Linux Standard Base Core Specification 2.0.1
Copyright © 2004 Free Standards Group

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Portions of the text are copyrighted by the following parties:

• The Regents of the University of California
• Free Software Foundation
• Ian F. Darwin
• Paul Vixie
• BSDI (now Wind River)
• Andrew G Morgan
• Jean-loup Gailly and Mark Adler
• Massachusetts Institute of Technology

These excerpts are being used in accordance with their respective licenses.

Linux is a trademark of Linus Torvalds.
UNIX a registered trademark of the Open Group in the United States and other countries.
LSB is a trademark of the Free Standards Group in the USA and other countries.
AMD is a trademark of Advanced Micro Devices, Inc.
Intel and Itanium are registered trademarks and Intel386 is a trademarks of Intel Corporation.
OpenGL is a registered trademark of Silicon Graphics, Inc.
Specification Introduction
Specification Introduction
Table of Contents

Foreword ................................................................................................................................. i
Introduction .............................................................................................................................. ii
I. Introductory Elements........................................................................................................ 3
  1. Scope........................................................................................................................... 1
      1.1. General.............................................................................................................. 1
      1.2. Module Specific Scope ................................................................................... 1
  2. Normative References .................................................................................................. 2
  3. Requirements................................................................................................................ 5
      3.1. Relevant Libraries............................................................................................ 5
      3.2. LSB Implementation Conformance ............................................................... 5
      3.3. LSB Application Conformance ..................................................................... 6
  4. Definitions..................................................................................................................... 7
  5. Terminology................................................................................................................... 8
  6. Documentation Conventions ......................................................................................... 9
List of Tables

2-1. Normative References ......................................................................................................................... 2
3-1. Standard Library Names ....................................................................................................................... 5
3-2. Standard Library Names defined in the Architecture Specific Supplement ........................................ 5
Foreword

This is version 2.0.1 of the Linux Standard Base Core Specification. An implementation of this version of the specification may not claim to be an implementation of the Linux Standard Base unless it has successfully completed the compliance process as defined by the Free Standards Group.
Introduction

The LSB defines a binary interface for application programs that are compiled and packaged for LSB-conforming implementations on many different hardware architectures. Since a binary specification shall include information specific to the computer processor architecture for which it is intended, it is not possible for a single document to specify the interface for all possible LSB-conforming implementations. Therefore, the LSB is a family of specifications, rather than a single one.

This document should be used in conjunction with the documents it references. This document enumerates the system components it includes, but descriptions of those components may be included entirely or partly in this document, partly in other documents, or entirely in other reference documents. For example, the section that describes system service routines includes a list of the system routines supported in this interface, formal declarations of the data structures they use that are visible to applications, and a pointer to the underlying referenced specification for information about the syntax and semantics of each call. Only those routines not described in standards referenced by this document, or extensions to those standards, are described in the detail. Information referenced in this way is as much a part of this document as is the information explicitly included here.
I. Introductory Elements
Chapter 1. Scope

1.1. General
The Linux Standard Base (LSB) defines a system interface for compiled applications and a minimal environment for support of installation scripts. Its purpose is to enable a uniform industry standard environment for high-volume applications conforming to the LSB.

These specifications are composed of two basic parts: A common specification ("LSB-generic") describing those parts of the interface that remain constant across all implementations of the LSB, and an architecture-specific specification ("LSB-arch") describing the parts of the interface that vary by processor architecture. Together, the LSB-generic and the architecture-specific supplement for a single hardware architecture provide a complete interface specification for compiled application programs on systems that share a common hardware architecture.

The LSB-generic document shall be used in conjunction with an architecture-specific supplement. Whenever a section of the LSB-generic specification shall be supplemented by architecture-specific information, the LSB-generic document includes a reference to the architecture supplement. Architecture supplements may also contain additional information that is not referenced in the LSB-generic document.

The LSB contains both a set of Application Program Interfaces (APIs) and Application Binary Interfaces (ABIs). APIs may appear in the source code of portable applications, while the compiled binary of that application may use the larger set of ABIs. A conforming implementation shall provide all of the ABIs listed here. The compilation system may replace (e.g. by macro definition) certain APIs with calls to one or more of the underlying binary interfaces, and may insert calls to binary interfaces as needed.

The LSB is primarily a binary interface definition. Not all of the source level APIs available to applications may be contained in this specification.

1.2. Module Specific Scope
This is the Core module of the Linux Standards Base (LSB). This module provides the fundemental system interfaces, libraries, and runtime environment upon which all conforming applications and libraries depend.

Interfaces described in this module are mandatory except where explicitly listed otherwise. Core interfaces may be supplemented by other modules; all modules are built upon the core.
Chapter 2. Normative References

The specifications listed below are referenced in whole or in part by the Linux Standard Base. In this specification, where only a particular section of one of these references is identified, then the normative reference is to that section alone, and the rest of the referenced document is informative.

Table 2-1. Normative References

<table>
<thead>
<tr>
<th>Name</th>
<th>Title</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DWARF Debugging Information Format</td>
<td>DWARF Debugging Information Format, Revision 2.0.0 (July 27, 1993)</td>
<td><a href="http://www.eagercon.com/dwarf/dwarf-2.0.0.pdf">http://www.eagercon.com/dwarf/dwarf-2.0.0.pdf</a></td>
</tr>
<tr>
<td>Filesystem Hierarchy Standard</td>
<td>Filesystem Hierarchy Standard (FHS) 2.3</td>
<td><a href="http://www.pathname.com/fhs/">http://www.pathname.com/fhs/</a></td>
</tr>
<tr>
<td></td>
<td>ISO/IEC 9945-4:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 4: Rationale</td>
<td></td>
</tr>
<tr>
<td>Linux Allocated Device Registry</td>
<td>LINUX ALLOCATED DEVICES</td>
<td><a href="http://www.lanana.org/docs/device-">http://www.lanana.org/docs/device-</a></td>
</tr>
<tr>
<td>Name</td>
<td>Title</td>
<td>URL</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>PAM</td>
<td>Open Software Foundation, Request For Comments: 86.0 , October 1995, V. Samar &amp; R. Schemers (SunSoft)</td>
<td><a href="http://www.opengroup.org/tech/rfc/mirror-rfc/rfc86.0.txt">http://www.opengroup.org/tech/rfc/mirror-rfc/rfc86.0.txt</a></td>
</tr>
<tr>
<td>SVID Issue 4</td>
<td>System V Interface Definition, Fourth Edition</td>
<td></td>
</tr>
<tr>
<td>this specification</td>
<td>Linux Standard Base</td>
<td><a href="http://www.linuxbase.org/spec/">http://www.linuxbase.org/spec/</a></td>
</tr>
</tbody>
</table>
## Chapter 2. Normative References

<table>
<thead>
<tr>
<th>Name</th>
<th>Title</th>
<th>URL</th>
</tr>
</thead>
</table>
Chapter 3. Requirements

3.1. Relevant Libraries

1 The libraries listed in Table 3-1 shall be available on a Linux Standard Base system, with the specified runtime names.
2 The libraries listed in Table 3-2 are architecture specific, but shall be available on all LSB conforming systems. This list may be supplemented or amended by the architecture-specific specification.

Table 3-1. Standard Library Names

<table>
<thead>
<tr>
<th>Library</th>
<th>Runtime Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>libcrypt</td>
<td>libcrypt.so.1</td>
</tr>
<tr>
<td>libdl</td>
<td>libdl.so.2</td>
</tr>
<tr>
<td>libncurses</td>
<td>libncurses.so.5</td>
</tr>
<tr>
<td>libpthread</td>
<td>libpthread.so.0</td>
</tr>
<tr>
<td>libutil</td>
<td>libutil.so.1</td>
</tr>
<tr>
<td>libz</td>
<td>libz.so.1</td>
</tr>
<tr>
<td>libpam</td>
<td>libpam.so.0</td>
</tr>
<tr>
<td>libgcc_s</td>
<td>libgcc_s.so.1</td>
</tr>
</tbody>
</table>

Table 3-2. Standard Library Names defined in the Architecture Specific Supplement

<table>
<thead>
<tr>
<th>Library</th>
<th>Runtime Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>libc</td>
<td>See archLSB</td>
</tr>
<tr>
<td>libm</td>
<td>See archLSB</td>
</tr>
<tr>
<td>proginterp</td>
<td>See archLSB</td>
</tr>
</tbody>
</table>

These libraries will be in an implementation-defined directory which the dynamic linker shall search by default.

3.2. LSB Implementation Conformance

A conforming implementation shall satisfy the following requirements:

- The implementation shall implement fully the architecture described in the hardware manual for the target processor architecture.
- The implementation shall be capable of executing compiled applications having the format and using the system interfaces described in this document.
• The implementation shall provide libraries containing the interfaces specified by this document, and shall provide a
dynamic linking mechanism that allows these interfaces to be attached to applications at runtime. All the interfaces
shall behave as specified in this document.
• The map of virtual memory provided by the implementation shall conform to the requirements of this document.
• The implementation's low-level behavior with respect to function call linkage, system traps, signals, and other such
activities shall conform to the formats described in this document.
• The implementation shall provide all of the mandatory interfaces in their entirety.
• The implementation may provide one or more of the optional interfaces. Each optional interface that is provided
shall be provided in its entirety. The product documentation shall state which optional interfaces are provided.
• The implementation shall provide all files and utilities specified as part of this document in the format defined here
and in other referenced documents. All commands and utilities shall behave as required by this document. The
implementation shall also provide all mandatory components of an application's runtime environment that are
included or referenced in this document.
• The implementation, when provided with standard data formats and values at a named interface, shall provide the
behavior defined for those values and data formats at that interface. However, a conforming implementation may
consist of components which are separately packaged and/or sold. For example, a vendor of a conforming
implementation might sell the hardware, operating system, and windowing system as separately packaged items.
• The implementation may provide additional interfaces with different names. It may also provide additional
behavior corresponding to data values outside the standard ranges, for standard named interfaces.

3.3. LSB Application Conformance

A conforming application shall satisfy the following requirements:
• Its executable files are either shell scripts or object files in the format defined for the Object File Format system
interface.
• Its object files participate in dynamic linking as defined in the Program Loading and Linking System interface.
• It employs only the instructions, traps, and other low-level facilities defined in the Low-Level System interface as
being for use by applications.
• If it requires any optional interface defined in this document in order to be installed or to execute successfully, the
requirement for that optional interface is stated in the application's documentation.
• It does not use any interface or data format that is not required to be provided by a conforming implementation,
unless:
  • If such an interface or data format is supplied by another application through direct invocation of that application
during execution, that application is in turn an LSB conforming application.
  • The use of that interface or data format, as well as its source, is identified in the documentation of the application.
  • It shall not use any values for a named interface that are reserved for vendor extensions.
A strictly conforming application does not require or use any interface, facility, or implementation-defined extension
that is not defined in this document in order to be installed or to execute successfully.
Chapter 4. Definitions

For the purposes of this document, the following definitions, as specified in the ISO/IEC Directives, Part 2, 2001, 4th Edition, apply:

- **can**
  - be able to; there is a possibility of; it is possible to

- **cannot**
  - be unable to; there is no possibility of; it is not possible to

- **may**
  - is permitted; is allowed; is permissible

- **need not**
  - it is not required that; no...is required

- **shall**
  - is to; is required to; it is required that; has to; only...is permitted; it is necessary

- **shall not**
  - is not allowed [permitted] [acceptable] [permissible]; is required to be not; is required that...be not; is not to be

- **should**
  - it is recommended that; ought to

- **should not**
  - it is not recommended that; ought not to
Chapter 5. Terminology

For the purposes of this document, the following terms apply:

archLSB

The architectural part of the LSB Specification which describes the specific parts of the interface that are platform specific. The archLSB is complementary to the gLSB.

Binary Standard

The total set of interfaces that are available to be used in the compiled binary code of a conforming application.

gLSB

The common part of the LSB Specification that describes those parts of the interface that remain constant across all hardware implementations of the LSB.

implementation-defined

Describes a value or behavior that is not defined by this document but is selected by an implementor. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence of the value or behavior. An application that relies on such a value or behavior cannot be assured to be portable across conforming implementations. The implementor shall document such a value or behavior so that it can be used correctly by an application.

Shell Script

A file that is read by an interpreter (e.g., awk). The first line of the shell script includes a reference to its interpreter binary.

Source Standard

The set of interfaces that are available to be used in the source code of a conforming application.

undefined

Describes the nature of a value or behavior not defined by this document which results from use of an invalid program construct or invalid data input. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

unspecified

Describes the nature of a value or behavior not specified by this document which results from use of a valid program construct or valid data input. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

Other terms and definitions used in this document shall have the same meaning as defined in Chapter 3 of the Base Definitions volume of ISO POSIX (2003).
Chapter 6. Documentation Conventions

Throughout this document, the following typographic conventions are used:

- **function()**
  - the name of a function

- **command**
  - the name of a command or utility

- **CONSTANT**
  - a constant value

- **parameter**
  - a parameter

- **variable**
  - a variable

Throughout this specification, several tables of interfaces are presented. Each entry in these tables has the following format:

- **name**
  - the name of the interface

- **(symver)**
  - An optional symbol version identifier, if required.

- **[refno]**
  - A reference number indexing the table of referenced specifications that follows this table.

For example,

```plaintext
forkpty(GLIBC_2.0) [1]
```

refers to the interface named `forkpty` with symbol version `GLIBC_2.0` that is defined in the first of the listed references below the table.
ELF Specification
ELF Specification
Table of Contents

I. Low Level System Information ................................................................. 15
   1. Operating System Interface ............................................................... 1

II. Object Format ......................................................................................... 2
   2. Object Files .......................................................................................... 3
   3. Sections ............................................................................................... 4
      3.1. Sections Types .............................................................................. 4
      3.1.1. ELF Section Types ................................................................... 4
      3.1.2. Additional Section Types ....................................................... 7
   4. Special Sections ................................................................................... 8
      4.1. Special Sections .......................................................................... 8
      4.1.1. ELF Special Sections ............................................................. 8
      4.1.2. Additional Special Sections ............................................... 11
   5. Symbol Mapping .................................................................................. 13
      5.1. Symbol Mapping .......................................................................... 13
      5.1.1. C Language ............................................................................ 13
   6. DWARF Extensions .............................................................................. 14
   7. EH Frame Header ............................................................................... 16
      7.1. DWARF Exception Header Encoding ....................................... 17
   8. Symbol Versioning .............................................................................. 18
      8.1. Symbol Version Table ............................................................... 18
      8.2. Version Definitions ................................................................. 18
      8.3. Version Requirements ............................................................. 19
      8.4. Startup Sequence ....................................................................... 21
      8.5. Symbol Resolution ................................................................. 21
   9. ABI note tag ....................................................................................... 22

III. Dynamic Linking .................................................................................. 23
   10. Program Loading and Dynamic Linking ......................................... 24
   11. Program Header ................................................................................. 25
   12. Dynamic Entries ............................................................................... 26
      12.1. Dynamic Entries ....................................................................... 26
      12.1.1. ELF Dynamic Entries ......................................................... 26
      12.1.2. Additional Dynamic Entries ............................................ 28
# List of Tables

3-1. ELF Section Types .................................................................................................................. 4
3-2. Additional Section Types ........................................................................................................ 7
4-1. ELF Special Sections ............................................................................................................... 8
4-2. Additional Special Sections ................................................................................................. 11
6-1. Additional DWARF Call Frame Instructions ........................................................................ 14
7-1. .eh_frame_hdr Section Format ............................................................................................. 16
7-2. DWARF Exception Header value format .............................................................................. 17
7-3. DWARF Exception Header application .................................................................................. 17
11-1. Linux Segment Types .......................................................................................................... 25
List of Figures

8-1. Version Definition Entries.................................................................18
8-2. Version Definition Auxiliary Entries................................................19
8-3. Version Needed Entries....................................................................20
8-4. Version Needed Auxiliary Entries.....................................................20
I. Low Level System Information
Chapter 1. Operating System Interface

1 LSB-conforming applications shall assume that stack, heap and other allocated memory regions will be non-executable. The application must take steps to make them executable if needed.
II. Object Format
Chapter 2. Object Files

LSB-conforming implementations shall support the object file Executable and Linking Format (ELF), which is defined by the following documents:

- System V ABI
- System V ABI Update
- this document
- an architecture-specific LSB specification

Conforming implementations may also support other unspecified object file formats.
Chapter 3. Sections

As described in System V ABI, an ELF object file contains a number of sections.

3.1. Sections Types

The section header table is an array of Elf32_Shdr or Elf64_Shdr structures as described in System V ABI. The sh_type member shall be either a value from Table 3-1, drawn from the System V ABI, or one of the additional values specified in Table 3-2.

A section header's sh_type member specifies the section's semantics.

3.1.1. ELF Section Types

The following section types are defined in the System V ABI and the System V ABI Update.

Table 3-1. ELF Section Types

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHT_DYNAMIC</td>
<td>0x6</td>
<td>The section holds information for dynamic linking. Currently, an object file shall have only one dynamic section, but this restriction may be relaxed in the future. See ‘Dynamic Section’ in Chapter 5 for details.</td>
</tr>
<tr>
<td>SHT_DYNSYM</td>
<td>0xb</td>
<td>This section holds a minimal set of symbols adequate for dynamic linking. See also SHT_SYMTAB. Currently, an object file may have either a section of SHT_SYMTAB type or a section of SHT_DYNSYM type, but not both. This restriction may be relaxed in the future.</td>
</tr>
<tr>
<td>SHT_FINI_ARRAY</td>
<td>0xf</td>
<td>This section contains an array of pointers to termination functions, as described in ‘Initialization and Termination Functions’ in Chapter 5. Each pointer in the array is taken as a parameterless procedure with a void return.</td>
</tr>
<tr>
<td>SHT_HASH</td>
<td>0x5</td>
<td>The section holds a symbol hash table. Currently, an object file shall have only one hash table, but this restriction may be relaxed in the</td>
</tr>
<tr>
<td>Name</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>future. See `Hash Table' in the Chapter 5 for details.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHT_HIPROC</td>
<td>0x7fffffff</td>
<td>Values in this inclusive range are reserved for processor-specific semantics.</td>
</tr>
<tr>
<td>SHT_HIUSER</td>
<td>0xffffffff</td>
<td>This value specifies the upper bound of the range of indexes reserved for application programs. Section types between SHT_LOUSER and SHT_HIUSER can be used by the application, without conflicting with current or future system-defined section types.</td>
</tr>
<tr>
<td>SHT_INIT_ARRAY</td>
<td>0xe</td>
<td>This section contains an array of pointers to initialization functions, as described in `Initialization and Termination Functions' in Chapter 5. Each pointer in the array is taken as a parameterless procedure with a void return.</td>
</tr>
<tr>
<td>SHT_LOPROC</td>
<td>0x70000000</td>
<td>Values in this inclusive range are reserved for processor-specific semantics.</td>
</tr>
<tr>
<td>SHT_LOUSER</td>
<td>0x80000000</td>
<td>This value specifies the lower bound of the range of indexes reserved for application programs.</td>
</tr>
<tr>
<td>SHT_NOBITS</td>
<td>0x8</td>
<td>A section of this type occupies no space in the file but otherwise resembles SHT_PROGBITS. Although this section contains no bytes, the sh_offset member contains the conceptual file offset.</td>
</tr>
<tr>
<td>SHT_NOTE</td>
<td>0x7</td>
<td>The section holds information that marks the file in some way. See `Note Section' in Chapter 5 for details.</td>
</tr>
<tr>
<td>SHT_NULL</td>
<td>0x0</td>
<td>This value marks the section header as inactive; it does not have an associated section. Other members of the section header have undefined values.</td>
</tr>
<tr>
<td>Name</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SHT_PREINIT_ARRAY</td>
<td>0x10</td>
<td>This section contains an array of pointers to functions that are invoked before all other initialization functions, as described in 'Initialization and Termination Functions' in Chapter 5. Each pointer in the array is taken as a parameterless procedure with a void return.</td>
</tr>
<tr>
<td>SHT_PROGBITS</td>
<td>0x1</td>
<td>The section holds information defined by the program, whose format and meaning are determined solely by the program.</td>
</tr>
<tr>
<td>SHT_REL</td>
<td>0x9</td>
<td>The section holds relocation entries without explicit addends, such as type Elf32_Rel for the 32-bit class of object files or type Elf64_Rel for the 64-bit class of object files. An object file may have multiple relocation sections. See &quot;Relocation&quot;</td>
</tr>
<tr>
<td>SHT_RELA</td>
<td>0x4</td>
<td>The section holds relocation entries with explicit addends, such as type Elf32_Rela for the 32-bit class of object files or type Elf64_Rela for the 64-bit class of object files. An object file may have multiple relocation sections. See &quot;Relocation&quot;</td>
</tr>
<tr>
<td>SHT_SHLIB</td>
<td>0xa</td>
<td>This section type is reserved but has unspecified semantics.</td>
</tr>
<tr>
<td>SHT_STRTAB</td>
<td>0x3</td>
<td>The section holds a string table. An object file may have multiple string table sections. See `String Table' below for details.</td>
</tr>
<tr>
<td>SHT_SYMTAB</td>
<td>0x2</td>
<td>This section holds a symbol table. Currently, an object file may have either a section of SHT_SYMTAB type or a section of SHT_DYNSYM type, but not both. This restriction may be relaxed in the future. Typically, SHT_SYMTAB provides symbols for link editing, though it may also be used for dynamic linking. As a complete symbol table, it</td>
</tr>
</tbody>
</table>
may contain many symbols unnecessary for dynamic linking.

3.1.2. Additional Section Types

The following additional section types are defined here.

Table 3-2. Additional Section Types

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHT_GNU_verdef</td>
<td>0x6ffffffd</td>
<td>This section contains the symbol versions that are provided.</td>
</tr>
<tr>
<td>SHT_GNU_verneed</td>
<td>0x6ffffffe</td>
<td>This section contains the symbol versions that are required.</td>
</tr>
<tr>
<td>SHT_GNU_versym</td>
<td>0x6fffffff</td>
<td>This section contains the Symbol Version Table.</td>
</tr>
</tbody>
</table>
Chapter 4. Special Sections

4.1. Special Sections

Various sections hold program and control information. Sections in the lists below are used by the system and have the indicated types and attributes.

4.1.1. ELF Special Sections

The following sections are defined in the System V ABI and the System V ABI Update.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>.bss</td>
<td>SHT_NOBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.comment</td>
<td>SHT_PROGBITS</td>
<td>0</td>
</tr>
<tr>
<td>.data</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.data1</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.debug</td>
<td>SHT_PROGBITS</td>
<td>0</td>
</tr>
<tr>
<td>.dynamic</td>
<td>SHT_DYNAMIC</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.dynstr</td>
<td>SHT_STRTAB</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.dynsym</td>
<td>SHT_DYNSYM</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.fini</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_EXCEINSTR</td>
</tr>
<tr>
<td>.fini_array</td>
<td>SHT_FINI_ARRAY</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.hash</td>
<td>SHT_HASH</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.init</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_EXCEINSTR</td>
</tr>
<tr>
<td>.init_array</td>
<td>SHT_INIT_ARRAY</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.interp</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.line</td>
<td>SHT_PROGBITS</td>
<td>0</td>
</tr>
<tr>
<td>.note</td>
<td>SHT_NOTE</td>
<td>0</td>
</tr>
<tr>
<td>.preinit_array</td>
<td>SHT_PREINIT_ARRAY</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.rodata</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.rodata1</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.shstrtab</td>
<td>SHT_STRTAB</td>
<td>0</td>
</tr>
</tbody>
</table>
Chapter 4. Special Sections

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>.strtab</td>
<td>SHT_STRTAB</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.symtab</td>
<td>SHT_SYMTAB</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.text</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_EXECINSTR</td>
</tr>
</tbody>
</table>

This section holds data that contributes to the program's memory image. The program may treat this data as uninitialized. However, the system shall initialize this data with zeroes when the program begins to run. The section occupies no file space, as indicated by the section type, SHT_NOBITS.

This section holds version control information.

This section holds initialized data that contribute to the program's memory image.

This section holds initialized data that contribute to the program's memory image.

This section holds information for symbolic debugging. The contents are unspecified. All section names with the prefix .debug hold information for symbolic debugging. The contents of these sections are unspecified.

This section holds dynamic linking information. The section's attributes will include the SHF_ALLOC bit. Whether the SHF_WRITE bit is set is processor specific. See Chapter 5 for more information.

This section holds strings needed for dynamic linking, most commonly the strings that represent the names associated with symbol table entries. See Chapter 5 for more information.

This section holds the dynamic linking symbol table, as described in `Symbol Table'. See Chapter 5 for more information.

This section holds executable instructions that contribute to the process termination code. That is, when a program exits normally, the system arranges to execute the code in this section.

This section holds an array of function pointers that contributes to a single termination array for the executable or shared object containing the section.
Chapter 4. Special Sections

This section holds a symbol hash table. See `Hash Table' in Chapter 5 for more information.

This section holds executable instructions that contribute to the process initialization code. When a program starts to run, the system arranges to execute the code in this section before calling the main program entry point (called main for C programs).

This section holds an array of function pointers that contributes to a single initialization array for the executable or shared object containing the section.

This section holds the path name of a program interpreter. If the file has a loadable segment that includes relocation, the sections' attributes will include the SHF_ALLOC bit; otherwise, that bit will be off. See Chapter 5 for more information.

This section holds line number information for symbolic debugging, which describes the correspondence between the source program and the machine code. The contents are unspecified.

This section holds information in the format that `Note Section' in Chapter 5 describes of the System V Application Binary Interface, Edition 4.1.

This section holds an array of function pointers that contributes to a single pre-initialization array for the executable or shared object containing the section.

This section holds read-only data that typically contribute to a non-writable segment in the process image. See `Program Header' in Chapter 5 for more information.

This section holds read-only data that typically contribute to a non-writable segment in the process image. See `Program Header' in Chapter 5 for more information.

This section holds section names.

This section holds strings, most commonly the strings that represent the names associated with symbol table entries. If the file has a loadable segment that includes the symbol string table, the section's attributes will include the SHF_ALLOC bit; otherwise,
Chapter 4. Special Sections

This section holds a symbol table, as `Symbol Table'. in this chapter describes. If the file has a loadable segment that includes the symbol table, the section's attributes will include the SHF_ALLOC bit; otherwise, that bit will be off.

This section holds the `text,' or executable instructions, of a program.

4.1.2. Additional Special Sections

Object files in an LSB conforming application may also contain one or more of the additional special sections described below.

Table 4-2. Additional Special Sections

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>.ctors</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.dtors</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.eh_frame</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.eh_frame_hdr</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.gnu.version</td>
<td>SHT_GNU_versym</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.gnu.version_d</td>
<td>SHT_GNU_verdef</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.gnu.version_r</td>
<td>SHT_GNU_verneed</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.jcr</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.note.ABI-tag</td>
<td>SHT_NOTE</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.stab</td>
<td>SHT_PROGBITS</td>
<td>0</td>
</tr>
<tr>
<td>.stabstr</td>
<td>SHT_STRTAB</td>
<td>0</td>
</tr>
</tbody>
</table>

.ctors

This section contains a list of global constructor function pointers.

.dtors

This section contains a list of global destructor function pointers.

.eh_frame

This section contains information necessary for frame unwinding during exception handling.

.eh_frame_hdr

This section contains a pointer to the .eh_frame section which is accessible to the runtime support code of a C++ application. This section may also contain a binary search table which may be used by the runtime support code to more efficiently access records in the .eh_frame section.
This section contains the Symbol Version Table.

This section contains the Version Definitions.

This section contains the Version Requirements.

This section contains information necessary for registering compiled Java classes. The contents are compiler-specific and used by compiler initialization functions.

Specify ABI details.

This section contains debugging information. The contents are not specified as part of the LSB.

This section contains strings associated with the debugging information contained in the .stab section.
Chapter 5. Symbol Mapping

This chapter defines how names are mapped from the source symbol to the object symbol.

5.1. Symbol Mapping

Symbols in a source program are translated by the compilation system into symbols that exist in the object file. The rules for this translation are defined here.

5.1.1. C Language

External C symbols have the same names in C and object files' symbol tables.
In addition to the Call Frame Instructions defined in section 6.4.2 of DWARF Debugging Information Format, the following Call Frame Instructions may also be used.

Table 6-1. Additional DWARF Call Frame Instructions

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW_CFA_expression</td>
<td>0x10</td>
<td>The DW_CFA_expression instruction takes two operands: an unsigned LEB128 value representing a register number, and a DW_FORM_block value representing a DWARF expression. The required action is to establish the DWARF expression as the means by which the address in which the given register contents are found may be computed. The value of the CFA is pushed on the DWARF evaluation stack prior to execution of the DWARF expression. The DW_OP_call2, DW_OP_call4, DW_OP_call_ref and DW_OP_push_object_address DWARF operators (see Section 2.4.1 of DWARF Debugging Information Format) cannot be used in such a DWARF expression.</td>
</tr>
<tr>
<td>DW_CFA_offset_extended_sf</td>
<td>0x11</td>
<td>The DW_CFA_offset_extended_sf instruction takes two operands: an unsigned LEB128 value representing a register number and a signed LEB128 factored offset. This instruction is identical to DW_CFA_offset_extended except that the second operand is signed.</td>
</tr>
<tr>
<td>DW_CFA_def_cfa_sf</td>
<td>0x12</td>
<td>The DW_CFA_def_cfa_sf instruction takes two operands: an unsigned LEB128 value representing a register number and a signed LEB128 factored offset. This instruction is identical to DW_CFA_def_cfa except that the</td>
</tr>
</tbody>
</table>
## Chapter 6. DWARF Extensions

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW_CFA_def_cfa_offset_sf</td>
<td>0x13</td>
<td>The DW_CFA_def_cfa_offset_sf instruction takes a signed LEB128 operand representing a factored offset. This instruction is identical to DW_CFA_def_cfa_offset except that the operand is signed and factored.</td>
</tr>
<tr>
<td>DW_CFA_GNU_args_size</td>
<td>0x2e</td>
<td>The DW_CFA_def_cfa_offset_sf instruction takes an unsigned LEB128 operand representing an argument size.</td>
</tr>
<tr>
<td>DW_CFA_GNU_negative_offset_extended</td>
<td>0x2f</td>
<td>The DW_CFA_def_cfa_sf instruction takes two operands: an unsigned LEB128 value representing a register number and an unsigned LEB128 which represents the magnitude of the offset. This instruction is identical to DW_CFA_offset_extended_sf except that the operand is subtracted to produce the offset. This instruction is obsoleted by DW_CFA_offset_extended_sf.</td>
</tr>
</tbody>
</table>
Chapter 7. EH Frame Header

The .eh_frame_hdr section contains additional information about the .eh_frame section. A pointer to the start of the .eh_frame data, and optionally, a binary search table of pointers to the .eh_frame records are found in this section.

Data in this section is encoded according to the DWARF Exception Header Encoding described below.

<table>
<thead>
<tr>
<th>Encoding</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>unsigned byte</td>
<td>version</td>
</tr>
<tr>
<td>unsigned byte</td>
<td>eh_frame_ptr_enc</td>
</tr>
<tr>
<td>unsigned byte</td>
<td>fde_count_enc</td>
</tr>
<tr>
<td>unsigned byte</td>
<td>table_enc</td>
</tr>
<tr>
<td>encoded</td>
<td>eh_frame_ptr</td>
</tr>
<tr>
<td>encoded</td>
<td>fde_count</td>
</tr>
<tr>
<td></td>
<td>binary search table</td>
</tr>
</tbody>
</table>

version

Version of the .eh_frame_hdr format. This value shall be 1.

eh_frame_ptr_enc

The encoding format of the eh_frame_ptr field.

fde_count_enc

The encoding format of the fde_count field. A value of DW_EH_PE_omit indicates the binary search table is not present.

table_enc

The encoding format of the entries in the binary search table. A value of DW_EH_PE_omit indicates the binary search table is not present.
eh_frame_ptr

The encoded value of the pointer to the start of the .eh_frame section.
fde_count

The encoded value of the count of entries in the binary search table.
A binary search table containing fde_count entries. Each entry of the table consists of two encoded values, the initial location, and the address. The entries are sorted in an increasing order by the initial location value.

### 7.1. DWARF Exception Header Encoding

The DWARF Exception Header Encoding is used to describe the type of data used in the .eh_frame_hdr section. The upper 4 bits indicate how the value is to be applied. The lower 4 bits indicate the format of the data.

#### Table 7-2. DWARF Exception Header value format

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW_EH_PE_omit</td>
<td>0xff</td>
<td>No value is present.</td>
</tr>
<tr>
<td>DW_EH_PE_uleb128</td>
<td>0x01</td>
<td>Unsigned value is encoded using the Little Endian Base 128 (LEB128) as defined by DWARF Debugging Information Format.</td>
</tr>
<tr>
<td>DW_EH_PE_udata2</td>
<td>0x02</td>
<td>A 2 bytes unsigned value.</td>
</tr>
<tr>
<td>DW_EH_PE_udata4</td>
<td>0x03</td>
<td>A 4 bytes unsigned value.</td>
</tr>
<tr>
<td>DW_EH_PE_udata8</td>
<td>0x04</td>
<td>An 8 bytes unsigned value.</td>
</tr>
<tr>
<td>DW_EH_PE_sleb128</td>
<td>0x09</td>
<td>Signed value is encoded using the Little Endian Base 128 (LEB128) as defined by DWARF Debugging Information Format.</td>
</tr>
<tr>
<td>DW_EH_PE_sdata2</td>
<td>0x0A</td>
<td>A 2 bytes signed value.</td>
</tr>
<tr>
<td>DW_EH_PE_sdata4</td>
<td>0x0B</td>
<td>A 4 bytes signed value.</td>
</tr>
<tr>
<td>DW_EH_PE_sdata8</td>
<td>0x0C</td>
<td>An 8 bytes signed value.</td>
</tr>
</tbody>
</table>

#### Table 7-3. DWARF Exception Header application

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW_EH_PE_absptr</td>
<td>0x00</td>
<td>Value is used with no modification.</td>
</tr>
<tr>
<td>DW_EH_PE_pcrel</td>
<td>0x10</td>
<td>Value is relative to the current program counter.</td>
</tr>
<tr>
<td>DW_EH_PE_datarel</td>
<td>0x30</td>
<td>Value is relative to the beginning of the .eh_frame_hdr section.</td>
</tr>
<tr>
<td>DW_EH_PE_omit</td>
<td>0xff</td>
<td>No value is present.</td>
</tr>
</tbody>
</table>
Chapter 8. Symbol Versioning

This chapter describes the Symbol Versioning mechanism. All ELF objects may provide or depend on versioned symbols. Symbol Versioning is implemented by 3 section types: SHT_GNU_versym, SHT_GNU_verdef, and SHT_GNU_verneed.

The prefix Elfxx in the following descriptions and code fragments stands for either "Elf32" or "Elf64", depending on the architecture.

Versions are described by strings. The structures that are used for symbol versions also contain a member that holds the ELF hashing values of the strings. This allows for more efficient processing.

8.1. Symbol Version Table

The Symbol Version Table is contained in the special section .gnu.version which has a section type of SHT_GNU_versym. This section has the same number of entries as the Dynamic Symbol Table.

This section contains an array of elements of type Elfxx_Half. Each entry specifies the version defined for or required by the corresponding symbol in the Dynamic Symbol Table.

The values in the Symbol Version Table are unique to the object in which they are located. These values are identifiers that are provided by the the vna_other member of the Elfxx_Vernaux structure or the vd_ndx member of the Elfxx_Verdef structure.

The values 0 and 1 are reserved.

0

The symbol is local, not available outside the object.

1

The symbol is defined in this object and is globally available.

All other values are used to identify version strings located in one of the other Symbol Version sections. The value itself is not the version associated with the symbol. The string identified by the value defines the version of the symbol.

8.2. Version Definitions

Symbol definitions are contained in the special section .gnu.version_d which has a section type of SHT_GNU_verdef. The number of entries in this section is contained in the DT_VERDEFNUM entry of the Dynamic Section. The sh_link member of the section header points to the section that contains the strings referenced by this section.

