます。

```
ENCRYPT$DEFINE_KEY ( key-name , key-value , key-flags )
```

ENCRYPT$GENERATE_KEY( ) API では，ランダムな鍵の値を生成することができます。

```
ENCRYPT$GENERATE_KEY (algorithm-name , key-length
            [,factor-a] [,factor-b] [,factor-c]
            [,key buffer])
```

**AES 鍵フラグ**

以下の AES マスクを他のフラグと組み合わせて， key-flags パラメータで (参照渡しのロングワードとして) 使用することができます。対応する AES 鍵の値を，プログラム中のビットのテストで使用することができます。 API ENCRYPT$DEFINE_KEY( ), ENCRYPT$DELETE_KEY( ), および ENCRYPT$GENERATE_KEY( ) で，KEY_AES 鍵フラグを使用して AES 鍵を指定します。

- ENCRYPT$M_KEY_AES
- ENCRYPT$V_KEY_AES

### 3.5.7 鍵に関する注意事項

以下に鍵に関する情報を示します。

- 作成された AES 鍵は，暗号化されて (常に AESCBC128 とマスタ鍵を使用) 論理名テーブルに格納されます。暗号化操作の際，鍵が取り出され，復号化されて，暗号化または復号化操作で選択されたアルゴリズムと鍵のサイズに応じて， 16 バイト， 24 バイト， 32 バイトの鍵として使用されます。
- リテラルでない DES 鍵は圧縮 (大文字に変換) されます。使用できる文字は A–Z, 0–9，ドル記号 ($)，ピリオド (.)，アンダスコア (_) であり，その他の文字はスペースに変換され，複数のスペースは削除されます。 AES ASCII キーの値は圧縮されません。
- 鍵を作成する場合は，後で選択されたアルゴリズム/鍵サイズで使用する際の最小鍵長を満たすように注意してください。 8 バイトの DES 鍵では問題は発生しません。必要以上に長い鍵 (リテラルまたは非リテラル鍵) は，適切な 16 バイト， 24 バイト， 32 バイトの鍵サイズに折りたたまれます。

- 鍵の名前は，論理名テーブル (SYSTEM， JOB， GROUP，または PROCESS (デフォルト)) に格納されている鍵の論理名です。値は ASCII (通常のテキスト鍵)， 16 進/バイナリのいずれかとなります。リテラル鍵 (key-flags = ENCRYPT$M_LITERAL_KEY) を作成すると，値はリテラル値として格納され，圧縮されません。
- また，後からENCRYPT$INIT( ) API に鍵を渡す際には，論理名テーブルに格納されている鍵と一致させるように注意してください。つまり，記述子の型によって DES 鍵の扱い方が決まります。テキストは圧縮され，バイナリ値は圧縮されません。 AES 鍵の値は圧縮されません。鍵フラグ (1=リテラル， 0=名前) は，key-nameパラメータの解釈方法 (リテラル値を直接 INIT に渡すか，論理名の検索，変換，復号化用に鍵の名前を渡すか) を決定します。鍵の種類が正しくない場合，たとえば鍵フラグが 0 (名前) で，鍵の名前ではなくリテラルの鍵値を指定した場合に出力されるエラーに注意してください。リテラル値として鍵の名前を指定した場合にもエラーが発生します。 ENCRYPT$INIT( ) API では，型が DSC$K_DTYPE_T， DSC$K_DTYPE_VT，および DSC$K_DTYPE_Z の鍵名記述子は， DES 鍵の値を圧縮することを指定します。 AES 鍵の値は圧縮されません。
- ENCRYPT$GENERATE_KEY( ) を使用して AES 鍵を生成するときに，鍵の長さとして 16 の倍数でない値を指定するとエラーになります。

### 3.5.8 AES 鍵の削除

AES (および DES) 鍵は， Encrypt のコマンド行修飾子/REMOVE_KEY，または API ENCRYPT$DELETE_KEY( ) で削除します。

```
$ ENCRYPT/REMOVE_KEY KEYNAME /AES
```

ユーザの秘密鍵は，マスタ鍵で暗号化され，論理名テーブル (PROCESS， JOB， GROUP，または SYSTEM—ENCRYP$SYSTEM テーブル) に格納されます。省略時の論理名テーブルは PROCESS です。 PROCESS 論理名テーブル以外のテーブルから鍵を削除するには， ENCRYPT /REMOVE_KEY コマンドで適切な修飾子 (/JOB，/GROUP，または/SYSTEM) を指定する必要があります。

ユーザの秘密鍵名は一意であるため， DES 鍵と AES 鍵のどちらであっても，同じ論理名テーブルには同じ名前の鍵が 1 つしか存在できません。そのため，/AES 修飾子が実装されていますが，この修飾子は不要です。

### 3.5.9 ENCRYPT$DELETE_KEY( ) API

Encrypt API を使用して論理名テーブルから鍵を削除するには，削除対象の鍵の名前を指定します。どの論理名テーブルから削除するかをフラグで指定します。

```
ENCRYPT$DELETE_KEY (key-name , key-flags)
```

**AES 鍵フラグ**

key-flagsパラメータでは，他のフラグとともに (論理和をとって)，以下の AES マスクを (参照渡しのロングワードとして) 使用することができます。対応する AES 鍵の値は，プログラム内でビットのテストを行うために使用できます。 ENCRYPT$DEFINE_KEY( )， ENCRYPT$DELETE_KEY( )，および ENCRYPT$GENERATE_KEY( ) の各 API で AES 鍵を指定するには， KEY_AES 鍵フラグを使用します。

- ENCRYPT$M_KEY_AES
- ENCRYPT$V_KEY_AES

### 3.5.10 ファイルの暗号化と復号化

鍵を作成したら，ファイルの暗号化や復号化を行うことができます。それには，コマンド行で ENCRYPT コマンドおよび DECRYPT コマンドを使用するか，ENCRYPT$ENCRYPT_FILE() API を使用します。

ファイルの暗号化では， RMS ファイルを，固定長の 512 バイト・レコードで暗号化します。ファイル作成日付と変更日付，ファイルが順編成ファイルなのか索引編成ファイルなのか，そのレコード形式 (STREAM_LF， VAR など) などのファイルの特性と属性は保持されます。ユーザはファイルの暗号化に使用する鍵とデータアルゴリズムを指定しますが，ユーザの鍵は，ランダムな鍵，初期化ベクタ (IV)，ランダム鍵レコード中のデータ・アルゴリズムを暗号化するために使用されます。ユーザが指定したデータ・アルゴリズムを使用してファイルの属性，機能レコード，そのデータ・レコードを暗号化するのは，ランダムな鍵です。

ファイルを復号化する際には，ユーザが指定した鍵はランダム鍵レコードを復号化するために使用され，ランダムな鍵 (データ鍵)， IV，データ・アルゴリズムなどが取得されます。次に，ランダムな鍵， IV，データ・アルゴリズムを使用して，固定長の 512 バイト・レコードからファイル属性，機能レコード，データ・レコードが復号化され，元の形式に復元されます。

#### 3.5.10.1 ファイルの暗号化と復号化の省略時のモード —DESCBC

特に指定しなければ，コマンド行からファイルを暗号化するときには， Encrypt は DESCBC アルゴリズムを使用してファイルを暗号化します。すなわち，鍵やデータ・アルゴリズムがコマンド行で指定されていない場合， DESCBC アルゴリズムとモードが使用されます。

DESCBC を使用して，ファイルfile-nameを鍵key-nameで暗号化し，ファイルfile-nameに出力する例は次のようになります。

```
$ ENCRYPT file-name key-name
```

DESCBC を使用してファイルを復号化するには，次のコマンドを使用します。

```
$ DECRYPT file-name key-name
```

### 3.5.10.2 AES データ・アルゴリズムと AES 鍵アルゴリズムの指定

ファイルを暗号化する際に省略時の DESCBC 以外のアルゴリズムを選択するには，DCL の ENCRYPT コマンドでデータと鍵のアルゴリズム修飾子が指定可能で，DECRYPT コマンドでは鍵アルゴリズム修飾子が指定可能です。

AES を使用してファイルを暗号化する場合は，/DATA_ALGORITHM=AESmmmkkkと/KEY_ALGORITHM=AESmmmkkkの両方を指定します。

- mmmは，AES モード，ECB，CBC，CFB，OFB のいずれか
- kkkは，鍵のサイズ，128 ビット，192 ビット，256 ビットのいずれか (16 バイト，24 バイト，32 バイトの鍵)

Encrypt は，鍵と鍵アルゴリズムが一致することを期待します。AES 鍵は AES 鍵アルゴリズムとともに使用し，DES 鍵は DES 鍵アルゴリズムとともに使用する必要があります。ENCRYPT コマンドで/DATA_ALGORITHM=AESmmmkkkを指定しなかった場合の省略時のデータ・アルゴリズムは DES となります。DES 鍵と KEY_ALGORITHM=DES を使用する場合も同じことがいえます。データは強力なアルゴリズムで保護されますが，鍵はそうではありません。

**注意**

**AES 鍵と DES 鍵，および各データ・アルゴリズムを混在させる機能は，OpenVMS Version 8.3 で使用できなくなりました。混在させた場合は，ENCRYPT$_AESMIXDES エラー状態となります。**

AES を使用してファイルを復号化する場合は，/KEY_ALGORITHM=AESmmmkkk修飾子だけを指定します。これは，ランダムな鍵が格納されているランダム鍵レコードを復号化するためにこの鍵アルゴリズムが使用され，ランダムな鍵を使用してファイルのデータ・レコードが復号化されるためです。データ・アルゴリズムの指定は不要であり，実際，修飾子が認識できないというエラー・メッセージが表示されます。

**注意**

**暗号化操作では，/KEY_ALGORITHM を指定せずに/DATA_ALGORITHM=AES を指定すると，エラーが発生します。省略時のアルゴリズム DESCBC を使用して，ランダムな鍵とファイル情報が格納されているランダム鍵レコードが暗号化されます。しかし，Encrypt は，ユーザの鍵が KEY アルゴリズムと一致することを期待します。一致しないと，エラーが発生します。つまり，key-nameが AES 鍵の名前と値の場合，鍵が論理名テーブルから取り出され，DES マスタ鍵で復号化されると，鍵は誤って復号化され，操作が失敗して次のメッセージが表示されます。**

```
%STR-F-FATINTERR, fatal internal error
```

ENCRYPT /DATA_ALGORITHM=AES /KEY_ALGORITHM=AES
AES には，省略時の暗号化および復号化ルーチン (AESCBC128) があり，モードと鍵のサイズが指定されずに AES が指定された場合 (すなわち/AES だけが指定された場合) に使用されます。これは，次の例のように， AES ファイル暗号化のショートカットとして使用できます。

```
$ ENCRYPT file-name key-name /KEY=AES /DATA=AES
```

#### 3.5.10.3 鍵アルゴリズムのみの指定

ファイルを復号化する際に省略時の DESCBC 以外のアルゴリズムを選択するには，DCL DECRYPT コマンドで鍵アルゴリズム修飾子だけを指定します。 AES を使用して復号化するときには，/KEY_ALGORITHM=AESmmmkkk 修飾子だけを指定します。ここで mmm は AES モードです。

指定が必要なのは鍵アルゴリズムだけです。データ・アルゴリズムは，暗号化されたファイルの鍵レコード中に，他のファイル情報とともに格納されています。鍵レコードは，ファイルを暗号化する際にユーザが指定した暗号鍵で暗号化されています。復号化操作の際，ユーザの鍵を使用してデータ鍵 (暗号化の際に生成されたランダムな鍵) が格納された鍵レコードを復号化し，そのアルゴリズムとともに使用して，ファイル中の残りのデータ・レコードが復号化されます。

### 3.5.11 ENCRYPT$ENCRYPT_FILE() API

#### AES ファイル・フラグ

ENCRYPT$ENCRYPT_FILE() API の形式は次のとおりです。

```
ENCRYPT$ENCRYPT_FILE(input-file, output-file,
                     key-name, algorithm, file-flags,
                     item-list )
```

ENCRYPT$ENCRYPT_FILE( ) API では， AES アルゴリズムを使用してファイルを暗号化する場合に使用される，追加の FILE_AES フラグ・マスク (および値) があります。 ENCRYPT$ENCRYPT_FILE_FLAGS は，暗号化の方向，ファイルの圧縮などのファイル操作を制御するために使用されます。 FILE_AES フラグは，ファイルの AES の初期化や暗号化操作を制御し， AES 鍵にフラグを設定します。

- ENCRYPT$M_FILE_AES
- ENCRYPT$V_FILE_AES

オプションの項目リストは，データ・アルゴリズム・パラメータを指定変更するために使用します。その目的は，アルゴリズムを，機能的に似ていて名前が異なる別のアルゴリズムで置き換えることです。ランダム鍵レコードに格納されているアルゴリズム名を，オーバーライド記述子でユーザが指定したアルゴリズムの名前で指定変更します。これにより，復号化環境のアルゴリズム名と異なるアルゴリズム名を使用して暗号化されたファイルを開くための方法が提供されます。

### 3.5.12 レコードの暗号化と復号化

ファイル・レコードは，次の Encrypt API で暗号化および復号化を行うことができます。

```
ENCRYPT$ENCRYPT_ONE_RECORD (input, output, key-name, algorithm)
ENCRYPT$DECRYPT_ONE_RECORD (input, output, key-name, algorithm)
```

1 レコードの暗号化で AES を使用するには，最初に AES 鍵を作成し，論理名テーブルに格納します (暗号化される)。 AES 鍵の名前と，選択した AESmmmkkk (モードと鍵のサイズ) アルゴリズムに対する ASCII テキストが格納されている記述子のアドレスを指定します。入力バッファと出力バッファ (記述子のアドレス) も指定します。

これらの 1 レコードの API では，鍵が論理名テーブルにすでに存在していることが前提です。これらの API は，主に少量のデータまたは数レコードだけを暗号化および復号化するために使用します。これは， ENCRYPT$ENCRYPT_ONE_RECORD( ) API の呼び出し時にオーバヘッドとなる操作があるためです。この API を呼び出すと，関数 ENCRYPT$INIT(), ENCRYPT$ENCRYPT()および ENCRYPT$DECRYPT(), ENCRYPT$FINI()も呼び出されます。 INIT, DECRYPT，および FINI 関数は，ユーザがkey-nameパラメータで指定した鍵の名前を使用して論理名テーブル中の鍵を復号化するために，最初の INITIAL の中で再帰的に呼び出されます。

多数のレコードを暗号化または復号化する必要がある場合は，ENCRYPT$xxCRYPT_ONE_RECORD()呼び出しを使用**しない**ことをお勧めします。代わりに， API 関数 ENCRYPT$ENCRYPT()および ENCRYPT$DECRYPT()を使用して，通常の操作を行ってください。そのためには，暗号化や復号化を行う前に ENCRYPT$INIT()関数を使用してコンテキストを初期化し，アプリケーションが終了する前に ENCRYPT$FINI() API を呼び出してメモリ構造体を解放します。

### 3.5.13 データの暗号化と復号化

ENCRYPT$ENCRYPT()ルーチンと ENCRYPT$DECRYPT()ルーチンは，最大 64 K バイトのデータを暗号化処理するために，アプリケーションによって使用されます。

```
ENCRYPT$ENCRYPT (context, input, output [,output-length] [,initialization-vector]
ENCRYPT$DECRYPT (context, input, output [,output-length] [,initialization-vector]
```

これらのルーチンでは， ENCRYPT$ENCRYPT()や ENCRYPT$DECRYPT()を使用してデータ・ブロックの暗号化や復号化を行う前に， ENCRYPT$INIT()ルーチンで暗号化コンテキストを初期化する必要があります。コンテキスト・データ構造体を解放するために，最後に ENCRYPT$FINI()ルーチンを呼び出します。

出力バッファは，ブロック長の単位でパディングされたブロックを格納できる必要があります。ブロック長は， AES の場合は 16 バイトです (DES では 8 バイトです)。output-lengthの値とinitialization-vector (IV) パラメータは省略できます。output-lengthは，書き込まれる (暗号化または復号化される) バイト数です。

AES IV は， 16 バイトの値を参照するポインタです。内部構造体は， AES に対応できるように拡張されています。 DES IV は， 8 バイトの値を参照するクォドワードです。

### 3.5.14 長さとブロック・モード・パディング

AES ブロックモード・アルゴリズム (AESCBCxxxおよび AESECBxxx) は， 16 バイトのブロック境界までデータをパディングします。 AES の場合，暗号化と復号化を行うと， 1 バイトは 16 バイト， 72 バイトは 80 バイトなどとなります。 AES のパディング文字は，パディングされたバイト数を表す 16 進数です。たとえば， 1 バイトのデータを暗号化すると，暗号化された 1 バイトのデータの後ろに 15 文字の 0F がパディングされ， 72 バイトのデータを暗号化すると，暗号化された 72 バイトのデータの後ろに 08 が 8 バイト分パディングされます。 DESECB モードと DESCBC モードでは，常にゼロがパディングされます。文字ストリーム・モード (AESCFBxxx, AESOFBxxx, DESCFB) ではデータがパディングされず，output-lengthパラメータは実際のバイト数に一致します。

### 3.5.15 新しい AES 暗号化鍵，フラグ・マスク，値

新しい AES 暗号化 API ルーチンはありません。ただし， AES アルゴリズムとさまざまな鍵長の値に対応するために， OpenVMS Version 8.3 では AES 鍵と AES ファイル・フラグ・マスクおよび値が追加されていいます。

- AES 鍵フラグ

  ENCRYPT$DEFINE_KEY( ), ENCRYPT$DELETE_KEY( ), ENCRYPT$GENERATE_KEY( ) の各 API に対して， KEY_AES マスクの値を AES 鍵として指定します (参照渡しのロングワードとして指定)。

  — ENCRYPT$M_KEY_AES

  — ENCRYPT$V_KEY_AES

- AES ファイル・フラグ

  追加の FILE_AES フラグ・マスク (および値) は， AES アルゴリズムを使用してファイルを暗号化する際に ENCRYPT$ENCRYPT_FILE( ) API で指定します。

  ENCRYPT$ENCRYPT_FILE_FLAGS フラグは，暗号化の方向，ファイルの圧縮などのファイル操作を制御するために使用します。 FILE_AES フラグは，ファイルの AES 初期化と暗号化操作および以下の AES 鍵のフラグを制御します。

— ENCRYPT$M_FILE_AES

— ENCRYPT$V_FILE_AES

AES アルゴリズム，モード，および鍵長 (128 ビット， 192 ビット，または 256 ビット) は， API ENCRYPT$ENCRYPT_FILE( ) および ENCRYPT$INIT( ) のalgorithmパラメータで指定するか， ENCRYPT$GENERATE_KEY( ) API のalgorithm-nameパラメータで指定します。このパラメータは，以下のように文字列記述子の形式で，参照渡し (ポインタ) です。

- ブロック・モードの暗号化
  - — AESCBC128 ! Cipher Block Chaining
  - — AESCBC192 ! Cipher Block Chaining
  - — AESCBC256 ! Cipher Block Chaining
  - — AESECB128 ! Electronic Code Book
  - — AESECB192 ! Electronic Code Book
  - — AESECB256 ! Electronic Code Book
- ストリーム・モードの暗号化
  - — AESCFB128 ! Cipher Feedback
  - — AESCFB192 ! Cipher Feedback
  - — AESCFB256 ! Cipher Feedback
  - — AESOFB128 ! Output Feedback
  - — AESOFB192 ! Output Feedback
  - — AESOFB256 ! Output Feedback

**注意**

**AESCBC128 は省略時の暗号化であり，ユーザの鍵の暗号化および復号化を行って論理名に格納するためにも使用されます。これらの暗号化は，新しいイメージ・ファイル SYS$SHARE:ENCRYPT$ALG$AES.EXE 内のアルゴリズム・テーブルに格納された順序 (上記の順序) で検索されます。**

### 3.5.16 サポートされない AES 暗号化の操作

以下の AES 暗号化操作はサポートされないため，お勧めできません。

- MAC (Message Authentication Code)

  MAC (Message Authentication Code) は，ファイルのデータまたはそのセキュリティ設定に対する変更を検出します。現在， MAC 操作では DES だけがサポートされています。 AES はサポートされていません。

MAC は，コマンド修飾子/AUTHENTICATE で使用します。 MAC は，ファイルのデータ (およびセキュリティ属性) を暗号化し， 2 つの個別のデータベース (Db) に格納します。ファイルの変更を検出するために， MAC が再計算され， Db の MAC と比較されます。

認証コードは，/UPDATE 修飾子で生成され，/OUTPUT=file-name修飾子でログに記録または表示されます。次に例を示します。

```
$ encrypt /AUTHENTICATE /UPDATE *.exe KeyName /out=tt:
```

MAC は IV も使用しますが，基となるアルゴリズムおよびキーが設定されたファイル MAC のモードは DESCBC です。 MAC は，ファイルのデータとファイルのセキュリティ属性が，最終的に DESCBC で暗号化されたブロックです。

- ENCRYPT/COMPRESS

  ENCRYPT/COMPRESS を BACKUP ファイル・セーブ・セットで使用すると，復号化の際にエラーが発生するため，お勧めしません。これは，通常，ゼロでない/GROUP_SIZE を指定して作成した大規模なセーブ・セットで発生します。

  ENCRYPT/COMPRESS は正しく行われますが，復号化に失敗する可能性があります。暗号化操作の際に，/DELETE 修飾子を使用してオリジナルの BACKUP セーブ・セット・ファイルを削除する場合は，大きな問題になります。

- AES でのファイルの暗号化

  AES ファイルを暗号化する際には，/DATA=AESmmmkkkと /KEY=AESmmmkkkアルゴリズムの両方を指定します。ここで，mmmはモード (CBC， ECB， CFB， OFB のいずれか)，kkkは鍵サイズ (128 ビット， 192 ビット，または 256 ビット) です。

- AES および DES の鍵とアルゴリズムの混在

  Encrypt は，鍵と鍵アルゴリズムが一致することを期待します。 AES 鍵は AES 鍵アルゴリズムとともに使用し， DES 鍵は DES 鍵アルゴリズムとともに使用する必要があります。/DATA_ALGORITHM 修飾子に AES を指定せず，/KEY_ALGORITHM 修飾子に AES 鍵とともに AES を指定すると，省略時のデータ・アルゴリズムは DES となります。セキュリティ上の理由から，このような指定を行うと，コマンド行で ENCRYPT$_AESMIXDES エラーが発生します。 DES 鍵と KEY_ALGORITHM=DES を使用する場合も同じことがいえます。データは強力なアルゴリズムで保護されますが，鍵はそうではありません。鍵とデータ・アルゴリズムで DES と AES を混在させるコマンド行の機能は， OpenVMS 8.3 で使用できなくなりました。 AES と DES の鍵とアルゴリズムを混在させると，他のエラーが発生することもあります。

## 3.6 Monitor ユーティリティの機能拡張

ここでは， Monitor ユーティリティの機能拡張について説明します。

### 3.6.1 Align **コマンド** (I64 **のみ**)

Monitor ユーティリティは，アラインメント・フォルトに関する情報を表示するように拡張されました。この新しい MONITOR ALIGN コマンドは， Integrity サーバの OpenVMS でのみ有効で， Integrity サーバ・システムでの性能の問題をトラブルシューティングする際に役立ちます。

MONITOR ALIGN クラスは，秒あたりのアラインメント・フォルトの発生率と総アラインメント・フォルト数を，モード (カーネル・モード，エグゼクティブ・モード，スーパバイザ・モード，およびユーザ・モード) ごとに表示します。秒あたりのアラインメント・フォルト率が非常に高い場合は， SDA 経由で動作する Alignment Fault ユーティリティ (FLT) を使用し，アラインメント・フォルトの原因を解析してください。

Integrity サーバ・システムでは，すべてのアラインメント・フォルトはオペレーティング・システムで処理されるため，アラインメント・フォルトの発生率を追跡するためにカウンタをカウントアップすることができます。 Alpha では，アラインメント・フォルトはコンソールの PALcode で修正されるため，カウンタをカウントアップするためには大きなオーバヘッドが必要になります。そのため， MONITOR ALIGN コマンドは Integrity サーバでのみ使用できます。

ヘッダ・インクルード・ファイル$MONDEF も拡張され，新しい ALIGN クラスのレコード定義が追加されています。以前は，各クラス・タイプ・レコードの定数は用意されていませんでしたが， Version 8.3 の$MONDEF では，クラス番号用のシンボリック定数定義を MNR_CLS$K_xxx としてインクルードします。

次の例を参照してください。

```
$ monitor align
                        ALIGNMENT FAULT STATISTICS
                             on node MTDIB9
                         11-JAN-2006 16:58:07.25

                                    CUR        AVE        MIN        MAX

    Kernel Alignment Faults       19529.00   19529.00   19529.00   19529.00
    Exec Alignment Faults          7581.00    7581.00    7581.00    7581.00
    Super Alignment Faults            0.00       0.00       0.00       0.00
    User Alignment Faults        164972.00  164972.00  164972.00  164972.00

 Total Alignment Faults       192082.00  192082.00  192082.00  192082.00
```

### 3.6.2 PROCESSES **クラス向けの新しいクラス名修飾子**

MONITOR ユーティリティの PROCESSES クラスでは，プロセスごとのモードの使用状況を監視するために， 4 つの新しいクラス名修飾子を使用することができます。これは，さまざまな CPU モードを最も多く消費しているプロセスを見つけるのに役立ちます。たとえば，大量のスーパバイザ・モードが使用されていることが

MONITOR MODES コマンドで表示された場合は，新しい MONITOR PROCESSES /TOPSUPERVISOR の表示によって，どのプロセス (したがってどのユーザ) が消費しているかが分かります。

新しい修飾子を次の表に示します。

**表 3–2 MONITOR ユーティリティの PROCESSES クラスに対するクラス名修飾子**

| コマンドと修飾子 | 説明 |
|---|---|
| MONITOR PROCESSES /TOPKERNEL | カーネル・モードの使用量が上位のプロセス |
| MONITOR PROCESSES /TOPEXECUTIVE | エグゼクティブ・モードの使用量が上位のプロセス |
| MONITOR PROCESSES /TOPSUPERVISOR | スーパバイザ・モードの使用量が上位のプロセス |
| MONITOR PROCESSES /TOPUSER | ユーザ・モードの使用量が上位のプロセス |

詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』の「Monitor ユーティリティ (MONITOR)」の章を参照してください。

### 3.6.3 MONITOR PROCESSES/TOPSUPERVISOR の例

新しい MONITOR PROCESSES/TOPSUPERVISOR 修飾子を使用すると，どのプロセスがスーパバイザ・モードで CPU を最も消費しているかが分かります。この修飾子についての詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』を参照してください。

次の例は，『OpenVMS システム管理者マニュアル』の次の版の「システムに関する情報の入手」の章の「リアルタイム表示監視の使用法」の項に記載されます。

例

```
$ MONITOR PROCESSES/TOPSUPERVISOR
```

このコマンドを実行すると，スーパバイザ・モードの CPU 時間を最も消費している 16 個のプロセスを示した棒グラフが表示されます。値は， 1 秒あたりの時間が 10ms クロック単位の数で表現されます。

コマンドを実行すると，次のように表示されます。

```
                OpenVMS Monitor Utility
            TOP SUPERVISOR MODE PROCESSES
                   on node QUEBIT
              7-DEC-2005 14:04:24.19
```

```
                        0         25        50        75       100
                        + - - - - + - - - - + - - - - + - - - - +
74E000AD  BATCH_3   5  **
74E000AC  BATCH_2   4  *
74E000AA  BATCH_1   3  *
74E000AB  _RTA3:    3  *
```

## 3.7 EVA コントローラと MSA コントローラのアクティブ - アクティブ機能向けのマルチパスの機能強化

Enterprise Virtual Array (EVA) 4000/6000/8000 ストレージ・システムと MSA1500 ストレージ・システムのコントローラは， AO (active optimized) パスと ANO (active non-optimized) パスを備えています。この機能は， EVA 3000/5000 ストレージ・システムでも提供されています。 ANO パスを使用した場合，読み込み入出力性能のペナルティがあります。

OpenVMS のマルチパス機能は，入出力性能を向上させるために， AO パスと ANO パスを区別するように拡張されました。性能向上の度合いは，入出力サイズやキューの長さによって変わります。キューが長いほどユーザが体感する性能向上も大きくなります。

OpenVMS のマルチパス機能についての詳細は，『OpenVMS Cluster 構成ガイド』を参照してください。これらのストレージ・システムのコントローラについての詳細は，次の URL を参照してください。

`http://www.hp.com/country/us/en/prodserv/storage.html`

EVA 4000/6000/8000 コントローラについての詳細は，[Browse by capacity]の[Enterprise]を選択し，目的のストレージ・システムを選択してください。

同様に， EVA 3000 および 4000 コントローラについての詳細は，[Browse by capacity]の[Mid-range]を選択し，目的のストレージ・システムを選択してください。 MSA 1500 コントローラについての詳細は，[Browse by capacity]の[Entry-level]を選択し，[MSA 1500]を選択してください。

## 3.8 OpenVMS クラスタ・インターコネクト

OpenVMS Version 8.3 の LAN ベースのクラスタ通信ドライバ (PEdriver) では，以下の機能が追加されています。

- データの圧縮
- 数ギガビットの通信速度と長距離性能の強化

データの圧縮は，ノード間の LAN の速度によってデータ転送速度が制限されており，かつ CPU 能力に余裕がある場合に，2 台の OpenVMS ノード間でのデータ転送に要する時間を短縮するために使用します。たとえば，シャドウ・コピー時間を短縮したり，E3 や DS3，FDDI，100Mb Ethernet などの比較的低速な回線で接続されているディザスタ・トレラント・クラスタ・サイト間での MSCP サービス性能を向上させるために使用します。PEdriver のデータ圧縮機能は，SCACP，Availability Manager，SYSGEN パラメータ NISCS_PORT_SERV のいずれかを使用して有効にすることができます。

ノード間で転送中となりえるパケットの数は，LAN リンクの速度とノード間の距離に応じて増やす必要があります。これまで，PEdriver の送受信ウィンドウ (バッファリング容量) は固定で，31 個の処理待ちのパケットを格納することができました。OpenVMS Version 8.3 からは，PEdriver は，ノード間のクラスタ通信で現在使用されているローカルおよびリモートの LAN アダプタの速度に基づいて，自動的に送受信ウィンドウ・サイズ (他のネットワーク・プロトコルではパイプ・クォータと呼ばれることもあります) を選択します。さらに，SCACP および Availability Manager では，自動的に選択されるウィンドウ・サイズを管理者が指定変更することができます。

詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』および『HP OpenVMS Availability Manager User's Guide』の SCACP ユーティリティの章と NISCS_PORT_SERV を参照してください。

## 3.9 OpenVMS オペレーティング・システム媒体のパッチ関連のメニュー・オプション

OpenVMS オペレーティング・システムの配布媒体のメイン・メニューに，パッチ関連の操作を行うための新しいオプション (7) が追加されました。オプション 7 を選択すると，サブメニューが表示され，パッチ・キットの検索，パッチのインストール，回復データが存在する最近のパッチの削除，回復データの表示と削除を行うことができます。これにより，オペレーティング・システムがブートできない場合 (PCSI PRODUCT コマンドが実行できない場合) でもこれらの操作を実行することができます。メニュー・オプションの例を示します。

```
Please choose one of the following:

  1) Upgrade, install or reconfigure OpenVMS I64 Version X8.3-BBV
  2) Display layered products that this procedure can install
  3) Install or upgrade layered products
  4) Show installed products
  5) Reconfigure installed products
  6) Remove installed products
  7) Find, Install, or Undo patches; Show or Delete Recovery Data
  8) Execute DCL commands and procedures
  9) Shut down this system

Enter CHOICE or ? for help: (1/2/3/4/5/6/7/8/9/?)
```

メイン・メニューのオプション 2 とオプション 3 も引き続きパッチ操作を実行するために使用できますが，新しいオプション 7 のサブメニュー操作を使用することをお勧めします。これらの操作で提供される機能はより広範囲で信頼性が高くなっています。たとえば，新しいサブメニュー・オプションを使用してパッチをインストールする場合は，回復データが自動的に保存されます。また，新しいオプションでは，デフォルトのターゲット・デバイスに加え，操作を行う対象を指定できます。対象の指定では，ワイルドカードを使用できます。

## 3.10 PCSI ユーティリティの機能拡張

OpenVMS Version 8.3 では， PCSI ユーティリティが以下のように拡張されています。

- PRODUCT ANALYZE PDB コマンドを使用した，製品データベースの手動検証
- PRODUCT INSTALL コマンドを使用した，製品データベースの自動検証
- ODS-5 ボリュームの完全サポート
- いくつかの PRODUCT コマンドによる，署名済み製品キットの検証

### 3.10.1 PRODUCT ANALYZE PDB

新しい PRODUCT ANALYZE PDB コマンドは，製品データベースの構造的な一貫性を検証し，状況によっては，軽微な修復を行います。コマンドの動作を以下に示します。

- ルート・ファイル PCSI$ROOT.PCSI$DATABASE で参照されているすべての SYS$SYSTEM:*.PCSI$DATABASE ファイルを読み込む。
- これらのファイルのすべてのフィールドにおいて，構文が正しいことを確認する。
- 既知の破損パターンが見つかり，修復が可能な場合には，軽微な修復を自動的に行う。
- 復旧できない破損が見つかった場合は，データベースを再構築する方法についての手順を表示する。

コマンドで指定可能な修飾子については，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』の「POLYCENTER Software Installation ユーティリティ (PCSI)」の章を参照してください。

### 3.10.2 製品データベースの自動的な検証

PRODUCT INSTALL コマンドは，操作開始時と，製品データベースを更新する際に，製品データベースを自動的に検証するように拡張されました。検証動作と修復動作は， PRODUCT ANALYZE PDB コマンドとほぼ同じです。データベースを変更する他の PRODUCT コマンド (PRODUCT RECONFIGURE や PRODUCT REMOVE など) も，検証パスを実行します。

### 3.10.3 ODS-5 ボリュームのサポート

PCSI ユーティリティは，拡張された ODS-5 ディレクトリとファイル指定構文を完全にサポートするようになりました。 OpenVMS Version 8.3 の前では，製品キットを ODS-5 ディスクにインストールすることはできましたが，すべてのファイルは ODS-2 の構文規則を使用したインストール先ボリュームに配置されていました (たとえば，ファイル名は大文字だけでした)。 ODS-5 のサポートにより，キットの開発者は，以下の拡張された構文を使用したファイルをインストールするキットを作成することができます。

- 大文字，小文字，大文字と小文字が混在した名前
- ファイル指定での複数のドット
- 拡張文字セット
- 長い名前

拡張された構文を使用するには，製品記述ファイル (PDF) 中の DIRECTORY 文または FILE 文の名前指定パラメータを引用符で囲みます。たとえば，次の 2 つの行を含む PDF ファイルがあるとします。

```
file [test]file_one.txt ;
file "[test]File_Two.txt" ;
```

インストール先ディスクが ODS-5 ボリュームの場合， OpenVMS Version 8.3 用の PCSI ユーティリティは，最初のファイルを FILE_ONE.TXT としてインストールし， 2 番目のファイルを File_Two.txt としてインストールします。旧バージョンの PCSI ユーティリティとの互換性を保つため，引用符で囲まれていない指定は大文字のみの名前に変換されます。

また，製品の開発者は，次の形式の新しい PDF ステートメントを使用することで，ボリュームが ODS-2 と ODS-5 のどちらなのかをテストすることができます。

```
IF ( < FILESYSTEM { ODS-2 | ODS-5 } [ VOLUME { DESTINATION | SYSTEM } ] > ) ;
```

たとえば，インストール先ボリュームが ODS-5 の場合にだけ条件的にステートメントを処理するには，次の行を使用します。

```
IF ( < FILESYSTEM ods-5 VOLUME destination > ) ;
...
END IF ;
```

### 3.10.4 製品キットの Secure Delivery のサポート

いくつかの PRODUCT コマンドが拡張され，製品キットの Secure Delivery がサポートされました。 Secure Delivery については，第 5.2 節を参照してください。 PCSI ユーティリティに，以下の機能が追加されました。

- 製品キットを読み込むすべての PRODUCT コマンドで，関連付けられているデジタル署名ファイル (マニフェストとも呼ばれます) がソース・ディレクトリにあれば，署名済みキットが使用の前に検証されるようになりました。該当するコマンドは， PRODUCT CONFIGURE， COPY， EXTRACT， INSTALL， LIST， RECONFIGURE，および REGISTER PRODUCT です。デジタル署名ファイルのファイル名は製品キットと同じで，ファイル・タイプには "_ESW"が付加されます。たとえば， HP-I64VMS-TEST-V0100–1.PCSI$COMPRESSED_ESW のようになります。
- 新しい PCSI ユーティリティでは署名されていないキットをインストールすることができます。以前のバージョンの PCSI ユーティリティでも，署名済みのキットをインストールすることができますが，検証は行われません。
- キットの検証は，/OPTIONS=NOVALIDATE_KIT 修飾子で無効にすることができます。
- PRODUCT COPY コマンドは，指定されたキットとそのマニフェスト・ファイル (存在する場合) の両方をコピーします。
- PRODUCT SHOW HISTORY コマンドには， VAL という新しいフィールドがあり，製品の検証状態を示します。状態コードには以下のものがあります。
  - VAL — キットは正常に検証されました。
  - SYS — キットは， OpenVMS のインストールまたはアップグレード中に OS の媒体からインストールされましたが，検証はされていません。
  - (U) — キットは署名されていないため検証されていません。
  - (M) — キットのソース・ディレクトリにマニフェスト・ファイルがないため検証されていません。
  - (D) — ユーザ要求によって検証が無効にされているため，キットは検証されていません。
  - (C) — CDSA が使用できないためキットは検証されていません。
  - ハイフン — 操作が対象外です (製品の削除など)
  - 空白 — 操作は Secure Delivery をサポートしていない PCSI のバージョンで実行されました。