Figure 8-1. Version Definition Entries

typedef struct {
   Elfxx_Half    vd_version;
   Elfxx_Half    vd_flags;
   Elfxx_Half    vd_ndx;
   Elfxx_Half    vd_cnt;
}
Chapter 8. Symbol Versioning

```c
    Elfxx_Word    vd_hash;
    Elfxx_Word    vd_aux;
    Elfxx_Word    vd_next;

    } Elfxx_Verdef;

    vd_version
    Version revision. This value is currently set to 1, and will be reset if the versioning implementation is
    incompatibly altered.

    vd_flags
    Version information flag bitmask.

    vd_ndx
    Version index numeric value referencing the SHT_GNU_versym section.

    vd_cnt
    Number of associated verdaux array entries.

    vd_hash
    Version name hash value (ELF hash function).

    vd_aux
    Offset to a corresponding entry in the verdaux array, in bytes.

    vd_next
    Offset to the next verdef entry, in bytes.

Figure 8-2. Version Definition Auxiliary Entries

typedef struct {
    Elfxx_Word    vda_name;
    Elfxx_Word    vda_next;
    } Elfxx_Verdaux;

    vda_name
    Offset to the version or dependency name string in the section header, in bytes.

    vda_next
    Offset to the next verdaux entry, in bytes.

8.3. Version Requirements

Symbol definitions are contained in the special section .gnu.version_r which has a section type of
SHT_GNU_verneed. The number of entries in this section is contained in the DT_VERNEEDNUM entry of the Dynamic
Section. The sh_link member of the section header points to the section that contains the strings referenced by this
section.
typedef struct {
    Elfxx_Half vn_version;
    Elfxx_Half vn_cnt;
    Elfxx_Word vn_file;
    Elfxx_Word vn_aux;
    Elfxx_Word vn_next;
} Elfxx_Verneed;

vn_version
Version of structure. This value is currently set to 1, and will be reset if the versioning implementation is
incompatibly altered.

vn_cnt
Number of associated verneed array entries.

vn_file
Offset to the file name string in the section header, in bytes.

vn_aux
Offset to a corresponding entry in the vernaux array, in bytes.

vn_next
Offset to the next verneed entry, in bytes.

typedef struct {
    Elfxx_Word vna_hash;
    Elfxx_Half vna_flags;
    Elfxx_Half vna_other;
    Elfxx_Word vna_name;
    Elfxx_Word vna_next;
} Elfxx_Vernaux;

vna_hash
Dependency name hash value (ELF hash function).

vna_flags
Dependency information flag bitmask.

vna_other
Object file version identifier used in the .gnu.version symbol version array. Bit number 15 controls whether or
not the object is hidden; if this bit is set, the object cannot be used and the static linker will ignore the symbol's
presence in the object.

vna_name
Offset to the dependency name string in the section header, in bytes.
8.4. Startup Sequence

When loading a sharable object, version definition data from the loaded object is analyzed to assure that it meets the version requirements of the calling object. The dynamic loader retrieves the entries in the caller's Elfxx_Verneed array and attempts to find matching definition information in the loaded Elfxx_Verdef table.

Each object and dependency is tested in turn. If a symbol definition is missing, the loader returns an error. A warning is issued instead of a hard error when the vna_flags bit for VER_FLG_WEAK is set in the Elfxx_Vernaux entry.

When the versions referenced by undefined symbols in the loaded object are found, version availability is certified. The test completes without error and the object is made available.

8.5. Symbol Resolution

When symbol versioning is used in an object, relocations extend the performance of definition testing beyond the simple match of symbol name strings: the version of the reference shall also equal the name of the definition. The same index that is used in the symbol table can be referenced in the SHT_GNU_versym section, and the value of this index is then used to acquire name data. The corresponding requirement string is retrieved from the Elfxx_Verneed array, and likewise, the corresponding definition string from the Elfxx_Verdef table.

Bit number 15 of the version symbol controls whether or not the object is hidden; if this bit is set, the object cannot be used and the static linker will ignore the symbol's presence in the object.

Results differ in the interaction of objects that variously use symbol versioning.

- The object with the reference and the object with the definitions may both use versioning. All described matching is processed in this case. A fatal error is triggered when no matching definition can be found in the object whose name is the one referenced by the vn_name element in the Elfxx_Verneed entry.
- The object with the reference may not use versioning, while the object with the definitions does. In this instance, only the definition with index numbers 1 and 2 will be used in the reference match, the same identified by the static linker as the base definition. In infrequent cases where the static linker was not used, as in calls to dlopen(), a version that does not have the base definition index is acceptable as long as it is the only version for which the symbol is defined.
- The object with the reference may use versioning, but the object with the definitions specifies none. A matching symbol is accepted in this case. A fatal error is triggered in the unlikely event that a corruption in the required symbols list obscured an outdated object file and caused a match on the object filename in the Elfxx_Verneed entry.
- Finally, both the object with the reference and the object with the definitions may not use versioning. The behavior in this instance defaults to pre-existing symbol rules.
Chapter 9. ABI note tag

Every executable shall contain a section named .note.ABI-tag of type SHT_NOTE. This section is structured as a note section as documented in the ELF spec. The section shall contain at least the following entry. The name field (namesz/name) contains the string "GNU". The type field shall be 1. The descsz field shall be at least 16, and the first 16 bytes of the desc field shall be as follows.

The first 32-bit word of the desc field shall be 0 (this signifies a Linux executable). The second, third, and fourth 32-bit words of the desc field contain the earliest compatible kernel version. For example, if the 3 words are 2, 2, and 5, this signifies a 2.2.5 kernel.
III. Dynamic Linking
Chapter 10. Program Loading and Dynamic Linking

LSB-conforming implementations shall support the object file information and system actions that create running programs as specified in the System V ABI and System V ABI Update and as supplemented by this document and an architecture-specific LSB specification.

Any shared object that is loaded shall contain sufficient DT_NEEDED records to satisfy the symbols on the shared library.
Chapter 11. Program Header

In addition to the Segment Types defined in the System V ABI and System V ABI Update the following Segment Types shall also be supported.

Table 11-1. Linux Segment Types

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PT_GNU_EH_FRAME</td>
<td>0x6474e550</td>
</tr>
<tr>
<td>PT_GNU_STACK</td>
<td>0x6474e551</td>
</tr>
</tbody>
</table>

PT_GNU_EH_FRAME

The array element specifies the location and size of the exception handling information as defined by the .eh_frame_hdr section.

PT_GNU_STACK

The p_flags member specifies the permissions on the segment containing the stack and is used to indicate whether the stack should be executable. The absence of this header indicates that the stack will be executable.
Chapter 12. Dynamic Entries

A dynamic entry’s \texttt{d_t.ag} member controls the interpretation of \texttt{d_un}.

12.1. Dynamic Entries

12.1.1. ELF Dynamic Entries

The following dynamic entries are defined in the System V ABI and System V ABI Update.

\begin{itemize}
\item \texttt{DT_BIND_NOW} \quad Process relocations of object
\item \texttt{DT_DEBUG} \quad For debugging; unspecified
\item \texttt{DT_FINI} \quad Address of termination function
\item \texttt{DT_HASH} \quad Address of symbol hash table
\item \texttt{DT_HIPROC} \quad End of processor-specific
\item \texttt{DT_INIT} \quad Address of init function
\item \texttt{DT_JMPREL} \quad Address of PLT relocs
\item \texttt{DT_LOPROC} \quad Start of processor-specific
\item \texttt{DT_NEEDED} \quad Name of needed library
\item \texttt{DT_NULL} \quad Marks end of dynamic section
\item \texttt{DT_PLTREL} \quad Type of reloc in PLT
\end{itemize}
Chapter 12. Dynamic Entries

25 DT_PLTRELSZ
26 Size in bytes of PLT relocations
27 DT_REL
28 Address of Rel relocations
29 DT_RELA
30 Address of Rela relocations
31 DT_RELAENT
32 Size of one Rela relocation
33 DT_RELASZ
34 Total size of Rela relocations
35 DT_RELENT
36 Size of one Rel relocation
37 DT_RELSZ
38 Total size of Rel relocations
39 DT_RPATH
40 Library search path
41 DT_SONAME
42 Name of shared object
43 DT_STRSZ
44 Size of string table
45 DT_STRTAB
46 Address of string table
47 DT_SYMBOLIC
48 Start symbol search here
49 DT_SYMENT
50 Size of one symbol table entry
51 DT_SYMTAB
52 Address of symbol table
53 DT_TEXTREL
54 Reloc might modify .text
12.1.2. Additional Dynamic Entries

The following dynamic entries are defined here.

DT_ADDRRNGHI

Values from DT_ADDRRNGLO through DT_ADDRRNGHI are reserved for definition by an archLSB.

DT_ADDRRNGLO

Values from DT_ADDRRNGLO through DT_ADDRRNGHI are reserved for definition by an archLSB.

DT_AUXILIARY

Shared object to load before self

DT_FILTER

Shared object to get values from

DT_FINI_ARRAY

The address of an array of pointers to termination functions.

DT_FINI_ARRAYSZ

Size in bytes of DT_FINI_ARRAY

DT_HIOS

Values from DT_LOOS through DT_HIOS are reserved for definition by specific operating systems.

DT_INIT_ARRAY

The address of an array of pointers to initialization functions.

DT_INIT_ARRAYSZ

Size in bytes of DT_INIT_ARRAY

DT_LOOS

Values from DT_LOOS through DT_HIOS are reserved for definition by specific operating systems.

DT_NUM

Number of dynamic entry tags defined (excepting reserved ranges).

DT_POSFLAG_1

Flags for DT_* entries, effecting the following DT_* entry

DT_RELCOUNT

All Elf32_Rel R_*_RELATIVE relocations have been placed into a single block and this entry specifies the number of entries in that block. This permits ld.so.1 to streamline the processing of RELATIVE relocations.
Chapter 12. Dynamic Entries

DT_SYMINENT
Entry size of syminfo

DT_SYMINFO
Address of the Syminfo table.

DT_SYMINFSZ
Size of syminfo table (in bytes)

DT_VALRNGHI
Entries which fall between DT_VALRNGHI & DT_VALRNGLO use the Dyn.d_un.d_val field of the Elf*_Dyn structure.

DT_VALRNGLO
Entries which fall between DT_VALRNGHI & DT_VALRNGLO use the Dyn.d_un.d_val field of the Elf*_Dyn structure.

DT_VERDEF
Address of version definition table

DT_VERDEFNUM
Number of version definitions

DT_VERNEED
Address of table with needed versions

DT_VERNEEDNUM
Number of needed versions

DT_VERSYM
Address of the table provided by the .gnu.version section.
Linux Standard Base Specification
Linux Standard Base Specification
# Table of Contents

I. Base Libraries

1. Libraries

1.1. Program Interpreter

1.2. Interfaces for libc

1.2.1. RPC

1.2.1.1. Interfaces for RPC

1.2.2. System Calls

1.2.2.1. Interfaces for System Calls

1.2.3. Standard I/O

1.2.3.1. Interfaces for Standard I/O

1.2.4. Signal Handling

1.2.4.1. Interfaces for Signal Handling

1.2.5. Localization Functions

1.2.5.1. Interfaces for Localization Functions

1.2.6. Socket Interface

1.2.6.1. Interfaces for Socket Interface

1.2.7. Wide Characters

1.2.7.1. Interfaces for Wide Characters

1.2.8. String Functions

1.2.8.1. Interfaces for String Functions

1.2.9. IPC Functions

1.2.9.1. Interfaces for IPC Functions

1.2.10. Regular Expressions

1.2.10.1. Interfaces for Regular Expressions

1.2.11. Character Type Functions

1.2.11.1. Interfaces for Character Type Functions

1.2.12. Time Manipulation

1.2.12.1. Interfaces for Time Manipulation

1.2.13. Terminal Interface Functions

1.2.13.1. Interfaces for Terminal Interface Functions

1.2.14. System Database Interface

1.2.14.1. Interfaces for System Database Interface

1.2.15. Language Support

1.2.15.1. Interfaces for Language Support

1.2.16. Large File Support

1.2.16.1. Interfaces for Large File Support

1.2.17. Standard Library

1.2.17.1. Interfaces for Standard Library

1.3. Data Definitions for libc

1.3.1. assert.h

1.3.2. ctype.h

1.3.3. dirent.h

1.3.4. errno.h
| 1.3.37. stdio.h | ................................................... | 44 |
| 1.3.36. stddef.h | ................................................... | 44 |
| 1.3.35. signal.h | ................................................... | 44 |
| 1.3.34. setjmp.h | ................................................... | 44 |
| 1.3.33. search.h | ................................................... | 44 |
| 1.3.32. sched.h | ................................................... | 44 |
| 1.3.31. rpc/xdr.h | ................................................... | 44 |
| 1.3.30. rpc/types.h | ................................................... | 44 |
| 1.3.29. rpc/svc.h | ................................................... | 44 |
| 1.3.28. rpc/rpc_msg.h | ................................................... | 44 |
| 1.3.27. rpc/epns.h | ................................................... | 44 |
| 1.3.26. rpc/auth.h | ................................................... | 44 |
| 1.3.25. regex.h | ................................................... | 44 |
| 1.3.24. pwd.h | ................................................... | 44 |
| 1.3.23. pty.h | ................................................... | 44 |
| 1.3.22. nl_types.h | ................................................... | 44 |
| 1.3.21. netinet/udp.h | ................................................... | 44 |
| 1.3.20. netinet/tcp.h | ................................................... | 44 |
| 1.3.19. netinet/in.h | ................................................... | 44 |
| 1.3.18. netdb.h | ................................................... | 44 |
| 1.3.17. net/if.h | ................................................... | 44 |
| 1.3.16. locale.h | ................................................... | 44 |
| 1.3.15. limits.h | ................................................... | 44 |
| 1.3.14. langinfo.h | ................................................... | 44 |
| 1.3.13. inttypes.h | ................................................... | 44 |
| 1.3.12. icp.h | ................................................... | 44 |
| 1.3.11. grp.h | ................................................... | 44 |
| 1.3.10. glob.h | ................................................... | 44 |
| 1.3.9. getopt.h | ................................................... | 44 |
| 1.3.8. ftw.h | ................................................... | 44 |
| 1.3.7. fnmatch.h | ................................................... | 44 |
| 1.3.6. fmtmsg.h | ................................................... | 44 |
| 1.3.5. fcntl.h | ................................................... | 44 |
| 1.3.4. file.h | ................................................... | 44 |
| 1.3.3. file.h | ................................................... | 44 |
| 1.3.2. file.h | ................................................... | 44 |
| 1.3.1. file.h | ................................................... | 44 |
## 1.4. Interface Definitions for libc

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>__stpcpy</td>
<td>78</td>
</tr>
<tr>
<td>__sigsetjmp</td>
<td>77</td>
</tr>
<tr>
<td>__register_atfork</td>
<td>77</td>
</tr>
<tr>
<td>__rawmemchr</td>
<td>76</td>
</tr>
<tr>
<td>__mempcpy</td>
<td>76</td>
</tr>
<tr>
<td>__libc_start_main</td>
<td>75</td>
</tr>
<tr>
<td>__libc_current_sigrtmin</td>
<td>74</td>
</tr>
<tr>
<td>__libc_current_sigrtmax</td>
<td>74</td>
</tr>
<tr>
<td>__isnanl</td>
<td>74</td>
</tr>
<tr>
<td>__isnanf</td>
<td>73</td>
</tr>
<tr>
<td>__isinf</td>
<td>72</td>
</tr>
<tr>
<td>__isninf</td>
<td>72</td>
</tr>
<tr>
<td>__isinf1</td>
<td>72</td>
</tr>
<tr>
<td>__isnan</td>
<td>73</td>
</tr>
<tr>
<td>__isnanf1</td>
<td>73</td>
</tr>
<tr>
<td>__isinf1l</td>
<td>74</td>
</tr>
<tr>
<td>__libc_current_sigrtmax</td>
<td>74</td>
</tr>
<tr>
<td>__libc_current_sigrtmin</td>
<td>74</td>
</tr>
<tr>
<td>__libc_start_main</td>
<td>75</td>
</tr>
<tr>
<td>__lxstat</td>
<td>75</td>
</tr>
<tr>
<td>__mempcpy</td>
<td>76</td>
</tr>
<tr>
<td>__rawmemchr</td>
<td>76</td>
</tr>
<tr>
<td>__register_atfork</td>
<td>77</td>
</tr>
<tr>
<td>__sigsetjmp</td>
<td>77</td>
</tr>
<tr>
<td>__stpcpy</td>
<td>78</td>
</tr>
<tr>
<td>__strdup</td>
<td>78</td>
</tr>
<tr>
<td>Function</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>fopen</td>
<td>113</td>
</tr>
<tr>
<td>freopen</td>
<td>113</td>
</tr>
<tr>
<td>getdomainname</td>
<td>114</td>
</tr>
<tr>
<td>gethostbyname_r</td>
<td>115</td>
</tr>
<tr>
<td>getloadavg</td>
<td>115</td>
</tr>
<tr>
<td>getopt</td>
<td>115</td>
</tr>
<tr>
<td>getopt_long</td>
<td>118</td>
</tr>
<tr>
<td>getopt_long_only</td>
<td>119</td>
</tr>
<tr>
<td>gettext</td>
<td>120</td>
</tr>
<tr>
<td>getutent</td>
<td>121</td>
</tr>
<tr>
<td>getutent_r</td>
<td>121</td>
</tr>
<tr>
<td>glob64</td>
<td>122</td>
</tr>
<tr>
<td>globfree64</td>
<td>124</td>
</tr>
<tr>
<td>initgroups</td>
<td>124</td>
</tr>
<tr>
<td>ioctl</td>
<td>126</td>
</tr>
<tr>
<td>sockio</td>
<td>127</td>
</tr>
<tr>
<td>kill</td>
<td>129</td>
</tr>
<tr>
<td>mbsnrtowcs</td>
<td>130</td>
</tr>
<tr>
<td>memmem</td>
<td>131</td>
</tr>
<tr>
<td>memrchr</td>
<td>132</td>
</tr>
<tr>
<td>ngettext</td>
<td>133</td>
</tr>
<tr>
<td>obstack_free</td>
<td>134</td>
</tr>
<tr>
<td>open</td>
<td>134</td>
</tr>
<tr>
<td>opterr</td>
<td>135</td>
</tr>
<tr>
<td>optind</td>
<td>135</td>
</tr>
<tr>
<td>optopt</td>
<td>135</td>
</tr>
<tr>
<td>pmap_getport</td>
<td>136</td>
</tr>
<tr>
<td>pmap_set</td>
<td>137</td>
</tr>
<tr>
<td>pmap_unset</td>
<td>137</td>
</tr>
<tr>
<td>psignal</td>
<td>138</td>
</tr>
<tr>
<td>random_r</td>
<td>138</td>
</tr>
<tr>
<td>setbuffer</td>
<td>139</td>
</tr>
<tr>
<td>setdomainname</td>
<td>139</td>
</tr>
<tr>
<td>setgroups</td>
<td>140</td>
</tr>
<tr>
<td>sethostid</td>
<td>141</td>
</tr>
<tr>
<td>sethostname</td>
<td>141</td>
</tr>
<tr>
<td>setsockopt</td>
<td>143</td>
</tr>
<tr>
<td>setutent</td>
<td>144</td>
</tr>
<tr>
<td>sigandset</td>
<td>145</td>
</tr>
<tr>
<td>sigblock</td>
<td>145</td>
</tr>
<tr>
<td>sigemptyset</td>
<td>146</td>
</tr>
<tr>
<td>sigemptyset</td>
<td>147</td>
</tr>
<tr>
<td>sigemptyset</td>
<td>148</td>
</tr>
<tr>
<td>sigreturn</td>
<td>149</td>
</tr>
<tr>
<td>stime</td>
<td>149</td>
</tr>
<tr>
<td>stpcpy</td>
<td>150</td>
</tr>
<tr>
<td>stpncpy</td>
<td>151</td>
</tr>
<tr>
<td>strcasestr</td>
<td>152</td>
</tr>
</tbody>
</table>

`xxxvi`
<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>strlen</td>
<td>152</td>
</tr>
<tr>
<td>strfry</td>
<td>153</td>
</tr>
<tr>
<td>strndup</td>
<td>153</td>
</tr>
<tr>
<td>strlen</td>
<td>154</td>
</tr>
<tr>
<td>strftime</td>
<td>155</td>
</tr>
<tr>
<td>strsep</td>
<td>156</td>
</tr>
<tr>
<td>strsignal</td>
<td>156</td>
</tr>
<tr>
<td>strtoq</td>
<td>157</td>
</tr>
<tr>
<td>strtoq</td>
<td>158</td>
</tr>
<tr>
<td>strverscmp</td>
<td>160</td>
</tr>
<tr>
<td>svc_register</td>
<td>161</td>
</tr>
<tr>
<td>svc_run</td>
<td>161</td>
</tr>
<tr>
<td>svc_sendreply</td>
<td>162</td>
</tr>
<tr>
<td>svctcp_create</td>
<td>162</td>
</tr>
<tr>
<td>svcudp_create</td>
<td>163</td>
</tr>
<tr>
<td>system</td>
<td>163</td>
</tr>
<tr>
<td>textdomain</td>
<td>164</td>
</tr>
<tr>
<td>unlink</td>
<td>165</td>
</tr>
<tr>
<td>vasprintf</td>
<td>165</td>
</tr>
<tr>
<td>vfprintf</td>
<td>166</td>
</tr>
<tr>
<td>verrx</td>
<td>167</td>
</tr>
<tr>
<td>vsyslog</td>
<td>167</td>
</tr>
<tr>
<td>wait3</td>
<td>168</td>
</tr>
<tr>
<td>wait4</td>
<td>168</td>
</tr>
<tr>
<td>waitpid</td>
<td>170</td>
</tr>
<tr>
<td>warn</td>
<td>171</td>
</tr>
<tr>
<td>warnx</td>
<td>172</td>
</tr>
<tr>
<td>wcpncpy</td>
<td>173</td>
</tr>
<tr>
<td>wcpncpy</td>
<td>173</td>
</tr>
<tr>
<td>wcscasecmp</td>
<td>174</td>
</tr>
<tr>
<td>wcsdup</td>
<td>174</td>
</tr>
<tr>
<td>wcscasecmp</td>
<td>174</td>
</tr>
<tr>
<td>wcscasecmp</td>
<td>175</td>
</tr>
<tr>
<td>wcsnlen</td>
<td>176</td>
</tr>
<tr>
<td>wcsnrtombs</td>
<td>177</td>
</tr>
<tr>
<td>wcstoq</td>
<td>178</td>
</tr>
<tr>
<td>wcstoq</td>
<td>179</td>
</tr>
<tr>
<td>xdr_u_int</td>
<td>179</td>
</tr>
</tbody>
</table>

1.5. Interfaces for libm

1.5.1. Math

1.5.1.1. Interfaces for Math

1.6. Data Definitions for libm

1.6.1. complex.h

1.6.2. math.h

1.7. Interfaces for libthread

1.7.1. Realtime Threads

1.7.1.1. Interfaces for Realtime Threads

1.7.2. Advanced Realtime Threads

1.7.2.1. Interfaces for Advanced Realtime Threads
II. Utility Libraries .............................................................. 214

1.8. Data Definitions for libpthread ........................................ 185
1.8.1. pthread.h ................................................................. 185
1.8.2. semaphore.h ............................................................ 187

1.9. Interface Definitions for libpthread .................................. 188
_pthread_cleanup_pop .................................................. 189
_pthread_cleanup_push ................................................. 189

1.10. Interfaces for libgcc_s ................................................ 189
1.10.1. Unwind Library .................................................. 189
1.10.1.1. Interfaces for Unwind Library .......................... 189

1.11. Data Definitions for libgcc_s ....................................... 189
1.11.1. unwind.h ............................................................. 189

1.12. Interfaces for libdl .................................................. 190
1.12.1. Dynamic Loader .................................................. 191
1.12.1.1. Interfaces for Dynamic Loader ......................... 191

1.13. Data Definitions for libdl ........................................... 191
1.13.1. dlfcn.h ............................................................... 191

1.14. Interface Definitions for libdl ..................................... 191
dladdr ................................................................. 192
dlopen ................................................................. 194
dlsym ................................................................. 194

1.15. Interfaces for libcrypt ............................................... 194
1.15.1. Encryption ......................................................... 195
1.15.1.1. Interfaces for Encryption ................................. 195

1.16. Interfaces for libpam .................................................. 195
1.16.1. Pluggable Authentication API ................................. 195
1.16.1.1. Interfaces for Pluggable Authentication API ....... 195

1.17. Data Definitions for libpam ......................................... 196
1.17.1. security/pam_appl.h ........................................... 196

1.18. Interface Definitions for libpam .................................. 197
pam_acct_mgt ......................................................... 198
pam_authorize ......................................................... 199
pam_chauthtok ......................................................... 201
pam_close_session ...................................................... 202
pam_end ............................................................... 203
pam_fail_delay .......................................................... 204
pam_get_item ............................................................ 205
pam_getenvlist .......................................................... 206
pam_open_session ...................................................... 207
pam_set_item ............................................................ 208
pam_setcred ............................................................ 210
pam_start .............................................................. 211
pam_strerror ........................................................... 212

II. Utility Libraries .............................................................. 214

2. utility Libraries .......................................................... 215
2.1. Interfaces for libz .................................................... 215
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1.1.</td>
<td>Compression Library</td>
<td>215</td>
</tr>
<tr>
<td>2.1.1.1</td>
<td>Interfaces for Compression Library</td>
<td>215</td>
</tr>
<tr>
<td>2.2.</td>
<td>Data Definitions for libz</td>
<td>216</td>
</tr>
<tr>
<td>2.2.1.</td>
<td>zlib.h</td>
<td>216</td>
</tr>
<tr>
<td>2.3.</td>
<td>Interfaces for libncurses</td>
<td>217</td>
</tr>
<tr>
<td>2.3.1.</td>
<td>Curses</td>
<td>218</td>
</tr>
<tr>
<td>2.3.1.1.</td>
<td>Interfaces for Curses</td>
<td>218</td>
</tr>
<tr>
<td>2.4.</td>
<td>Data Definitions for libncurses</td>
<td>220</td>
</tr>
<tr>
<td>2.4.1.</td>
<td>curses.h</td>
<td>220</td>
</tr>
<tr>
<td>2.5.</td>
<td>Interfaces for libutil</td>
<td>225</td>
</tr>
<tr>
<td>2.5.1.</td>
<td>Utility Functions</td>
<td>225</td>
</tr>
<tr>
<td>2.5.1.1.</td>
<td>Interfaces for Utility Functions</td>
<td>225</td>
</tr>
<tr>
<td>2.6.</td>
<td>Interface Definitions for libutil</td>
<td>226</td>
</tr>
<tr>
<td></td>
<td>forkpty</td>
<td>226</td>
</tr>
<tr>
<td></td>
<td>login</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td>login_tty</td>
<td>228</td>
</tr>
<tr>
<td></td>
<td>logout</td>
<td>229</td>
</tr>
<tr>
<td></td>
<td>logwtmp</td>
<td>230</td>
</tr>
<tr>
<td></td>
<td>openpty</td>
<td>231</td>
</tr>
<tr>
<td>III.</td>
<td>Commands and Utilities</td>
<td>232</td>
</tr>
<tr>
<td>3.</td>
<td>Commands and Utilities</td>
<td>233</td>
</tr>
<tr>
<td>3.1.</td>
<td>Commands and Utilities</td>
<td>233</td>
</tr>
<tr>
<td>3.2.</td>
<td>Command Behavior</td>
<td>234</td>
</tr>
<tr>
<td></td>
<td>ar</td>
<td>235</td>
</tr>
<tr>
<td></td>
<td>at</td>
<td>236</td>
</tr>
<tr>
<td></td>
<td>awk</td>
<td>236</td>
</tr>
<tr>
<td></td>
<td>batch</td>
<td>237</td>
</tr>
<tr>
<td></td>
<td>bc</td>
<td>237</td>
</tr>
<tr>
<td></td>
<td>chfn</td>
<td>238</td>
</tr>
<tr>
<td></td>
<td>chgrp</td>
<td>240</td>
</tr>
<tr>
<td></td>
<td>chown</td>
<td>240</td>
</tr>
<tr>
<td></td>
<td>chsh</td>
<td>241</td>
</tr>
<tr>
<td></td>
<td>col</td>
<td>241</td>
</tr>
<tr>
<td></td>
<td>cpio</td>
<td>241</td>
</tr>
<tr>
<td></td>
<td>crontab</td>
<td>242</td>
</tr>
<tr>
<td></td>
<td>cut</td>
<td>242</td>
</tr>
<tr>
<td></td>
<td>df</td>
<td>243</td>
</tr>
<tr>
<td></td>
<td>dmesg</td>
<td>243</td>
</tr>
<tr>
<td></td>
<td>du</td>
<td>244</td>
</tr>
<tr>
<td></td>
<td>echo</td>
<td>244</td>
</tr>
<tr>
<td></td>
<td>egrep</td>
<td>245</td>
</tr>
<tr>
<td></td>
<td>fgrep</td>
<td>245</td>
</tr>
<tr>
<td></td>
<td>file</td>
<td>245</td>
</tr>
<tr>
<td></td>
<td>find</td>
<td>246</td>
</tr>
<tr>
<td></td>
<td>fuser</td>
<td>246</td>
</tr>
<tr>
<td></td>
<td>gettext</td>
<td>247</td>
</tr>
<tr>
<td></td>
<td>grep</td>
<td>248</td>
</tr>
</tbody>
</table>
IV. Execution Environment

4. File System Hierarchy
   4.1. /dev

5. Additional Recommendations
   5.1. Minimal granted Directory and File permissions
   5.2. Recommendations for applications on ownership and permissions
      5.2.1. Directory Write Permissions
V. System Initialization ................................................................. 306

VI. Users & Groups ................................................................. 315

A. Alphabetical Listing of Interfaces .............................................. 318
List of Tables

1-1. libc Definition .......................................................... 1
1-2. libc - RPC Function Interfaces ........................................... 1
1-3. libc - System Calls Function Interfaces .................................. 2
1-4. libc - Standard I/O Function Interfaces ................................... 3
1-5. libc - Standard I/O Data Interfaces ....................................... 4
1-6. libc - Signal Handling Function Interfaces ................................. 4
1-7. libc - Signal Handling Data Interfaces ..................................... 5
1-8. libc - Localization Functions Function Interfaces ....................... 5
1-9. libc - Localization Functions Data Interfaces ......................... 6
1-10. libc - Socket Interface Function Interfaces ............................. 6
1-11. libc - Socket Interface Deprecated Function Interfaces ............. 6
1-12. libc - Wide Characters Function Interfaces ............................ 7
1-13. libc - String Functions Function Interfaces ............................ 8
1-14. libc - IPC Functions Function Interfaces ............................... 8
1-15. libc - Regular Expressions Function Interfaces ....................... 9
1-16. libc - Regular Expressions Deprecated Function Interfaces .......... 9
1-17. libc - Regular Expressions Deprecated Data Interfaces .............. 9
1-18. libc - Character Type Functions Function Interfaces ................ 10
1-19. libc - Time Manipulation Function Interfaces .......................... 10
1-20. libc - Time Manipulation Deprecated Function Interfaces ........... 11
1-21. libc - Time Manipulation Data Interfaces ................................ 11
1-22. libc - Terminal Interface Functions Function Interfaces ............ 11
1-23. libc - System Database Interface Function Interfaces ................ 12
1-24. libc - Language Support Function Interfaces .......................... 12
1-25. libc - Large File Support Function Interfaces .......................... 12
1-26. libc - Standard Library Function Interfaces ............................ 13
1-27. libc - Standard Library Data Interfaces .................................. 14
1-28. libc - Examples ................................................................ 160
1-29. libm Definition ............................................................ 180
1-30. libm - Math Function Interfaces ........................................... 180
1-31. libm - Math Data Interfaces ............................................... 182
1-32. libpthread Definition ....................................................... 183
1-33. libpthread - Posix Threads Function Interfaces ....................... 184
1-34. libgcc_s Definition ........................................................ 189
1-35. libdl Definition ............................................................ 190
1-36. libdl - Dynamic Loader Function Interfaces ............................. 191
1-37. libcrypt Definition ........................................................ 195
1-38. libcrypt - Encryption Function Interfaces .............................. 195
1-39. libpam Definition ............................................................ 195
1-40. libpam - Pluggable Authentication API Function Interfaces ........... 195
2-1. libz Definition ............................................................. 215
2-2. libz - Compression Library Function Interfaces ....................... 215
2-3. libncurses Definition ....................................................... 218
I. Base Libraries
Chapter 1. Libraries

An LSB-conforming implementation shall support some base libraries which provide interfaces for accessing the operating system, processor and other hardware in the system.

1.1. Program Interpreter

The Program Interpreter is specified in the appropriate architecture-specific LSB specification.

1.2. Interfaces for libc

Table 1-1 defines the library name and shared object name for the libc library.

<table>
<thead>
<tr>
<th>Library:</th>
<th>libc</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>See archLSB.</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications:

Large File Support
this specification
SUSv2
SVID Issue 3
SVID Issue 4

1.2.1. RPC

1.2.1.1. Interfaces for RPC

An LSB conforming implementation shall provide the generic functions for RPC specified in Table 1-2, with the full functionality as described in the referenced underlying specification.

|---------------------|----------------|---------------------|--------------|---------------|
Chapter 1. Libraries

Referenced Specification(s)
[1]. SVID Issue 4
[2]. this specification
[3]. SVID Issue 3

1.2.2. System Calls

1.2.2.1. Interfaces for System Calls

An LSB conforming implementation shall provide the generic functions for System Calls specified in Table 1-3, with the full functionality as described in the referenced underlying specification.

Table 1-3. libc - System Calls Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>__fxstat</td>
<td>chmod</td>
<td>getwd</td>
<td>read</td>
<td>setrlimit</td>
</tr>
<tr>
<td>__getpgid</td>
<td>fchown</td>
<td>initgroups</td>
<td>readdir</td>
<td>setrlimit64</td>
</tr>
<tr>
<td>__lxstat</td>
<td>fcntl</td>
<td>ioctl</td>
<td>readdir_r</td>
<td>setsid</td>
</tr>
<tr>
<td>__xmknod</td>
<td>fdatasync</td>
<td>kill</td>
<td>readlink</td>
<td>setuid</td>
</tr>
<tr>
<td>__xstat</td>
<td>flock</td>
<td>killpg</td>
<td>readv</td>
<td>sleep</td>
</tr>
<tr>
<td>access</td>
<td>fork</td>
<td>lchown</td>
<td>rename</td>
<td>statvfs</td>
</tr>
<tr>
<td>acct</td>
<td>fstatvfs</td>
<td>link</td>
<td>rmdir</td>
<td>stime</td>
</tr>
<tr>
<td>alarm</td>
<td>fsync</td>
<td>lockf</td>
<td>sbrk</td>
<td>symlink</td>
</tr>
<tr>
<td>brk</td>
<td>ftime</td>
<td>lseek</td>
<td>sched_get_priority_max</td>
<td>sync</td>
</tr>
<tr>
<td>chdir</td>
<td>ftruncate</td>
<td>mkdir</td>
<td>sched_get_priority_min</td>
<td>sysconf</td>
</tr>
<tr>
<td>chmod</td>
<td>getcontext</td>
<td>mkfifo</td>
<td>sched_getparam</td>
<td>time</td>
</tr>
<tr>
<td>chown</td>
<td>getegid</td>
<td>mlock</td>
<td>sched_getscheduler</td>
<td>times</td>
</tr>
</tbody>
</table>
Chapter 1. Libraries

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
</table>

Referenced Specification(s)

[1]. this specification


[3]. Large File Support

[4]. SUSv2

1.2.3. Standard I/O

1.2.3.1. Interfaces for Standard I/O

An LSB conforming implementation shall provide the generic functions for Standard I/O specified in Table 1-4, with the full functionality as described in the referenced underlying specification.

Table 1-4. libc - Standard I/O Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>telldir [2]</td>
</tr>
</tbody>
</table>
Chapter 1. Libraries

|----------------|-----------------|----------------|-------------|---------------|

Referenced Specification(s)
[1]. this specification
[3]. SUSv2

An LSB conforming implementation shall provide the generic data interfaces for Standard I/O specified in Table 1-5, with the full functionality as described in the referenced underlying specification.

Table 1-5. libc - Standard I/O Data Interfaces


Referenced Specification(s)

1.2.4. Signal Handling

1.2.4.1. Interfaces for Signal Handling

An LSB conforming implementation shall provide the generic functions for Signal Handling specified in Table 1-6, with the full functionality as described in the referenced underlying specification.

Table 1-6. libc - Signal Handling Function Interfaces

Chapter 1. Libraries

An LSB conforming implementation shall provide the generic data interfaces for Signal Handling specified in Table 1-7, with the full functionality as described in the referenced underlying specification.

Table 1-7. libc - Signal Handling Data Interfaces

| Interface         | 1.2.5. Localization Functions |

Referenced Specification(s)

[1]. this specification


[3]. SUSv2

An LSB conforming implementation shall provide the generic functions for Localization Functions specified in Table 1-8, with the full functionality as described in the referenced underlying specification.

Table 1-8. libc - Localization Functions Function Interfaces

| Interface         | 1.2.5. Localization Functions |

Referenced Specification(s)

[1]. this specification

An LSB conforming implementation shall provide the generic data interfaces for Localization Functions specified in Table 1-9, with the full functionality as described in the referenced underlying specification.

Table 1-9. libc - Localization Functions Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>_nl_msg_cat_cntr</td>
</tr>
</tbody>
</table>

Referenced Specification(s)
[1]. this specification

1.2.6. Socket Interface

1.2.6.1. Interfaces for Socket Interface
An LSB conforming implementation shall provide the generic functions for Socket Interface specified in Table 1-10, with the full functionality as described in the referenced underlying specification.

Table 1-10. libc - Socket Interface Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>__h_errno_location</td>
</tr>
</tbody>
</table>

Referenced Specification(s)
[1]. this specification

An LSB conforming implementation shall provide the generic deprecated functions for Socket Interface specified in Table 1-11, with the full functionality as described in the referenced underlying specification.

These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 1-11. libc - Socket Interface Deprecated Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>gethostbyname_r [1]</td>
</tr>
</tbody>
</table>

Referenced Specification(s)
[1]. this specification
1.2.7. Wide Characters

1.2.7.1. Interfaces for Wide Characters

An LSB conforming implementation shall provide the generic functions for Wide Characters specified in Table 1-12, with the full functionality as described in the referenced underlying specification.

Table 1-12. libc - Wide Characters Function Interfaces

|------------------|-----------------------|------------------------|-----------------------|------------------------|------------------------|

|------------------|-----------------------|------------------------|------------------------|------------------------|

<table>
<thead>
<tr>
<th></th>
<th>__wcstoul_internal [1]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>wcs cas ecmp [1]</td>
</tr>
<tr>
<td></td>
<td>wsr t oms [2]</td>
</tr>
<tr>
<td></td>
<td>wcs xfr m [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>btowc [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>putwc [2]</td>
</tr>
<tr>
<td></td>
<td>wescat [2]</td>
</tr>
<tr>
<td></td>
<td>wcspn [2]</td>
</tr>
<tr>
<td></td>
<td>wctob [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>fgetwc [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>putwchar [2]</td>
</tr>
<tr>
<td></td>
<td>wesc chr [2]</td>
</tr>
<tr>
<td></td>
<td>wesstr [2]</td>
</tr>
<tr>
<td></td>
<td>wctomb [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>fgetws [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>swprintf [2]</td>
</tr>
<tr>
<td></td>
<td>wescmp [2]</td>
</tr>
<tr>
<td></td>
<td>westod [2]</td>
</tr>
<tr>
<td></td>
<td>wctrans [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>fputwc [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>swscanf [2]</td>
</tr>
<tr>
<td></td>
<td>wscoll [2]</td>
</tr>
<tr>
<td></td>
<td>wstof [2]</td>
</tr>
<tr>
<td></td>
<td>wctype [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>fputws [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>towctrans [2]</td>
</tr>
<tr>
<td></td>
<td>wescpy [2]</td>
</tr>
<tr>
<td></td>
<td>westoimax [2]</td>
</tr>
<tr>
<td></td>
<td>wcwidth [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>fwide [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>towlower [2]</td>
</tr>
<tr>
<td></td>
<td>wces cspn [2]</td>
</tr>
<tr>
<td></td>
<td>westok [2]</td>
</tr>
<tr>
<td></td>
<td>wmemchr [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>fwprintf [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>towupper [2]</td>
</tr>
<tr>
<td></td>
<td>wcsdup [1]</td>
</tr>
<tr>
<td></td>
<td>westol [2]</td>
</tr>
<tr>
<td></td>
<td>wmemcmp [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>fwscanf [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ungetwc [2]</td>
</tr>
<tr>
<td></td>
<td>wesftime [2]</td>
</tr>
<tr>
<td></td>
<td>westold [2]</td>
</tr>
<tr>
<td></td>
<td>wmem cpy [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>getwc [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>vf wprintf [2]</td>
</tr>
<tr>
<td></td>
<td>wes len [2]</td>
</tr>
<tr>
<td></td>
<td>westoll [2]</td>
</tr>
<tr>
<td></td>
<td>wmemmove [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>getwchar [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>vf ws c anf [2]</td>
</tr>
<tr>
<td></td>
<td>wesncasecmp [1]</td>
</tr>
<tr>
<td></td>
<td>westombs [2]</td>
</tr>
<tr>
<td></td>
<td>wmemset [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>mb len [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>vswprintf [2]</td>
</tr>
<tr>
<td></td>
<td>wesncat [2]</td>
</tr>
<tr>
<td></td>
<td>westoq [1]</td>
</tr>
<tr>
<td></td>
<td>wprintf [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>mbr len [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>vswscanf [2]</td>
</tr>
<tr>
<td></td>
<td>wesncmp [2]</td>
</tr>
<tr>
<td></td>
<td>westoul [2]</td>
</tr>
<tr>
<td></td>
<td>wcsanf [2]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>mbrtowc [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>vwprintf [2]</td>
</tr>
<tr>
<td></td>
<td>wescmpy [2]</td>
</tr>
<tr>
<td></td>
<td>westoull [2]</td>
</tr>
</tbody>
</table>

Referenced Specification(s)

[1]. this specification
1.2.8. String Functions

1.2.8.1. Interfaces for String Functions

An LSB conforming implementation shall provide the generic functions for String Functions specified in Table 1-13, with the full functionality as described in the referenced underlying specification.

Table 1-13. libc - String Functions Function Interfaces

|---------------------|-------------|-------------------|-------------------|---------------|

Referenced Specification(s)

[1]. this specification

1.2.9. IPC Functions

1.2.9.1. Interfaces for IPC Functions

An LSB conforming implementation shall provide the generic functions for IPC Functions specified in Table 1-14, with the full functionality as described in the referenced underlying specification.

Table 1-14. libc - IPC Functions Function Interfaces

|----------|-------------|------------|------------|-------------|
Chapter 1. Libraries

Referenced Specification(s)

1.2.10. Regular Expressions

1.2.10.1. Interfaces for Regular Expressions

An LSB conforming implementation shall provide the generic functions for Regular Expressions specified in Table 1-15, with the full functionality as described in the referenced underlying specification.

Table 1-15. libc - Regular Expressions Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>regcomp [1]</td>
</tr>
<tr>
<td>regerror [1]</td>
</tr>
<tr>
<td>regexec [1]</td>
</tr>
<tr>
<td>regfree [1]</td>
</tr>
</tbody>
</table>

Referenced Specification(s)

An LSB conforming implementation shall provide the generic deprecated functions for Regular Expressions specified in Table 1-16, with the full functionality as described in the referenced underlying specification.

These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 1-16. libc - Regular Expressions Deprecated Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>advance [1]</td>
</tr>
<tr>
<td>re_comp [1]</td>
</tr>
<tr>
<td>re_exec [1]</td>
</tr>
<tr>
<td>step [1]</td>
</tr>
</tbody>
</table>

Referenced Specification(s)
[1]. SUSv2

An LSB conforming implementation shall provide the generic deprecated data interfaces for Regular Expressions specified in Table 1-17, with the full functionality as described in the referenced underlying specification.

These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 1-17. libc - Regular Expressions Deprecated Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>loc1 [1]</td>
</tr>
<tr>
<td>loc2 [1]</td>
</tr>
<tr>
<td>locs [1]</td>
</tr>
</tbody>
</table>

Referenced Specification(s)
[1]. SUSv2
1.2.11. Character Type Functions

1.2.11.1. Interfaces for Character Type Functions

An LSB conforming implementation shall provide the generic functions for Character Type Functions specified in Table 1-18, with the full functionality as described in the referenced underlying specification.

**Table 1-18. libc - Character Type Functions Function Interfaces**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>__ctype_b_loc(GLIBC_2.3) [1]</td>
<td></td>
</tr>
<tr>
<td>isalpha [2]</td>
<td></td>
</tr>
<tr>
<td>ispunct [2]</td>
<td></td>
</tr>
<tr>
<td>iswctype [2]</td>
<td></td>
</tr>
<tr>
<td>iswupper [2]</td>
<td></td>
</tr>
<tr>
<td>__ctype_get_mb_cur_max [1]</td>
<td></td>
</tr>
<tr>
<td>isascii [2]</td>
<td></td>
</tr>
<tr>
<td>isspace [2]</td>
<td></td>
</tr>
<tr>
<td>iswdigit [2]</td>
<td></td>
</tr>
<tr>
<td>iswxdigit [2]</td>
<td></td>
</tr>
<tr>
<td>__ctype_toupper_loc(GLIBC_2.3) [1]</td>
<td></td>
</tr>
<tr>
<td>iscntrl [2]</td>
<td></td>
</tr>
<tr>
<td>isupper [2]</td>
<td></td>
</tr>
<tr>
<td>iswgraph [2]</td>
<td></td>
</tr>
<tr>
<td>isxdigit [2]</td>
<td></td>
</tr>
<tr>
<td>__ctypetolower_loc(GLIBC_2.3) [1]</td>
<td></td>
</tr>
<tr>
<td>isdigit [2]</td>
<td></td>
</tr>
<tr>
<td>iswalnum [2]</td>
<td></td>
</tr>
<tr>
<td>iswlower [2]</td>
<td></td>
</tr>
<tr>
<td>toascii [2]</td>
<td></td>
</tr>
<tr>
<td>_tolower [2]</td>
<td></td>
</tr>
<tr>
<td>isgraph [2]</td>
<td></td>
</tr>
<tr>
<td>iswalpha [2]</td>
<td></td>
</tr>
<tr>
<td>iswprint [2]</td>
<td></td>
</tr>
<tr>
<td>tolower [2]</td>
<td></td>
</tr>
<tr>
<td>isalnum [2]</td>
<td></td>
</tr>
<tr>
<td>isprint [2]</td>
<td></td>
</tr>
<tr>
<td>iswcntrl [2]</td>
<td></td>
</tr>
<tr>
<td>iswspace [2]</td>
<td></td>
</tr>
</tbody>
</table>

*Referenced Specification(s)*

[1]. this specification

1.2.12. Time Manipulation

1.2.12.1. Interfaces for Time Manipulation

An LSB conforming implementation shall provide the generic functions for Time Manipulation specified in Table 1-19, with the full functionality as described in the referenced underlying specification.

**Table 1-19. libc - Time Manipulation Function Interfaces**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>adjtime [1]</td>
<td></td>
</tr>
<tr>
<td>ctime [2]</td>
<td></td>
</tr>
<tr>
<td>gmtime [2]</td>
<td></td>
</tr>
<tr>
<td>localtime_r [2]</td>
<td></td>
</tr>
<tr>
<td>ualarm [2]</td>
<td></td>
</tr>
<tr>
<td>asctime [2]</td>
<td></td>
</tr>
<tr>
<td>ctime_r [2]</td>
<td></td>
</tr>
<tr>
<td>gmtime_r [2]</td>
<td></td>
</tr>
<tr>
<td>mktime [2]</td>
<td></td>
</tr>
<tr>
<td>asctime_r [2]</td>
<td></td>
</tr>
<tr>
<td>difftime [2]</td>
<td></td>
</tr>
<tr>
<td>localtime [2]</td>
<td></td>
</tr>
<tr>
<td>tzset [2]</td>
<td></td>
</tr>
</tbody>
</table>

*Referenced Specification(s)*

[1]. this specification

An LSB conforming implementation shall provide the generic deprecated functions for Time Manipulation specified in Table 1-20, with the full functionality as described in the referenced underlying specification.
These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 1-20. libc - Time Manipulation Deprecated Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>adjtimex</td>
<td>[1]</td>
</tr>
</tbody>
</table>

Referenced Specification(s)

[1]. this specification

An LSB conforming implementation shall provide the generic data interfaces for Time Manipulation specified in Table 1-21, with the full functionality as described in the referenced underlying specification.

Table 1-21. libc - Time Manipulation Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>__daylight</td>
<td>[1]</td>
</tr>
<tr>
<td>__timezone</td>
<td>[1]</td>
</tr>
<tr>
<td>timezone</td>
<td>[2]</td>
</tr>
<tr>
<td>__timezone</td>
<td>[1]</td>
</tr>
<tr>
<td>daylight</td>
<td>[2]</td>
</tr>
<tr>
<td>tzname</td>
<td>[2]</td>
</tr>
</tbody>
</table>

Referenced Specification(s)

[1]. this specification

1.2.13. Terminal Interface Functions

1.2.13.1. Interfaces for Terminal Interface Functions

An LSB conforming implementation shall provide the generic functions for Terminal Interface Functions specified in Table 1-22, with the full functionality as described in the referenced underlying specification.

Table 1-22. libc - Terminal Interface Functions Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>cfgetispeed</td>
<td>[1]</td>
</tr>
<tr>
<td>cfsetispeed</td>
<td>[1]</td>
</tr>
<tr>
<td>tcdrain</td>
<td>[1]</td>
</tr>
<tr>
<td>tcgetattr</td>
<td>[1]</td>
</tr>
<tr>
<td>tcsendbreak</td>
<td>[1]</td>
</tr>
<tr>
<td>cfgetospeed</td>
<td>[1]</td>
</tr>
<tr>
<td>cfsetospeed</td>
<td>[1]</td>
</tr>
<tr>
<td>tcflow</td>
<td>[1]</td>
</tr>
<tr>
<td>tcgetpgrp</td>
<td>[1]</td>
</tr>
<tr>
<td>tcsetattr</td>
<td>[1]</td>
</tr>
<tr>
<td>cfmakeraw</td>
<td>[2]</td>
</tr>
<tr>
<td>cfsetspeed</td>
<td>[2]</td>
</tr>
<tr>
<td>tcflush</td>
<td>[1]</td>
</tr>
<tr>
<td>tcgetsid</td>
<td>[1]</td>
</tr>
<tr>
<td>tcsetpgrp</td>
<td>[1]</td>
</tr>
</tbody>
</table>

Referenced Specification(s)

[2]. this specification

1.2.14. System Database Interface

1.2.14.1. Interfaces for System Database Interface

An LSB conforming implementation shall provide the generic functions for System Database Interface specified in Table 1-23, with the full functionality as described in the referenced underlying specification.
Table 1-23. libc - System Database Interface Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
</table>

Referenced Specification(s)

[2]. this specification
[3]. SUSv2

1.2.15. Language Support

1.2.15.1. Interfaces for Language Support

An LSB conforming implementation shall provide the generic functions for Language Support specified in Table 1-24, with the full functionality as described in the referenced underlying specification.

Table 1-24. libc - Language Support Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>__obstack_free [1]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Referenced Specification(s)

[1]. this specification

1.2.16. Large File Support

1.2.16.1. Interfaces for Large File Support

An LSB conforming implementation shall provide the generic functions for Large File Support specified in Table 1-25, with the full functionality as described in the referenced underlying specification.

Table 1-25. libc - Large File Support Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
</table>
Chapter 1. Libraries

Referenced Specification(s)

[1]. this specification

[2]. Large File Support

1.2.17. Standard Library

1.2.17.1. Interfaces for Standard Library

An LSB conforming implementation shall provide the generic functions for Standard Library specified in Table 1-26, with the full functionality as described in the referenced underlying specification.

Table 1-26. libc - Standard Library Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>_Exit</td>
<td>dirname</td>
<td>glob</td>
<td>lsearch</td>
<td>srand</td>
</tr>
<tr>
<td>_assert_fail</td>
<td>div</td>
<td>glob64</td>
<td>makecontext</td>
<td>srand48</td>
</tr>
<tr>
<td>_cxa_atexit</td>
<td>drand48</td>
<td>globfree</td>
<td>malloc</td>
<td>srandom</td>
</tr>
<tr>
<td>_errno_location</td>
<td>ecvt</td>
<td>globfree64</td>
<td>memmem</td>
<td>strtod</td>
</tr>
<tr>
<td>_fpending</td>
<td>erand48</td>
<td>grantpt</td>
<td>mkstemp</td>
<td>strtol</td>
</tr>
<tr>
<td>_getpagesize</td>
<td>err</td>
<td>hcreate</td>
<td>mktmp</td>
<td>strtof</td>
</tr>
<tr>
<td>__isinf</td>
<td>error</td>
<td>hdestroy</td>
<td>mrand48</td>
<td>swapcontext</td>
</tr>
<tr>
<td>__isinf_f</td>
<td>errx</td>
<td>bssearch</td>
<td>nftw</td>
<td>syslog</td>
</tr>
<tr>
<td>__isinf_l</td>
<td>fcvt</td>
<td>htonl</td>
<td>nrand48</td>
<td>system</td>
</tr>
<tr>
<td>__isnan</td>
<td>fmtmsg</td>
<td>htons</td>
<td>ntohl</td>
<td>tddelete</td>
</tr>
<tr>
<td>__isnanf</td>
<td>fnmatch</td>
<td>imaxabs</td>
<td>ntohs</td>
<td>tfind</td>
</tr>
<tr>
<td>__isnanl</td>
<td>fpathconf</td>
<td>imaxdiv</td>
<td>openlog</td>
<td>tmpfile</td>
</tr>
<tr>
<td>__sysconf</td>
<td>free</td>
<td>inet_addr</td>
<td>perror</td>
<td>tmpnam</td>
</tr>
<tr>
<td>_exit</td>
<td>freeaddrinfo</td>
<td>inet_ntoa</td>
<td>posix_memalign</td>
<td>tssearch</td>
</tr>
<tr>
<td>_longjmp</td>
<td>fttrylockfile</td>
<td>inet_ntop</td>
<td>pthread</td>
<td>ttyname</td>
</tr>
<tr>
<td>_setjmp</td>
<td>ftw</td>
<td>inet_nton</td>
<td>putenv</td>
<td>ttyname_r</td>
</tr>
<tr>
<td>a64l</td>
<td>funlockfile</td>
<td>initstate</td>
<td>qsort</td>
<td>twalk</td>
</tr>
<tr>
<td>abort</td>
<td>gai_strerror</td>
<td>insqque</td>
<td>rand</td>
<td>unlockpt</td>
</tr>
<tr>
<td>abs</td>
<td>gcvt</td>
<td>isatty</td>
<td>rand_r</td>
<td>unsetenv</td>
</tr>
</tbody>
</table>
Referenced Specification(s)
[2]. this specification
[3]. SUSv2

An LSB conforming implementation shall provide the generic data interfaces for Standard Library specified in Table 1-27, with the full functionality as described in the referenced underlying specification.

Table 1-27. libc - Standard Library Data Interfaces


Referenced Specification(s)
[1]. this specification

1.3. Data Definitions for libc

This section defines global identifiers and their values that are associated with interfaces contained in libc. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content.

These definitions are intended to supplement those provided in the referenced underlying specifications.

This specification uses ISO/IEC 9899 C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.
1.3.1. assert.h

The assert.h header shall define the assert macro. It refers to the macro NDEBUG, which is not defined in this header. If NDEBUG is defined before the inclusion of this header, the assert macro shall be defined as described below, otherwise the macro shall behave as described in assert in ISO/IEC 9945 POSIX.

```c
#define assert(expr)    ((void)0)
```

1.3.2. ctype.h

```c
enum
{
  _ISupper, _ISlower, _ISalpha, _ISdigit, _ISxdigit, _ISspace, _ISprint,
  _ISgraph, _ISblank, _ISCntrl, _ISpunct, _ISalnum
};
```

1.3.3. dirent.h

```c
typedef struct __dirstream DIR;

struct dirent
{
  long d_ino;
  off_t d_off;
  unsigned short d_reclen;
  unsigned char d_type;
  char d_name[256];
};

struct dirent64
{
  uint64_t d_ino;
  int64_t d_off;
  unsigned short d_reclen;
  unsigned char d_type;
  char d_name[256];
};
```

1.3.4. errno.h

```c
#define errno   (*__errno_location())

#define EPERM   1
#define ECHILD  10
#define ENETDOWN        100
```
259  #define ENETUNREACH  101  
260  #define ENETRESET  102  
261  #define ECONNABORTED  103  
262  #define ECONNRESET  104  
263  #define ENOBUFS  105  
264  #define EISCONN  106  
265  #define ENOTCONN  107  
266  #define ESHUTDOWN  108  
267  #define ETOOMANYREFS  109  
268  #define EAGAIN  11  
269  #define ETIMEDOUT  110  
270  #define ECONNREFUSED  111  
271  #define EHOSTDOWN  112  
272  #define EHOSTUNREACH  113  
273  #define EALREADY  114  
274  #define EINVAL  115  
275  #define EUCLEAN  116  
276  #define ENOTNAM  117  
277  #define EREMOTEIO  118  
278  #define EMEDIUMTYPE  119  
279  #define EISNAM  120  
280  #define EREMOTEIO  121  
281  #define EDQUOT  122  
282  #define EDEADLK  123  
283  #define EEXIST  124  
284  #define ECANCELED  125  
285  #define ENAMETOOLONG  126  
286  #define EACCES  127  
287  #define EFAULT  128  
288  #define ENOTBLK  129  
289  #define EBUSY   130  
290  #define ENOMEDIUM  131  
291  #define ENOMEM  132  
292  #define EEXIST  133  
293  #define EXDEV   134  
294  #define EEXIST  135  
295  #define ENODEV  136  
296  #define ENOTDIR  137  
297  #define ENOTTY  138  
298  #define ETXTBSY  139  
299  #define EFBIG   140  
300  #define ENOSPC  141  
301  #define ESPIPE  142  
302  #define ESRCH   143  
303  #define EROFS   144  
304  #define EMLINK  145  
305  #define EPIPE   146  
306  #define EDOM    147  
307  #define ERANGE  148  
308  #define EDEADLK 149  
309  #define ENAMETOOLONG  150
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>312</td>
<td>#define ENOLCK</td>
<td>37</td>
</tr>
<tr>
<td>313</td>
<td>#define ENOSYS</td>
<td>38</td>
</tr>
<tr>
<td>314</td>
<td>#define ENOTEMPTY</td>
<td>39</td>
</tr>
<tr>
<td>315</td>
<td>#define EINTR</td>
<td>4</td>
</tr>
<tr>
<td>316</td>
<td>#define ELOOP</td>
<td>40</td>
</tr>
<tr>
<td>317</td>
<td>#define ENOMSG</td>
<td>42</td>
</tr>
<tr>
<td>318</td>
<td>#define EIDRM</td>
<td>43</td>
</tr>
<tr>
<td>319</td>
<td>#define ECHRNG</td>
<td>44</td>
</tr>
<tr>
<td>320</td>
<td>#define EL2NSYNC</td>
<td>45</td>
</tr>
<tr>
<td>321</td>
<td>#define EL3HLT</td>
<td>46</td>
</tr>
<tr>
<td>322</td>
<td>#define EL3RST</td>
<td>47</td>
</tr>
<tr>
<td>323</td>
<td>#define ELNRNG</td>
<td>48</td>
</tr>
<tr>
<td>324</td>
<td>#define EUNATCH</td>
<td>49</td>
</tr>
<tr>
<td>325</td>
<td>#define EIO</td>
<td>5</td>
</tr>
<tr>
<td>326</td>
<td>#define ENOANO</td>
<td>55</td>
</tr>
<tr>
<td>327</td>
<td>#define EADDRQC</td>
<td>56</td>
</tr>
<tr>
<td>328</td>
<td>#define EADSLT</td>
<td>57</td>
</tr>
<tr>
<td>329</td>
<td>#define EBFONT</td>
<td>59</td>
</tr>
<tr>
<td>330</td>
<td>#define ENXIO</td>
<td>6</td>
</tr>
<tr>
<td>331</td>
<td>#define ENOSTR</td>
<td>60</td>
</tr>
<tr>
<td>332</td>
<td>#define ENODATA</td>
<td>61</td>
</tr>
<tr>
<td>333</td>
<td>#define ETIME</td>
<td>62</td>
</tr>
<tr>
<td>334</td>
<td>#define ENSR</td>
<td>63</td>
</tr>
<tr>
<td>335</td>
<td>#define ENONET</td>
<td>64</td>
</tr>
<tr>
<td>336</td>
<td>#define ENOPKG</td>
<td>65</td>
</tr>
<tr>
<td>337</td>
<td>#define EREMOTE</td>
<td>66</td>
</tr>
<tr>
<td>338</td>
<td>#define ENOLINK</td>
<td>67</td>
</tr>
<tr>
<td>339</td>
<td>#define EADV</td>
<td>68</td>
</tr>
<tr>
<td>340</td>
<td>#define ESRMNT</td>
<td>69</td>
</tr>
<tr>
<td>341</td>
<td>#define E2BIG</td>
<td>7</td>
</tr>
<tr>
<td>342</td>
<td>#define ECOMM</td>
<td>70</td>
</tr>
<tr>
<td>343</td>
<td>#define EPROTO</td>
<td>71</td>
</tr>
<tr>
<td>344</td>
<td>#define EMULTIHOP</td>
<td>72</td>
</tr>
<tr>
<td>345</td>
<td>#define EDOTDOT</td>
<td>73</td>
</tr>
<tr>
<td>346</td>
<td>#define EBADSMSG</td>
<td>74</td>
</tr>
<tr>
<td>347</td>
<td>#define EOVERFLOW</td>
<td>75</td>
</tr>
<tr>
<td>348</td>
<td>#define ENOTUNIQ</td>
<td>76</td>
</tr>
<tr>
<td>349</td>
<td>#define EBADF</td>
<td>77</td>
</tr>
<tr>
<td>350</td>
<td>#define EREMCHG</td>
<td>78</td>
</tr>
<tr>
<td>351</td>
<td>#define ELIBACC</td>
<td>79</td>
</tr>
<tr>
<td>352</td>
<td>#define ENOEEXEC</td>
<td>8</td>
</tr>
<tr>
<td>353</td>
<td>#define ELIBBAD</td>
<td>80</td>
</tr>
<tr>
<td>354</td>
<td>#define ELIBSCN</td>
<td>81</td>
</tr>
<tr>
<td>355</td>
<td>#define ELIBMAX</td>
<td>82</td>
</tr>
<tr>
<td>356</td>
<td>#define ELIBEXEC</td>
<td>83</td>
</tr>
<tr>
<td>357</td>
<td>#define EILSEQ</td>
<td>84</td>
</tr>
<tr>
<td>358</td>
<td>#define ERESTART</td>
<td>85</td>
</tr>
<tr>
<td>359</td>
<td>#define ESTRPIPE</td>
<td>86</td>
</tr>
<tr>
<td>360</td>
<td>#define EUSERS</td>
<td>87</td>
</tr>
<tr>
<td>361</td>
<td>#define ENOTSOCK</td>
<td>88</td>
</tr>
<tr>
<td>362</td>
<td>#define EDESTADDRREQ</td>
<td>89</td>
</tr>
<tr>
<td>363</td>
<td>#define EBADF</td>
<td>9</td>
</tr>
<tr>
<td>364</td>
<td>#define EMSGSIZE</td>
<td>90</td>
</tr>
</tbody>
</table>
1.3.5. fcntl.h

#define O_RDONLY        00
#define O_ACCMODE       0003
#define O_WRONLY        01
#define O_CREAT 0100
#define O_TRUNC 01000
#define O_SYNC  010000
#define O_RDWR  02
#define O_EXCL  0200
#define O_APPEND        02000
#define O_ASYNC 020000
#define O_NOCTTY        0400
#define O_NDELAY        04000
#define O_NONBLOCK      04000
#define FD_CLOEXEC      1
#define F_DUPFD 0
#define F_RDLCK 0
#define F_GETFD 1
1.3.6. fmtmsg.h

#define MM_HARD 1
#define MM_NRECOV 128
#define MM_UTIL 16
#define MM_SOFT 2
#define MM_OPSYS 32
#define MM_FIRM 4
#define MM_RECOVER 64
#define MM_APPL 8
#define MM_NOSEV 0
#define MM_HALT 1
#define MM_ERROR 2
#define MM_NULLLBL ((char *) 0)

1.3.7. fnmatch.h

#define FNM_PATHNAME (1<<0)
#define FNM_NOESCAPE (1<<1)
#define FNM_PERIOD (1<<2)
#define FNM_NOMATCH 1

1.3.8. ftw.h

#define FTW_D FTW_D
#define FTW_DNR FTW_DNR
#define FTW_DP FTW_DP
#define FTW_F FTW_F
#define FTW_NS FTW_NS
#define FTW_SL FTW_SL
#define FTW_SLN FTW_SLN

enum
{
 FTW_F, FTW_D, FTW_DNR, FTW_NS, FTW_SL, FTW_DP, FTW_SLN
}
Chapter 1. Libraries

```
enum {
    FTW_PHYS, FTW_MOUNT, FTW_CHDIR, FTW_DEPTH
}

struct FTW {
    int base;
    int level;
}

typedef int (*__ftw_func_t) (char *__filename, struct stat * __status,
                           int __flag);
typedef int (*__ftw64_func_t) (char *__filename, struct stat64 * __status,
                              int __flag);
typedef int (*__nftw_func_t) (char *__filename, struct stat * __status,
                              int __flag, struct FTW * __info);
typedef int (*__nftw64_func_t) (char *__filename, struct stat64 * __status,
                              int __flag, struct FTW * __info);

1.3.9. getopt.h

#define no_argument     0
#define required_argument       1
#define optional_argument       2

struct option {
    char *name;
    int has_arg;
    int *flag;
    int val;
};

1.3.10. glob.h

#define GLOB_ERR           (1<<0)
#define GLOB_MARK           (1<<1)
#define GLOB_BRACE          (1<<10)
#define GLOB_NOMAGIC        (1<<11)
#define GLOB_TILDE          (1<<12)
#define GLOB_ONLYDIR        (1<<13)
#define GLOB_TILDE_CHECK    (1<<14)
#define GLOB_NOSORT         (1<<2)
#define GLOB_DOFFS          (1<<3)
```
#define GLOB_NOCHECK    (1<<4)
#define GLOB_APPEND     (1<<5)
#define GLOB_NOESCAPE   (1<<6)
#define GLOB_PERIOD     (1<<7)
#define GLOB_MAGCHAR    (1<<8)
#define GLOB_ALTDIRFUNC (1<<9)
#define GLOB_NOSPACE    1
#define GLOB_ABORTED    2
#define GLOB_NOMATCH    3
#define GLOB_NOSYS      4

typedef struct
{
    size_t gl_pathc;
    char **gl_pathv;
    size_t gl_offs;
    int gl_flags;
    void (*gl_closedir) (void *);
    struct dirent *( *gl_readdir) (void *);
    void *(*gl_opendir) (const char *);
    int *( *gl_lstat) (const char *, struct stat *);
    int *( *gl_stat) (const char *, struct stat *);
} glob_t;

typedef struct
{
    size_t gl_pathc;
    char **gl_pathv;
    size_t gl_offs;
    int gl_flags;
    void (*gl_closedir) (void *);
    struct dirent64 *( *gl_readdir64) (void *);
    void *(*gl_opendir) (const char *);
    int *( *gl_lstat) (const char *, struct stat *);
    int *( *gl_stat) (const char *, struct stat *);
} glob64_t;

1.3.11. grp.h

struct group
{
    char *gr_name;
    char *gr_passwd;
    gid_t gr_gid;
    char **gr_mem;
} ;
1.3.12. iconv.h

typedef void *iconv_t;

1.3.13. inttypes.h

typedef lldiv_t imaxdiv_t;
typedef unsigned char uint8_t;
typedef unsigned short uint16_t;
typedef unsigned int uint32_t;

1.3.14. langinfo.h

#define ABDAY_1 0x20000
#define ABDAY_2 0x20001
#define ABDAY_3 0x20002
#define ABDAY_4 0x20003
#define ABDAY_5 0x20004
#define ABDAY_6 0x20005
#define ABDAY_7 0x20006
#define DAY_1   0x20007
#define DAY_2   0x20008
#define DAY_3   0x20009
#define DAY_4   0x2000A
#define DAY_5   0x2000B
#define DAY_6   0x2000C
#define DAY_7   0x2000D
#define ABMON_1 0x2000E
#define ABMON_2 0x2000F
#define ABMON_3 0x20010
#define ABMON_4 0x20011
#define ABMON_5 0x20012
#define ABMON_6 0x20013
#define ABMON_7 0x20014
#define ABMON_8 0x20015
#define ABMON_9 0x20016
#define ABMON_10 0x20017
#define ABMON_11 0x20018
#define ABMON_12 0x20019
#define MON_1   0x2001A
#define MON_2   0x2001B
#define MON_3   0x2001C
#define MON_4   0x2001D
#define MON_5   0x2001E
#define MON_6   0x2001F
#define MON_7   0x20020
# define MON_8   0x20021
# define MON_9   0x20022
# define MON_10  0x20023
# define MON_11  0x20024
# define MON_12  0x20025

# define AM_STR  0x20026
# define PM_STR  0x20027

# define D_T_FMT 0x20028
# define D_FMT   0x20029
# define T_FMT   0x2002A
# define T_FMT_AMPM 0x2002B

# define ERA     0x2002C
# define ERA_D_FMT       0x2002E
# define ALT_DIGITS      0x2002F
# define ERA_D_T_FMT     0x20030
# define ERA_T_FMT       0x20031

# define CODESET 14

# define CRNCYSTR        0x4000F

# define RADIXCHAR       0x10000
# define THOUSEP 0x10001
# define YESEXPR 0x50000
# define NOEXPR  0x50001
# define YESSTR  0x50002
# define NOSTR   0x50003

1.3.15. limits.h

# define LLONG_MIN       (-LLONG_MAX-1LL)
# define ULLONG_MAX      18446744073709551615ULL
# define OPEN_MAX        256
# define PATH_MAX        4096
# define LLONG_MAX       9223372036854775807LL
# define SSIZE_MAX       LONG_MAX

# define MB_LEN_MAX      16

# define SCHAR_MIN       (-128)
# define SCHAR_MAX       127
# define UCHAR_MAX       255
# define CHAR_BIT        8

# define SHRT_MIN        (-32768)
# define SHRT_MAX        32767
# define USHRT_MAX       65535

# define LLONG_MIN       (-LLONG_MAX-1LL)
# define ULLONG_MAX      18446744073709551615ULL
# define OPEN_MAX        256
# define PATH_MAX        4096
# define LLONG_MAX       9223372036854775807LL
# define SSIZE_MAX       LONG_MAX

# define MB_LEN_MAX      16

# define SCHAR_MIN       (-128)
# define SCHAR_MAX       127
# define UCHAR_MAX       255
# define CHAR_BIT        8

# define SHRT_MIN        (-32768)
# define SHRT_MAX        32767
# define USHRT_MAX       65535
Chapter 1. Libraries

1.3.16. locale.h

```c
#define LC_CTYPE 0
#define LC_NUMERIC 1
#define LC_TELEPHONE 10
#define LC_MEASUREMENT 11
#define LC_IDENTIFICATION 12
#define LC_TIME 2
#define LC_COLLATE 3
#define LC_MONETARY 4
#define LC_MESSAGES 5
#define LC_ALL 6
#define LC_PAPER 7
#define LC_NAME 8
#define LC_ADDRESS 9

struct lconv
{
    char *decimal_point;
    char *thousands_sep;
    char *grouping;
    char *int_curr_symbol;
    char *currency_symbol;
    char *mon_decimal_point;
    char *mon_thousands_sep;
    char *mon_grouping;
    char *positive_sign;
    char *negative_sign;
    char int_frac_digits;
    char frac_digits;
    char p_cs_precedes;
    char p_sep_by_space;
    char n_cs_precedes;
    char n_sep_by_space;
    char p_sign_posn;
    char n_sign_posn;
    char *int_p_cs_precedes;
    char *int_p_sep_by_space;
    char *int_n_cs_precedes;
    char *int_n_sep_by_space;
    char *int_p_sign_posn;
    char *int_n_sign_posn;
};
```
typedef struct __locale_struct
{
    struct locale_data *__locales[13];
    const unsigned short *__ctype_b;
    const int *__ctypetolower;
    const int *__ctuptoupper;
    const char *__names[13];
} *__locale_t;

1.3.17. net/if.h

#define IF_NAMESIZE       16
#define IFF_UP            0x01
#define IFF_BROADCAST     0x02
#define IFF_DEBUG         0x04
#define IFF_LOOPBACK      0x08
#define IFF_POINTOPOINT   0x10
#define IFF_PROMISC       0x100
#define IFF_MULTICAST     0x1000
#define IFF_NOTRAILERS    0x20
#define IFF_RUNNING       0x40
#define IFF_NOARP         0x80

struct ifaddr
{
    struct sockaddr ifa_addr;
    union
    {
        struct sockaddr ifu_broadaddr;
        struct sockaddr ifu_dstaddr;
    }
    void *ifa_ifp;
    void *ifa_next;
};

#define IFNAMSIZ IF_NAMESIZE

struct ifreq
{
    union
    {
        char ifrn_name[IFNAMSIZ];
    }
    ifr_ifrn;
    union
    {
        struct sockaddr ifru_addr;
    }
struct sockaddr ifru_dstaddr;
struct sockaddr ifru_broadaddr;
struct sockaddr ifru_netmask;
struct sockaddr ifru_hwaddr;
short ifru_flags;
int ifru_ivalue;
int ifru_mtu;
char ifru_slave[IFNAMSIZ];
char ifru_newname[IFNAMSIZ];
caddr_t ifru_data;
struct ifmap ifru_map;
}
ifr_ifru;
;

struct ifconf
{
  int ifc_len;
  union
  {
    caddr_t ifcu_buf;
    struct ifreq *ifcu_req;
  }
  ifc_ifcu;
}
;

struct servent
{
  char *s_name;
  char **s_aliases;
  int s_port;
  char *s_proto;
}

1.3.18. netdb.h

#define h_errno (*__h_errno_location ())
#define NETDB_INTERNAL  -1
#define NETDB_SUCCESS   0
#define HOST_NOT_FOUND  1
#define IPPORT_RESERVED 1024
#define NI_MAXHOST      1025
#define TRY_AGAIN       2
#define NO_RECOVERY     3
#define NI_MAXSERV      32
#define NO_DATA 4
#define h_addr  h_addr_list[0]
#define NO_ADDRESS      NO_DATA

struct servent
{
  char *s_name;
  char **s_aliases;
  int s_port;
  char *s_proto;
}
;

struct hostent
{
    char *h_name;
    char **h_aliases;
    int h_addrtype;
    int h_length;
    char **h_addr_list;
}

struct protoent
{
    char *p_name;
    char **p_aliases;
    int p_proto;
}

struct netent
{
    char *n_name;
    char **n_aliases;
    int n_addrtype;
    unsigned int n_net;
}

#define AI_PASSIVE      0x0001
#define AI_CANONNAME    0x0002
#define AI_NUMERICHOST  0x0004

struct addrinfo
{
    int ai_flags;
    int ai_family;
    int ai_socktype;
    int ai_protocol;
    socklen_t ai_addrlen;
    struct sockaddr *ai_addr;
    char *ai_canonname;
    struct addrinfo *ai_next;
}

#define NI_NUMERICHOST  1
#define NI_DGRAM        16
#define NI_NUMERICSERV  2
#define NI_NOFQDN       4
#define NI_NAMEREQD     8
#define EAI_BADFLAGS    -1
#define EAI_MEMORY      -10
#define EAI_SYSTEM      -11
#define EAI_NONAME      -2
#define EAI_AGAIN       -3
#define EAI_FAIL        -4
#define EAI_NODATA      -5
#define EAI_FAMILY      -6
#define EAI_SOCKTYPE    -7
#define EAI_SERVICE     -8
#define EAI_ADDRFAMILY  -9

1.3.19. netinet/in.h

#define IPPROTO_IP      0
#define IPPROTO_ICMP    1
#define IPPROTO_UDP     17
#define IPPROTO_IGMP    2
#define IPPROTO_RAW     255
#define IPPROTO_IPV6    41
#define IPPROTO_ICMPV6  58
#define IPPROTO_TCP     6

typedef uint16_t in_port_t;

struct in_addr
{
    uint32_t s_addr;
};

typedef uint32_t in_addr_t;
#define INADDR_NONE     ((in_addr_t) 0xffffffff)
#define INADDR_BROADCAST        (0xffffffff)
#define INADDR_ANY      0

struct in6_addr
{
    union
    {
        uint8_t u6_addr8[16];
        uint16_t u6_addr16[8];
        uint32_t u6_addr32[4];
    }
    in6_u;
};
#define IN6ADDR_ANY_INIT        { { { 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 } } }
#define IN6ADDR_LOOPBACK_INIT   { { { 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1 } } }

#define INET_ADDRSTRLEN 16

struct sockaddr_in
{
    sa_family_t sin_family;
    unsigned short sin_port;
    struct in_addr sin_addr;
    unsigned char sin_zero[8];
};
Chapter 1. Libraries

```
#define INET6_ADDRSTRLEN 46

struct sockaddr_in6
{
  unsigned short sin6_family;
  uint16_t sin6_port;
  uint32_t sin6_flowinfo;
  struct in6_addr sin6_addr;
  uint32_t sin6_scope_id;
}

#define SOL_IP  0
#define IP_TOS  1
#define IPV6_UNICAST_HOPS 16
#define IPV6_MULTICAST_IF 17
#define IPV6_MULTICAST_HOPS 18
#define IPV6_MULTICAST_LOOP 19
#define IPV6_JOIN_GROUP 20
#define IPV6_LEAVE_GROUP 21
#define IPV6_V6ONLY    26
#define IP_MULTICAST_IF 32
#define IP_MULTICAST_TTL 33
#define IP_MULTICAST_LOOP 34
#define IP_ADD_MEMBERSHIP 35
#define IP_DROP_MEMBERSHIP 36

struct ipv6_mreq
{
  struct in6_addr ipv6mr_multiaddr;
  int ipv6mr_interface;
}

struct ip_mreq
{
  struct in_addr imr_multiaddr;
  struct in_addr imr_interface;
}
```

1.3.20. netinet/tcp.h

```
#define TCP_NODELAY 1
#define SOL_TCP 6
```

1.3.21. netinet/udp.h

```
#define SOL_UDP 17
```
1.3.22. nl_types.h

937 #define NL_CAT_LOCALE   1
938 #define NL_SETD 1
939
typedef void *nl_catd;
940
typedef int nl_item;

1.3.23. pty.h

944 struct winsize
945 {
946    unsigned short ws_row;
947    unsigned short ws_col;
948    unsigned short ws_xpixel;
949    unsigned short ws_ypixel;
950    unsigned short ws_ypixel;
951 }
952 ;

1.3.24. pwd.h

953 struct passwd
954 {
955    char *pw_name;
956    char *pw_passwd;
957    uid_t pw_uid;
958    gid_t pw gid;
959    char *pw_gecos;
960    char *pw_dir;
961    char *pw_shell;
962 
963 }
964 ;

1.3.25. regex.h

965 #define RE_BACKSLASH_ESCAPE_IN_LISTS   ((unsigned long int)1)
966 #define RE_BK_PLUS_QM   (RE_BACKSLASH_ESCAPE_IN_LISTS<<1)
967 #define RE_SYNTAX_AWK   (RE_BACKSLASH_ESCAPE_IN_LISTS|RE_DOT_NOT_NULL|RE_NO_BK_PARENS|
968   RE_NO_BK_REFS| RE_NO_BK_VBAR| RE_NO_EMPTY_RANGES| RE_DOT_NEWLINE|
969   RE_CONTEXT_INDEP_ANCHORS| RE_UNMATCHED_RIGHT_PAREN_ORD | RE_NO_GNU_OPS)
970 #define RE_CHAR_CLASSES (RE_BK_PLUS_QM<<1)
971 #define RE_SYNTAX_GREP
972 (RE_BK_PLUS_QM|RE_CHAR_CLASSES|RE_HAT_LISTS_NOT_NEWLINE|RE_INTERVALS|RE_NEWLINE_ALT)
974 #define RE_CONTEXT_INDEP_ANCHORS        (RE_CHAR_CLASSES<<1)
Chapter 1. Libraries

#define RE_SYNTAX_EGREP \(\text{RE_CHAR_CLASSES|RE_CONTEXT_INDEP_ANCHORS|RE_CONTEXT_INDEP_OPS|RE_HAT_LISTS_NOT_NEWLINE|RE_NEWLINE_ALT|RE_NO_BK_PARENS|RE_NO_BK_VBAR}\)
#define _RE_SYNTAX_POSIX_COMMON \(\text{RE_CHAR_CLASSES|RE_DOT_NE\\LINE|RE_DOT_NOT_NULL|RE_INTERVALS|RE_NO_EMPTY_RANGES}\)
#define RE_CONTEXT_INDEP_OPS \(\text{RE_CONTEXT_INDEP_ANCHORS<<1}\)
#define RE_CONTEXT_INVALID_OPS \(\text{RE_CONTEXT_INDEP_OPS<<1}\)
#define RE_DOT_NEWLINE \(\text{RE_CONTEXT_INVALID_OPS<<1}\)
#define RE_INVALID_INTERVAL_ORD \(\text{RE_DEBUG<<1}\)
#define RE_DOT_NOT_NULL \(\text{RE_DOT_NEWLINE<<1}\)
#define RE_HAT_LISTS_NOT_NEWLINE \(\text{RE_DOT_NOT_NULL<<1}\)
#define RE_INTERVALS \(\text{RE_HAT_LISTS_NOT_NEWLINE<<1}\)
#define RE_LIMITED_OPS \(\text{RE_INTERVALS<<1}\)
#define RE_NEWLINE_ALT \(\text{RE_LIMITED_OPS<<1}\)
#define RE_NO_BK_BRACES \(\text{RE_NEWLINE_ALT<<1}\)
#define RE_NO_BK_PARENS \(\text{RE_NO_BK_BRACES<<1}\)
#define RE_NO_BK_REFS \(\text{RE_NO_BK_PARENS<<1}\)
#define RE_NO_BK_VBAR \(\text{RE_NO_BK_REFS<<1}\)
#define RE_NO_EMPTY_RANGES \(\text{RE_NO_BK_VBAR<<1}\)
#define RE_UNMATCHED_RIGHT_PAREN_ORD \(\text{RE_NO_EMPTY_RANGES<<1}\)
#define RE_DEBUG \(\text{RE_NO_GNU_OPS<<1}\)
#define RE_NO_GNU_OPS \(\text{RE_NO_POSIX_BACKTRACKING<<1}\)
#define RE_SYNTAX_POSIX_EGREP \(\text{RE_SYNTAX_EGREP|RE_INTERVALS|RE_NO_BK_BRACES|RE_INVALID_INTERVAL_ORD}\)
#define RE_SYNTAX_POSIX_AWK \(\text{RE_SYNTAX_POSIX_EXTENDED|RE_BACKSLASH_ESCAPE_IN_LISTS|RE_INTERVALS|RE_NO_GNU_OPS}\)
#define RE_NO_POSIX_BACKTRACKING \(\text{RE_UNMATCHED_RIGHT_PAREN_ORD<<1}\)
#define RE_SYNTAX_POSIX_BASIC \(\text{RE_SYNTAX_POSIX_COMMON|RE_BK_PLUS_QM}\)
#define RE_SYNTAX_POSIX_EXTENDED \(\text{RE_SYNTAX_POSIX_COMMON|RE_CONTEXT_INDEP_ANCHORS|RE_CONTEXT_INDEP_OPS|RE_NO_BK_BRACES|RE_NO_BK_PARENS|RE_NO_BK_VBAR|RE_CONTEXT_INVALID_OPS|RE_UNMATCHED_RIGHT_PAREN_ORD}\)
#define RE_SYNTAX_POSIX_MINIMAL_EXTENDED \(\text{RE_SYNTAX_POSIX_COMMON|RE_CONTEXT_INDEP_ANCHORS|RE_CONTEXT_INVALID_OPS|RE_NO_BK_BRACES|RE_NO_BK_PARENS|RE_NO_BK_VBAR|RE_UNMATCHED_RIGHT_PAREN_ORD}\)
#define RE_SYNTAX_POSIX_MINIMAL_BASIC \(\text{RE_SYNTAX_POSIX_COMMON|RE_LIMITED_OPS}\)
#define RE_SYNTAX_ED RE_SYNTAX_POSIX_BASIC
#define RE_SYNTAX_SED RE_SYNTAX_POSIX_BASIC

typedef unsigned long reg_syntax_t;

typedef struct re_pattern_buffer {
  unsigned char *buffer;
  unsigned long allocated;
  unsigned long used;
  reg_syntax_t syntax;
  char *fastmap;
  char *translate;
  size_t re_nsub;
  unsigned int can_be_null:1;
  unsigned int regs_allocated:2;
  unsigned int fastmap_accurate:1;
  unsigned int no_sub:1;
}
Chapter 1. Libraries

unsigned int not_bol:1;
unsigned int not_eol:1;
unsigned int newline_anchor:1;

regex_t;
typedef int regoff_t;
typedef struct {
    regoff_t rm_so;
    regoff_t rm_eo;
} regmatch_t;
#define REG_NOTEOL      (1<<1)
#define REG_ICASE       (REG_EXTENDED<<1)
#define REG_NEWLINE     (REG_ICASE<<1)
#define REG_NOSUB       (REG_NEWLINE<<1)
#define REG_NOMATCH     -1
#define REG_EXTENDED    1
#define REG_NOTBOL      1

1.3.26. rpc/auth.h

enum auth_stat {
    AUTH_OK, AUTH_BADCRED = 1, AUTH_REJECTEDCRED = 2, AUTH_BADVERF = 3, AUTH_REJECTEDVERF = 4, AUTH_TOOWEAK = 5, AUTH_INVALIDRESP = 6, AUTH_FAILED = 7
};
union des_block {
    struct {
        u_int32_t high;
        u_int32_t low;
    } key;
    char c[8];
};
struct opaque_auth {
    enum_t oa_flavor;
    caddr_t oa_base;
    u_int oa_length;
};
typedef struct AUTH
struct opaque_auth ah_cred;
struct opaque_auth ah_verf;
union des_block ah_key;
struct auth_ops *ah_ops;
caddr_t ah_private;
}

AUTH;

struct auth_ops
{
    void (*ah_nextverf) (struct AUTH *);
    int (*ah_marshall) (struct AUTH *, XDR *);
    int (*ah_validate) (struct AUTH *, struct opaque_auth *);
    int (*ah_refresh) (struct AUTH *);
    void (*ah_destroy) (struct AUTH *);
}

1.3.27. rpc/clnt.h

#define clnt_control(cl,rq,in)  ((*(cl)->cl_ops->cl_control)(cl,rq,in))
#define clnt_abort(rh)  ((*(rh)->cl_ops->cl_abort)(rh))
#define clnt_call(rh, proc, xargs, argsp, xres, resp, secs)     ((*(rh)->cl_ops->cl_call)(rh,
proc, xargs, argsp, xres, resp, secs))
#define clnt_destroy(rh)        ((*(rh)->cl_ops->cl_destroy)(rh))
#define clnt_freeres(rh,xres,resp)      ((*(rh)->cl_ops->cl_freeres)(rh,xres,resp))
#define clnt_geterr(rh,errp)    ((*(rh)->cl_ops->cl_geterr)(rh, errp))
#define NULLPROC        ((u_long)0)
#define CLSET_TIMEOUT   1
#define CLGET_XID       10
#define CLSET_XID       11
#define CLGET_VERS      12
#define CLSET_VERS      13
#define CLGET_PROG      14
#define CLSET_PROG      15
#define CLGET_TIMEOUT   2
#define CLGET_SERVER_ADDR       3
#define CLSET_RETRY_TIMEOUT     4
#define CLGET_RETRY_TIMEOUT     5
#define CLGET_FD        6
#define CLGET_SVC_ADDR  7
#define CLSET_FD_CLOSE  8
#define CLSET_FD_NCLOSE 9

enum clnt_stat
{
    RPC_SUCCESS, RPC_CANTENCODEARGS = 1, RPC_CANTDECODERES = 2, RPC_CANTSEND =
    3, RPC_CANTRECV = 4, RPC_TIMEDOUT = 5, RPC_VERSMISMATCH =
    6, RPC_AUTHERROR = 7, RPC_PROGUNAVAIL = 8, RPC_PROGVERSISMATCH =
    9, RPC_PROCUNAVAIL = 10, RPC_CANTDECODEARGS = 11, RPC_SYSTEMERROR =
typedef struct CLIENT
{
    struct AUTH *cl_auth;
    struct clnt_ops *cl_ops;
    caddr_t cl_private;
} CLIENT;

struct clnt_ops
{
    enum clnt_stat (*cl_call) (struct CLIENT *, u_long, xdrproc_t, caddr_t, xdrproc_t, caddr_t, struct timeval);
    void (*cl_abort) (void);
    void (*cl_geterr) (struct CLIENT *, struct rpc_err *);
    bool_t (*cl_freeres) (struct CLIENT *, xdrproc_t, caddr_t);
    void (*cl_destroy) (struct CLIENT *);
    bool_t (*cl_control) (struct CLIENT *, int, char *);
};
1.3.28. rpc/rpc_msg.h

```c
1176 enum msg_type
1177 {  
1179    CALL, REPLY = 1
1180 }  
1181 ;  
1182 enum reply_stat
1183 {  
1184    MSG_ACCEPTED, MSG_DENIED = 1
1185 }  
1186 ;  
1187 enum accept_stat
1188 {  
1189    SUCCESS, PROG_UNAVAIL = 1, PROG_MISMATCH = 2, Proc_UNAVAIL =
1190    3, GARBAGE_ARGS = 4, SYSTEM_ERR = 5
1191 }  
1192 ;  
1193 enum reject_stat
1194 {  
1195    RPC_MISMATCH, AUTH_ERROR = 1
1196 }  
1197 ;
1198
1199 struct accepted_reply
1200 {  
1201    struct opaque_auth ar_verf;
1202    enum accept_stat ar_stat;
1203    union
1204    {  
1205        struct
1206        {  
1207            unsigned long low;
1208            unsigned long high;
1209        }
1210        AR_versions;
1211    struct
1212    {  
1213        caddr_t where;
1214        xdrproc_t proc;
1215    }
1216    AR_results;
1217    }  
1218    ru;
1219 }  
1220 ;  
1221 struct rejected_reply
1222 {  
1223    enum reject_stat rj_stat;
1224    union
```
{  
  struct  
  {  
    unsigned long low;  
    unsigned long high;  
  }  
  RJ_versions;  
  enum auth_stat RJ_why;  
}  
ru;  
};  

struct reply_body  
{  
  enum reply_stat rp_stat;  
  union  
  {  
    struct accepted_reply RP_ar;  
    struct rejected_reply RP_dr;  
  }  
  ru;  
};  

struct call_body  
{  
  unsigned long cb_rpcvers;  
  unsigned long cb_prog;  
  unsigned long cb_vers;  
  unsigned long cb_proc;  
  struct opaque_auth cb_cred;  
  struct opaque_auth cb_verf;  
};  
;

struct rpc_msg  
{  
  unsigned long rm_xid;  
  enum msg_type rm_direction;  
  union  
  {  
    struct call_body RM_cmb;  
    struct reply_body RM_rmb;  
  }  
  ru;  
};  
;

1.3.29. rpc/svc.h
# define svc_freeargs(xprt, xargs, argsp) (*(xprt)->xp_ops->xp_freeargs)((xprt), (xargs), (argsp))
# define svc_getargs(xprt, xargs, argsp) (*(xprt)->xp_ops->xp_getargs)((xprt), (xargs), (argsp))
# define RPC_ANYSOCK -1

typedef struct SVCXPRT {
  int xp_sock;
  u_short xp_port;
  struct xp_ops *xp_ops;
  int xp_addrlen;
  struct sockaddr_in xp_raddr;
  struct opaque_auth xp_verf;
  caddr_t xp_p1;
  caddr_t xp_p2;
  char xp_pad[256];
} SVCXPRT;

struct svc_req {
  rpcprog_t rq_prog;
  rpcvers_t rq_vers;
  rpcproc_t rq_proc;
  struct opaque_auth rq_cred;
  caddr_t rq_clntcred;
  SVCXPRT *rq_xprt;
};

typedef void (__dispatch_fn_t) (struct svc_req *, SVCXPRT *);

typedef int bool_t;
typedef int enum_t;
typedef unsigned long rpcprog_t;

1.3.30. rpc/types.h
typedef unsigned long rpcvers_t;
typedef unsigned long rpcproc_t;
typedef unsigned long rpcprot_t;

1.3.31. rpc/xdr.h

enum xdr_op
{
    XDR_ENCODE, XDR_DECODE, XDR_FREE
};
typedef struct XDR {
    enum xdr_op x_op;
    struct xdr_ops *x_ops;
    caddr_t x_public;
    caddr_t x_private;
    caddr_t x_base;
    int x_handy;
} XDR;

struct xdr_ops {
    bool_t (*x_getlong) (XDR * __xdrs, long *__lp);
    bool_t (*x_putlong) (XDR * __xdrs, long *__lp);
    bool_t (*x_getbytes) (XDR * __xdrs, caddr_t __addr, u_int __len);
    bool_t (*x_putbytes) (XDR * __xdrs, char *__addr, u_int __len);
    u_int (*x_getpostn) (XDR * __xdrs);
    bool_t (*x_setpostn) (XDR * __xdrs, u_int __pos);
    int32_t *(*x_inline) (XDR * __xdrs, int __len);
    void (*x_destroy) (XDR * __xdrs);
    bool_t (*x_getint32) (XDR * __xdrs, int32_t * __ip);
    bool_t (*x_putint32) (XDR * __xdrs, int32_t * __ip);
};
typedef bool_t (*xdrproc_t) (XDR *, void *, ...);

struct xdr_discrim {
    int value;
    xdrproc_t proc;
};

1.3.32. sched.h

#define SCHED_OTHER     0
#define SCHED_FIFO      1
1.3.33. search.h

typedef struct entry {
    char *key;
    void *data;
} ENTRY;

typedef enum {
    FIND, ENTER
} ACTION;

typedef enum {
    preorder, postorder, endorder, leaf
} VISIT;

typedef void (*__action_fn_t) (void *__nodep, VISIT __value, int __level);

1.3.34. setjmp.h

#define setjmp(env)     _setjmp(env)
#define sigsetjmp(a,b)  __sigsetjmp(a,b)

struct __jmp_buf_tag {
    __jmp_buf __jmpbuf;
    int __mask_was_saved;
    sigset_t __saved_mask;
}

typedef struct __jmp_buf_tag jmp_buf[1];
typedef jmp_buf sigjmp_buf;

1.3.35. signal.h

#define SIGRTMAX        (__libc_current_sigrtmax ())
#define SIGRTMIN        (__libc_current_sigrtmin ())
#define SIG_BLOCK 0
#define SIG_UNBLOCK 1
#define SIG_SETMASK 2
#define NSIG 65

typedef int sig_atomic_t;
struct sigstack
{
    void *ss_sp;
    int ss_onstack;
};
typedef void (*sighandler_t)(int);
#define SIG_HOLD ((sighandler_t)2)
#define SIG_ERR ((sighandler_t)-1)
#define SIG_DFL ((sighandler_t)0)
#define SIG_IGN ((sighandler_t)1)
#define SIGHUP 1
#define SIGUSR1 10
#define SIGSEGV 11
#define SIGUSR2 12
#define SIGPIPE 13
#define SIGALRM 14
#define SIGTERM 15
#define SIGSTKFLT 16
#define SIGCHLD 17
#define SIGCONT 18
#define SIGSTOP 19
#define SIGINT 2
#define SIGTSTP 20
#define SIGTTIN 21
#define SIGTTOU 22
#define SIGQUIT 3
#define SIGPWR 30
#define SIGSYS 31
#define SIGUNUSED 31
#define SIGILL 4
#define SIGTRAP 5
#define SIGABRT 6
#define SIGIOT 6
#define SIGBUS 7
#define SIGFPE 8
#define SIGKILL 9
#define SIGCLD SIGCHLD
#define SIGUNUSED 31
Chapter 1. Libraries

```c
#define SIGPOLL SIGIO
#define SV_ONSTACK (1<<0)
#define SV_INTERRUPT (1<<1)
#define SV_RESETHAND (1<<2)

typedef union sigval
{
    int sival_int;
    void *sival_ptr;
} signal_t;
#define SIGEV_SIGNAL 0
#define SIGEV_NONE 1
#define SIGEV_THREAD 2

typedef struct sigevent
{
    sigval_t sigev_value;
    int sigev_signo;
    int sigev_notify;
    union
    {
        int _pad[SIGEV_PAD_SIZE];
        struct
        {
            void (*sigev_thread_func) (sigval_t);
            void *attribute;
        }
        _sigev_thread;
    }
    _sigev_un;
} sigevent_t;
#define si_pid _sifields._kill._pid
#define si_uid _sifields._kill._uid
#define si_value _sifields._rt._sigval
#define si_int _sifields._rt._sigval.sival_int
#define si_ptr _sifields._rt._sigval.sival_ptr
#define si_status _sifields._sigchld._status
#define si_stime _sifields._sigchld._stime
#define si_utime _sifields._sigchld._utime
#define si_addr _sifields._sigfault._addr
#define si_band _sifields._sigpoll._band
#define si_fd _sifields._sigpoll._fd
#define si_timer1 _sifields._timer._timer1
#define si_timer2 _sifields._timer._timer2

typedef struct siginfo
{
    int si_signo;
    int si_errno;
    int si_code;
```

41
union
{
  int _pad[SI_PAD_SIZE];
struct
{
  pid_t _pid;
  uid_t _uid;
}
_kill;
struct
{
  unsigned int _timer1;
  unsigned int _timer2;
}
_timer;
struct
{
  pid_t _pid;
  uid_t _uid;
  sigval_t _sigval;
}
_rt;
struct
{
  pid_t _pid;
  uid_t _uid;
  int _status;
  clock_t _utime;
  clock_t _stime;
}
_sigchld;
struct
{
  void *-_addr;
}
_sigfault;
struct
{
  int _band;
  int _fd;
}
_sigpoll;
}
sifields;
siginfo_t;
#define SI_QUEUE        -1
#define SI_TIMER        -2
#define SI_MESGQ        -3
#define SI_ASYNCIO      -4
#define SI_SIGIO        -5
#define SI_TKILL        -6
#define SI_ASYNCNL      -60
# Chapter 1. Libraries

```
1572  #define SI_USER 0
1573  #define SI_KERNEL 0x80
1574
1575  #define ILL_ILLOPC 1
1576  #define ILL_ILLOPN 2
1577  #define ILL_ILLADR 3
1578  #define ILL_ILLTRP 4
1579  #define ILL_PRVOPC 5
1580  #define ILL_PRVREG 6
1581  #define ILL_COPROC 7
1582  #define ILL_BADSTK 8
1583
1584  #define FPE_INTDIV 1
1585  #define FPE_INTOVF 2
1586  #define FPE_FLTDIV 3
1587  #define FPE_FLTOVF 4
1588  #define FPE_FLTUND 5
1589  #define FPE_FLTRES 6
1590  #define FPE_FLTINV 7
1591  #define FPE_FLTSUB 8
1592
1593  #define SEGV_MAPERR 1
1594  #define SEGV_ACCERR 2
1595
1596  #define BUS_ADRALN 1
1597  #define BUS_ADRERR 2
1598  #define BUS_OBJERR 3
1599
1600  #define TRAP_BRKPT 1
1601  #define TRAP_TRACE 2
1602
1603  #define CLD_EXITED 1
1604  #define CLD_KILLED 2
1605  #define CLD_DUMPED 3
1606  #define CLD_TRAPPED 4
1607  #define CLD_STOPPED 5
1608  #define CLD_CONTINUED 6
1609
1610  #define POLL_IN 1
1611  #define POLL_OUT 2
1612  #define POLL_MSG 3
1613  #define POLL_ERR 4
1614  #define POLL_PRI 5
1615  #define POLL_HUP 6
1616
1617  typedef struct
1618  {
1619    unsigned long sig[_SIGSET_NWORDS];
1620  }
1621  sigset_t;
1622  #define SA_NOCLDSTOP 0x00000001
1623  #define SA_NOCLDWAIT 0x00000002
1624  #define SA_SIGINFO 0x00000004
```
#define SA_ONSTACK       0x08000000
#define SA_RESTART       0x10000000
#define SA_INTERRUPT     0x20000000
#define SA_NODEFER       0x40000000
#define SA_RESETHAND     0x80000000
#define SA_NOMASK        SA_NODEFER
#define SA_ONESHOT        SA_RESETHAND

typedef struct sigaltstack
{
    void *ss_sp;
    int ss_flags;
    size_t ss_size;
}
stack_t;
#define SS_ONSTACK         1
#define SS_DISABLE         2

1.3.36. stddef.h

#define offsetof(TYPE,MEMBER)   ((size_t)& ((TYPE*)0)->MEMBER)
#define NULL    (0L)

typedef int wchar_t;

1.3.37. stdio.h

#define EOF     (-1)
#define P_tmpdir        "/tmp"
#define FOPEN_MAX       16
#define L_tmpnam        20
#define FILENAME_MAX    4096
#define BUFSIZ  8192
#define L_ctermid       9
#define L_cuserid       9

typedef struct
{
    off_t __pos;
    mbstate_t __state;
}
fpos_t;
typedef struct
{
    off64_t __pos;
    mbstate_t __state;
}
fpos64_t;
typedef struct _IO_FILE FILE;
1.3.38. stdlib.h

```c
typedef int (*__compar_fn_t) (const void *, const void *);
struct random_data
{
    int32_t *fptr;
    int32_t *rptr;
    int32_t *state;
    int rand_type;
    int rand_deg;
    int rand_sep;
    int32_t *end_ptr;
};
```

```c
typedef struct
{
    int quot;
    int rem;
} div_t;
```

```c
typedef struct
{
    long quot;
    long rem;
} ldiv_t;
```

```c
typedef struct
{
    long long quot;
    long long rem;
} lldiv_t;
```

1.3.39. sys/file.h

```c
#define LOCK_SH 1
#define LOCK_EX 2
#define LOCK_NB 4
```
1.3.40. sys/ipc.h

1.3.41. sys/mman.h

1.3.42. sys/msg.h

1.3.43. sys/param.h

1.3.44. sys/poll.h
struct pollfd
{
    int fd;
    short events;
    short revents;
};

typedef unsigned long nfds_t;

1.3.45. sys/resource.h

define RUSAGE_CHILDREN (-1)
define RUSAGE_BOTH (-2)
define RLIM_INFINITY (~0UL)
define RLIM_SAVED_CUR -1
define RLIM_SAVED_MAX -1
define RLIMIT_CPU 0
define RUSAGE_SELF 0
define RLIMIT_FSIZE 1
define RLIMIT_DATA 2
define RLIMIT_STACK 3
define RLIMIT_CORE 4
define RLIMIT_NOFILE 7
define RLIMIT_AS 9
typedef unsigned long rlim_t;
typedef unsigned long long rlim64_t;
typedef int __rlimit_resource_t;

struct rlimit
{
    rlim_t rlim_cur;
    rlim_t rlim_max;
};

struct rlimit64
{
    rlim64_t rlim_cur;
    rlim64_t rlim_max;
};

struct rusage
{
    struct timeval ru_utime;
    struct timeval ru_stime;
    long ru_maxrss;
    long ru_ixrss;
    long ru_idrss;
    long ru_isrss;
    long ru_minflt;
1.3.46. sys/sem.h

```
#define SEM_UNDO  0x1000
#define GETPID  11
#define GETVAL  12
#define GETALL  13
#define GETNCNT 14
#define GETZCNT 15
#define SETVAL  16
#define SETALL  17

struct sembuf
{
    short sem_num;
    short sem_op;
    short sem_flg;
};
```

1.3.47. sys/shm.h

```
#define SHM_RDONLY      010000
#define SHM_W   0200
#define SHM_RND 020000
#define SHM_R   0400
#define SHM_REMAP       040000
#define SHM_LOCK        11
```
1.3.48. sys/socket.h

#define SHM_UNLOCK  12
#define SHUT_RD 0
#define MSG_WAITALL  0x100
#define MSG_TRUNC 0x20
#define MSG_EOR 0x80
#define SIOCGIFCONF 0x8912
#define SIOCGIFFLAGS 0x8913
#define SIOCGIFADDR 0x8915
#define SIOCGIFNETMASK 0x891b
#define MSG_OOB 1
#define SHUT_WR 1
#define MSG_PEEK 2
#define SHUT_RDWR 2
#define MSG_DONTROUTE 4
#define MSG_CTRUNC 8
#define PF_UNSPEC AF_UNSPEC

struct linger
{
    int l_onoff;
    int l_linger;
};

struct cmsghdr
{
    size_t cmsg_len;
    int cmsg_level;
    int cmsg_type;
};

struct iovec
{
    void *iov_base;
    size_t iov_len;
};

typedef unsigned short sa_family_t;
typedef unsigned int socklen_t;

struct sockaddr
{
    sa_family_t sa_family;
    char sa_data[14];
};

struct sockaddr_storage
{
}
Chapter 1. Libraries

1.3.49. sys/stat.h
#define S_ISBLK(m)      (((m)& S_IFMT)==S_IFBLK)
#define S_ISCHR(m)      (((m)& S_IFMT)==S_IFCHR)
#define S_ISDIR(m)      (((m)& S_IFMT)==S_IFDIR)
#define S_ISFIFO(m)     (((m)& S_IFMT)==S_IFIFO)
#define S_ISLNK(m)      (((m)& S_IFMT)==S_IFLNK)
#define S_ISREG(m)      (((m)& S_IFMT)==S_IFREG)
#define S_ISSOCK(m)     (((m)& S_IFMT)==S_IFSOCK)
#define S_TYPEISMQ(buf) ((buf)->st_mode - (buf)->st_mode)
#define S_TYPEISSEM(buf)  ((buf)->st_mode - (buf)->st_mode)
#define S_TYPEISSHM(buf)  ((buf)->st_mode - (buf)->st_mode)
#define S_IRWXU (S_IREAD|S_IWRITE|S_IEXEC)
#define S_IROTH (S_IRGRP>>3)
#define S_IRGRP (S_IRUSR>>3)
#define S_IRWXO (S_IRWXG>>3)
#define S_IRWXG (S_IRWXU>>3)
#define S_IWOTH (S_IWGRP>>3)
#define S_IWGRP (S_IWUSR>>3)
#define S_IXOTH (S_IXGRP>>3)
#define S_IXGRP (S_IXUSR>>3)
#define S_ISVTX 01000
#define S_IXUSR 0x0040
#define S_IWUSR 0x0080
#define S_IRUSR 0x0100
#define S_ISGID 0x0400
#define S_ISUID 0x0800
#define S_IFIFO 0x1000
#define S_IFCHR 0x20000
#define S_IFDIR 0x4000
#define S_IFBLK 0x6000
#define S_IFREG 0x8000
#define S_IFLNK 0xa000
#define S_IFSOCK 0xc000
#define S_IFMT  0xf000
#define st_atime        st_atim.tv_sec
#define st_ctime        st_ctim.tv_sec
#define st_mtime        st_mtim.tv_sec
#define S_IREAD S_IRUSR
#define S_IWRITE S_IWUSR
#define S_IEXEC S_IXUSR

1.3.50. sys/time.h

#define ITIMER_REAL     0
#define ITIMER_VIRTUAL  1
#define ITIMER_PROF     2

struct timezone
{
    int tz_minuteswest;
    int tz_dostime;
}

51
Chapter 1. Libraries

typedef int __itimer_which_t;

struct timespec {
    time_t tv_sec;
    long tv_nsec;
} ;

struct timeval {
    time_t tv_sec;
    suseconds_t tv_usec;
} ;

struct itimerval {
    struct timeval it_interval;
    struct timeval it_value;
} ;

1.3.51. sys/timeb.h

struct timeb {
    time_t time;
    unsigned short millitm;
    short timezone;
    short dstflag;
} ;

1.3.52. sys/times.h

struct tms {
    clock_t tms_utime;
    clock_t tms_stime;
    clock_t tms_cutime;
    clock_t tms_cstime;
} ;

1.3.53. sys/types.h
Chapter 1. Libraries