- PRODUCT LIST コマンドの出力に，キットが署名されているか，マニフェスト・ファイルあるかなどの情報が追加されています。

Secure Delivery をサポートする PRODUCT コマンドと修飾子についての詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』の「POLYCENTER Software Installation ユーティリティ (PCSI)」の章を参照してください。

### 3.10.5 2 つの修飾子におけるデフォルトの変更

OpenVMS Version 8.3 では，以下のコマンドで修飾子のデフォルト値が変更されました。

- PRODUCT INSTALL

  以下のデフォルトが変更されました。

  — 製品をインストールする場合，/RECOVERY_MODE 修飾子がデフォルトとなりました (以前のデフォルトは/NORECOVERY_MODE でした)。

  — パッチと必須のアップデート・キットをインストールする場合には，/SAVE_RECOVERY_DATA 修飾子がデフォルトとなりました (以前のデフォルトは/NOSAVE_RECOVERY_DATA でした)。

- PRODUCT RECONFIGURE

  製品を再構成する場合，/RECOVERY_MODE 修飾子がデフォルトとなりました (以前のデフォルトは/NORECOVERY_MODE でした)。

## 3.11 SANCP ユーティリティ

SANCP ユーティリティを使用すると，特定の Fibre Channel ストレージ・ポート上の論理ユニット番号 (LUN) へのパス全体でのアクティブな入出力の数を制限することができます。ストレージ・ポートは，個別またはワイルドカードによる WWID (World Wide ID) または製品 ID 部分文字列で選択することができます。

SANCP ユーティリティは，コマンド行で渡されたコマンドと修飾子を処理するため， DCL スクリプトから実行できます。コマンドを指定せずに起動すると，プロンプトが表示されます。 SANCP ユーティリティについての詳細は， SANCP のヘルプ・ファシリティを参照してください。

## 3.12 SAS ユーティリティ (I64 のみ)

SAS ユーティリティ (SAS$UTIL) は， OpenVMS システムの管理と診断を行うツールです。このユーティリティを使用して， HP 8 Internal Port Serial Attached SCSI

Host Bus Adapter (SAS コントローラ) の IR (Integrated RAID) 機能を構成することが可能です。

IR (Integrated RAID) は，高い性能，ストレージ容量， RAID 構成による冗長性のいずれかまたはすべてが必要な場合に使用されます。 OpenVMS Version 8.3 は，Integrated RAID 1 または IM (Integrated Mirroring) とそれに関連するグローバル・ホット・スペア機能だけをサポートします。

詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』を参照してください。

## 3.13 SCACP ユーティリティ

OpenVMS Version 8.3では， SCACP に以下の新機能が追加されています。

- データの圧縮
- 数ギガビットの通信速度と距離性能の強化

以降の項では，これらの新機能について説明します。

### 3.13.1 データ圧縮の管理

SCACP SET VC コマンドに/COMPRESSION (または/NOCOMPRESSION) 修飾子が追加されました。これらの修飾子を使用すると，指定した PEdriver VC による圧縮されたデータの送信が有効または無効になります。省略時の指定は /NOCOMPRESSION です。

また，システム・パラメータ NISCS_PORT_SERV のビット 3 を設定することで，VC での圧縮の使用を有効にすることができます。/NOCOMPRESSION 修飾子では， NISCS_PORT_SERV のビット 2 を設定することで有効にした圧縮は無効になりません。

### 3.13.2 数ギガビットのスケーリング

SET VC COMMAND /WINDOW=RECEIVE_SIZE=value修飾子および /WINDOW=TRANSMIT_SIZE=value修飾子を使用して， PEdriver VC に対して自動的に計算された送受信ウィンドウ・サイズを指定変更することができます。/WINDOW=NORECEIVE_SIZE 修飾子と/WINDOW=NOTRANSMIT_SIZE 修飾子は，ウィンドウ・サイズの管理者による指定変更を取り消すために使用できます。/WINDOW=(NORECEIVE_SIZE,NOTRANSMIT_SIZE) を使用すると，送信ウィンドウ・サイズと受信ウィンドウ・サイズの両方の指定変更を， 1 つのコマンドで取り消すことができます。

これらの新しいコマンド修飾子を使用すると，VC で十分な受信バッファが確保され，肯定応答を待たずに十分なパケットを送信でき，ノード間の帯域幅を最大限に利用することができます。VC の無駄な閉鎖を避けるために，これらのコマンドには実行順序に制限があります。これらの制限については，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』および SCACP のヘルプ・ファシリティを参照してください。

新しい SCACP コマンド CALCULATE WINDOW_SIZE を使用すると，VC が使用する最大ウィンドウ・サイズを調べることができます。このコマンドには 2 つの必須の修飾子/DISTANCE=KILOMETERS (または MILES)=nと/SPEED=sがあります。ここで，nは，2 つのノード間のケーブル長で，sは，ノード間のクラスタ通信で使用しているすべてのリンクの帯域幅の合計です。

SCACP SHOW VC コマンドでは，VC 上で圧縮が有効になっているかどうかが表示されるようになり，送受信ウィンドウ・サイズの管理設定の欄が追加されています。

詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』および『HP OpenVMS Availability Manager User's Guide』の SCACP ユーティリティの章と NISCS_PORT_SERV を参照してください。

## 3.14 HP OpenVMS I64 シリアル・マルチプレクサ (MUX) のサポート (I64 のみ)

RS232 シリアル回線とマルチプレクサは，従来の端末接続から低速のシステム間通信，さらには遠隔機器との通信まで，さまざまな作業で使用します。OpenVMS では，従来からオプション・カード・ベースのマルチプレクサと同時にシリアル回線の追加をサポートしてきました。このソリューションでは，専用の入出力スロットが必要で，使用できるオプション・カードの選択肢が限られていました。

ユニバーサル・シリアル・バス (USB) が業界標準のプラットフォームで広く採用されるにしたがい，OpenVMS では，オプション・カード・ベースのマルチプレクサから離れ，HP Integrity サーバにシリアル回線を追加するために USB が採用されました。構成にかかわらず 8 回線または 16 回線が収容できる 1 つまたは 2 つのオプション・カードを使用するのではなく，要件を正確に満たすように USB を構成できるようになりました。

テストの結果，USB ベースのシリアル・マルチプレクサは，オプション・カードと同程度 (またはそれ以上) の性能を発揮し，システムのオーバヘッドも少ないことが分かりました。実際に，オーバヘッドはオプション・カード・ベースのマルチプレクサよりも小さいという結果が出ています。

HP MUX のサポートについての詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』を参照してください。

## 3.15 Spinlock Trace **ユーティリティ** (SPL)

SDA を通じて使用可能な Spinlock Trace ユーティリティは，さまざまなスピンロック保持時間とスピンロック待ち時間を，サイクル単位ではなくナノ秒単位でレポートするように変更されました。これにより，サイクル・カウンタ周波数が異なるシステム間で収集された情報を容易に比較できるようになりました。

表示も拡張され， P2 空間および S2 空間で動作させるようにコードを移行する際に，PC の完全な 64 ビット・アドレスが表示されるようになりました。また， PC アドレスのデコード処理が拡張され， I64 のモジュール，ルーチン，およびオフセットが表示されるようになりました。 SPL ANALYZE コマンドと SPL SHOW COLLECT コマンドでは，特に指定しなくても追加の PC デコードが表示されますが， SPL SHOW TRACE [/SUMMARY]では/FULL 修飾子を指定した場合にだけその情報が表示されます。これは，すでに表示が込み入っているためです。

## 3.16 HP OpenVMS System Analysis Tools

ここでは， System Analysis Tools ユーティリティで提供される新機能について説明します。本リリースでは，『HP OpenVMS System Analysis Tools Manual』は更新されていませんが，本書と『HP OpenVMS V8.3 リリース・ノート[翻訳版]』に記載されている追加と変更は， SDA ユーティリティのオンライン・ヘルプと，ANALYZE および System Service Logging に関連するコマンドのオンライン・ヘルプに含まれています。

### 3.16.1 System Dump Debugger

System Dump Debugger (SDD) は， OpenVMS Alpha だけでなく， OpenVMS I64 でもサポートされるようになりました。

### 3.16.2 System Dump Analyzer

ここでは，以下の新しい SDA のコマンドまたは SDA の拡張コマンドと，新しい呼び出し可能ルーチンの拡張機能， SDA コマンドのいくつかの新しい修飾子について説明します。

- COLLECT コマンド
- SHOW CLASS コマンド
- SHOW EFI コマンド
- SHOW VHPT コマンド
- VALIDATE POOL コマンド
- VALIDATE PROCESS コマンド

- CLUE REGISTER
- CLUE SCSI
- SDA$CBB_BOOLEAN_OPER ルーチン
- SDA$CBB_CLEAR_BIT ルーチン
- SDA$CBB_COPY ルーチン
- SDA$CBB_FFC ルーチン
- SDA$CBB_FFS ルーチン
- SDA$CBB_INIT ルーチン
- SDA$CBB_SET_BIT ルーチン
- SDA$CBB_TEST_BIT ルーチン
- SDA$DELETE_PREFIX ルーチン
- SDA$FID_TO_NAME ルーチン
- SDA$GET_FLAGS ルーチン

Common Bitmask Block (CBB) ルーチン SDA$CBB_xxxは，システムで使用している CPU について記述する CBB 構造体のローカルなコピーに対して使用するように設計されています。 CBB 構造体の長さは，少なくとも CBB$K_STATIC_BLOCK バイトと想定されます。これらのルーチンで使用されるさまざまな CBB 定数とフィールド名の定義は， SYS$LIBRARY:SYS$LIB_C.TLB 内の CBBDEF.H にあります。

一連のルーチンは，考えられる CBB 関連の操作をすべて網羅することを目的としたものではなく，必要であることが分かっている操作を提供するためのものです。 CPU について記述すること以外の目的で設定された CBB 構造体では，これらのルーチンが期待どおりに動作しないことがあります。

# COLLECT

OpenVMS Alpha と OpenVMS I64 上で，ファイル名変換データに対するファイル識別子を収集し， OpenVMS I64 上でのみアンワインド・データを処理します。

## フォーマット

COLLECT *[**修飾子**]*

## パラメータ

**なし**

## 修飾子

/LOG
COLLECT コマンドの進行状況に関する情報を表示します。たとえば，スキャン中のプロセス名や，アンワインド・データを収集するイメージの名前 (Integrity サーバ上) などを表示します。

/SAVE [=**ファイル名**]
収集データを個別のファイルに書き込みます。デフォルトでは，ファイル・タイプが.COLLECT で名前がダンプ・ファイルと同じファイルが，ダンプ・ファイルと同じディレクトリに作成されます。

/UNDO
以前 COLLECT コマンドで収集されたすべてのファイルまたはアンワインド・データを， SDA のメモリから削除します。 COLLECT/UNDO は，分析中のダンプ・ファイルに追加済みのファイルとアンワインド・データや，個別の収集ファイルにすでに書き込まれているファイルとアンワインド・データには影響を与えません。

## 説明

ダンプを分析する際に，システムのクラッシュ時にダンプ・ファイルに書き込むことができないデータが利用できると便利です。このデータには，ファイル識別子に関連した完全なファイル指定が含まれます。 OpenVMS for Integrity Servers の場合は，プロセス内でアクティブ化されたイメージのアンワインド・データも含まれます。

ダンプが書き出されたシステムでダンプを分析する場合は，現在の SDA セッションから COLLECT コマンドを使用して，このデータを収集することができます。分析のために別の場所にダンプをコピーしている場合は， COPY/COLLECT コマンドを使用してデータを収集し，書き出されるコピーに追加することができます。COPY/COLLECT コマンドを COLLECT コマンドの後に使用すると，すでに収集済みのデータがダンプ・コピーに追加されます。

すべてのファイルまたはアンワインド・データを正常に収集するためには，システムのクラッシュ時にマウントされていたすべてのディスクを再マウントし， SDA を実行しているプロセスからアクセスできるようにする必要があります。

COPY と COLLECT を 1 回の手順で実行できない場合は， COLLECT/SAVE コマンドで収集結果を個別のファイルに書き込み，後でダンプ・ファイルとともに使用できます。その後， COPY コマンドで 2 つのファイルを結合します。

# SHOW CLASS

システムまたは分析中のダンプでアクティブなスケジューリング・クラスに関する情報を表示します。

## フォーマット

SHOW CLASS *[**クラス名** | /ALL]*

## パラメータ

**クラス名**
表示するクラスの名前です。

## 修飾子

/ALL
すべてのアクティブなクラスの詳細を表示することを指示します。

## 説明

SDA は，システム内でアクティブなスケジューリング・クラスに関する情報を表示します。デフォルトでは，クラスの要約が表示されます。

# SHOW EFI (I64 **のみ**)

Extensible Firmware Interface (EFI) のデータ構造の情報を表示します。現在, SDA で提供される唯一の表示は EFI メモリ・マップです。

## フォーマット

SHOW EFI　*/MEMMAP [=ALL] [***範囲***]*

## パラメータ

**範囲**
表示するエントリまたはエントリの範囲を，以下の構文を使用して指定します。

m— エントリmを表示します。
m:n— エントリmからnを表示します。
m;n—mから始まるn個のエントリを表示します。

/MEMMAP=ALL とともに範囲を指定することはできません。

## 修飾子

/MEMMAP [=ALL]
EFI メモリ・マップを表示します。この修飾子は必須です。デフォルトでは， EPI メモリ・マップ中のruntime属性を持つエントリだけが表示されます。/MEMMAP=ALL 修飾子を指定すると，すべてのエントリが表示されます。

/MEMMAP=ALL 修飾子を指定すると同時に，表示対象のエントリの範囲を指定することはできません。

## 説明

SDA は，システムまたはダンプ内の EFI メモリ・マップを探して内容を表示します。範囲を指定しないと， SDA はメモリ・マップの位置とサイズに関する情報も表示します。

# SHOW VHPT (I64 **のみ**)

仮想ハッシュ・ページ・テーブル (VHPT) のデータを表示します。

## フォーマット

SHOW VHPT *[ /CPU = { n | * } [ /ALL ] [* **範囲** *] ]*

## パラメータ

**範囲**
表示するエントリまたはエントリの範囲を，以下の構文を使用して指定します。

m—VHPT エントリmを表示します。
m:n— mからnの VHPT エントリを表示します。
m;n—mから始まるn個の VHPT エントリを表示します。

範囲は，/CPU 修飾子で単独の CPU を指定した場合にのみ指定できます。

## 修飾子

/CPU = { n | * }
1 つまたはすべての CPU について， VHPT の詳細な内容を表示することを指示します。デフォルトでは， VHPT 情報の要約が表示されます。

/ALL
指定した CPU のすべての VHPT を表示します。/ALL を指定しない場合，有効なタグを持つエントリだけが表示されます。

## 説明

OpenVMS I64 システム上の仮想ハッシュ・ページ・テーブルの内容を表示します。デフォルトでは， VHPT エントリの要約が表示されます。複数の CPU を指定すると，個々の CPU について VHPT エントリの詳細が表示されます。単独の CPU を指定すると，その CPU の VHPT エントリが表示されます。

詳細表示では，以下の項目が表示されます。

**表** 3–3

| **エントリ** | VHPT **エントリ番号** |
|---|---|
| ビット | 以下の 1 つ以上のフラグ<br><br>P—Present<br>A—Accessed<br>D—Dirty<br>E—Exception deferral<br>I—Tag invalid (/ALL を指定した場合のみ表示) |
| MA | 以下のメモリ属性のいずれか<br><br>WB—Write Back<br>UC—Uncacheable<br>UCE—Uncacheable Exported<br>WC—Write Coalescing<br>NaT—NaTPage |
| AR/PL | ページのアクセス権と特権レベル。数字 (0-7) と英字 (K, E, S, U) で構成され，各モードでのページへのアクセス許可を決定します。 |
| KESU | 各モードでページに対して許されるアクセス。これは，前記のエントリ AR/PL の値を解釈したものです。アクセス・コードの説明については，『HP OpenVMS System Analysis Tools Manual』を参照してください。 |
| 物理アドレス | この VHPT エントリの開始物理アドレス |
| ページ・サイズ | この VHPT エントリで表されるページのサイズ。 VHPT エントリのページ・サイズの範囲は， 4KB ～ 4GB です。可能なすべてのページ・サイズが OpenVMS for Integrity Servers で使用されるわけではありません。 |
| タグ | VHPT エントリの変換タグ |
| Quad4 | デバッグ用に OpenVMS for Integrity Servers で記録された情報。このクォドワードの内容は変更される可能性があります。 |

# VALIDATE POOL

POOLCHECK バグチェックおよびシステム・ダンプを生成するときにシステム・プール割り当てルーチンが使用するのと同じアルゴリズムを使用して，すべての空きプール・パケットの POOLCHECK スタイルの破損を確認します。

## フォーマット

VALIDATE POOL *{ /ALL (d) | /BAP | /NONPAGED | /PAGED }*
*[ /HEADER | /MAXIMUM_BYTES [ =* n *] /SUMMARY ]*

## パラメータ

**なし**

## 修飾子

/ALL
すべての種類のプール (非ページング・プール，ページング・プール，バス・アドレス指定可能プール) に対して空きパケットを確認します。これは省略時の指定です。

/BAP
バス・アドレス指定可能プール内の空きパケットを確認します。

/HEADER
見つかった壊れている空きパケットの最初の 16 バイトだけを表示します。

/MAXIMUM_BYTES[=n]
見つかった壊れている空きパケットの最初のnバイトだけを表示します。値を指定せずに/MAXIMUM_BYTES を指定すると，デフォルトは 64 バイトとなります。

/NONPAGED
非ページング・プールの空きパケットを確認します。

/PAGED
ページング・プールの空きパケットを確認します。

/SUMMARY
見つかった壊れているプール・パケットの要約だけを表示します。

## 説明

VALIDATE POOL コマンドは，壊れている空きプール・パケットに関する情報を表示します。システム・パラメータの POOLCHECK または SYSTEM_CHECK のどちらかを使用してプールのチェックが有効になっている場合にだけ有効です (これらのシステム・パラメータについての詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』を参照してください)。

# VALIDATE PROCESS

プロセス・データ構造を検証します。現在利用できる唯一の検証は，空きプロセス・プール・パケットの POOLCHECK スタイルの破壊の確認です。この確認では，POOLCHECK バグ・チェックおよびシステム・ダンプを生成するときにシステム・プール割り当てルーチンが使用するのと同じアルゴリズムを使用します。

## フォーマット

VALIDATE PROCESS/POOL *[ = { P0 | P1 | IMGACT | ALL (d) } ]*
*[ /ADDRESS = PCB* **アドレス** | **プロセス名** | *ALL*
*| /ID = nn | /INDEX = nn | /NEXT |*
*/SYSTEM ]*
*[ /HEADER | /MAXIMUM_BYTES [ = n ]*
*/SUMMARY ]*

## パラメータ

ALL
システム内のすべてのプロセスを検証することを指示します。

**プロセス名**
検証対象のプロセスの名前です。プロセス名に含めることができるのは，最大 15 文字の大文字，数字，アンダスコア (_)，ドル記号 ($)，コロン (:)，およびその他いくつかのプリント可能文字です。その他の文字 (小文字を含む) が含まれている場合は，プロセス名を引用符 (" ") で囲む必要があります。

## 修飾子

/ADDRESS = PCB **アドレス**
検証対象のプロセスのプロセス制御ブロック (PCB) アドレスを指定します。

/HEADER
見つかった壊れている空きパケットの最初の 16 バイトだけを表示します。

/ID = nn
/INDEX = nn
検証対象のプロセスを，システムのソフトウェア PCB リストのインデックス，またはプロセス識別番号で指定します。nnには以下の値を指定できます。

- プロセス・インデックス自体

- プロセス識別番号 (PID) または拡張 PID のロングワード。 SDA は，この値から正しいインデックスを取り出します。複数のカーネル・スレッドを持つプロセスのどのスレッドの PID や拡張 PID でも指定できます。以降のコマンドで表示されるスレッド固有のデータは，そのスレッドのデータです。

あるプロセスに対するこれらの値を取得するには， SDA コマンド SHOW SUMMARY/THREADS を実行します。/ID=nn修飾子と/INDEX=nn修飾子は，どちらかを使用できます。

/MAXIMUM_BYTES[=n]
見つかった壊れている空きパケットの最初のnバイトだけを表示します。値を指定せずに/MAXIMUM_BYTES を指定すると，デフォルトは 64 バイトとなります。

/NEXT
プロセス・リスト内の次のプロセスを探して，そのプロセスを検証することを指示します。プロセス・リスト内にそれ以上プロセスがない場合は， SDA からエラーが返されます。

/POOL[ = { P0 | P1 | IMGACT | ALL (d) } ]
プロセス・プールの検証を実行します。この修飾子は必須です。/POOL 修飾子に対してキーワードを指定することで，検証するプロセス・プール (P0， P1，イメージ・アクティベータ・プール) をユーザが指定することができます。デフォルトでは，すべてのプロセス・プールが検証されます。

/SUMMARY
見つかった壊れているプール・パケットの要約だけを表示します。

/SYSTEM
この修飾子は， SET PROCESS/SYSTEM および SHOW PROCESS/SYSTEM との互換性のためにあります。検証可能なシステム・プロセスに関連付けられているプールはありません。 SDA は現在のプロセス・コンテキストをシステム・プロセスに設定し，次のテキストを出力します。

```
Options ignored for System process:  POOL
```

## 説明

VALIDATE PROCESS コマンドは，プロセス名で指定されたプロセス，/ID または /INDEX 修飾子で指定されたプロセス，システムのプロセス・リスト内の次のプロセス，システム・プロセス，すべてのプロセスのいずれかを検証します。 VALIDATE PROCESS コマンドは，修飾子とパラメータの指定に従って，前述のように暗黙的な SET PROCESS コマンドを実行します。デフォルトでは， VALIDATE PROCESS コマンドは，『HP OpenVMS System Analysis Tools Manual』で定義されているように， SDA の現在のプロセスを検証します。

現在利用できる唯一の検証は，空きプール・パケットの POOLCHECK スタイルの破壊の確認です。このコマンドは，システム・パラメータの POOLCHECK または SYSTEM_CHECK のどちらかを使用してプールのチェックが有効になっている場合にだけ有効です (これらのシステム・パラメータについての詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』を参照してください)。

**プロセス名**，/ADDRESS，/ID，/INDEX，/NEXT，/SYSTEM のいずれかを使用してプロセスを指定すると，そのプロセスが，以降のコマンドで使用される SDA の現在のプロセスとなります。

# CLUE REGISTER

クラッシュ CPU のアクティブなレジスタ・セットを表示します。 CLUE REGISTER コマンドは，クラッシュ・ダンプの分析にのみ使用できます。

## フォーマット

CLUE REGISTER *[/CPU [cpu-id | ALL]*
*| /PROCESS [/ADDRESS=n | INDEX=n*
*| /IDENTIFICATION=n |* **プロセス名** *| ALL]]*

## パラメータ

ALL
/CPU とともに使用すると，システム内のすべての CPU に関する情報を要求します。/PROCESS とともに使用すると，システムに存在するすべてのプロセスに関する情報を要求します。

cpu-id
/CPU とともに使用し，情報を表示する CPU の番号を指定します。cpu-idパラメータを使用すると， CLUE REGISTER コマンドは暗黙的に SET CPU コマンドを実行し，指定された CPU を以降の SDA コマンドでの現在の CPU とします。

**プロセス名**
/PROCESS とともに使用し，情報を表示するプロセスの名前を指定します。**プロセス名**パラメータ，/ADDRESS 修飾子，/INDEX 修飾子，/IDENTIFICATION 修飾子のいずれかを使用すると， CLUE REGISTER コマンドは暗黙的に SET PROCESS コマンドを実行し，指定されたプロセスが以降の SDA コマンドでの現在のプロセスとなります。システム内のプロセスの名前を確認するには， SHOW SUMMARY コマンドを実行します。

**プロセス名**に含めることができるのは，最大 15 文字の英数字，アンダスコア(_)，およびドル記号($)です。その他の文字が含まれている場合は，**プロセス名**を引用符 (") で囲む必要があります。

## 修飾子

/ADDRESS=n
CLUE REGISTER/PROCESS とともに使用し，対象とするプロセスの PCB アドレスを指定します。

/CPU [cpu-id | ALL]
CPU のレジスタを表示することを示します。 CPU を番号で指定するか，すべての CPU を示す ALL を指定します。

/IDENTIFICATION=n
CLUE REGISTER/PROCESS とともに使用し，対象とするプロセスの識別番号を指定します

/INDEX=n
CLUE REGISTER/PROCESS とともに使用し，対象とするプロセスのインデックスを指定します。

/PROCESS [**プロセス名** | ALL]
プロセスのレジスタを表示することを示します。プロセスは，修飾子/ADDRESS, /IDENTIFICATION，/INDEX で指定するか，名前で指定します。すべてのプロセスを対象とする場合には ALL を指定します。

## 説明

CLUE REGISTER コマンドは，クラッシュ CPU のアクティブなレジスタ・セットを表示します。また，既知のデータ構造をすべて識別し，システムの仮想アドレスをすべてシンボル化し，プロセッサ状態 (PS) を解釈し， R0 を条件コードとして解釈します。

/CPU と/PROCESS のどちらも指定しないと，パラメータ (cpu-idや**プロセス名**) は無視され， SDA の現在のプロセスのレジスタが表示されます。

# CLUE SCSI

SCSI と Fibre Channel に関する情報を表示します。

## フォーマット

CLUE SCSI *{/CONNECTION=n | /PORT=n | /REQUEST=n | /SUMMARY}*

## 修飾子

/CONNECTION=SCDT **アドレス**
SCSI 接続に関する情報を表示し， SCDT アドレスが示す SCSI 接続記述子のデータ構造をデコードします。

/PORT=SPDT **アドレス**
すべてのポート記述子または SPDT アドレスが示す特定のポート記述子を表示します。

/REQUEST=SCDRP **アドレス**
SCSI 要求に関する情報を表示し， SCDRP アドレスが示す SCSI クラス・ドライバ要求パケットをデコードします。

/SUMMARY
すべての SCSI および FC のポートとデバイスの要約と，その種類とリビジョンを表示します。

# SDA$CBB_BOOLEAN_OPER

CBB のペアに対して，ブール演算を行います。

## フォーマット

int sda$cbb_boolean_oper *(CBB_PQ input_cbb, CBB_PQ output_cbb, int operation);*

## 引数

input_cbb

OpenVMS **用法** address
**データ型** CBB **構造体**
**アクセス** **読み取り専用**
**受け渡し方** **参照渡し**

1 番目の (入力) CBB 構造体のアドレス。

output_cbb

OpenVMS **用法** address
**データ型** CBB **構造体**
**アクセス** **読み書き**
**受け渡し方** **参照渡し**

2 番目の (出力) CBB 構造体のアドレス。

operation

OpenVMS **用法** longword
**データ型** **ロングワード (符号なし)**
**アクセス** **読み取り専用**
**受け渡し方** **値渡し**

実行する演算を以下の演算の中から指定します。

- CBB$C_OR—2 つの CBB の論理和を実行し，結果 (B = A | B) を出力 CBB に書き込みます。
- CBB$C_BIC—2 つの CBB の補数を使用した論理積を実行し，結果 (B = B & ~A) を出力 CBB に書き込みます。

## 説明

2 の CBB 構造体に対してブール演算を実行し，結果を 2 番目の (出力) 構造体に書き込みます。

## 返される状態値

| | |
|---|---|
| SS$_WASCLR | 結果の出力 CBB のビットがすべてオフの場合，値 0 が返されます。 |
| SS$_WASSET | 結果の出力 CBB のいずれかのビットがオンの場合，値 1 が返されます。 |
| SS$_BADPARAM | 入力 CBB と出力 CBB の有効なビット数が異なります。 |

# SDA$CBB_CLEAR_BIT

CBB 中の指定されたビットをクリアします。

## フォーマット

int sda$cbb_clear_bit *(CBB_PQ cbb, int bit);*

## 引数

cbb

OpenVMS **用法** address
**データ型** CBB **構造体**
**アクセス** **読み書き**
**受け渡し方** **参照渡し**

変更対象の CBB 構造体のアドレス。

bit

OpenVMS **用法** longword
**データ型** **ロングワード (符号なし)**
**アクセス** **読み取り専用**
**受け渡し方** **値渡し**

CBB 中のクリアするビット。ビット番号が -1 の場合，すべてのビットがクリアされます。

## 説明

CBB 中の指定されたビットまたはすべてのビットをクリアします。

## 返される状態値

| | |
|---|---|
| SS$_NORMAL | 正常終了 |
| SS$_BADPARAM | ビット番号が範囲外です。 |

# SDA$CBB_COPY

ある CBB の内容を別の CBB にコピーします。

## フォーマット

int sda$cbb_copy *(CBB_PQ input_cbb, CBB_PQ output_cbb);*

## 引数

input_cbb

OpenVMS **用法** address
**データ型** CBB **構造体**
**アクセス** **読み取り専用**
**受け渡し方** **参照渡し**

コピー元の CBB 構造体のアドレス。

output_cbb

OpenVMS **用法** address
**データ型** CBB **構造体**
**アクセス** **書き込み専用**
**受け渡し方** **参照渡し**

コピー先の CBB 構造体のアドレス。

## 説明

指定した CBB をコピーします。

## 返される状態値

なし

# SDA$CBB_FFC

CBB 中で最初にオフになっているビットを探します。

## フォーマット

int sda$cbb_ffc *(CBB_PQ cbb, int start_bit);*

## 引数

cbb

OpenVMS **用法** address
**データ型** CBB **構造体**
**アクセス** **読み取り専用**
**受け渡し方** **参照渡し**

検索対象の CBB 構造体のアドレス。

start_bit

OpenVMS **用法** longword
**データ型** **ロングワード (符号なし)**
**アクセス** **読み取り専用**
**受け渡し方** **値渡し**

CBB 中の検索を開始する最初のビット。

## 説明

指定されたビットから CBB 構造体を検索し，オフになっているビットを探します。

## 返される状態値

| | |
|---|---|
| Bit_number | オフになっているビットが見つかった場合，そのビット番号が返されます。オフになっているビットが見つからない場合 (start_bit から cbb->cbb$l_valid_bits までのすべてのビットがオンの場合) は，有効なビットの数が返されます。 |

# SDA$CBB_FFS

CBB 中で最初にオンになっているビットを探します。

## フォーマット

int sda$cbb_ffs *(CBB_PQ cbb, int start_bit);*

## 引数

cbb

OpenVMS **用法** address
**データ型** CBB **構造体**
**アクセス** **読み取り専用**
**受け渡し方** **参照渡し**

検索対象の CBB 構造体のアドレス。

start_bit

OpenVMS **用法** longword
**データ型** **ロングワード (符号なし)**
**アクセス** **読み取り専用**
**受け渡し方** **値渡し**

CBB 中の検索を開始する最初のビット。

## 説明

指定されたビットから CBB 構造体を検索し，オンになっているビットを探します。

## 返される状態値

| | |
|---|---|
| Bit_number | オンになっているビットが見つかった場合，そのビット番号が返されます。オンになっているビットが見つからない場合 (start_bit から cbb->cbb$l_valid_bits までのすべてのビットがオフの場合) は，有効なビットの数が返されます。 |

# SDA$CBB_INIT

CBB 構造体を既知の状態に初期化します。

## フォーマット

void sda$cbb_init *(CBB_PQ cbb);*

## 引数

cbb

OpenVMS **用法** address
**データ型** CBB **構造体**
**アクセス** **読み取り専用**
**受け渡し方** **参照渡し**

初期化対象の CBB 構造体のアドレス。

## 説明

レイアウトを示す CBB のフィールドは, CPU CBB に必要な内容に初期化されます。実際のビットマスクはゼロになります。

## 返される状態値

なし

# SDA$CBB_SET_BIT

CBB の指定されたビットをオンにします。

## フォーマット

int sda$cbb_set_bit *(CBB_PQ cbb,int bit);*

## 引数

cbb

OpenVMS **用法** address
**データ型** CBB **構造体**
**アクセス** **読み書き**
**受け渡し方** **参照渡し**

変更対象の CBB 構造体のアドレス。

bit

OpenVMS **用法** longword
**データ型** **ロングワード (符号なし)**
**アクセス** **読み取り専用**
**受け渡し方** **値渡し**

CBB 中のオンにするビット。ビット番号が -1 の場合，すべてのビットがオンになります。

## 説明

CBB 中の指定されたビットまたはすべてのビットをオンにします。

## 返される状態値

| | |
|---|---|
| SS$_NORMAL | 正常終了 |
| SS$_BADPARAM | ビット番号が範囲外です。 |

# SDA$CBB_TEST_BIT

CBB 中の指定されたビットをテストします。

## フォーマット

int sda$cbb_test_bit *(CBB_PQ cbb,int bit);*

## 引数

cbb

OpenVMS **用法** address
**データ型** CBB **構造体**
**アクセス** **読み取り専用**
**受け渡し方** **参照渡し**

テスト対象の CBB 構造体のアドレス。

bit

OpenVMS **用法** longword
**データ型** **ロングワード (符号なし)**
**アクセス** **読み取り専用**
**受け渡し方** **値渡し**

CBB 中のテスト対象のビット。

## 説明

CBB 中の指定されたビットをテストしその値を返します。

## 返される状態値

| | |
|---|---|
| SS$_WASSET | 指定されたビットはオンです。 |
| SS$_WASCLR | 指定されたビットはオフです。 |
| SS$_BADPARAM | ビット番号が範囲外です。 |

# SDA$DELETE_PREFIX

指定された接頭辞を持つすべてのシンボルを削除します。

## フォーマット

void sda$delete_prefix *(char *prefix);*

## 引数

prefix

OpenVMS **用法** char_string
**データ型** **文字列**
**アクセス** **読み取り専用**
**受け渡し方** **参照渡し**

接頭辞文字列のアドレス。

## 説明

このルーチンは，SDA シンボル・テーブルを検索し，指定された文字列で始まるすべてのシンボルを削除します。

## 返される状態値

なし

# SDA$FID_TO_NAME

ファイル識別子 (FID) を対応するファイル名に変換します。

## フォーマット

int sda$fid_to_name *(char *devptr, unsigned short *fidptr, char *bufptr, int buflen );*

## 引数

devptr

OpenVMS **用法** char_string
**データ型** **文字列**
**アクセス** **読み取り専用**
**受け渡し方** **参照渡し**

デバイス名文字列のアドレス。デバイス名は，割り当てクラス・デバイス名 (ALLDEVNAM) 形式で指定する必要がありますが，先頭のアンダスコアや末尾のコロンは無視されます。

fidptr

OpenVMS **用法** address
**データ型** **ファイル識別子**
**アクセス** **読み取り専用**
**受け渡し方** **参照渡し**

3 つの単語からなるファイル識別子のアドレス。

bufptr

OpenVMS **用法** char_string
**データ型** **文字列**
**アクセス** **書き込み専用**
**受け渡し方** **参照渡し**

ファイル名文字列を格納する文字列バッファのアドレス。

buflen

OpenVMS **用法** longword
**データ型** **ロングワード (符号なし)**
**アクセス** **読み取り専用**
**受け渡し方** **値渡し**

文字列バッファの最大長。

## 説明

現在のシステムを分析する際に，このルーチンは LIB$FID_TO_NAME を呼び出してファイル識別子をファイル名に変換します。ダンプを分析する際に，ファイル・データ・コレクションが利用可能で，指定されたディスクとファイル識別子がコレクションに含まれている場合は，記録されているファイル名が返されます。コレクションがない (システム全体，このディスク，このファイルに対して) 場合，このルーチンはエラー状態 SDA$_NOCOLLECT を返します。

## 返される状態値

| | |
|---|---|
| SDA$_SUCCESS | ファイル識別子は正常に変換されました。 |
| SDA$_COLLECT | システム，指定されたディスク，またはファイル識別子に対してコレクションがありません。 |
| その他 | LIB$FID_TO_NAME を呼び出したときにエラーが発生しました。 |

# SDA$GET_FLAGS

SDA がどのように使用されているかを示す環境フラグを取得します。

## フォーマット

int sda$get_flags *(SDA_FLAGS *flagaddr);*

## 引数

flagaddr

OpenVMS **用法** address
**データ型** SDA_FLAGS **構造体**
**アクセス** **書き込み専用**
**受け渡し方** **参照渡し**

環境フラグを返す場所を指すアドレス。

## 説明

SDA には一連のフラグ・ビットがあり，現在のシステム，システム・ダンプ，プロセス・ダンプのどれを分析するために使用されているかなどを示します。一連のビットは SYS$LIBRARY:SYS$LIB_C.TLB 内の SDA_FLAGSDEF.H で定義されています。

## 返される状態値

なし

### 3.16.3 ANALYZE コマンドの修飾子

SDA ANALYZE コマンドの新しい/COLLECTION 修飾子は，ファイル識別子の変換データまたはアンワインド・データが別のファイルにあることを SDA に指示します。コマンド文字列でこの修飾子を指定する場合，その前に/CRASH_DUMP 修飾子も指定する必要があります。次の形式を使用します。

```
/CRASH_DUMP/COLLECTION = コレクション・ファイル名
```