```c
#define FD_ISSET(d,set)   ((set)->fds_bits[((d)/(8*sizeof(long)))]&
   (1<<(d)%(8*sizeof(long))))
#define FD_CLR(d,set)     ((set)->fds_bits[((d)/(8*sizeof(long)))]&
   =~(1<<(d)%(8*sizeof(long))))
#define FD_SET(d,set)     ((set)->fds_bits[((d)/(8*sizeof(long)))]|=(1<<(d)%(8*sizeof(long))))
#define FALSE   0
#define TRUE    1
#define FD_SETSIZE      1024
#define FD_ZERO(fdsetp) bzero(fdsetp, sizeof(*(fdsetp)))

typedef signed char int8_t;
typedef short int16_t;
typedef int int32_t;
typedef unsigned char u_int8_t;
typedef unsigned short u_int16_t;
typedef unsigned int u_int32_t;
typedef unsigned int uid_t;
typedef int pid_t;
typedef unsigned long off_t;
typedef int key_t;
typedef long suseconds_t;
typedef unsigned int u_int;
typedef struct
   {
      int __val[2];
   }
   fsid_t;
typedef unsigned int useconds_t;
typedef unsigned long blksize_t;
typedef long fd_mask;
typedef int timer_t;
typedef int clockid_t;
typedef unsigned int id_t;

typedef unsigned int64_t;
typedef long long loff_t;
typedef unsigned long blkcnt_t;
typedef unsigned long fsblkcnt_t;
typedef unsigned long fsfilcnt_t;
typedef unsigned long long blkcnt64_t;
typedef unsigned long long fsblkcnt64_t;
typedef unsigned long long fsfilcnt64_t;
typedef unsigned char u_char;
typedef unsigned short u_short;
typedef unsigned long u_long;
typedef unsigned long ino_t;
typedef unsigned int gid_t;
typedef unsigned long dev_t;
typedef unsigned int mode_t;
typedef unsigned long nlink_t;
```
typedef char *caddr_t;

typedef struct {
  unsigned long fds_bits[___FDSET_LONGS];
} fd_set;

typedef long clock_t;
typedef long time_t;

1.3.54. sys/un.h

#define UNIX_PATH_MAX   108

struct sockaddr_un {
  sa_family_t sun_family;
  char sun_path[UNIX_PATH_MAX];
};

1.3.55. sys/utsname.h

#define SYS_NMLN        65

struct utsname {
  char sysname[65];
  char nodename[65];
  char release[65];
  char version[65];
  char machine[65];
  char domainname[65];
};

1.3.56. sys/wait.h

#define WIFSIGNALED(status)     (!WIFSTOPPED(status) & & !WIFEXITED(status))
#define WIFSTOPPED(status)      (((status) & 0xff) == 0x7f)
#define WEXITSTATUS(status)     (((status) & 0xff00) >> 8)
#define WTERMSIG(status)        ((status) & 0x7f)
#define WCOREDUMP(status)       ((status) & 0x80)
#define WIFEXITED(status)       (WTERMSIG(status) == 0)
#define WNOHANG 0x00000001
#define WUNTRACED 0x00000002
#define WCOREFLAG 0x80
#define WSTOPSIG(status)        WEXITSTATUS(status)
typedef enum
{
    P_ALL, P_PID, P_PGID
}
idtype_t;

1.3.57. syslog.h

#define LOG_EMERG       0
#define LOG_PRIMASK     0x07
#define LOG_ALERT       1
#define LOG_CRIT        2
#define LOG_ERR 3
#define LOG_WARNING     4
#define LOG_NOTICE      5
#define LOG_INFO        6
#define LOG_DEBUG       7

#define LOG_KERN        (0<<3)
#define LOG_AUTHPRIV    (10<<3)
#define LOG_FTP (11<<3)
#define LOG_USER        (1<<3)
#define LOG_MAIL        (2<<3)
#define LOG_DAEMON      (3<<3)
#define LOG_AUTH        (4<<3)
#define LOG_SYSLOG      (5<<3)
#define LOG_LPR (6<<3)
#define LOG_NEWS        (7<<3)
#define LOG_UUCP        (8<<3)
#define LOG_CRON        (9<<3)
#define LOG_FACMASK     0x03f8

#define LOG_LOCAL0      (16<<3)
#define LOG_LOCAL1      (17<<3)
#define LOG_LOCAL2      (18<<3)
#define LOG_LOCAL3      (19<<3)
#define LOG_LOCAL4      (20<<3)
#define LOG_LOCAL5      (21<<3)
#define LOG_LOCAL6      (22<<3)
#define LOG_LOCAL7      (23<<3)

#define LOG_UPTO(pri)   ((1 << ((pri)+1)) - 1)
#define LOG_MASK(pri)   (1 << (pri))
#define LOG_PID 0x01
#define LOG_CONS        0x02
#define LOG_ODELAY      0x04
#define LOG_NDELAY      0x08
#define LOG_NOWAIT      0x10
#define LOG_PERROR      0x20
1.3.58. termios.h

```c
#define TCIFLUSH        0
#define TCOFF  0
#define TCSANOW 0
#define BS0     0000000
#define CR0     0000000
#define FF0     0000000
#define NL0     0000000
#define TAB0    0000000
#define VT0     0000000
#define OPOST   0000001
#define OCRNL   0000010
#define ONOCR   0000020
#define ONLRET  0000040
#define OFILL   0000100
#define OFDEL   0000200
#define NL1     0000400
#define TCOFLUSH        1
#define TCOON   1
#define TCSADRAIN       1
#define TCIOFF  2
#define TCIOFUSH       2
#define TCSIOM   3
typedef unsigned int speed_t;
typedef unsigned char cc_t;
typedef unsigned int tcflag_t;
#define NCCS    32

struct termios
{
    tcflag_t c_iflag;
    tcflag_t c_oflag;
    tcflag_t c_cflag;
    tcflag_t c_lflag;
    cc_t c_line;
    cc_t c_cc[NCCS];
    speed_t c_ispeed;
    speed_t c_ospeed;
};
#define VINTR   0
#define VQUIT   1
#define VLNEXT  15
#define VERASE  2
#define VKILL   3
#define VEOF    4
#define IGNBRK 0000001
```
Chapter 1. Libraries

1.3.59. time.h

```c
# define CLK_TCK ((clock_t)__sysconf(2))
# define CLOCK_REALTIME 0
# define TIMER_ABSTIME 1
# define CLOCKS_PER_SEC 1000000

struct tm {
  int tm_sec;
  int tm_min;
  int tm_hour;
  int tm_mday;
  int tm_mon;
  int tm_year;
  int tm_wday;
  int tm_yday;
  int tm_isdst;
  long tm_gmtoff;
};
```
Chapter 1. Libraries

1.3.60. ulimit.h

#define UL_GETFSIZE 1
#define UL_SETFSIZE 2

1.3.61. unistd.h

#define SEEK_SET 0
#define STDIN_FILENO 0
#define SEEK_CUR 1
#define STDOUT_FILENO 1
#define SEEK_END 2
#define STDERR_FILENO 2

typedef long long off64_t;
#define F_OK 0
#define X_OK 1
#define W_OK 2
#define R_OK 4
#define _POSIX_VDISABLE '\0'
#define _POSIX_CHOWN_RESTRICTED 1
#define _POSIX_JOB_CONTROL 1
#define _POSIX_NO_TRUNC 1
#define _POSIX_SHELL 1
#define _POSIX_FSYNC 200112
#define _POSIX_MAPPED_FILES 200112
#define _POSIX_MEMLOCK 200112
#define _POSIX_MEMLOCK_RANGE 200112
#define _POSIX_MEMORY_PROTECTION 200112
#define _POSIX_SEMAPHORES 200112
#define _POSIX_SHARED_MEMORY_OBJECTS 200112
#define _POSIX_TIMERS 200112
#define _POSIX2_C_BIND 200112L
#define _POSIX2_VERSION 200112L
#define _POSIX_THREADS 200112L
#define _POSIX_VERSION 200112L
#define _PC_LINK_MAX 0
#define _PC_MAX_CANON 1
# Chapter 1. Libraries

#define _PC_ASYNC_IO 10
#define _PC_PRIO_IO 11
#define _PC_FILESIZEBITS 13
#define _PC_REC_INCR_XFER_SIZE 14
#define _PC_REC_MIN_XFER_SIZE 16
#define _PC_REC_XFER_ALIGN 17
#define _PC_ALLOC_SIZE_MIN 18
#define _PC_MAX_INPUT 2
#define _PC_2_SYMLINKS 20
#define _PC_NAME_MAX 3
#define _PC_PATH_MAX 4
#define _PC_PIPE_BUF 5
#define _PC_CHOWN_RESTRICTED 6
#define _PC_NO_TRUNC 7
#define _PC_VDISABLE 8
#define _PC_SYNC_IO 9

#define _SC_ARG_MAX 0
#define _SC_CHILD_MAX 1
#define _SC_PRIORITY_SCHEDULING 10
#define _SC_TIMERS 11
#define _SC_ASYNCIO 12
#define _SC_XBS5_ILP32_OFF32 125
#define _SC_XBS5_ILP32_OFFBIG 126
#define _SC_XBS5_LP64_OFF64 127
#define _SC_XBS5_LPBIG_OFFBIG 128
#define _SC_XOPEN_LEGACY 129
#define _SC_PRIORITIZED_IO 13
#define _SC_XOPEN_REALTIME 130
#define _SC_XOPEN_REALTIME_THREADS 131
#define _SC_ADVISORY_INFO 132
#define _SC_BARRIERS 133
#define _SC_CLOCK_SELECTION 137
#define _SC_CPUTIME 138
#define _SC_THREAD_CPUTIME 139
#define _SC_SYNCHRONIZED_IO 14
#define _SC_MONOTONIC_CLOCK 149
#define _SC_FSYNC 15
#define _SC_READER_WRITER_LOCKS 153
#define _SC_SPIN_LOCKS 154
#define _SC_SPIN_LOCKS 155
#define _SC_SHELL 157
#define _SC_SPAWN 159
#define _SC_MAPPED_FILES 16
#define _SC_SPORADIC_SERVER 160
#define _SC_THREAD_SPORADIC_SERVER 161
#define _SC_TIMEOUTS 164
#define _SC_TYPED_MEMORY_OBJECTS 165
#define _SC_2_PBS_ACCOUNTING 169
#define _SC_MEMLOCK 17
#define _SC_2_PBS_LOCATE 170
#define _SC_2_PBS_MESSAGE 171
#define _SC_2_PBS_TRACK 172
#define _SC_SYMLOOP_MAX 173
#define _SC_2_PBS_CHECKPOINT 175
#define _SC_V6_ILP32_OFF32 176
#define _SC_V6_ILP32_OFFBIG 177
#define _SC_V6_LP64_OFF64 178
#define _SC_V6_LPBIG_OFFBIG 179
#define _SC_MEMLOCK_RANGE 18
#define _SC_HOST_NAME_MAX 180
#define _SC_TRACE 181
#define _SC_TRACE_EVENT_FILTER 182
#define _SC_TRACE_INHERIT 183
#define _SC_TRACE_LOG 184
#define _SC_MEMORY_PROTECTION 19
#define _SC_CLK_TCK 2
#define _SC_MESSAGE_PASSING 20
#define _SC_SEMAPHORES 21
#define _SC_SHARED_MEMORY_OBJECTS 22
#define _SC_AIO_LISTIO_MAX 23
#define _SC_AIO_MAX 24
#define _SC_AIO_PRIO_DELTA_MAX 25
#define _SC_DELAYTIMER_MAX 26
#define _SC_MQ_OPEN_MAX 27
#define _SC_MQ_PRIO_MAX 28
#define _SC_VERSION 29
#define _SC_NGROUPS_MAX 3
#define _SC_PAGESIZE 30
#define _SC_RTSIG_MAX 31
#define _SC_SEM_NSEMS_MAX 32
#define _SC_SEM_VALUE_MAX 33
#define _SC_SIGQUEUE_MAX 34
#define _SC TIMER_MAX 35
#define _SC_BC_BASE_MAX 36
#define _SC_BC_DIM_MAX 37
#define _SC_BC_SCALE_MAX 38
#define _SC_BC_STRING_MAX 39
#define _SC_OPEN_MAX 4
#define _SC_COLL_WEIGHTS_MAX 40
#define _SC_EXPR_NEST_MAX 42
#define _SC_LINE_MAX 43
#define _SC_RE_DUP_MAX 44
#define _SC_2_VERSION 46
#define _SC_2_C_BIND 47
#define _SC_2_C_DEV 48
#define _SC_STREAM_MAX 49
#define _SC_2_FORT_RUN 50
#define _SC_2_SW_DEV 51
#define _SC_2_LOCALEDEF 52
#define _SC_TZNAME_MAX 6
#define _SC_IOV_MAX 60
#define _SC_THREADS 67
#define _SC_THREAD_SAFE_FUNCTIONS 68
#define _SC_GETGR_R_SIZE_MAX 69
#define _SC_JOB_CONTROL 7
#define _SC_GETPW_R_SIZE_MAX 70
#define _SC_LOGIN_NAME_MAX 71
#define _SC_TTY_NAME_MAX 72
#define _SC_THREAD_DESTRUCTOR_ITERATIONS 73
#define _SC_THREAD_KEYS_MAX 74
#define _SC_THREAD_STACK_MIN 75
#define _SC_THREAD_THREADS_MAX 76
#define _SC_THREAD_ATTR_STACKADDR 77
#define _SC_THREAD_ATTR_STACKSIZE 78
#define _SC_THREAD_PRIORITY_SCHEDULING 79
#define _SC_SAVED_IDS 8
#define _SC_THREAD_PRIO_INHERIT 80
#define _SC_THREAD_PRIO_PROTECT 81
#define _SC_THREAD_PROCESS_SHARED 82
#define _SC_ATEXIT_MAX 87
#define _SC_PASS_MAX 88
#define _SC_XOPEN_VERSION 89
#define _SC_REALTIME_SIGNALS 9
#define _SC_XOPEN_UNIX 91
#define _SC_XOPEN_CRYPT 92
#define _SC_XOPEN_ENH_I18N 93
#define _SC_XOPEN_SHM 94
#define _SC_2_CHAR_TERM 95
#define _SC_2_C_VERSION 96
#define _SC_2_UPE 97
#define _CS_PATH 0
#define _POSIX_REGEXP 1
#define _CS_XBS5_ILP32_OFF32_CFLAGS 100
#define _CS_XBS5_ILP32_OFF32_LDFLAGS 101
#define _CS_XBS5_ILP32_OFF32_LIBS 102
#define _CS_XBS5_ILP32_OFF32_LINTFLAGS 103
#define _CS_XBS5_ILP32_OFFBIG_CFLAGS 104
#define _CS_XBS5_ILP32_OFFBIG_LDFLAGS 105
#define _CS_XBS5_ILP32_OFFBIG_LIBS 106
#define _CS_XBS5_ILP32_OFFBIG_LINTFLAGS 107
#define _CS_XBS5_LP64_OFF64_CFLAGS 108
#define _CS_XBS5_LP64_OFF64_LDFLAGS 109
#define _CS_XBS5_LP64_OFF64_LIBS 110
#define _CS_XBS5_LP64_OFF64_LINTFLAGS 111
#define _CS_XBS5_LPBIG_OFFBIG_CFLAGS 112
#define _CS_XBS5_LPBIG_OFFBIG_LDFLAGS 113
#define _CS_XBS5_LPBIG_OFFBIG_LIBS 114
#define _CS_XBS5_LPBIG_OFFBIG_LINTFLAGS 115
#define F_ULOCK 0
1.3.62. utime.h

```c
struct utimbuf
{
    time_t actime;
    time_t modtime;
};
```

1.3.63. utmp.h

```c
#define UT_HOSTSIZE     256
#define UT_LINESIZE     32
#define UT_NAMESIZE     32

struct exit_status
{
    short e_termination;
    short e_exit;
};
```

1.3.64. wchar.h

```c
#define WEOF    (0xffffffffu)
#define WCHAR_MAX       0x7FFFFFFF
#define WCHAR_MIN       0x80000000
```

1.3.65. wctype.h

```c
typedef unsigned long wctype_t;
typedef unsigned int wint_t;
```
typedef const int32_t *wctrans_t;
typedef struct {
    int count;
    wint_t value;
} __mbstate_t;
typedef __mbstate_t mbstate_t;
enum {
    WRDE_DOFFS, WRDE_APPEND, WRDE_NOCMD, WRDE_REUSE, WRDE_SHOWERR, WRDE_UNDEF,
    __WRDE_FLAGS
};
typedef struct {
    int we_wordc;
    char **we_wordv;
    int we_offs;
} wordexp_t;
enum {
    WRDE_NOSYS, WRDE_NOSPACE, WRDE_BADCHAR, WRDE_BADVAL, WRDE_CMDSUB,
    WRDE_SYNTAX
};

1.3.66. wordexp.h

1.4. Interface Definitions for libc

The following interfaces are included in libc and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed above for libc shall behave as described in the referenced base document.
_IO_feof

Name

_IO_feof — alias for feof

Synopsis

int _IO_feof(_IO_FILE *__fp);

Description

_IO_feof tests the end-of-file indicator for the stream pointed to by __fp, returning a non-zero value if it is set.

_IO_feof is not in the source standard; it is only in the binary standard.

_IO_getc

Name

_IO_getc — alias for getc

Synopsis

int _IO_getc(_IO_FILE *__fp);

Description

_IO_getc reads the next character from __fp and returns it as an unsigned char cast to an int, or EOF on end-of-file or error.

_IO_getc is not in the source standard; it is only in the binary standard.

_IO_putc

Name

_IO_putc — alias for putc

Synopsis

int _IO_putc(int __c, _IO_FILE *__fp);

Description

_IO_putc writes the character __c, cast to an unsigned char, to __fp.

_IO_putc is not in the source standard; it is only in the binary standard.
_IO_puts

Name

_IO_puts — alias for puts

Synopsis

```c
int _IO_puts(const char *__s);
```

Description

_IO_puts writes the string __s and a trailing newline to stdout.
_IOC_puts is not in the source standard; it is only in the binary standard.

__assert_fail

Name

__assert_fail — abort the program after false assertion

Synopsis

```c
void __assert_fail(const char *assertion, const char *file, unsigned int line, const char *function);
```

Description

The __assert_fail function is used to implement the assert interface of ISO POSIX (2003). The __assert_fail function shall print the given file filename, line line number, function function name and a message on the standard error stream in an unspecified format, and abort program execution via the abort function.

For example:

```
a.c:10: foobar: Assertion a == b failed.
```

If function is NULL, __assert_fail shall omit information about the function.

assertion, file, and line shall be non-NULL.

The __assert_fail function is not in the source standard; it is only in the binary standard. The assert interface is not in the binary standard; it is only in the source standard. The assert may be implemented as a macro.
__ctype_b_loc

Name
__ctype_b_loc — accessor function for __ctype_b array for ctype functions

Synopsis
#include <ctype.h>
define unsigned short int **ctype_b_loc (void);

Description
The __ctype_b_loc function shall return a pointer into an array of characters in the current locale that contains
caracteristics for each character in the current character set. The array shall contain a total of 384 characters, and can
be indexed with any signed or unsigned char (i.e. with an index value between −128 and 255). If the application is
multithreaded, the array shall be local to the current thread.

This interface is not in the source standard; it is only in the binary standard.

Return Value
The __ctype_b_loc function shall return a pointer to the array of characters to be used for the ctype family of
functions (see <ctype.h>).

__ctype_get_mb_cur_max

Name
__ctype_get_mb_cur_max — maximum length of a multibyte character in the current locale

Synopsis
size_t __ctype_get_mb_cur_max(void);

Description
__ctype_get_mb_cur_max returns the maximum length of a multibyte character in the current locale.
__ctype_get_mb_cur_max is not in the source standard; it is only in the binary standard.
__ctype_tolower_loc

Name
__ctype_tolower_loc — accessor function for __ctype_b_tolower array for ctype tolower() function

Synopsis
#include <ctype.h>
int32_t **__ctype_tolower_loc(void);

Description
The __ctype_tolower_loc function shall return a pointer into an array characters in the current locale that contains lower case equivalents for each character in the current character set. The array shall contain a total of 384 characters, and can be indexed with any signed or unsigned char (i.e. with an index value between -128 and 255). If the application is multithreaded, the array shall be local to the current thread.

This interface is not in the source standard; it is only in the binary standard.

__ctype_toupper_loc

Name
__ctype_toupper_loc — accessor function for __ctype_b_toupper array for ctype toupper() function

Synopsis
#include <ctype.h>
int32_t **__ctype_toupper_loc(void);

Description
The __ctype_toupper_loc function shall return a pointer into an array characters in the current locale that contains upper case equivalents for each character in the current character set. The array shall contain a total of 384 characters, and can be indexed with any signed or unsigned char (i.e. with an index value between -128 and 255). If the application is multithreaded, the array shall be local to the current thread.

This interface is not in the source standard; it is only in the binary standard.
__cxa_atexit

Name
__cxa_atexit — register a function to be called by exit or when a shared library is unloaded

Synopsis
int __cxa_atexit(void (*func) (void *), void *arg, void *dso_handle);

Description
__cxa_atexit registers a function to be called by exit or when a shared library is unloaded.
The __cxa_atexit function is used to implement atexit, as described in ISO POSIX (2003). Calling __cxa_atexit(func)
from the statically linked part of an application shall be equivalent to
__cxa_atexit(func, NULL, NULL)

__cxa_atexit is not in the source standard; it is only in the binary standard. atexit is not in the binary standard; it is
only in the source standard.

__daylight

Name
__daylight — Daylight savings time flag

Synopsis
int __daylight;

Description
The integer variable __daylight shall implement the daylight savings time flag daylight as specified in the ISO POSIX (2003) header file <time.h>.

__daylight is not in the source standard; it is only in the binary standard. daylight is not in the binary standard; it is
only in the source standard.
__environ

Name
__environ — alias for environ - user environment

Synopsis
extern char **__environ;

Description
__environ is an alias for environ - user environment.
__environ has the same specification as environ.
__environ is not in the source standard; it is only in the binary standard.

__errno_location

Name
__errno_location — address of errno variable

Synopsis
int *__errno_location(void);

Description
__errno_location is not in the source standard; it is only in the binary standard.

__fpending

Name
__fpending — returns in bytes the amount of output pending on a stream

Synopsis
#include <stdio_ext.h>
size_t __fpending(FILE *stream);

Description
__fpending returns the amount of output in bytes pending on a stream.
__fpending is not in the source standard; it is only in the binary standard.
__getpagesize

Name

__getpagesize — alias for getpagesize - get current page size

Synopsis

int __getpagesize(void);

Description

__getpagesize is an alias for getpagesize - get current page size.
__getpagesize has the same specification as getpagesize.
__getpagesize is not in the source standard; it is only in the binary standard.

__getpgid

Name

__getpgid — get the process group id

Synopsis

pid_t __getpgid(pid_t pid);

Description

__getpgid has the same specification as getpgid.
__getpgid is not in the source standard; it is only in the binary standard.
__h_errno_location

Name
2657  __h_errno_location — address of h_errno variable

Synopsis
2658  int *__h_errno_location(void);

Description
2659  __h_errno_location returns the address of the h_errno variable, where h_errno is as specified in the Single Unix Specification.
2660  __h_errno_location is not in the source standard; it is only in the binary standard. Note that h_errno itself is only in the source standard; it is not in the binary standard.

__isinf

Name
2663  __isinf — test for infinity

Synopsis
2664  int __isinf(double arg);

Description
2665  __isinf has the same specification as isinf in the Single UNIX Specification, Version 3, except that the argument type for __isinf is known to be double.
2666  __isinf is not in the source standard; it is only in the binary standard.
__isinff

Name
__isinff — test for infinity

Synopsis
int __isinff(float arg);

Description
__isinff has the same specification as isinf in the Single UNIX Specification, Version 3, except that the argument type for __isinff is known to be float.
__isinff is not in the source standard; it is only in the binary standard.

__isinfl

Name
__isinfl — test for infinity

Synopsis
int __isinfl(long double arg);

Description
__isinfl has the same specification as isinf in the Single UNIX Specification, Version 3, except that the argument type for __isinfl is known to be long double.
__isinfl is not in the source standard; it is only in the binary standard.
__isnan

Name
2678  __isnan — test for infinity

Synopsis
2679  int __isnan(double arg);

Description
2680  __isnan has the same specification as isnan in the Single UNIX Specification, Version 3, except that the argument
type for __isnan is known to be double.
2682  __isnan is not in the source standard; it is only in the binary standard.

__isnanf

Name
2683  __isnanf — test for infinity

Synopsis
2684  int __isnanf(float arg);

Description
2685  __isnanf has the same specification as isnan in the Single UNIX Specification, Version 3, except that the argument
type for __isnanf is known to be float.
2687  __isnanf is not in the source standard; it is only in the binary standard.
__isnanl

Name
__isnanl — test for infinity

Synopsis
int __isnanl(long double arg);

Description
__isnanl has the same specification as isnan in the Single UNIX Specification, Version 3, except that the argument type for __isnanl is known to be long double.
__isnanl is not in the source standard; it is only in the binary standard.

__libc_current_sigrtmax

Name
__libc_current_sigrtmax — return number of available real-time signal with lowest priority

Synopsis
int __libc_current_sigrtmax(void);

Description
__libc_current_sigrtmax returns the number of an available real-time signal with the lowest priority.
__libc_current_sigrtmax is not in the source standard; it is only in the binary standard.

__libc_current_sigrtmin

Name
__libc_current_sigrtmin — return number of available real-time signal with highest priority

Synopsis
int __libc_current_sigrtmin(void);

Description
__libc_current_sigrtmin returns the number of an available real-time signal with the highest priority.
__libc_current_sigrtmin is not in the source standard; it is only in the binary standard.
__libc_start_main

Name

__libc_start_main — initialization routine

Synopsis

```c
#include <stdatomic.h>

int __libc_start_main(int (*main) (int, char**, char**), int argc, char *__unbounded *__unbounded ubp_av, void (*init) (void), void (*fini) (void), void (*rtld_fini) (void), void (*__unbounded stack_end));
```

Description

The __libc_start_main function shall initialize the process, call the main function with appropriate arguments, and handle the return from main.

__libc_start_main is not in the source standard; it is only in the binary standard.

__lxstat

Name

__lxstat — inline wrapper around call to lxstat

Synopsis

```c
#include <ctype.h>

int __lxstat(int version, char *__path, (struct stat *__statbuf));
```

Description

__lxstat is an inline wrapper around call to lxstat.

__lxstat is not in the source standard; it is only in the binary standard.
__mempcpy

Name

__mempcpy — copy given number of bytes of source to destination

Synopsis

```c
#include <string.h>

ptr_t __mempcpy(ptr_t restrict dest, const ptr_t restrict src, size_t n);
```

Description

__mempcpy copies \( n \) bytes of source to destination, returning pointer to bytes after the last written byte.

__mempcpy is not in the source standard; it is only in the binary standard.

__rawmemchr

Name

__rawmemchr — scan memory

Synopsis

```c
#include <string.h>

ptr_t __rawmemchr(const ptr_t s, int c);
```

Description

__rawmemchr searches in \( s \) for \( c \).

__rawmemchr is a weak alias to rawmemchr. It is similar to memchr, but it has no length limit.

__rawmemchr is not in the source standard; it is only in the binary standard.
\texttt{\_\_register\_atfork}

\textbf{Name}
\texttt{\_\_register\_atfork} — alias for \texttt{register\_atfork}

\textbf{Synopsis}
\begin{verbatim}
 int \_\_register\_atfork(void (*\texttt{prepare})(), void (*\texttt{parent})(), void (*\texttt{child})(), void *\texttt{\_\_dso\_handle});
\end{verbatim}

\textbf{Description}
\texttt{\_\_register\_atfork} implements \texttt{pthread\_atfork} as specified in ISO POSIX (2003). The additional parameter \texttt{\_\_dso\_handle} allows a shared object to pass in it's handle so that functions registered by \texttt{\_\_register\_atfork} can be unregistered by the runtime when the shared object is unloaded.

\texttt{\_\_sigsetjmp}

\textbf{Name}
\texttt{\_\_sigsetjmp} — save stack context for non-local goto

\textbf{Synopsis}
\begin{verbatim}
 int \_\_sigsetjmp(jmp\_buf \texttt{env}, int \texttt{savemask});
\end{verbatim}

\textbf{Description}
\texttt{\_\_sigsetjmp} has the same behavior as \texttt{sigsetjmp} as specified by ISO POSIX (2003). \texttt{\_\_sigsetjmp} is not in the source standard; it is only in the binary standard.
__stpcpy

Name

__stpcpy — copy a string returning a pointer to its end

Synopsis

```c
#include <string.h>
char *__stpcpy(char *dest, const char *src);
```

Description

__stpcpy copies the string src (including the terminating /0 character) to the array dest. The strings may not overlap, and dest must be large enough to receive the copy.

Return Value

__stpcpy returns a pointer to the end of the string dest (that is, the address of the terminating NULL character) rather than the beginning.

__stpcpy has the same specification as stpcpy.

__stpcpy is not in the source standard; it is only in the binary standard.

__strdup

Name

__strdup — alias for strdup

Synopsis

```c
char *__strdup(const char string);
```

Description

__strdup has the same specification as strdup.

__strdup is not in the source standard; it is only in the binary standard.
__strtod_internal

Name
__strtod_internal — underlying function for strtod

Synopsis
double __strtod_internal(const char *__nptr, char **__endptr, int __group);

Description
__group shall be 0 or the behavior of __strtod_internal is undefined.
__strtod_internal(__nptr, __endptr, 0) has the same specification as strtod(__nptr, __endptr).
__strtod_internal is not in the source standard; it is only in the binary standard.

__strtof_internal

Name
__strtof_internal — underlying function for strtof

Synopsis
float __strtof_internal(const char *__nptr, char **__endptr, int __group);

Description
__group shall be 0 or the behavior of __strtof_internal is undefined.
__strtof_internal(__nptr, __endptr, 0) has the same specification as strtof(__nptr, __endptr).
__strtof_internal is not in the source standard; it is only in the binary standard.
__strtok_r

Name
__strtok_r — alias for strtok_r

Synopsis
char *__strtok_r(char *__restrict s, __const char *__restrict delim, char **__restrict save_ptr);

Description
__strtok_r has the same specification as strtok_r.
__strtok_r is not in the source standard; it is only in the binary standard.

__strtol_internal

Name
__strtol_internal — alias for strtol

Synopsis
long int __strtol_internal(const char *__nptr, char **__endptr, int __base, int __group);

Description
__group shall be 0 or the behavior of __strtol_internal is undefined.
__strtol_internal(__nptr, __endptr, __base, 0) has the same specification as strtol(__nptr, __endptr, __base).
__strtol_internal is not in the source standard; it is only in the binary standard.
__strtold_internal

Name
__strtold_internal — underlying function for strtold

Synopsis
long double __strtold_internal(const char *__nptr, char **__endptr, int __group);

Description
__group shall be 0 or the behavior of __strtold_internal is undefined.
__strtold_internal (__nptr, __endptr, 0) has the same specification as strtold (__nptr, __endptr).
__strtold_internal is not in the source standard; it is only in the binary standard.

__strtoll_internal

Name
__strtoll_internal — underlying function for strtoll

Synopsis
long long __strtoll_internal(const char *__nptr, char **__endptr, int __base, int __group);

Description
__group shall be 0 or the behavior of __strtoll_internal is undefined.
__strtoll_internal (__nptr, __endptr, __base, 0) has the same specification as strtoll (__nptr, __endptr, __base).
__strtoll_internal is not in the source standard; it is only in the binary standard.
__strtoul_internal

Name

__strtoul_internal — underlying function for strtoul

Synopsis

unsigned long int __strtoul_internal(const char *__nptr, char **__endptr, int __base, int __group);

Description

__group shall be 0 or the behavior of __strtoul_internal is undefined.

__strtoul_internal(__nptr, __endptr, __base, 0) has the same specification as strtoul(__nptr, __endptr, __base).

__strtoul_internal is not in the source standard; it is only in the binary standard.

__strtoull_internal

Name

__strtoull_internal — underlying function for strtoull

Synopsis

unsigned long long __strtoull_internal(const char *__nptr, char **__endptr, int __base, int __group);

Description

__group shall be 0 or the behavior of __strtoull_internal is undefined.

__strtoull_internal(__nptr, __endptr, __base, 0) has the same specification as strtoull(__nptr, __endptr, __base).

__strtoull_internal is not in the source standard; it is only in the binary standard.
__sysconf

Name
__sysconf — get configuration information at runtime

Synopsis
#include <unistd.h>
long __sysconf(int name);

Description
__sysconf gets configuration information at runtime.
__sysconf is weak alias to sysconf.
__sysconf has the same specification as sysconf.
__sysconf is not in the source standard; it is only in the binary standard.

__sysv_signal

Name
__sysv_signal — signal handling

Synopsis
__sighandler_t __sysv_signal(int sig, __sighandler_t handler);

Description
__sysv_signal has the same behavior as signal as specified by ISO POSIX (2003).
__sysv_signal is not in the source standard; it is only in the binary standard.
__timezone

Name
— global variable containing timezone

Synopsis
long int __ timezone;

Description
__timezone has the same specification as timezone in the ISO POSIX (2003).

__tzname

Name
— global variable containing the timezone

Synopsis
char *__tzname[2];

Description
__tzname has the same specification as tzname in the ISO POSIX (2003).
Note that the array size of 2 is explicit in the ISO POSIX (2003), but not in the SUSv2.

__wcstod_internal

Name
__wcstod_internal — underlying function for wcstod

Synopsis
double __wcstod_internal(const wchar_t *nptr, wchar_t **endptr, int group);

Description
group shall be 0 or the behavior of __wcstod_internal is undefined.
__wcstod_internal (nptr, endptr, 0) has the same specification as wcstod (nptr, endptr).
__wcstod_internal is not in the source standard; it is only in the binary standard.
__wcstof_internal

Name

__wcstof_internal — underlying function for wcstof

Synopsis

float __wcstof_internal(const wchar_t *nptr, wchar_t **endptr, int group);

Description

group shall be 0 or the behavior of __wcstof_internal is undefined.
__wcstof_internal(nptr, endptr, 0) has the same specification as wcstof(nptr, endptr).
__wcstof_internal is not in the source standard; it is only in the binary standard.

__wcstol_internal

Name

__wcstol_internal — underlying function for wcstol

Synopsis

long __wcstol_internal(const wchar_t *nptr, wchar_t **endptr, int base, int group);

Description

group shall be 0 or the behavior of __wcstol_internal is undefined.
__wcstol_internal(nptr, endptr, base, 0) has the same specification as wcstol(nptr, endptr, base).
__wcstol_internal is not in the source standard; it is only in the binary standard.
__wcstold_internal

Name
2826  __wcstold_internal — underlying function for wcstold

Synopsis
2827  long double __wcstold_internal(const wchar_t *nptr, wchar_t **endptr, int group);

Description
2828  group shall be 0 or the behavior of __wcstold_internal is undefined.
2829  __wcstold_internal(nptr, endptr, 0) has the same specification as wcstold(nptr, endptr).
2830  __wcstold_internal is not in the source standard; it is only in the binary standard.

__wcstoul_internal

Name
2831  __wcstoul_internal — underlying function for wcstoul

Synopsis
2832  unsigned long __wcstoul_internal(const wchar_t *restrict nptr, wchar_t **restrict endptr,
2833  int base, int group);

Description
2834  group shall be 0 or the behavior of __wcstoul_internal is undefined.
2835  __wcstoul_internal(nptr, endptr, base, 0) has the same specification as wcstoul(nptr, endptr,
2836  base).
2837  __wcstoul_internal is not in the source standard; it is only in the binary standard.
__xmknod

**Name**

__xmknod — make block or character special file

**Synopsis**

```c
int __xmknod(int ver, const char *path, mode_t mode, dev_t *dev);
```

**Description**

The __xmknod shall implement the mknod interface from ISO POSIX (2003).

__xmknod(1, path, mode, dev) has the same specification as mknod(path, mode, dev).

ver shall be 1 or the behavior of __xmknod is undefined.

The __xmknod function is not in the source standard; it is only in the binary standard. The mknod function is not in the binary standard; it is only in the source standard.

__xstat

**Name**

__xstat — Get File Status

**Synopsis**

```c
#include <sys/stat.h>
#include <unistd.h>
int __xstat(int ver, const char *path, (struct stat *stat_buf));
int __lxstat(int ver, const char *path, (struct stat *stat_buf));
int __fxstat(int ver, int fildes, (struct stat *stat_buf));
```

**Description**

The functions __xstat, __lxstat, and __fxstat shall implement the ISO POSIX (2003) functions stat, lstat, and fstat respectively.

ver shall be 3 or the behavior of these functions is undefined.

__xstat(3, path, stat_buf) shall behave as stat(path, stat_buf) as specified by ISO POSIX (2003).

__lxstat(3, path, stat_buf) shall behave as lstat(path, stat_buf) as specified by ISO POSIX (2003).

__fxstat(3, fildes, stat_buf) shall behave as fstat(fildes, stat_buf) as specified by ISO POSIX (2003).

__xstat, __lxstat, and __fxstat are not in the source standard; they are only in the binary standard.

stat, lstat, and fstat are not in the binary standard; they are only in the source standard.
__xstat64

Name
__xstat64 — Get File Status

Synopsis

#define _LARGEFILE_SOURCE 1
#include <sys/stat.h>
#include <unistd.h>
int __xstat64(int ver, const char *path, (struct stat64 *stat_buf));
int __lxstat64(int ver, const char *path, (struct stat64 *stat_buf));
int __fxstat64(int ver, int fildes, (struct stat64 *stat_buf));

Description
The functions __xstat64, __lxstat64, and __fxstat64 shall implement the Large File Support functions
stat64, lstat64, and fstat64 respectively.
ver shall be 3 or the behavior of these functions is undefined.
__xstat64(3, path, stat_buf) shall behave as stat(path, stat_buf) as specified by Large File Support.
__lxstat64(3, path, stat_buf) shall behave as lstat(path, stat_buf) as specified by Large File Support.
__fxstat64(3, fildes, stat_buf) shall behave as fstat(fildes, stat_buf) as specified by Large File Support.
__xstat64, __lxstat64, and __fxstat64 are not in the source standard; they are only in the binary standard.
stat64, lstat64, and fstat64 are not in the binary standard; they are only in the source standard.

_environ

Name
__environ — alias for environ - user environment

Synopsis
extern char **_environ;

Description
__environ is an alias for environ - user environment.
_nl_msg_cat_cntr

Name
2879 _nl_msg_cat_cntr — new catalog load counter

Synopsis
2880 #include <libintl.h>
2881 extern int _nl_msg_cat_cntr;

Description
2883 _nl_msg_cat_cntr is incremented each time a new catalog is loaded. It is a variable defined in loadmsgcat.c and is used by Message catalogs for internationalization.

_obstack_begin

Name
2885 _obstack_begin — initialize an obstack for use

Synopsis
2886 #include <obstack.h>
2887 int _obstack_begin(struct obstack *, int, int, void (*)(long), void (*)(void *));

Description
2888 _obstack_begin initializes an obstack for use.

Future Directions
2889 Future versions of this specification may not include support for this interface.
_obstack_newchunk

Name
_obstack_newchunk — allocate a new current chunk of memory for the obstack

Synopsis

#include <obstack.h>

void _obstack_newchunk(struct obstack *, int);

Description

_obstack_newchunk allocates a new current chunk of memory for the obstack.

Future Directions

Future versions of this specification may not include support for this interface.

/sys_errlist

Name
_sys_errlist — array containing the "C" locale strings used by strerror()

Synopsis

#include <stdio.h>

extern const char *const _sys_errlist[];

Description

_sys_errlist is an array containing the "C" locale strings used by strerror. This normally should not be used
directly. strerror provides all of the needed functionality.

/sys_siglist

Name
_sys_siglist — array containing the names of the signal names

Synopsis

#include <signal.h>
extern const char *const _sys_siglist[NSIG];

Description

_sys_siglist is an array containing the names of the signal names.

The _sys_siglist array is only in the binary standard; it is not in the source standard. Applications wishing to access the names of signals should use the _strsignal function.
**acct**

**Name**
acct — switch process accounting on or off

**Synopsis**
```
#include <dirent.h>
int acct(const char *filename);
```

**Description**
When *filename* is the name of an existing file, acct turns accounting on and appends a record to *filename* for each terminating process. When *filename* is NULL, acct turns accounting off.

**Return Value**
On success, 0 is returned. On error, -1 is returned and the global variable *errno* is set appropriately.

**Errors**
- **ENOSYS**
  BSD process accounting has not been enabled when the operating system kernel was compiled. The kernel configuration parameter controlling this feature is `CONFIG_BSD_PROCESS_ACCT`.
- **ENOMEM**
  Out of memory.
- **EPERM**
  The calling process has no permission to enable process accounting.
- **EACCES**
  *filename* is not a regular file.
- **EIO**
  Error writing to the *filename*.
- **EUSERS**
  There are no more free file structures or we run out of memory.
adjtime

Name
adjtime — correct the time to allow synchronization of the system clock

Synopsis
#include <time.h>
int adjtime((const struct timeval *delta), (struct timeval *olddelta));

Description
adjtime makes small adjustments to the system time as returned by gettimeofday(2), advancing or retarding it by
the time specified by the timeval delta. If delta is negative, the clock is slowed down by incrementing it more
slowly than normal until the correction is complete. If delta is positive, a larger increment than normal is used. The
skew used to perform the correction is generally a fraction of one percent. Thus, the time is always a monotonically
increasing function. A time correction from an earlier call to adjtime may not be finished when adjtime is called
again. If olddelta is non-NULL, the structure pointed to will contain, upon return, the number of microseconds still
to be corrected from the earlier call.
adjtime may be used by time servers that synchronize the clocks of computers in a local area network. Such time
servers would slow down the clocks of some machines and speed up the clocks of others to bring them to the average
network time.
The adjtime is restricted to the super-user.

Return Value
On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors
EFAULT
An argument points outside the process's allocated address space.
EPERM
The process's effective user ID is not that of the super-user.
adjtimex

Name
adjtimex — tune kernel clock (DEPRECATED)

Synopsis
#include <sys/timex.h>
int adjtimex((struct timex *buf));

Description
The adjtimex function is deprecated from the LSB and is expected to disappear from a future version of the LSB.
The LSB generally does not include interfaces unlikely to be used by software applications.

Linux uses David L. Mills’ clock adjustment algorithm (see RFC 1305). adjtimex reads and optionally sets
adjustment parameters for this algorithm. adjtimex takes a pointer to a timex structure, updates kernel parameters
from field values, and returns the same structure with current kernel values. This structure is declared as follows:

struct timex {
    int modes;          /* mode selector */
    long offset;         /* time offset (usec) */
    long freq;           /* frequency offset (scaled ppm) */
    long maxerror;       /* maximum error (usec) */
    long esterror;       /* estimated error (usec) */
    int status;         /* clock command/status */
    long constant;       /* pll time constant */
    long precision; /* clock precision (usec) (read only) */
    long tolerance; /* clock frequency tolerance (ppm)
        (read only) */
    struct timeval time; /* current time (read only) */
    long tick;           /* usecs between clock ticks */
};

modes determines which parameters, if any, to set. modes may contain a bitwise-or combination of zero or more of
the following bits:

#define ADJ_OFFSET 0x0001 /* time offset */
#define ADJ_FREQUENCY 0x0002 /* frequency offset */
#define ADJ_MAXERROR 0x0004 /* maximum error */
#define ADJ_ESTERROR 0x0008 /* estimated error */
#define ADJ_STATUS 0x0010 /* clock status */
#define ADJ_TIMECONST 0x0020 /* pll time constant */
#define ADJ_TICK 0x4000 /* tick value */
Chapter 1. Libraries

### #define ADJ_OFFSET_SINGLESLOT 0x8001 /* old-fashioned adjtime */

Ordinary users are restricted to a 0 value for modes. Only the superuser may set any parameters.

**Return Value**

On success, adjtimex returns the clock state:

```c
#define TIME_OK   0  /* clock synchronized */
#define TIME_INS  1  /* insert leap second */
#define TIME_DEL  2  /* delete leap second */
#define TIME_OOP  3  /* leap second in progress */
#define TIME_WAIT 4  /* leap second has occurred */
#define TIME_BAD  5  /* clock not synchronized */
```

On error, the global variable errno is set to -1.

**Errors**

- **EFAULT**
  - `buf` does not point to writable memory.

- **EPERM**
  - `buf.mode` is nonzero and the user is not super-user.

- **EINVAL**
  - An attempt is made to set `buf.offset` to a value outside of the range $-131071$ to $+131071$, or to set `buf.status` to a value other than those listed above, or to set `buf.tick` to a value outside of the range $900000/\text{HZ}$ to $1100000/\text{HZ}$, where HZ is the system timer interrupt frequency.
asprintf

Name

asprintf — write formatted output to a dynamically allocated string

Synopsis

```c
#include <stdio.h>

int asprintf(char ** restrict ptr, const char * restrict format ...);
```

Description

The `asprintf` function shall behave as `sprintf`, except that the output string shall be dynamically allocated space of sufficient length to hold the resulting string. The address of this dynamically allocated string shall be stored in the location referenced by `ptr`.

Return Value

Refer to `fprintf`.

Errors

Refer to `fprintf`.
bind_textdomain_codeset

Name
bind_textdomain_codeset — specify encoding for message retrieval

Synopsis
```
#include <libintl.h>
char * bind_textdomain_codeset (const char * domainname, const char * codeset);
```

Description
The bind_textdomain_codeset function can be used to specify the output codeset for message catalogs for domain domainname. The codeset argument shall be a valid codeset name which can be used for the iconv_open function, or a null pointer. If the codeset argument is the null pointer, then function returns the currently selected codeset for the domain with the name domainname. It shall return a null pointer if no codeset has yet been selected
Each successive call to bind_textdomain_codeset function overrides the settings made by the preceding call with the same domainname.
The bind_textdomain_codeset function shall return a pointer to a string containing the name of the selected codeset. The string shall be allocated internally in the function and shall not be changed or freed by the user.
The bind_textdomain_codeset function returns a pointer to a string containing the name of the selected codeset. The string is allocated internally in the function and shall not be changed by the user.

Parameters
domainname
The domainname argument is applied to the currently active LC_MESSAGE locale. It is equivalent in syntax and meaning to the domainname argument to textdomain, except that the selection of the domain is valid only for the duration of the call.
codeset
The name of the output codeset for the selected domain, or NULL to select the current codeset.
If domainname is the null pointer, or is an empty string, bind_textdomain_codeset shall fail, but need not set errno.

Return Value
Returns the currently selected codeset name. It returns a null pointer if no codeset has yet been selected.

Errors
ENOMEM
Insufficient memory available to allocate return value.
See Also

ggettext (baselib-gettext.html), dgettext, ngettext, dngettext, dcgettext, dcnggettext, textdomain, bindtextdomain

bindresvport

Name

bindresvport — bind socket to privileged IP port

Synopsis

#include <sys/types.h>
#include <rpc.rpc.h>

int bindresvport(int sd, struct sockaddr_in *sin);

Description

If the process has appropriate privilege, the bindresvport function shall bind a socket to a privileged IP port.

Return Value

On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors

EPERM

The process did not have appropriate privilege.

EPFNOSUPPORT

Address of sin did not match address family of sd.
bindtextdomain

Name
bindtextdomain — specify the location of a message catalog

Synopsis
#include <libintl.h>
char *bindtextdomain(const char *domainname, const char *dirname);

Description
The bindtextdomain shall set the base directory of the hierarchy containing message catalogs for a given message domain.
The bindtextdomain function specifies that the domainname message catalog can be found in the dirname directory hierarchy, rather than in the system default locale data base.
If dirname is not NULL, the base directory for message catalogs belonging to domain domainname shall be set to dirname. If dirname is NULL, the base directory for message catalogs shall not be altered.
The function shall make copies of the argument strings as needed.
dirname can be an absolute or relative pathname.
Applications that wish to use chdir should always use absolute pathnames to avoid misadventently selecting the wrong or non-existant directory.
If domainname is the null pointer, or is an empty string, bindtextdomain shall fail, but need not set errno.
The bindtextdomain function shall return a pointer to a string containing the name of the selected directory. The string shall be allocated internally in the function and shall not be changed or freed by the user.

Return Value
On success, bindtextdomain shall return a pointer to a string containing the directory pathname currently bound to the domain. On failure, a NULL pointer is returned, and the global variable errno may be set to indicate the error.

Errors
ENOMEM
Insufficient memory was available.

See Also
gettext (baselib-gettext.html), dgettext, ngettext, dnlgettext, dnlgettext, dcngettext, dcngettext, textdomain, bind_textdomain_codeset
**cfmakeraw**

**Name**

`cfmakeraw` — get and set terminal attributes

**Synopsis**

```c
#include <termios.h>

void cfmakeraw(struct termios *termios_p);
```

**Description**

The `cfmakeraw` function shall set the attributes of the termios structure referenced by `termios_p` as follows:

```c
termios_p->c_iflag &= ~(IGNBRK|BRKINT|PARMRK|ISTRIP |INLCR|IGNCR|ICRNL|IXON);

termios_p->c_oflag &= ~OPOST;

termios_p->c_lflag &= ~(ECHO|ECHONL|ICANON|ISIG|IEXTEN);

termios_p->c_cflag &= ~(CSIZE|PARENB);

termios_p->c_cflag |= CS8;
```

`termios_p` shall point to a termios structure that contains the following members:

```c
tcflag_t c_iflag;       /* input modes */
tcflag_t c_oflag;      /* output modes */
tcflag_t c_cflag;      /* control modes */
tcflag_t c_lflag;      /* local modes */
cc_t c_cc[NCCS];       /* control chars */
```
**cfsetspeed**

**Name**

`cfsetspeed` — set terminal input and output data rate

**Synopsis**

```c
#include <termios.h>

int cfsetspeed(struct termios *t, speedt speed);
```

**Description**

`cfsetspeed` sets the baud rate values in the termios structure. The effects of the function on the terminal as described below do not become effective, nor are all errors detected, until the `tcsetattr` function is called. Certain values for baud rates set in termios and passed to `tcsetattr` have special meanings.

**Getting and Setting the Baud Rate**

Input and output baud rates are found in the termios structure. The unsigned integer `speed_t` is typedef'd in the include file `termios.h`. The value of the integer corresponds directly to the baud rate being represented; however, the following symbolic values are defined.

```c
#define B0      0
#define B50     50
#define B75     75
#define B110    110
#define B134    134
#define B150    150
#define B200    200
#define B300    300
#define B600    600
#define B1200   1200
#define B1800   1800
#define B2400   2400
#define B4800   4800
#define B9600   9600
#define B19200  19200
#define B38400  38400
#ifndef _POSIX_SOURCE
#define EXTA    19200
#define EXTB    38400
#endif
```
cfsetspeed sets both the input and output baud rates in the termios structure referenced by t to speed.

Return Value

On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors

EINVAL

Invalid speed argument

creat

Name

creat — open a file

Description

creat is as specified in ISO POSIX (2003), but with differences as listed below.

May return ENODEV in place of ENXIO

Where the ISO POSIX (2003) specifies an ENXIO return, the implementation may return either ENXIO or ENODEV. Implementations are encouraged to return ENXIO.

As of spring 2004, we don't know of any Linux kernel patches to switch to ENXIO, but we believe that such a kernel patch would be accepted if submitted.
**daemon**

**Name**

daemon — run in the background

**Synopsis**

```c
#include <unistd.h>
int daemon(int nochdir, int noclose);
```

**Description**

The `daemon` function shall create a new process, detached from the controlling terminal. If successful, the calling process shall exit and the new process shall continue to execute the application in the background. If `nochdir` evaluates to true, the current directory shall not be changed. Otherwise, `daemon` shall change the current working directory to the root (`/`). If `noclose` evaluates to true the standard input, standard output, and standard error file descriptors shall not be altered. Otherwise, `daemon` shall close the standard input, standard output and standard error file descriptors and reopen them attached to `/dev/null`.

**Return Value**

On error, -1 is returned, and the global variable `errno` is set to any of the errors specified for the library functions `fork` and `setsid`.

**dcgettext**

**Name**

dcgettext — perform domain and category specific lookup in message catalog

**Synopsis**

```c
#include <libintl.h>
```
Chapter 1. Libraries

#include <locale.h>
char *dcgettext(const char *domainname, const char *msgid, int category);

Description

The dcgettext function is a domain specified version of gettext.

The dcgettext function shall lookup the translation in the current locale of the message identified by msgid in the domain specified by domainname and in the locale category specified by category. If domainname is NULL, the current default domain shall be used. The msgid argument shall be a NULL-terminated string to be matched in the catalogue. category shall specify the locale category to be used for retrieving message strings. The category parameter shall be one of LC_CTYPE, LC_COLLATE, LC_MESSAGES, LC_MONETARY, LC_NUMERIC, or LC_TIME. The default domain shall not be changed by a call to dcgettext.

Return Value

If a translation was found in one of the specified catalogs, it shall be converted to the current locale's codeset and returned. The resulting NULL-terminated string shall be allocated by the dcgettext function, and must not be modified or freed. If no translation was found, or category was invalid, msgid shall be returned.

Errors

dcgettext shall not modify the errno global variable.

See Also

ggettext (baselib-gettext.html), dgettext, ngettext, dgettext, dcgettext, textdomain, bindtextdomain, bind_textdomain_codeset

dcngettext

Name

dcngettext — perform domain and category specific lookup in message catalog with plural

Synopsis

#include <libintl.h>
Chapter 1. Libraries

#include <locale.h>

char *dcngettext(const char *domainname, const char *msgid1, const char *msgid2, unsigned long int n, int category);

Description

The dcngettext function is a domain specific version of gettext, capable of returning either a singular or plural form of the message. The dcngettext function shall lookup the translation in the current locale of the message identified by msgid1 in the domain specified by domainname and in the locale category specified by category. If domainname is NULL, the current default domain shall be used. The msgid1 argument shall be a NULL-terminated string to be matched in the catalogue. category shall specify the locale category to be used for retrieving message strings. The category parameter shall be one of LC_CTYPE, LC_COLLATE, LC_MESSAGES, LC_MONETARY, LC_NUMERIC, or LC_TIME. The default domain shall not be changed by a call to dgettext. If n is 1 then the singular version of the message is returned, otherwise one of the plural forms is returned, depending on the value of n and the current locale settings.

Return Value

If a translation corresponding to the value of n was found in one of the specified catalogs for msgid1, it shall be converted to the current locale's codeset and returned. The resulting NULL-terminated string shall be allocated by the dcngettext function, and must not be modified or freed. If no translation was found, or category was invalid, msgid1 shall be returned if n has the value 1, otherwise msgid2 shall be returned.

Errors

dcngettext shall not modify the errno global variable.

See Also

ggettext (baselib-gettext.html), dgettext, ngettext, dgettext, dgettext, dcgettext, textdomain, bindtextdomain, bind_textdomain_codeset
dgettext

Name

dgettext — perform lookup in message catalog for the current LC_MESSAGES locale

Synopsis

#include <libintl.h>

char *dgettext(const char *domainname, const char *msgid);

Description

dgettext is a domain specified version of gettext.

Parameters

domainname

dgettext applies domainname to the currently active LC_MESSAGE locale. This usage is equivalent in syntax and meaning to the textdomain function's application of domainname, except that the selection of the domain in dgettext is valid only for the duration of the call.

msgid

a NULL-terminated string to be matched in the catalogue with respect to a specific domain and the current locale.

Return Value

On success of a msgid query, the translated NULL-terminated string is returned. On error, the original msgid is returned. The length of the string returned is undetermined until dgettext is called.

Errors

dgettext will not modify the errno global variable.

See Also

gettext (baselib-gettext.html), dgettext, ngettext, dngettext, dcgettext, dcngettext, textdomain, bindtextdomain, bind_textdomain_codeset
**dngettext**

**Name**

dngettext — perform lookup in message catalog for the current locale

**Synopsis**

```c
#include <libintl.h>

char *dngettext(const char *domainname, const char *msgid1, const char *msgid2, unsigned long int n);
```

**Description**

dngettext shall be equivalent to a call to
dcngettext(domainname, msgid1, msgid2, n, LC_MESSAGES)

See dgettext for more information.

**See Also**

ggettext (baselib-gettext.html), dgettext, ngettext, dcgettext, dcngettext, textdomain, bindtextdomain,
bind_textdomain_codeset
err

Name
3194 err — display formatted error messages

Synopsis
3195 #include <err.h>
3196 void err(int eval, const char *fmt ...);

Description
3197 The err function shall display a formatted error message on the standard error stream. First, err shall write the last
3198 component of the program name, a colon character, and a space character. If fmt is non-NULL, it shall be used as a
3199 format string for the printf family of functions, and err shall write the formatted message, a colon character, and a
3200 space. Finally, the error message string affiliated with the current value of the global variable errno shall be written,
3201 followed by a newline character.
3202 The err function shall not return, the program shall terminate with the exit value of eval.

See Also
3203 error, errx

Return Value
3204 None.

Errors
3205 None.
error

Name

total — print error message

Synopsis

void error(int exitstatus, int errnum, const char *format ...);

Description
	error shall print a message to standard error.

total shall build the message from the following elements in their specified order:

1. the program name. If the application has provided a function named error_print_progmame, error shall call
   this to supply the program name; otherwise, error uses the content of the global variable program_name.

2. the colon and space characters, then the result of using the printf-style format and the optional arguments.

3. if errnum is nonzero, error shall add the colon and space characters, then the result of strerror(errnum).

4. a newline.

If exitstatus is nonzero, error shall call exit(exitstatus).

See Also

total, errx
errx

Name
errx — display formatted error message and exit

Synopsis
#include <err.h>

void errx(int eval, const char *fmt ...);

Description
The errx function shall display a formatted error message on the standard error stream. The last component of the program name, a colon character, and a space shall be output. If fmt is non-NULL, it shall be used as the format string for the printf family of functions, and the formatted error message, a colon character, and a space shall be output. The output shall be followed by a newline character.

errx does not return, but shall exit with the value of eval.

Return Value
None.

Errors
None.

See Also
error, err

fcntl

Name
fcntl — file control

Description
fcntl is as specified in ISO POSIX (2003), but with differences as listed below.

Implementation may set O_LARGEFILE
According to the Single UNIX Specification, only an application sets fcntl flags, for example O_LARGEFILE.
However, this specification also allows an implementation to set O_LARGEFILE in the case where the system default behavior matches the O_LARGEFILE behavior, for example if sizeof(off_t) is 8. Thus, calling fcntl with the F_GETFL command may return O_LARGEFILE as well as flags explicitly set by the application.
fflush_unlocked

Name

fflush_unlocked — non thread safe fflush

Description

fflush_unlocked is the same as fflush except that it need not be thread safe. That is, it may only be invoked in the ways which are legal for getc_unlocked.

fgetwc_unlocked

Name

fgetwc_unlocked — non thread safe fgetwc

Description

fgetwc_unlocked is the same as fgetwc except that it need not be thread safe. That is, it may only be invoked in the ways which are legal for getc_unlocked.
flock

Name

flock — apply or remove an advisory lock on an open file

Synopsis

int flock(int fd, int operation);

Description

flock applies or removes an advisory lock on the open file fd. Valid operation types are:

- LOCK_SH
  - Shared lock. More than one process may hold a shared lock for a given file at a given time.

- LOCK_EX
  - Exclusive lock. Only one process may hold an exclusive lock for a given file at a given time.

- LOCK_UN
  - Unlock.

- LOCK_NB
  - Don't block when locking. May be specified (by oring) along with one of the other operations.

A single file may not simultaneously have both shared and exclusive locks.

Return Value

On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors

- EWOULDBLOCK
  - The file is locked and the LOCK_NB flag was selected.
fopen

Name

fopen — open a file

Description

fopen is as specified in ISO POSIX (2003), but with differences as listed below.

May return ENODEV in place of ENXIO

Where the ISO POSIX (2003) specifies an ENXIO return, the implementation may return either ENXIO or ENODEV. Implementations are encouraged to return ENXIO.

As of spring 2004, we don't know of any Linux kernel patches to switch to ENXIO, but we believe that such a kernel patch would be accepted if submitted.

freopen

Name

freopen — open a file

Description

freopen is as specified in ISO POSIX (2003), but with differences as listed below.

May return ENODEV in place of ENXIO

Where the ISO POSIX (2003) specifies an ENXIO return, the implementation may return either ENXIO or ENODEV. Implementations are encouraged to return ENXIO.

As of spring 2004, we don't know of any Linux kernel patches to switch to ENXIO, but we believe that such a kernel patch would be accepted if submitted.
**getdomainname**

**Name**

getdomainname — get NIS domain name (DEPRECATED).

**Synopsis**

```c
#include <unistd.h>

int getdomainname (char *name, size_t namelen);
```

**Description**

If the Network Information System (NIS) is in use, `getdomainname` shall copy the NIS domain name to the supplied buffer identified by `name`, with maximum length `namelen`. If the NIS domain name is not currently set, `getdomainname` shall copy the string "(none)" to the `name`. If `namelen` is less the length of the string to be copied, `getdomainname` may either truncate the string to `namelen` characters and place it in `name` (without a terminating null character), or may fail with EINVAL.

Note that the NIS domain name is not the same as the domain portion of a fully qualified domain name (for example, in DNS).

**Return Value**

On success, `getdomainname` shall return 0. Otherwise, it shall return -1 and set `errno` to indicate the error.

**Errors**

EINVAL

- `name` was a null pointer.

EINVAL

- The buffer identified by `name` and `namelen` is of insufficient size to store the NIS domain name string, and the implementation considers this an error.

**Future Directions**

The LSB does not include other NIS interfaces, and a future version of this specification may deprecate this interface.

Application developers should avoid using this interface where possible.
gethostbyname_r

Name

gethostbyname_r — find network host database entry matching host name (DEPRECATED)

Synopsis

```c
int gethostbyname_r(__const char *__restrict __name, (struct hostent *__restrict __result_buf), char *__restrict __buf, size_t __buflen, (struct hostent **__restrict __result), int *__restrict __h_errno);
```

Description

The gethostbyname_r function is deprecated; applications should call getaddrinfo instead.

gethostbyname_r is a reentrant version of gethostbyname that searches the network host database for a host name match.

getloadavg

Name

getloadavg — get system load averages

Synopsis

```c
#include <stdlib.h>
int getloadavg(double loadavg[], int nelem);
```

Description

getloadavg returns the number of processes in the system run queue averaged over various periods of time. Up to nelem samples are retrieved and assigned to successive elements of loadavg[]. The system imposes a maximum of 3 samples, representing averages over the last 1, 5, and 15 minutes, respectively.

getopt

Name

gopt — parse command line options

Synopsis

```c
#include <unistd.h>
int getopt(int argc, char * const argv[], const char *optstring);
extern char *optarg;
```
extern int optind, opterr, optopt;

Description
The getopt function shall parse command line arguments as described in ISO POSIX (2003), with the following
exceptions, where LSB and POSIX specifications vary. LSB systems shall implement the modified behaviors
described below.

Argument Ordering
The getopt function can process command line arguments referenced by argv in one of three ways:

PERMUTE
the order of arguments in argv is altered so that all options (and their arguments) are moved in front of all of the
operands. This is the default behavior.

This behavior has undefined results if argv is not modifiable. This is to support historic behavior predating
the use of const and ISO C (1999). The function prototype was aligned with ISO POSIX (2003) despite the
fact that it modifies argv, and the library maintainers are unwilling to change this.

REQUIRE_ORDER
The arguments in argv are processed in exactly the order given, and option processing stops when the first
non-option argument is reached, or when the element of argv is "--". This ordering can be enforced either by
setting the environment variable POSIXLY_CORRECT, or by setting the first character of optstring to '+'.

RETURN_IN_ORDER
The order of arguments is not altered, and all arguments are processed. Non-option arguments (operands) are
handled as if they were the argument to an option with the value 1 ('\001'). This ordering is selected by setting the
first character of optstring to '-'

Option Characteristics

LSB specifies that:
- an element of argv that starts with "-" (and is not exactly "-" or "--") is an option element.
- characters of an option element, aside from the initial "-", are option characters.

POSIX specifies that:
- applications using getopt shall obey the following syntax guidelines:
  - option name is a single alphanumeric character from the portable character set
  - option is preceded by the '-' delimiter character
  - options without option-arguments should be accepted when grouped behind one '-' delimiter
  - each option and option-argument is a separate argument
  - option-arguments are not optional
  - all options should precede operands on the command line
  - the argument "--" is accepted as a delimiter indicating the end of options and the consideration of subsequent
    arguments, if any, as operands
Chapter 1. Libraries

- historical implementations of `getopt` support other characters as options as an allowed extension, but applications that use extensions are not maximally portable.
- support for multi-byte option characters is only possible when such characters can be represented as type `int`.
- applications that call any utility with a first operand starting with `-' should usually specify `"--"` to mark the end of the options. Standard utilities that do not support this guideline indicate that fact in the OPTIONS section of the utility description.

Extensions

`LSB` specifies that:

* if a character is followed by two colons, the option takes an optional argument; if there is text in the current `argv` element, it is returned in `optarg`, otherwise `optarg` is set to 0.
* if `optstring` contains `W` followed by a semi-colon (`;`), then `--W foo` is treated as the long option `--foo`.

See `getopt_long` for a description of long options.

* The first character of `optstring` shall modify the behavior of `getopt` as follows:
  * if the first character is `+'`, then `REQUIRE_ORDER` processing shall be in effect (see above)
  * if the first character is `-'`, then `RETURN_IN_ORDER` processing shall be in effect (see above)
  * if the first character is `:`, then `getopt` shall return `:` instead of `?` to indicate a missing option argument, and shall not print any diagnostic message to `stderr`.

`POSIX` specifies that:

* the `--W` option is reserved for implementation extensions.

Return Values

`LSB` specifies the following additional `getopt` return values:

* `\001` is returned if `RETURN_IN_ORDER` argument ordering is in effect, and the next argument is an operand, not an option. The argument is available in `optarg`.

Any other return value has the same meaning as for `POSIX`.

`POSIX` specifies the following `getopt` return values:

* the next option character is returned, if found successfully.
* `:` is returned if a parameter is missing for one of the options and the first character of `optstring` is `:`.
* `?` is returned if an unknown option character not in `optstring` is encountered, or if `getopt` detects a missing argument and the first character of `optstring` is not `:`.
* `-1` is returned for the end of the option list.

Environment Variables

`LSB` specifies that:

* if the variable `POSIXLY_CORRECT` is set, option processing stops as soon as a non-option argument is encountered.
the variable _[PID]_GNU_nonoption_argv_flags_(where [PID] is the process ID for the current process), contains a space separated list of arguments that should not be treated as arguments even though they appear to be so.

Rationale
This was used by bash 2.0 to communicate to GNU libc which arguments resulted from wildcard expansion and so should not be considered as options. This behavior was removed in bash version 2.01, but the support remains in GNU libc.
This behavior is DEPRECATED in this version of the LSB; future revisions of this specification may not include this requirement.

getopt_long

Name
getopt_long — parse command line options

Synopsis
#define _GNU_SOURCE
#include <getopt.h>
int getopt_long(int argc, char * const argv[], const char *opstring, (const struct option *
longopts), int *longindex);

Description
g getopt_long works like getopt except that it also accepts long options, started out by two dashes. Long option names may be abbreviated if the abbreviation is unique or is an exact match for some defined option. A long option may take a parameter, of the form --arg=param or --arg param.

 longopts is a pointer to the first element of an array of struct option declared in getopt.h as:

 struct option {
 const char *name;
 int has_arg;
 int *flag;
 int val;
 };

The fields in this structure have the following meaning:

 name
 The name of the long option.

 has_arg
 One of:
argument (or 0) if the option does not take an argument,
required_argument (or 1) if the option requires an argument, or
optional_argument (or 2) if the option takes an optional argument.

flag

specifies how results are returned for a long option. If flag is NULL, then getopt_long shall return val. (For example, the calling program may set val to the equivalent short option character.) Otherwise, getopt_long returns 0, and flag shall point to a variable which shall be set to val if the option is found, but left unchanged if the option is not found.

val

The value to return, or to load into the variable pointed to by flag.

Return Value

gopt_long returns the option character if a short option was found successfully, or ":" if there was a missing parameter for one of the options, or "?" for an unknown option character, or -1 for the end of the option list.

For a long option, getopt_long returns val if flag is NULL, and 0 otherwise. Error and -1 returns are the same as for getopt, plus "?" for an ambiguous match or an extraneous parameter.

gopt_long_only

Name

gopt_long_only — parse command line options

Synopsis

#define _GNU_SOURCE
#include <getopt.h>
int getopt_long_only(int argc, char * const argv[], const char *optstring, (const struct
option *longopts), int *longindex);

Description

gopt_long_only is like getopt_long, but "-" as well as "--" can indicate a long option. If an option that starts with "-" (not "--") doesn't match a long option, but does match a short option, it is parsed as a short option instead.

Return Value

gopt_long_only returns the option character if the option was found successfully, or ":" if there was a missing parameter for one of the options, or "?" for an unknown option character, or -1 for the end of the option list.

gopt_long_only also returns the option character when a short option is recognized. For a long option, they return val if flag is NULL, and 0 otherwise. Error and -1 returns are the same as for getopt, plus "?" for an ambiguous match or an extraneous parameter.
gettext

Name
gettext — Search message catalogs for a string

Synopsis
#include <libintl.h>
char *gettext(const char *msgid);

Description
The gettext function shall search the currently selected message catalogs for a string identified by the string msgid. If a string is located, that string shall be returned. The gettext function is equivalent to dcgettext(NULL, msgid, LC_MESSAGES).

Return Value
If a string is found in the currently selected message catalogs for msgid, then a pointer to that string shall be returned. Otherwise, a pointer to msgid shall be returned. Applications shall not modify the string returned by gettext.

Errors
None.
The gettext function shall not modify errno.

See Also
dgettext, ngettext, dnlgettext, dcgettext, dnlcgettext, textdomain, bindtextdomain, bind_textdomain_codeset
getutent

Name
getutent — access user accounting database entries

Synopsis
#include <utmp.h>
struct utmp *getutent(void);

Description
The getutent function shall read the next entry from the user accounting database.

Return Value
Upon successful completion, getutent shall return a pointer to a utmp structure containing a copy of the requested entry in the user accounting database. Otherwise, a null pointer shall be returned. The return value may point to a static area which is overwritten by a subsequent call to getutent.

Errors
None defined.

getutent_r

Name
getutent_r — access user accounting database entries

Synopsis
int getutent_r(struct utmp * buffer, struct utmp ** result);

Description
The getutent_r function is a reentrant version of the getutent function. On entry, buffer should point to a user supplied buffer to which the next entry in the database will be copied, and result should point to a location where the result will be stored.

Return Value
On success, getutent_r shall return 0 and set the location referenced by result to a pointer to buffer. Otherwise, getutent_r shall return −1 and set the location referenced by result to NULL.
glob64

Name

glob64 — find pathnames matching a pattern (Large File Support)

Synopsis

#include <glob.h>

int glob64(const char *pattern, int flags, int (*errfunc) (const char *, int), glob64_t *pglob);

Description

The glob64 function is a large-file version of the glob defined in ISO POSIX (2003). It shall search for pathnames matching pattern according to the rules used by the shell, /bin/sh. No tilde expansion or parameter substitution is done; see wordexp.

The results of a glob64 call are stored in the structure pointed to by pglob, which is a glob64_t declared in glob.h with the following members:

typedef struct
{
  size_t gl_pathc;
  char **gl_pathv;
  size_t gl_offs;
  int gl_flags;
  void (*gl_closedir) (void *);
  struct dirent64 *(*gl_readdir64) (void *);
  void (*gl_opendir) (const char *);
  int (*gl_lstat) (const char *, struct stat *);
  int (*gl_stat) (const char *, struct stat *);
}
glob64_t;

Structure members with the same name as corresponding members of a \texttt{glob_t} as defined in ISO POSIX (2003) shall have the same purpose.

Other members are defined as follows:

\texttt{gl\_flags}

- reserved for internal use

\texttt{gl\_closedir}

- pointer to a function capable of closing a directory opened by \texttt{gl\_opendir}

\texttt{gl\_readdir64}

- pointer to a function capable of reading entries in a large directory

\texttt{gl\_opendir}

- pointer to a function capable of opening a large directory

\texttt{gl\_stat}

- pointer to a function capable of returning file status for a large file

\texttt{gl\_lstat}

- pointer to a function capable of returning file status information for a large file or symbolic link

A large file or large directory is one with a size which cannot be represented by a variable of type \texttt{off_t}.

**Return Value**

On success, 0 is returned. Other possible returns are:

GLOB_NOSPACE
- out of memory

GLOB_ABORTED
- read error

GLOB_NOMATCH
- no match found
globfree64

Name
3497 globfree64 — free memory from glob64() (Large File Support)

Synopsis
3498 #include <glob.h>
3499 void globfree64(glob64_t *pglob);

Description
3500 globfree64 frees the dynamically allocated storage from an earlier call to glob64.
3501 globfree64 is a 64-bit version of globfree.

initgroups

Name
3502 initgroups — initialize the supplementary group access list

Synopsis
3503 #include <grp.h>
Chapter 1. Libraries

#include <sys/types.h>

int initgroups(const char *user, gid_t group);

Description

If the process has appropriate privilege, the initgroups function shall initialize the Supplementary Group IDs for the current process by reading the group database and using all groups of which user is a member. The additional group group is also added to the list.

Return Value

On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors

EPERM
The calling process does not have sufficient privileges.

ENOMEM
Insufficient memory to allocate group information structure.

See Also

setgroups
ioctl

Name
ioctl — control device

Synopsis
#include <sys/ioctl.h>

int ioctl (int d, int request, ...);

Description
The ioctl function shall manipulate the underlying device parameters of special files. \( d \) shall be an open file descriptor referring to a special file. The ioctl function shall take three parameters; the type and value of the third parameter is dependent on the device and request.

Conforming LSB applications shall not call ioctl except in situations explicitly stated in this specification.

Return Value
On success, 0 is returned. An ioctl may use the return value as an output parameter and return a non-negative value on success. On error, -1 is returned and the global variable errno is set appropriately.

Errors
EBADF
\( d \) is not a valid descriptor.

EFAULT
The third parameter references an inaccessible memory area.

EINVAL
request or the third parameter is not valid.

ENOTTY
\( d \) is not associated with a character special device.

ENOTTY
The specified request does not apply to the kind of object that \( d \) references.
sockio

Name

sockio — socket ioctl commands

Synopsis

#include <sys/socket.h>
#include <net/if.h>
#include <netinet/in.h>

int ioctl(int sockfd, int request, char *argp);

## Description

Socket ioctl commands are a subset of the ioctl calls, which can perform a variety of functions on sockets.

`sockfd` shall contain the value of a file descriptor that was created with the `socket` or `accept` calls.

Socket ioctl commands apply to the underlying network interfaces, and affect the entire system, not just the file descriptor used to issue the `ioctl`.

The following values for `request` are accepted:

### SIOCGIFCONF

Gets the interface configuration list for the system.

SIOCGIFCONF is similar to the `if_nameindex` family found in the ISO POSIX (2003) or the `getifaddrs` family found in BSD derived systems.

`argp` shall point to a `ifconf` structure, as described in `<net/if.h>`. Before calling, the caller shall set the `ifc_ifcu.ifcu_req` field to point to an array of `ifreq` structures, and set `ifc_len` to the size in bytes of this allocated array. Upon return, `ifc_len` will contain the size in bytes of the array which was actually used. If it is the same as the length upon calling, the caller should assume that the array was too small and try again with a larger array.

On success, SIOCGIFCONF can return any nonnegative value.

### Rationale

Historical UNIX systems disagree on the meaning of the return value.

### SIOCGIFFLAGS

Gets the interface flags for the indicated interface. `argp` shall point to a `ifreq` structure. Before calling, the caller should fill in the `ifr_name` field with the interface name, and upon return, the `ifr_ifru.ifru_flags` field is set with the interface flags.

### SIOCGIFADDR

Gets the interface address for the given interface. `argp` shall point to a `ifreq` structure. Before calling, the caller should fill in the `ifr_name` field with the interface name, and upon return, the `ifr_ifru.ifru_addr` field is set with the interface address.

### SIOCGIFNETMASK

Gets the network mask for the given interface. `argp` shall point to a `ifreq` structure. Before calling, the caller should fill in the `ifr_name` field with the interface name, and upon return, the `ifr_ifru.ifru_netmask` field is set with the network mask.

### FIONREAD

Returns the amount of queued unread data in the receive buffer. `argp` shall point to an integer where the result is to be placed.
Return Value

On success, if request is SIOCGIFCONF, a non-negative integer shall be returned. If request is not SIOCGIFCONF, on success 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors

EBADF

sockfd is not a valid descriptor.

EFAULT

argp references an inaccessible memory area.

ENOTTY

The specified request does not apply to the kind of object that the descriptor sockfd references.

EINVAL

Either request or argp is invalid.

ENOTCONN

The operation is only defined on a connected socket, but the socket wasn't connected.

kill

Name

kill — send a signal

Synopsis

#include <signal.h>

int kill(pid_t pid, int sig);

Description

kill is as specified in the ISO POSIX (2003), but with differences as listed below.

Process ID -1 doesn’t affect calling process

If pid is specified as -1, sig shall not be sent to the calling process. Other than this, the rules in the ISO POSIX (2003) apply.

Rationale

This was a deliberate Linus decision after an unpopular experiment in including the calling process in the 2.5.1 kernel. See "What does it mean to signal everybody?", Linux Weekly News, 20 December 2001, http://lwn.net/2001/1220/kernel.php3
mbsnrtowcs

Name
mbsnrtowcs — convert a multibyte string to a wide character string

Synopsis

```c
#include <wchar.h>

size_t mbsnrtowcs(wchar_t *dest, const char **src, size_t nms, size_t len, mbstate_t *ps);
```

Description
mbsnrtowcs is like mbsrtowcs, except that the number of bytes to be converted, starting at src, is limited to nms.

If dest is not a NULL pointer, mbsnrtowcs converts at most nms bytes from the multibyte string src to a wide-character string starting at dest. At most, len wide characters are written to dest. The state ps is updated.

The conversion is effectively performed by repeatedly calling:
where \( n \) is some positive number, as long as this call succeeds, and then incrementing \( dest \) by one and \( src \) by the number of bytes consumed.

The conversion can stop for three reasons:

- An invalid multibyte sequence has been encountered. In this case \( src \) is left pointing to the invalid multibyte sequence, \((\text{size}_t)(-1)\) is returned, and \( \text{errno} \) is set to EILSEQ.
- The \( nms \) limit forces a stop, or \( len \) non-L'0' wide characters have been stored at \( dest \). In this case, \( src \) is left pointing to the next multibyte sequence to be converted, and the number of wide characters written to \( dest \) is returned.
- The multibyte string has been completely converted, including the terminating '0' (which has the side effect of bringing back \( ps \) to the initial state). In this case, \( src \) is set to NULL, and the number of wide characters written to \( dest \), excluding the terminating L'0' character, is returned.

If \( dest \) is NULL, \( len \) is ignored, and the conversion proceeds as above, except that the converted wide characters are not written out to memory, and that no destination length limit exists.

In both of the above cases, if \( ps \) is a NULL pointer, a static anonymous state only known to \text{mbsnrtowcs} \ is used instead.

The programmer shall ensure that there is room for at least \( len \) wide characters at \( dest \).

**Return Value**

\text{mbsnrtowcs} returns the number of wide characters that make up the converted part of the wide character string, not including the terminating null wide character. If an invalid multibyte sequence was encountered, \((\text{size}_t)(-1)\) is returned, and the global variable \( \text{errno} \) is set to EILSEQ.

**Notes**

The behavior of \text{mbsnrtowcs} depends on the \text{LC_CTYPE} category of the current locale.

Passing NULL as \( ps \) is not multi-thread safe.

**memmem**

**Name**

\text{memmem} — locate bytes

**Synopsis**

\#define \_GNU\_SOURCE
#include <string.h>

void *memmem(const void *haystack, size_t haystacklen, const void *needle, size_t needlelen);

**Description**

`memmem` finds the start of the first occurrence of the byte array referenced by `needle` of length `needlelen` in the memory area `haystack` of length `haystacklen`.

**Return Value**

`memmem` returns a pointer to the beginning of the byte array, or `NULL` if the byte array is not found.

**Notes**

Earlier versions of the C library (prior to glibc 2.1) contained a `memmem` with various problems, and application developers should treat this function with care.

**memrchr**

**Name**

`memrchr` — scan memory for a character

**Synopsis**

```c
#include <string.h>
void *memrchr(const void *s, int c, size_t n);
```

**Description**

The `memrchr` function shall locate the last occurrence of `c` (converted to an unsigned char) in the initial `n` bytes (each interpreted as an unsigned char) of the object pointed to by `s`.

**Return Value**

The `memrchr` shall return a pointer to the located byte, or a null pointer if the byte does not occur in the object.

**Errors**

No errors are defined.

**See Also**

`memchr`
ngettext

Name
ngettext — Search message catalogs for plural string

Synopsis
#include <libintl.h>
char *ngettext(const char *msgid1, const char *msgid2, unsigned long int n);

Description
The ngettext function shall search the currently selected message catalogs for a string matching the singular string msgid1. If a string is located, and if \( n \) is 1, that string shall be returned. If \( n \) is not 1, a pluralized version (dependant on \( n \)) of the string shall be returned.

The ngettext function is equivalent to dcgettext(NULL, msgid1, msgid2, n, LC_MESSAGES).

Return Value
If a string is found in the currently selected message catalogs for msgid1, then if \( n \) is 1 a pointer to the located string shall be returned. If \( n \) is not 1, a pointer to an appropriately pluralized version of the string shall be returned. If no message could be found in the currently selected message catalogs, then if \( n \) is 1, a pointer to msgid1 shall be returned, otherwise a pointer to msgid2 shall be returned.

Applications shall not modify the string returned by ngettext.

Errors
None.
The ngettext function shall not modify errno.

See Also
ggettext (baselib-gettext.html), dgettext, ngettext, dngettext, dcgettext, dcngettext, textdomain, bindtextdomain, bind_textdomain_codeset
obstack_free

Name

obstack_free — free an object in the obstack

Synopsis

#include <obstack.h>

void obstack_free(struct obstack *obstack, void *block);

Description

obstack_free frees an object in the obstack.

Future Directions

Future versions of this specification may not include support for this interface.

open

Name

open — open a file

Synopsis

#include <sys/stat.h>

#include <fcntl.h>

int open(const char *path, int oflag, ...);

Description

The open function shall behave as specified in ISO POSIX (2003), except with differences as listed below.

May return ENODEV in place of ENXIO

Where ISO POSIX (2003) specifies an ENXIO return, a conforming implementation may return either ENXIO or ENODEV. Implementations are encouraged to return ENXIO.

Rationale

As of spring 2004, no Linux kernel patches to switch to ENXIO are known, but it is believed that such a kernel patch would be accepted if submitted.
opterr

Name
3674 opterr — external variable used in getopt()

Synopsis
3675 extern int opterr;

Description
3676 opterr is used as a flag to suppress an error message generated by getopt. When opterr is set to 0, it suppresses the error message generated by getopt when that function does not recognize an option character.

optind

Name
3678 optind — external variable used in getopt()

Synopsis
3679 extern int optind;

Description
3680 optind holds the current index of the array argv[], which contains the command line options being parsed by getopt.

optopt

Name
3682 optopt — external variable used in getopt()

Synopsis
3683 extern int optopt;

Description
3684 optopt holds the unknown option character when that option character is not recognized by getopt.
**pmap_getport**

**Name**

pmap_getport — Find the port number assigned to a service registered with a portmapper.

**Synopsis**

```c
#include <pmap_clnt.h>

u_short *pmap_getport(struct sockaddr_in *address, __const u_long program, __const u_long *version, u_int protocol);
```

**Description**

The **pmap_getport** function shall return the port number assigned to a service registered with a RPC Binding service running on a given target system, using the protocol described in RFC 1833: Binding Protocols for ONC RPC Version 2. The **pmap_getport** function shall be called given the RPC program number `program`, the program version `version`, and transport protocol `protocol`. Conforming implementations shall support both IPPROTO_UDP and IPPROTO_TCP protocols. On entry, `address` shall specify the address of the system on which the portmapper to be contacted resides. The value of `address->sin_port` shall be ignored, and the standard value for the portmapper port shall always be used.

Security and network restrictions may prevent a conforming application from contacting a remote RPC Binding Service.

**Return Value**

On success, the **pmap_getport** function shall return the port number in host byte order of the RPC application registered with the remote portmapper. On failure, if either the program was not registered or the remote portmapper service could not be reached, the **pmap_getport** function shall return 0. If the remote portmap service could not be reached, the status is left in the global variable `rpc_createerr`. 
**pmap_set**

**Name**

pmap_set — Establishes mapping to machine's RPC Bind service.

**Synopsis**

```c
#include <rpc/pmap_clnt.h>

*pmap_set(__const u_long program, __const u_long version, int protocol, u_short port);
```

**Description**

pmap_set establishes a mapping between the triple \( [\text{program}, \text{version}, \text{protocol}] \) and \( \text{port} \) on the machine's RPC Bind service. The value of \( \text{protocol} \) is most likely IPPROTO_UDP or IPPROTO_TCP. Automatically done by svc_register.

**Return Value**

pmap_set returns 1 if it succeeds, 0 otherwise.

**pmap_unset**

**Name**

pmap_unset — Destroys RPC Binding

**Synopsis**

```c
#include <rpc/rpc.h>

void pmap_unset(u_long prognum, u_long versnum);
```

**Description**

As a user interface to the RPC Bind service, pmap_unset destroys all mapping between the triple \( [\text{prognum}, \text{versnum}, *] \) and \( \text{ports} \) on the machine's RPC Bind service.

**Return Value**

pmap_unset returns 1 if it succeeds, zero otherwise.
**psignal**

**Name**

psignal — print signal message

**Synopsis**

```c
#include <signal.h>

void psignal(int sig, const char *s);

extern const char *const sys_siglist[]
```

**Description**

The `psignal` function shall display a message on the `stderr` stream. If `s` is not the null pointer, and does not point to an empty string (e.g. "\0"), the message shall consist of the string `s`, a colon, a space, and a string describing the signal number `sig`; otherwise `psignal` shall display only a message describing the signal number `sig`. If `sig` is invalid, the message displayed shall indicate an unknown signal.

The array `sys_siglist` holds the signal description strings indexed by signal number.

**Return Value**

`psignal` returns no value.

**random_r**

**Name**

random_r — generate random number

**Synopsis**

```c
int random_r((struct random_data *__restrict __buf), int32_t *__restrict __result);
```

**Description**

`random_r` is a reentrant version of `random`, which generates a pseudorandom number.

**Future Directions**

Since this function requires support from other functions not specified in this specification (most notably `initstate_r`), a future version of this specification may deprecate this interface.
setbuffer

Name

setbuffer — stream buffering operation

Synopsis

```
#include <stdio.h>

void setbuffer(FILE *stream, char *buf, size_t size);
```

Description

setbuffer is an alias for the call to setvbuf. It works the same, except that the size of the buffer in setbuffer is up to the caller, rather than being determined by the default BUFSIZ.

setdomainname

Name

setdomainname — set NIS domain name (DEPRECATED).

Synopsis

```
#include <unistd.h>

int setdomainname (char *name, size_t namelen);
```

Description

If NIS is in use, set the NIS domain name. Note that this is not the same as the domain name which provides the domain portion of a fully qualified domain name (for example, in DNS). If NIS is not in use, this function may set the domain name anyway, or it may fail.

This call shall fail unless the caller has appropriate privileges.

`namelen` shall be the length of the string pointed to by `name`.

Return Value

On success, `setdomainname` shall return 0. Otherwise, it shall return -1 and set `errno` to indicate the error.

Errors

`EPERM`

The process did not have sufficient privilege to set the domain name.

`EINVAL`

`name` is a null pointer.
setgroups

Name

`setgroups` — set list of supplementary group IDs

Synopsis

```c
#include <grp.h>
int setgroups(size_t size, const gid_t *list);
```

Description

If the process has appropriate privilege, the `setgroups` function shall set the supplementary group IDs for the current process. `list` shall reference an array of `size` group IDs. A process may have at most `NGROUPS_MAX` supplementary group IDs.

Return Value

On successful completion, 0 is returned. On error, -1 is returned and the `errno` is set to indicate the error.

Errors

- **EFAULT**
  - `list` has an invalid address.
- **EPERM**
  - The process does not have appropriate privileges.
- **EINVAL**
  - `size` is greater than `NGROUPS_MAX`. 
sethostid

Name
sethostid — set the unique identifier of the current host

Synopsis
#include <unistd.h>
int sethostid(long int hostid);

Description
sethostid sets a unique 32-bit identifier for the current machine. The 32-bit identifier is intended to be unique among all UNIX systems in existence. This normally resembles the Internet address for the local machine as returned by gethostbyname(3), and thus usually never needs to be set.

The sethostid call is restricted to the superuser.

hostid is stored in the file /etc/hostid.

Return Value
gethostid returns the 32-bit identifier for the current host as set by sethostid(2).

Files
/etc/hostid

sethostname

Name
sethostname — set host name

Synopsis
#include <unistd.h>
#include <sys/param.h.h>

Chapter 1. Libraries
#include <sys/utsname.h>

int sethostname(const char *name, size_t len);

### Description

If the process has appropriate privileges, the `sethostname` function shall change the host name for the current machine. The `name` shall point to a null-terminated string of at most `len` bytes that holds the new hostname.

If the symbol `HOST_NAME_MAX` is defined, or if `sysconf(_SC_HOST_NAME_MAX)` returns a value greater than 0, this value shall represent the maximum length of the new hostname. Otherwise, if the symbol `MAXHOSTLEN` is defined, this value shall represent the maximum length for the new hostname. If none of these values are defined, the maximum length shall be the size of the `nodename` field of the utsname structure.

### Return Value

On success, 0 is returned. On error, -1 is returned and the global variable `errno` is set appropriately.

### Errors

- **EINVAL**
  
  `len` is negative or larger than the maximum allowed size.

- **EPERM**
  
  the process did not have appropriate privilege.

- **EFAULT**
  
  `name` is an invalid address.

### Rationale

ISO POSIX (2003) guarantees that:

- Maximum length of a host name (not including the terminating null) as returned from the `gethostname` function shall be at least 255 bytes.

The glibc C library does not currently define `HOST_NAME_MAX`, and although it provides the name `__SC_HOST_NAME_MAX` a call to `sysconf` returns -1 and does not alter `errno` in this case (indicating that there is no restriction on the hostname length). However, the glibc manual indicates that some implementations may have `MAXHOSTNAMELEN` as a means of detecting the maximum length, while the Linux kernel at release 2.4 and 2.6 stores this hostname in the utsname structure. While the glibc manual suggests simply shortening the name until `sethostname` succeeds, the LSB requires that one of the first four mechanisms works. Future versions of glibc may provide a more reasonable result from `sysconf(__SC_HOST_NAME_MAX).`
setsockopt

Name
setsockopt — set options on sockets

Synopsis
#include <sys/socket.h>
#include <netinet/in.h>
int setsockopt(int sockfd, int level, int optname, void *optval, socklen_t optlen);

Description
In addition to the setsockopt options specified in SUSv3, setsockopt also supports the options specified here.
The following setsockopt operations are provided for level IPPROTO_IP:

IP_MULTICAST_TTL
Set or reads the time-to-live value of outgoing multicast packets for this socket. optval is a pointer to an integer
which contains the new TTL value.

IP_MULTICAST_LOOP
Sets a boolean flag indicating whether multicast packets originating locally should be looped back to the local
sockets. optval is a pointer to an integer which contains the new flag value.

IP_ADD_MEMBERSHIP
Join a multicast group. optval is a pointer to a ip_mreq structure. Before calling, the caller should fill in the
imr_multiaddr field with the multicast group address and the imr_address field with the address of the
local interface. If imr_address is set to INADDR_ANY, then an appropriate interface is chosen by the
system.

IP_DROP_MEMBERSHIP
Leave a multicast group. optval is a pointer to a ip_mreq structure containing the same values as were used
with IP_ADD_MEMBERSHIP.

IP_MULTICAST_IF
Set the local device for a multicast socket. optval is a pointer to a ip_mreq structure initialized in the same
manner as with IP_ADD_MEMBERSHIP.
The ip_mreq structure contains two struct in_addr fields: imr_multiaddr and imr_address.

Return Value
On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.
setutent

Name

3826 setutent — access user accounting database entries

Synopsis

3827 #include <utmp.h>
3828 void setutent(void);

Description

3829 The setutent function shall reset the user accounting database such that the next call to getutent shall be return the
3830 first record in the database. It is recommended to call it before any of the other functions that operate on the user
3831 accounting databases (e.g. getutent)

Return Value

3832 None.
sigandset

Name

sigandset — build a new signal set by combining the two input sets using logical AND

Synopsis

```c
#include <signal.h>

int sigandset(sigset_t *set, const sigset_t *left, const sigset_t *right);
```

Description

The sigandset shall combine the two signal sets referenced by `left` and `right`, using a logical AND operation, and shall place the result in the location referenced by `set`. The resulting signal set shall contain only signals that are in both the set referenced by `left` and the set referenced by `right`.

Return Value

On success, `sigandset` shall return 0. Otherwise, `sigandset` shall return -1 and set `errno` to indicate the error.

Errors

**EINVAL**

One or more of `set`, `left`, or `right` was a null pointer.

See Also

sigorset

sigblock

Name

sigblock — manipulate the signal mask

Synopsis

```c
#include _BSD_SOURCE
```
```plaintext
#include <signal.h>

int sigblock(int mask);

Description
The sigblock function shall add the signals corresponding to the bits set in mask to the set of signals currently being blocked from delivery.

Return Value
The sigblock function shall return the previous signal mask.

Errors
None.

Notes
sigblock is made obsolete by sigprocmask(2). A future version of this specification may deprecate this function.

siggetmask

Name
siggetmask — manipulate the signal mask

Synopsis
#define _BSD_SOURCE
#include <signal.h>
int siggetmask(void);

Description
The siggetmask function shall return the current set of masked signals.