SDA は，ダンプ・ファイルを分析する際に，コレクションがファイル識別子変換データ (OpenVMS Alpha および OpenVMS for Integrity Servers の両方) およびアンワインド・データ (OpenVMS for Integrity Servers のみ) で構成されているかどうかの追加情報を提供します。通常，このデータは，ダンプ・ファイルを SDA COPY/COLLECT コマンドでコピーするときに保存されますが， COLLECT/SAVE コマンドを使用して個別のファイルに保存することもできます。

デフォルトでは， COLLECT/SAVE は，ダンプ・ファイルと同じ名前で同じディレクトリに.COLLECT ファイルを作成します。以降の ANALYZE/CRASH_DUMP コマンドは自動的にこのファイルを使用します。コレクション・ファイルが別の場所にある場合や，以前ダンプ・ファイルに追加したコレクションが不完全な場合は (たとえば， SDA COPY の時点でディスクがマウントされていなかった場合)，/COLLECTION 修飾子を使用して別のコレクション・ファイルを指定することができます。

コレクション・ファイル名の少なくとも 1 つのフィールドを指定する必要があります。他のフィールドのデフォルトは，ファイル名はダンプ・ファイルと同じで最も高い世代番号になり，場所はダンプ・ファイルと同じで，ファイル・タイプは.COLLECT になります。

### 3.16.4 DUMP コマンドの修飾子

SDA DUMP コマンドには以下の修飾子が新たに追加されています。

- /BYTE— 各データ項目をバイトとして出力します。
- /NOSUPPRESS— データを 16 進形式で表示する際に，先頭のゼロを抑制しないことを SDA に指示します。
- /WORD— 各データ項目を WORD として出力します。

### 3.16.5 SEARCH コマンドの修飾子

SDA SEARCH コマンドには，新たに/IGNORE_CASE 修飾子が追加されています。この修飾子は，文字列を検索する際に，英字の大文字と小文字を区別しないことを SDA に指示します。デフォルトでは，完全一致で検索します。この修飾子は，値の検索では無視されます。

### 3.16.6 SHOW CLUSTER コマンドの新しい修飾子

SDA SHOW CLUSTER コマンドでは，新たに/CIRCUIT=pb-addr修飾子が追加されています。この修飾子を指定すると，特定のパスの OpenVMS Cluster システム情報だけが表示されます。ここで，pb-addrはそのパス・ブロックのアドレスです。この修飾子は，修飾子/ADDRESS=n，/CSID=csid，および/NODE=nameとともに指定することはできません。/CIRCUIT=pb-addr修飾子を指定すると， SHOW CLUSTER コマンドは，指定したパス・ブロックから得た情報だけを表示します。

### 3.16.7 SHOW CRASH の修飾子

SDA SHOW CRASH コマンドには，新たに/ALL 修飾子が追加されています。この修飾子を指定すると，すべての CPU の例外データが表示されます。デフォルトでは，レジスタ (Alpha の場合) または例外フレームの内容 (Integrity サーバの場合) は，バグチェック CPUEXIT または DBGCPUEXIT を持つ CPU では表示から省略されます。

### 3.16.8 SHOW DUMP コマンドの修飾子

以下の修飾子が新たに SHOW DUMP コマンドに追加されています。

- /COLLECTION[= { ALL | n }]

  COPY/COLLECT を使用してダンプのコピーに追加されているか， COLLECT /SAVE を使用して個別のコレクション・ファイルに書き出されているファイル識別子やアンワインド・データ・コレクションの内容を表示します。デフォルトでは，コレクションの要約が表示されます。単独のエントリまたはすべてのエントリの詳細を表示するように指定できます。nは，コレクション・エントリの開始ブロック番号で，コレクションの要約に表示されます。

- /FILE= { COLLECTION | DUMP }

  個別のコレクション・ファイルを使用している場合，/FILE 修飾子は，どちらのファイルに SHOW DUMP を適用するかを示します。デフォルトでは，コマンド SHOW DUMP/SUMMARY， SHOW DUMP/HEADER， SHOW DUMP /COLLECTION，および SHOW DUMP/ALL は，両方のファイルに適用されます。デフォルトでは， SHOW DUMP/BLOCK はダンプ・ファイルに適用されます。その他すべての修飾子はダンプ・ファイルだけに適用することができます。

### 3.16.9 SDA SHOW PROCESS の修飾子

SDA SHOW PROCESS コマンドに，新たに/CHECK 修飾子が追加されました。この修飾子を指定すると，すべての空きプロセス・プール・パケットの POOLCHECK スタイルの破壊がチェックされます。チェックでは，システムが POOLCHECK のクラッシュ・ダンプを生成するときとまったく同じ方法が使われます。

### 3.16.10 SHOW RESOURCES/STATUS コマンドに追加されたキーワード

以下のキーワードが新たに SHOW RESOURCES/STATUS コマンドに追加されています。

- RM_FORCE— 強制的にツリーを移動する。
- RM_FREEZE— このノードのリソース・ツリーを凍結する。
- RM_INTEREST— マスタに関心がないため，再マスタリングを行う。
- XVAL_VALID— 最後の値ブロックは長いブロックだった。

### 3.16.11 SHOW UNWIND の修飾子

SDA SHOW UNWIND コマンドには，新たに修飾子/IMAGE=nameが追加されています。この修飾子を指定すると，指定されたシステム・イメージ (ワイルドカードが使用可能) のすべてのアンワインド記述子の詳細が表示されます。

## 3.17 システム・パラメータ

OpenVMS Version 8.3 では，いくつかのシステム・パラメータが追加されています。次の表に，これらの新しいパラメータの簡単な説明を示します。 (パラメータの詳細な説明は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』を参照してください。)

| パラメータ | 説明 |
|---|---|
| EXECSTACKPAGES | (Alpha および I64) EXECSTACKPAGES は， RMS の各実行スタックに対して割り当てるページの数を制御します。 |
| GB_CACHEALLMAX | (Alpha および I64) RMS グローバル・バッファの DEFAULT オプションが有効な状態でファイルが RMS に接続すると，キャッシュされるブロックの数は， GB_CACHEALLMAX パラメータの最大，またはファイルの割合のうち，グローバル・カウントが大きくなる方になります。 |
| GB_DEFPERCENT | (Alpha および I64) RMS グローバル・バッファの“DEFAULT”オプションが有効な状態でファイルを RMS に接続すると，ファイルの割合 (GB_DEFPERCENT) と GB_CACHEALLMAX ブロックのうち，グローバル・バッファ・カウントが大きくなる方までキャッシュされます。 |
| IO_PRCPU_BITMAP | (Alpha および I64) このパラメータは，最大 1024 個の CPU を表すビットマップを構成します。このビットマップ中に設定されるビットは，ファスト・パスの優先 CPU として使用可能な CPU を表します。 IO_PRCPU_BITMAP の省略時の設定では，すべてのビットが設定されます。(CPU 0 から CPU 1023 はすべて，ファスト・パスのポート割り当てが有効です。)<br>プライマリ CPU を優先 CPU として使用したくない場合は， IO_PRCPU_BITMAP のビットをオフにします。これにより，プライマリ CPU は，ファスト・パス IO 操作以外の操作用に予約されます。 |

| パラメータ | 説明 |
|---|---|
| LOCKRMWT | 0 ～ 10 の値を設定でき，省略時の設定は 5 です。再マスタリングの要否は，マスタとリモート・ノードの間のロック再マスタリングの重みの差に基づいて判断されます。 LOCKRMWT は動的パラメータです。 |
| SCD_HARD_OFFLD | スケジューラ・ハード・オフロード・パラメータは， CPU ビットマスク・パラメータです。各ビットは CPU ID に対応します。ビットがオンの場合， OpenVMS のスケジューラは，この CPU のハード・アフィニティがプロセスに対して設定されていないかぎり，その CPU にプロセスをスケジューリングしません。プライマリ CPU に対応するビットは無視されます。 SCH_HARD_OFFLD は動的パラメータです。 |
| SCH_SOFT_OFFLD | スケジューラ・ソフト・オフロード・パラメータは， CPU ビットマスク・パラメータです。各ビットは CPU ID に対応します。ビットがオンの場合， OpenVMS のスケジューラは，この CPU へのプロセスのスケジューリングを避けようとします。しかし，他に空き CPU がない場合は，この CPU にプロセスがスケジューリングされます。 SCH_SOFT_OFFLD は動的パラメータです。 |
| SCHED_FLAG | この特別なパラメータは弊社によって使用され，変更される可能性があります。弊社が推奨しないかぎり，このパラメータは変更しないでください。 |
| SMP_CPU_BITMAP | (Alpha および I64) このパラメータは，対応するビットが最大 1024 個の CPU を表すビットマップです。このビットマップの各ビットがオンの場合，インスタンスのブート時に，対応する CPU が OpenVMS のシンメトリック・マルチプロセシング環境のアクティブ・セットに自動的に参加しようとすることを示します。 |
| VCC_PAGESIZE | (Alpha および I64) VCC_PAGESIZE は，弊社での使用のために予約されている特別なパラメータです。 Extended File Cache の将来のバージョンでこのパラメータを使用する予定です。 |
| VCC_RSVD | (Alpha および I64) VCC_RSVD は，弊社での使用のために予約されている特別なパラメータです。 Extended File Cache の将来のバージョンでこのパラメータを使用する予定です。 |

## 3.18 システム・サービス・ロギングの機能拡張

OpenVMS Version 8.3 では，システム・サービス・ロギング (SSLOG) 機構が以下のように拡張されています。

- システム・サービス要求をログに記録する際に，サービスの要求元の CPU，カーネル・スレッド，および POSIX スレッド ID が記録されるようになりました。

  ANALYZE/SSLOG ユーティリティでは，各エントリの他の内容とともに，この新しい情報が表示されるようになりました。

  /SELECT の新しい値を指定することで，これらの特性に基づいてエントリを選択して表示することができます。

| 値 | 説明 |
|---|---|
| CPU | CPU ID |
| KTID | カーネル・スレッド ID |

| 値 | 説明 |
|---|---|
| TID | POSIX スレッド ID |

- システム・サービス要求を，プロセスのバッファにだけ記録し，ファイルには記録しないように要求することができます。ただし，これは非常に限定された使い方であり，ログに記録された情報にアクセスするためには，メモリ中またはクラッシュ・ダンプ内のロギング・バッファを見つけて手動で整形する必要があるため，お勧めできません。

  このロギング方法は， SET PROCESS/SSLOG 修飾子の FLAGS の新しい値で指定します。

  ```
  SET PROCESS/SSLOG=(STATE=ON[, FLAGS=[NO]FILE])
  ```

  このフラグの省略時の値は FILE です。

- システム・サービス・ロギングが有効な状態でプロセスが作成されることがあります。これは，ログイングを有効にしたプロセスがサブプロセスを作成した場合に起こります。親のロギング特性が子に引き継がれるためです。

  また，新しい$RUN コマンド修飾子を使用して，ロギングを有効にしたプロセスを明示的に作成することができます。 RUN コマンドの構文は次のとおりです。

  ```
  RUN /SSLOG_ENABLE=([COUNT=x][,SIZE=y]
     [,FLAGS=([NO]ARG,[NO]FILE))]
  ```

  また，$CREPRC サービスに対して次の新しいパラメータを指定して依頼することもできます。

  – 引数stsflagのフラグ PRC$M_SSLOG_ENABLE を設定すると，新しいプロセスでシステム・サービス・ロギングを有効にするように要求します。

  – ロギング特性は，項目リスト・エントリ・タイプ PRC$C_SSLOG_FLAGS, PRC$C_SSLOG_BUFSIZE，および PRC$C_SSLOG_BUFCNT を通じて指定することができます。

  プロセスの作成方法にかかわらず，プロセスの最初のイメージが完全に起動されるまではロギングが開始されません。

システム・サービス・ロギングの詳しい説明については，『HP OpenVMS System Analysis Tools Manual』を参照してください。

## 3.19 LDAP 認証のための SYS$ACM 対応のイメージ LOGINOUT.EXE および SETP0.EXE

**重要**

**本項で説明するイメージは，「実用前の段階」のイメージであり，実運用での使用には適していません。さらなる「実用品質」の厳格なテストと**

**認定が完了したら，保守アップデート** (ECO) **が行われ，** SYS$ACM **対応のイメージと**setp0**イメージの運用目的での展開が可能となります。**

本リリースでは， SYS$ACM システム・サービスを使用してユーザ認証とパスワードの変更を行うイメージ LOGINOUT.EXE および SETP0.EXE (SET PASSWORD) がオプションで提供されています。

これらのイメージを使用すると，ログインとパスワードの変更要求は SYS$ACM サービスに送られ， ACME_SERVER プロセスの認証エージェントで処理されます。

VMS の認証エージェントは，デフォルトで標準の VMS ログイン要求とパスワード変更要求を処理するように構成されています。また， LDAP Version 3 ディレクトリ・サーバを使用してログイン要求とパスワード変更要求を処理する LDAP 認証エージェントをインストールすることもできます。

詳細は， SYS$HELP:ACME_DEV_README.TXT ファイルを参照してください。

## 3.20 タイム・ゾーンの追加

OpenVMS Version 8.3 では， tzdata2006b という名前のタイム・ゾーン公開データベースに基づき， 544 個のタイム・ゾーンが提供されています。 OpenVMS Version 8.3 では，以下の 5 つのタイム・ゾーンが新たに追加されました。

Australia/Currie
America/Coral_Harbour
America/Indiana/Vincennes
America/Indiana/Petersburg
America/Moncton

Version 8.2–1 では，以下の 12 のタイム・ゾーンが追加されましたが，文書化されていませんでした。

America/Argentina/Buenos_Aires
America/Argentina/Catamarca
America/Argentina/Comodrivadavia
America/Argentina/Cordoba
America/Argentina/Jujuy
America/Argentina/La_Rioja
America/Argentina/Mendoza
America/Argentina/Rio_Gallegos
America/Argentina/San_Juan
America/Argentina/Tucuman

America/Argentina/Ushuaia
Europe/Mariehamn

これらの新しいタイム・ゾーンは，『OpenVMS システム管理者マニュアル』の次回の更新時に付録に追加されます。

以下のタイム・ゾーンは削除されました。

- SystemV/AST4ADT
- SystemV/EST5EDT
- SystemV/CST6CDT
- SystemV/MST7MDT
- SystemV/PST8PDT
- SystemV/YST9YDT
- SystemV/AST4
- SystemV/EST5
- SystemV/CST6
- SystemV/MST7
- SystemV/PST8
- SystemV/YST9
- SystemV/HST10

**注意**

**米国では，「**2005 **年エネルギー政策法** (Energy Policy Act in 2005)**」が通過し** 2007 **年** 3 **月から施行されます。これにより，夏時間** (DST) **は** 3 **月の第** 2 **日曜日から始まります (現在は** 4 **月の第** 1 **日曜日です**)。 DST **は，** 11 **月の第** 1 **日曜日に終わります (現在は** 10 **月の最後の日曜日です**)。 OpenVMS Version 8.3 **では，最新のタイム・ゾーン規則が導入されています。**

OpenVMS Versions 7.3–2**，** 8.2**，および** 8.2–1 **用のパッチ・キットが，** OpenVMS Alpha Version 8.3 Operating System **の** CD **および** OpenVMS for Integrity Servers Version 8.3 **の** DVD **で提供されています。**

## 3.21 OpenVMS での仮想 LAN (VLAN) のサポート

仮想 LAN (VLAN) は， LAN のブロードキャスト・ドメインをより小さな範囲にセグメント化するための仕組みです。 IEEE 802.1Q 規格では， VLAN の運用と動作が定義されています。 OpenVMS の実装では，一部の OpenVMS LAN ドライバで IEEE 802.1Q のサポートが追加されており， OpenVMS は，単一の LAN アダプタを使用して， VLAN のタグ付きパケットを LAN アプリケーションにルーティングできるようになりました。

VLAN を使用して以下のことができます。

- ネットワーク・セキュリティやトラフィックの封じ込めを目的とした，ネットワーク上でのセグメント限定 LAN トラフィック
- VLAN で分離されたネットワークによるアドレス管理の単純化

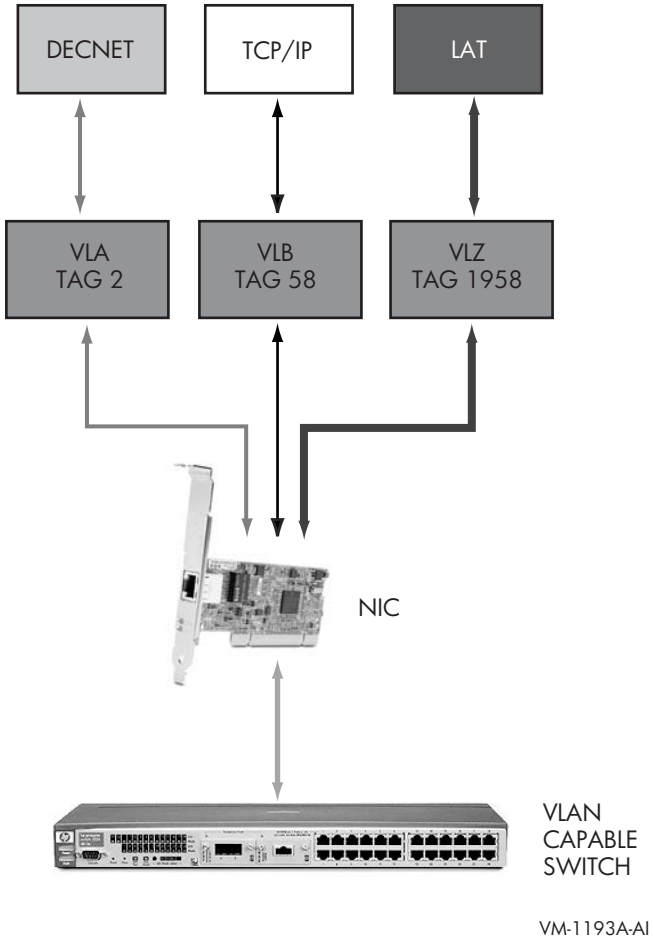
### VLAN の設計

OpenVMS では， VLAN は LAN アプリケーションに対して仮想 LAN デバイスを提供します。仮想 LAN デバイスは，単一の IEEE 802.1Q タグを物理 LAN デバイス上の通信に関連付けます。仮想 LAN デバイスを使用すると，物理 LAN デバイス上で任意の LAN アプリケーション (たとえば， SCA， DECnet， TCP/IP， LAT) を実行することができ，図 3–1 に示すホスト間通信が可能です。

**注意**

**DECnet-Plus および DECnet Phase IV は，VLAN デバイス上で動作するように設定できます。**

**図 3–1 仮想 LAN**




OpenVMS の VLAN は，新しいドライバ SYS$VLANDRIVER.EXE を通じて実装されています。このドライバは，仮想 LAN デバイスを提供します。また，既存の LAN ドライバも， VLAN タグを処理するように更新されています。 LANCP.EXE および LANACP.EXE は更新され， VLAN デバイスの作成と非アクティブ化，状態情報と設定情報の表示ができるようになりました。

OpenVMS の VLAN サブシステムは，特に性能に注意して設計されています。そのため， VLAN のサポートを使用することによる性能の低下はごくわずかです。

VLAN デバイスを設定する場合には， VLAN デバイスが物理 LAN デバイスと同じロック機構を共有する点に注意してください。たとえば， VLAN デバイスと基になっている物理 LAN デバイス上で OpenVMS クラスタ・プロトコルを同時に実行してもメリットはなく，実際には性能が低下します。

### 3.21.1 VLAN **サポートの詳細**

サポートされているすべての Gigabit および 10-Gb (I64 のみ) の LAN デバイスは，Alpha システムおよび I64 システム上で， VLAN トラフィックを処理することができます。

VLAN 関連のサポートに関するその他の詳細事項を以下に示します。

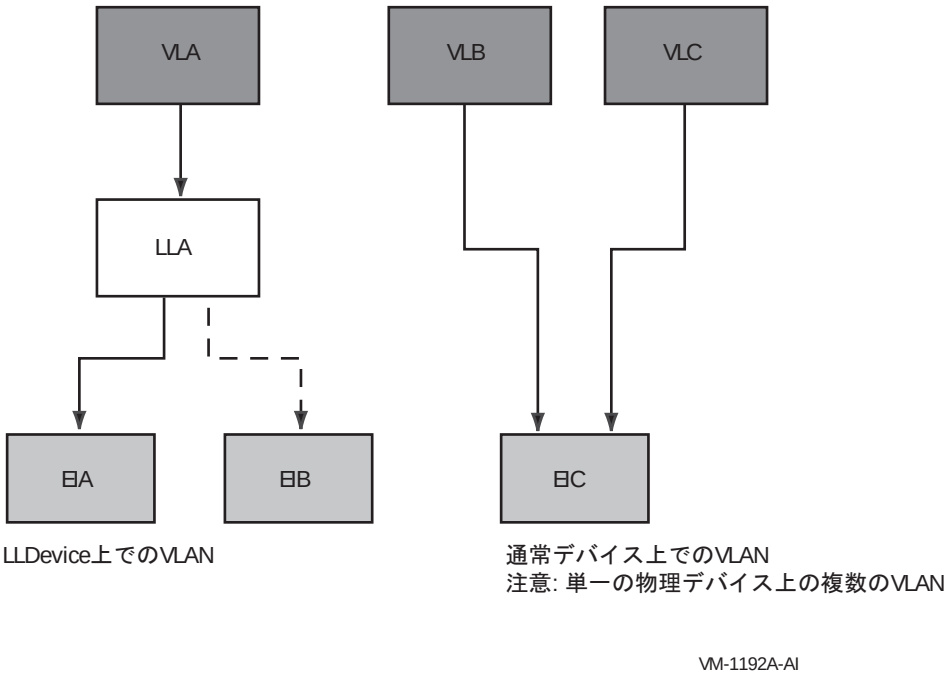
- スイッチのサポート

  VLAN 構成では，スイッチに対する唯一の要件は， IEEE 802.1Q 仕様に準拠していることです。スイッチに対する VLAN ユーザ・インタフェースの標準はありません。そのため，特に異なるスイッチにまたがった VLAN を設定する場合など，スイッチを設定する際には十分注意してください。

- LAN フェールオーバのサポート

  図 3–2 に， LAN フェールオーバのサポートを示します。

**図** 3–2 LAN **フェールオーバのサポート**




  LAN フェールオーバ・セットのすべてのメンバが VLAN 対応デバイスの場合は，そのセットをソースとして使用する VLAN を作成することができます。しかし，VLAN デバイスを使用してフェールオーバ・セットを構築することはできません。

- サポートされている機能

VLAN デバイスは，ファスト・パス，オートネゴシエーション，ジャンボ・フレーム設定などの機能を，基になっている物理 LAN デバイスから継承します。機能を変更する必要がある場合は，基となる物理 LAN デバイスを変更する必要があります。

- 制限事項

  VLAN デバイス上でのサテライト・ブートはサポートされていません。OpenVMS LAN ブート・ドライバには， VLAN のサポートが含まれていないため， VLAN デバイスを使用して OpenVMS システムをブートすることはできません。

  現時点では， OpenVMS で VLAN デバイスの自動設定はサポートされていません。 LANCP コマンドを使用して明示的に VLAN デバイスを作成する必要があります。

### 3.21.2 システム上の VLAN の管理

VLAN デバイスを作成する前に，それをホストする VLAN 対応の物理 LAN デバイスが VLAN 対応のスイッチに接続されていることを確認してください。また，選択したスイッチ・ポートが VLAN タグ付きのトラフィックを処理できるように構成されていることも確認してください。

以降の項では，その他の VLAN 管理の詳細について説明します。

#### 3.21.2.1 スイッチ・ポートのプローブ

VLAN デバイスの管理を容易にするために， OpenVMS の LAN には IEEE 802.1Q 管理機能の限定的なサポートが含まれています。 LANCP 修飾子は，スイッチのポートをプローブし， VLAN 設定情報の一覧を表示するのに役立ちます。新しいコマンドは次のとおりです。

```
LANCP> SHOW DEVICE PHYSICAL-LAN-DEVICE/VLAN
```

コマンドを入力すると， LANCP は IEEE 802.1Q GVRP (Generic Attribute Registration Protocol) VLAN Registration Protocol パケットをリッスンし，以下の内容を表示します。

- スイッチ・ポート上に設定された VLAN タグ
- 物理 LAN デバイス上に設定されている VLAN デバイス

以下に例を示します。

```
LANCP> SHOW DEVICE LLB /VLAN

Listening for VLAN configuration on LLB0 ......
   VLAN tag 190 configured as VLB
   VLAN tag 206 configured as VLJ
   VLAN tag 207 not configured
```

このコマンドでは，スイッチ・ポート上で GVRP 機能が有効になっている場合にのみ VLAN 情報が表示されます。

### 3.21.2.2 VLAN デバイスの作成

VLAN デバイスを作成するには，次の形式で LANCP コマンドを入力します。

```
LANCP> SET DEVICE VLc/VLAN_DEVICE=PHYSICAL-LAN-DEVICE/
TAG=value
```

各項目の内容は以下のとおりです。

- VLcは，仮想 LAN デバイスの名前です (cはコントローラ文字 a ～ z)。
- PHYSICAL-LAN-DEVICE は， VLAN をホストする LAN デバイスです。
- valueは， IEEE 802.1Q タグです (有効な範囲は 1 ～ 4095)。

次に例を示します。

```
LANCP> SET DEVICE VLA/VLAN=EIB/TAG=42
```

このコマンドは，物理 LAN デバイスが存在しないか，物理 LAN デバイスが VLAN 対応でないか， VLAN タグが正しくないと失敗します。

**説明文と LAN デバイスの関連付け**

このバージョンの OpenVMS では，説明文を LAN デバイスに関連付ける機能も追加されています。それには， LANCP SET コマンドまたは DEFINE DEVICE コマンドで修飾子/DESCRIPTION=<quoted-string>を指定し，状況説明を追加します。たとえば， VLAN デバイスを“Finance VLAN”の一部として区別するには，次のコマンドを入力します。

```
LANCP> SET DEVICE VLA/DESCRIPTION="Finance VLAN"
```

### 3.21.2.3 仮想 LAN デバイスの非アクティブ化

VLAN デバイスを非アクティブ化するには，次のコマンド形式を使用します。

```
LANCP> SET DEVICE VLc/NOVLAN
```

デバイスが使用中の場合 (つまり，他のアプリケーションがそのデバイスを使用している場合) は，このコマンドは失敗します。

### 3.21.2.4 VLAN デバイス情報の表示

VLAN デバイスに関する情報を表示するには， LANCP コマンドの SHOW DEVICE および SHOW CONFIGURATION を使用します。以下に例を示します。

```
LANCP> SHOW DEVICE VLK/CHARACTERISTICS
```

```
Device Characteristics VLK0:
                    Value   Characteristic
                    _____   ______________
                    ...
                    "206"   VLAN 802.1Q tag
                      "1"   VLAN device flags
    "Procurve 2315 P15"   VLAN description
                 Link Up   Link state
```

```
Device Parent Medium/User Version Link Speed Duplex Size     MAC Address    Current Address   Type
------ ------ ----------- ------- ---- ----- ------ ---- ----------------- ----------------- ----
EWA0          Ethernet    X-51    Up   1000  Full   1500 00-D0-59-61-72-F3 AA-00-04-00-1B-4D UTP DEGXA-TA
EWB0          Ethernet    X-51    Up    100  Full   1500 00-D0-59-61-72-D8 00-D0-59-61-72-D8 UTP DEGXA-TA
EWC0          Ethernet    X-59    Up   1000  Full   1500 00-60-CF-21-71-9C AA-00-00-21-71-9C UTP DEGPA-TA
EWD0          Ethernet    X-59    Up   1000  Full   1500 00-60-CF-20-9A-C6 00-60-CF-20-9A-C6 UTP DEGPA-TA
EIA0          Ethernet    X-16    Up   1000  Full   1500 00-12-79-9E-20-AE AA-00-04-00-1B-4D UTP AB352A
EIB0          Ethernet    X-16    Up   1000  Full   1500 00-12-79-9E-20-AF 00-12-79-9E-20-AF UTP AB352A
LLB0          Ethernet    X-19    Up   1000  Full   1500 AA-00-00-21-71-9C AA-00-00-21-71-9C DEGPA-TA
VLB0          Ethernet    X-BA1   Up   1000  Full   1500 AA-00-00-21-71-9C AA-00-00-21-71-9C LLB
VLC0          Ethernet    X-BA1   Up   1000  Full   1500 00-12-79-9E-20-AF 00-12-79-9E-20-AF UTP EIB
VLD0          Ethernet    X-BA1   Down  100  Full   1500 00-00-00-00-00-00 00-00-00-00-00-00
VLJ0          Ethernet    X-BA1   Up   1000  Full   1500 AA-00-00-21-71-9C AA-00-00-21-71-9C LLB
VLK0          Ethernet    X-BA1   Up   1000  Full   1500 00-12-79-9E-20-AE AA-00-04-00-1B-4D UTP EIA
```

### 3.21.3 VLAN のトラブルシューティング

ほとんどの VLAN の問題は設定に関するものです。 VLAN の問題をトラブルシューティングする際に確認すべき事柄の一覧を以下に示します。

1. OpenVMS では，すべての LAN デバイスが VLAN 対応になっているわけではありません。 VLAN 対応でないデバイス上で VLAN を作成しようとすると，LANCP がエラー・メッセージを表示します。

   LAN デバイスが VLAN 対応であることを確認するには， SDA を使用してデバイス特性を確認します。次のコマンドを入力します。

   ```
   $ ANALYZE/SYSTEM

   SDA> SHOW LAN/DEVICE=physical-device-name
   または
   SDA> LAN DEVICE/DEVICE=physical-device-name
   ```

   文字列“VLAN”で示される，デバイス特性の VLAN ビット 4 が設定されている必要があります。

2. LAN デバイスが接続されているスイッチ・ポート上で， VLAN 機能が有効になっていることと，正しい VLAN タグが設定されていることを確認します。スイッチで GVRP が有効になっている場合は，次の LANCP コマンドを入力することで VLAN タグが有効になっていることを確認できます。

   ```
   LANCP> SHOW DEVICE physical-device-name/VLAN
   ```

このコマンドでは，スイッチ・ポートに設定されている VLAN タグが表示されます。次に，表示されたタグが， VLAN デバイスを作成する時に使用したものと同じであることを確認します。

3. VLAN デバイスが正しく設定されていることを確認します。次のコマンドを入力して， VLAN ドライバが保持している特性とステータスを表示します。

```
LANCP> SHOW DEVICE vlan-device-name/INTERNAL_COUNTERS
```

以下に例を示します。

```
LANCP> SHOW DEVICE VLC/INTERNAL_COUNTERS

Device Internal Counters VLC0:
                Value  Counter
                _____  _______
                       --- Internal Driver Counters ---
            "     EIB"  Device name
              00000001  Device Flag 1 <online>
                   190  VLAN Tag ID
              86514000  Physical LSB
                 11834  Failure status
      FFFFFFFF 805E28CC  Failure PC
```

以下の内容を確認します。

a. デバイス名とタグが， VLAN デバイスを作成した時に指定したものと同じであることを確認します。

b. Device Flag 1 フィールドの“online”ビットがオンになっていることを確認します。オンになっていない場合は，障害ステータスで詳しい情報が得られることがあります。

c. Physical LSB フィールドは，物理 LAN デバイスの LSB (LAN Station Block) 構造体のアドレスです。このデバイスの特性とステータスを表示するには，次のコマンドを入力します。

```
$ ANALYZE/SYSTEM
SDA> LAN DEVICE/ADDRESS=physical LSB address
```

OpenVMS の VLAN サポートについての詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』を参照してください。

## 3.22 Volume Shadowing for OpenVMS

OpenVMS Version 8.3 では， HP Volume Shadowing for OpenVMS に以下の機能が追加されています。

- ボリューム処理時の自動的なビットマップの作成
- SET SHADOW の新しい修飾子/RESET

### 3.22.1 ボリューム処理時の自動的なビットマップの作成

ボリューム処理時の自動的なビットマップの作成とは，1 台以上のシャドウ・セット・メンバとの接続が失われ，シャドウ・メンバのタイムアウト時間内に回復しなかった場合に，既存の HBMM ビットマップがミニコピー・ビットマップとして機能できるようになることを意味します。

このような接続が失われると，シャドウ・セットのボリューム処理は一時停止されます。つまり，接続が回復するか，タイムアウト時間 (SHADOW_MBR_TMO の値で決まります) が満了するまで，書き込みと読み込みは一時的に停止されます。

タイムアウト時間内に接続が回復しないと，そのメンバはシャドウ・セットから除外され，残りのメンバに対する読み書き入出力が再開され，ビットマップによって書き込みが追跡されます。このビットマップの名前は HBMMxから rrsexに変更され，除外されたメンバ用のミニコピー・ビットマップとして機能します。

**注意**

**1 台または 2 台のメンバが除外され，すべてのメンバがシャドウ・セット中のメンバシップに復元された後も，** HBMM **ビットマップの機能は有効なままです。** HBMM **ビットマップの機能は，シャドウ・セットが 3 台のメンバで構成され，1 台のメンバが除外された場合にだけ役立ちます。**

除外されたシャドウ・セット・メンバのいずれかに対する接続が回復すると，シャドウ・セットに再度マウントすることができます。除外されたメンバのメタデータが既存のビットマップと一致する場合は，それがミニコピー操作で使用され，メンバがシャドウ・セットに復帰します。2 台目のシャドウ・セット・メンバも同時に除外された場合は，そのメンバもそのビットマップを使用することができます。メンバがシャドウ・セットに復帰した後，ビットマップの名前は HBMM ビットマップ名に戻ります。

1 台以上のメンバがシャドウ・セットから除外されている時間を最小限にする理由は以下のとおりです。

- シャドウ・セットのメンバが減っている間，データの可用性が危険な状態にあります。
- シャドウ・セット・メンバが除外されても，残りのメンバに対する読み書きは継続されます。除外されたメンバが復帰する前に多数の書き込みがあると，そのメンバをシャドウ・セットに復帰させるのに時間がかかります。これは，ディザスタ・トレラント (DT) 構成で特に重要です。

ボリューム処理時の自動的なビットマップ作成機能が追加されるまでは，接続が回復した後で除外されたメンバをシャドウ・セットに復帰させる作業には時間がかかりました。除外されたメンバは，フル・コピーを行わないと復帰させることができません

でした。ビットマップをミニコピー操作に利用できるようになったことにより，フル・コピー操作に比べて時間が大幅に短縮されました。

ボリューム処理時の自動的なビットマップ作成を有効にするには，そのシャドウ・セットの HBMM ポリシーを設定し，ポリシーに新しい MULTIUSE キーワードを追加します。詳細は，『HP OpenVMS Version 8.2 新機能説明書』の HBMM の章を参照してください。

### 3.22.2 SET SHADOW の新しい/RESET 修飾子

本リリースでは， SET SHADOW コマンドに/RESET 修飾子が追加されています。SET SHADOW/RESET=COUNTERS は，シャドウ・セットごとに保持されているシャドウイング専用のカウンタをリセットします。

リセットして 0 にされるカウンタは以下のとおりです。

HBMM Reset Count
Copy Hotblocks
Copy Collisions
SCP Merge Repair Cnt
APP Merge Repair Cnt

これらのカウンタの現在の設定を表示するには， SHOW SHADOW コマンドを使用します。

HBMM Reset Count は， RESET_THRESHOLD に達した回数を表します。RESET_THRESHOLD は，ビットマップをどの程度の頻度で作成するかを決める設定です。 HBMM Reset Count をクリアできることから，システム管理者は，しきい値リセットの頻度をより詳細に評価することができます。

SET SHADOW/RESET についての詳細は，『OpenVMS DCL ディクショナリ: N–Z』および DCL のヘルプを参照してください。

# 4

# OpenVMS 上での光媒体のマスタリング

この章では， OpenVMS 上での CD 媒体や DVD 媒体の作成 (マスタリング) について説明します。

CD 媒体や DVD 媒体の作成では，以下の作業を行います。

1. ステージング領域でのディスク・ボリューム構造の作成
2. ボリューム構造へのファイルの追加
3. 光媒体へのマスタのコピー

OpenVMS では，ステージング領域として論理ディスク (LD) デバイスを使用し，INITIALIZE， MOUNT， COPY，および BACKUP などの DCL コマンドを使用して，ステージング領域にディスク・ボリュームを作成してデータを格納します。次に， COPY/RECORDABLE_MEDIA コマンドを使用してディスク・ボリュームの内容をコピーします。

## 4.1 LD，CD，および DVD デバイスの概要

ここでは， OpenVMS 上での光媒体のマスタリングに関連する概念について説明します。

### 4.1.1 論理ディスク・デバイス

論理ディスク (LD) デバイスは，光媒体に書き込むデータのマスタ・コピーをステージングするための仕組みとして使用されます。 LD ディスク・デバイスを使用して記録操作の元データを作成した後， COPY/RECORDABLE_MEDIA コマンドを実行してマスタを光媒体にコピーします。

LD ディスク・デバイスを作成して管理するには， LD ユーティリティを使用します。その後，標準的な OpenVMS DCL コマンドを使用して LD ディスク・デバイスを初期化，マウント，アクセスします。

LD ディスク・デバイスについての詳細は，『OpenVMS システム管理者マニュアル』を参照してください。

### 4.1.2 CD デバイスと DVD デバイス

光媒体デバイスでは，さまざまな記録形式が使用できます。一般に， OpenVMS では，使用するターゲット・デバイスでサポートされている形式の媒体を読み込むことができます。

OpenVMS では，以下の 4 つの形式の媒体に対して記録できます。

| 形式 | 説明 |
|---|---|
| CD-R | Compact Disc Recordable |
| CD-RW | Compact Disc Rewritable |
| DVD+R | Digital Versatile Disc Recordable |
| DVD+RW | Digital Versatile Disc Rewritable |

ターゲットとなる CD デバイスと DVD デバイスの特性と機能は，システム，記録デバイス，記録媒体に依存します。たとえば，ローカルのハードウェア構成やソフトウェア構成によって， CD の最大許容記録速度が， CD 記録デバイスがサポートしている速度よりも遅い速度に制限されることがあります。必要な入出力帯域幅を備えていない OpenVMS システムで CD を記録しようとすると，ターゲット CD デバイスのデータ・キャッシュでアンダフローが発生することも考えられます。このような操作を行うと，記録エラーとなって失敗し，記録媒体が無駄になります。

記録デバイスはさまざまな記録形式と媒体をサポートしています。逆に， OpenVMS や特定のデバイスでサポートされていない記録形式もあります。現在サポートされているデバイス・ハードウェアとそれに関係するプラットフォーム構成については，次の Web サイトを参照してください。

http://www.hp.com/go/server/

使用している I64 プラットフォームまたは Alpha プラットフォームを探し，そのプラットフォームのサポート・マトリックスを探してください。

## 4.2 データ・ディスクのマスタリングを行うための一般的な手順

光媒体のマスタリング (記録する，または焼くと言うこともあります) を行うための手順は以下のとおりです。

1. 次のコマンドを実行して， OpenVMS 論理ディスク (LD) を作成します。

```
$ @SYS$STARTUP:LD$STARTUP.COM
```

**注意**

**LD$STARTUP を実行するためには，特権 TMPMBX， NETMBX，および SYSLCK が必要です。この手順の後の方で使用する COPY/RECORDABLE_MEDIA コマンドは，必要な特権とともにインストールされています。**

このコマンドは，OpenVMS をブートするたびに一度だけ実行します。このコマンドを自動的に実行するようにするには，サイト固有のシステム・スタートアップ・プロシージャ SYS$MANAGER:SYSTARTUP_VMS.COM にこのコマンドを追加します。このようにすると，OpenVMS システムがブートするたびにコマンドが実行されます。

2. 媒体マスタのステージング領域として使用する論理ディスク (LD) を作成します。この LD ディスク・デバイスは，通常の物理ディスク・デバイスのように使用できますが，サイズの指定やサイズ変更が容易に行えるという柔軟性を備えています。また，必要に応じてデバイスの作成や削除を行うことができます。

   LD ディスク・デバイスへの接続や管理を行うための LD ドライバは，バックアップ・ストレージ・ファイルを使用します。これにより，リブートしても LD ディスク・デバイスの内容が保持されます。LD ディスク・デバイスの容量 (および対応するバックアップ・ファイルのサイズ) は，格納するファイルやボリューム構造データのサイズに等しいか，それよりも大きい必要があります。また，LD ディスク・デバイスの容量は，ターゲットとなる光媒体の容量に等しいか，それよりも小さい必要があります。マスタの内容は，ターゲット媒体に収まる必要があります。

   通常，大まかな最大容量は以下のとおりです。

| 媒体 | 最大ブロック数 | 容量 |
|---|---|---|
| CD-R | 1,200,000 ブロック | 600 MB |
| CD-RW | 1,400,000 ブロック | 700 MB |
| 単層 DVD+R | 9,180,416 ブロック | 4.6 GB |
| 単層 DVD+RW | 9,180,416 ブロック | 4.6 GB |

   作成できる LD のサイズは，ターゲット媒体の最大サイズまでです。光媒体では，4 ブロック (2048 バイト) のセクタ・サイズを使用するため，必ず 4 ブロックの整数倍の容量を持つ LD ディスク・デバイスを作成して使用する必要があります。推奨される容量は，16 ブロックの整数倍の容量です。

3. LD マスタを作成するには，まずマスタの LD バッキング・ストレージ・ファイルを作成します。コマンドの例を次に示します

   ```
   $ LD CREATE /size=9180416 filespec.ISO
   ```

   この LD ストレージ・ファイルの作成は，一度だけ行います。

4. LD ストレージ・ファイルを LD 論理ディスクに接続します。コマンドの例を次に示します

   ```
   $ LD CONNECT filespec.ISO LDA1:
   ```

OpenVMS システムをブートするたびに LD ディスク・デバイスに再接続する必要があります。 LD CONNECT コマンドを，サイト固有のシステム・スタートアップ・プロシージャ SYSTARTUP_VMS.COM に追加すれば，システムがブートするたびに自動的にコマンドが実行されます。

5. マスタを使用するための準備を行います。

   先に進む前に， LD マスタの内容を完全に消去することをお勧めします。これにより，ローカル・システムに関する機密情報が意図せずに公開されてしまうのを防ぐことができます。ディスク・マスタの内容を消去する方法には，さまざまなものがあります。たとえば， ODS-2 または ODS-5 のボリューム構造を作成している場合は， DCL コマンド INITIALIZE/ERASE を使用する方法があります。

   ターゲット媒体で OpenVMS ODS-2 または ODS-5 のボリューム構造を使用する場合は， DCL コマンド INITIALIZE を使用してボリューム構造を作成します。次に，標準の MOUNT コマンドを使用して，他の OpenVMS コマンドからマスタ・ディスク・ボリュームにアクセスできるようにします。

   コマンドの例を以下に示します。

   ```
   $ INITIALIZE LDA1: ボリューム・ラベル -
       /SYSTEM [/ERASE] [/...] -
       [/CLUSTER=n] [/STRUCTURE=n] [/...]
   $ MOUNT LDA1: ボリューム・ラベル
   ```

6. ボリューム構造が使用可能になったら，マスタにデータをコピーします。

   LD マスタにコピーするデータには，データ・ファイル，インストール・キット，実行可能イメージ，ツールなどのファイルが含まれます。 ODS-2 や ODS-5 のボリュームなどの標準的な物理ディスク形式と同様に， BACKUP， COPY， CREATE/DIRECTORY などの標準的な DCL コマンドとプロシージャを使用して， LD マスタの内容を作成することができます。

   ODS-2 または ODS-5 のボリューム構造を使用する場合は， OpenVMS セキュリティ識別子や ACL をマスタに格納しないでください。これらの情報はシステム固有であり，記録した媒体を他の OpenVMS システムでマウントまたはアクセスすると，アクセスが予期せずに許可されたり拒否されることがあります。

7. マスタが格納されている LD ディスク・デバイスに必要な内容をコピーし終えたら，次のコマンドを使用してデバイスをディスマウントします。

   ```
   $ DISMOUNT LDA1:
   ```

8. LD マスタの内容を光媒体に記録します。

   まず，適切な空の媒体を光媒体ディスク・ドライブに挿入します。その後，次のコマンドを入力します。

   ```
   $ COPY/RECORDABLE_MEDIA LDA1: DQA0: -
   _$ [/FORMAT][/BELL][/SPEED=speed][/VERIFY]
   ```

   このコマンドは， LDA1: マスタの内容をターゲット・デバイスにコピーします。

この例では，以下の点に注意してください。

- ターゲット・デバイスは DQA0: であり，書き換え可能な媒体が挿入されているものと想定しています。ターゲット・デバイス名は，お使いのハードウェアの構成によって変わります。
- /FORMAT 修飾子は，書き換え可能な媒体にのみ適用できます。この修飾子を指定すると，書き換え可能媒体が消去され，記録が可能な状態になります。
- /SPEED 修飾子を指定することで，デフォルトで設定される速度よりも記録速度を遅くすることができます。この修飾子が必要になるのは， CD や DVD の記録でバッファ・アンダランやデータ不足エラーとなる場合や， CD 形式の記録の際に，記録速度の遅い CD 媒体を使った場合です (つまり， CD 媒体の定格速度が， CD 記録デバイスの定格速度よりも遅い場合)。

  /SPEED を使用して， CD または DVD の記録速度を指定できます。指定できる最大速度は，使用している OpenVMS システムの入出力性能で決まります。この修飾子を使用して指定できる速度は，ターゲット・ドライブとターゲット記録媒体の定格最大記録速度によって制限されます。

  /SPEED は，速度を早めるのではなく，速度を遅くするための仕組みです。品質が低い媒体や，特に欠陥のある媒体を使用する場合など，問題があるときに速度を遅くする必要があります。

最大速度のエンコード方法には， CD 媒体と DVD 媒体で以下の違いがあります。

- CD 媒体では，媒体の製造時に最大速度が設定されているものの，最大速度がエンコードされていません。エンコードされた上限がないため，記録するときに規定の速度を超えてしまう可能性が高くなります。
- DVD 媒体では，最大速度がエンコードされています。記録速度は，媒体で規定されている上限を超えることはできません。

媒体にかかわらず，構成内のその他の制限によって，最大記録速度が低く抑えられる場合があります。この最大速度を超えると，記録操作は失敗します。

/BELL 修飾子を指定すると，操作の完了時にベルが鳴ります。

/VERIFY 修飾子を指定すると，記録操作が完了した後で，記録された媒体の内容が読み込まれ入力データと比較されます。

9. 光媒体を正常にマスタリングした後で，マスタリング用の LD 論理ディスクが必要でなくなった場合は，関連付けられているバックアップ・ストレージ・ファイルが専有していたディスク領域を解放することができます。次のコマンドを入力して，システムから LDA1: デバイスを切断し削除します。その後バックアップ・ストレージ・ファイルを削除します。

```
$ LD DISCONNECT LDA1:
$ DELETE filespec.ISO;*
```

## 4.3 例

**光媒体のマスタリングの例**

以下のコマンド・シーケンスは，LD ディスク LDA600: を使用して，CD 対応の記録デバイス上で 600 MB の CD-R 媒体を作成する方法を示しています。ストレージ・コピー・ファイルはデバイス DISK$SCRATCH: に格納されます。このシーケンスでは，空の CD-R ディスクがターゲット・ドライブ DQA0: に挿入されているものと想定しています。ディレクトリが作成され，ユーザの LOGIN.COM ファイルがそのディレクトリにコピーされます。記録が完了すると，記録済みの媒体には，[DATA]という OpenVMS ディレクトリが 1 つだけ含まれます。

```
$ @SYS$STARTUP:LD$STARTUP
$ LD CREATE DISK$SCRATCH:[000000]CD600.ISO/SIZE=1200000
$ LD CONNECT DISK$SCRATCH:[000000]CD600.ISO LDA600:
$ INITIALIZE/ERASE/SYSTEM LDA600: CDDATA
$ MOUNT/SYSTEM LDA600: CDDATA
$ CREATE/DIRECTORY LDA600:[DATA]
$ COPY SYS$LOGIN:LOGIN.COM LDA600:[DATA]
$ DISMOUNT LDA600:
$ COPY/RECORDABLE_MEDIA LDA600: DQA0:/VERIFY/BELL
$ LD DISCONNECT LDA600:
$ DELETE DISK$SCRATCH:[000000]CD600.ISO;
```

**DVD へのフォーマットとデータの記録の例**

次の例は，媒体をフォーマットし，ディスク・マスタ LDA600: に格納されたデータを DVD+RW ドライブとそれに挿入された DVD+RW 媒体に記録する例を示します。この媒体はデバイス DQA0: 上にあります。

```
$ COPY/RECORDABLE_MEDIA LDA600: DQA0:/FORMAT

HP OpenVMS CD-R/RW and DVD+R/RW Utility, V1.0-1 Copyright 1976,
2006 Hewlett-Packard Development Company, L.P.

Output device vendor: HL-DT-ST
Output device product name: DVD+RW GCA-4040N
Output device revision: 1.15
Commencing media format operation Formatting may require up to an hour
Output medium format: DVD+RW
Input data from: LDA600:
Writing data to: DQA0:
Input data size: 1200000 blocks

Starting operation at: 08:48:17

16 sectors written
```

```
30000 sectors written; estimated completion in 00:07:36; at 08:56:44 37000
sectors written; estimated completion in 00:07:37; at 08:56:57 46000
sectors written; estimated completion in 00:07:15; at 08:56:50 57000
sectors written; estimated completion in 00:06:43; at 08:56:34 71000
sectors written; estimated completion in 00:06:33; at 08:56:48 88000
sectors written; estimated completion in 00:05:56; at 08:56:39 110000
sectors written; estimated completion in 00:05:23; at 08:56:42 137000
sectors written; estimated completion in 00:04:37; at 08:56:41 171000
sectors written; estimated completion in 00:03:33; at 08:56:33 213000
sectors written; estimated completion in 00:02:23; at 08:56:32 266000
sectors written; estimated completion in 00:00:59; at 08:56:36 300000
sectors written; operation completed

Operation completed at: 08:56:32
Elapsed time for operation: 00:08:14
Synchronizing with output device cache
Processing completed
$
```

# 5
# プログラミング機能

この章では， HP OpenVMS の本バージョンにおける，アプリケーション・プログラマおよびシステム・プログラマに関係がある新機能について説明します。

## 5.1 C 実行時ライブラリの機能拡張

以降の項では， OpenVMS Version 8.3 での C 実行時ライブラリ (RTL: Run-Time Library) の機能拡張について説明します。拡張された機能としては， UNIX とのポータビリティの向上，標準への準拠，より柔軟なユーザ制御の機能選択があります。新しい C RTL 関数も追加されています。詳細は，『*C* **ランタイム・ライブラリ・リファレンス・マニュアル**』を参照してください。

### 5.1.1 シンボリック・リンクと POSIX 準拠のパス名のサポート

OpenVMS Version 8.3 以降では， Open Group 準拠のシンボリック・リンクのサポートと， POSIX 準拠のパス名のサポートが含まれています。その目的は， UNIX や Linux のアプリケーションを OpenVMS に移植したり， UNIX スタイルの開発環境を使用しているパートナや顧客を支援し，移植作業に関連するアプリケーションの開発コストと複雑さを削減することです。

このサポートがあっても， OpenVMS システム上での UNIX のファイルの 100%の互換性は保証されません。 UNIX や Linux のアプリケーションを OpenVMS に移植するためには，変更を行わなければならない場合もありえます。

シンボリック・リンクと POSIX のパス名の処理をサポートするために，以下の機能が OpenVMS で提供されています。

- 以下の Open Group 準拠のシンボリック・リンク関数が C の実行時ライブラリに追加されています。

  ```
  symlink
  readlink
  unlink
  realpath
  lchown
  lstat
  ```

- creat，open，delete，および remove などの既存の C RTL 関数は， Open Group のシンボリック・リンクの仕様に準拠した振る舞いをするようになりました。

- RMS では，C RTL で上記の関数の実装に対応します。SYS$OPEN，SYS$CREATE，SYS$PARSE，および SYS$SEARCH などの RMS ルーチンは，シンボリック・リンクをサポートするようになりました。
- OpenVMS でパスの解釈や検索の際にシンボリック・リンクが見つかると，その内容は POSIX パス名として解釈されます。POSIX パス名が OpenVMS でサポートされるようになり，C RTL と RMS インタフェースで使用可能になりました。
- 新しい機能論理名 DECC$POSIX_COMPLIANT_PATHNAMES が C RTL に追加されました。これは，アプリケーションが POSIX 準拠モードで動作することを示します。
- DCL コマンド CREATE/SYMLINK は，シンボリック・リンクを作成するために使用します。
- DCL コマンド SET ROOT は，システムの POSIX ルートを作成するために提供されています。
- 2 つの GNV ユーティリティ mnt および umnt が，マウント・ポイントを設定するために提供されています。
- 各種の DCL コマンドとユーティリティが，シンボリック・リンクを適切に処理するように変更されました。
- TCP/IP Services for OpenVMS Network File System (NFS) のクライアントとサーバが，ODS5 ボリューム上のシンボリック・リンクをサポートするように拡張されています。
- ln (シンボリック・リンクを作成) や ls (シンボリック・リンクの内容を表示) などの関連する GNV ユーティリティが，シンボリック・リンクにアクセスしそれを管理できるように更新されています。

シンボリック・リンクおよび POSIX パス名処理についての詳細は，『*C* **ランタイム・ライブラリ・リファレンス・マニュアル**』の第 12 章を参照してください。

### 5.1.2 バイト単位のロック

C RTL は，fcntl 関数のコマンド F_GETLK，F_SETLK，および F_SETLKW を使用したバイト単位のファイル・ロックをサポートしています。この機能は，X/Open の仕様で定義されています。この機能を実装するために OpenVMS のロック・マネージャが使用されます。バイト単位のロックは，クラスタにまたがって使用することができます。ロックするオフセットには，32 ビットの符号無し整数に収まる値を使用できます。詳細は，『*C* **ランタイム・ライブラリ・リファレンス・マニュアル**』の fcntl 関数を参照してください。

### 5.1.3 新しい C RTL 関数

第 5.1.1 項に示したシンボリック・リンク関数の他に，X/Open 仕様に基づく以下の新しい関数が C RTL に追加されています。

```
crypt
setkey
encrypt
fchmod
```

### 5.1.4 C RTL の TCP/IP ヘッダ・ファイルの更新

CRTL では，ユーザが TCP/IP を呼び出すためのヘッダ・ファイルが提供されています。これらのヘッダには多数の問題があり，単純な TCP/IP プログラミング以外では使用できません。

以前は，いくつかのリリースの TCP/IP で，修正されたヘッダがプログラミング・サンプルの領域に提供されていました。C RTL に対するこの機能拡張により，これらの修正済みヘッダは C RTL ヘッダ・ライブラリ (DECC$RTLDEF.TLB) に格納されました。詳細は，『*OpenVMS V8.3* **リリース・ノート**』の C RTL の項を参照してください。

## 5.2 CDSA for OpenVMS および Secure Delivery

CDSA Version 2.2 for OpenVMS は，オープン・ソース・プロジェクトの CDSA を基にしています。また，OpenVMS 上での CDSA の実装は，Intel V2.0 Release 3 リファレンス・プラットフォームを基にしています。CDSA Version 2.2 for OpenVMS の新機能には，Secure Delivery と HRS (Human Recognition Service Standard) のサポートがあります。ここではこれらの機能について説明します。

- HRS (Human Recognition Service Standard) のサポート

  CDSA Version 2.2 for OpenVMS には，CDSA とともに動作する，HRS (Human Recognition Service) のサポートが含まれています。HRS は，CSSM (Common Security Services Manager) の EMM (Elective Module Manager) です。これによって汎用の認証サービスが提供され，CDSA で動作する何らかの形の生体認証 (バイオメトリクス) とともに使用するのに適しています。

  HRS は，アプリケーション開発者とバイオメトリクス技術開発者の両方が使用することを目的としています。登録，確認，識別の基本機能を網羅しており，バイオメトリクス・サービス・プロバイダ (BSP) が本人識別情報を最適な性能で管理できるようにするためのデータベース・インタフェースが含まれています。

- Secure Delivery

Secure Delivery では，公開鍵とディジタル署名技術を使用して， OpenVMS のユーザが， OpenVMS および他の OpenVMS ベンダから提供されたファイルの認証と検証を行うことが可能となります。

Secure Delivery では，ファイルに対するディジタル署名を作成できるため，ファイルとそれに関連するマニフェストをインターネットや CD/DVD 媒体などの保護されていないチャネル経由で配布することができます。ファイルが宛先のシステムに届いたら，マニフェストを使用して作成元を認証し，ターゲット・ファイルの内容を検証します。ターゲット・ファイルまたはマニフェストが何らかの形で改ざんされていた場合は，検証処理が失敗します。ファイルに署名するために使用された証明書が取り消されている場合は，検証処理に失敗します。

Secure Delivery は PCSI に組み込まれており， OpenVMS にインストールするソフトウェアが改ざんされていないことをインストール前に自動的に確認します。PCSI はインストールしようとしているすべてのキットでマニフェストがあるかどうかを確認します。マニフェストが見つからないと PCSI は警告を発し，処理を続行するかどうかを尋ねます。マニフェストが見つかったもののキットと一致しない場合は，インストールが中止されます。 PCSI データベースには，インストール時にキットが Secure Delivery を使用したかどうかについての情報が格納されます。

OpenVMS Version 8.3 の配布媒体に含まれているほとんどのキットは， Secure Delivery を使用して署名されています。 OpenVMS I64 では， OpenVMS のアップグレードの**最中またはその後**にインストールされる，マニフェストを持つレイヤード・プロダクト・キットが検証されます。 OpenVMS Alpha では，OpenVMS のアップグレード**後に**インストールされる，マニフェストを持つレイヤード・プロダクト・キットが検証されます。

OpenVMS Version 8.3 で Secure Delivery が有効になる前に作成されたキットも， OpenVMS Version 8.3 上にインストールすることができますが，これらのキットは， PCSI 履歴ファイル中で検証済みのキットとしてではなく，署名なしとしてマークされます。 Version 8.3 より前にインストールされた製品については，PCSI 履歴ファイル内の検証ステータスが空白になります。

詳細は，『Open Source Security for OpenVMS Alpha, Volume 1: Common Data Security Architecture』を参照してください。

CDSA の詳細は，次の場所にある Common Data Security Architecture の Web サイトを参照してください。

```
http://sourceforge.net/projects/cdsa/
```

## 5.3 デッドロック待ち

OpenVMS V8.3 では，1 秒未満のデッドロック待ち時間をプロセスがロック・マネージャに対して宣言できるようになりました。それには，システム・サービス$SET_PROCESS_PROPERTIES と，新しい項目コード PPROP$C_DEADLOCK_WAIT を使用します。この 1 秒未満のデッドロック待ち時間は，システム・パラメータ DEADLOCK_WAIT の時間より優先されます。さらに，システム・サービス$GETJPI とレキシカル関数 F$GETJPI では，項目コード JPI$_DEADLOCK_WAIT および DEADLOCK_WAIT を使用することで，この時間を読み出すことができます。使用方法の詳細については，『OpenVMS System Services Reference Manual』および『OpenVMS DCL ディクショナリ』を参照してください。

システム・パラメータ DEADLOCK_WAIT は秒単位であるため，設定できる最小の値は 1 秒です。システム・サービス$SET_PROCESS_PROPERTIES で設定した 1 秒未満のデッドロック待ち時間は，現在のイメージに対してだけ有効で，イメージが終了するとクリアされます。渡されるパラメータは 100n 秒単位の値で，1 秒を超えることはできません。値が小さすぎる場合は，最小値の 10m 秒に設定されます。パラメータ値ゼロでこのシステム・サービスを呼び出すと，以前設定した 1 秒未満の待ち時間をクリアすることができます。次の例を参照してください。

```
[...]
#define TEN_MSEC 100000

uint64 dead_wait;
uint64 prev_value;

 //
 // Set a 0.5 second deadlock wait time for the current process
 //
 dead_wait = 50 * TEN_MSEC;
 status = sys$set_process_properties ( 0, 0, 0, PPROP$C_DEADLOCK_WAIT, dead_wait, &prev_value );
[...]
```

## 5.4 デバッガの新機能

ここでは，OpenVMS Version 8.3 での OpenVMS Debugger の新機能について説明します。

### 5.4.1 C++ の演算子名のサポートの強化

Alpha システムと Integrity サーバ・システムで，C++ の演算子名のサポートが強化されました。特に，ユーザ定義の演算子名がサポートされるようになりました。この変更の前までは，演算子名を%NAME で囲む必要がありました。

以下に例を示します。

```
DBG> SHOW SYMBOL /FULL operator ==
routine C::operator==
    type signature: bool operator==(C)
    code address: 198716, size: 40 bytes
    procedure descriptor address: 65752
DBG> SET BREAK operator==
```

### 5.4.2 SET MODULE コマンドが省略可能になった

Alpha システムと I64 システムで， OpenVMS Debugger が自動的にモジュール情報を設定できるようになり，明示的な SET MODULE コマンドは省略可能となりました。この機能が実現されたのには 2 つの理由があります。

- デバッガはグローバル・シンボル名を認識するため，シンボルから，それが宣言されているモジュールを決定することができます。
- デバッガは，デバッガ・コマンドでモジュール名を指定すると，モジュール情報を自動的にロードします。たとえば，次のように入力したとします。

  ```
  DBG> SET BREAK X\Y
  ```

  デバッガは，モジュール X のモジュール情報がロードされていることを確認し，Y という名前のルーチンの情報を探します。以前のデバッガでは，モジュール X 内の情報がロードされていないことを示すメッセージが表示されましたが，本バージョンではこの情報が自動的にロードされるようになりました。

### 5.4.3 SHOW STACK コマンドの新しい修飾子

Alpha システムと Integrity サーバ・システムでは， SHOW STACK コマンドで /START_LEVEL=n修飾子が指定できるようになりました。この修飾子は， SHOW STACK に対して，呼び出しフレーム・レベルnのスタック情報から表示を開始するように指示します。

たとえば，フレーム 3 のスタック情報だけを表示するには，次のコマンドを入力します。

```
DBG> SHOW STACK/START=3 1
```

4 番目と 5 番目のスタック・フレームの詳細を表示するには，次のコマンドを入力します。

```
DBG> SHOW STACK/START=4 2
```

### 5.4.4 型のないストレージ位置に対する省略時のデータ型の変更

Alpha システムと Integrity サーバ・システムで，型のないストレージ位置に対する省略時のデータ型がロングワード (32 ビット) からクォドワード (64 ビット) に変更されました。データ型情報があるストレージ位置については，これまでどおり，関連する型に応じて表示されます。

### 5.4.5 SHOW SYMBOL コマンドでのオーバロード・シンボル・サポートの改善

Alpha システムと Integrity サーバ・システムでは， SHOW SYMBOL コマンドが機能拡張されて，オーバロード・シンボル名を認識するようになりました。これまでは，各種のオーバロード名が表示されるだけでした。この機能拡張によって，名前とその名前に関連付けられた情報が表示されるようになりました。以下に例を示します。

```
DBG> show symbol/full g
overloaded name C::g
    routine C::g(char)
        type signature: void g(char)
        address: 132224, size: 128 bytes
    routine C::g(long)
        type signature: void g(long)
        address: 132480, size: 96 bytes
```

### 5.4.6 GNAT Pro (Ada 95) コンパイラが Integrity サーバ・システムでも利用可能 (I64 のみ)

GNAT Pro (Ada 95) コンパイラが Integrity サーバ向けの OpenVMS 上でサポートされました。この製品については，直接 AdaCore 社にお問い合わせください。

弊社では HP Ada (Ada 83) コンパイラを OpenVMS Alpha から Integrity サーバ向けの OpenVMS に移植する予定はありません。

### 5.4.7 P2 空間にロードされたプログラムのデバッグをサポート

OpenVMS Version 8.3 Debugger では， P2 空間にロードされたプログラムのデバッグが可能となりました。

### 5.4.8 SET WATCH コマンドの改良

メモリ内の領域に対するウォッチポイント (静的ウォッチポイントと呼びます) では，非同期システム・サービスによる書き込みが検出されないことがあり，時には非同期システム・サービスによる書き込みが失敗することもあります。たとえば，SYS$QIO による出力パラメータ IOSB への非同期書き込みは，その IOSB が直接ウ

ォッチされている場合，あるいは，アクティブな静的ウォッチポイントと同じページ上にあるだけの場合も失敗することがあります。

本バージョンでは，デバッガがこの状態を検出するようになりました。問題が発生する可能性がある場合，デバッガは静的ウォッチポイントと非同期システム・サービスの衝突についてユーザに警告します。以下に例を示します。

```
DBG> g
%DEBUG-I-ASYNCSSWAT, possible asynchronous system service and static watchpoint collision
break at LARGE_UNION\main\%LINE 24192+60
DBG> sho call
 module name    routine name     line           rel PC           abs PC
*LARGE_UNION    main            24192       00000000000003A0 00000000000303A0
*LARGE_UNION    __main          24155       0000000000000110 0000000000030110
                                            FFFFFFFF80B90630 FFFFFFFF80B90630
DBG> ex/sour %line 24192
module LARGE_UNION
 24192:            sstatus = sys$getsyi (EFN$C_ENF, &sysid, 0, &syi_ile, &myiosb, 0, 0);
```

この条件が検出された場合にどのようにすればよいかを確認するには，デバッガで HELP MESSAGE ASYNCSSWAT と入力してください。

### 5.4.9 整数レジスタでの NaT (Not a Thing) のサポート

従来は，整数レジスタの NaT ビットが設定されたときに，デバッガはユーザに通知しませんでした。ユーザは，%GRNAT0 レジスタのビットを調べ，この情報を確認する必要がありました。以下の例では，整数レジスタ R9 および R10 の NaT ビットがオンになっています。

```
DBG> ex %r9:%r12
TEST\%R9:       0000000000000000
TEST\%R10:      0000000000000000
TEST\%R11:      0000000000000000
TEST\%SP:       000000007AC8FB70
DBG> ex/bin grnat0 <9,4,0>
TEST\%GRNAT0+1: 0110
DBG>
```

整数レジスタの NaT ビットがオンになっていると，デバッガが文字列 "NaT"を表示するようになりました。以下に例を示します。

```
DBG> ex %r9:%r12
TEST\%R9:       0000000000000000
TEST\%R10:      NaT
TEST\%R11:      NaT
TEST\%SP:       000000007AC8FB70
DBG> ex/bin grnat0 <9,4,0>
TEST\%GRNAT0+1: 0110
DBG>
```

型のオーバーライドを指定することで， NaT ビットがオンのレジスタの実際の生の値を表示することができます。以下に例を示します。

```
DBG> ex %r10
TEST\%R10:      NaT
DBG> ex/quad %r10
TEST\%R10:      0000000000000000
DBG>
```

### 5.4.10 デバッガの使いやすさの向上: モジュールの自動ロード

モジュール名がパス名に含まれていると，モジュールのシンボル・テーブルがデバッガによって自動的にロードされるようになりました。以前は，たとえば SET BREAK M\R などのコマンドでは，モジュール M のシンボルがロードされていないと，"unknown symbol R"で失敗していました。本バージョンでは，デバッガはまずモジュール M のシンボルをロードし，次にシンボル R を見つけるようになったため，このコマンドは成功します。

### 5.4.11 C++ デストラクタのサポートの強化

C++ のデストラクタ名がデバッガで認識されるようになりました。以前は，ユーザがデストラクタ名を%NAME レキシカルで囲む必要がありましたが，これは不要になりました。以下の例に新しい動作を示します。

```
DBG> examine C::~C
C::~C:              alloc       r35 = ar.pfs, 3F, 01, 00
DBG>
DBG> ex/source ~C
module CXXDOCEXAMPLE
    37:     ~C() {}
```

### 5.4.12 C++ テンプレート名のサポート

C++ のテンプレート名がデバッガで認識されサポートされるようになりました。以下に例を示します。

```
DBG> e Map<string, int>::operator[]
Map<string, int>::operator[]:               alloc       r34 = ar.pfs, 1E, 05, 00
```

### 5.4.13 Ada プログラムのサポートの強化

デバッガで，Ada で記述されたプログラムが部分的にサポートされるようになりました。デバッガは，パッケージ，子ユニット，プロシージャ，変数，単純なデータ型，モジュール型など，ほとんどの Ada 構造を理解します。タグ付きの型や制約なし配列へのポインタなど，より複雑なデータ型は，将来のリリースでサポートされる予定です。

## 5.5 Kerberos for OpenVMS

Kerberos Version 3.0 for OpenVMS は， MIT Kerberos V5 Release 1.4.1 を基にしています。(Kerberos の以前のバージョンである Version 2.1 は， MIT Kerberos V5 Release 1.2.6 を基にしていました。)

Kerberos Version 3.0 の新機能を以下に示します。

- MIT Kerberos V5 Release 1.4.1 へのアップグレード

  Kerberos for OpenVMS Version 3.0 では，コード・ベースが MIT Kerberos V5 Release 1.4.1 にアップグレードされています。 MIT Kerberos の各リリースでの変更点については，次の Web サイトにあるそれぞれのリリースの Readme ファイルを参照してください。

  ```
  http://web.mit.edu/kerberos/historical.html
  ```

- Kerberos ACME エージェント

  Kerberos for OpenVMS Version 3.0 には， Kerberos ACME エージェントが，Advanced Developer's Kit (ADK) の一部として含まれています。(これは， MIT Kerberos の標準部分である Kerberos ユーティリティで提供されている，既存の Kerberos 認証に対する追加機能です。この機能は， Kerberos を使用した UNIX システムでの pam_krb5 と同等の機能を提供します。)

  OpenVMS の以前のバージョンでは， Kerberos for OpenVMS ユーザは複数のログイン手順を踏む必要がありました。まず， OpenVMS 自体にログインし，次に Kerberos 証明書を入手するためにログインが必要でした。これらの手順では，異なるプリンシパル名 (ユーザ名) とパスワードが使用されていました。

  Kerberos ACME エージェントを使用するには， ACME Login Advanced Developer's Kit をインストールして設定します。インストールと設定については， SYS$HELP:ACME_DEV_README.TXT ファイルを参照してください。 Kerberos ACME エージェントの構成方法については，『HP Open Source Security for OpenVMS, Volume 3: Kerberos』を参照してください。

  ユーザ認証は， OpenVMS UAF (利用者登録ファイル) ではなく， Kerberos の KDC データベースを参照して処理されます。この新機能により， OpenVMS システム管理者の自由度が増します。ユーザ・データベースを統合し，複数の

OpenVMS システムとクラスタが自動的に単一の KDC を使用してユーザを認証するように構成することができます。

- AES 暗号化のサポート

  Kerberos for OpenVMS に， AES (Advanced Encryption Standard) のサポートが追加されました。 AES は対称鍵暗号方式であり，一般に使用されている DES (Data Encryption Standard) を置き換えるものです。 2003 年の 6 月に，米国政府は， AES が最も高いセキュリティ・レベルである最高機密レベルの機密情報を保護するのに十分な安全性を備えていると宣言しました。

- Kerberized SSH のサポート

  Kerberos for OpenVMS は， HP TCP/IP Services Version 5.6 for OpenVMS で実現された Kerberized SSH 機能をサポートしています。 Kerberized SSH では， Kerberos 証明書と SSH (secure shell) 接続を組み合わせて使用することができます。 (SSH では，ネットワーク経由で別のコンピュータにログインし，コマンドの実行やコンピュータ間でのファイルの移動を行うことができます。)

- クライアント・ライブラリでの TCP のサポート

  Kerberos for OpenVMS には，クライアント・ライブラリでの TCP のサポートが含まれています。これは，大量の PAC データのあるチケット向けの Microsoft の相互運用性拡張機能です。

- スレッド・セーフな KRB5 ライブラリ
- Kerberos RPC ライブラリでの RPCSEC_GSS 認証

Kerberos は，従来の (共有秘密鍵) 暗号方式を使用し，信頼できる第三者の認証サービスとして認証を実行します。 Kerberos は，プリンシパルの身元を確認する手段を提供しますが，ホスト・オペレーティング・システムによる認証に頼らず，ホスト・アドレスの信用にも基づいていません。また，ネットワーク上のすべてのホストの物理的なセキュリティが不要で，ネットワーク内でやり取りされるパケットは自由に読み取り，変更，挿入ができるという前提に立っています。クライアントとサーバが Kerberos を使用して身元を証明した後は，すべての通信を暗号化して，プライバシとデータの一貫性を確保することができます。

詳細は，『HP Open Source Security for OpenVMS, Volume 3: Kerberos』を参照してください。

最新版の Kerberos for OpenVMS をダウンロードする方法については，次の Web サイトを参照してください。

```
http://h71000.www7.hp.com/openvms/products/kerberos/
```

Kerberos についての詳細は，次の場所にある MIT Kerberos の Web サイトを参照してください。

```
http://web.mit.edu/kerberos/www/
```

## 5.6 リンカ・ユーティリティの機能拡張

C, C++, Ada などのコンパイラで生成されたオブジェクト・モジュールでは，シンボル・テーブル中のシンボルが変更されている可能性があります。これは，一般に「マングル化」と呼ばれます。これらの名前はリンカが参照する際のシンボル名であり，シンボルを解決するためにリンカが使用します。

マングル化を行う理由としては，プログラミング言語のオーバロード機能や，一意に名前を短縮する必要性が挙げられます。このようなモジュールをリンクし，シンボル未定義のメッセージが表示された場合は，リンカはオブジェクト・モジュールのシンボル・テーブルから取り出したシンボル名 (つまり，マングル化された名前) だけを表示します。このように処理されるため，未定義のマングル化されたシンボルとソース・コードのデマングル化された名前を照合するのが難しくなります。そのため，表示名情報 (DNI) を生成する将来のコンパイラをサポートするために，リンカは表示名情報を処理できるようになりました。また，リンカはソース・コード名を表示します。つまり，リンカは未定義のシンボル名をデマングル化することができます。さらに，ユニバーサル・シンボル (つまり，共用可能イメージからエクスポートされるシンボル) にデマングル化情報がある場合，リンカは生成される共用可能イメージにその情報を含めることができます。

シンボル解決処理は変更されていません。リンカは引き続きシンボル定義のマングル化されたシンボル名を使用してシンボル参照を解決します。シンボル・ベクタ・オプションも変更されておらず，これまでと同様に，名前 (マングル化された名前) がシンボル・テーブルに見つかる必要があります。

新しいコマンド行修飾子/DNI を使用して，デマングル化情報の処理を制御することができます。リンカに対してシンボル名のデマングル化を許可し，作成する共用可能イメージに，必要なデマングル化情報を格納するには，/DNI (省略時の指定) を指定します。以下の場合には/NODNI を指定します。