Notes
siggetmask is made obsolete by sigprocmask(2).
```
sigisemptyset

Name

sigisemptyset — check for empty signal set

Synopsis

#include <signal.h>

int sigisemptyset (const sigset_t *set);

Description

The sigisemptyset function shall check for empty signal set referenced by set.

Return Value

The sigisemptyset function shall return a positive non-zero value if the signal set referenced by set is empty, or zero if this set is empty. On error, sigisemptyset shall return -1 and set errno to indicate the error.

Errors

EINVAL

set is a null pointer.
sigorset

Name

sigorset — build a new signal set by combining the two input sets using logical OR

Synopsis

```c
#include <signal.h>

int sigorset(sigset_t *set, const sigset_t *left, const sigset_t *right);
```

Description

The `sigorset` shall combine the two signal sets referenced by `left` and `right`, using a logical OR operation, and shall place the result in the location referenced by `set`. The resulting signal set shall contain only signals that are in either the set referenced by `left` or the set referenced by `right`.

Return Value

On success, `sigorset` shall return 0. Otherwise, `sigorset` shall return -1 and set `errno` to indicate the error.

Errors

- EINVAL
  
  One or more of `set`, `left`, or `right` was a null pointer.

See Also

- sigorset
sigreturn

Name

sigreturn — return from signal handler and cleanup stack frame

Synopsis

int sigreturn(unsigned long __unused);

Description

The sigreturn function is used by the system to cleanup after a signal handler has returned. This function is not in the source standard; it is only in the binary standard.

Return Value

sigreturn never returns.

stime

Name

stime — set time

Synopsis

#define _SVID_SOURCE
Chapter 1. Libraries

150

```c
#include <time.h>
int stime(time_t *t);
```

**Description**

If the process has appropriate privilege, the `stime` function shall set the system's idea of the time and date. Time, referenced by `t`, is measured in seconds from the epoch (defined in ISO POSIX (2003) as 00:00:00 UTC January 1, 1970).

**Return Value**

On success, `stime` shall return 0. Otherwise, `stime` shall return -1 and `errno` shall be set to indicate the error.

**Errors**

- **EPERM**
  - The process does not have appropriate privilege.
- **EINVAL**
  - `t` is a null pointer.

---

**stpcpy**

**Name**

`stpcpy` — copy a string returning a pointer to its end

**Synopsis**

```c
#include <string.h>
char *stpcpy(char * restrict dest, const char * restrict src);
```

**Description**

The `stpcpy` function shall copy the string pointed to by `src` (including the terminating '0' character) to the array pointed to by `dest`. The strings may not overlap, and the destination string `dest` shall be large enough to receive the copy.

**Return Value**

`stpcpy` returns a pointer to the end of the string `dest` (that is, the address of the terminating '0' character) rather than the beginning.

**Example**

This program uses `stpcpy` to concatenate `foo` and `bar` to produce `foob`, which it then prints.

```c
#include <string.h>
```
int main (void)
{
    char buffer[256];
    char *to = buffer;
    to = stpcpy (to, "foo");
    to = stpcpy (to, "bar");
    printf ("%s\n", buffer);
}

## stpncpy

**Name**

stpncpy — copy a fixed-size string, returning a pointer to its end

**Synopsis**

#include <string.h>

char *stpncpy (char * restrict dest, const char * restrict src, size_t n);

**Description**

The stpncpy function shall copy at most \( n \) characters from the string pointed to by \( src \), including the terminating \( \texttt{\0} \) character, to the array pointed to by \( dest \). Exactly \( n \) characters are written at \( dest \). If the length strlen (src) is smaller than \( n \), the remaining characters in \( dest \) are filled with \( \texttt{\0} \) characters. If the length strlen (src) is greater than or equal to \( n \), \( dest \) will not be \( \texttt{\0} \) terminated.

The strings may not overlap.

The programmer shall ensure that there is room for at least \( n \) characters at \( dest \).

**Return Value**

The stpncpy function shall return a pointer to the terminating NULL in \( dest \), or, if \( dest \) is not NULL-terminated, \( dest + n \).
strcasestr

Name
strcasestr — locate a substring ignoring case

Synopsis
#include <string.h>
char *strcasestr(const char *s1, const char *s2);

Description
The strcasestr shall behave as strstr, except that it shall ignore the case of both strings. The strcasestr function shall be locale aware; that is strcasestr shall behave as if both strings had been converted to lower case in the current locale before the comparison is performed.

Return Value
Upon successful completion, strcasestr shall return a pointer to the located string or a null pointer if the string is not found. If s2 points to a string with zero length, the function shall return s1.

strerror_r

Name
strerror_r — reentrant version of strerror

Synopsis
#include <string.h>
char *strerror_r(int errnum, char *buf, size_t buflen);

Description
strerror_r is a reentrant version of strerror. strerror_r returns a pointer to an error message corresponding to error number errnum. The returned pointer may point within the buffer buf (at most buflen bytes).

Note the optional use of the buffer, unlike the strerror_r found in ISO POSIX (2003), in which the message is always copied into the supplied buffer. The return types also differ.
strfry

Name
strfry — randomize a string

Synopsis
#include <string.h>
char *strfry(char *string);

Description
strfry randomizes the contents of string by using rand(3) to randomly swap characters in the string. The result is an anagram of string.

Return Value
strfry returns a pointer to the randomized string.

strndup

Name
strndup — return a malloc'd copy of at most the specified number of bytes of a string

Synopsis
#include <string.h>
char *strndup(const char *string, size_t n);

Description
The strndup function shall return a malloc'd copy of at most n bytes of string. The resultant string shall be terminated even if no NULL terminator appears before string+n.

Return Value
On success, strndup shall return a pointer to a newly allocated block of memory containing a copy of at most n bytes of string. Otherwise, strndup shall return NULL and set errno to indicate the error.

Errors
ENOMEM
Insufficient memory available.
strnlen

Name

strnlen — determine the length of a fixed-size string

Synopsis

```c
#include <string.h>
size_t strnlen(const char *s, size_t maxlen);
```

Description

strnlen returns the number of characters in the string `s`, not including the terminating `\0` character, but at most `maxlen`. In doing this, `strnlen` looks only at the first `maxlen` characters at `s` and never beyond `s + maxlen`.

Return Value

strnlen returns `strlen(s)`, if that is less than `maxlen`, or `maxlen` if there is no `\0` character among the first `maxlen` characters pointed to by `s`. 
### strftime

**Name**

`strftime` — parse a time string

**Description**

The `strftime` shall behave as specified in the ISO POSIX (2003) with differences as listed below.

**Number of leading zeroes may be limited**

The ISO POSIX (2003) specifies fields for which "leading zeros are permitted but not required"; however, applications shall not expect to be able to supply more leading zeroes for these fields than would be implied by the range of the field. Implementations may choose to either match an input with excess leading zeroes, or treat this as a non-matching input. For example, `%j` has a range of 001 to 366, so 0, 00, 000, 001, and 045 are acceptable inputs, but inputs such as 0000, 0366 and the like are not.

**Rationale**

`glibc` developers consider it appropriate behavior to forbid excess leading zeroes. When trying to parse a given input against several format strings, forbidding excess leading zeroes could be helpful. For example, if one matches 0011-12-26 against `%m-%d-%Y` and then against `%Y-%m-%d`, it seems useful for the first match to fail, as it would be perverse to parse that date as November 12, year 26. The second pattern parses it as December 26, year 11.

The ISO POSIX (2003) is not explicit that an unlimited number of leading zeroes are required, although it may imply this. The LSB explicitly allows implementations to have either behavior. Future versions of this standard may require implementations to forbid excess leading zeroes.

An Interpretation Request is currently pending against ISO POSIX (2003) for this matter.
**strsep**

**Name**

strsep — extract token from string

**Synopsis**

```
#include <string.h>
char *strsep(char **stringp, const char *delim);
```

**Description**

The `strsep` function shall find the first token in the string referenced by the pointer `stringp`, using the characters in `delim` as delimiters.

If `stringp` is NULL, `strsep` shall return NULL and do nothing else.

If `stringp` is non-NULL, `strsep` shall find the first token in the string referenced by `stringp`, where tokens are delimited by characters in the string `delim`. This token shall be terminated with a `\0` character by overwriting the delimiter, and `stringp` shall be updated to point past the token. In case no delimiter was found, the token is taken to be the entire string referenced by `stringp`, and the location referenced by `stringp` is made NULL.

**Return Value**

`strsep` shall return a pointer to the beginning of the token.

**Notes**

The `strsep` function was introduced as a replacement for `strtok`, since the latter cannot handle empty fields. However, `strtok` conforms to ISO C (1999) and to ISO POSIX (2003) and hence is more portable.

**See Also**

`strtok`, `strtok_r`.

**strsignal**

**Name**

strsignal — return string describing signal

**Synopsis**

```
#define _GNU_SOURCE
```

156
#include <string.h>

```
char *strsignal(int sig);
```

extern const char * const sys_siglist[];

**Description**

The `strsignal` function shall return a pointer to a string describing the signal number `sig`. The string can only be used until the next call to `strsignal`.

The array `sys_siglist` holds the signal description strings indexed by signal number. This array should not be accessed directly by applications.

**Return Value**

If `sig` is a valid signal number, `strsignal` shall return a pointer to the appropriate description string. Otherwise, `strsignal` shall return either a pointer to the string "unknown signal", or a null pointer.

Although the function is not declared as returning a pointer to a constant character string, applications shall not modify the returned string.

**strtoq**

**Name**

`strtoq` — convert string value to a long or quad_t integer

**Synopsis**

```
#include <sys/types.h>
#include <stdlib.h>
```
Chapter 1. Libraries

#include <limits.h>
quadt strtoq(const char *nptr, char **endptr, int base);

Description

strtoq converts the string *nptr to a quadt value. The conversion is done according to the given base, which shall be
between 2 and 36 inclusive, or be the special value 0.
nptr may begin with an arbitrary amount of white space (as determined by isspace(3)), followed by a single
optional + or - sign character. If base is 0 or 16, the string may then include a 0x prefix, and the number will be read
in base 16; otherwise, a 0 base is taken as 10 (decimal), unless the next character is 0, in which case it is taken as 8
(octal).
The remainder of the string is converted to a long value in the obvious manner, stopping at the first character which is
not a valid digit in the given base. (In bases above 10, the letter A in either upper or lower case represents 10, B
represents 11, and so forth, with Z representing 35.)

Return Value

strtoq returns the result of the conversion, unless the value would underflow or overflow. If an underflow occurs,
strtoq returns QUAD_MIN. If an overflow occurs, strtoq returns QUAD_MAX. In both cases, the global variable
errno is set to ERANGE.

Errors

ERANGE
The given string was out of range; the value converted has been clamped.

strtoq

Name

strtoq — convert a string to an uquad_t

Synopsis

#include <sys/types.h>
#include <stdlib.h>
Chapter 1. Libraries

```c
#include <limits.h>

uquadt strtouq(const char *nptr, char **endptr, int base);
```

Description

`strtouq` converts the string `nptr` to a uquadt value. The conversion is done according to the given base, which shall be between 2 and 36 inclusive, or be the special value 0.

`nptr` may begin with an arbitrary amount of white space (as determined by `isspace(3)`), followed by a single optional + or - sign character. If `base` is 0 or 16, the string may then include a 0x prefix, and the number will be read in base 16; otherwise, a 0 base is taken as 10 (decimal), unless the next character is 0, in which case it is taken as 8 (octal).

The remainder of the string is converted to an unsigned long value in the obvious manner, stopping at the end of the string or at the first character that does not produce a valid digit in the given base. (In bases above 10, the letter A in either upper or lower case represents 10, B represents 11, and so forth, with Z representing 35.)

Return Value

On success, `strtouq` returns either the result of the conversion or, if there was a leading minus sign, the negation of the result of the conversion, unless the original (non-negated) value would overflow. In the case of an overflow the function returns `UQUAD_MAX` and the global variable `errno` is set to `ERANGE`.

Errors

`ERANGE`

The given string was out of range; the value converted has been clamped.
**strverscmp**

**Name**

strverscmp — compare strings holding name and indices/version numbers

**Synopsis**

```c
#include <string.h>

int strverscmp(const char *s1, const char *s2);
```

**Description**

The `strverscmp` function shall compare two strings in a similar manner to `strcmp`. If `s1` and `s2` contain no digits, `strverscmp` shall behave as `strcmp`.

The strings are compared by scanning from left to right. If a digit or sequence of digits is encountered in both strings at the same position, the digit sequence is specially compared, as described below. If the digit sequences compared equal, the string comparison resumes in both `s1` and `s2` after the digit sequence.

Digit sequences are classified as either "integral" or "fractional". A fractional digit sequence begins with a '0'; otherwise the digit sequence shall be treated as an integral digit sequence.

If two integral digit sequences are encountered, they shall be compared as integers for equality. A fractional digit sequence shall always compare less than an integral digit sequence. If two fractional digit sequences are being compared, then if the common prefix contains only leading zeroes, the longer part shall compare less than the shorter; otherwise the comparison shall be strictly numeric.

**Examples**

Table 1-1. Examples

<table>
<thead>
<tr>
<th>Call</th>
<th>Return Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>strverscmp(&quot;no digit&quot;, &quot;no digit&quot;)</code></td>
<td>0 /* same behavior as strcmp */</td>
</tr>
<tr>
<td><code>strverscmp(&quot;item#99&quot;, &quot;item#100&quot;)</code></td>
<td>&lt; 0 /* same prefix, but 99 &lt; 100 */</td>
</tr>
<tr>
<td><code>strverscmp(&quot;alpha1&quot;, &quot;alpha001&quot;)</code></td>
<td>&gt; 0 /* fractional part inferior to integral */</td>
</tr>
<tr>
<td><code>strverscmp(&quot;part1_f012&quot;, &quot;part1_f01&quot;)</code></td>
<td>&gt; 0 /* two fractional parts */</td>
</tr>
<tr>
<td><code>strverscmp(&quot;foo.009&quot;, &quot;foo.0&quot;)</code></td>
<td>&lt; 0 /* two fractional parts but with leading zeroes only */</td>
</tr>
</tbody>
</table>
svc_register

Name
svc_register — Register Remote Procedure Call Interface

Synopsis
#include <rpc/rpc.h>
#include <rpc/svc.h>

void svc_register(SVCXPRT *xprt, u_long prognum, u_long versnum, void (*dispatch)(), u_long protocol);

Description
The svc_register function shall associate the program identified by prognum at version versnum with the service dispatch procedure, dispatch. If protocol is zero, the service is not registered with the portmap service. If protocol is non-zero, then a mapping of the triple [prognum, versnum, protocol] to xprt->xp_port is established with the local portmap service. The procedure dispatch has the following form:

int dispatch(struct svc_req * request, SVCXPRT * xprt);

Return Value
svc_register returns 1 if it succeeds, and zero otherwise.

svc_run

Name
svc_run — Waits for RPC requests to arrive and calls service procedure.

Synopsis
#include <rpc/svc.h>

void svc_run(void);

Description
The svc_run function shall wait for RPC requests to arrive, read and unpack each request, and dispatch it to the appropriate registered handler. Under normal conditions, svc_run shall not return; it shall only return if serious errors occur that prevent further processing.
**svc_sendreply**

**Name**

csvc_sendreply — called by RPC service’s dispatch routine

**Synopsis**

```c
svc_sendreply(SVCXPRT *xprt, xdrproc_t outproc, char out);
```

**Description**

Called by an RPC service’s dispatch routine to send the results of a remote procedure call. The parameter `xprt` is the request's associated transport handle; `outproc` is the XDR routine which is used to encode the results; and `out` is the address of the results. This routine returns one if it succeeds, zero otherwise.

**svctcp_create**

**Name**

svctcp_create — Creates a TCP/IP-based RPC service transport.

**Synopsis**

```c
#include <rpc/rpc.h>
SVCXPRT *svctcp_create(int sock, u_int send_buf_size, u_int recv_buf_size);
```

**Description**

svctcp_create creates a TCP/IP-based RPC service transport, to which it returns a pointer. The transport is associated with the socket `sock`, which may be `RPC_ANYSOCK`, in which case a new socket is created. If the socket is not bound to a local TCP port, then this routine binds it to an arbitrary port. Upon completion, `xprt->xp_sock` is the transport's socket descriptor, and `xprt->xp_port` is the transport's port number. Since TCP-based RPC uses buffered I/O, users may specify the size of buffers; values of zero choose suitable defaults.

**Return Value**

svctcp_create returns NULL if it fails, or a pointer to the RPC service transport otherwise.
svcudp_create

Name
svcudp_create — Creates a UDP-based RPC service transport.

Synopsis
SVCXPRT *
svcudp_create(int sock);

Description
This call is equivalent to svcudp_bufcreate (sock, SZ, SZ) for some default size SZ.

system

Name
system — execute a shell command

Synopsis
#include <stdlib.h>
int system(const char *string);

Description
The system function shall behave as described in ISO POSIX (2003).

Notes
The fact that system ignores interrupts is often not what a program wants. ISO POSIX (2003) describes some of the consequences; an additional consequence is that a program calling system from a loop cannot be reliably interrupted. Many programs will want to use the exec family of functions instead.

Do not use system from a program with suid or sgid privileges, because unexpected values for some environment variables might be used to subvert system integrity. Use the exec family of functions instead, but not execvp or execvp. system will not, in fact, work properly from programs with suid or sgid privileges on systems on which /bin/sh is bash version 2, since bash 2 drops privileges on startup. (Debian uses a modified bash which does not do this when invoked as sh.)

The check for the availability of /bin/sh is not actually performed; it is always assumed to be available. ISO C (1999) specifies the check, but ISO POSIX (2003) specifies that the return shall always be nonzero, since a system without the shell is not conforming, and it is this that is implemented.

It is possible for the shell command to return 127, so that code is not a sure indication that the execve call failed; check the global variable errno to make sure.
textdomain

Name
textdomain — set the current default message domain

Synopsis

#include <libintl.h>

char *textdomain(const char *domainname);

Description

The textdomain function shall set the current default message domain to domainname. Subsequent calls to gettext and ngettext use the default message domain.

If domainname is NULL, the default message domain shall not be altered.

If domainname is "", textdomain shall reset the default domain to the system default of "messages".

Return

On success, textdomain shall return the currently selected domain. Otherwise, a null pointer shall be returned, and errno set to indicate the error.

Errors

ENOMEM

Insufficient memory available.
**unlink**

**Name**
unlink — remove a directory entry

**Synopsis**
```c
int unlink(const char *path);
```

**Description**
unlink is as specified in ISO POSIX (2003), but with differences as listed below.

See also Additional behaviors: unlink/link on directory.

**May return EISDIR on directories**
If `path` specifies a directory, the implementation may return EISDIR instead of EPERM as specified by ISO POSIX (2003).

**Rationale**
The Linux kernel has deliberately chosen EISDIR for this case and does not expect to change (Al Viro, personal communication).

**vasprintf**

**Name**
vasprintf — write formatted output to a dynamically allocated string

**Synopsis**
```c
#include <stdarg.h>
```

Chapter 1. Libraries

#include <stdio.h>

int ** restrict ptr, const char * restrict format, va_list arg);

Description

The vasprintf function shall write formatted output to a dynamically allocated string, and store the address of that string in the location referenced by ptr. It shall behave as asprintf, except that instead of being called with a variable number of arguments, it is called with an argument list as defined by <stdarg.h>.

Return Value

Refer to fprintf.

Errors

Refer to fprintf.

vdprintf

Name

vdprintf — write formatted output to a file descriptor

Synopsis

#include <stdio.h>

int vdprintf(int fd, const char * restrict format, va_list arg);

Description

The vdprintf shall behave as vfprintf, except that the first argument is a file descriptor rather than a STDIO stream.

Return Value

Refer to fprintf.

Errors

Refer to fprintf.
**verrx**

**Name**

verrx — display formatted error message and exit

**Synopsis**

```c
#include <stdarg.h>
#include <err.h>
void verrx(int eval, const char *fmt, va_list args);
```

**Description**

The `verrx` shall behave as `errx` except that instead of being called with a variable number of arguments, it is called with an argument list as defined by `<stdarg.h>`. `verrx` does not return, but exits with the value of `eval`.

**Return Value**

None.

**Errors**

None.

---

**vsyslog**

**Name**

vsyslog — log to system log

**Synopsis**

```c
#include <stdarg.h>
#include <syslog.h>
void vsyslog(int priority, char *message, va_list arglist);
```

**Description**

The `vsyslog` function is identical to `syslog` as specified in ISO POSIX (2003), except that `arglist` (as defined by `<stdarg.h>`) replaces the variable number of arguments.
**wait3**

**Name**

`wait3` — wait for child process

**Description**

`wait3` is as specified in the SUSv2 but with differences as listed below.

- **WCONTINUED and WIFCONTINUED optional**
  - Implementations need not support the functionality of `WCONTINUED` or `WIFCONTINUED`.

**wait4**

**Name**

`wait4` — wait for process termination, BSD style

**Synopsis**

```c
#include <sys/types.h>
#include <sys/resource.h>
```
Chapter 1. Libraries

```c
#include <sys/wait.h>
pid_t wait4(pid_t pid, int *status, int options, (struct rusage *rusage));
```

**Description**

`wait4` suspends execution of the current process until a child (as specified by `pid`) has exited, or until a signal is delivered whose action is to terminate the current process or to call a signal handling function. If a child (as requested by `pid`) has already exited by the time of the call (a so-called "zombie" process), the function returns immediately. Any system resources used by the child are freed.

The value of `pid` can be one of:

- `< -1`
  - wait for any child process whose process group ID is equal to the absolute value of `pid`.
- `-1`
  - wait for any child process; this is equivalent to calling `wait3`.
- `0`
  - wait for any child process whose process group ID is equal to that of the calling process.
- `> 0`
  - wait for the child whose process ID is equal to the value of `pid`.

The value of options is a bitwise or of zero or more of the following constants:

- `WNOHANG`
  - return immediately if no child is there to be waited for.
- `WUNTRACED`
  - return for children that are stopped, and whose status has not been reported.

If status is not NULL, `wait4` stores status information in the location `status`. This status can be evaluated with the following macros:

- `WIFEXITED(status)`
  - is nonzero if the child exited normally.
- `WEXITSTATUS(status)`
  - evaluates to the least significant eight bits of the return code of the child that terminated, which may have been set as the argument to a call to `exit` or as the argument for a return statement in the main program. This macro can only be evaluated if `WIFEXITED` returned nonzero.
- `WIFSIGNALED(status)`
  - returns true if the child process exited because of a signal that was not caught.
- `WTERMSIG(status)`

These macros take the `status` value (an `int`) as an argument -- not a pointer to the value!
returns the number of the signal that caused the child process to terminate. This macro can only be evaluated if
WIFSIGNALED returned nonzero.

WIFSTOPPED(status)
returns true if the child process that caused the return is currently stopped; this is only possible if the call was
done using WUNTRACED.

WSTOPSIG(status)
returns the number of the signal that caused the child to stop. This macro can only be evaluated if WIFSTOPPED
returned nonzero.

If rusage is not NULL, the struct rusage (as defined in sys/resource.h) that it points to will be filled with
accounting information. (See getrusage(2) for details.

Return Value
On success, the process ID of the child that exited is returned. On error, -1 is returned (in particular, when no
unwaited-for child processes of the specified kind exist), or 0 if WNOHANG was used and no child was available yet. In
the latter two cases, the global variable errno is set appropriately.

Errors
ECHILD
No unwaited-for child process as specified does exist.

ERESTARTSYS
A WNOHANG was not set and an unblocked signal or a SIGCHILD was caught. This error is returned by the system
call. The library interface is not allowed to return ERESTARTSYS, but will return EINTR.

waitpid
Name
waitpid — wait for child process

Description
waitpid is as specified in ISO POSIX (2003), but with differences as listed below.

Need not support WCONTINUED or WIFCONTINUED
Implementations need not support the functionality of WCONTINUED or WIFCONTINUED.
**warn**

**Name**

warn — formatted error messages

**Synopsis**

```c
#include <err.h>
void warn(const char *fmt ...);
```

**Description**

The `warn` function shall display a formatted error message on the standard error stream. The output shall consist of the last component of the program name, a colon character, and a space character. If `fmt` is non-NULL, it shall be used as a format string for the `printf` family of functions, and the formatted message, a colon character, and a space are written to `stderr`. Finally, the error message string affiliated with the current value of the global variable `errno` shall be written to `stderr`, followed by a newline character.

**Return Value**

None.

**Errors**

None.
warnx

Name

4232 warnx — formatted error messages

Synopsis

4233 #include <err.h>
4234 void warnx(const char *fmt ...);

Description

4235 The warnx function shall display a formatted error message on the standard error stream. The last component of the
4236 program name, a colon character, and a space shall be output. If fmt is non-NULL, it shall be used as the format string
4237 for the printf family of functions, and the formatted error message, a colon character, and a space shall be output.
4238 The output shall be followed by a newline character.

Return Value

4239 None.

Errors

4240 None.
**wcpcpy**

**Name**

wcpcpy — copy a wide character string, returning a pointer to its end

**Synopsis**

```c
#include <wchar.h>

wchar_t *wcpcpy(wchar_t *dest, const wchar_t *src);
```

**Description**

wcpcpy is the wide-character equivalent of stpcpy. It copies the wide character string `src`, including the terminating `L\0` character, to the array `dest`. The strings may not overlap. The programmer shall ensure that there is room for at least `wcslen(src)+1` wide characters at `dest`.

**Return Value**

wcpcpy returns a pointer to the end of the wide-character string `dest`, that is, a pointer to the terminating `L\0` character.

**wcpncpy**

**Name**

wcpncpy — copy a fixed-size string of wide characters, returning a pointer to its end

**Synopsis**

```c
#include <wchar.h>

wchar_t *wcpncpy(wchar_t *dest, const wchar_t *src, size_t n);
```

**Description**

wcpncpy is the wide-character equivalent of stpncpy. It copies at most `n` wide characters from the wide-character string `src`, including the terminating `L\0` character, to the array `dest`. Exactly `n` wide characters are written at `dest`. The strings may not overlap. The programmer shall ensure that there is room for at least `n` wide characters at `dest`.

**Return Value**

wcpncpy returns a pointer to the wide character one past the last non-null wide character written.
**wcscasecmp**

**Name**
wcscasecmp — compare two wide-character strings, ignoring case

**Synopsis**
```c
#include <wchar.h>

int wcscasecmp(const wchar_t *s1, const wchar_t *s2);
```

**Description**
wscasecmp is the wide-character equivalent of strcasecmp. It compares the wide-character string s1 and the wide-character string s2, ignoring case differences (towupper, towlower).

**Return Value**
wscasecmp returns 0 if the wide-character strings s1 and s2 are equal except for case distinctions. It returns a positive integer if s1 is greater than s2, ignoring case. It returns a negative integer if s1 is smaller than s2, ignoring case.

**Notes**
The behavior of wcscasecmp depends upon the LC_CTYPE category of the current locale.

**wcsdup**

**Name**
wcsdup — duplicate a wide-character string

**Synopsis**
```c
#include <wchar.h>

wchar_t *wcsdup(const wchar_t *s);
```

**Description**
wcsdup is the wide-character equivalent of strdup. It allocates and returns a new wide-character string whose initial contents is a duplicate of the wide-character string s.

Memory for the new wide-character string is obtained with malloc(3), and can be freed with free(3).

**Return Value**
wcsdup returns a pointer to the new wide-character string, or NULL if sufficient memory was not available.
**wcsncasecmp**

**Name**

wcsncasecmp — compare two fixed-size wide-character strings, ignoring case

**Synopsis**

```c
#include <wchar.h>

int wcsncasecmp(const wchar_t *s1, const wchar_t *s2, size_t n);
```

**Description**

wcsncasecmp is the wide-character equivalent of strncasecmp. It compares the wide-character string `s1` and the wide-character string `s2`, but at most `n` wide characters from each string, ignoring case differences (towupper, towlower).

**Return Value**

wcsncasecmp returns 0 if the wide-character strings `s1` and `s2`, truncated to at most length `n`, are equal except for case distinctions. It returns a positive integer if truncated `s1` is greater than truncated `s2`, ignoring case. It returns a negative integer if truncated `s1` is smaller than truncated `s2`, ignoring case.

**Notes**

The behavior of wcsncasecmp depends upon the LC_CTYPE category of the current locale.
### wcsnlen

**Name**

wcsnlen — determine the length of a fixed-size wide-character string

**Synopsis**

```c
#include <wchar.h>

size_t wcsnlen(const wchar_t *s, size_t maxlen);
```

**Description**

wcsnlen is the wide-character equivalent of strnlen. It returns the number of wide-characters in the string \( s \), not including the terminating \( L\'\0' \) character, but at most \( maxlen \). In doing this, wcsnlen looks only at the first \( maxlen \) wide-characters at \( s \) and never beyond \( s + maxlen \).

**Return Value**

wcsnlen returns \( wcslen(s) \) if that is less than \( maxlen \), or \( maxlen \) if there is no \( L\'\0' \) character among the first \( maxlen \) wide characters pointed to by \( s \).

**Notes**

The behavior of wcsncasecmp depends on the LC_CTYPE category of the current locale.
**wcsnrtombs**

**Name**

wcsnrtombs — convert a wide character string to a multi-byte string

**Synopsis**

```c
#include <wchar.h>

size_t wcsnrtombs(char *dest, const wchar_t **src, size_t nwc, size_t len, mbstate_t *ps);
```

**Description**

wcsnrtombs is like wcsrtombs, except that the number of wide characters to be converted, starting at src, is limited to nwc.

If dest is not a NULL pointer, wcsnrtombs converts at most nwc wide characters from the wide-character string src to a multibyte string starting at dest. At most len bytes are written to dest. The state ps is updated.

The conversion is effectively performed by repeatedly calling:

```c
wcrtomb(dest, *src, ps)
```

as long as this call succeeds, and then incrementing dest by the number of bytes written and src by 1.

The conversion can stop for three reasons:

- A wide character has been encountered that cannot be represented as a multibyte sequence (according to the current locale). In this case src is left pointing to the invalid wide character, (size_t)(-1) is returned, and errno is set to EILSEQ.
- nws wide characters have been converted without encountering a L'\0', or the length limit forces a stop. In this case, src is left pointing to the next wide character to be converted, and the number bytes written to dest is returned.
- The wide-character string has been completely converted, including the terminating L'\0' (which has the side effect of bringing back ps to the initial state). In this case, src is set to NULL, and the number of bytes written to dest, excluding the terminating L'\0' byte, is returned.

If dest is NULL, len is ignored, and the conversion proceeds as above, except that the converted bytes are not written out to memory, and that no destination length limit exists.

In both of the above cases, if ps is a NULL pointer, a static anonymous state only known to wcsnrtombs is used instead.

The programmer shall ensure that there is room for at least len bytes at dest.

**Return Value**

wcsnrtombs returns the number of bytes that make up the converted part of multibyte sequence, not including the terminating L'\0' byte. If a wide character was encountered which could not be converted, (size_t)(-1) is returned, and the global variable errno set to EILSEQ.

**Notes**
The behavior of wcsnrtombs depends on the LC_CTYPE category of the current locale.

Passing NULL as ps is not multi-thread safe.

**wcstoq**

**Name**

**wcstoq** — convert wide string to long long int representation

**Synopsis**

```c
#include <wchar.h>
long long int wcstoq(const wchar_t * restrict nptr, wchar_t ** restrict endptr, int base);
```

**Description**

The *wcstoq* function shall convert the initial portion of the wide string *nptr* to long long int representation. It is identical to *wcstoll*.

**Return Value**

Refer to *wcstoll*.

**Errors**

Refer to *wcstoll*. 
wcstouq

Name

wcstouq — convert wide string to unsigned long long int representation

Synopsis

#include <wchar.h>

unsigned long long int wcstouq(const wchar_t * restrict nptr, wchar_t ** restrict endptr, int base);

Description

The wcstouq function shall convert the initial portion of the wide string nptr to unsigned long long int representation. It is identical to wcstoull.

Return Value

Refer to wcstoull.

Errors

Refer to wcstoull.

xdr_u_int

Name

xdr_u_int — library routines for external data representation

Synopsis

int xdr_u_int(XDR * xdrs, unsigned int * up);

Description

xdr_u_int is a filter primitive that translates between C unsigned integers and their external representations.

Return Value

On success, 1 is returned. On error, 0 is returned.

1.5. Interfaces for libm

Table 1-29 defines the library name and shared object name for the libm library
Chapter 1. Libraries

Table 1-29. libm Definition

<table>
<thead>
<tr>
<th>Library:</th>
<th>libm</th>
</tr>
</thead>
</table>

| SONAME: | See archLSB. |

The behavior of the interfaces in this library is specified by the following specifications:
ISO C (1999)
SUSv2

1.5.1. Math

1.5.1.1. Interfaces for Math

An LSB conforming implementation shall provide the generic functions for Math specified in Table 1-30, with the full functionality as described in the referenced underlying specification.

Table 1-30. libm - Math Function Interfaces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Library Function</td>
<td>Library Function</td>
<td>Library Function</td>
<td>Library Function</td>
<td>Library Function</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------</td>
<td>----------------------</td>
<td>----------------------</td>
<td>----------------------</td>
</tr>
</tbody>
</table>
Chapter 1. Libraries

| --- | --- | --- | --- | --- |

Referenced Specification(s)

[3]. SUSv2

An LSB conforming implementation shall provide the generic data interfaces for Math specified in Table 1-31, with the full functionality as described in the referenced underlying specification.

Table 1-31. libm - Math Data Interfaces

<table>
<thead>
<tr>
<th>signgam [1]</th>
</tr>
</thead>
</table>

Referenced Specification(s)


1.6. Data Definitions for libm

This section defines global identifiers and their values that are associated with interfaces contained in libm. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content.

These definitions are intended to supplement those provided in the referenced underlying specifications.

This specification uses ISO/IEC 9899 C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

1.6.1. complex.h

#define complex _Complex

1.6.2. math.h

#define DOMAIN 1
#define SING 2
Chapter 1. Libraries

1.7. Interfaces for libpthread

Table 1-32 defines the library name and shared object name for the libpthread library

Table 1-32. libpthread Definition

<table>
<thead>
<tr>
<th>Library:</th>
<th>libpthread</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libpthread.so.0</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications:

Large File Support

this specification
1.7.1. Realtime Threads

1.7.1.1. Interfaces for Realtime Threads

No external functions are defined for libpthread - Realtime Threads

1.7.2. Advanced Realtime Threads

1.7.2.1. Interfaces for Advanced Realtime Threads

No external functions are defined for libpthread - Advanced Realtime Threads

1.7.3. Posix Threads

1.7.3.1. Interfaces for Posix Threads

An LSB conforming implementation shall provide the generic functions for Posix Threads specified in Table 1-33, with the full functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>libpthread - Posix Threads Function Interfaces</th>
</tr>
</thead>
</table>
Chapter 1. Libraries

1.8. Data Definitions for libpthread

This section defines global identifiers and their values that are associated with interfaces contained in libpthread.

These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content.

These definitions are intended to supplement those provided in the referenced underlying specifications.

This specification uses ISO/IEC 9899 C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

1.8.1. pthread.h

```c
#define PTHREAD_MUTEX_DEFAULT 1
#define PTHREAD_MUTEX_NORMAL 1
#define PTHREAD_MUTEX_TIMED_NP 1
#define PTHREAD_MUTEX_RECURSIVE 2
#define PTHREAD_RWLOCK_DEFAULT_NP 2
#define PTHREAD_MUTEX_ERRORCHECK 3
#define pthread_cleanup_pop(execute) _pthread_cleanup_pop(& _buffer,(execute));
#define __LOCK_INITIALIZER { 0, 0 }
#define PTHREAD_RWLOCK_INITIALIZER { __LOCK_INITIALIZER, 0, NULL, NULL, NULL,PTHREAD_RWLOCK_DEFAULT_NP, PTHREAD_PROCESS_PRIVATE }
#define PTHREAD_MUTEX_INITIALIZER {0,0,0,PTHREAD_MUTEX_TIMED_NP,__LOCK_INITIALIZER}
```
```c
#define pthread_cleanup_push(routine, arg)       {struct _pthread_cleanup_buffer
    __buffer; pthread_cleanup_push(& __buffer, (routine), (arg));
#define PTHREAD_COND_INITIALIZER        {__LOCK_INITIALIZER,0}

struct _pthread_cleanup_buffer
{
    void (*__routine) (void *);
    void *__arg;
    int __canceltype;
    struct _pthread_cleanup_buffer *__prev;
};
typedef unsigned int pthread_key_t;
typedef int pthread_once_t;
typedef long long __pthread_cond_align_t;
typedef unsigned long pthread_t;
struct _pthread_fastlock
{
    long __status;
    int __spinlock;
};
typedef struct _pthread_descr_struct *_pthread_descr;
typedef struct
{
    int __m_reserved;
    int __m_count;
    _pthread_descr __m_owner;
    int __m_kind;
    struct _pthread_fastlock __m_lock;
};
pthread_mutex_t;
typedef struct
{
    int __mutexkind;
};
pthread_mutexattr_t;
typedef struct
{
    int __detachstate;
    int __schedpolicy;
    struct sched_param __schedparam;
    int __inheritsched;
    int __scope;
    size_t __guardsize;
    int __stackaddr_set;
    void *__stackaddr;
    unsigned long __stacksize;
};
```

Chapter 1. Libraries
Chapter 1. Libraries

typedef struct
{
    struct _pthread_fastlock __c_lock;
    _pthread_descr __c_waiting;
    char __padding[48 - sizeof (struct _pthread_fastlock) -
                  sizeof (_pthread_descr) - sizeof (__pthread_cond_align_t)];
    __pthread_cond_align_t __align;
}
pthread_cond_t;

typedef struct
{
    int __dummy;
}
pthread_condattr_t;

typedef struct _pthread_rwlock_t
{
    struct _pthread_fastlock __rw_lock;
    int __rw_readers;
    _pthread_descr __rw_writer;
    _pthread_descr __rw_read_waiting;
    _pthread_descr __rw_write_waiting;
    int __rw_kind;
    int __rw_pshared;
}
pthread_rwlock_t;

typedef struct
{
    int __lockkind;
    int __pshared;
}
pthread_rwlockattr_t;

#define PTHREAD_CREATE_JOINABLE 0
#define PTHREAD_INHERIT_SCHED 0
#define PTHREAD_ONCE_INIT 0
#define PTHREAD_PROCESS_PRIVATE 0
#define PTHREAD_CREATE_DETACHED 1
#define PTHREAD_EXPLICIT_SCHED 1
#define PTHREAD_PROCESS_SHARED 1
#define PTHREAD_CANCELED ((void*)-1)
#define PTHREAD_CANCEL_DEFERRED 0
#define PTHREAD_CANCEL_ENABLE 0
#define PTHREAD_CANCEL_ASYNCHRONOUS 1
#define PTHREAD_CANCEL_DISABLE 1

1.8.2. semaphore.h
typedef struct {
  struct _pthread_fastlock __sem_lock;
  int __sem_value;
  _pthread_descr __sem_waiting;
} sem_t;

#define SEM_FAILED      ((sem_t*)0)

#define SEM_VALUE_MAX   ((int)((~0u)>>1))

1.9. Interface Definitions for libpthread

The following interfaces are included in libpthread and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed above for libpthread shall behave as described in the referenced base document.

_pthread_cleanup_pop

Name

_pthread_cleanup_pop — establish cancellation handlers

Synopsis

#include <pthread.h>

void _pthread_cleanup_pop(struct _pthread_cleanup_buffer *, int);

Description

The _pthread_cleanup_pop function provides an implementation of the pthread_cleanup_pop macro described in ISO POSIX (2003).

The _pthread_cleanup_pop function is not in the source standard; it is only in the binary standard.
_pthread_cleanup_push

Name
(pthread_cleanup_push — establish cancellation handlers)

Synopsis

#include <pthread.h>

void _pthread_cleanup_push(struct _pthread_cleanup_buffer *, void (*)(void *), void *);

Description

The _pthread_cleanup_push function provides an implementation of the pthread_cleanup_push macro described in ISO POSIX (2003).

The _pthread_cleanup_push function is not in the source standard; it is only in the binary standard.

1.10. Interfaces for libgcc_s

Table 1-34 defines the library name and shared object name for the libgcc_s library

<table>
<thead>
<tr>
<th>Library:</th>
<th>libgcc_s</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libgcc_s.so.1</td>
</tr>
</tbody>
</table>

1.10.1. Unwind Library

1.10.1.1. Interfaces for Unwind Library

No external functions are defined for libgcc_s - Unwind Library

1.11. Data Definitions for libgcc_s

This section defines global identifiers and their values that are associated with interfaces contained in libgcc_s. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content.

These definitions are intended to supplement those provided in the referenced underlying specifications.

This specification uses ISO/IEC 9899 C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

1.11.1. unwind.h

struct dwarf_eh_base
Chapter 1. Libraries

```c
{  
   void *tbase;
   void *dbase;
   void *func;
 }

;  
struct _Unwind_Context;

typedef unsigned int _Unwind_Ptr;
typedef unsigned int _Unwind_Word;

typedef enum
{
   _URC_NO_REASON, _URC_FOREIGN_EXCEPTION_CAUGHT = 1, _URC_FATAL_PHASE2_ERROR =
   2, _URC_FATAL_PHASE1_ERROR = 3, _URC_NORMAL_STOP = 4, _URC_END_OF_STACK =
   5, _URC_HANDLER_FOUND = 6, _URC_INSTALL_CONTEXT =
   7, _URC_CONTINUE_UNWIND = 8
}

struct _Unwind_Reason_Code;

struct _Unwind_Exception
{
   _Unwind_Exception_Class;
   _Unwind_Exception_Cleanup_Fn;
   _Unwind_Word;
   _Unwind_Word;
}

#define _UA_SEARCH_PHASE        1
#define _UA_END_OF_STACK        16
#define _UA_CLEANUP_PHASE       2
#define _UA_HANDLER_FRAME       4
#define _UA_FORCE_UNWIND        8
```

1.12. Interfaces for libdl

Table 1-35 defines the library name and shared object name for the libdl library

<table>
<thead>
<tr>
<th>Library:</th>
<th>libdl</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libdl.so.2</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications:

this specification
1.12.1. Dynamic Loader

1.12.1.1. Interfaces for Dynamic Loader

An LSB conforming implementation shall provide the generic functions for Dynamic Loader specified in Table 1-36, with the full functionality as described in the referenced underlying specification.

|------------|-------------|-------------|------------|-------------|

Referenced Specification(s)

[1]. this specification

1.13. Data Definitions for libdl

This section defines global identifiers and their values that are associated with interfaces contained in libdl. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content.

These definitions are intended to supplement those provided in the referenced underlying specifications.

This specification uses ISO/IEC 9899 C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

1.13.1. dlfcn.h

```c
#define RTLD_NEXT       ((void *) -11)
#define RTLD_LOCAL      0
#define RTLD_LAZY       0x00001
#define RTLD_NOW        0x00002
#define RTLD_GLOBAL     0x00100

typedef struct
{
    char *dli_fname;
    void *dli_fbase;
    char *dli_sname;
    void *dli_saddr;
} Dl_info;
```

1.14. Interface Definitions for libdl

The following interfaces are included in libdl and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.
Other interfaces listed above for libdl shall behave as described in the referenced base document.

**dladdr**

**Name**

dladdr — find the shared object containing a given address

**Synopsis**

```
#include <dlfcn.h>

typedef struct {
    const char  *dli_fname;
    void        *dli_fbase;
    const char  *dli_sname;
    void        *dli_saddr;
} dladdr_t;
```
**Description**

The `dladdr` function shall query the dynamic linker for information about the shared object containing the address `addr`. The information shall be returned in the user supplied data structure referenced by `dlip`.

The structure shall contain at least the following members:

- `dli_fname`
  - The pathname of the shared object containing the address

- `dli_fbase`
  - The base address at which the shared object is mapped into the address space of the calling process.

- `dli_sname`
  - The name of the nearest runtime symbol with value less than or equal to `addr`. Where possible, the symbol name shall be returned as it would appear in C source code.

  If no symbol with a suitable value is found, both this field and `dli_saddr` shall be set to `NULL`.

- `dli_saddr`
  - The address of the symbol returned in `dli_sname`.

The behavior of `dladdr` is only specified in dynamically linked programs.

**Return Value**

On success, `dladdr` shall return non-zero, and the structure referenced by `dlip` shall be filled in as described.

Otherwise, `dladdr` shall return zero, and the cause of the error can be fetched with `dlerr`.

**Errors**

See `dlerr`.

**Environment**

- `LD_LIBRARY_PATH` directory search-path for object files
**dlopen**

**Name**

dlopen — open dynamic object

**Synopsis**

```c
#include <dlfcn.h>

void * dlopen(const char *filename, int flag);
```

**Description**

dlopen shall behave as specified in ISO POSIX (2003), but with additional behaviors listed below.

If the file argument does not contain a slash character, then the system shall look for a library of that name in at least the following directories, and use the first one which is found:

- The directories specified by the DT_RPATH dynamic entry.
- The directories specified in the LD_LIBRARY_PATH environment variable (which is a colon separated list of pathnames). This step shall be skipped for setuid and setgid executables.
- A set of directories sufficient to contain the libraries specified in this standard.

Traditionally, /lib and /usr/lib. This case would also cover cases in which the system used the mechanism of /etc/ld.so.conf and /etc/ld.so.cache to provide access.

Example: An application which is not linked against libm may choose to dlopen libm.

**dlsym**

**Name**

dlsym — obtain the address of a symbol from a dlopen object

**Description**

dlsym is as specified in the ISO POSIX (2003), but with differences as listed below.

The special purpose value for handle RTLD_NEXT

The value RTLD_NEXT, which is reserved for future use shall be available, with the behavior as described in ISO POSIX (2003).

**1.15. Interfaces for libcrypt**

Table 1-37 defines the library name and shared object name for the libcrypt library
Chapter 1. Libraries

Table 1-37. libcrypt Definition

<table>
<thead>
<tr>
<th>Library:</th>
<th>libcrypt</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libcrypt.so.1</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications:


1.15.1. Encryption

1.15.1.1. Interfaces for Encryption

An LSB conforming implementation shall provide the generic functions for Encryption specified in Table 1-38, with the full functionality as described in the referenced underlying specification.

Table 1-38. libcrypt - Encryption Function Interfaces


Referenced Specification(s)


1.16. Interfaces for libpam

Table 1-39 defines the library name and shared object name for the libpam library

Table 1-39. libpam Definition

<table>
<thead>
<tr>
<th>Library:</th>
<th>libpam</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libpam.so.0</td>
</tr>
</tbody>
</table>

A single service name, other, shall always be present. The behavior of this service shall be determined by the system administrator. Additional service names may also exist. ¹

The behavior of the interfaces in this library is specified by the following specifications:

1.16.1. Pluggable Authentication API

1.16.1.1. Interfaces for Pluggable Authentication API

An LSB conforming implementation shall provide the generic functions for Pluggable Authentication API specified in Table 1-40, with the full functionality as described in the referenced underlying specification.

Table 1-40. libpam - Pluggable Authentication API Function Interfaces

1.17. Data Definitions for libpam

This section defines global identifiers and their values that are associated with interfaces contained in libpam. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content.

These definitions are intended to supplement those provided in the referenced underlying specifications.

This specification uses ISO/IEC 9899 C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

1.17.1. security/pam_appl.h

```c
typedef struct pam_handle pam_handle_t;
struct pam_message
{
    int msg_style;
    const char *msg;
}
;
struct pam_response
{
    char *resp;
    int resp_retcode;
}
;
struct pam_conv
{
    int (*conv)(int num_msg, const struct pam_message * *msg,
                struct pam_response * *resp, void *appdata_ptr);

    void *appdata_ptr;
}

#define PAM_PROMPT_ECHO_OFF     1
#define PAM_PROMPT_ECHO_ON      2
#define PAM_ERROR_MSG   3
#define PAM_TEXT_INFO   4
```
1.18. Interface Definitions for libpam

The following interfaces are included in libpam and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed above for libpam shall behave as described in the referenced base document.
pam_acct_mgmt

Name

pam_acct_mgmt — establish the status of a user's account

Synopsis

```c
#include <security/pam_appl.h>

int pam_acct_mgmt(pam_handle_t *pamh, int flags);
```

Description

pam_acct_mgmt establishes the account's usability and the user's accessibility to the system. It is typically called after the user has been authenticated.

flags may be specified as any valid flag (namely, one of those applicable to the flags argument of pam_authenticate). Additionally, the value of flags may be logically or'd with PAM_SILENT.

Return Value

- **PAM_SUCCESS**
  - Success.
- **PAM_NEW_AUTHTOK_REQD**
  - User is valid, but user's authentication token has expired. The correct response to this return-value is to require that the user satisfy the pam_chauthtok function before obtaining service. It may not be possible for an application to do this. In such a case, the user should be denied access until the account password is updated.
- **PAM_ACCT_EXPIRED**
  - User is no longer permitted access to the system.
- **PAM_AUTH_ERR**
  - Authentication error.
- **PAM_PERM_DENIED**
  - User is not permitted to gain access at this time.
- **PAM_USER_UNKNOWN**
  - User is not known to a module's account management component.

Errors

May be translated to text with pam_strerror.
**pam_authenticate**

**Name**

pam_authenticate — authenticate the user

**Synopsis**