- デマングル化された名前が表示されないようにしたい場合。
- デマングル化情報を共用可能イメージに格納したくない場合。

グローバル・シンボル定義の変換テーブルが格納された新しいマップ・セクションを要求することができます。このテーブルは，マングル化されたシンボル名とデマングル化されたシンボル名を関連付けます。デフォルトでは，リンカはマップ・ファイル内にこのセクションを生成しません。このセクションの生成を要求するには，/FULL 修飾子にキーワード DEMANGLED_SYMBOLS を指定します。/FULL 修飾子の他のキーワードと同様に，/MAP 修飾子を指定します。最後に，/DNI 修飾子を指定します。マップ中の変換テーブルは，シンボル・ベクタのエントリを確認するのに役立ちます。

以下の例 (編集済み) は，デマングル化された名前がリンカのメッセージ内に表示される様子を示したものです。

```
$ cre foo.cxx
extern double foo (int) ;
double bar (void) { return foo (123); }
Ctrl/Z
$ cxx foo
$ link foo
%ILINK-W-NUDFSYMS, 1 undefined symbol:
%ILINK-I-UDFSYM,        CXX$_Z3FOOI1RLFMIE
%ILINK-W-USEUNDEF, undefined symbol CXX$_Z3FOOI1RLFMIE referenced
        source code name: "double foo(int)"
        section: .text
        offset: %X0000000000000020  slot: 2
        module: FOO
        file: DISK$USER:[JOE]FOO.OBJ;1
$
```

デマングル化情報のセクションは，オブジェクト・モジュールまたは共用可能イメージの一部です。シンボル名をデマングル化するために必要な情報は，すべてそのセクションに含まれているか，その情報のデマングル化を行うファシリテータ・ルーチン名と共用可能イメージ名が含まれます。この共用可能イメージは，通常は SYS$LIBRARY にあり，すでに OpenVMS 上にある言語の実行時環境によって提供されるか，言語コンパイラをインストールする際に提供されます。ファシリテータ・ルーチンが必要になると，リンカはイメージを起動してルーチンを呼び出します。ルーチンが共用可能イメージになかったり，イメージが見つからない場合など，いずれかの時点でこれに失敗すると，リンカは問題を示す情報メッセージを表示します。リンク操作は，デマングル化情報の共用可能イメージへの格納と同様に，ファシリテータ・ルーチンの呼び出しの成否とは無関係です。

## 5.7 ライブラリアンを使用したデマングル化された名前とマングル化された名前の一覧表示 (I64 のみ)

LIBRARY コマンドに/DEMANGLED_SYMBOLS 修飾子が追加されました。この修飾子を使用すると，ライブラリアンは ELF オブジェクトまたは ELF 共用可能イメージ・ライブラリ内の，言語プロセッサによって変更されたシンボル (マングリングと呼ばれます) と，それに対応するデマングル化された名前 (つまり，ソース・コードに記述されている名前) の一覧を表示します。マングル化された名前は，オブジェクト・モジュールの外部シンボル (または共用イメージのユニバーサル・シンボル) として出力されます。言語プロセッサが名前をマングル化する理由の 1 つは， C++ 言語の機能である関数のオーバロードです。

ライブラリアンは，オブジェクト・モジュールまたは共用可能イメージから取り出したシンボルを保存します。これらのシンボルには，マングル化されたシンボルが含まれています。これらのシンボル名をデマングル化するために必要な情報と，それに役

立つ共用可能イメージの名前が，オブジェクト・モジュールまたは共用可能イメージに格納されています。

ELF オブジェクト・ライブラリの場合は，オブジェクト・モジュールは完全にライブラリに格納されます。デマングル化情報を取得するために，オブジェクト・モジュールがメモリにマップされ検索されます。場合によっては，言語固有のデマングル化共用可能イメージもメモリにマップされて起動されます。シンボルはオブジェクト・ライブラリ・モジュールから読み込まれ，モジュールで提供される情報を使用してデマングル化されます。オブジェクト・ライブラリ・モジュールのどのシンボルがライブラリのシンボル名テーブルにあるかを示すために，ライブラリのシンボル名テーブルが検索されます。以降のモジュールも同様に処理され，一覧が生成されます。

ELF 共用可能イメージ・ライブラリの場合は，共用可能イメージはライブラリに格納されません。これは OpenVMS Alpha および OpenVMS VAX の共用可能イメージ・ライブラリと似ています。ただし，エクスポートされたシンボルとその他の最小限の情報だけがライブラリに格納されます。そのため，ライブラリアンは，デマングル化情報を取得するために，ライブラリの外部にある共用可能イメージを検索します。

ライブラリアンは 3 段階の検索を行います。この検索は，ファイルを正しく取得できると停止します。

1. まず，ライブラリアンはライブラリ・モジュール名に対して論理名の翻訳を行います。
2. それに失敗すると，ライブラリアンはファイル・タイプ.EXEを使用して，ライブラリがあるディスクとディレクトリ内でモジュール名を検索します。
3. それに失敗すると，ライブラリアンは最後に論理名 IA64$LIBRARY が示すデバイスとディレクトリを参照し，名前がライブラリ・モジュールのファイル名で，ファイル・タイプが.EXEのファイルを探します。ファイル指定が見つかると，ライブラリアンはそれをメモリにマップし， ELF オブジェクト・ライブラリで説明したのと同じ手順を実行します。

従来からある/OUTPUT=修飾子を使用して，デマングル化処理で生成された出力を任意の指定したファイルに出力することができます。/OUTPUT=修飾子を指定しないと，ライブラリアンは省略時の指定として現在のディスクおよびディレクトリに，ライブラリのファイル名と同じ名前で，ファイル・タイプが.LISのファイルに処理結果を出力します。

従来からある/ONLY=修飾子を使用すれば，デマングル化で選択されるライブラリ・モジュールを限定することができます。

たとえば，オブジェクト・ライブラリのデマングル化されたシンボルを表示し，出力を端末画面に送るには，コマンド行プロンプトで次の DCL コマンドを入力します。

```
$ LIBRARY /DEMANGLED_SYMBOLS /OUTPUT=SYS$OUTPUT OBJLIB.OLB
```

次の例では，共用可能イメージ・ライブラリ SHAREIMG.OLB のデマングル化されたシンボルを，ファイル DUMP.LIS に出力します。

```
$ LIBRARY /DEMANGLED_SYMBOLS /OUTPUT=DUMP.LIS SHAREIMG.OLB
```

次の例では，共用可能イメージ・ライブラリ SHAREIMG.OLB のデマングル化されたシンボルを，省略時のファイル指定 SYS$DISK:[ ]SHAREIMG.LIS に出力します。

```
$ LIBRARY /DEMANGLED_SYMBOLS SHAREIMG.OLB
```

次の例では，オブジェクト・ライブラリ OBJLIB.OLB 内のオブジェクト・モジュール MY_OBJ のデマングル化されたシンボルを，省略時のファイル指定 SYS$DISK:[ ]OBJLIB.LIS に出力します。

```
$ LIBRARY /DEMANGLED_SYMBOLS /ONLY=MY_OBJ OBJLIB.OLB
```

## 5.8 OpenVMS Alpha システム用の HP MACRO コンパイラ

MACRO コンパイラは， Alpha システム用の最新の GEM バックエンドを使用するようにアップグレードされました。また，以下に示すいくつかの機能拡張も行われています。

- /ARCHITECTURE 修飾子が追加されました。指定可能な値は， GENERIC, HOST， EV4， EV5， EV56， EV6， EV67，および EV7 です。 EV56 以降のアーキテクチャ値では， VAX の演算に対応する Alpha のバイト/ワード命令を，コンパイラが自動的に生成するようになりました。また，命令スケジューラが/ARCHITECTURE の値を適切に使用します。
- コマンド行サマリとほとんどの PAL 呼び出しのシンボル名を含め，機械語コードのリストが改良されました。
- 以下の 3 つの Alpha 命令のビルトインが追加されました。

  __ EVAX_CTLZ <RQ,WQ> CTLZ 命令を生成 (先行のゼロをカウント)

  __ EVAX_CTPOP <RQ,WQ> CTPOP 命令を生成 (ビットをカウント)

  __ EVAX_CTTZ <RQ,WQ> CTTZ 命令を生成 (後続のゼロをカウント)

  これらの新しいビルトインは， I64 コンパイラでも使用でき，同等の Itanium 命令が生成されます。
- コンパイラのイメージ名が SYS$SYSTEM:MACRO.EXE に変更されました。新しいコンパイラによる処理結果が正しくない場合に備えて，キット内には以前のイメージ SYS$SYSTEM:ALPHA_MACRO.EXE も残されています。新しいコンパイラは OpenVMS のエンジニアリングにおいて，システム自体をビルドするためのコンパイラとして使用されており，すでに広範なテストを実施済みです。以前の SYS$SYSTEM:ALPHA_MACRO.EXE コンパイラは，将来のリリースで削除される予定です。

## 5.9 RMS (Record Management System) の機能拡張

ここでは， OpenVMS Version 8.3 における RMS の機能拡張について説明します。

### 5.9.1 RMS CONVERT/FDL および CREATE/FDL の機能拡張

CONVERT/FDL ルーチンと CREATE/FDL ルーチンは， FDL ファイル指定として FDL 文字列を渡すことができるように拡張されています。/FDL 修飾子の値が引用符で囲まれた文字列であり，引用符で囲まれた POSIX パス名でない場合は， FDL 文字列として FDL パーサに渡されます。文字列中の引用符で囲まれた値は，二重の引用符で囲む必要があります。 FDL 文字列の使用方法についての詳細は，『HP OpenVMS Utility Routines Manual』の「FDL」の項を参照してください。

次の例を参照してください。

```
$  CONVERT/FDL="TITLE ""This is an FDL string"";File;org SEQ;Record;size 80" -
_$( input_file:  output_file:)
$   CREATE/FDL="record;format fixed;size 100" file.dat
```

### 5.9.2 RMS グローバル・バッファの索引編成ファイルに対する機能強化

OpenVMS Version 8.3 よりも前のリリースでは， RMS グローバル・バッファは P0 (32 ビット・アドレス) 空間だけにマップされていました。そのため，あるプロセスがアクセスするすべてのファイルに対して指定されたグローバル・バッファの合計は，プロセスあたり 1 GB 未満に制限されていました。このプロセスあたりの上限があることで， RMS ユーザがファイルごとに指定できるグローバル・バッファの数が十分ではありませんでした。また，グローバル・キャッシュに割り当てるファイルあたりの最大サイズも制限されていました (現在は 32767 バッファ)。

RMS グローバル・バッファの全体的な拡張性と性能を向上させるため，このリリースでは，索引編成ファイル用にグローバル・バッファが以下のように拡張されています。

- プロセスあたりの合計が 1 GB 未満という制限をなくすために，索引編成ファイルの RMS グローバル・バッファが P2 (64 ビット・アドレス) 空間にマッピングされるようになりました。 P2 空間にマッピングされた索引編成ファイル用のグローバル・バッファを利用するためにアプリケーションを変更する必要はありません。ただし，この機能拡張を利用するためには，索引編成ファイルに対してグローバル・バッファを有効にする必要があります。
- 索引編成ファイルでは，グローバル・キャッシュに割り当てるファイル当たりの最大サイズが，符号付きワード (32767) から符号付きロングワード (21 億) に変更されました。ファイルあたりの最大サイズの拡大を利用するためには， SET FILE/GLOBAL_BUFFER 修飾子で新しいオプションを使用する必要があります。

RMS グローバル・バッファはノードごとにローカルであるため，これらの機能拡張は， OpenVMS VAX または以前のバージョンのノードに対する変更を行わなくても，複合クラスタ環境の OpenVMS Alpha および OpenVMS I64 Version 8.3 のノード上で実施できるように設計されています。 Version 8.3 の Alpha ノードおよび Integrity サーバ・ノードでは，索引編成ファイルに対するグローバル・キャッシュ・サイズを増やすことができますが，他のノードでは， Version 8.3 よりも前のグローバル・キャッシュ・サイズのままで動作させることができます。これにより， OpenVMS VAX を含む以前のバージョンの OpenVMS を使用しているクラスタに，索引編成ファイルでより大きなグローバル・バッファ・キャッシュを使用できる Version 8.3 (またはそれ以降) の Alpha ノードまたは Integrity サーバ・ノードを段階的に追加することができるため，この機能は特に魅力的です。

### 5.9.3 新しい形式のグローバル・バッファ仕様

SET FILE/GLOBAL_BUFFER コマンドで，以下の 2 つの形式の指定が可能になりました。

1. SET FILE/[NO]GLOBAL_BUFFER[=n **ファイル名**]

   nには，共用可能なグローバル・バッファの数を，以前の値で設定します (最大 32767)。これにより，以前のバージョンの OpenVMS との間でファイルの互換性が保たれ，値はファイルのヘッダ内の元の位置に格納されます。

2. SET FILE/[NO]GLOBAL_BUFFER[=keyword[=n]] **ファイル名**

   keywordには以下のいずれかを指定します。

   - COUNT=n— 値nは，グローバル・バッファの数をロングワードで設定します。
   - PERCENT=p— 値pは，グローバル・キャッシュのサイズを，ファイルが現在使用している合計ブロック数に対する割合で指定します。
   - DEFAULT—RMS に対して，実行時にグローバル・バッファに関する 2 つの SYSGEN パラメータ GB_CACHEALLMAX および GB_DEFPERCENT を使用するアルゴリズムに基づいてグローバル・キャッシュ・サイズを再計算するように指示します。

   値nは，ファイルのヘッダ内の異なる位置に格納される新しい値を設定します。

たとえば，以下の 4 つのコマンドの動作はすべて異なります。

1. `$ SET FILE/GLOBAL_BUFFER=20 NEWFILE.DAT ! 互換性のある (古い) グローバル・バッファ・カウントを設定 (制限あり)`
2. `$ SET FILE/GLOBAL_BUFFER=COUNT=1000 NEWFILE.DAT ! 新しいグローバル・バッファ・カウントを設定 (制限なし)`
3. `$ SET FILE/GLOBAL_BUFFER=PERCENT=50 INVENTORY.DAT ! RMS に対し，ファイルの割合としてカウントを計算するよう指示`

4. $ SET FILE/GLOBAL_BUFFER=DEFAULT INV.INX ! RMS **に対し，合計ファイル・サイズに基づいてカウントを計算するよう指示**

古い構文 (SET FILE/GLOBAL_BUFFER=n) で指定されたグローバル・バッファでは，ファイル・ヘッダのある場所に設定が格納されます。新しい構文 (SET FILE /GLOBAL_BUFFER=keyword[=n]) では，ファイル・ヘッダの別の場所に格納され，新しいバリエーションのグローバル・バッファが使用されるため，より多くのバッファが使用可能になるとともに，ファイルの拡大に伴って自動的にサイズを調整することができます。

コマンド行では，1 つのバージョンのグローバル・バッファ修飾子だけを指定できる点に注意してください。以下に古いグローバル・バッファ指定と新しいグローバル・バッファ指定の両方を使用する例を示します。古い互換性のあるグローバル・バッファ・カウント値 (Version 8.3 よりも前の OpenVMS) には 100 を設定し，新しい値 (Version 8.3 以降) には 10000 を設定します。

```
$ SET FILE/GLOBAL_BUFFER=100 NEWFILE.DAT
$ SET FILE/GLOBAL_BUFFER=COUNT=10000 NEWFILE.DAT
```

OpenVMS のバージョンが混在するクラスタ環境では，異なるノード上で同じファイルを開くことができ，それぞれのノードで異なるグローバル・バッファ・カウントが使用されます。 Version 8.3 よりも前のノードでは，古い互換性のある設定が使用され， Version 8.3 以降のノードでは，新しい値が使用されます。

## 5.9.4 XABFHC **に新しいフィールドを追加**

読み取り専用の XABFHC (ファイル・ヘッダ特性) に，以下の 2 つのフィールドが追加されました。

| フィールド・オフセット | バイト数 | 説明 |
|---|---|---|
| XAB$B_RECATTR_FLAGS | 1 | レコード属性領域のためのフラグ |
| XAB$L_GBC32 | 4 | 拡張されたロングワードのグローバル・バッファ・カウント |

これらのフィールドのフィールド記述子は以下のとおりです。

- XAB$L_GBC32 — 拡張されたロングワードのグローバル・バッファ・カウントまたは割合。 XAB$B_RECATTR_FLAGS フィールドで XAB$V_GBC_PERCENT フラグを設定した場合は，このフィールドには割合を設定します。
- XAB$B_RECATTR_FLAGS — ファイル・ヘッダ内のレコード属性領域のためのフラグ。現在以下のフラグが実装されています。
  - XAB$V_GBC_PERCENT — グローバル・バッファ・カウント内の値が，ファイルが現在使用している総ブロック数に対する割合を表していることを示します。ファイルが現在使用している総ブロック数に基づいて，マップする実際のサイズが再計算されるため，時間とともに動的にサイズを拡大することができ

ます。サイズは，ノードに対する最初のアクセスでグローバル・キャッシュが作成されるたびに，実行時に決定されます。ユーザは， 100 パーセントよりも大きな割合を指定することで，いったんオープンされるとすぐにはクローズされず，急速にサイズが大きくなるようなファイルにも対応できます。

— XAB$V_GBC_DEFAULT — このフラグは， RMS に対し，グローバル・バッファに関係する 2 つの SYSGEN パラメータ GB_CACHEALLMAX と GB_DEFPERCENT を使用するアルゴリズムに基づいて，グローバル・キャッシュ・サイズを実行時に再計算するように指示します。省略時のオプションが有効な場合で，ファイルのサイズ (ブロック単位) が GB_CACHEALLMAX パラメータで指定されたサイズ以下の場合は， RMS はファイル全体をキャッシュできるだけのグローバル・バッファを割り当てます。サイズ (ブロック単位) が GB_CACHEALL MAX パラメータで指定されたサイズよりも大きい場合には， RMS はファイルのうち GB_DEFPERCENT パラメータ (グローバル・バッファの省略時の割合) で指定された割合のブロックをキャッシュできるだけのグローバル・バッファを割り当てます。

### 5.9.5 新しい RMS フィールド値

以下のフィールド値が追加されました。

| フィールド値 | 意味 |
|---|---|
| FAT$L_GBC32 | 拡張されたロングワードのグローバル・バッファ・カウント |
| FAT$RECATTR_FLAGS | レコード属性フラグ。以下のビット値が定義されています。<br><br>FAT$M_GBC_PERCENT—FAT$L_GBC32 の値をカウントではなく割合として解釈します。<br>FAT$M_GBC_DEFAULT—RMS は，グローバル・バッファ・カウントの省略時の値を設定して，FAT$W_GBC や FAT$L_GBC32 の値を無視します。 |

『OpenVMS I/O User's Reference Manual』から抜粋した図 5–1 は，新しいフィールドを反映するように更新されています。

**図 5–1 ACP-QIO レコード属性領域**


*FAT$V_RTYPE ビット 0 3; FAT$V_FILEORG ビット 4 7

ZK-0641-AI

## 5.9.6 グローバル・バッファ・キャッシュのサイズを決定するための，RMS のファイルごと新しい管理オプション

本リリースでは，ファイルごとの管理オプションが 2 つ追加されており，すべての RMS ファイル編成 (順編成，相対編成，索引編成) のグローバル・バッファ・キャッシュのサイズをより簡単に決定することができます。これらの新しいオプションは，既存のグローバル・バッファ・カウント・オプションを強化するものです。

- PERCENT — グローバル・バッファ・カウントをサイズで指定するのではなく，ファイルが現在使用している総ブロック数に対する割合としてグローバル・キャッシュのサイズを指定します。ファイルが現在使用している総ブロック数に基づいて，マップする実際のサイズが再計算されるため，時間とともに動的にサイズを拡大することができます。サイズは，ノード上のファイルに対する最初のアクセスでグローバル・キャッシュが作成されるたびに，実行時に決定されます。ユーザは，100 パーセントよりも大きな割合を指定することで，いったんオープンされるとすぐにはクローズされないファイルでもキャッシュのサイズをすぐに拡大することができます。
- DEFAULT — 実行時にノード上のファイルへの最初のアクセスでグローバル・キャッシュが作成されるたびに，いくつかのファイル特性に基づいてグローバル・キャッシュ・サイズを計算するよう RMS に指示します。

RMS グローバル・バッファ・カウント (GBC) の省略時の指定は，グローバル・バッファ (GB) に関係する 2 つの新しい SYSGEN パラメータ GB_CACHEALLMAX および GB_DEFPERCENT を使用するアルゴリズムに基づいて計算する方法です。省略時のオプションが有効な場合で，ファイルのサイズ (ブロック単位) が GB_

CACHEALLMAX パラメータで指定されたサイズ以下の場合は， RMS はファイル全体をキャッシュできるだけのグローバル・バッファを割り当てます。

サイズ (ブロック単位) が GB_CACHEALL MAX パラメータで指定されたサイズよりも大きな場合には， RMS はファイルのうち GB_DEFPERCENT パラメータ (グローバル・バッファの省略時の割合) で指定された割合のブロックをキャッシュできるだけのグローバル・バッファを割り当てます。 100 パーセントよりも大きな割合を指定して，ファイルが大きくなった場合に備えてキャッシュ領域を増やすこともできます。

### 5.9.7 ファイルに接続されたグローバル・バッファ・キャッシュのサイズ (XAB$_GBC)

SENSEMODE 操作で$CONNECT または$DISPLAY に対する項目リスト XAB とともに項目コード XAB$_GBC を使用すると，ファイルに接続されたグローバル・バッファの実際の数を参照することができます。ファイルのグローバル・セクションは，各ノード上での最初のアクセスによって作成されファイルと接続されます。

項目コード XAB$_GBC は，接続されているキャッシュのサイズを返すために 4 バイトのバッファを必要とします。 SETMODE をこの項目とともに使用することはできません。

このオプションは， DECnet の操作ではサポートされず，無視されます。

### 5.9.8 グローバル・バッファ数 (XAB$_GBC32)

項目コード XAB$_GBC32 は， SETMODE 操作で$CREATE に対して項目リスト XAB とともに使用できます。この項目コードは，ファイルを作成する際に，ロングワードのグローバル・バッファ数を永久属性としてファイル・ヘッダのレコード属性領域に設定します。また， SETMODE で$CONNECT に対して項目リスト XAB とともに使用して，実行時にファイル・ヘッダ内の永久属性より優先させることもできます。この優先指定は，そのプロセスが各ノード上でキャッシュに接続する最初のプロセスである場合にだけ適用されます。

項目コード XAB$_GBC32 を使用しても，ファイルに接続されているグローバル・バッファ数を参照することはできません。実際のグローバル・バッファ数を参照するには，項目コード XAB$_GBC を使用します。

項目コード XAB$_GBC32 は，キャッシュ・サイズを格納するために 4 バイトのバッファを必要とします。キャッシュ・サイズは，実際の個数として要求することも，実行時のファイルの合計ブロック数のパーセンテージとして要求することもできます。キャッシュ・サイズをパーセンテージで返すように指定するためには，項目リスト XAB に項目コード XAB$_GBCFLAGS も含める必要があります。これが含まれてい

ないと，項目コード XAB$_GBC32 で返されるキャッシュ・サイズの値は，個数となります。

索引編成ファイルの個数として，最大値%x7FFFFFFF を指定できます。順編成ファイルと相対編成ファイルは，最大 32767 個に制限されます。いったんオープンされるとめったにクローズされないファイルでは，ファイルが素早く拡大されるように，100 パーセントよりも大きなパーセンテージを指定することができます。

このオプションは， DECnet の操作ではサポートされず，無視されます。

### 5.9.9 **グローバル・バッファ・フラグ** (XAB$_GBCFLAGS)

項目コード XAB$_GBCFLAGS は， SETMODE 操作で$CREATE に対する項目リスト XAB とともに使用することができます。この項目コードは，ファイルを作成する際に，グローバル・バッファ・フラグの値を永久属性としてファイル・ヘッダのレコード属性領域に設定します。また， SETMODE で$CONNECT に対して項目リスト XAB とともに使用して，実行時にファイル・ヘッダ内の永久属性より優先させることもできます。この優先指定は，そのプロセスが各ノード上でキャッシュに接続する最初のプロセスである場合にだけ適用されます。

グローバル・バッファ・フラグの値を参照するには， SENSEMODE 操作で$CONNECT または$DISPLAY に対する項目リスト XAB とともに項目コード XAB$_GBCFLAGS を使用します。最初に接続したプロセスがファイルのグローバル・セクションを作成するときにグローバル・バッファ数を計算するために使用したグローバル・バッファ・フラグが返されます。

以下の 2 つのフラグが使用可能です。

- XAB$M_GBC_PERCENT — グローバル・バッファ数の値を，現在ファイル内にある合計ブロック数のパーセンテージとして表現することを指示します。これにより RMS は，ファイル中で使用されている現在の合計ブロック数に基づいてマッピングする実際のサイズを再計算するため，時間とともにサイズが動的に拡張できます。実行時にノード上での最初のアクセスによってグローバル・キャッシュが作成されるたびにサイズが決定されます。いったんオープンされるとめったにクローズされないファイルについては，ファイルが素早く拡大できるように，100 パーセントよりも大きなパーセンテージを指定できます。
- XAB$M_GBC_DEFAULT — 2 つのグローバル・バッファ (GB) SYSGEN パラメータ GB_CACHEALLMAX および GB_DEFPERCENT を使用したアルゴリズムに基づいて，グローバル・キャッシュ・サイズを実行時に再計算するよう RMS に要求します。デフォルトのオプションが有効で，ファイルのサイズ (ブロック単位) が GB_CACHEALLMAX パラメータで指定されたサイズ以下の場合は， RMS はファイル全体をキャッシュするのに十分なグローバル・バッファを割り当てます。サイズ (ブロック単位) が GB_CACHEALLMAX パラメータで指定されたサイズよりも大きい場合は， RMS はファイルのうち GB_DEFPERCENT (グローバ

ル・バッファのデフォルト・パーセント) パラメータで指定されたパーセンテージをキャッシュするのに十分なグローバル・バッファを割り当てます。

項目コード XAB$_GBCFLAGS では，XAB$_GBC_PERCENT または XAB$_GBC_DEFAULT のフラグ値を格納するために，4 バイトのバッファが必要です。

このオプションは DECnet の操作ではサポートされず，無視されます。

## 5.10 HP SSL for OpenVMS

SSL (Secure Sockets Layer) は，インターネットを介して機密情報を安全にやり取りするための，オープン・スタンダードなセキュリティ・プロトコルです。HP SSL Version 1.3 は，OpenSSL 0.9.7e を基にしています。(以前のバージョンの HP SSL は，OpenSSL 0.9.7d を基にしていました。)

HP SSL は，OpenVMS I64，OpenVMS Alpha，および OpenVMS VAX でサポートされています。

HP SSL Version 1.3 は，レイヤード・プロダクトとしてではなく，SIP (システム統合製品) として OpenVMS オペレーティング・システムに含まれています。Version 1.3 には，OpenSSL 0.9.7e に含まれているバグ修正も反映されています。これらの機能について以下で説明します。

- SIP (システム統合製品) としての SSL

  SSL for OpenVMS は，OpenVMS Version 8.3 をインストールまたはアップグレードすると，自動的にインストールされます。個別に PCSI ファイルをインストールする必要はなくなりました。

- OpenSSL 0.9.7e でのバグ修正

  OpenSSL 0.9.7d から OpenSSL 0.9.7e への主な変更点は以下のとおりです。

  - マルチスレッド環境で CRL をチェックする場合の競合条件を修正
  - 拡張コードに対して Delta CRL を追加
  - 警報が正常に送信されるように s3_pkt.c を修正
  - OpenSSL 証明書作成ユーティリティを使用した場合に，発行者名とシリアル番号が重複するケースを削減 (RFC3280 違反)

HP SSL は，インターネット等の TCP/IP ネットワークを介した通信に関する以下の 3 つの基本的なセキュリティ問題に対処しています。

- SSL サーバの認証 — ユーザがサーバの身元を確認することができます。SSL 対応のクライアント・ソフトウェアは，公開鍵暗号化の標準技術を使用して，サーバの証明書と公開 ID が正当かどうかと，その証明書がクライアントの信頼できる認証局 (CA) リストに載っている CA によって発行されたものであるかどうかを確認します。サーバの認証は，PC のユーザが Web で買い物をするためにクレジッ

ト・カード番号を送信する際に，送信先のサーバの身元を確認したい場合などに使用します。

- SSL クライアントの認証 — サーバがユーザの身元を確認することができます。サーバの認証で使用したのと同じ手法を使用し， SSL 対応のサーバ・ソフトウェアは，クライアントの証明書と公開 ID が正当かどうかと，その証明書がサーバの信頼できる認証局 (CA) リストに載っている CA によって発行されたものであるかどうかを確認します。クライアントの認証は，銀行が機密の財務情報を顧客に送る際に，受信者の身元を確認したい場合などに使用します。
- 暗号化された SSL 接続 — クライアントとサーバの間で送信されるすべての情報を，送信側のソフトウェアで暗号化し，受信側のソフトウェアで復号化します。これにより，高い機密性が得られます。プライベートなトランザクションでは，どちらの側にとっても機密性は重要です。また，暗号化された SSL 接続上で送信されるすべてのデータは，転送中に改ざんされていないかどうかを自動的に検出できる仕組みで保護されています。

詳細は，『HP Open Source Security for OpenVMS, Volume 2: HP SSL for OpenVMS』を参照してください。

最新版の HP SSL for OpenVMS をダウンロードする方法については，次の Web サイトを参照してください。

```
http://h71000.www7.hp.com/openvms/products/ssl/
```

OpenSSL についての詳細は，次の場所にある OpenSSL の Web サイトを参照してください。

```
http://www.openssl.org/
```

## 5.11 システム・サービスの新しい情報と新しい項目コード

いくつかのシステム・サービスに新しい項目コードが追加されています。また，いくつかの項目コードで情報が追加されています。ここでは，これらのトピックについて説明します。

### 5.11.1 $GETDVI: 新しい項目コードと項目コード情報

OpenVMS Version 8.3 では，以降の項で説明する項目コードが追加されており，項目コードに関する情報も追加されています。

#### 5.11.1.1 $GETDVI の新しい項目コード

$GETDVI の新しい項目コードは以下のとおりです。

```
DVI$_DEVICE_MAX_IO_SIZE
DVI$_FC_HBA_FIRMWARE_REV
DVI$_LAN_ALL_MULTICAST_MODE
DVI$_LAN_AUTONEG_ENABLED
DVI$_LAN_DEFAULT_MAC_ADDRESS
DVI$_LAN_FULL_DUPLEX
DVI$_LAN_JUMBO_FRAMES_ENABLED
DVI$_LAN_LINK_STATE_VALID
DVI$_LAN_LINK_UP
DVI$_LAN_MAC_ADDRESS
DVI$_LAN_PROMISCUOUS_MODE
DVI$_LAN_PROTOCOL_NAME
DVI$_LAN_PROTOCOL_TYPE
DVI$_LAN_SPEED
DVI$_MAILBOX_BUFFER_QUOTA
DVI$_MAILBOX_INITIAL_QUOTA
DVI$_PREFERRED_CPU_BITMAP
DVI$_VOLUME_PENDING_WRITE_ERR
DVI$_VOLUME_RETAIN_MAX
DVI$_VOLUME_RETAIN_MIN
DVI$_VOLUME_SPOOLED_DEV_CNT
DVI$_VOLUME_WINDOW
```

#### 5.11.1.2 $GETDVI の項目コード情報

文字列データ型を返す項目コードでは，返されるデータを格納するのに十分な大きさのバッファを渡さないと，エラーにならずにデータが切り捨てられます。$GETDVI が完了したら，文字列を返すそれぞれの項目コードについて，項目リスト記述子の，返された長さフィールドを確認することをお勧めします。

返された長さが，返却データを保持するために割り当てたバッファのサイズと等しい場合は，データが切り捨てられている可能性があります。その場合は，割り当てたバッファのサイズよりも返却データの長さが小さくなるまで，バッファのサイズを増やしながら繰り返し$GETDVI を呼び出します。

項目コードの説明で別途指定されていないかぎり，返却文字列を保持するために 32 バイトのバッファを使用することをお勧めします。$GETDVI は，バッファの未使用の領域をヌル文字で埋めます。

### 5.11.2 $GETJPI の新しい項目コード

システム・サービス$GETJPI には，新しい項目コード JPI$_DEADLOCK_WAIT が追加されています。

### 5.11.3 $GETSYI の新しい項目コード

システム・サービス$GETSYI には，以下の新しい項目コードが追加されています。

SYS$_ACTIVE_CPU_BITMAP
SYI$_AVAIL_CPU_MASk
SYI$_BOOT_DEVICE
SYI$_IO_PRCPU_BITMAP
SYI$_POWERED_CPU_BITMAP

### 5.11.4 $GETDVI，$GETJPI，$GETLKI，$GETQUI，$GETSYI のサービス情報

このシステム・サービス呼び出しの完了と同期を取るためにイベント・フラグを使用していない場合は，イベント・フラグとして値 EFN$C_ENF (イベント・フラグなし) を使用することを強くお勧めします。$EFNDEF マクロは EFN$C_ENF を定義しています。詳細は，『HP OpenVMS Programming Concepts Manual』を参照してください。

### 5.11.5 $GETUAI の新しい項目コード

システム・サービス$GETUAI には，以下の新しい項目コードが追加されています。

UAI$V_DISPWDSYNCH
UAI$V_VMSAUTH

### 5.11.6 システム・サービスに対するその他の変更

$IO_FASTPATH と$PROCESS_AFFINITY では，新しい CPU 名前空間プロジェクトに関連して，エントリが追加または変更されています。$SET_PROCESS_PROPERTIESW では，システム・サービス・ログに関連してその他の追加と変更が行われています。

詳細は，『OpenVMS System Services Reference Manual』を参照してください。

## 5.12 トレースバック機能

プログラムの位置をシンボル化する呼び出し可能インタフェースが， OpenVMS Alpha と OpenVMS I64 の両方で提供されるようになりました。以前は，このインタフェースは OpenVMS I64 でのみ提供されていました。

Alpha システムでの新しいインタフェースの名前は TBK$ALPHA_SYMBOLIZE であり， Integrity サーバ・システムの TBK$I64_SYMBOLIZE ルーチンに似ています。

Integrity サーバでは，本リリースで提供された Alpha のルーチン・インタフェース (TBK$ALPHA_SYMBOLIZE) に合わせて，ルーチン・インタフェース (TBK$I64_SYMBOLIZE) が変更されました。この変更は，旧製品との互換性があります。つまり，古いインタフェースは文書化されなくなりましたが，現在古いインタフェースを使用しているプログラム向けにサポートは継続されます。

どちらのインタフェースも USER， SUPER，および EXEC のモードからの呼び出しがサポートされています。以前の OpenVMS I64 では， USER モードからの呼び出ししかサポートされていませんでした。

Integrity サーバおよび Alpha 向けのトレースバック・シンボル化ルーチンについての詳細は，『HP OpenVMS Utility Routines Manual』を参照してください。

# 6

# InfoServer ユーティリティ

この章では， OpenVMS I64 および OpenVMS Alpha で利用可能な InfoServer ユーティリティの機能について説明します。この章には， InfoServer ハードウェアと InfoServer アプリケーションの比較と InfoServer ユーティリティ・コマンドのリファレンスが含まれています。

## 6.1 InfoServer ユーティリティの概要

InfoServer アプリケーションを使用すると， LAN 上の仮想ディスク・デバイス用のサービスを作成することができます。

仮想ディスク・デバイスには，以下のものがあります。

- DVD ドライブ
- 特定のディスク・ドライブ: SCSI および Fibre Channel
- CD ドライブ
- パーティション (コンテナ・ファイルとも言う)

### InfoServer ハードウェアと InfoServer アプリケーションの比較

OpenVMS の新しい InfoServer アプリケーションは，以前の InfoServer ハードウェアとは，いくつかの点で大きく異なっています。最も大きな違いは，以下のとおりです。

- DCL スタイルのコマンド構文の使用
- デバイスのサービスを作成する前に，そのデバイスをマウントしておかなくてはならない点
- DVD ドライブ用サービスの作成のサポート
- テープ・デバイスはサポートされない
- CD-R (CD-recordable) ドライブはサポートされない
- 自動マウントはサポートされない

### 6.1.1 InfoServer の使用方法の概要

InfoServer ユーティリティ・コマンドを使用すると，以下の作業が実行できます。

- LAN 上の仮想ディスク・デバイスのサービスの作成と削除
- 一連のアクティブな InfoServer サービスの保存
- 既存のサービスの属性の変更
- サービスに接続されているサーバとノードについての情報の表示
- 1 つまたは複数のサービスについて，サービス固有の情報の表示
- LASTport/Disk サーバの起動とサーバおよびキャッシュの各種の特性の設定

InfoServer は以下の方法で起動できます。

- **RUN コマンドの使用**

  RUN コマンドを使用して InfoServer を起動するには， DCL コマンド・プロンプトで次のように入力します。

  ```
  $ RUN SYS$SYSTEM:ESS$INFOSERVER
  ```

  InfoServer ユーティリティからプロンプトが表示されたら，次のように InfoServer のコマンドを入力します。

  ```
  InfoServer> SHOW SERVER
  ```

  InfoServer がコマンドの実行を完了すると，再び InfoServer>プロンプトが表示されます。ユーティリティを終了するまで，この動作が繰り返されます。

- **InfoServer をフォーリン・コマンドとして定義**

  DCL プロンプト，または起動コマンド・ファイルまたはログイン・コマンド・ファイルで，次のように入力することで， InfoServer をフォーリン・コマンドとして定義することができます。

  ```
  $ InfoServer :== $ESS$INFOSERVER
  ```

  ログイン・コマンド・ファイルを実行した後は， DCL プロンプトで INFOSERVER と入力するだけで，このユーティリティを起動できます。

  ```
  $ INFOSERVER
  ```

  以下の点に注意してください。

  – InfoServer をフォーリン・コマンドとして使用し，また InfoServer のコマンドを入力すると，次のように，このユーティリティはコマンドを実行した後に終了し， DCL コマンド・プロンプトの状態に戻ります。

    ```
    $ InfoServer SHOW SERVER
    $
    ```

- InfoServer のコマンドを指定せずに InfoServer をフォーリン・コマンドとして使用すると，このユーティリティは InfoServer>プロンプトを表示するので，その段階で，次のように，コマンドが入力できます。

```
$ InfoServer
InfoServer>  SHOW SERVER
```

**注意**

**すべての** InfoServer **コマンドは，** SYSPRV **および** OPER **の権限を必要とします。**

InfoServer ユーティリティを終了するには， InfoServer>プロンプトで EXIT コマンドを入力するか Ctrl/Z を押します。

InfoServer ユーティリティについての情報を表示するには， InfoServer>プロンプトで HELP コマンドを入力します。

### 6.1.2 InfoServer **コマンド**

以降の項で， InfoServer コマンドの説明と例を示します。

## CREATE SERVICE

指定したデバイスまたはパーティションのサービスを作成します。

**使用方法**:

- すべてのデバイスは，システムワイドにマウントし，プロセスがログアウトしたときにマウント解除されないようにする必要があります。
- 読み書きサービスを持つデバイスは，/FOREIGN 修飾子を付けてマウントし，OpenVMS から見えないようにする必要があります。
- 読み取り専用サービスを持つデバイスは，/NOWRITE 修飾子または/FOREIGN 修飾子を付けてマウントし，誰もローカルに変更できないようにする必要があります。
- 読み取り専用または読み書きアクセスで OpenVMS にマウントされるディスクから，パーティションのサービスを提供できます。
- 今回のリリースでは，パーティションのサポートは制限されています。

## フォーマット

CREATE SERVICE *serviceName device-or-partitionName*

## パラメータ

serviceName
LAN 上で使用されるサービスの名前です。サービス名は英数字とドル記号 ($) で構成されます。長さは最大で 255 文字です。

device-or-partitionName
LAN 上で使用される OpenVMS のディスク・デバイスまたはパーティションの名前です。入力するデバイス名またはパーティション名は，作成済みのものでなければなりません。

デバイス名とパーティション名について，以下に説明します。

- デバイス名

  LAN にサービスを提供するデバイスは， OpenVMS ディスク・デバイスです。InfoServer デバイス名を指定するときには， OpenVMS デバイス名を使用します。デバイス名は， SHOW SERVICES コマンドで表示される名前に完全に一致するか，またはワイルドカードを含むかのいずれかでなければならないことに注意してください。

InfoServer ユーティリティでワイルドカードが使用できるところでは，OpenVMS で使用されるのと同じワイルドカードが使用できます。パーセント文字 (%) は正確に 1 文字に一致し，アスタリスク文字 (*) は 0 文字以上の文字列に一致します。

ディスク指定はコロンで終わる必要があります。

- パーティション名

  パーティションは，ネットワークにサービスが提供されるコンテナ・ファイルです。したがって，デフォルトのファイル・タイプが.ESS$PARTITION の OpenVMS ファイル名を持っています。デバイス名，ディレクトリ名，ファイル名を含むパーティション名の長さは，最大で 242 文字です。

  今回のバージョンでは，パーティションのサポートは制限されています。ハード・ドライブの分割利用には， LD デバイスを使用することをお勧めします。詳細は， DCL コマンドの LD HELP を参照してください。

## 修飾子

/CLASS=className
全体の LASTport Disk (LAD) 名前空間のサブセットを指定します。

クラス名の目的は，名前空間を分割して，クライアントに対して意味のある名前だけを示すことです。クラス名を使用すると， 2 つのサービスに同じ名前を付けて，互いに競合しないようにすることもできます。

たとえば，いくつかのクライアント・システムで使用される異なるオンディスク構造に対して，異なるクラス名をつけることができます。あるクライアント・システムでは SERVICEA/CLASS=ODS-2 を使用し，別のクライアント・システムでは SERVICEA/CLASS=ISO_9660 を使用します。サービス名は同じ (SERVICEA) ですが，クラス名は異なります。

使用するクラス名は，作成するサービスに接続するクライアント・システムに依存します。デフォルトのクラス名は ODS_2 です。たとえば， OpenVMS システムは， InfoServer デバイスをマウントするとき， ODS_2 名前空間を使用します。OpenVMS クライアントは， ODS_2 サービス・クラスに属しているサービスのみ利用できることに注意してください。

有効なクラス名は，次のとおりです。

| | |
|---|---|
| V2.0 | PCSA MS-DOS クライアントが理解する名前 |
| Unformatted | 仮想ディスクは形式を持っていません |
| MSDOS | MSDOS 仮想ディスク |
| ODS_2 | VMS 仮想ディスク |
| UNIX | UNIX 仮想ディスク |
| ISO_9660 | ISO 9660 CD 形式 |
| HIGH_SIERRA | MS-DOS CD 形式 |
| APPLE | Macintosh HFS 形式 |
| SUN | Sun 形式 |

/ENCODED_PASSWORD=hexstring
SAVE コマンドでこの修飾子が作成されます。パスワードは平文で保存されず，ハッシュ化されたパスワードの値が SAVE 操作の一環として書き込まれるため，パスワードを人目にさらすことなく，サービスを作成しなおすことができます。

SAVE コマンドが作成するコマンド・プロシージャを編集してサービス名を変更した場合には，エンコードされたパスワード値が有効ではなくなってしまうことに注意してください。その場合には，/PASSWORD 修飾子を使用して，サービスに別のパスワードを設定する必要があります。

/PASSWORD=passwordString
/NOPASSWORD (**デフォルト**)
オプションのサービス・アクセス制御パスワードを指定します。パスワードが設定されたサービスにクライアント・システムがアクセスするためには，パスワードを指定する必要があります。

パスワードは最大 39 文字の英数字文字列です。パスワードを指定しなかった場合には，クライアント・システムではこのサービスにアクセスする場合に，パスワードの入力を求められません。

テキストのパスワードは暗号化された形式で，他のサービス情報とともにメモリに保存されます。

/RATING=DYNAMIC
/RATING=STATIC=value
該当するサービスが複数ある場合には，クライアントはサービス評価点を使用してサービスを選択します。最高のサービス評価点を持ったサービスが選択されます。

システムは動的サービス評価点を負荷に応じて調整します。静的評価点を 0 ～ 65535 の範囲で指定することもできます。システムは静的評価点は調整しません。

静的評価点を使用する場合の例は，サービスのコピーの 1 つから，別のコピーにクライアントを移行させる場合です。クライアントを移動させたいサービスに静的評価点 0 を設定しておけば，新しいクライアントが評価点 0 のサービスに接続することはありません。代わりに，それよりも高い評価点を持つサービスに接続します。すべての現在のクライアントがサービスとの接続を解除したときに，そのサービスを安全に削除することができます。

/READAHEAD (**デフォルト**)
/NOREADAHEAD
キャッシュ・ブロックを埋めるためのディスクの読み取りが必要になったときに，/READAHEAD 修飾子は，読み取り範囲が，要求された最初のブロックから，バケット境界の最後までであることを指示します。先読みすることによって，必要なディスク・ブロックが前もってキャッシュにロードされるので，シーケンシャル操作の速度が向上します。

/READAHEAD 修飾子と/READBEHIND 修飾子の両方を指定しておくと，キャッシュ・バケット内のブロックが要求されたときに，そのブロックのバケット範囲全体がキャッシュに読み込まれます。

/READBEHIND
/NOREADBEHIND (**デフォルト**)
キャッシュ・ブロックを埋めるためのディスクの読み取りが必要になったときに，/READBEHIND 修飾子は，読み取り範囲が，キャッシュ・バケット境界の先頭から要求されたブロックまでであることを指示します。

/READAHEAD 修飾子と/READBEHIND 修飾子の両方を指定すると，キャッシュ・バケット内のブロックが要求されたときに，そのブロックのバケット範囲全体がキャッシュに読み込まれます。

/READERS=number (**デフォルトは，**READERS=1000)
/NOREADERS
クライアントの読み取りアクセスで許される同時接続の最大数を指定します。デフォルトは 1000 接続です。値 0 は書き込み専用であることを意味します。

クライアントがサービスに対して読み取り専用アクセスまたは読み書きアクセスを要求した場合には，システムは読み取り接続としてカウントします。

/WRITERS
/NOWRITERS (**デフォルト**)
サービスが 1 つの書き込み接続を許すことを指定します。

## 例

1. 
```
$ SHOW DEVICE MOVMAN$DQA0:/full
```

```
Disk MOVMAN$DQA0:, device type Compaq  CRD-8322B, is online, file-oriented
    device, shareable, served to cluster via MSCP Server, error logging is
    enabled.
```

```
 Error count                0    Operations completed
 Owner process             ""    Owner UIC                [SYSTEM]
 Owner process ID     00000000    Dev Prot        S:RWPL,O:RWPL,G:R,W
 Reference count             0    Default buffer size             512
 Total blocks         16515072    Sectors per track                63
 Total cylinders         16384    Tracks per cylinder              16

$ MOUNT/SYSTEM dqa0 OVMSIPS11
Volume is write locked
OVMSIPS11 mounted on _MOVMAN$DQA0:

$ InfoServer
InfoServer> CREATE SERVICE VMS_SIPS_V11 _MOVMAN$DQA0:
%INFOSRVR-I-CRESERV, service VMS_SIPS_V11 [ODS-2] created for
_MOVMAN$DQA0:.
```

この例は， CD デバイス用のサービスの作成方法を示しています。

- SHOW DEVICE . . . /FULL コマンドは，_MOVMAN$DQA0 CD について完全な情報リストを表示します。
- MOUNT/SYSTEM コマンドは， OVMSIPS11 ボリュームを_MOVMAN$DQA0: CD にマウントします。
- InfoServer の CREATE SERVICE コマンドは，_MOVMAN$DQA0 CD 上に VMS_SIPS_V11 サービスを作成します。

2. $LD CREATE KIT1/SIZE-100000

```
 $DIRECTORY KIT1
Directory DKB0:[DISKS]

KIT1.DSK;1       100000/100008   29-APR-2005 14:14:43.49

Total of 1 file, 100000/100008 blocks.

 $ LD CONNECT KIT1
%LD-I-UNIT, Allocated device is MOVMAN$LDA1:

 $ CREATE SERVICE TEST_KIT_1 MOVMAN$LDA1:
%INFOSRVR-I-CRESERV, service TEST_KIT_1 [ODS-2] created for
 _MOVMAN$LDA1:
```

この例は，論理ディスク (LD) デバイス用のサービスの作成方法を示しています。

- LD CREATE KIT1 コマンドは，論理ディスクとして使用できる連続ファイル KIT1 を作成します。
- DIRECTORY KIT1 コマンドは KIT1 に関する情報を表示します。
- LD CONNECT KIT1 は，論理ディスク・ファイル KIT1 を論理ディスク・デバイス MOVMAN$LDA1: に接続します。

- INITIALIZE コマンドは，MOVMAN$LDA1: LD デバイスをフォーマットします。
- MOUNT コマンドは，LD デバイスを利用できるようにします。
- CREATE SERVICE コマンドは，_MOVMAN$LDA1 LD デバイス上に TEST_KIT_1 サービスを作成します。

## DELETE SERVICE

1 つまたは複数のサービスを削除します。

## フォーマット

DELETE SERVICE *serviceName [device-or-partitionName]*

## パラメータ

serviceName
LAN 上で使用されるサービスの名前です。サービス名は英数字とドル記号 ($) で構成されます。長さは最大で 255 文字です。ワイルドカードも許されています。

InfoServer ユーティリティでワイルドカードが使用できるところでは， OpenVMS で使用されるのと同じワイルドカードが使用できます。パーセント文字 (%) は正確に 1 文字に一致し，アスタリスク文字 (*) は 0 文字以上の文字列に一致します。

device-or-partitionName
デバイス名またはパーティション名は， LAN 上で使用される OpenVMS ディスク・デバイスまたはパーティションの名前です。入力するデバイス名またはパーティション名は，作成済みのものでなければなりません。

デバイス名とパーティション名について，以下に説明します。

- デバイス名

  LAN にサービスを提供するデバイスは， OpenVMS ディスク・デバイスです。InfoServer デバイス名を指定するときには， OpenVMS デバイス名を使用します。デバイス名は， SHOW SERVICES コマンドで表示される名前に完全に一致するか，またはワイルドカードを含むかのいずれかでなければならないことに注意してください。

  InfoServer ユーティリティでワイルドカードが使用できるところでは，OpenVMS で使用されるのと同じワイルドカードが使用できます。パーセント文字 (%) は正確に 1 文字に一致し，アスタリスク文字 (*) は 0 文字以上の文字列に一致します。

  ディスク指定はコロンで終わる必要があります。

- パーティション名

  パーティションは，ネットワークにサービスが提供されるコンテナ・ファイルです。したがって，デフォルトのファイル・タイプが.ESS$PARTITION の

OpenVMS ファイル名を持っています。デバイス名，ディレクトリ名，ファイル名を含むパーティション名の長さは，最大で 242 文字です。

パーティション名は選択するサービスを特定する場合に使用できます。

今回のバージョンでは，パーティションのサポートは制限されています。パーティションに分割されたハード・ドライブをサポートする場合には， LD デバイスを使用することをお勧めします。詳細は， DCL コマンドの LD HELP を参照してください。

## 修飾子

/CLASS=className
全体の LASTport Disk (LAD) 名前空間のサブセットを指定します。

クラス名の目的は，名前空間を分割して，クライアントに対して意味のある名前だけを示すことです。クラス名を使用すると， 2 つのサービスに同じ名前を付けて，互いに競合しないようにすることもできます。

たとえば，いくつかのクライアント・システムで使用される異なるオンディスク構造に対して，異なるクラス名をつけることができます。あるクライアント・システムでは SERVICEA/CLASS=ODS-2 を使用し，別のクライアント・システムでは SERVICEA/CLASS=ISO_9660 を使用します。サービス名は同じ (SERVICEA) ですが，クラス名は異なります。

使用するクラス名は，作成するサービスに接続するクライアント・システムに依存します。デフォルトのクラス名は ODS_2 です。たとえば， OpenVMS システムは， InfoServer デバイスをマウントするとき， ODS_2 名前空間を使用します。OpenVMS クライアントは， ODS_2 サービス・クラスに属しているサービスのみ利用できることに注意してください。

有効なクラス名は，次のとおりです。

| | |
|---|---|
| V2.0 | PCSA MS-DOS クライアントが理解する名前 |
| Unformatted | 仮想ディスクは形式を持っていません |
| MSDOS | MSDOS 仮想ディスク |
| ODS_2 | VMS 仮想ディスク |
| UNIX | UNIX 仮想ディスク |
| ISO_9660 | ISO 9660 CD 形式 |
| HIGH_SIERRA | MS-DOS CD 形式 |
| APPLE | Macintosh HFS 形式 |
| SUN | Sun 形式 |

/CONFIRM (**デフォルト**)
/NOCONFIRM
サービスを削除するときに確認が求められます。/NOCONFIRM を指定した場合でも，接続が残っている場合には，確認が求められます。

各サービスの削除操作の前に確認を行うかどうかを制御します。以下の応答が有効です。

```
YES      NO       QUIT
TRUE     FALSE    Ctrl/Z
1        0        ALL
    Return キー
```

**使用上の注意**:

- 単語で答える場合には，大文字と小文字を混在させて構いません。単語による応答は， 1 文字以上の英字に省略できます (たとえば， TRUE に対して， T， TR， TRU)。ただし，省略する場合には，一意である必要があります。
- 肯定的な応答は， YES， TRUE，および 1 です。否定的な応答は， NO， FALSE， 0，および Return キーを押すことです。
- QUIT の入力または Ctrl/Z の押下は，その時点でコマンドの処理を停止させたいことを意味します。
- ALL を入力して応答した場合には，コマンドは処理を続行して，その後，プロンプトは表示されなくなります。

/DISCONNECT
/NODISCONNECT (**デフォルト**)
接続されているセッションが残っているサービスを削除しようとしたときのデフォルトの確認プロンプト処理を変更します。サービスに接続しているセッションがあって/DISCONNECT 修飾子を指定していない場合には，サービスの削除に確認が求められます。

サービスを削除する際に確認を求められないようにするには，/NOCONFIRM 修飾子と/DISCONNECT 修飾子の両方を指定します。

## 例

```
$ SHOW SERVICES




   Service Name         [Service Class] Device or File
   -------------------- --------------- ------------------------
   HUDSON               [ODS-2]         _MOVERS$LDA1: [ 1 Connection]
   BAFFIN               [ODS-2]         _MOVERS$LDA1:
   FUNDY                [ODS-2]         _MOVERS$LDA1:
   3 services found.

$ DELETE SERVICE HUDSON
```

```
Service HUDSON has 1 session connected!
Delete service HUDSON [ODS-2] for _MOVERS$LDA1:? [N]:
```

最初のコマンドでは，1 つの接続中のセッションを含む，3 つのサービスを表示しています。2 番目のコマンドでは HUDSON サービスを削除しています。このコマンドは，HUDSON に 1 つのセッションが接続されていることを示すメッセージを表示し，そのサービスを削除してよいか確認するプロンプトを表示しています。

## EXIT

InfoServer の実行を終了します。 Ctrl/Z を押しても，終了することができます。

## フォーマット

EXIT

## HELP

InfoServer のオンライン・ヘルプを表示します。

ESS$INFOSERVER は， OpenVMS のアプリケーションとして実装された LASTport/Disk サーバのユーザ・インタフェースです。動作はハードウェア InfoServer 製品と似ていますが，完全に同じではありません。

## フォーマット

HELP *[topic]*

## パラメータ

topic
ヘルプを要求するトピックを指定します。

## 例

```
$ INFOSERVER HELP SHOW SESSIONS
```

このコマンドは， InfoServer の SHOW SESSIONS コマンドについてのヘルプを表示します。

## SAVE

現在アクティブなサービスのセットを，コマンド・プロシージャ内の一連のコマンドとして保存します。システムをリブートしたときにこのコマンド・プロシージャを実行すれば，現在のサービスのセットを再現することができます。

## フォーマット

SAVE *procedureName*

## パラメータ

procedureName
現在のサーバの状態を復活させるためのコマンド・プロシージャを作成します。プロシージャ名は，作成するコマンド・プロシージャの OpenVMS ファイル名です。ファイル・タイプを指定しない場合には，デフォルトで.COM になります。

デフォルトのプロシージャ名は ESS$LAD_SERVICES.COM です。

## 例

```
$ SHOW SERVICES


    Service Name         [Service Class] Device or File
    -------------------- --------------- -------------
    BASELEVEL_A          [ODS-2]         _INFOS$LDA1:
    BASELEVEL_B          [ODS-2]         _INFOS$LDA2:
    BASELEVEL_C          [ODS-2]         _INFOS$LDA3:
    BASELEVEL_D          [ODS-2]         _INFOS$LDA4:
    FIELD_TEST_BASELEVEL [ODS-2]         _INFOS$LDA2:
    CURRENT_BASELEVEL    [ODS-2]         _INFOS$LDA3:
    EXPERIMENTAL_BASELEVEL
                         [ODS-2]         _INFOS$LDA4:
    %INFOSRVR-I-FOUND, 7 services found.

$ SAVE BASELEVELS
```

```
$! Created by the OpenVMS InfoServer SAVE command on 22-APR-2005
14:34:02.48
$ Set NoOn
$ Infoserver := $ESS$INFOSERVER
$!
$! The comment for each service includes the current device name.
$!
$!****************************************************************
$!  BASELEVEL_A [ODS_2] - _BILBO$LDA1: 1
$!****************************************************************
$ LD Connect/Symbol _BILBO$DKB0:[DISKS]BASELEVEL_A.DSK;1 2
$ LD_UNIT_1 := LDA'LD_UNIT': 3
$ If $STATUS Then Mount/System/NoWrite 'LD_UNIT_1' BASELEVELA 4
 $ INFOSERVER Create Service BASELEVEL_A 'LD_UNIT_1' - 5
        /Class=ODS_2/Readers=1000/NoWriters -
        /Readahead/NoReadbehind -
        /Rating=Dynamic
$!****************************************************************
$!  BASELEVEL_B [ODS_2] - _BILBO$LDA2:
$!****************************************************************
$ LD Connect/Symbol _BILBO$DKB0:[DISKS]BASELEVEL_B.DSK;1
$ LD_UNIT_2 := LDA'LD_UNIT':
$ If $STATUS Then Mount/System/NoWrite 'LD_UNIT_2' BASELEVELB
$ INFOSERVER Create Service BASELEVEL_B 'LD_UNIT_2' -
        /Class=ODS_2/Readers=1000/NoWriters -
        /Readahead/NoReadbehind -
        /Rating=Dynamic
$!****************************************************************
$!  BASELEVEL_C [ODS_2] - _BILBO$LDA3:
$!****************************************************************
$ LD Connect/Symbol _BILBO$DKB0:[DISKS]BASELEVEL_C.DSK;1
$ LD_UNIT_3 := LDA'LD_UNIT':
$ If $STATUS Then Mount/System/NoWrite 'LD_UNIT_3' BASELEVELC
$ INFOSERVER Create Service BASELEVEL_C 'LD_UNIT_3' -
        /Class=ODS_2/Readers=1000/NoWriters -
        /Readahead/NoReadbehind -
        /Rating=Dynamic
$!****************************************************************
$!  BASELEVEL_D [ODS_2] - _BILBO$LDA4:
$!****************************************************************
$ LD Connect/Symbol _BILBO$DKB0:[DISKS]BASELEVEL_D.DSK;1
$ LD_UNIT_4 := LDA'LD_UNIT':
$ If $STATUS Then Mount/System/NoWrite 'LD_UNIT_4' BASELEVELD
$ INFOSERVER Create Service BASELEVEL_D 'LD_UNIT_4' -
        /Class=ODS_2/Readers=1000/NoWriters -
        /Readahead/NoReadbehind -
        /Rating=Dynamic -
        /Encoded_Password=481C6B9081E742C2
           ! Invalid if service name changes 6
$!****************************************************************
$!  FIELD_TEST_BASELEVEL [ODS_2] - _BILBO$LDA2:
$!****************************************************************
$ INFOSERVER Create Service FIELD_TEST_BASELEVEL 'LD_UNIT_2' - 7
        /Class=ODS_2/Readers=1000/NoWriters -
```

```
        /Readahead/NoReadbehind -
        /Rating=Dynamic
$!*****************************************************************
$ INFOSERVER Create Service CURRENT_BASELEVEL 'LD_UNIT_3' -
        /Class=ODS_2/Readers=1000/NoWriters -
        /Readahead/NoReadbehind -
        /Rating=Dynamic
$!*****************************************************************
$!  EXPERIMENTAL_BASELEVEL [ODS_2] - _BILBO$LDA4:
$!*****************************************************************
$ INFOSERVER Create Service EXPERIMENTAL_BASELEVEL 'LD_UNIT_4' -
        /Class=ODS_2/Readers=1000/NoWriters -
        /Readahead/NoReadbehind -
        /Rating=Dynamic -
        /Encoded_Password=01F1D7374C0B81EC
          ! Invalid if service name changes 8
$ Exit
```

この例の SHOW SERVICES コマンドは，現在このサーバが提供しているサービスを表示します。ここには一連のソフトウェア・ベースレベルがあり，それぞれの論理ディスクと LAN に提供しているサービスが示されています。ベースレベルには a ～ d のラベルが付けられていますが，対応する英字をユーザが覚えなくてもいいように名前も付けられています。

デバイス LDA2， LDA3， LDA4 にはそれぞれ 2 つのサービスが割り当てられていることに注意してください。

例の中の数字は，以下の説明の番号に対応しています。

1 各デバイスのコメントには， SAVE コマンドを実行したときのデバイス名が含まれています。 LD デバイスは擬似ディスク・デバイスであり，ユニット番号は接続するたびに変わります。

2 このコマンドは， LD デバイスをコンテナ・ファイルに接続して，ユニット番号を DCL のシンボル LD_UNIT に代入します。

3 コンテナ・ファイルに割り当てられる各デバイスに対して，一意のシンボルが作成されます。

4 このコマンドは， SAVE コマンドの実行時にデバイスに付いていたボリューム・ラベルを指定して，デバイスをマウントします。

5 当該デバイスに対する InfoServer サービスが再作成されます。

6 EXPERIMENTAL_BASELEVEL サービスはパスワードで保護されています。セキュリティを確保するために，パスワードは暗号化された形式でコマンド・プロシージャ内に格納されています。 2 つのサービスは同じパスワードを持っていますが，暗号化されたパスワードは異なっていることに注意してください。

7 FIELD_TEST_BASELEVEL と BASELEVEL_B は同じ LD デバイスを指しているため，別のデバイスは作成されず，作成済みのユニットを参照するために，正しいユニット (シンボルは LD_UNIT_2) が使用されます。

**8** 6 番の説明を参照してください。

## SET SERVICE

既存のサービスの属性を変更します。

## フォーマット

SET SERVICE　*serviceName [device-or-partitionName]*

## パラメータ

serviceName
LAN 上で使用されるサービスの名前です。サービス名は英数字とドル記号 ($) で構成されます。長さは最大で 255 文字です。

device-or-partitionName
LAN 上で使用される OpenVMS のディスク・デバイスまたはパーティションの名前です。入力するデバイス名またはパーティション名は，作成済みのものでなければなりません。

デバイス名とパーティション名について，以下に説明します。

- デバイス名

  LAN にサービスを提供するデバイスは， OpenVMS ディスク・デバイスです。InfoServer デバイス名を指定するときには， OpenVMS デバイス名を使用します。デバイス名は， SHOW SERVICES コマンドで表示される名前に完全に一致するか，またはワイルドカードを含むかのいずれかでなければならないことに注意してください。

  InfoServer ユーティリティでワイルドカードが使用できるところでは，OpenVMS で使用されるのと同じワイルドカードが使用できます。パーセント文字 (%) は正確に 1 文字に一致し，アスタリスク文字 (*) は 0 文字以上の文字列に一致します。

  ディスク指定はコロンで終わる必要があります。

- パーティション名

  パーティションは，ネットワークにサービスが提供されるコンテナ・ファイルです。したがって，デフォルトのファイル・タイプが.ESS$PARTITION の OpenVMS ファイル名を持っています。デバイス名，ディレクトリ名，ファイル名を含むパーティション名の長さは，最大で 242 文字です。

  パーティション名は選択するサービスを特定する場合に使用できます。

今回のバージョンでは，パーティションのサポートは制限されています。パーティションに分割されたハード・ドライブをサポートする場合には， LD デバイスを使用することをお勧めします。詳細は， DCL コマンドの LD HELP を参照してください。

## 修飾子

/CLASS=className
全体の LASTport Disk (LAD) 名前空間のサブセットを指定します。

クラス名の目的は，名前空間を分割して，クライアントに対して意味のある名前だけを示すことです。クラス名を使用すると， 2 つのサービスに同じ名前を付けて，互いに競合しないようにすることもできます。

たとえば，いくつかのクライアント・システムで使用される異なるオンディスク構造に対して，異なるクラス名をつけることができます。あるクライアント・システムでは SERVICEA/CLASS=ODS-2 を使用し，別のクライアント・システムでは SERVICEA/CLASS=ISO_9660 を使用します。サービス名は同じ (SERVICEA) ですが，クラス名は異なります。

使用するクラス名は，作成するサービスに接続するクライアント・システムに依存します。デフォルトのクラス名は ODS_2 です。たとえば， OpenVMS システムは， InfoServer デバイスをマウントするとき， ODS_2 名前空間を使用します。OpenVMS クライアントは， ODS_2 サービス・クラスに属しているサービスのみ利用できることに注意してください。

有効なクラス名は，次のとおりです。

| | |
|---|---|
| V2.0 | PCSA MS-DOS クライアントが理解する名前 |
| Unformatted | 仮想ディスクは形式を持っていません |
| MSDOS | MSDOS 仮想ディスク |
| ODS_2 | VMS 仮想ディスク |
| UNIX | UNIX 仮想ディスク |
| ISO_9660 | ISO 9660 CD 形式 |
| HIGH_SIERRA | MS-DOS CD 形式 |
| APPLE | Macintosh HFS 形式 |
| SUN | Sun 形式 |

/PASSWORD=passwordString
/NOPASSWORD
オプションのサービス・アクセス制御パスワードを指定します。パスワードが設定されたサービスにクライアント・システムがアクセスするためには，パスワードを指定する必要があります。

パスワードは最大 39 文字の英数字文字列です。パスワードを指定しなかった場合には，クライアント・システムではこのサービスにアクセスする場合に，パスワードの入力を求められません。

テキストのパスワードは暗号化された形式で，他のサービス情報とともにメモリに保存されます。

/RATING=DYNAMIC
/RATING=STATIC=value
該当するサービスが複数ある場合には，クライアントはサービス評価点を使用してサービスを選択します。最高のサービス評価点を持ったサービスが選択されます。

システムは動的サービス評価点を負荷に応じて調整します。

静的評価点を 0 〜 65535 の範囲で指定することもできます。システムは静的評価点は調整しません。

/READAHEAD
/NOREADAHEAD
キャッシュ・ブロックを埋めるためのディスクの読み取りが必要になったときに，/READAHEAD 修飾子は，読み取り範囲が，要求された最初のブロックから，バケット境界の最後までであることを指示します。先読みすることによって，必要なディスク・ブロックが前もってキャッシュにロードされるので，シーケンシャル操作の速度が向上します。

/READAHEAD 修飾子と/READBEHIND 修飾子の両方を指定しておくと，キャッシュ・バケット内のブロックが要求されたときに，そのブロックのバケット範囲全体がキャッシュに読み込まれます。

/READBEHIND
/NOREADBEHIND
キャッシュ・ブロックを埋めるためのディスクの読み取りが必要になったときに，/READBEHIND 修飾子は，読み取り範囲が，キャッシュ・バケット境界の先頭から要求されたブロックまでであることを指示します。

/READAHEAD 修飾子と/READBEHIND 修飾子の両方を指定すると，キャッシュ・バケット内のブロックが要求されたときに，そのブロックのバケット範囲全体がキャッシュに読み込まれます。

/READERS=number
クライアントの読み取りアクセスで許される同時接続の最大数を指定します。

## 例

```
$ INFOSERVER SET SERVICE FUNDY/NOPASSWORD



    Service FUNDY [ODS-2] modified.

$ INFOSERVER SHOW SERVICES FUNDY/FULL

    FUNDY [ODS-2]                          Access: Read-only
      File or device: _MOVERS$LDA1: [750000 blocks]
      Flags: 00000000D2 {No Writers,Static Rating,Readbehind,Readahead}
      Rating:        Static,    42         Password:         Disabled
      Max Readers:             1000        Max Writers:             0
      Curr Readers:               0        Curr Writers:            0
      Reads:                      0        Writes:                  0
      Blocks Read:                0        Blocks Written:          0
```

この例の最初のコマンドは FUNDY サービスを変更して，クライアントがサービスにアクセスするときにパスワードを入力しなくてすむようにします。2 番目のコマンドは FUNDY サービスを表示します。パスワードの使用が無効になったことが示されています (SHOW SERVICES コマンドの例では， FUNDY サービスでパスワードの使用が有効になっていたことに注意してください)。

## SHOW SERVER

サーバ (すなわち，サービスを提供するシステム) に関する情報を表示します。

## フォーマット

SHOW SERVER

## 例

```
$ INFOSERVER SHOW SERVER


    Node MOVERS [COMPAQ Professional Workstation XP1000] running OpenVMS XALD-BL2

    LASTport/Disk Server Version 1.2

    Max Services:            64              Write Quota:          0
    Cache Buckets:         4096              Cache Bucket Size:   32 blocks
    Cache Size:        67108864 bytes
    Hits:                  478               Hit Percentage:      59%
    Misses:                328

    Current Sessions:        0               Peak Sessions:          1

                           Read                    Write
    Requests:               40                       0
    Blocks:                319                       0
    Errors:                  0                       0
    Aborted:                 0                       0
    Conflicts:               0                       0
```

このコマンドは，クライアントにサービスを提供しているサーバに関する情報を表示します。表示される情報には，以下のものがあります。

- このサーバが同時に提供できるサービスの最大数
- 現在のキャッシュ・サイズ
- キャッシュの有効度についての統計情報
- 現在の同時接続クライアント数と過去の最大の同時接続クライアント数
- I/O 統計情報

# SHOW SERVICES

SHOW SERVICES コマンドは，サーバが提供している 1 つまたはすべてのサービスについて，サービス固有の情報を表示します。この情報には，サービスに関連付けられたデバイス番号と，接続されているセッション数が含まれます。

SHOW SERVICES コマンドでは，ワイルドカード表現が使用できます。 InfoServer ユーティリティでワイルドカードが使用できるところでは， OpenVMS で使用されるのと同じワイルドカードが使用できます。パーセント文字 (%) は正確に 1 文字に一致し，アスタリスク文字 (*) は 0 文字以上の文字列に一致します。

## フォーマット

SHOW SERVICES *[serviceName] [options...]*

## パラメータ

serviceName
LAN 上で使用されるサービスの名前です。サービス名は英数字とドル記号 ($) で構成されます。長さは最大で 255 文字です。省略した場合には，サービス名のデフォルトは全サービスです。

InfoServer ユーティリティでワイルドカードが使用できるところでは， OpenVMS で使用されるのと同じワイルドカードが使用できます。パーセント文字 (%) は正確に 1 文字に一致し，アスタリスク文字 (*) は 0 文字以上の文字列に一致します。

## 修飾子

/BRIEF (**デフォルト**)
BRIEF オプションでは，選択したそれぞれのサービスについて，省略されたオンライン・サマリ情報が表示されます。 BRIEF がデフォルトです。

/FULL
FULL オプションでは，選択したそれぞれのサービスについて，サービス固有のすべての情報が表示されます。

## 例

1. INFOSERVER> SHOW SERVICES

```
Service Name         [Service Class] Device or File
------------------- --------------- ------------------------
HUDSON                [ODS-2]         _MOVERS$LDA1:
BAFFIN                [ODS-2]         _MOVERS$LDA1:
FUNDY                 [ODS-2]         _MOVERS$LDA1:
3 services found.
```

このコマンドは，接続中のすべてのサービスについて，デフォルトの BRIEF オンライン・サマリを表示しています。

2. INFOSERVER> SHOW SERVICES/FULL

```
HUDSON [ODS-2]                          Access: Read-only
  File or device: _MOVERS$LDA1: [750000 blocks]
  Flags: 0000000082 {No Writers,Readahead}
  Rating:       Dynamic, 65535        Password:         Disabled
  Max Readers:           1000         Max Writers:            0
  Curr Readers:             0         Curr Writers:           0
  Reads:                    0         Writes:                 0
  Blocks Read:              0         Blocks Written:         0

BAFFIN [ODS-2]                          Access: Read-only
  File or device: _MOVERS$LDA1: [750000 blocks]
  Flags: 0000000082 {No Writers,Readahead}
  Rating:       Dynamic, 65535        Password:         Disabled
  Max Readers:           1000         Max Writers:            0
  Curr Readers:             0         Curr Writers:           0
  Reads:                    0         Writes:                 0
  Blocks Read:              0         Blocks Written:         0

FUNDY [ODS-2]                           Access: Read-only
  File or device: _MOVERS$LDA1: [750000 blocks]
  Flags: 00000000D2 {No Writers,Static Rating,Readbehind,Readahead}
  Rating:       Static,    42         Password:         Enabled
  Max Readers:           1000         Max Writers:            0
  Curr Readers:             0         Curr Writers:           0
  Reads:                    0         Writes:                 0
  Blocks Read:              0         Blocks Written:         0
3 services found.
```

このコマンドは，接続中のすべてのサービスについて，サービス固有のすべての情報を表示しています。 HUDSON サービスと BAFFIN サービスではパスワード

が無効で， FUNDY サービスでは有効になっていることに注意してください。

## SHOW SESSIONS

サービスに接続されているクライアント・ノードに関する情報を表示します。

## フォーマット

SHOW SESSIONS *[serviceName] [device-or-partitionName]]*

## パラメータ

serviceName
LAN 上で使用されるサービスの名前です。サービス名は英数字とドル記号 ($)，およびワイルドカードで構成されます。長さは最大で 255 文字です。省略した場合には，サービス名のデフォルトはすべてのサービスです。

InfoServer ユーティリティでワイルドカードが使用できるところでは， OpenVMS で使用されるのと同じワイルドカードが使用できます。パーセント文字 (%) は正確に 1 文字に一致し，アスタリスク文字 (*) は 0 文字以上の文字列に一致します。

device-or-partitionName
デバイス名またはパーティション名は， LAN 上で使用される OpenVMS ディスク・デバイスまたはパーティションの名前です。入力するデバイス名またはパーティション名は，作成済みのものでなければなりません。

デバイス名とパーティション名について，以下に説明します。

- デバイス名

  LAN にサービスを提供するデバイスは， OpenVMS ディスク・デバイスです。InfoServer デバイス名を指定するときには， OpenVMS デバイス名を使用します。デバイス名は， SHOW SERVICES コマンドで表示される名前に完全に一致するか，またはワイルドカードを含むかのいずれかでなければならないことに注意してください。