```c
#include <security/pam_appl.h>
int pam_authenticate(pam_handle_t *pamh, int flags);
```

**Description**

pam_authenticate serves as an interface to the authentication mechanisms of the loaded modules. 

`flags` is an optional parameter that may be specified by the following value:

- **PAM_DISALLOW_NULL_AUTHTOK**
  - Instruct the authentication modules to return PAM_AUTH_ERR if the user does not have a registered authorization token.
  - Additionally, the value of `flags` may be logically or'd with PAM_SILENT.
  - The process may need to be privileged in order to successfully call this function.

**Return Value**

- **PAM_SUCCESS**
  - Success.
- **PAM_AUTH_ERR**
  - User was not authenticated or process did not have sufficient privileges to perform authentication.
- **PAM_CRED_INSUFFICIENT**
  - Application does not have sufficient credentials to authenticate the user.
- **PAM_AUTHINFO_UNAVAIL**
  - Modules were not able to access the authentication information. This might be due to a network or hardware failure, etc.
- **PAM_USER_UNKNOWN**
  - Supplied username is not known to the authentication service.
- **PAM_MAXTRIES**
  - One or more authentication modules has reached its limit of tries authenticating the user. Do not try again.
- **PAM_ABORT**
  - One or more authentication modules failed to load.
Errors

4862  May be translated to text with `pam_strerror`. 
pam_chauthtok

Name

pam_chauthtok — change the authentication token for a given user

Synopsis

```c
#include <security/pam_appl.h>

int pam_chauthtok(pam_handle_t *pamh, const int flags);
```

Description

pam_chauthtok is used to change the authentication token for a given user as indicated by the state associated with the handle `pamh`.

`flags` is an optional parameter that may be specified by the following value:

- `PAM_CHANGE_EXPIRED_AUTHTOK`
  - User's authentication token should only be changed if it has expired.
  - Additionally, the value of `flags` may be logically or'd with `PAM_SILENT`.

RETURN VALUE

- `PAM_SUCCESS`
  - Success.
- `PAM_AUTHTOK_ERR`
  - A module was unable to obtain the new authentication token.
- `PAM_AUTHTOK_RECOVER_ERR`
  - A module was unable to obtain the old authentication token.
- `PAM_AUTHTOK_LOCK_BUSY`
  - One or more modules were unable to change the authentication token since it is currently locked.
- `PAM_AUTHTOK_DISABLE_AGING`
  - Authentication token aging has been disabled for at least one of the modules.
- `PAM_PERM_DENIED`
  - Permission denied.
- `PAM_TRY_AGAIN`
  - Not all modules were in a position to update the authentication token(s). In such a case, none of the user's authentication tokens are updated.
Chapter 1. Libraries

4877 PAM_USER_UNKNOWN
4878 User is not known to the authentication token changing service.

ERRORS
4889 May be translated to text with pam_strerror.

pam_close_session

Name
4890 pam_close_session — indicate that an authenticated session has ended

Synopsis
4891 #include <security/pam_appl.h>
4892 int pam_close_session(pam_handle_t *pamh, int flags);

Description
4893 pam_close_session is used to indicate that an authenticated session has ended. It is used to inform the module that
4894 the user is exiting a session. It should be possible for the PAM library to open a session and close the same session
4895 from different applications.
4896 flags may have the value PAM_SILENT to indicate that no output should be generated as a result of this function call.

Return Value
4897 PAM_SUCCESS
4898 Success.
4899 PAM_SESSION_ERR
4900 One of the required loaded modules was unable to close a session for the user.

Errors
4901 May be translated to text with pam_strerror.
**pam_end**

**Name**

pam_end — terminate the use of the PAM library

**Synopsis**

```c
#include <security/pam_appl.h>

int pam_end(pam_handle_t *pamh, int pam_status);
```

**Description**

pam_end terminates use of the PAM library. On success, the contents of *pamh* are no longer valid, and all memory associated with it is invalid.

Normally, *pam_status* is passed the value PAM_SUCCESS, but in the event of an unsuccessful service application, the appropriate PAM error return value should be used.

**Return Value**

- **PAM_SUCCESS**
  - Success.

**Errors**

May be translated to text with `pam_strerror`. 
**pam_fail_delay**

**Name**

pam_fail_delay — specify delay time to use on authentication error

**Synopsis**

```c
#include <security/pam_appl.h>
int pam_fail_delay(pam_handle_t *pamh, unsigned int micro_sec);
```

**Description**

pam_fail_delay specifies the minimum delay for the PAM library to use when an authentication error occurs. The actual delay can vary by as much at 25%. If this function is called multiple times, the longest time specified by any of the call will be used.

The delay is invoked if an authentication error occurs during the `pam_authenticate` or `pam_chauthtok` function calls.

Independent of the success of `pam_authenticate` or `pam_chauthtok`, the delay time is reset to its default value of 0 when the PAM library returns control to the application from these two functions.

**Return Value**

PAM_SUCCESS

Success.

**Errors**

May be translated to text with `pam_strerror`.
pam_get_item

Name

pam_get_item — obtain the value of the indicated item.

Synopsis

```c
#include <security/pam_appl.h>
int pam_get_item(const pam_handle_t *pamh, int item_type, const void **item);
```

Description

pam_get_item obtains the value of the indicated item_type. The possible values of item_type are the same as listed for pam_set_item.

On success, item contains a pointer to the value of the corresponding item. Note that this is a pointer to the actual data and should not be free’ed or over-written.

Return Value

- PAM_SUCCESS
  - Success.
- PAM_PERM_DENIED
  - Application passed a NULL pointer for item.
- PAM_BAD_ITEM
  - Application attempted to get an undefined item.

Errors

May be translated to text with pam_strerror.
**pam_getenvlist**

**Name**

*pam_getenvlist* — returns a pointer to the complete PAM environment.

**Synopsis**

```c
#include <security/pam_appl.h>
char * const *pam_getenvlist(pam_handle_t *pamh);
```

**Description**

*pam_getenvlist* returns a pointer to the complete PAM environment. This pointer points to an array of pointers to NULL-terminated strings and must be terminated by a NULL pointer. Each string has the form "name=value". The PAM library module allocates memory for the returned value and the associated strings. The calling application is responsible for freeing this memory.

**Return Value**

*pam_getenvlist* returns an array of string pointers containing the PAM environment. On error, NULL is returned.
Chapter 1. Libraries

**pam_open_session**

**Name**

pam_open_session — used to indicate that an authenticated session has been initiated

**Synopsis**

```c
#include <security/pam_appl.h>
int pam_open_session(pam_handle_t *pamh, int flags);
```

**Description**

pam_handle_t is used to indicate that an authenticated session has begun. It is used to inform the module that the user is currently in a session. It should be possible for the PAM library to open a session and close the same session from different applications.

flags may have the value PAM_SILENT to indicate that no output be generated as a result of this function call.

**Return Value**

- **PAM_SUCCESS**
  - Success.
- **PAM_SESSION_ERR**
  - One of the loaded modules was unable to open a session for the user.

**ERRORS**

May be translated to text with pam_strerror.
pam_set_item

Name
4959 pam_set_item — (re)set the value of an item.

Synopsis
4960 #include <security/pam_appl.h>
4961 int pam_set_item(pam_handle_t *pamh, int item_type, const void *item);

Description
4962 pam_set_item (re)sets the value of one of the following item_types:
4963 PAM_SERVICE
4964 service name
4965 PAM_USER
4966 user name
4967 PAM_TTY
4968 terminal name
4969 The value for a device file should include the /dev/ prefix. The value for graphical, X-based, applications should be the $DISPLAY variable.
4970 PAM_RHOST
4971 remote host name
4972 PAM_CONV
4973 conversation structure
4974 PAM_RUSER
4975 remote user name
4976 PAM_USER_PROMPT
4977 string to be used when prompting for a user's name
4978 The default value for this string is Please enter username: .
4979 For all item_types other than PAM_CONV, item is a pointer to a NULL-terminated character string. In the case of PAM_CONV, item points to an initialized pam_conv structure.

Return Value
4982 PAM_SUCCESS
4983 Success.
Chapter 1. Libraries

An attempt was made to replace the conversation structure with a NULL value.

Function ran out of memory making a copy of the item.

Application attempted to set an undefined item.

Errors

May be translated to text with `pam_strerror`. 
**pam_setcred**

**Name**

pam_setcred — set the module-specific credentials of the user

**Synopsis**

```c
#include <security/pam_appl.h>

extern int pam_setcred(pam_handle_t *pamh, int flags);
```

**Description**

pam_setcred sets the module-specific credentials of the user. It is usually called after the user has been authenticated, after the account management function has been called and after a session has been opened for the user.

`flags` maybe specified from among the following values:

- **PAM_ESTABLISH_CRED**
  - set credentials for the authentication service
- **PAM_DELETE_CRED**
  - delete credentials associated with the authentication service
- **PAM_REINITIALIZE_CRED**
  - reinitialize the user credentials
- **PAM_REFRESH_CRED**
  - extend lifetime of the user credentials

Additionally, the value of `flags` may be logically or'd with **PAM_SILENT**.

**Return Value**

- **PAM_SUCCESS**
  - Success.
- **PAM_CRED_UNAVAIL**
  - Module cannot retrieve the user's credentials.
- **PAM_CRED_EXPIRED**
  - User's credentials have expired.
- **PAM_USER_UNKNOWN**
  - User is not known to an authentication module.
- **PAM_CRED_ERR**
Chapter 1. Libraries

Module was unable to set the credentials of the user.

Errors

May be translated to text with pam_strerror.

**pam_start**

Name

pam_start — initialize the PAM library

Synopsis

```
#include <security/pam_appl.h>

int pam_start(const char *service_name, const char *user, const (struct pam_conv *
*pam_conversation), pam_handle_t **pamh);
```

Description

pam_start is used to initialize the PAM library. It must be called prior to any other usage of the PAM library. On success, *pamh becomes a handle that provides continuity for successive calls to the PAM library. pam_start expects arguments as follows: the service_name of the program, the username of the individual to be authenticated, a pointer to an application-supplied pam_conv structure, and a pointer to a pam_handle_t pointer.

An application must provide the conversation function used for direct communication between a loaded module and the application. The application also typically provides a means for the module to prompt the user for a password, etc.

The structure, pam_conv, is defined to be,

```
struct pam_conv {
    int (*conv) (int num_msg,
                const struct pam_message * *msg,
                struct pam_response * *resp,
                void *appdata_ptr);
    void *appdata_ptr;
```
It is initialized by the application before it is passed to the library. The contents of this structure are attached to the *pamh handle. The point of this argument is to provide a mechanism for any loaded module to interact directly with the application program; this is why it is called a conversation structure.

When a module calls the referenced conv function, appdata_ptr is set to the second element of this structure. The other arguments of a call to conv concern the information exchanged by module and application. num_msg holds the length of the array of pointers passed via msg. On success, the pointer resp points to an array of num_msg pam_response structures, holding the application-supplied text. Note that resp is a struct pam_response array and not an array of pointers.

**Return Value**

PAM_SUCCESS
Success.

PAM_BUF_ERR
Memory allocation error.

PAM_ABORT
Internal abort.

**ERRORS**

May be translated to text with pam_strerror.

**pam_strerror**

**Name**

pam_strerror — returns a string describing the PAM error

**Synopsis**

#include <security/pam_appl.h>

const char * pam_strerror(pam_handle_t *pamh, int errnum);

**Description**

pam_strerror returns a string describing the PAM error associated with errnum.

**Return Value**

On success, this function returns a description of the indicated error. The application should not free or modify this string. This returned string will not be translated.
Notes

1. Future versions of this specification might define additional service names.
II. Utility Libraries
Chapter 2. utility Libraries

An LSB-conforming implementation shall also support some utility libraries which are built on top of the interfaces provided by the base libraries. These libraries implement common functionality, and hide additional system dependent information such as file formats and device names.

2.1. Interfaces for libz

Table 2-1 defines the library name and shared object name for the libz library

<table>
<thead>
<tr>
<th>Library:</th>
<th>libz</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libz.so.1</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications:

zlib Manual

2.1.1. Compression Library

2.1.1.1. Interfaces for Compression Library

An LSB conforming implementation shall provide the generic functions for Compression Library specified in Table 2-2, with the full functionality as described in the referenced underlying specification.

Table 2-2. libz - Compression Library Function Interfaces

|-------------------|----------|------------------|------------|------------|-------------------|

Referenced Specification(s)

[1]. zlib Manual
2.2. Data Definitions for libz

This section defines global identifiers and their values that are associated with interfaces contained in libz. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content.

These definitions are intended to supplement those provided in the referenced underlying specifications.

This specification uses ISO/IEC 9899 C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

2.2.1. zlib.h

```c
#define Z_NULL   0
#define MAX_WBITS 15
#define MAX_MEM_LEVEL 9
#define deflateInit2(strm,level,method,windowBits,memLevel,strategy)
deflateInit2_((strm),(level),(method),(windowBits),(memLevel),(strategy),ZLIB_VERSION,
sizeof(z_stream))
#define deflateInit(strm,level) deflateInit_((strm), (level), ZLIB_VERSION,
sizeof(z_stream))
#define inflateInit2(strm,windowBits)   inflateInit2_((strm), (windowBits),
ZLIB_VERSION,
sizeof(z_stream))
#define inflateInit(strm)       inflateInit_((strm), ZLIB_VERSION, sizeof(z_stream))

typedef int intf;

typedef void *voidpf;
typedef unsigned int uInt;
typedef unsigned long uLong;
typedef uLong uLongf;
typedef void *voidp;
typedef unsigned char Byte;
typedef off_t z_off_t;
typedef void *const voidpc;

typedef voidpf (*alloc_func) (voidpf opaque, uInt items, uInt size);
typedef void (*free_func) (voidpf opaque, voidpf address);
struct internal_state
{
    int dummy;
};
typedef Byte Bytef;
typedef uInt uIntf;
typedef struct z_stream_s
{
    Bytef *next_in;
    uInt avail_in;
};
```
Chapter 2. utility Libraries

2.3. Interfaces for libncurses

Table 2-3 defines the library name and shared object name for the libncurses library
Chapter 2. utility Libraries

Table 2-3. libncurses Definition

<table>
<thead>
<tr>
<th>Library:</th>
<th>libncurses</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libncurses.so.5</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications:

X/Open Curses

2.3.1. Interfaces for Curses

2.3.1.1. Interfaces for Curses

An LSB conforming implementation shall provide the generic functions for Curses specified in Table 2-4, with the full functionality as described in the referenced underlying specification.

Table 2-4. libncurses - Curses Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addch [1]</td>
<td></td>
</tr>
<tr>
<td>has_ic [1]</td>
<td></td>
</tr>
<tr>
<td>mvwaddchstr [1]</td>
<td></td>
</tr>
<tr>
<td>scr_init [1]</td>
<td></td>
</tr>
<tr>
<td>vwscanw [1]</td>
<td></td>
</tr>
<tr>
<td>addchnstr [1]</td>
<td></td>
</tr>
<tr>
<td>has_il [1]</td>
<td></td>
</tr>
<tr>
<td>mvwaddchstr [1]</td>
<td></td>
</tr>
<tr>
<td>scr_restore [1]</td>
<td></td>
</tr>
<tr>
<td>waddch [1]</td>
<td></td>
</tr>
<tr>
<td>addchstr [1]</td>
<td></td>
</tr>
<tr>
<td>hline [1]</td>
<td></td>
</tr>
<tr>
<td>mvwaddnstr [1]</td>
<td></td>
</tr>
<tr>
<td>scr_set [1]</td>
<td></td>
</tr>
<tr>
<td>waddchnstr [1]</td>
<td></td>
</tr>
<tr>
<td>addnstr [1]</td>
<td></td>
</tr>
<tr>
<td>idcok [1]</td>
<td></td>
</tr>
<tr>
<td>mvwaddstr [1]</td>
<td></td>
</tr>
<tr>
<td>scrl [1]</td>
<td></td>
</tr>
<tr>
<td>waddchstr [1]</td>
<td></td>
</tr>
<tr>
<td>addstr [1]</td>
<td></td>
</tr>
<tr>
<td>idlok [1]</td>
<td></td>
</tr>
<tr>
<td>mvwchgat [1]</td>
<td></td>
</tr>
<tr>
<td>scroll [1]</td>
<td></td>
</tr>
<tr>
<td>waddnstr [1]</td>
<td></td>
</tr>
<tr>
<td>attr_get [1]</td>
<td></td>
</tr>
<tr>
<td>immedok [1]</td>
<td></td>
</tr>
<tr>
<td>mvwdelch [1]</td>
<td></td>
</tr>
<tr>
<td>scrollok [1]</td>
<td></td>
</tr>
<tr>
<td>waddstr [1]</td>
<td></td>
</tr>
<tr>
<td>attr_off [1]</td>
<td></td>
</tr>
<tr>
<td>inch [1]</td>
<td></td>
</tr>
<tr>
<td>mvwgetch [1]</td>
<td></td>
</tr>
<tr>
<td>set_curterm [1]</td>
<td></td>
</tr>
<tr>
<td>wattr_get [1]</td>
<td></td>
</tr>
<tr>
<td>attr_on [1]</td>
<td></td>
</tr>
<tr>
<td>inchstr [1]</td>
<td></td>
</tr>
<tr>
<td>mvwgetnstr [1]</td>
<td></td>
</tr>
<tr>
<td>set_term [1]</td>
<td></td>
</tr>
<tr>
<td>wattr_off [1]</td>
<td></td>
</tr>
<tr>
<td>attr_set [1]</td>
<td></td>
</tr>
<tr>
<td>inchstr [1]</td>
<td></td>
</tr>
<tr>
<td>mvwgetstr [1]</td>
<td></td>
</tr>
<tr>
<td>setscrreg [1]</td>
<td></td>
</tr>
<tr>
<td>wattr_on [1]</td>
<td></td>
</tr>
<tr>
<td>attrtloff [1]</td>
<td></td>
</tr>
<tr>
<td>init_color [1]</td>
<td></td>
</tr>
<tr>
<td>mvwhline [1]</td>
<td></td>
</tr>
<tr>
<td>setupterm [1]</td>
<td></td>
</tr>
<tr>
<td>wattr_set [1]</td>
<td></td>
</tr>
<tr>
<td>atrtron [1]</td>
<td></td>
</tr>
<tr>
<td>init_pair [1]</td>
<td></td>
</tr>
<tr>
<td>mvwin [1]</td>
<td></td>
</tr>
<tr>
<td>slk_attr_set [1]</td>
<td></td>
</tr>
<tr>
<td>wattroff [1]</td>
<td></td>
</tr>
<tr>
<td>atrtset [1]</td>
<td></td>
</tr>
<tr>
<td>initscr [1]</td>
<td></td>
</tr>
<tr>
<td>mvwinch [1]</td>
<td></td>
</tr>
<tr>
<td>slk_attroff [1]</td>
<td></td>
</tr>
<tr>
<td>wattron [1]</td>
<td></td>
</tr>
<tr>
<td>baudrate [1]</td>
<td></td>
</tr>
<tr>
<td>instr [1]</td>
<td></td>
</tr>
<tr>
<td>mvwinchnstr [1]</td>
<td></td>
</tr>
<tr>
<td>slk_attron [1]</td>
<td></td>
</tr>
<tr>
<td>wattrset [1]</td>
<td></td>
</tr>
<tr>
<td>beep [1]</td>
<td></td>
</tr>
<tr>
<td>insch [1]</td>
<td></td>
</tr>
<tr>
<td>mvwinchstr [1]</td>
<td></td>
</tr>
<tr>
<td>slk_atrset [1]</td>
<td></td>
</tr>
<tr>
<td>wbkgd [1]</td>
<td></td>
</tr>
<tr>
<td>bkgd [1]</td>
<td></td>
</tr>
<tr>
<td>insdelln [1]</td>
<td></td>
</tr>
<tr>
<td>mvwinnstr [1]</td>
<td></td>
</tr>
<tr>
<td>slk_clear [1]</td>
<td></td>
</tr>
<tr>
<td>wbkgdset [1]</td>
<td></td>
</tr>
<tr>
<td>bkgdset [1]</td>
<td></td>
</tr>
<tr>
<td>insertln [1]</td>
<td></td>
</tr>
<tr>
<td>mvwinsch [1]</td>
<td></td>
</tr>
<tr>
<td>slk_color [1]</td>
<td></td>
</tr>
<tr>
<td>wborder [1]</td>
<td></td>
</tr>
<tr>
<td>border [1]</td>
<td></td>
</tr>
<tr>
<td>insnstr [1]</td>
<td></td>
</tr>
<tr>
<td>mvwinsnstr [1]</td>
<td></td>
</tr>
<tr>
<td>slk_init [1]</td>
<td></td>
</tr>
<tr>
<td>wchgt [1]</td>
<td></td>
</tr>
<tr>
<td>box [1]</td>
<td></td>
</tr>
<tr>
<td>insnstr [1]</td>
<td></td>
</tr>
<tr>
<td>mvwinsnstr [1]</td>
<td></td>
</tr>
<tr>
<td>slk_label [1]</td>
<td></td>
</tr>
<tr>
<td>wclear [1]</td>
<td></td>
</tr>
<tr>
<td>can_change_color [1]</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Function</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------</td>
</tr>
</tbody>
</table>
Chapter 2. utility Libraries

|-----------|---------------|-------------|-------------|---------------|

Referenced Specification(s)

[1]. X/Open Curses

An LSB conforming implementation shall provide the generic data interfaces for Curses specified in Table 2-5, with the full functionality as described in the referenced underlying specification.

### Table 2-5. libncurses - Curses Data Interfaces

|------------|----------|-------------|------------|

Referenced Specification(s)

[1]. X/Open Curses

## 2.4. Data Definitions for libncurses

This section defines global identifiers and their values that are associated with interfaces contained in libncurses. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content.

These definitions are intended to supplement those provided in the referenced underlying specifications.

This specification uses ISO/IEC 9899 C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

### 2.4.1. curses.h

```c
#define ERR     (-1)
#define OK      (0)
#define ACS_RARROW      (acs_map['+'])
#define ACS_LARROW      (acs_map[','])
#define ACS_UARROW      (acs_map['-'])
#define ACS_DARROW      (acs_map['.'])
#define ACS_BLOCK       (acs_map['0'])
#define ACS_CKBOARD     (acs_map['a'])
#define ACS_DEGREE      (acs_map['f'])
```
Chapter 2. utility Libraries

#define ACS_PLMINUS     (acs_map['g'])
#define ACS_BOARD       (acs_map['h'])
#define ACS_LANTERN     (acs_map['i'])
#define ACS_LRCORNER    (acs_map['j'])
#define ACS_URCORNER    (acs_map['k'])
#define ACS_ULCORNER    (acs_map['l'])
#define ACS_LLCORNER    (acs_map['m'])
#define ACS_PLUS        (acs_map['n'])
#define ACS_S1  (acs_map['o'])
#define ACS_HLINE       (acs_map['q'])
#define ACS_S9  (acs_map['s'])
#define ACS_LTEE        (acs_map['t'])
#define ACS_RTEE        (acs_map['u'])
#define ACS_BTEE        (acs_map['v'])
#define ACS_TTEE        (acs_map['w'])
#define ACS_VLINE       (acs_map['x'])
#define ACS_DIAMOND     (acs_map['`'])
#define ACS_BULLET      (acs_map['~'])
#define getmaxyx(win,y,x)  
(y=(win)?((win)->_maxy+1):ERR,x=(win)?((win)->_maxx+1):ERR)
#define getbegyx(win,y,x)       
(y=(win)?(win)->_begy:ERR,x=(win)?(win)->_begx:ERR)
#define getyx(win,y,x)  
(y=(win)?(win)->_cury:ERR,x=(win)?(win)->_curx:ERR)
#define getparyx(win,y,x)       
(y=(win)?(win)->_pary:ERR,x=(win)?(win)->_parx:ERR)
#define WA_ALTCHARSET   A_ALTCHARSET
#define WA_ATTRIBUTES   A_ATTRIBUTES
#define WA_BLINK        A_BLINK
#define WA_BOLD A_BOLD
#define WA_DIM  A_DIM
#define WA_HORIZONTAL   A_HORIZONTAL
#define WA_INVIS        A_INVIS
#define WA_LEFT A_LEFT
#define WA_LOW  A_LOW
#define WA_NORMAL       A_NORMAL
#define WA_PROTECT      A_PROTECT
#define WA_REVERSE      A_REVERSE
#define WA_RIGHT        A_RIGHT
#define WA_STANDOUT     A_STANDOUT
#define WA_TOP        A_TOP
#define WA_UNDERLINE    A_UNDERLINE
#define WA_VERTCAL    A_VERTCAL
#define A_REVERSE       NCURSES_BITS(1UL,10)
#define COLOR_BLACK     0
#define COLOR_RED       1
#define COLOR_GREEN     2
#define COLOR_YELLOW    3
#define COLOR_BLUE      4
#define COLOR_MAGENTA   5
#define COLOR_CYAN      6
#define COLOR_WHITE     7
#define _SUBWIN 0x01
Chapter 2. utility Libraries

```c
#define _ENDLINE 0x02
#define _FULLWIN 0x04
#define _ISPAD 0x10
#define _HASMOVED 0x20

typedef unsigned char bool;
typedef unsigned long chtype;
typedef struct screen SCREEN;
typedef struct _win_st WINDOW;
typedef chtype attr_t;
typedef struct
{
    attr_t attr;
    wchar_t chars[5];
}
cchar_t;
struct pdat
{
    short _pad_y;
    short _pad_x;
    short _pad_top;
    short _pad_left;
    short _pad_bottom;
    short _pad_right;
}
;
struct _win_st
{
    short _cury;
    short _curx;
    short _maxy;
    short _maxx;
    short _begy;
    short _begx;
    short _flags;
   (attr_t)_attrs;
    chtype _bkgd;
    bool _notimeout;
    bool _clear;
    bool _leaveok;
    bool _scroll;
    bool _idlok;
    bool _idcok;
    bool _immed;
    bool _sync;
    bool _use_keypad;
    int _delay;
    struct ldat * _line;
    short _regtop;
    short _regbottom;
    int _parx;
```
Chapter 2. utility Libraries

```c
int _pary;
WINDOW *parent;
struct pdat _pad;
short _yoffset;
cchar_t _bkgrnd;
```

```c
#define KEY_CODE_YES    0400
#define KEY_BREAK       0401
#define KEY_MIN 0401
#define KEY_DOWN        0402
#define KEY_UP  0403
#define KEY_LEFT        0404
#define KEY_RIGHT       0405
#define KEY_HOME        0406
#define KEY_BACKSPACE   0407
#define KEY_F0  0410
#define KEY_DL 0510
#define KEY_IL 0511
#define KEY_DC 0512
#define KEY_IC 0513
#define KEY_EIC 0514
#define KEY_CLEAR       0515
#define KEY_EOS 0516
#define KEY_EOL 0517
#define KEY_SF  0520
#define KEY_SR  0521
#define KEY_NPAGE       0522
#define KEY_PPAGE 0523
#define KEY_STAB 0524
#define KEY_CTAB 0525
#define KEY_CATAB 0526
#define KEY_ENTER 0527
#define KEY_SRESET      0530
#define KEY_RESET 0531
#define KEY_PRINT 0532
#define KEY_LL  0533
#define KEY_A1  0534
#define KEY_A3  0535
#define KEY_B2  0536
#define KEY_C1  0537
#define KEY_C3  0540
#define KEY_BTAB 0541
#define KEY_BEG 0542
#define KEY_CANCEL 0543
#define KEY_CLOSE 0544
#define KEY_COMMAND 0545
#define KEY_COPY 0546
#define KEY_CREATE 0547
#define KEY_END 0550
#define KEY_EXIT 0551
#define KEY_FIND 0552
#define KEY_HELP 0553
```
Chapter 2. utility Libraries

#define KEY_MARK 0554
#define KEY_MESSAGE 0555
#define KEY_MOVE 0556
#define KEY_NEXT 0557
#define KEY_OPEN 0560
#define KEY_OPTIONS 0561
#define KEY_PREVIOUS 0562
#define KEY_REDO 0563
#define KEY_REFERENCE 0564
#define KEY_REFRESH 0565
#define KEY_REPLACE 0566
#define KEY_RESTART 0567
#define KEY_RESUME 0570
#define KEY_SAVE 0571
#define KEY_SBEG 0572
#define KEY_SCANCEL 0573
#define KEY_SCOMMAND 0574
#define KEY_SCOPY 0575
#define KEY_SCREATE 0576
#define KEY_SDC 0577
#define KEY(SDL 0600)
#define KEY_SELECT 0601
#define KEY_SEND 0602
#define KEY_SEOL 0603
#define KEY_SEXIT 0604
#define KEY_SFIND 0605
#define KEY_SHELP 0606
#define KEY_SHOME 0607
#define KEY_SIC 0610
#define KEY_SLEFT 0611
#define KEY_SMMESSAGE 0612
#define KEY_SMMOVE 0613
#define KEY_SNEXT 0614
#define KEY_SOPTIONS 0615
#define KEY_SPREREVIOUS 0616
#define KEY_SPRINT 0617
#define KEY_SREDO 0620
#define KEY_SREPLACE 0621
#define KEY_SRIGHT 0622
#define KEY_SRSUME 0623
#define KEY_SSSAVE 0624
#define KEY_SSUSPEND 0625
#define KEY_SUNDO 0626
#define KEY_SUSPEND 0627
#define KEY_SUNDO 0628
#define KEY_SREPLACE 0629
#define KEY_SV_MAX 0777

#define PAIR_NUMBER(a) (((a)& A_COLOR)>>8)
#define NCURSES_BITS(mask,shift) (((mask)<<((shift)+8))
#define A_CHARTEXT (NCURSES_BITS(1UL,0)-1UL)
#define A_NORMAL 0L
Chapter 2. utility Libraries

2.5. Interfaces for libutil

Table 2-6 defines the library name and shared object name for the libutil library

<table>
<thead>
<tr>
<th>Library:</th>
<th>libutil</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libutil.so.1</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications:

this specification

2.5.1. Utility Functions

2.5.1.1. Interfaces for Utility Functions

An LSB conforming implementation shall provide the generic functions for Utility Functions specified in Table 2-7, with the full functionality as described in the referenced underlying specification.

Table 2-7. libutil - Utility Functions Function Interfaces

|-------------|---------------|-------------|

Referenced Specification(s)

[1]. this specification
2.6. Interface Definitions for libutil

The following interfaces are included in libutil and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed above for libutil shall behave as described in the referenced base document.

**forkpty**

**Name**

`forkpty` — Create a new process attached to an available pseudo-terminal

**Synopsis**

```c
#include <pty.h>

int forkpty(int *amaster, char *name, struct termios *termp, struct winsize *winp);
```

**Description**

The `forkpty()` function shall find and open a pseudo-terminal device pair in the same manner as the `openpty()` function. If a pseudo-terminal is available, `forkpty` shall create a new process in the same manner as the `fork()` function, and prepares the new process for login in the same manner as `login_tty()`.

If `termp` is not null, it shall refer to a `termios` structure that shall be used to initialize the characteristics of the slave device. If `winp` is not null, it shall refer to a `winsize` structure used to initialize the window size of the slave device.

**Return Value**

On success, the parent process shall return the process id of the child, and the child shall return 0. On error, no new process shall be created, -1 shall be returned, and `errno` shall be set appropriately. On success, the parent process shall receive the file descriptor of the master side of the pseudo-terminal in the location referenced by `amaster`, and, if `name` is not NULL, the filename of the slave device in `name`.

**Errors**

- **EAGAIN**
  
  Unable to create a new process.

- **ENOENT**

  There are no available pseudo-terminals.

- **ENOMEM**

  Insufficient memory was available.
login

Name
login — login utility function

Synopsis
#include <utmp.h>
void login (struct utmp * ut);

Description
The login function shall update the user accounting databases. The ut parameter shall reference a utmp structure for all fields except the following:
1. The ut_type field shall be set to USER_PROCESS.
2. The ut_pid field shall be set to the process identifier for the current process.
3. The ut_line field shall be set to the name of the controlling terminal device. The name shall be found by examining the device associated with the standard input, output and error streams in sequence, until one associated with a terminal device is found. If none of these streams refers to a terminal device, the ut_line field shall be set to "???". If the terminal device is in the /dev directory hierarchy, the ut_line field shall not contain the leading "/dev/", otherwise it shall be set to the final component of the pathname of the device. If the user accounting database imposes a limit on the size of the ut_line field, it shall truncate the name, but any such limit shall not be smaller than UT_LINESIZE (including a terminating null character).

Return Value
None

Errors
None
login_tty

Name

login_tty — Prepare a terminal for login

Synopsis

#include <utmp.h>
int login_tty (int fdr);

Description

The login_tty() function shall prepare the terminal device referenced by the file descriptor fdr. This function shall create a new session, make the terminal the controlling terminal for the current process, and set the standard input, output, and error streams of the current process to the terminal. If fdr is not the standard input, output or error stream, then login_tty() shall close fdr.

Return Value

On success, login_tty() shall return zero; otherwise -1 is returned, and errno shall be set appropriately.

Errors

ENOTTY

fdr does not refer to a terminal device.
logout

Name
logout — logout utility function

Synopsis

```c
#include <utmp.h>
int logout (const char * line );
```

Description

Given the device `line`, the `logout` function shall search the user accounting database which is read by `getutent` for an entry with the corresponding line, and with the type of `USER_PROCESS`. If a corresponding entry is located, it shall be updated as follows:

1. The `ut_name` field shall be set to zeroes (`UT_NAMESIZE` NUL bytes).
2. The `ut_host` field shall be set to zeroes (`UT_HOSTSIZE` NUL bytes).
3. The `ut_tv` shall be set to the current time of day.
4. The `ut_type` field shall be set to `DEAD_PROCESS`.

Return Value

On success, the `logout()` function shall return non-zero. Zero is returned if there was no entry to remove, or if the `utmp` file could not be opened or updated.
logwtmp

Name

logwtmp — append an entry to the wtmp file

Synopsis

```c
#include <utmp.h>
void logwtmp (const char * line, const char * name, const char * host);
```

Description

If the process has permission to update the user accounting databases, the logwtmp function shall append a record to the user accounting database that records all logins and logouts. The record to be appended shall be constructed as follows:

1. The `ut_line` field shall be initialized from `line`. If the user accounting database imposes a limit on the size of the `ut_line` field, it shall truncate the value, but any such limit shall not be smaller than `UT_LINESIZE` (including a terminating null character).

2. The `ut_name` field shall be initialized from `name`. If the user accounting database imposes a limit on the size of the `ut_name` field, it shall truncate the value, but any such limit shall not be smaller than `UT_NAMESIZE` (including a terminating null character).

3. The `ut_host` field shall be initialized from `host`. If the user accounting database imposes a limit on the size of the `ut_host` field, it shall truncate the value, but any such limit shall not be smaller than `UT_HOSTSIZE` (including a terminating null character).

4. If the `name` parameter does not refer to an empty string (i.e. `""`), the `ut_type` field shall be set to `USER_PROCESS`; otherwise the `ut_type` field shall be set to `DEAD_PROCESS`.

5. The `ut_id` field shall be set to the process identifier for the current process.

6. The `ut_tv` field shall be set to the current time of day.

If a process does not have write access to the user accounting database, the logwtmp function will not update it. Since the function does not return any value, an application has no way of knowing whether it succeeded or failed.

Return Value

None.
openpty

Name
openpty — find and open an available pseudo-terminal

Synopsis
#include <pty.h>
int openpty(int *amaster, int *slave, char *name, struct termios *termp, struct winsize *winp);

Description
The openpty() function shall find an available pseudo-terminal and return file descriptors for the master and slave devices in the locations referenced by amaster and slave respectively. If name is not NULL, the filename of the slave shall be placed in the user supplied buffer referenced by name. If termp is not NULL, it shall point to a termios structure used to initialize the terminal parameters of the slave pseudo-terminal device. If winp is not NULL, it shall point to a winsize structure used to initialize the window size parameters of the slave pseudo-terminal device.

Return Value
On success, zero is returned. On error, -1 is returned, and errno is set appropriately.

Errors
ENOENT
There are no available pseudo-terminals.
III. Commands and Utilities
Chapter 3. Commands and Utilities

3.1. Commands and Utilities

Table 3-1 lists the Commands and Utilities required to be present on a conforming system. These commands and utilities shall behave as described in the relevant underlying specification, with the following exceptions:

1. If any operand (except one which follows --) starts with a hyphen, the behavior is unspecified.

Rationale (Informative)

Applications should place options before operands, or use --, as needed. This text is needed because GNU option parsing differs from POSIX. For example, ls . -a in GNU ls means to list the current directory, showing all files (that is, "." is an operand and --a is an option). In POSIX, "." and --a are both operands, and the command means to list the current directory, and also the file named --a. Suggesting that applications rely on the setting of the POSIXLY_CORRECT environment variable, or try to set it, seems worse than just asking the applications to invoke commands in ways which work with either the POSIX or GNU behaviors.

The behavior of the interfaces described in this section is specified by the following standards.

this specification

Table 3-1. Commands and Utilities

<table>
<thead>
<tr>
<th>[ ]</th>
<th>[ ]</th>
<th>[ ]</th>
<th>[ ]</th>
<th>[ ]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ar</td>
<td>at</td>
<td>awk</td>
<td>basename</td>
</tr>
<tr>
<td>batch</td>
<td>bc</td>
<td>cat</td>
<td>chfn</td>
<td>chgrp</td>
</tr>
<tr>
<td>chmod</td>
<td>chown</td>
<td>chsh</td>
<td>cksum</td>
<td>cmp</td>
</tr>
<tr>
<td>col</td>
<td>comm</td>
<td>cp</td>
<td>cpio</td>
<td>crontab</td>
</tr>
<tr>
<td>csplit</td>
<td>cut</td>
<td>date</td>
<td>dd</td>
<td>df</td>
</tr>
<tr>
<td>diff</td>
<td>dirname</td>
<td>dmesg</td>
<td>du</td>
<td>echo</td>
</tr>
<tr>
<td>egrep</td>
<td>env</td>
<td>expand</td>
<td>expr</td>
<td>false</td>
</tr>
<tr>
<td>fgrep</td>
<td>file</td>
<td>find</td>
<td>fold</td>
<td>fuser</td>
</tr>
<tr>
<td>gencat</td>
<td>getconf</td>
<td>gettext</td>
<td>grep</td>
<td>groupadd</td>
</tr>
<tr>
<td>groupdel</td>
<td>groupmod</td>
<td>groups</td>
<td>gunzip</td>
<td>gzip</td>
</tr>
<tr>
<td>head</td>
<td>hostname</td>
<td>iconv</td>
<td>id</td>
<td>install</td>
</tr>
<tr>
<td>install_initd</td>
<td>ipcrm</td>
<td>ipcs</td>
<td>join</td>
<td>kill</td>
</tr>
<tr>
<td>killall</td>
<td>ln</td>
<td>locale</td>
<td>localedef</td>
<td>logname</td>
</tr>
<tr>
<td>lpr</td>
<td>ls</td>
<td>lsb_release</td>
<td>m4</td>
<td>make</td>
</tr>
<tr>
<td>man</td>
<td>md5sum</td>
<td>mkdir</td>
<td>mkfifo</td>
<td>mknod</td>
</tr>
</tbody>
</table>
### 3.2. Command Behavior

This section contains descriptions for commands and utilities whose specified behavior in the LSB contradicts or extends the standards referenced. It also contains commands and utilities only required by the LSB and not specified by other standards.
ar

Name
21  ar — create and maintain library archives (LSB DEPRECATED)

Description
22  ar is deprecated from the LSB and is expected to disappear from a future version of the LSB.

  Rationale
24  The LSB generally does not include software development utilities nor does it specify .o and .a file formats.
25  ar is as specified in ISO POSIX (2003) but with differences as listed below.

Differences
26  -T
27  -C
28  need not be accepted.
29  -l
30  has unspecified behavior.
31  -q
32  has unspecified behavior; using -r is suggested.
Chapter 3. Commands and Utilities

**at**

**Name**

at — examine or delete jobs for later execution

**Description**

at is as specified in ISO POSIX (2003) but with differences as listed below.

**Differences**

- `d`
  - is functionally equivalent to the `-r` option specified in ISO POSIX (2003).
- `-r`
  - need not be supported, but the `-d` option is equivalent.
- `-t` time
  - need not be supported.

**Files**

The files at.allow and at.deny reside in /etc rather than /usr/lib/cron.

**awk**

**Name**

awk — pattern scanning and processing language

**Description**

awk is as specified in ISO POSIX (2003) but with differences as listed below.

**Differences**

Certain aspects of internationalized regular expressions are optional; see Internationalization and Regular Expressions>.
**batch**

**Name**
batch — schedule commands to be executed in a batch queue

**Description**
The specification for batch is as specified in ISO POSIX (2003), but with the following differences as listed below.

**Files**
The files at.allow and at.deny reside in /etc rather than /usr/lib/cron.

**bc**

**Name**
bc — An arbitrary precision calculator language

**Description**
bc is as specified in ISO POSIX (2003) but with differences as listed below.

**Differences**
The bc language may be extended in an implementation defined manner. If an implementation supports extensions, it shall also support the additional options:
- `-s|--standard`
  processes exactly the POSIX bc language.
- `-w|--warn`
  gives warnings for extensions to POSIX bc.
chfn

Name
chfn — change user name and information

Synopsis
chfn [-f full_name] [-h home_phone] [user]

Description
chfn shall update the user database. An unprivileged user may only change the fields for their own account, a user with appropriate privileges may change the fields for any account.
The fields full_name and home_phone may contain any character except:
any control character
comma
colon
equal sign

If none of the options are selected, chfn operates in an interactive fashion. The prompts and expected input in interactive mode are unspecified and should not be relied upon.

As it is possible for the system to be configured to restrict which fields a non-privileged user is permitted to change, applications should be written to gracefully handle these situations.

Standard Options

- \texttt{-f full\_name}
  sets the user's full name.

- \texttt{-h home\_phone}
  sets the user's home phone number.

Future Directions

The following two options are expected to be added in a future version of the LSB:

- \texttt{-o office}
  sets the user's office room number.

- \texttt{-p office\_phone}
  sets the user's office phone number.

Note that some implementations contain a "-o other" option which specifies an additional field called "other".

Traditionally, this field is not subject to the constraints about legitimate characters in fields. Also, one traditionally shall have appropriate privileges to change the other field. At this point there is no consensus about whether it is desirable to specify the other field; applications may wish to avoid using it.

The "-w work\_phone" field found in some implementations should be replaced by the "-p office\_phone" field. The "-r room\_number" field found in some implementations is the equivalent of the "-o office" option mentioned above; which one of these two options to specify will depend on implementation experience and the decision regarding the other field.
chgrp

Name

chgrp — change file group

Description

chgrp is as specified in ISO POSIX (2003) but with differences as listed below.

Differences

The -L, -H, and -P options need not be supported.

chown

Name

chown — change file owner and group

Description

chown is as specified in ISO POSIX (2003) but with differences as listed below.

Differences

The -L, -H, and -P options need not be supported.
**chsh**

**Name**
chsh — change login shell

**Synopsis**

```
chsh [-s login_shell][user]
```

**Description**

chsh changes the user login shell. This determines the name of the user's initial login command. An unprivileged user may only change the login shell for their own account, a user with appropriate privilege may change the login shell for any account specified by user.

Unless the user has appropriate privilege, the initial login command name shall be one of those listed in `/etc/shells`. The `login_shell` shall be the absolute path (i.e. it must start with `/`) to an executable file. Accounts which are restricted (in an implementation-defined manner) may not change their login shell.

If the `-s` option is not selected, chsh operates in an interactive mode. The prompts and expected input in this mode are unspecified.

**Standard Options**

```
-s login_shell
```

sets the login shell.

**col**

**Name**

col — filter reverse line feeds from input

**Description**

col is as specified in the SUSv2 with the difference that the `-p` option has unspecified behavior.

Although col is shown as legacy in SUSv2, Version 2, it is not (yet) deprecated in the LSB.
cpi

Name

cpio — copy file archives in and out

Description

cpio is as specified in ISO POSIX (2003), but with differences as listed below.

Differences

Some elements of the Pattern Matching Notation are optional; see Internationalization and Pattern Matching Notation.

crontab

Name

crontab — maintain crontab files for individual users

Synopsis

crontab [-u user] file

crontab [-u user] {-l | -r | -e}

Description

crontab is as specified in ISO POSIX (2003), but with differences as listed below.

Files

The files cron.allow and cron.deny reside in /etc rather than /usr/lib/cron.
cut

Name

cut — split a file into sections determined by context lines

Description

cut is as specified in ISO POSIX (2003), but with differences as listed below.

Differences

- n
  has unspecified behavior.

df

Name

df — report filesystem disk space usage

Description

df is as specified in ISO POSIX (2003), but with the following differences.

If the -k option is not specified, disk space is shown in unspecified units. Applications should specify -k.

If an argument is the absolute file name of a disk device node containing a mounted filesystem, df shows the space available on that filesystem rather than on the filesystem containing the device node (which is always the root filesystem).
**dmesg**

**Name**

dmesg — print or control the system message buffer

**Synopsis**

dmesg [-c | -n level | -s bufsize]

**Description**

dmesg examines or controls the system message buffer. Only a user with appropriate privileges may modify the system message buffer parameters or contents.

**Standard Options**

- `-c`
  If the user has appropriate privilege, clears the system message buffer contents after printing.

- `-n level`
  If the user has appropriate privilege, sets the level at which logging of messages is done to the console.

- `-s bufsize`
  uses a buffer of bufsize to query the system message buffer. This is 16392 by default (this matches the default kernel syslog buffer size since 2.1.113). If you have set the kernel buffer to larger than the default then this option can be used to view the entire buffer.

**du**

**Name**

du — estimate file space usage

**Description**

du is as specified in ISO POSIX (2003), but with differences as listed below.

**Differences**

If the `-k` option is not specified, disk space is shown in unspecified units. Applications should specify `-k`.
Chapter 3. Commands and Utilities

**echo**

**Name**

echo — display a line of text

**Synopsis**

echo [STRING...]

**Description**

The `echo` command is as specified in ISO POSIX (2003), but with the following differences.

Unlike the behavior specified in ISO POSIX (2003), whether `echo` supports options is implementation defined. The behavior of `echo` if any arguments contain backslashes is also implementation defined. Conforming applications shall not run `echo` with a first argument starting with a hyphen, or with any arguments containing backslashes; they shall use `printf` in those cases.

The behavior specified here is similar to that specified by ISO POSIX (2003) without the XSI option. However, the LSB forbids all options and the latter forbids only `-n`.

**egrep**

**Name**

egrep — search a file with an ERE pattern

**Description**

egrep is equivalent to `grep -E`. For further details, see the specification for `grep`.

**fgrep**

**Name**

fgrep — search a file with a fixed pattern

**Description**

fgrep is equivalent to `grep -F`. For further details, see the specification for `grep`. 
file

Name
152 file — determine file type

Description
153 file is as specified in ISO POSIX (2003), but with differences as listed below.

Differences
154 The -M, -h, -d, and -i options need not be supported.

find

Name
155 find — search for files in a directory hierarchy

Description
156 find is as specified in ISO POSIX (2003), but with additional options as specified below.

Differences
157 Some elements of the Pattern Matching Notation are optional; see Internationalization and Pattern Matching Notation.

fuser

Name
158 fuser — identify processes using files or sockets

Description
159 fuser is as specified in ISO POSIX (2003), but with differences as listed below.

Differences
160 -c
161 has unspecified behavior.
162 -f
163 has unspecified behavior.
gettext

Name
gettext — retrieve text string from message catalog

Synopsis
gettext [options] [textdomain] msgid
gettext -s [options] msgid...

Description
The gettext utility retrieves a translated text string corresponding to string msgid from a message object generated with msgfmt utility.
The message object name is derived from the optional argument textdomain if present, otherwise from the
TEXTDOMAIN environment variable. If no domain is specified, or if a corresponding string cannot be found, gettext
prints msgid.
Ordinarily gettext looks for its message object in dirname/lang/LC_MESSAGES where dirname is the
implementation-defined default directory and lang is the locale name. If present, the TEXTDOMAINDIR environment
variable replaces the dirname.
This utility interprets C escape sequences such as \t for tab. Use \ \ to print a backslash. To produce a message on a
line of its own, either put a \n at the end of msgid, or use this command in conjunction with the printf utility.
When used with the -s option the gettext utility behaves like the echo utility, except that the message corresponding
to msgid in the selected catalog provides the arguments.

Options
-d domainname
--domain=domainname
PARAMETER translated messages from domainname.
-e
Enable expansion of some escape sequences.
-n
Suppress trailing newline.

Operands
The following operands are supported:
textdomain
A domain name used to retrieve the messages.
msgid
A key to retrieve the localized message.

**Environment Variables**

LANGUAGE
Specifies one or more locale names.

LANG
Specifies locale name.

LC_MESSAGES
Specifies messaging locale, and if present overrides LANG for messages.

TEXTDOMAIN
Specifies the text domain name, which is identical to the message object filename without `.mo` suffix.

TEXTDOMAINDIR
Specifies the pathname to the message catalog, and if present replaces the implementation-defined default directory.

**Exit Status**

The following exit values are returned:

0
Successful completion.

>0
An error occurred.

**grep**

**Name**
grep — print lines matching a pattern

**Description**
grep is as specified in ISO POSIX (2003), but with differences as listed below.

**LSB Differences**

Some elements of the Pattern Matching Notation are optional; see Internationalization and Pattern Matching Notation.
groupadd

Name

groupadd — create a new group

Synopsis


groupadd [-g gid [-o]] group

Description

If the caller has appropriate privilege, the groupadd command shall create a new group named group. The group name shall be unique in the group database. If no gid is specified, groupadd shall create the new group with a unique group ID.

Options

-\(g\) gid [-o]

The new group shall have group ID gid. If the -o option is not used, no other group shall have this group ID.

The value of gid shall be non-negative.

groupdel

Name


groupdel — delete a group

Synopsis


groupdel group

Description

If the caller has sufficient privilege, the groupdel command shall modify the system group database, deleting the group named group. If the group named group does not exist, groupdel shall issue a diagnostic message and exit with a non-zero exit status.
groupmod

Name
background groupmod — modify a group

Synopsis
background groupmod [-g gid [-o]] [-n group_name] group

Description
background If the caller has appropriate privilege, the groupmod command shall modify the entry in the system group database corresponding to a group named group.

Options
background -g gid [-o]
Modify the group's group ID, setting it to gid. If the -o option is not used, no other group shall have this group ID. The value of gid shall be non-negative.

 Only the group ID in the database is altered; any files with group ownership set to the original group ID are unchanged by this modification.

-n group_name
changes the name of the group from group to group_name.

groups

Name
background groups — display a group

Synopsis
background groups [user]

Description
background The groups command shall behave as id -Gn [user], as specified in ISO POSIX (2003). The optional user parameter will display the groups for the named user.
gunzip

Name

gunzip — uncompress files

Description

gunzip is equivalent to gzip -d. See the specification for gzip for further details.
gzip

Name

gzip — compress or expand files

Synopsis

gzip [-acdfhlNrtvV19] [-S suffix] [name...]

Description

The gzip command shall attempt to reduce the size of the named files. Whenever possible, each file is replaced by one
with the extension .gz, while keeping the same ownership modes, access and modification times. If no files are
specified, or if a file name is -, the standard input is compressed to the standard output. gzip shall only attempt to
compress regular files. In particular, it will ignore symbolic links.

When compressing, gzip uses the deflate algorithm specified in RFC 1951: DEFLATE Compressed Data Format
Specification and stores the result in a file using the gzip file format specified in RFC 1952: GZIP File Format
Specification.

Options

-a, --ascii
    does nothing on LSB conforming systems.
    This option may be deprecated in a future version of this specification.
-c, --stdout, --to-stdout
    writes output on standard output, leaving the original files unchanged. If there are several input files, the output
    consists of a sequence of independently compressed members. To obtain better compression, concatenate all
    input files before compressing them.
-d, --decompress, --uncompress
    the name operands are compressed files, and gzip shall decompress them.
-f, --force
    forces compression or decompression even if the file has multiple links or the corresponding file already exists,
    or if the compressed data is read from or written to a terminal. If the input data is not in a format recognized by
    gzip, and if the option --stdout is also given, copy the input data without change to the standard output: let
    gzip behave as cat. If -f is not given, and when not running in the background, gzip prompts to verify whether
    an existing file should be overwritten.
-l, --list
    lists the compressed size, uncompressed size, ration and uncompressed name for each compressed file. Gives the
    uncompressed size as -1 for files not in gzip format. Additionally displays method, crc and timestamp for the
    uncompress file when used in combination with --verbose.
For decompression, `gzip` shall support at least the following compression methods:

- deflate (RFC 1951: DEFLATE Compressed Data Format Specification)
- compress (ISO POSIX (2003))
- lzh (SCO `compress -H`)
- pack (Huffman encoding)

The crc shall be given as `ffffffff` for a file not in `gzip` format.

With `--name`, the uncompressed name, date and time are those stored within the compress file, if present.

With `--verbose`, the size totals and compression ratio for all files is also displayed, unless some sizes are unknown. With `--quiet`, the title and totals lines are not displayed.

`-L`, `--license`

displays the `gzip` license and quit.

`-n`, `--no-name`

does not save the original file name and time stamp by default when compressing. (The original name is always saved if the name had to be truncated.) When decompressing, do not restore the original file name if present (remove only the gzip suffix from the compressed file name) and do not restore the original time stamp if present (copy it from the compressed file). This option is the default when decompressing.

`-N`, `--name`

always saves the original file name and time stamp when compressing; this is the default. When decompressing, restore the original file name and time stamp if present. This option is useful on systems which have a limit on file name length or when the time stamp has been lost after a file transfer.

`-q`, `--quiet`

suppresses all warnings.

`-r`, `--recursive`

travels the directory structure recursively. If any of the file names specified on the command line are directories, `gzip` will descend into the directory and compress all the files it finds there (or decompress them in the case of `gunzip`).

`-S .suf`, `--suffix .suf`

uses suffix `.suf` instead of `.gz`.

`-t`, `--test`

checks the compressed file integrity.

`-v`, `--verbose`

displays the name and percentage reduction for each file compressed or decompressed.

`-#`, `--fast`, `--best`
regulates the speed of compression using the specified digit #, where \(-1\) or \(--\text{fast}\) indicates the fastest compression method (less compression) and \(-9\) or \(--\text{best}\) indicates the slowest compression method (best compression). The default compression level is \(-6\) (that is, biased towards high compression at expense of speed).

**LSB Deprecated Options**

The behaviors specified in this section are expected to disappear from a future version of the LSB; applications should only use the non-LSB-deprecated behaviors.