  InfoServer ユーティリティでワイルドカードが使用できるところでは，OpenVMS で使用されるのと同じワイルドカードが使用できます。パーセント文字 (%) は正確に 1 文字に一致し，アスタリスク文字 (*) は 0 文字以上の文字列に一致します。

  ディスク指定はコロンで終わる必要があります。

- パーティション名

パーティションは，ネットワークにサービスが提供されるコンテナ・ファイルです。したがって，デフォルトのファイル・タイプが.ESS$PARTITION の OpenVMS ファイル名を持っています。デバイス名，ディレクトリ名，ファイル名を含むパーティション名の長さは，最大で 242 文字です。

今回のバージョンでは，パーティションのサポートは制限されています。パーティションに分割されたハード・ドライブをサポートする場合には， LD デバイスを使用することをお勧めします。詳細は， DCL コマンドの LD HELP を参照してください。

## 修飾子

/ALL
クライアントが接続していない場合でも，選択基準に該当するすべてのサービスを表示します。この修飾子を省略した場合には，クライアントが接続しているサービスだけが表示されます。

## 例

1. $ INFOSERVER SHOW SESSIONS

```
HUDSON              [ODS-2]         _MOVERS$LDA1: [ 1 Connection]
1 service found.
```

2. $ INFOSERVER SHOW SESSIONS/ALL

```
HUDSON              [ODS-2]         _MOVERS$LDA1: [ 1 Connection]
BAFFIN              [ODS-2]         _MOVERS$LDA1:
FUNDY               [ODS-2]         _MOVERS$LDA1:
3 services found.
```

最初の例では，このコマンドはクライアント接続 HUDSON を持つサービスだけを表示しています。 2 番目の例では，このコマンドは，クライアントが接続されていないサービスも含め，すべてのセッションを表示しています。

## SPAWN

プロセスを生成して DCL コマンドを実行します。コマンドを指定しなかった場合には，生成したプロセスにコマンド端末が接続されます。コマンドを指定した場合には，そのコマンドが実行され，コマンドが完了すると，制御が親プロセスに戻ります。

## フォーマット

SPAWN *[DCL Command]*

## 例

```
InfoServer> SPAWN DIRECTORY

        .
        .
        .
    (output)
        .
        .
        .

InfoServer>
```

このコマンドは，プロセスを生成して DCL コマンドの DIRECTORY を実行します。コマンドの実行が完了すると，制御が InfoServer プロセスに戻ります。

## START SERVER

このコマンドは LASTport/Disk サーバを起動し，各種のサーバ特性とキャッシュ特性を設定します。

通常このコマンドは， SYS$STARTUP:ESS$LAD_STARTUP.DAT のデータを使用して， SYS$STARTUP:ESS$LAD_STARTUP.COM から実行されます。すべての変更は SYS$STARTUP:ESS$LAD_STARTUP.DAT ファイル内で行うことをお勧めします。

現在サービスが何も定義されていない場合には， START SERVER コマンドを会話型で使用して，修飾子を指定してサーバの設定を変更することもできます。

## フォーマット

START SERVER

## 修飾子

/BUFFER_SIZE=n
InfoServer のブロック・キャッシュは固定長バッファ (**バケット**とも呼ばれる) の配列として構成されます。/BUFFER_SIZE 修飾子では各バケットのサイズを指定します (/CACHE 修飾子では，バケット数を指定します)。

このパラメータの数値は， 3 ～ 8 の範囲の整数値です。それぞれの整数値は，次のように， 512 バイトのブロックを単位としたバケット・サイズを表わします。

```
3 -   8 ブロック (デフォルト)
4 -  16 ブロック
5 -  32 ブロック
6 -  64 ブロック
7 - 128 ブロック
8 - 256 ブロック
```

32 ブロックより大きなバケット・サイズは，多くの場合，適切ではありません。 OpenVMS クライアントでは， 31 ブロックより大きな入出力要求を 31 ブロックのかたまりにセグメント化するため，デフォルトのバケット先読み動作では，ディスクに対する不要な入出力動作が発生するからです。

/CACHE = number-of-buckets (**デフォルト**= 512)
InfoServer のブロック・キャッシュは固定長バッファ (**バケット**とも呼ばれる) の配列として構成されます。/CACHE 修飾子では，キャッシュのバケット数を指定します。 (/BUFFER_SIZE 修飾子では各バケットのサイズを指定します。)

16384 より大きな数を指定すると，性能に悪影響を与える可能性があります。必要な大きさのキャッシュを確保するためには，/BUFFER_SIZE 修飾子の値を増加させることを検討してください。

/MAXIMUM_SERVICES = maxservice (**デフォルト**= 256)
サーバの最大サービス数を設定します。これが同時に定義できるサービスの最大数になります。各サービス記述子はページングされないプールを消費します。ただし，未使用のサービス・スロットは 4 バイトしか消費しません。

最大の値は 1024 です。

/WRITE_QUOTA = n (**デフォルト**= 0)
サーバで許される同時非同期書き込みの数です。デフォルトの 0 は，すべての書き込み操作が同期して行われることを意味します。

## 例

```
$ InfoServer SHOW SERVER



    Node BILBO [HP rx2600  (900MHz/1.5MB)] running OpenVMS XAR8-D2Y
    LASTport/Disk Server Version 1.2

    Max Services:        64        Write Quota:          0
    Cache Buckets:     2048        Cache Bucket Size:   32 blocks
    Cache Size:    33554432 bytes
    Hits:                0         Hit Percentage:       0%
    Misses:              0

    Current Sessions:    0         Peak Sessions:        0

                       Read              Write
    Requests:             0                 0
    Blocks:               0                 0
    Errors:               0                 0
    Aborted:              0                 0
    Conflicts:            0                 0

$ InfoServer START SERVER/MAXIMUM_SERVICES=128/CACHE=2048/BUFF=5/WRITE=0

    %INFOSRVR-I-STARTED, LASTport/Disk server started.

$ InfoServer SHOW SERVER

    Node BILBO [HP rx2600  (900MHz/1.5MB)] running OpenVMS XAR8-D2Y
    LASTport/Disk Server Version 1.2
```

```
Max Services:    128        Write Quota:          0
Cache Buckets:  2048        Cache Bucket Size:   32 blocks
Cache Size: 33554432 bytes
Hits:               0       Hit Percentage:       0%
Misses:             0

Current Sessions:  0        Peak Sessions:        0

                  Read            Write
Requests:            0              0
Blocks:              0              0
Errors:              0              0
Aborted:             0              0
Conflicts:           0              0
```

この例の最初のコマンドは，サーバに関して現在の情報を表示します。2 番目のコマンドは，サーバを起動し，そのサーバのサービスの最大数を増加させます。3 番目のコマンドはサーバに関して新しい情報を表示します。この表示で，サービスの最大数が増加していることが分かります。

# 7

# 関連製品の機能

この章では，OpenVMS オペレーティング・システムの関連製品の主な新機能について説明します。OpenVMS 関連製品の一覧とディレクトリ情報については，使用しているオペレーティング・システムに対応する『Read Before Installing』を参照してください。

## 7.1 Distributed NetBeans for OpenVMS

Distributed NetBeans for OpenVMS を使用すると，NetBeans IDE をデスクトップ・システム上で実行し，リモートの OpenVMS Alpha システムおよび OpenVMS I64 システム用のアプリケーションを開発することができます。

Distributed NetBeans Version 1.1 には，NetBeans for OpenVMS プラグイン・モジュールで提供されているすべての機能 (C/C++，COBOL，FORTRAN，および PASCAL 言語のサポート，および MMS，BASH，DCL，CMS，および EDT キーパッドのサポート) が含まれています。Distributed NetBeans は，組み込み FTP ファイル・システムによるファイルおよび CMS ライブラリ (CMS グループを含む) へのアクセスと，Samba for OpenVMS および Advanced Server へのアクセスをサポートしています。

Distributed NetBeans for OpenVMS についての詳細と，最新のキットおよびドキュメントをダウンロードする方法については，次の Web サイトを参照してください。

```
http://www.hp.com/products/openvms/distributednetbeans/
```

## 7.2 Secure Web Browser for OpenVMS

Secure Web Browser Version 1.7-13 for OpenVMS は，Mozilla M1.7.13 を基にしており，重要なセキュリティ・バグ修正が含まれています。

Secure Web Browser for OpenVMS についての詳細と，最新のキットおよびドキュメントをダウンロードする方法については，次の Web サイトを参照してください。

```
http://www.hp.com/products/openvms/securewebbrowser/
```

## 7.3 Secure Web Server for OpenVMS

Secure Web Server Version 2.1 for OpenVMS は， Apache 2.0.52 を基にしています。新しい機能としては， suEXEC および mod_dav のサポートと， Version 2.0 での STREAM_LF の制限の解除があります。

また，新しいバージョンの PHP (CSWS_PHP Version 1.3)， mod_perl (CSWS_PERL Version 2.1)， Perl (PERL Version 5.8-6)，および Tomcat (CSWS_JAVA Version 3.0) が使用可能です。

Secure Web Server for OpenVMS についての詳細と，最新のキットおよびドキュメントをダウンロードする方法については，次の Web サイトを参照してください。

```
http://www.hp.com/products/openvms/securewebserver/
```

## 7.4 HP TCP/IP Services for OpenVMS Version 5.6

HP TCP/IP Services for OpenVMS Version 5.6 は， OpenVMS Version 8.3 をサポートしています。 TCP/IP Version 5.6 では，以下の機能が追加されています。

- BIND 9 リゾルバ

  新しいバージョンの BIND リゾルバでは， IPv6 トランスポートを使用して DNS エントリを解決できるようになっています。これは， BIND 8 に基づくリゾルバ機能が提供されていた， V5.5 およびその他の最近のリリースからのメジャー・アップグレードです。

- DNS/BIND Version 9.3.1 サーバ

  新しいバージョンの BIND サーバでは，セキュリティと安定性に関する継続的な改良が行われています。

- NFS クライアントの TCP サポート

  NFS クライアントの TCP サポートが追加され， NFS クライアントとサーバが，従来の UDP モードでの動作に加えて， TCP でも動作できるようになりました。この機能は， WAN (Wide Area Network) をまたいでファイル・システムをマウントする場合や，ファイアウォールを超えてファイル・システムをマウントする場合に便利です。

- Integrity サーバ用の NFS サーバのサポート

  OpenVMS Integrity サーバ・プラットフォーム用の NFS サーバが追加されました。

- NFS でのシンボリック・リンクのサポート

  NFS サーバがシンボリック・リンクを認識し，必要に応じてシンボリック・リンクを作成できるようになりました。

- NTP のセキュリティ更新 (SSL)

新しい NTP の機能では，暗号化によるセキュリティが提供され，システム・クロックを狂わせようとする攻撃に対する保護が強化されました。

- SMTP におけるゾーン内での複数ドメイン

  SMTP は，直接のローカル配信で，複数のドメイン名を認識するようになりました。

- SSH での Kerberos のサポート

  TCP/IP Services Version 5.6 for OpenVMS では， SSH での Kerberos のサポートが追加されています。 Kerberos は，マサチューセッツ工科大学で開発されたネットワーク認証プロトコルです。 SSH のパスワード認証方式が拡張され，Kerberos をサポートするようになりました。 Kerberos に基づく以下の 3 つの新しい SSH 認証方式がサポートされるようになりました。

  — gssapi-with-mic

  — kerberos-2@ssh.com

  — kerberos-tgt-2@ssh.com

  Kerberos についての詳細は，『HP Open Source Security for OpenVMS, Volume 3: Kerberos』を参照してください。

- TELNET での Kerberos のサポート

  TELNET サーバとクライアントでは， OpenVMS Version 8.3 で提供されるアップグレードされた Kerberos に対するサポートが追加されました。

- TELNET サーバのデバイス上限

  TELNET サーバで，セッションまたは TN デバイスが 9999 個までという制限がなくなりました。

- LPD および TELNETSYM での IPv6 のサポート

  プリント・ソフトウェア LPD と TELNETSYM では， IPv6 トランスポートを使用した印刷が可能となりました。

- OpenVMS Plus Mode での FTP 性能の向上

  特に，サーバとクライアントがどちらも OpenVMS システムの場合に， FTP サービスの性能が向上しました。

- TCPIP$CONFIG でのインタフェース設定の改良

  ローカル・インタフェースと IP アドレスを定義する際のメニュー方式の処理が大幅に見直され， failSAFE IP のサポートが強化されました。

- Encryption for OpenVMS は OpenVMS のインストールの一部としてインストール

  ローカル・インタフェースと IP アドレスを定義する際のメニュー方式の処理が大幅に見直され， failSAFE IP のサポートが強化されました。

## 7.5 Web Services Integration Toolkit for OpenVMS

Web Services Integration Toolkit Version 1.1 for OpenVMS は，旧来のアプリケーション・ロジックを公開する JavaBean を開発する際に大いに役立つ一連のツールを提供します。これらのツールは，個別に使用することも組み合わせて使用することもできるように設計されています。

Web Services Integration Toolkit for OpenVMS は，以下の作業を支援するツールを提供します。

- XML IDL ファイルの作成

  公開するインタフェースを記述した XML のインタフェース定義ファイル (IDL) を作成します。

- コンポーネントの生成

  WSIT サーバ・インタフェース・ラッパと WSIT JavaBean を生成します。また，オプションで Java®クライアントまたは Java Server Page (JSP) クライアントを生成します。

- 生成したコードの使用

  生成した WSIT JavaBean を， BEA Web Logic Server， Apache Axis， Java Message Service， Java Remote Method Invocation， Java Enterprise Edition (Java EE)，別の JavaBean のうち，選択した技術から呼び出します。コマンド行インタフェースで Java クライアントを使用するか， Web ブラウザで JSP クライアントを使用します。

- 既存の BridgeWorks 接続を XML IDL ファイルに変換

  既存の BridgeWorks 接続を XML IDL ファイルに変換し， Web Services Integration Toolkit で使用することもできます。

Web Services Integration Toolkit for OpenVMS についての詳細と，最新のキットおよびドキュメントをダウンロードする方法については，次の Web サイトを参照してください。

```
http://www.hp.com/products/openvms/wsit/
```

# 第2部
# OpenVMSの英語版ドキュメント

ここではOpenVMSの英語版ドキュメントについて説明します。日本語ドキュメントについては,『日本語HP OpenVMSリリース・ノート』を参照してください。

# 8

# OpenVMS 英語版ドキュメントの概要

OpenVMS Version 8.3 で改訂されたドキュメントおよび新規に追加されたドキュメントは以下のとおりです。

- 改訂されたドキュメント:
  - 『HP OpenVMS Availability Manager User's Guide』
  - 『HP C Run-Time Library Reference Manual for OpenVMS Systems』
  - 『HP OpenVMS DCL Dictionary: A-M』
  - 『HP OpenVMS DCL Dictionary: N-Z』
  - 『HP OpenVMS Delta/XDelta Debugger Manual』
  - 『OpenVMS Linker Utility Manual』
  - 『HP OpenVMS Management Station Overview and Release Notes』
  - 『Open Source Security for OpenVMS Alpha, Volume 1: Common Data Security Architecture』
  - 『HP Open Source Security for OpenVMS, Volume 2: HP SSL for OpenVMS』
  - 『HP Open Source Security for OpenVMS, Volume 3: Kerberos』
  - 『HP OpenVMS System Management Utilities Reference Manual: A-L』
  - 『HP OpenVMS System Management Utilities Reference Manual: M-Z』
  - 『OpenVMS System Services Reference Manual: A–GETUAI』
  - 『OpenVMS System Services Reference Manual: GETUTC–Z』
  - 『HP OpenVMS Utility Routines Manual』
- 新しいドキュメント:
  - 『HP OpenVMS Version 8.3 Upgrade and Installation Manual』
  - 『HP OpenVMS Version 8.3 New Features and Documentation Overview』
  - 『HP OpenVMS Version 8.3 Release Notes』
  - 『Guide to HP OpenVMS Version 8.3 Media』

# 9

# OpenVMS の英語版ドキュメント (印刷およびオンライン)

OpenVMS のドキュメントは次の形式で提供されます。

- 印刷物

  印刷されたドキュメントが必要な場合は，ほとんどの OpenVMS ドキュメントをこの形式で購入できます。個々の印刷マニュアルを個別に購入することはできませんが， OpenVMS 印刷ドキュメント・キットにはすべてのマニュアルが含まれています。ただし『Porting Applications from HP OpenVMS Alpha to OpenVMS I64 for Integrity Servers』の印刷版は単独で購入可能です。

- CD に収録されたオンライン・ドキュメント

  すべての OpenVMS ドキュメントは CD に収録されたオンライン形式で提供され，多くの関連製品に関するドキュメントも含まれています。ドキュメンテーション CD は OpenVMS メディア・キットに同梱されています。

- OpenVMS Web サイトで提供されるオンライン・ドキュメント

  OpenVMS のドキュメントは，アーカイブされたドキュメントを含めて，OpenVMS Web サイトで閲覧できます。

- オンライン・ヘルプ

  タスク関連情報が必要な場合は， OpenVMS コマンド，ユーティリティ，システム・ルーチンに関するオンライン・ヘルプを簡単に表示できます。

ここでは， OpenVMS ドキュメントが提供される形式と，各形式で提供されるドキュメントの名称を示します。

## 9.1 印刷ドキュメント

OpenVMS メディア・キット (インストレーション・キット) には，一部のドキュメントの印刷版が同梱されています。それ以外の印刷ドキュメントは別売のドキュメント・キットで注文可能です。ここでは， OpenVMS の印刷ドキュメントについて次のカテゴリに分類して説明します。

- メディア・キット
- ドキュメンテーション・セット: 基本セットとフル・セット，およびオペレーティング環境拡張
- システム統合製品

- アーカイブされたドキュメント

### 9.1.1 OpenVMS メディア・キットのドキュメント

OpenVMS Alpha および OpenVMS I64 システムのメディア・キット (インストレーション・キット) には，OpenVMS オペレーティング・システムの最新のバージョンを使用するのに必要なドキュメントが同梱されています。表 9–1 には，OpenVMS メディア・キットに含まれているドキュメントを示しています。提供されるドキュメントは，新規カスタマであるのか，サービス・カスタマであるのかに応じて異なります。新規カスタマにはすべてのドキュメントが提供されます。サービス・カスタマには，前回のリリース以降に更新されたドキュメントと新しいドキュメントだけが提供されます。

**注意**

**『OpenVMS License Management Utility Manual』，『Guide to HP OpenVMS Version 8.3 Media』および『HP OpenVMS Version 8.3 Upgrade and Installation Manual』は，メディア・キットでのみ提供されます。第 9.1.2 項で説明する OpenVMS フル・ドキュメント・セットには含まれていません。**

**表 9–1 OpenVMS メディア・キットに含まれるドキュメント**

| ドキュメント | |
|---|---|
| 『OpenVMS License Management Utility Manual』 | AA-PVXUG-TK |
| 『Guide to OpenVMS version 8.3 Media』 | BA322-90048 |
| 『HP OpenVMS Version 8.3 New Features and Documentation Overview』 | BA322-90046 |
| 『HP OpenVMS Version 8.3 Upgrade and Installation Manual』 | BA322-90045 |
| 『HP OpenVMS Version 8.3 Release Notes』 | BA322-90047 |

### 9.1.2 OpenVMS ドキュメンテーション・セット

OpenVMS のドキュメントは次のドキュメンテーション・セットで提供されます。

| ドキュメンテーション・セット | 説明 | Alpha パーツ番号 | I64 パーツ番号 |
|---|---|---|---|
| フル・セット | 主要なすべての OpenVMS リソースの広範囲にわたる説明情報が必要なユーザを対象にしている。すべての OpenVMS ドキュメントが 1 つのセットとして提供される。基本ドキュメンテーション・セットも含まれている。 | QA-001AA-GZ.8.3 | BA554MN |

| ドキュメンテーション・セット | 説明 | Alpha パーツ番号 | I64 パーツ番号 |
|---|---|---|---|
| 基本セット | フル・ドキュメンテーション・セットの一部。小規模なスタンドアロン・システムのシステム管理者や一般ユーザを対象にしている。最も一般的に使用される OpenVMS のドキュメントが含まれている。 | QA-09SAA-GZ.8.3 | BA555MN |

OpenVMS Alpha および OpenVMS I64 システムで，ドキュメント・セットの内容はほぼ共通です。 OpenVMS Alpha ドキュメント・セットと OpenVMS I64 ドキュメント・セットに含まれるドキュメントは 1 冊の例外を除き同一です。 OpenVMS Alpha ドキュメント・セットには， Alpha 専用ドキュメントの『COM, Registry, and Events for HP OpenVMS Developer's Guide』が含まれています。

表 9–2 は， OpenVMS のフル・ドキュメンテーション・セットまたは基本ドキュメンテーション・セットに含まれているドキュメントを示しています。各ドキュメントの詳細については，第 10.2 節を参照してください。

**表 9–2 OpenVMS フル・ドキュメンテーション・セット (QA-001AA-GZ.8.3/BA554MN)**

| ドキュメント | |
|---|---|
| **OpenVMS 基本ドキュメンテーション・セット** | **QA-09SAA-GZ.8.3 /BA555MN** |
| 『OpenVMS DCL Dictionary:A-M』[1] | AA-PV5KL-TK |
| 『OpenVMS DCL Dictionary:N-Z』[1] | AA-PV5LL-TK |
| 『HP OpenVMS Guide to System Security』[1] | AA-Q2HLH-TE |
| 『OpenVMS System Management Utilities Reference Manual:A-M』[1] | AA-PV5PK-TK |
| 『OpenVMS System Management Utilities Reference Manual:N-Z』[1] | AA-PV5QK-TK |
| 『OpenVMS System Manager's Manual, Volume 1:Essentials』[1] | AA-PV5MJ-TK |
| 『OpenVMS System Manager's Manual, Volume 2:Tuning, Monitoring, and Complex Systems』[1] | AA-PV5NJ-TK |
| 『OpenVMS User's Manual』[1] | AA-PV5JG-TK |
| 『OpenVMS Alpha V 8.3 New Features and Documentation Overview』[2] | BA322-90046 |
| 『OpenVMS Alpha V 8.3 Release Notes』[2] | BA322-90047 |
| **フル・ドキュメンテーション・セットに含まれている追加ドキュメント** | **QA-001AA-GZ.8.3** |
| 『HP OpenVMS Availability Manager User's Guide』[1] | AA-RNSJE-TE |
| 『COM, Registry, and Events for HP OpenVMS Developer's Guide』[3] | AA-RSCWC-TE |
| 『Compaq C Run-Time Library Reference Manual for OpenVMS Systems』[2] | AA-RSMUD-TE |
| 『HP C Run-Time Library Utilities Reference Manual』 | AA-R238C-TE |

[1] V8.3 で変更されたドキュメント

[2] V8.3 で新規に提供されるドキュメント

[3] Alpha のみ - QA-001AA-GZ.8.3 に含まれる。

(次ページに続く)

表 9–2 (続き) OpenVMS フル・ドキュメンテーション・セット (QA-001AA-GZ.8.3/BA554MN)

| ドキュメント | |
|---|---|
| フル・ドキュメンテーション・セットに含まれている追加ドキュメント | QA-001AA-GZ.8.3 |
| 『Compaq Portable Mathematics Library』 | AA-PV6VE-TE |
| 『DECamds User's Guide』 | AA-Q3JSE-TE |
| 『DEC Text Processing Utility Reference Manual』 | AA-PWCCD-TE |
| 『Extensible Versatile Editor Reference Manual』 | AA-PWCDD-TE |
| 『Guidelines for OpenVMS Cluster Configurations』[1] | AA-Q28LH-TK |
| 『Guide to Creating OpenVMS Modular Procedures』 | AA-PV6AD-TK |
| 『Guide to OpenVMS File Applications』[1] | AA-PV6PE-TK |
| 『Guide to the POSIX Threads Library』 | AA-QSBPD-TE |
| 『Guide to the DEC Text Processing Utility』 | AA-PWCBD-TE |
| 『Open Source Security for OpenVMS Alpha, Volume 1: Common Data Security Architecture』[2] | AA-RSCUC-TE |
| 『HP Open Source Security for OpenVMS,Volume 2: HP SSL for OpenVMS』[2] | AA-RSCVD-TE |
| 『HP Open Source Security for OpenVMS, Volume 3: Kerberos』[1] | AA-RUEBC-TE |
| 『OpenVMS Alpha Partitioning and Galaxy Guide』[1] | AA-REZQD-TE |
| 『HP OpenVMS Guide to Upgrading Privileged-Code Applications』 | AA-QSBGE-TE |
| 『HP OpenVMS System Analysis Tools Manual』[1] | AA-REZTE-TE |
| 『OpenVMS Calling Standard』 | AA-QSBBE-TE |
| 『OpenVMS Cluster Systems』[1] | AA-PV5WF-TK |
| 『HP OpenVMS Command Definition, Librarian, and Message Utilities Manual』 | AA-QSBDE-TE |
| 『OpenVMS Debugger Manual』 | AA-QSBJE-TE |
| 『HP OpenVMS Delta/XDelta Debugger Manual』 | AA-PWCAF-TE |
| 『OpenVMS I/O User's Reference Manual』[1] | AA-PV6SG-TK |
| 『OpenVMS Linker Utility Manual』 | AA-PV6CF-TK |
| 『OpenVMS MACRO-32 Porting and User's Guide』 | AA-PV64E-TE |
| 『OpenVMS Management Station Overview and Release Notes』 | AA-QJGCH-TE |
| 『OpenVMS Performance Management』 | AA-R237C-TE |
| 『Porting Applications from HP OpenVMS Alpha to HP OpenVMS Industry Standard 64 for Integrity Servers』[2] | BA442-90001 |
| 『OpenVMS Programming Concepts Manual, Volume I』[1] | AA-RNSHD-TK |
| 『OpenVMS Programming Concepts Manual, Volume II』[1] | AA-PV67H-TK |
| ) | |
| 『OpenVMS Record Management Services Reference Manual』[1] | AA-PV6RE-TK |
| 『OpenVMS Record Management Utilities Reference Manual』 | AA-PV6QD-TK |
| 『OpenVMS RTL General Purpose (OTS$) Manual』 | AA-PV6HE-TK |
| 『OpenVMS RTL Library (LIB$) Manual』 | AA-QSBHE-TE |

[1]V8.3 で変更されたドキュメント

[2]V8.3 で新規に提供されるドキュメント

(次ページに続く)

表 9–2 (続き) OpenVMS フル・ドキュメンテーション・セット (QA-001AA-GZ.8.3/BA554MN)

| ドキュメント | |
|---|---|
| フル・ドキュメンテーション・セットに含まれている追加ドキュメント | QA-001AA-GZ.8.3 |
| 『OpenVMS RTL Screen Management (SMG$) Manual』 | AA-PV6LD-TK |
| 『OpenVMS RTL String Manipulation (STR$) Manual』 | AA-PV6MD-TK |
| 『OpenVMS System Messages: Companion Guide for Help Message Users』 | AA-PV5TD-TK |
| 『OpenVMS System Services Reference Manual: A–GETUAI』[1] | AA-QSBMH-TE |
| 『OpenVMS System Services Reference Manual: GETUTC–Z』[1] | AA-QSBNH-TE |
| 『HP OpenVMS Utility Routines Manual』[1] | AA-PV6EG-TK |
| 『OpenVMS VAX RTL Mathematics (MTH$) Manual』 | AA-PVXJD-TE |
| 『OpenVMS VAX System Dump Analyzer Utility Manual』 | AA-PV6TD-TE |
| 『POLYCENTER Software Installation Utility Developer's Guide』 | AA-Q28MF-TK |
| 『VAX MACRO and Instruction Set Reference Manual』 | AA-PS6GD-TE |
| 『Volume Shadowing for OpenVMS』[1] | AA-PVXMK-TE |

[1]V8.3 で変更されたドキュメント

### 9.1.3 オペレーティング環境拡張ドキュメント・セット (I64 のみ)

オペレーティング環境拡張ドキュメント・セットには，OpenVMS OE に含まれる製品のマニュアルが含まれています。このドキュメント・セットに含まれるドキュメントの一覧は第 10.5 節を参照してください。

### 9.1.4 システム統合製品のドキュメント

システム統合製品 (SIP) は，OpenVMS ソフトウェアに含まれていますが，これらの製品を使用するには個別のライセンスを購入する必要があります。表 9–3 は，システム統合製品に関連するドキュメントを示しています。

表 9–3 システム統合製品のドキュメント

| システム統合製品 | 関連ドキュメント |
|---|---|
| HP Galaxy Software Architecture on OpenVMS Alpha | このドキュメントは OpenVMS フル・ドキュメンテーション・セットに含まれている。 |
| OpenVMS Clusters | OpenVMS Cluster ドキュメントは OpenVMS フル・ドキュメンテーション・セットに含まれている。 |
| RMS Journaling for OpenVMS | RMS Journaling for OpenVMS のマニュアルは，下記の URL の OpenVMS ドキュメント Web サイトで HTML 形式で提供されています。http://www.hp.com/go/openvms/doc |
| Volume Shadowing for OpenVMS | このドキュメントは OpenVMS フル・ドキュメンテーション・セットに含まれている。 |

### 9.1.5 アーカイブされた OpenVMS ドキュメント

OpenVMS では， OpenVMS オペレーティング・システムのドキュメントを継続的に更新，変更，拡張しています。必要に応じてドキュメントはアーカイブに移されます。アーカイブされたドキュメントは，次の Web サイトからアクセスすることができます。

http://www.hp.com/go/openvms/doc

アーカイブされた OpenVMS のドキュメントの一覧については，第 10.6 節を参照してください。

## 9.2 OpenVMS ドキュメントの開発ツール

OpenVMS ドキュメント・チームは SGML (Standard Generalized Markup Language) ベースのツールを使用してマニュアルを作成しています。 SGML は業界標準の記述言語であり，お客様にとっても OpenVMS ドキュメンテーションにとっても多くの利点があります。

SGML で作成されたドキュメントとその他のドキュメントでは見た目が異なります。 HTML， PDF，および印刷ドキュメントのすべてでこの違いが見られますが，これは使用したツールの違いによるものです。

下記の Version 8.3 ドキュメントはこの新しいツールを使用して作成されています。

- 『HP Open Source Security for OpenVMS, Volume 2: HP SSL for OpenVMS』
- 『HP Open Source Security for OpenVMS, Volume 3: Kerberos』
- 『HP OpenVMS Version 8.3 Upgrade and Installation Manual』
- 『Guide to HP OpenVMS Version 8.3 Media』
- 『OpenVMS I/O User's Reference Manual』
- 『HP OpenVMS System Manager's Manual, Volume 1:Essentials』
- 『HP OpenVMS System Manager's Manual, Volume 2:Turning,Monitoring,and Complex Systems』

## 9.3 CD に収録されているオンライン・ドキュメント

OpenVMS オペレーティング・システムおよび多くの関連製品のオンライン・ドキュメントは， OpenVMS システムおよび Windows プラットフォームで共通に使用できる ISO9660 Level 2 形式の 1 枚の CD に収録されています。

### 9.3.1 オンライン形式

ドキュメンテーション CD-ROM には，ドキュメントがさまざまな形式で収録されています。

| ドキュメント | 提供される形式 |
|---|---|
| 現在のバージョンの OpenVMS のドキュメント | HTML, PDF |
| 『HP OpenVMS Version 8.3 Upgrade and Installation』 | HTML, PDF |
| 『HP OpenVMS Version 8.3 Release Notes』 | HTML, PDF |
| 『HP OpenVMS V 8.3 New Features and Documentation Overview』 | HTML, PDF |
| レイヤード製品のドキュメント | HTML, PDF |

以前のバージョンで提供していた Bookreader ファイルは提供されなくなりました。

ドキュメントへのアクセス方法の詳細は『HP OpenVMS V8.3 インストレーション・ガイド[翻訳版]』を参照してください。

## 9.4 OpenVMS Web サイトで提供されるオンライン・ドキュメント

次の OpenVMS Web サイトでは， OpenVMS のドキュメントがさまざまなオンライン形式で提供されています。

```
http://www.hp.com/gp/openvms/doc
```

このサイトには， OpenVMS フル・ドキュメンテーション・セットに含まれるドキュメントの最新のバージョン，および特定のレイヤード製品のドキュメントへのリンクが掲載されています。

## 9.5 オンライン・ヘルプ

OpenVMS オペレーティング・システムでは，フル・ドキュメンテーション・セットに説明されているコマンド，ユーティリティ，システム・ルーチンのオンライン・ヘルプが提供されます。

システム・メッセージのオンライン説明にアクセスするには，ヘルプ・メッセージ機能を使用します。さらに，ヘルプ・メッセージ・データベースに書き込んだメッセージ・ドキュメンテーションなど，独自のソース・ファイルを追加することができます。『OpenVMS System Messages: Companion Guide for Help Message Users』では，ヘルプ・メッセージ機能の使い方が説明されています。また，次のように入力して，ヘルプ・メッセージに関する DCL ヘルプにアクセスすることもできます。

```
$ HELP HELP/MESSAGE
```

OpenVMS ユーティリティ・ルーチンに関する参照情報は，オンライン・ヘルプで提供されるようになりました。

# 10

# OpenVMS のドキュメントの説明

この章では，次の OpenVMS ドキュメントについて簡単に説明します。

- OpenVMS メディア・キットに含まれるドキュメント (第 10.1 節)
- OpenVMS ベース・ドキュメンテーション・セットおよびフル・ドキュメンテーション・セットに含まれるドキュメント (第 10.2 節と第 10.3 節)
- RMS Journaling のドキュメント (第 10.4 節)
- OpenVMS I64 OE 拡張キットに含まれているドキュメント
- アーカイブされたドキュメント (第 10.6 節)

## 10.1 OpenVMS メディア・キットに含まれるドキュメント

『Guide to HP OpenVMS Version 8.3 Media』
**（翻訳版は『HP OpenVMS Version 8.3 CD/DVD ユーザーズ・ガイド』）**
OpenVMS オペレーティング・システムおよびドキュメント CD に関する情報を提供します。 OpenVMS Alpha および OpenVMS I64 Version 8.3メディア・キットの内容を示し，インストール情報へのポインタを示し，ドキュメント CD-ROM に収録されているドキュメントへのアクセス方法を示します。

『OpenVMS License Management Utility Manual』
OpenVMS のライセンス管理ツールである LMF (License Management Facility) について説明します。 LMF には， License Management ユーティリティ (LICENSE) と，ソフトウェア・ライセンスの登録，管理，追跡に使用するコマンド・プロシージャ VMSLICENSE.COM が含まれています。

『HP OpenVMS Version 8.3 Upgrade and Installation Manual』
**（翻訳版は『HP OpenVMS V8.3 インストレーション・ガイド[翻訳版]』）**
OpenVMS Alpha および OpenVMS I64 オペレーティング・システムのインストレーション手順を示します。ブート，シャットダウン，バックアップ，ライセンス・プロシージャに関する情報が記載されています。

『OpenVMS Version 8.3 New Features and Documentation Overview』
**（翻訳版は『**HP OpenVMS V8.3 **新機能説明書』）**
OpenVMS I64 および Alpha オペレーティング・システム 8.3 リリースの新しいコンポーネントと拡張されたコンポーネントについて説明します。 V8.3 で変更された OpenVMS ドキュメントについて説明し， OpenVMS ドキュメントの印刷形式およびオンライン形式についても説明します。

『OpenVMS Version 8.3 Release Notes』
**（翻訳版は『**HP OpenVMS V8.3 **リリース・ノート[翻訳版]』）**
ソフトウェアの変更点，インストール，アップグレード，互換性情報，ソフトウェアの新しい問題点と制約事項および既存の問題点と制約事項，ソフトウェアとドキュメントの修正について説明します。

**注意**

**日本語** OpenVMS **のメディア・キット (インストレーション・キット) には，上記の他に『日本語** OpenVMS **リリース・ノート』および『日本語** OpenVMS **インストレーション・ガイド』も含まれます。**

## 10.2 OpenVMS 基本ドキュメント・セットのドキュメント

『OpenVMS DCL Dictinary』
**（翻訳版は『**OpenVMS DCL **ディクショナリ』）**
DCL (DIGITAL Command Language) について説明し，すべての DCL コマンドとレキシカル関数の詳細な参照情報と例をアルファベット順に示します。このドキュメントは 2 分冊になっています。

『HP OpenVMS Guide to System Security』
**(翻訳版は『**OpenVMS **システム・セキュリティ・ガイド』)**
OpenVMS オペレーティング・システムで提供されるセキュリティ機能について説明します。具体的なセキュリティ・ニーズを示し，それぞれの状況に応じた各機能の目的と適切な応用を示します。

『OpenVMS System Management Utilities Reference Manual』
**（翻訳版は『**OpenVMS **システム管理ユーティリティ・リファレンス・マニュアル』）**
システム管理作業を実行するためのユーティリティや，システムへのアクセスとリソースの制御と監視に使用するツールについて参照情報を示します。 AUTOGEN コマンド・プロシージャについても説明します。このドキュメントは 2 分冊になっています。

『OpenVMS System Manager's Manual, Volume 1: Essentials』
**（翻訳版は『**OpenVMS **システム管理者マニュアル (上巻)』）**
システムの起動，ソフトウェアのインストール，プリント・キューとバッチ・キューの設定など，日常的に行う操作の設定と管理の方法について説明します。また，日常的に行うディスク操作と磁気テープ操作についても説明します。

『OpenVMS System Manage's Manual, Volume 2: Tuning, Monitoring, and Complex Systems』
**（翻訳版は『**OpenVMS **システム管理者マニュアル (下巻)』）**
ネットワークの構成と制御，システムの監視，システム・パラメータの管理の方法について説明します。また， OpenVMS Cluster システム，ネットワーク環境，DECdtm 機能についても説明します。

『OpenVMS User's Manual』
**（翻訳版は『**OpenVMS **ユーザーズ・マニュアル』）**
オペレーティング・システムの概要を示し，基本的な概念，タスク情報，日常のコンピューティング・タスクを実行するための参照情報も示します。ファイルとディレクトリの操作方法についても説明します。また，次の追加トピックも含まれています。

- Mail ユーティリティと Phone ユーティリティによるメッセージの送信
- Sort/Merge ユーティリティの使用
- 論理名とシンボルの使用
- コマンド・プロシージャの作成
- EVE および EDT テキスト・エディタによるファイルの編集

『OpenVMS Version 8.3 New Features and Documentation Overview』
**（翻訳版は『**HP OpenVMS V8.3 **新機能説明書』）**
V8.3 リリースの新しいコンポーネントと拡張されたコンポーネントについて説明します。 V8.3 で変更された OpenVMS ドキュメントについて説明し， OpenVMS ドキュメントの印刷形式およびオンライン形式についても説明します。

『OpenVMS Version 8.3 Release Notes』
**（翻訳版は『**HP OpenVMS V8.3 **リリース・ノート[翻訳版]』）**
ソフトウェアの変更点，インストール，アップグレード，互換性情報，ソフトウェアの新しい問題点と制約事項および既存の問題点と制約事項，ソフトウェアとドキュメントの修正について説明します。

## 10.3 OpenVMS フル・ドキュメンテーション・セットの追加ドキュメント

『HP OpenVMS Availability Manager User's Guide』
OpenVMS Alpha または Windows ノードから HP Availability Manager システム管理ツールを使用して，拡張ローカル・エリア・ネットワーク (LAN) で 1 つ以上の OpenVMS ノードを監視する方法と，詳細な分析のために特定のノードまたはプロセスを監視する方法について説明します。

『COM, Registry, and Events for HP OpenVMS Developer's Guide』
OpenVMS 環境と Windows NT 環境の間で簡単に移行できるアプリケーションを開発するプログラマを対象にしたドキュメントです。既存の OpenVMS アプリケーションやデータをカプセル化する場合や， OpenVMS システム用に新しい COM アプリケーションを開発する場合は，このドキュメントを参照してください。

また， OpenVMS Registry を使用して OpenVMS システムだけに関する情報を格納する場合や， OpenVMS レジストリ情報と Windows NT レジストリ情報の両方を格納するための共用の格納場所として OpenVMS Registry を使用する場合も，このドキュメントを参照してください。このドキュメントは,

以前は『OpenVMS Connectivity Developer Guide』 (翻訳版は『OpenVMS コネクティビティ開発者ガイド』) という名称でオンラインで提供されていました。

『HP C Run-Time Library Reference Manual for OpenVMS Systems』
**（翻訳版は『HP C ランタイム・ライブラリ・リファレンス・マニュアル』）**
I/O 操作，文字および文字列操作，算術演算，エラー検出，サブプロセスの生成，システム・アクセス，画面管理などを実行する HP C RTL の関数とマクロに関する参照情報を示します。オペレーティング・システム間での移植に関する問題点， TCP/IP プロトコル用にインターネット・アプリケーション・プログラムを作成するために使用する HP C for OpenVMS ソケット・ルーチンについても説明します。

『Compaq C Run-Time Library Uilities Reference Manual』
**（翻訳版は『Compaq C 国際化ユーティリティ・リファレンス・マニュアル』）**
国際化ソフトウェア・アプリケーションでローカリゼーションとタイム・ゾーン・データを管理するための C ランタイム・ライブラリ・ユーティリティの詳細な使い方と参照情報を示します。

『Compaq Portable Mathematics Library』
DPML (Compaq Portable Mathematics Library) の算術演算ルーチンについて説明します。これらのルーチンは OpenVMS Alpha システムでのみ提供されます。 VAX プログラマは『OpenVMS VAX RTL Mathematics (MTH$) Manual』を参照してください。

『DECamds User's Guide』
DECamds ソフトウェアのインストールと使用方法について説明します。 DECamds は， OpenVMS システムおよび OpenVMS Cluster 環境でイベントの監視，診断，追跡を行うためのシステム管理ツールです。

『DEC Text Processing Utility Reference Manual』
DECTPU (DEC Text Processing Utility) について説明し， DECTPU に対する EDT キーパッド・エミュレータ・インタフェースに関する参照情報を示します。

『Extensible Versatile Editor Reference Manual』
EVE テキスト・エディタに関するコマンド参照情報を示します。また， EDT コマンドと EVE コマンドの間の相互参照も示します。

『Guidelines for OpenVMS Cluster Configurations』
**（翻訳版は『**OpenVMS Cluster **構成ガイド』）**
このドキュメントでは，システム，インターコネクト，ストレージ・デバイス，ソフトウェアを選択するのに役立つ情報を示します。高い可用性，拡張性，パフォーマンス，容易なシステム管理を実現するためにこれらのコンポーネントを構成するのに役立ちます。このドキュメントでは， OpenVMS Cluster システムで SCSI および Fibre Channel を使用する方法についても詳しく説明しています。

『Guide to Creating OpenVMS Modular Procedures』
プログラムを複数のモジュールに分割し，各モジュールを個別のプロシージャとしてコーディングすることにより，複雑なプログラミング作業を実行する方法について説明します。

『Guide to OpenVMS File Applications』
RMS (Record Management Services) を使用して，効率のよいデータ・ファイルの設計，作成，管理を行うためのガイドラインを示します。このドキュメントは， RMS ファイルを使用するプログラム，特にパフォーマンスが重要視されるプログラムを取り扱うアプリケーション・プログラマおよび設計者を対象にしています。

『Guide to the POSIX Threads Library』
弊社のマルチスレッド・ランタイム・ライブラリである POSIX Threads Library (以前の名称は DECthreads) パッケージについて説明します。このパッケージに含まれているルーチンは， 1 つのプロセスで提供されるアドレス空間内で複数の実行スレッドを作成し，制御することができます。このドキュメントでは使い方のヒントと参照情報の両方を示し， 3 つのインタフェースについて説明しています。 3 つのインタフェースとは， IEEE POSIX 1003.1c 標準規格に準拠したルーチン (pthreadと呼びます)，非スレッド・アプリケーションでスレッド関連サービスを提供するルーチン (スレッド独立サービスまたはtisと呼びます)，上位互換性のある安定したインタフェースを提供する弊社固有のルーチン (cmaと呼びます) です。

『Guide to the DEC Text Processing Utility』
DECTPU プログラムの開発の概要について説明します。

『Open Source Security for OpenVMS Alpha, Volume 1: Common Data Security Architecture』
CDSA (Common Data Security Architecture) を使用して，プログラムにセキュリティ機能を追加するアプリケーション開発者を対象にしています。 CDSA について説明し，インストールと初期化について説明し，サンプル・プログラムも提供します。 CDSA アプリケーション・プログラミング・インタフェース・モジュールが含まれています。

『HP Open Source Security for OpenVMS, Volume 2: HP SSL for OpenVMS』
HP SSL (HP Secure Sockets Layer) for OpenVMS Alpha で OpenVMS アプリケーションへの通信リンクを保護することを検討しているアプリケーション開発者を対象にしています。インストールの方法とリリース・ノートを示し，サンプル・プログラムを提供します。 OpenSSL アプリケーション・プログラミング・インタフェース・モジュールのプログラミング情報と参照情報が示されています。

『HP Open Source Security for OpenVMS, Volume 3: Kerberos』
ネットワーク接続でクライアントがサーバに対して認証を求め (そしてサーバがクライアントに対して認証を提供し) 安全に通信できるように， Kerberos プロトコルで文字の暗号化機能を実装したいアプリケーション開発者を対象にしています。

『OpenVMS Alpha Partitioning and Galaxy Guide』
**（翻訳版は**『OpenVMS Alpha **パーティショニングおよび** Galaxy **ガイド**』**）**
OpenVMS Alpha Version 7.3-2 で提供されるすべての OpenVMS Galaxy 機能の使い方について詳しく説明します。 AlphaServer 8400， 8200， 4100 システムで OpenVMS Galaxy コンピューティング環境を作成，管理，使用する手順も示します。

『HP OpenVMS Guide to Upgrading Privileged-Code Applications』
OpenVMS Alpha Version 7.0 で OpenVMS Alpha の 64 ビット仮想アドレッシングおよびカーネル・スレッドがサポートされるようになった結果， Alpha の特権付きコード・アプリケーションおよびデバイス・ドライバに影響を与える可能性のある OpenVMS Alpha Version 7.0 の変更点について説明します。

OpenVMS Alpha Version 7.0 より前のバージョンで作成された特権付きコード・アプリケーションは，このガイドの説明に従ってソース・コードを変更する必要があります。

『OpenVMS System Analysis Tools Manual』
次のシステム分析ツールについて詳しく説明します。また， DOSD (dump off system disk) 機能と DELTA/XDELTA デバッガの概要も示します。

- System Dump Analysis (SDA)
- System code debugger (SCD)
- System dump debugger (SDD)
- Watchpoint ユーティリティ

このドキュメントは，システム障害の原因を調べ，デバイス・ドライバなどのカーネル・モード・コードをデバッグしなければならないシステム・プログラマを対象しています。

『OpenVMS Calling Standard』
OpenVMS オペレーティング・システムの呼び出し標準規約について説明します。

『OpenVMS Cluster Systems』
**（翻訳版は『**OpenVMS Cluster **システム』）**
OpenVMS Cluster システムの構成と管理の手順およびガイドラインについて説明します。また，クラスタに接続されたシステムで高い可用性，構築ブロックの拡張，統一されたシステム管理を実現する方法についても説明します。

『HP OpenVMS Command Definition, Librarian, and Message Utilities Manual』
次のユーティリティについて説明し，参照情報も示します。

- Command Definition ユーティリティ
- Librarian ユーティリティ
- Message ユーティリティ

『OpenVMS Debugger Manual』
**（翻訳版は『**HP OpenVMS **デバッガ説明書』）**
プログラマを対象に OpenVMS Debugger の機能について説明します。

『HP OpenVMS Delta/XDelta Debugger Manual』
特権付きプロセッサ・モードまたは引き上げられた割り込み優先順位レベルで動作するプログラムをデバッグするために使用する Delta/XDelta ユーティリティについて説明します。

『OpenVMS I/O User’s Reference Manual』
オペレーティング・システムに付属しているデバイス・ドライバを使用して，システム・プログラマが I/O 操作をプログラミングするのに必要な情報を示します。

『OpenVMS Linker Utility Manual』
Linker ユーティリティを使用して，OpenVMS システムで動作するイメージを作成する方法について説明します。また，リンク修飾子とリンク・オプションを使用してリンク操作を制御する方法についても説明します。

『HP OpenVMS MACRO Compiler Porting and User's Guide』
MACRO-32 コンパイラの機能を使用して，既存のVAX MACROアセンブリ言語コードを OpenVMS Alpha システムに移植する方法について説明します。既存の OpenVMS Alpha のコードを OpenVMS I64 システムへ移植する方法についても説明してます。また，コンパイラの 64 ビット・アドレッシングのサポート機能を使用する方法についても説明します。

『OpenVMS Management Station Overview and Release Notes』
OpenVMS Management Station の概要とリリース・ノートを示し，このソフトウェアの使い方の概要も示します。OpenVMS Management Station は，OpenVMS システムでユーザ・アカウントやプリンタの管理作業を行うシステム管理者やその他の人を対象した，Microsoft Windows ベースの強力な管理ツールです。

『OpenVMS Performance Management』
OpenVMS システムでパフォーマンスを最適化するために使用する手法について説明します。

『Porting Applications from HP OpenVMS Alpha to HP OpenVMS Industry Standard 64 for Integrity Servers』
**（翻訳版は**『HP OpenVMS Alpha **から** OpenVMS I64 **へのアプリケーション・ポーティング・ガイド**』**）**
HP OpenVMS Alpha から HP OpenVMS I64 へ移行しようとしているアプリケーション開発者に移行計画の枠組みを提供します。

『HP OpenVMS Programming Concepts Manual』
プロセスの生成，カーネル・スレッドとカーネル・スレッド・プロセス構造，プロセス間通信，プロセス制御，データの共用，条件処理，AST などの概念について説明します。この 2 分冊のドキュメントでは，システム・サービス，ユーティリティ・ルーチン，ランタイム・ライブラリ (RTL) ルーチンを使用して，OpenVMS の機能を利用する方法を説明します。

『OpenVMS Record Management Services Reference Manual』
RMS データ・ファイルを使用するすべてのプログラマを対象に，参照情報と使用方法を示します。

『OpenVMS Record Management Utilities Reference Manual』
次の RMS ユーティリティに関する説明と参照情報を示します。

- Analyze/RMS_File ユーティリティ
- Convert and Convert/Reclaim ユーティリティ
- File Definition Language 機能

『OpenVMS RTL General Purpose (OTS$) Manual』
OpenVMS ランタイム・ライブラリの OTS$機能に含まれる汎用ルーチンについて説明します。 I64, Alpha, VAX 固有のルーチンを示し，各システムで異なる動作を実行するルーチンについても説明します。

『OpenVMS RTL Library (LIB$) Manual』
OpenVMS ランタイム・ライブラリの LIB$機能に含まれる汎用ルーチンについて説明します。 I64, Alpha, VAX 固有のルーチンを示し，各システムで異なる動作を実行するルーチンについても説明します。

『OpenVMS RTL Screen Management (SMG$) Manual』
OpenVMS ランタイム・ライブラリの SMG$機能に含まれる画面管理ルーチンについて説明します。 Alpha 固有のルーチンと VAX 固有のルーチンについて説明し，各システムで異なる動作を実行するルーチンについても説明します。

『OpenVMS RTL String Manipulation (STR$) Manual』
OpenVMS ランタイム・ライブラリの STR$機能に含まれる文字列操作ルーチンについて説明します。 Alpha 固有のルーチンと VAX 固有のルーチンを示し，各システムで異なる動作を実行するルーチンについても説明します。

『OpenVMS System Messages: Companion Guide for Help Message Users』
ヘルプ・メッセージを表示するためのツールであるヘルプ・メッセージ機能について説明します。 HELP/MESSAGE コマンドとその修飾子について説明し，ヘルプ・メッセージ・データベースのカスタマイズに関する詳細情報も示します。また，システムおよびヘルプ・メッセージ機能が完全に動作しないときに表示される可能性のあるメッセージの説明も示します。

『OpenVMS System Services Reference Manual』
リソースの制御，プロセス通信の実行， I/O の制御，その他のオペレーティング・システム機能を実行するためにオペレーティング・システムで使用するルーチンについて説明します。このドキュメントは 2 分冊になっています。

『HP OpenVMS Utility Routines Manual』
プログラムで特定の OpenVMS ユーティリティの呼び出し可能インタフェースを使用するためのルーチンについて説明します。

『OpenVMS VAX RTL Mathematics (MTH$) Manual』
OpenVMS ランタイム・ライブラリの MTH$機能に含まれる算術演算ルーチンについて説明します。このドキュメントは OpenVMS VAX を使用するプログラマを対象にしています (Alpha のプログラマは『Compaq Portable Mathematics Library』を参照してください)。

『OpenVMS VAX System Dump Analyzer Utility Manual』
System Dump Analyzer ユーティリティを使用して，システム障害を調べ，稼動中の OpenVMS VAX システムを確認する方法について説明します。 VAX のプログラマはこのドキュメントを参照してください。 Alpha および I64 のプログラマは『VMS System Dump Analyzer Utility Manual』を参照してください。

『POLYCENTER Software Installation Utility Developer's Guide』
POLYCENTER Software Installation ユーティリティを使用してインストールされるソフトウェア製品を開発する場合の手順とガイドラインを示します。このドキュメントは， OpenVMS オペレーティング・システムのレイヤード・ソフトウェア製品のインストール手順を設計する開発者を対象にしています。

『VAX MACRO and Instruction Set Reference Manual』
VAX MACROのアセンブラ・ディレクティブと VAX 命令セットの両方について説明します。

『Volume Shadowing for OpenVMS』
フェーズ II のボリューム・シャドウイングで高いデータ可用性を提供する方法について説明します。

## 10.4 RMS Journaling **のドキュメント**

『RMS Journaling for OpenVMS Manual』
3 種類の RMS Journaling について説明し， RMS Journaling をサポートする他の OpenVMS コンポーネントについても説明します。このドキュメントでは， RMS Recovery ユーティリティ (ジャーナリングを使用して保存したデータを回復するために使用します)，トランザクション処理システム・サービス， RMS Journaling を使用するときに必要なシステム管理タスクについても説明します。

## 10.5 OpenVMS I64 OE 拡張キットに含まれているドキュメント

以下に示すのは OpenVMS I64 オペレーティング環境に関係するドキュメントです。

- HP DECwindows Motif for OpenVMS Installation Guide
- HP DECwindows Motif for OpenVMS New Features
- HP DECwindows Motif for OpenVMS Documentation Overview
- HP DECwindows Motif for OpenVMS Management Guide
- HP DECnet-Plus for OpenVMS Installation and Configuration
- HP DECnet-Plus for OpenVMS Introduction and User's Guide
- HP DECnet-Plus Network Management
- HP DECnet-Plus for OpenVMS DECdts Programming Reference
- HP DECnet-Plus for OpenVMS DECdts Management
- HP DECnet-Plus for OpenVMS DECdns Management
- HP DECnet-Plus for OpenVMS Network Management Quick Reference Guide
- HP DECnet-Plus for OpenVMS OSAK Programming
- HP DECnet-Plus for OpenVMS OSAK Programming Reference
- HP DECnet-Plus for OpenVMS OSAK SPI Programming Reference
- HP DECnet-Plus for OpenVMS Problem Solving Manual
- HP DECnet-Plus for OpenVMS Programming Manual
- HP DECnet-Plus for OpenVMS FTAM and Virtual Terminal User and Management
- HP DECnet-Plus for OpenVMS Problem Solving
- HP DECnet-Plus for OpenVMS Network Control Language Reference
- HP DECnet-Plus for OpenVMS Planning Guide
- HP TCP/IP Services for OpenVMS Installation and Configuration
- HP TCP/IP Services for OpenVMS Sockets API and System Services Programming
- HP TCP/IP Services for OpenVMS Concepts and Planning
- HP TCP/IP Services for OpenVMS SNMP Programming Reference
- HP TCP/IP Services for OpenVMS ONC RPC Programming
- HP TCP/IP Services for OpenVMS Tuning and Troubleshooting
- HP TCP/IP Services for OpenVMS Guide to SSH for OpenVMS
- HP TCP/IP Services for OpenVMS Management
- HP TCP/IP Services for OpenVMS Management Command Reference

- HP TCP/IP Services for OpenVMS Management Command Quick Reference Card
- HP TCP/IP Services for OpenVMS User's Guide
- HP TCP/IP Services for OpenVMS UNIX Command Equivalents Reference Card
- HP TCP/IP Services for OpenVMS Guide to IPv6
- HP DECprint Supervisor (DCPS) for OpenVMS User's Guide
- HP DECprint Supervisor (DCPS) for OpenVMS Software Installation
- HP DECprint Supervisor (DCPS) for OpenVMS Manager's Guide
- HP DCE for OpenVMS Product Guide
- HP DCE for OpenVMS Reference Guide
- HP DCE for OpenVMS Installation and Configuration Guide

## 10.6 アーカイブされたドキュメント

表 10–1 は，アーカイブされた OpenVMS のドキュメントを示しています。アーカイブされたドキュメントのほとんどの情報は，他のドキュメントまたはオンライン・ヘルプに統合されているという点に注意してください。

**表 10–1 アーカイブされた OpenVMS のドキュメント**

| ドキュメント | |
|---|---|
| 『A Comparison of System Management on OpenVMS AXP and OpenVMS VAX』 | AA-PV71B-TE |
| 『Building Dependable Systems: The OpenVMS Approach』 | AA-PV5YB-TE |
| 『Creating an OpenVMS Alpha Device Driver from an OpenVMS VAX Device Driver』 | AA-R0Y8A-TE |
| 『Creating an OpenVMS AXP Step 2 Device Driver from a Step 1 Device Driver』 | AA-Q28TA-TE |
| 『Creating an OpenVMS AXP Step 2 Device Driver from an OpenVMS VAX Device Driver』 | AA-Q28UA-TE |
| 『Guide to OpenVMS AXP Performance Management』 | AA-Q28WA-TE |
| 『Guide to OpenVMS Performance Management』 | AA-PV5XA-TE |
| 『Migrating an Application from OpenVMS VAX to OpenVMS Alpha』 | AA-KSBKB-TE |
| 『Migrating an Environment from OpenVMS VAX to OpenVMS Alpha』 | AA-QSBLA-TE |
| 『Migrationg to an OpenVMS AXP System: Planning for Migration』 | AA-PV62A-TE |
| 『Migrating to an OpenVMS AXP System: Recompiling and Relinking Applications』 | AA-PV63A-TE |
| 『OpenVMS Alpha Guide to 64-Bit Addressing and VLM Features』 | AA-QSBCC-TE |

(次ページに続く)

**表 10–1 (続き) アーカイブされた OpenVMS のドキュメント**

| ドキュメント | |
|---|---|
| 『OpenVMS Alpha System Dump Analyzer Utility Manual』 | AA-PV6UC-TE |
| OpenVMS Alpha Version 7.3-1 New Features and Documentation Overview | AA-RSHYA-TE |
| OpenVMS Alpha Version 7.3-1 Release Notes | AA-RSD0A-TE |
| 『OpenVMS AXP Device Support: Developer's Guide』 | AA-Q28SA-TE |
| 『OpenVMS AXP Device Support: Reference』 | AA-Q28PA-TE |
| 『OpenVMS Bad Block Locator Utility Manual』 | AA-PS69A-TE |
| 『OpenVMS Compatibility Between VAX and Alpha』 | AA-PYQ4C-TE |
| 『OpenVMS Developer's Guide to VMSINSTAL』 | AA-PWBXA-TE |
| 『OpenVMS DIGITAL Standard Runoff Reference Manual』 | AA-PS6HA-TE |
| 『OpenVMS EDT Reference Manual』 | AA-PS6KA-TE |
| 『OpenVMS Exchange Utility Manual』 | AA-PS6AA-TE |
| 『OpenVMS Glossary』 | AA-PV5UA-TK |
| 『OpenVMS Guide to Extended File Specifications』 | AA-REZRB-TE |
| 『OpenVMS Master Index』 | AA-QSBSD-TE |
| 『OpenVMS National Character Set Utility Manual』 | AA-PS6FA-TE |
| 『OpenVMS Obsolete Features Manual』 | AA-PS6JA-TE |
| 『OpenVMS Programming Environment Manual』 | AA-PV66B-TK |
| 『OpenVMS Programming Interfaces: Calling a System Routine』 | AA-PV68B-TK |
| 『OpenVMS RTL DECtalk (DTK$) Manual』 | AA-PS6CA-TE |
| 『OpenVMS RTL Parallel Processing (PPL$) Manual』 | AA-PV6JA-TK |
| 『OpenVMS Software Overview』 | AA-PVXHB-TE |
| 『OpenVMS SUMSLP Utility Manual』 | AA-PS6EA-TE |
| 『OpenVMS System Messages and Recovery Procedures Reference Manual: A–L』 | AA-PVXKA-TE |
| 『OpenVMS System Messages and Recovery Procedures Reference Manual: M–Z』 | AA-PVXLA-TE |
| 『OpenVMS Terminal Fallback Utility Manual』 | AA-PS6BA-TE |
| 『OpenVMS VAX Card Reader, Line Printer, and LPA11–K I/O User's Reference Manual』 | AA-PVXGA-TE |
| 『OpenVMS VAX Device Support Manual』 | AA-PWC8A-TE |
| 『OpenVMS VAX Device Support Reference Manual』 | AA-PWC9A-TE |
| 『OpenVMS VAX Patch Utility Manual』 | AA-PS6DA-TE |
| 『OpenVMS Wide Area Network I/O User's Reference Manual』 | AA-PWC7A-TE |
| 『PDP–11 TECO User's Guide』 | AA-K420B-TC |
| 『POLYCENTER Software Installation Utility User's Guide』 | AA-Q28NA-TK |
| 『TCP/IP Networking on OpenVMS Systems』 | AA-QJGDB-TE |
| 『Standard TECO Text Editor and Corrector for the VAX, PDP–11, PDP-10, and PDP-8』 | CD-ROM でのみ提供 |

表 10–2 は，アーカイブされたネットワーキング・ドキュメントおよびインストール補足資料を示しています。

**表** 10–2 **アーカイブされたネットワーキング・ドキュメントおよびインストール補足資料**

| **ドキュメント** | |
|---|---|
| 『OpenVMS VAX Upgrade and Installation Supplement: VAX 8820, 8830, 8840』 | AA-PS6MA-TE |
| 『OpenVMS VAX Upgrade and Installation Supplement: VAX 8200, 8250, 8300, 8350』 | AA-PS6PA-TE |
| 『OpenVMS VAX Upgrade and Installation Supplement: VAX 8530, 8550, 8810 (8700), and 8820–N (8800)』 | AA-PS6QA-TE |
| 『OpenVMS VAX Upgrade and Installation Supplement: VAX 8600, 8650』 | AA-PS6UA-TE |
| 『VMS Upgrade and Installation Supplement: VAX–11/780, 785』 | AA-LB29B-TE |
| 『VMS Upgrade and Installation Supplement: VAX–11/750』 | AA-LB30B-TE |

ここでは，アーカイブされた OpenVMS ドキュメントについて説明します。

『A Comparison of System Management on OpenVMS AXP and OpenVMS VAX』
システム管理ツール， Alpha のページ・サイズがシステム管理操作に与える影響，システム・ディレクトリ構造，相互運用性に関する問題点，パフォーマンス情報について説明します。このドキュメントは， OpenVMS Alpha システムの管理方法を短時間に学習する必要のあるシステム管理者を対象にしています。

『Building Dependable Systems: The OpenVMS Approach』
ビジネス・アプリケーションで必要とされる信頼性を分析し，コンピューティング・システムを使用して，信頼性の達成目標をサポートする方法を判断するための，実際的な情報を示します。この情報の他に， OpenVMS や関連ハードウェア，レイヤード・ソフトウェア製品の信頼性機能の技術概要も補足されています。

『Creating an OpenVMS Alpha Device Driver from an OpenVMS VAX Device Driver』
OpenVMS VAX で使用されているデバイス・ドライバを OpenVMS Alpha で動作するデバイス・ドライバに変換する手順について説明します。このドキュメントには，Macro-32 で作成された Alpha ドライバを操作するためのデータ構造，ルーチン，マクロも含まれています。

『Creating an OpenVMS AXP Step 2 Device Driver from a Step 1 Device Driver』
Step 1 デバイス・ドライバ (OpenVMS AXP の初期のバージョンで使用) を Step 2 デバイス・ドライバにアップグレードする方法について説明します。 OpenVMS AXP Version 6.1 では， Step 2 のデバイス・ドライバが必要です。

『Creating an OpenVMS AXP Step 2 Device Driver from an OpenVMS VAX Device Driver』
OpenVMS VAX で使用されているデバイス・ドライバを，OpenVMS AXP Version 6.1 で使用される Step 2 デバイス・ドライバに移行する方法について説明します。

『Guide to OpenVMS AXP Performance Management』
OpenVMS Alpha システムでパフォーマンスを最適化するために使用される手法について説明します。

『Guide to OpenVMS Performance Management』
OpenVMS VAX システムでパフォーマンスを最適化するために使用される手法について説明します。

『Migrating an Application from OpenVMS VAX to OpenVMS Alpha』
**（翻訳版は『**OpenVMS VAX **から** OpenVMS Alpha **へのアプリケーションの移行』）**
OpenVMS VAX アプリケーションの OpenVMS Alpha バージョンを作成する方法について説明します。 VAX から Alpha への移行プロセスの概要を示し，移行の計画に役立つ情報も示します。移行計画で必要になる判断と，これらの判断を下すのに必要な情報の入手方法を説明します。さらに，このドキュメントでは，使用できる移行方法について説明し，各方法で必要な作業量を見積もり，各アプリケーションに最適な方法を選択できるようにします。

『Migrating an Environment from OpenVMS VAX to OpenVMS Alpha』
OpenVMS VAX システムから OpenVMS Alpha システムまたは複合アーキテクチャ・クラスタにコンピューティング環境を移行する方法について説明します。 VAX から Alpha への移行プロセスの概要を示し，VAX コンピュータと Alpha コンピュータでのシステム管理およびネットワーク管理の相違点について説明します。

『Migrating to an OpenVMS AXP System: Planning for Migration』
**（翻訳版は『**OpenVMS AXP **オペレーティング・システムへの移行**: **システム移行の手引き』）**
RISC アーキテクチャの一般的な特性を示し，Alpha アーキテクチャと VAX アーキテクチャを比較し，移行プロセスの概要を示し，弊社が提供している移行ツールの概要を示します。このドキュメントの内容は，アプリケーションにとって最適な移行方法を定義するのに役立ちます。

『Migrating to an OpenVMS AXP System: Recompiling and Relinking Applications
**（翻訳版は『**OpenVMS AXP **オペレーティング・システムへの移行**: **再コンパイルと再リンク』）**
高級言語アプリケーションを OpenVMS Alpha に移行するプログラマを対象に，詳細な技術情報を示します。アプリケーションの移行を容易にするための開発環境の設定方法を示し，プログラマが VAX アーキテクチャの要素に対するアプリケーションの依存性を識別するのに役立つ情報を提供し，これらの依存性を解決するのに役立つコンパイラ機能を紹介します。このドキュメントの各セクションでは，VAX アーキテ

クチャ機能に対する特定のアプリケーションの依存性，データ移植の問題点 (アライメントの問題点など)， VAX 共有メージの移植プロセスなどについて説明します。

『OpenVMS Alpha Guide to 64-Bit Addressing and VLM Features』
**（翻訳版は『**OpenVMS Alpha 64 **ビット・アドレッシングおよび** VLM **機能説明書』）**
OpenVMS Alpha オペレーティング・システムでの 64 ビット仮想アドレッシングおよび VLM (Very Large Memory) のサポートについて説明します。このドキュメントはシステム・プログラマおよびアプリケーション・プログラマを対象にしており，OpenVMS Alpha の 64 ビットおよび VLM 機能について，その特徴と利点を中心に説明します。また，これらの機能を利用して， 64 ビット・アドレスをサポートし，非常に大きい物理メモリを効率よく利用できるようにアプリケーション・プログラムを拡張する方法についても説明します。

『OpenVMS Alpha System Dump Analyzer Utility Manual』
System Dump Analyzer ユーティリティを使用して，システム障害を調べ，動作中の OpenVMS Alpha システムを確認する方法について説明します。 Alpha のプログラマはこのドキュメントを参照してください。 VAX のプログラマは『OpenVMS VAX System Dump Analyzer Utility Manual』を参照してください。

『OpenVMS AXP Device Support: Developer's Guide』
弊社が提供していないデバイス用に OpenVMS Alpha のドライバを開発する方法について説明します。

『OpenVMS AXP Device Support: Reference』
『Writing OpenVMS Alpha Device in C』用の参照情報を提供します。デバイス・ドライバのプログラミングで使用するデータ構造，マクロ，ルーチンについて説明します。

『OpenVMS Bad Block Locator Utility Manual』
Bad Block Locator ユーティリティを使用して，古いタイプのメディアで不良ブロックを検索する方法について説明します。

『OpenVMS Compatibility Between VAX and Alpha』
エンド・ユーザ，システム管理者，プログラマに提供される機能を中心に， VAX コンピュータと Alpha コンピュータで稼動する OpenVMS を比較します。

『OpenVMS Developer's Guide to VMSINSTAL』
VMSINSTAL コマンド・プロシージャについて説明し，弊社が推奨している標準に準拠したインストール・プロシージャを設計する場合のガイドラインを示します。このドキュメントは， OpenVMS オペレーティング・システムでレイヤード・ソフトウェア製品のインストール・プロシージャを設計する開発者を対象にしています。

『OpenVMS DIGITAL Standard Runoff Reference Manual』
DSR テキスト生成ユーティリティについて説明します。

『OpenVMS EDT Reference Manual』
EDT エディタの詳細な参照情報を示します。

『OpenVMS Exchange Utility Manual』
Exchange ユーティリティを使用して，外部フォーマットのボリュームと OpenVMS ネイティブ・ボリュームの間でファイルを転送する方法について説明します。

『OpenVMS Glossary』
ドキュメンテーション全体で使用している OpenVMS 固有の用語を定義します。

『OpenVMS Guide to Extended File Specifications』
**（翻訳版は**『OpenVMS Extended File Specifications **の手引き**』**）**
拡張ファイル指定の概要を示し，全体的な相違点，および拡張ファイル指定を OpenVMS 環境に導入した場合の影響について説明します。

『OpenVMS Master Index』
OpenVMS フル・ドキュメンテーション・セットに含まれるドキュメントから抽出した索引情報を提供します。

『OpenVMS National Character Set Utility Manual』
National Character Set ユーティリティを使用して NCS 定義ファイルを作成する方法について説明します。

『OpenVMS Obsolete Features Manual』
VMS V4.0 ～ V5.0 で廃止された DCL コマンド，システム・サービス， RTL ルーチン，ユーティリティを示します。 VMS V4.0 以降で廃止された DCL コマンド， RTL ルーチン，ユーティリティをまとめた付録もあります。

『OpenVMS Programming Environment Manual』
プログラミング環境を定義する弊社の製品およびツールの全般的な説明を示します。コンパイラ，リンカ，デバッガ， System Dump Analyzer，システム・サービス，ルーチン・ライブラリなどの機能やツールについて紹介します。

『OpenVMS Programming Interfaces: Calling a System Routine』
OpenVMS プログラミング・インタフェースについて説明し，ユーザ・プロシージャから OpenVMS システム・ルーチンを呼び出すための標準規約を定義します。さまざまな高級言語での Alpha および VAX データ・タイプのインプリメンテーションについても，このドキュメントに示しています。

『OpenVMS RTL DECtalk (DTK$) Manual』
OpenVMS ランタイム・ライブラリの DTK$機能に含まれる DECtalk サポート・ルーチンについて説明します。

『OpenVMS RTL Parallel Processing (PPL$) Manual』
OpenVMS ランタイム・ライブラリの PPL$機能に含まれる並列処理ルーチンについて説明します。 Alpha 固有のルーチンと VAX 固有のルーチンを示し，各システムで異なる動作を実行するルーチンについても説明します。

『OpenVMS Software Overview』
OpenVMS オペレーティング・システムの概要と，提供される一部の製品の概要を示します。

『OpenVMS SUMSLP Utility Manual』
SUMSLP バッチ指向エディタを使用して，操作ファイルを更新する方法について説明します。

『OpenVMS System Messages and Recovery Procedures Reference Manual』
オペレーティング・システムから出されるエラー・メッセージ，警告メッセージ，情報メッセージをアルファベット順に示します。また，各メッセージの意味と，各メッセージに対するユーザの対処法も示します。このドキュメントは 2 分冊になっています。

『OpenVMS Terminal Fallback Utility Manual』
Terminal Fallback ユーティリティを使用して，このユーティリティで使用できるライブラリ，文字変換テーブル，ターミナル・パラメータを管理する方法について説明します。

『OpenVMS VAX Card Reader, Line Printer, and LPA11–K I/O User's Reference Manual』
OpenVMS VAX でのカード・リーダー，ラボラトリ周辺アクセラレータ，ライン・プリンタのドライバについて説明します。

『OpenVMS VAX Device Support Manual』
弊社が提供していないデバイス向けの OpenVMS VAX ドライバを開発する方法について説明します。

『OpenVMS VAX Device Support Reference Manual』
『OpenVMS VAX Device Support Manual』用の参照情報を提供します。デバイス・ドライバのプログラミングで使用されるデータ構造，マクロ，ルーチンについて説明します。

『OpenVMS VAX Patch Utility Manual』
Patch ユーティリティを使用して，OpenVMS VAX の実行イメージおよび共有イメージを調べ，変更する方法について説明します。

『OpenVMS Wide Area Network I/O User's Reference Manual』
OpenVMS VAX での DMC11/DMR11，DMP11，DMF32，DR11–W，DRV11-WA，DR32，非同期 DDCMP インタフェース・ドライバについて説明します。

『PDP–11 TECO User's Guide』
PDP–11 TECO (Text Editor and Corrector) プログラムの操作手順について説明します。

『POLYCENTER Software Installation Utility User's Guide』
POLYCENTER Software Installation ユーティリティについて説明します。このユーティリティは，ユーティリティと互換性のあるソフトウェア製品のインストールと管理のために使用される新しいコンポーネントです。

『TCP/IP Networking on OpenVMS Systems』
TCP/IP ネットワーキングの概要を示し，TCP/IP 機能に対する OpenVMS DCL のサポートについて説明します。

# 索引

## I



## K



## L



## M



## N



## O



## P



## R



## S

## T



## V



## W



## ア



## ク



## サ



## シ

HP OpenVMS V8.3 **新機能説明書**

---

2006 年 10 月　発行

日本ヒューレット・パッカード株式会社

〒 140-8641 東京都品川区東品川 2-2-24 天王洲セントラルタワー

電話　(03)5463-6600 (大代表)

---

BA322-90059