```
-V, --version
```

displays the version number and compilation options, then quits.

**hostname**

**Name**

hostname — show or set the system's host name

**Synopsis**

```
hostname [name]
```

**Description**

*hostname* is used to either display or, with appropriate privileges, set the current host name of the system. The host name is used by many applications to identify the machine.

When called without any arguments, the program displays the name of the system as returned by the *gethostname* function.

When called with a *name* argument, and the user has appropriate privilege, the command sets the host name.

It is not specified if the hostname displayed will be a fully qualified domain name. Applications requiring a particular format of hostname should check the output and take appropriate action.
install

Name
install — copy files and set attributes

Synopsis
install [option...] SOURCE DEST
install [option...] SOURCE... DEST
install [-d|--directory] [option...] DIRECTORY...

Description
In the first two formats, copy SOURCE to DEST or multiple SOURCE(s) to the existing DIRECTORY, optionally setting permission modes and file ownership. In the third format, each DIRECTORY and any missing parent directories shall be created.

Standard Options
--backup[=METHOD]
makes a backup of each existing destination file. METHOD may be one of the following:
• none or off never make backups.
• numbered or t make numbered backups. A numbered backup has the form "%s.~%d~", target_name, version_number. Each backup shall increment the version number by 1.
• existing or nil numbered if numbered backups exist, or simple otherwise.
• simple or never append a suffix to the name. The default suffix is '‐', but can be overridden by setting
SIMPLE_BACKUP_SUFFIX in the environment, or via the -S or --suffix option.
If no METHOD is specified, the environment variable VERSION_CONTROL shall be examined for one of the above. Unambiguous abbreviations of METHOD shall be accepted. If no METHOD is specified, or if METHOD is empty, the backup method shall default to existing.
If METHOD is invalid or ambiguous, install shall fail and issue a diagnostic message.
-b
is equivalent to --backup=existing.
-d, --directory
treats all arguments as directory names; creates all components of the specified directories.
-D
creates all leading components of DEST except the last, then copies SOURCE to DEST; useful in the 1st format.
-g GROUP, --group=GROUP
if the user has appropriate privilege, sets group ownership, instead of process' current group. GROUP is either a name in the user group database, or a positive integer, which shall be used as a group-id.

-m MODE, --mode=MODE
sets permission mode (specified as in chmod), instead of the default rwxr-xr-x.

-o OWNER, --owner=OWNER
if the user has appropriate privilege, sets ownership. OWNER is either a name in the user login database, or a positive integer, which shall be used as a user-id.

-p, --preserve-timestamps
copies the access and modification times of SOURCE files to corresponding destination files.

-s, --strip
strips symbol tables, only for 1st and 2nd formats.

-S SUFFIX, --suffix=SUFFIX
equivalent to --backup=existing, except if a simple suffix is required, use SUFFIX.

--verbose
prints the name of each directory as it is created.

-v, --verbose
print the name of each file before copying it to stdout.

install_initd

Name
install_initd — install an init.d file

Synopsis

/usr/lib/lsb/install_initd initd_file

Description
install_initd shall install a system initialization file that has been copied to the /etc/init.d location such that this file shall be run at the appropriate point during system initialization. The install_initrd command is typically called in the postinstall script of a package. See also Section 8.4.
**ipcrm**

**Name**

`ipcrm` — Remove IPC Resources

**Synopsis**

```
ipcrm [-q msgid | -Q msgkey | -s semid | -S semkey | -m shmid | -M shmkey]...
ipcrm [shm | msg | msg] id...
```

**Description**

If any of the `–q`, `–Q`, `–s`, `–S`, `–m`, or `–M` arguments are given, the `ipcrm` shall behave as described in ISO POSIX (2003).

Otherwise, `ipcrm` shall remove the resource of the specified type identified by `id`.

**Future Directions**

A future revision of this specification may deprecate the second synopsis form.

**Rationale**

In its first Linux implementation, `ipcrm` used the second syntax shown in the SYNOPSIS. Functionality present in other implementations of `ipcrm` has since been added, namely the ability to delete resources by key (not just identifier), and to respect the same command line syntax. The previous syntax is still supported for backwards compatibility only.
**ipcs**

**Name**

`ipcs` — provide information on ipc facilities

**Synopsis**

`ipcs [-smq] [-tcp]`

**Description**

`ipcs` provides information on the ipc facilities for which the calling process has read access.

**Resource display options**

- `-m` shared memory segments.
- `-q` message queues.
- `-s` semaphore arrays.

**Output format options**

- `-t` time.
- `-p` pid.
- `-c` creator.

**Application Usage**

In some implementations of `ipcs` the `-a` option will print all information available. In other implementations the `-a` option will print all resource types. Therefore, applications shall not use the `-a` option.

Some implement of `ipcs` implement more output formats than are specified here. These options are not consistent between differing implementations of `ipcs`. Therefore, only the `-t` `-c` and `-p` option flags may be used. At least one of the `-t` `-c` and `-p` options shall be specified.
**killall**

**Name**

killall — kill processes by name

**Synopsis**

```bash
killall [-egiqvw] [-signal] name...
```

```bash
killall -l
```

```bash
killall -V
```

**Description**

**killall** sends a signal to all processes running any of the specified commands. If no signal name is specified, **SIGTERM** is sent. Signals can be specified either by name (e.g. `-HUP`) or by number (e.g. `-1`). Signal 0 (check if a process exists) can only be specified by number.

If the command name contains a slash (`/`), processes executing that particular file will be selected for killing, independent of their name.

**killall** returns a non-zero return code if no process has been killed for any of the listed commands. If at least one process has been killed for each command, **killall** returns zero.

A **killall** process never kills itself (but may kill other **killall** processes).

**Standard Options**

- `-e`

  requires an exact match for very long names. If a command name is longer than 15 characters, the full name may be unavailable (i.e. it is swapped out). In this case, **killall** will kill everything that matches within the first 15 characters. With `-e`, such entries are skipped. **killall** prints a message for each skipped entry if `-v` is specified in addition to `-e`.

- `-g`

  kills the process group to which the process belongs. The kill signal is only sent once per group, even if multiple processes belonging to the same process group were found.

- `-i`

  asks interactively for confirmation before killing.

- `-l`

  lists all known signal names.

- `-q`

  does not complain if no processes were killed.
-v reports if the signal was successfully sent.

**LSB Deprecated Options**

The behaviors specified in this section are expected to disappear from a future version of the LSB; applications should only use the non-LSB-deprecated behaviors.

-V displays version information.
**lpr**

**Name**

lpr — off line print

**Synopsis**


**Description**

lpr uses a spooling daemon to print the named files when facilities become available. If no names appear, the standard input is assumed.

**Standard Options**

- **-l**
  identifies binary data that is not to be filtered but sent as raw input to printer.

- **-p**
  formats with "pr" before sending to printer.

- **-Pprinter**
  sends output to the printer named printer instead of the default printer.

- **-h**
  suppresses header page.

- **-s**
  uses symbolic links.

- **-#copies**
  specifies copies as the number of copies to print.

- **-J name**
  specifies name as the job name for the header page.

- **-T title**
  specifies title as the title used for "pr".
**ls**

**Name**

`ls` — list directory contents

**Description**

`ls` is as specified in ISO POSIX (2003), but with differences listed below.

**Differences**

- `-l`
  
  If the file is a character special or block special file, the size of the file shall be replaced with two unsigned numbers in the format "%u, %u", representing the major and minor device numbers associated with the special file.
  
  The LSB does not specify the meaning of the major and minor devices numbers.

- `-p`
  
  in addition to ISO POSIX (2003) behavior of printing a slash for a directory, `ls -p` may display other characters for other file types.

Certain aspects of the pattern matching notation are optional; see Internationalization and Pattern Matching Notation.
Chapter 3. Commands and Utilities

**lsb_release**

**Name**
lsb_release — print distribution specific information

**Synopsis**
lsb_release [OPTION...]

**Description**
The `lsb_release` command prints certain LSB (Linux Standard Base) and Distribution information.
If no options are given, the `-v` option is assumed.

**Options**
- `-v`, `--version`
  displays version of LSB against which distribution is compliant. The version is expressed as a colon separated list of LSB module descriptions. LSB module descriptions are dash separated tuples containing the module name, version, and architecture name. The output is a single line of text of the following format:
  LSB Version: \\	<ListAsDescribedAbove>
- `-i`, `--id`
  displays string id of distributor. The output is a single line of text of the following format:
  Distributor ID: \\	<DistributorID>
- `-d`, `--description`
  displays single line text description of distribution. The output is of the following format:
  Description: \\	<Description>
- `-r`, `--release`
  displays release number of distribution. The output is a single line of text of the following format:
  Release: \\	<Release>
- `-c`, `--codename`
  displays codename according to distribution release. The output is a single line of text of the following format.
  Codename: \\	<Codename>
- `-a`, `--all`
  displays all of the above information.
- `-s`, `--short`
displays all of the above information in short output format.

-h, --help

displays a human-readable help message.

Examples

The following command will list the LSB Profiles which are currently supported on this platform.

eample% lsb_release -v


m4

Name

m4 — macro processor

Description

m4 is as specified in ISO POSIX (2003), but with extensions as listed below.

Extensions

-P

forces all builtins to be prefixed with m4_. For example, define becomes m4_define.

-I directory

Add directory to the end of the search path for includes.
md5sum

Name
md5sum — generate or check MD5 message digests

Synopsis
md5sum [-c [file] | file]

Description
For each file, write to standard output a line containing the MD5 message digest of that file, followed by one or more blank characters, followed by the name of the file. The MD5 message digest shall be calculated according to RFC 1321: The MD5 Message-Digest Algorithm and output as 32 hexadecimal digits.

If no file names are specified as operands, read from standard input and use "-" as the file name in the output.

Options
-c [file]
checks the MD5 message digest of all files named in file against the message digest listed in the same file. The actual format of file is the same as the output of md5sum. That is, each line in the file describes a file. If file is not specified, read message digests from stdin.

Exit Status
md5sum shall exit with status 0 if the sum was generated successfully, or, in check mode, if the check matched. Otherwise, md5sum shall exit with a non-zero status.
**mknod**

**Name**

`mknod` — make special files

**Synopsis**

```
mknod [-m mode | --mode=mode] name type [major minor]
mknod [--version]
```

**Description**

The `mknod` command shall create a special file named `name` of the given `type`. The `type` shall be one of the following:

- `b` creates a block (buffered) special file with the specified `major` and `minor` device numbers.
- `c, u` creates a character (unbuffered) special file with the specified `major` and `minor` device numbers.
- `p` creates a FIFO.

**Options**

- `-m mode, --mode=mode`
  
  create the special file with file access permissions set as described in `mode`. The permissions may be any absolute value (i.e. one not containing '+' or '-') acceptable to the `chmod` command.

- `--version`
  
  output version information and exit.

  This option may be deprecated in a future release of this specification.

  If `type` is `p`, `major` and `minor` shall not be specified. Otherwise, these parameters are mandatory.

**Future Directions**

This command may be deprecated in a future version of this specification. The `major` and `minor` operands are insufficiently portable to be specified usefully here. Only a FIFO can be portably created by this command, and the `mkfifo` command is a simpler interface for that purpose.
**mktemp**

**Name**

mktemp — make temporary file name (unique)

**Synopsis**

mktemp [-q] [-u] template

**Description**

The `mktemp` command takes the given file name `template` and overwrites a portion of it to create a file name. This file name shall be unique and suitable for use by the application.

The `template` should have at least six trailing 'X' characters. These characters are replaced with characters from the portable filename character set in order to generate a unique name.

If `mktemp` can successfully generate a unique file name, and the `-u` option is not present, the file shall be created with read and write permission only for the current user. The `mktemp` command shall write the filename generated to the standard output.

**Options**

- `-q`
  
  fail silently if an error occurs. Diagnostic messages to `stderr` are suppressed, but the command shall still exit with a non-zero exit status if an error occurs.

- `-u`
  
  operates in `unsafe` mode. A unique name is generated, but the temporary file shall be unlinked before `mktemp` exits. Use of this option is not encouraged.
more

Name
more — display files on a page-by-page basis

Description
more is as specified in ISO POSIX (2003), but with differences as listed below.

Differences
The more command need not respect the LINES and COLUMNS environment variables.
The following additional options may be supported:

- num
  specifies an integer which is the screen size (in lines).

+ num
  starts at line number num.

+/pattern
  Start at the first line matching the pattern, equivalent to executing the search forward (/) command with the given pattern immediately after opening each file.

The following options from ISO POSIX (2003) may behave differently:

- e
  has unspecified behavior.

- i
  has unspecified behavior.

- n
  has unspecified behavior.

- p
  Either clear the whole screen before displaying any text (instead of the usual scrolling behavior), or provide the behavior specified by ISO POSIX (2003). In the latter case, the syntax is "-p command".

- t
  has unspecified behavior.

The more command need not support the following interactive commands:
The `+num` and `+string` options are deprecated in SUSv2, and have been removed in ISO POSIX (2003); however this specification continues to specify them because the publicly available `util-linux` package does not support the replacement (`-p command`). The `+command` option as found in SUSv2 is more general than is specified here, but the `util-linux` package appears to only support the more specific `+num` and `+string` forms.
mount

Name

mount — mount a file system

Synopsis

mount [-hV]
mount [-a] [-fFnrsvw] [-t vfstype]
mount [-fnrsvw] [-o options [...]] [device | dir]
mount [-fnrsvw] [-t vfstype] [-o options] device dir

Description

As described in ISO POSIX (2003), all files in the system are organized in a directed graph, known as the file
hierarchy, rooted at / . These files can be spread out over several underlying devices. The mount command shall attach
the file system found on some underlying device to the file hierarchy.

Options

-v
invoke verbose mode. The mount command shall provide diagnostic messages on stdout.
-a
mount all filesystems (of the given types) mentioned in /etc/fstab.
-F
If the -a option is also present, fork a new incarnation of mount for each device to be mounted. This will do the
mounts on different devices or different NFS servers in parallel.
-f
cause everything to be done except for the actual system call; if it's not obvious, this `fakes' mounting the file
system.
-n
mount without writing in /etc/mtab. This is necessary for example when /etc is on a read-only file system.
-s
ignore mount options not supported by a filesystem type. Not all filesystems support this option.
-r
mount the file system read-only. A synonym is -o ro.
-w
mount the file system read/write. (default) A synonym is -o rw.
Chapter 3. Commands and Utilities

-L label
If the file /proc/partitions is supported, mount the partition that has the specified label.

-U uuid
If the file /proc/partitions is supported, mount the partition that has the specified uuid.

-t vfstype
indicate a file system type of vfstype.

More than one type may be specified in a comma separated list. The list of file system types can be prefixed with no to specify the file system types on which no action should be taken.

-o
options are specified with a -o flag followed by a comma-separated string of options. Some of these options are only useful when they appear in the /etc/fstab file. The following options apply to any file system that is being mounted:

  async
  perform all I/O to the file system asynchronously.

  atime
  update inode access time for each access. (default)

  auto
  in /etc/fstab, indicate the device is mountable with -a.

  defaults
  use default options: rw, suid, dev, exec, auto, nouser, async.

  dev
  interpret character or block special devices on the file system.

  exec
  permit execution of binaries.

  noatime
  do not update file access times on this file system.

  noauto
  in /etc/fstab, indicates the device is only explicitly mountable.

  nodel
  do not interpret character or block special devices on the file system.

  noexec
  do not allow execution of any binaries on the mounted file system.
Chapter 3. Commands and Utilities

nosuid

do not allow set-user-identifier or set-group-identifier bits to take effect.

nouser

forbid an unprivileged user to mount the file system. (default)

remount

remount an already-mounted file system. This is commonly used to change the mount options for a file system, especially to make a read-only file system writable.

ro

mount the file system read-only.

rw

mount the file system read-write.

suid

allow set-user-identifier or set-group-identifier bits to take effect.

sync

do all I/O to the file system synchronously.

user

allow an unprivileged user to mount the file system. This option implies the options noexec, nosuid, nodev unless overridden by subsequent options.

**LSB Deprecated Options**

The behaviors specified in this section are expected to disappear from a future version of the LSB; applications should only use the non-LSB-deprecated behaviors.

-V

output version and exit.
msgfmt

Name
msgfmt — create a message object from a message file

Synopsis
msgfmt [options...] filename...

Description
The msgfmt command generates a binary message catalog from a textual translation description. Message catalogs, or message object files, are stored in files with a .mo extension.

The format of message object files is not guaranteed to be portable. Message catalogs should always be generated on the target architecture using the msgfmt command.

The source message files, otherwise known as portable object files, have a .po extension. The filename operands shall be portable object files. The .po file contains messages to be displayed to users by system utilities or by application programs. The portable object files are text files, and the messages in them can be rewritten in any language supported by the system.

If any filename is -, a portable object file shall be read from the standard input.

The msgfmt command interprets data as characters according to the current setting of the LC_CTYPE locale category.

Options
-c
--check
Detect and diagnose input file anomalies which might represent translation errors. The msgid and msgstr strings are studied and compared. It is considered abnormal that one string starts or ends with a newline while the other does not.

If the message is flagged as c-format (see Comment Handling), check that the msgid string and the msgstr translation have the same number of % format specifiers, with matching types.

-D directory
--directory=directory
Add directory to list for input files search. If filename is not an absolute pathname and filename cannot be opened, search for it in directory. This option may be repeated. Directories shall be searched in order, with the leftmost directory searched first.

-f
--use-fuzzy
Use entries marked as fuzzy in output. If this option is not specified, such entries are not included into the output.

See Comment Handling below.
Chapter 3. Commands and Utilities

- \texttt{\texttt{o\ output\-file}}
- \texttt{--output\-file=output\-file}

Specify the output file name as \texttt{output\-file}. If multiple domains or duplicate msgids in the .po file are present, the behavior is unspecified. If output-file is \texttt{-}, output is written to standard output.

- \texttt{-S}
- \texttt{--strict}

Ensure that all output files have a .mo extension. Output files are named either by the \texttt{-o} (or \texttt{--output\-file}) option, or by domains found in the input files.

- \texttt{-v}
- \texttt{--verbose}

Print additional information to the standard error, including the number of translated strings processed.

**Operands**

The \texttt{filename} operands are treated as portable object files. The format of portable object files is defined in EXTENDED DESCRIPTION.

**Standard Input**

The standard input is not used unless a \texttt{filename} operand is specified as \texttt{"-"}.

**Environment Variables**

\texttt{LANGUAGE}

Specifies one or more locale names.

\texttt{LANG}

Specifies locale name.

\texttt{LC\_ALL}

Specifies locale name for all categories. If defined, overrides LANG, LC\_TYPE and LC\_MESSAGES.

\texttt{LC\_TYPE}

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

\texttt{LC\_MESSAGES}

Specifies messaging locale, and if present overrides LANG for messages.

**Standard Output**

The standard output is not used unless the option-argument of the \texttt{-o} option is specified as \texttt{-}.

**Extended Description**
The format of portable object files (.po files) is defined as follows. Each .po file contains one or more lines, with each line containing either a comment or a statement. Comments start the line with a hash mark (#) and end with the newline character. Empty lines, or lines containing only white-space, shall be ignored. Comments can in certain circumstances alter the behavior of msgfmt. See Comment Handling below for details on comment processing. The format of a statement is:

directive value

Each directive starts at the beginning of the line and is separated from value by white space (such as one or more space or tab characters). The value consists of one or more quoted strings separated by white space. If two or more strings are specified as value, they are normalized into single string using the string normalization syntax specified in ISO C (1999). The following directives are supported:

domain domainname
msgid message_identifier
msgid_plural untranslated_string_plural
msgstr message_string
msgstr\[n\] message_string

The behavior of the domain directive is affected by the options used. See OPTIONS for the behavior when the -o option is specified. If the -o option is not specified, the behavior of the domain directive is as follows:

1. All msgids from the beginning of each .po file to the first domain directive are put into a default message object file, messages (or messages.mo if the --strict option is specified).

2. When msgfmt encounters a domain domainname directive in the .po file, all following msgids until the next domain directive are put into the message object file domainname (or domainname.mo if --strict option is specified).

3. Duplicate msgids are defined in the scope of each domain. That is, amsgid is considered a duplicate only if the identical msgid exists in the same domain.

4. All duplicate msgids are ignored.

The msgid directive specifies the value of a message identifier associated with the directive that follows it. The msgid_plural directive specifies the plural form message specified to the plural message handling functions ngettext, dgettext or dcgettext. The message_identifier string identifies a target string to be used at retrieval time. Each statement containing a msgid directive shall be followed by a statement containing a msgstr directive or msgstr[n] directives.

The msgstr directive specifies the target string associated with the message_identifier string declared in the immediately preceding msgid directive.

The msgstr[n] (where n = 0, 1, 2, ...) directive specifies the target string to be used with plural form handling functions ngettext, dgettext and dcgettext.

Message strings can contain the following escape sequences:

<table>
<thead>
<tr>
<th>Escape Sequence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\n</td>
<td>newline</td>
</tr>
<tr>
<td>\t</td>
<td>tab</td>
</tr>
<tr>
<td>\v</td>
<td>vertical tab</td>
</tr>
</tbody>
</table>
Chapter 3. Commands and Utilities

| \b  | backspace  |
| \r  | carriage return |
| \f  | formfeed |
| \  | backslash |
| "  | double quote |
| \ddd | octal bit pattern |
| \xHH | hexadecimal bit pattern |

Comment Handling

Comments are introduced by a #, and continue to the end of the line. The second character (i.e. the character following the #) has special meaning. Regular comments should follow a space character. Other comment types include:

- `# normal-comments`
- `#. automatic-comments`
- `#: reference...`
- `#, flag`

Automatic and reference comments are typically generated by external utilities, and are not specified by the LSB. The `msgfmt` command shall ignore such comments.

Portable object files may be produced by unspecified tools. Some of the comment types described here may arise from the use of such tools. It is beyond the scope of this specification to describe these tools.

The #, comments require one or more flags separated by the comma (,) character. The following flags can be specified:

- `fuzzy`
  This flag shows that the following `msgstr` string might not be a correct translation. Only the translator (i.e. the individual undertaking the translation) can judge if the translation requires further modification, or is acceptable as is. Once satisfied with the translation, the translator then removes this fuzzy flag.
  If this flag is specified, the `msgfmt` utility will not generate the entry for the immediately following `msgid` in the output message catalog, unless the `--use-fuzzy` is specified.

- `c-format`
- `no-c-format`
  The `c-format` flag indicates that the `msgid` string is used as format string by `printf`-like functions. If the `c-format` flag is given for a string the `msgfmt` utility may perform additional tests to check to validity of the translation.

Plurals
The msgid entry with empty string (""") is called the header entry and is treated specially. If the message string for the header entry contains nplurals=value, the value indicates the number of plural forms. For example, if nplural=4, there are 4 plural forms. If nplurals is defined, there should be a plural=expression on the same line, separated by a semicolon (;) character. The expression is a C language expression to determine which version of msgstr[n] to be used based on the value of n, the last argument of ngettext, dngettext or dcngettext. For example:

nplural=2; plural=n == 1 ? 0 : 1

indicates that there are 2 plural forms in the language; msgstr[0] is used if n == 1, otherwise msgstr[1] is used.

Another example:

nplural=3; plural=n==1 ? 0 : n==2 ? 1 : 2

indicates that there are 3 plural forms in the language; msgstr[0] is used if n == 1, msgstr[1] is used if n == 2, otherwise msgstr[2] is used.

If the header entry contains charset= codeset string, the codeset is used to indicate the codeset to be used to encode the message strings. If the output string's codeset is different from the message string's codeset, codeset conversion from the message strings's codeset to the output string's codeset will be performed upon the call of gettext, dgettext, dcgettext, ngettext, dngettext, and dcngettext. The output string's codeset is determined by the current locale's codeset (the return value of nl_langinfo(CODESET)) by default, and can be changed by the call of bind_textdomain_codeset.

Exit Status

The following exit values are returned:

0
  Successful completion.

>0
  An error occurred.

Application Usage

Neither msgfmt nor any gettext function imposes a limit on the total length of a message. Installing message catalogs under the C locale is pointless, since they are ignored for the sake of efficiency.

Examples

Example 1: Examples of creating message objects from message files.

In this example module1.po, module2.po and module3.po are portable message object files.

example% cat module1.po
# default domain "messages"
msgid "message one"
Chapter 3. Commands and Utilities

```plaintext
example% cat module2.po

# default domain "messages"
msgid "message four"
msgstr "mensaje número cuatro"

#

domain "error_domain"
msgid "error five"
msgstr "error número cinco"

#

domain "window_domain"
msgid "window six"
msgstr "ventana número seises"
```

```plaintext
example% cat module3.po

# default domain "messages"
msgid "message seven"
msgstr "mensaje número siete"
```
The following command will produce the output files `messages`, `help_domain`, and `error_domain`.

```bash
example% msgfmt module1.po
```

The following command will produce the output files `messages`, `help_domain`, `error_domain`, and `window_domain`.

```bash
example% msgfmt module1.po module2.po
```

The following example will produce the output file `hello.mo`.

```bash
example% msgfmt -o hello.mo module3.po
```

**newgrp**

**Name**

`newgrp` — change group ID

**Synopsis**

```bash
newgrp [group]
```

**Description**

The `newgrp` command is as specified in ISO POSIX (2003), but with differences as listed below.

**Differences**

The `-l` option specified in ISO POSIX (2003) need not be supported.
od

Name

od — dump files in octal and other formats

Synopsis


od --traditional [options] [file] [[+]offset [.] [b]] [ [+]+label [.] [b]]

Description

od is as specified in ISO POSIX (2003), but with differences as listed below.

Differences

-w width, --width=[width]

each output line is limited to width bytes from the input.

--traditional

accepts arguments in traditional form.

The XSI optional behavior described in ISO POSIX (2003) is not supported unless the --traditional option is also specified.

Pre-POSIX and XSI Specifications

The LSB supports option intermixtures with the following pre-POSIX and XSI options:

-a

is equivalent to -t a, selects named characters.

-b

is equivalent to -t o1, selects octal bytes.

-c

is equivalent to -t c, selects characters.

-d

is equivalent to -t u2, selects unsigned decimal two byte units.

-f

is equivalent to -t fF, selects floats.

-i
is equivalent to \texttt{-t d2}, selects decimal two byte units.

This usage may change in future releases; portable applications should use \texttt{-t d2}.

\texttt{-l}

is equivalent to \texttt{-t d4}, selects decimal longs.

\texttt{-o}

is equivalent to \texttt{-t o2}, selects octal two byte units.

\texttt{-x}

is equivalent to \texttt{-t x2}, selects hexadecimal two byte units.

Note that the XSI option \texttt{-s} need not be supported.

\section*{Traditional Usage}

If the \texttt{--traditional} is specified, there may be between zero and three operands specified.

If no operands are specified, then od shall read the standard input.

If there is exactly one operand, and it is an offset of the form [+]\texttt{offset[.][b]}, then it shall be interpreted as specified in ISO POSIX (2003). The file to be dumped shall be the standard input.

If there are exactly two operands, and they are both of the form [+]\texttt{offset[.][b]}, then the first shall be an treated as an offset (as above), and the second shall be a label, in the same format as the offset. If a label is specified, then the first output line produced for each input block shall be preceded by the input offset, cumulative across input files, of the next byte to be written, followed by the label, in parentheses. The label shall increment in the same manner as the offset.

If there are three operands, then the first shall be the file to dump, the second the offset, and the third the label.
passwd

Name

passwd — change user password

Synopsis

passwd [-x max] [-n min] [-w warn] [-i inact] name
passwd [-l | -u] name

Description

passwd changes passwords for user and group accounts. A normal user may only change the password for their own account, the super user may change the password for any account. passwd also changes password expiry dates and intervals. Applications may not assume the format of prompts and anticipated input for user interaction, because they are unspecified.

Options

-x max
sets the maximum number of days a password remains valid.

-n min
sets the minimum number of days before a password may be changed.

-w warn
sets the number of days warning the user will receive before their password will expire.

-i inactive
disables an account after the password has been expired for the given number of days.

-l
disables an account by changing the password to a value which matches no possible encrypted value.

-u
re-enables an account by changing the password back to its previous value.
**patch**

**Name**

patch — apply a diff file to an original

**Description**

patch is as specified in ISO POSIX (2003), but with extensions as listed below.

**Extensions**

--binary
reads and write all files in binary mode, except for standard output and /dev/tty. This option has no effect on POSIX-compliant systems.

-u, --unified
interprets the patch file as a unified context diff.

**pidof**

**Name**

pidof — find the process ID of a running program

**Synopsis**

pidof [-s] [-x] [-o omitpid...] program...

**Description**

Return the process ID of a process which is running the program named on the command line.

**Options**

-s
instructs the program to only return one pid.

-x
causes the program to also return process id's of shells running the named scripts.

-o
omits processes with specified process id.
remove_initd

Name
remove_initd — clean up boot script system modifications introduced by install_initd

Synopsis
/usr/lib/lsb/remove_initd initd_file

Description
remove_initd processes the removal of the modifications made to a distribution's boot script system by the install_initd program. This cleanup is performed in the preuninstall script of a package; however, the package manager is still responsible for removing the /etc/init.d file. See also Section 8.4.

renice

Name
renice — alter priority of running processes

Description
renice is as specified in ISO POSIX (2003), but with differences as listed below.

Differences
-n increment
has unspecified behavior.

sed

Name
sed — stream editor

Description
sed is as specified in ISO POSIX (2003), but with differences as listed below.

LSB Differences
Certain aspects of internationalized regular expressions are optional; see Internationalization and Regular Expressions.>.
**sendmail**

**Name**

`sendmail` — an electronic mail transport agent

**Synopsis**

`sendmail` [options] [address...]

**Description**

To deliver electronic mail (email), applications shall support the interface provided by `/usr/sbin/sendmail` (described here). This interface shall be the default delivery method for applications.

This program sends an email message to one or more recipients, routing the message as necessary. This program is not intended as a user interface routine.

With no options, `sendmail` reads its standard input up to an end-of-file or a line consisting only of a single dot and sends a copy of the message found there to all of the addresses listed. It determines the network(s) to use based on the syntax and contents of the addresses.

It is recommended that applications use as few options as necessary, none if possible.

Some agents allow aliasing on the local system to be prevented by preceding the address with a backslash.

The format of messages shall be as defined in RFC 2822.

**Options**

- `-bm`
  
  reads mail from standard input and delivers to the recipient addresses. This is the default mode of operation.

- `-bp`
  
  lists information about messages currently in the input mail queue.

- `-bs`
  
  uses the SMTP protocol as described in RFC 2821; reads SMTP commands on standard input and writes SMTP responses on standard output.

  Note that RFC 2821 specifies \n (CR-LF) be used at the end of each line, but pipes almost always use \n (LF) instead. To deal with this, agents will accept both \n and \n at the end of each line. When accepting \n, the \r before the \n is silently discarded.

- `-F fullname`
  
  explicitly sets the full name of the sender for incoming mail unless the message already contains a From: message header.

  If the user running `sendmail` is not sufficiently trusted, then the actual sender may be indicated in the message, depending on the behavior of the agent.
-f name
explicitly sets the envelope sender address for incoming mail. If there is no From: header, the address specified in
the From: header will also be set.
If the user running sendmail is not sufficiently trusted, then the actual sender will be indicated in the message.

-i
ignores dots alone on lines by themselves in incoming messages. If -bs is also used, the behavior is unspecified.

-odb
delivers any mail in background, if supported; otherwise ignored.

-odf
delivers any mail in foreground, if supported; otherwise ignored.

-oem or -em
mails errors back to the sender. (default)

-oep or -ep
writes errors to the standard error output.

-oeq or -eq
does not send notification of errors to the sender. This only works for mail delivered locally.

-oi
is equivalent to -i.

-om
indicates that the sender of a message should receive a copy of the message if the sender appears in an alias
expansion. Ignored if aliases are not supported.

-t
reads the message to obtain recipients from the To:, Cc:, and Bcc: headers in the message instead of from the
command arguments. If a Bcc: header is present, it is removed from the message unless there is no To: or Cc:
header, in which case a Bcc: header with no data is created, in accordance with RFC 2822.
If there are any arguments, they specify addresses to which the message is not to be delivered. That is, the
argument addresses are removed from the recipients list obtained from the headers. Note: some agents implement
this behavior in reverse, adding addresses instead of removing them. Others may disallow addresses in argument
list. Therefore, applications should not put addresses in the argument list if -t is used.
This option is sometimes ignored when not in -bm mode (the default).

Exit status

0
successful completion on all addresses. This does not indicate successful delivery.
there was an error.

Notes/Rationale

This page is believed to reflect functionality provided by smail, exim and other implementations, not just the sendmail implementation.
shutdown

Name

`shutdown` — bring the system down

Synopsis

```
/sbin/shutdown [-t sec] [-arkcfF] time [warning-message]
```

Description

`shutdown` brings the system down in a secure way. All logged-in users are notified that the system is going down, and `login()` is blocked. It is possible to shut the system down immediately or after a specified delay. All processes are first notified that the system is going down by the signal SIGTERM. If neither the `-h` or the `-r` argument is used, then the default behavior is to take the system to runlevel one where administrative tasks can be run.

Standard Options

- `-a`  
  uses `/etc/shutdown.allow`.

- `-t sec`  
  tells init(8) to wait sec seconds between sending processes the warning and the kill signal, before changing to another runlevel.

- `-k`  
  doesn’t really shutdown; only sends the warning messages to everybody.

- `-r`  
  reboots after shutdown.

- `-h`  
  halts after shutdown. Powering off after halting is unspecified.

- `-f`  
  skips fsck on reboot.

- `-F`  
  forces fsck on reboot.

- `-c`  
  cancels an already running `shutdown`. With this option, it is of course not possible to give the time argument, but you can enter a explanatory message on the command line that will be sent to all users.

- `time`
specifies when to shut down.

The time argument can have different formats. First, it can be an absolute time in the format hh:mm, in which hh is the hour (1 or 2 digits) and mm is the minute of the hour (in two digits). Second, it can be in the format +m, in which m is the number of minutes to wait. The word now is an alias for +0.

If `shutdown` is called with a delay, it creates the advisory file `/etc/nologin` which causes programs such as `login(1)` to not allow new user logins. `shutdown` only removes this file if it is stopped before it can signal init (i.e. it is cancelled or something goes wrong). Otherwise it is the responsibility of the system shutdown or startup scripts to remove this file so that users can login.

`warning-message` specifies message to send all users.
**su**

**Name**

*su* — change user ID or become super-user

**Synopsis**

```
su [options] [-] [username [ARGS]]
```

**Description**

*su* is used to become another user during a login session. Invoked without a username, *su* defaults to becoming the super user. The optional argument `-` may be used to provide an environment similar to what the user would expect had the user logged in directly.

The user will be prompted for a password, if appropriate. Invalid passwords will produce an error message. All attempts, both valid and invalid, are logged to detect abuses of the system. Applications may not assume the format of prompts and anticipated input for user interaction, because they are unspecified.

An optional command can be executed. This is done by the shell specified in `/etc/passwd` for the target user unless the `-s` or `-m` options are used. Any arguments supplied after the username will be passed to the invoked shell (shell shall support the `-c` command line option in order for a command to be passed to it).

The current environment is passed to the new shell. The value of `$PATH` is reset to `/bin:/usr/bin` for normal users, or `/sbin:/bin:/usr/sbin:/usr/bin` for the super user. This may be changed with the `ENV_PATH` and `ENV_SUPATH` definitions in `/etc/login.defs`. When using the `-m` or `-p` options, the user's environment is not changed.

A subsystem login is indicated by the presence of a "*" as the first character of the login shell. The given home directory will be used as the root of a new filesystem which the user is actually logged into.

**Standard Options**

```
-   makes this a login shell.

-c, --command=command

-m, -p, --preserve-environment

-s, --shell=shell
```

makes this a login shell.

passes command to the invoked shell. It is passed directly to the invoked shell (using the shell's `-c` option), so its syntax is whatever that shell can accept.

does not reset environment variables, and keeps the same shell if it is present in `/etc/shells`.

uses shell instead of the default in `/etc/passwd`. The shell specified shall be present in `/etc/shells`. 
sync

Name

sync — flush filesystem buffers

Synopsis

sync

description

Force changed blocks to disk, update the super block.

tar

Name

tar — file archiver

Description

tar is as specified in SUSv2, but with differences as listed below.

Differences

Certain aspects of internationalized filename globbing are optional; see Internationalization and Pattern Matching Notation.

-h
doesn't dump symlinks; dumps the files they point to.

-z
filters the archive through gzip.
umount

Name
umount — unmount file systems

Synopsis
umount [-hV]
umount -a [-nrV] [-t vfstype]
ument [-nrV] device | dir

Description
umount detaches the file system(s) mentioned from the file hierarchy. A file system is specified by giving the
directory where it has been mounted.

Standard Options
-v
invokes verbose mode.
-n
unmounts without writing in /etc/mtab.
-r
tries to remount read-only if unmounting fails.
-a
unmounts all of the file systems described in /etc/mtab except for the proc filesystem.
-t vfstype
indicates that the actions should only be taken on file systems of the specified type. More than one type may be
specified in a comma separated list. The list of file system types can be prefixed with no to specify the file system
types on which no action should be taken.
-f
forces unmount (in case of an unreachable NFS system).

LSB Deprecated Options
The behaviors specified in this section are expected to disappear from a future version of the LSB; applications should
only use the non-LSB-deprecated behaviors.

-V
print version and exits.
useradd

Name
useradd — create a new user or update default new user information

Synopsis
useradd [-c comment] [-d home_dir]
   [-g initial_group] [-G group[,...]]
   [-m [-k skeleton_dir]] [-p passwd] [-r]
   [-s shell] [-u uid [ -o]] login
useradd -D [-g default_group] [-b default_home]
Chapter 3. Commands and Utilities

Description

When invoked without the -D option, **useradd** creates a new user account using the values specified on the command line and the default values from the system. The new user account will be entered into the system files as needed, the home directory will be created, and initial files copied, depending on the command line options.

When invoked with the -D option, **useradd** will either display the current default values, or update the default values from the command line. If no options are specified, **useradd** displays the current default values.

Standard Options

- **-c** comment
  - specifies the new user's password file comment field value.

- **-d** home_dir
  - creates the new user using home_dir as the value for the user's login directory. The default is to append the login name to default_home and use that as the login directory name.

- **-g** initial_group
  - specifies the group name or number of the user's initial login group. The group name shall exist. A group number shall refer to an already existing group. If -g is not specified, the implementation will follow the normal user default for that system. This may create a new group or choose a default group that normal users are placed in. Applications which require control of the groups into which a user is placed should specify -g.

- **-G** group,[...]
  - specifies a list of supplementary groups which the user is also a member of. Each group is separated from the next by a comma, with no intervening whitespace. The groups are subject to the same restrictions as the group given with the -g option. The default is for the user to belong only to the initial group.

- **-m** [-k skeleton_dir]
  - specifies the user's home directory will be created if it does not exist. The files contained in skeleton_dir will be copied to the home directory if the -k option is used, otherwise the files contained in /etc/skel will be used instead. Any directories contained in skeleton_dir or /etc/skel will be created in the user's home directory as well. The -k option is only valid in conjunction with the -m option. The default is to not create the directory and to not copy any files.

- **-p** passwd
  - is the encrypted password, as returned by crypt(3). The default is to disable the account.

- **-r**
  - creates a system account, that is, a user with a UID in the range reserved for system account users. If there is not a UID free in the reserved range the command will fail.

- **-s** shell
specifies the name of the user's login shell. The default is to leave this field blank, which causes the system to select the default login shell.

-u uid [-o]
specifies the numerical value of the user's ID. This value shall be unique, unless the -o option is used. The value shall be non-negative. The default is the smallest ID value greater than 499 which is not yet used.

**Change Default Options**

-b default_home
specifies the initial path prefix for a new user's home directory. The user's name will be affixed to the end of default_home to create the new directory name if the -d option is not used when creating a new account.

-g default_group
specifies the group name or ID for a new user's initial group. The named group shall exist, and a numerical group ID shall have an existing entry.

-s default_shell
specifies the name of the new user's login shell. The named program will be used for all future new user accounts.

-c comment
specifies the new user's password file comment field value.

**Application Usage**

The -D option will typically be used by system administration packages. Most applications should not change defaults which will affect other applications and users.
userdel

Name
userdel — delete a user account and related files

Synopsis
userdel [-r] login

Description
Delete the user account named login. If there is also a group named login, this command may delete the group as well, or may leave it alone.

Options
-r
removes files in the user’s home directory along with the home directory itself. Files located in other file system will have to be searched for and deleted manually.

usermod

Name
usermod — modify a user account

Synopsis
usermod [-c comment] [-d home_dir [ -m]]
[-g initial_group] [-G group[,...]]
[-l login_name] [-p passwd]
Options

- \( \text{-c command} \)
  
  specifies the new value of the user's password file comment field.

- \( \text{-d home_dir} \)
  
  specifies the user's new login directory. If the \( \text{-m} \) option is given the contents of the current home directory will be moved to the new home directory, which is created if it does not already exist.

- \( \text{-g initial_group} \)
  
  specifies the group name or number of the user's new initial login group. The group name shall exist. A group number shall refer to an already existing group.

- \( \text{-G group, [...]} \)
  
  specifies a list of supplementary groups which the user is also a member of. Each group is separated from the next by a comma, with no intervening whitespace. The groups are subject to the same restrictions as the group given with the \( \text{-g} \) option. If the user is currently a member of a group which is not listed, the user will be removed from the group.

- \( \text{-l login_name} \)
  
  changes the name of the user from login to login_name. Nothing else is changed. In particular, the user's home directory name should probably be changed to reflect the new login name.

- \( \text{-p passwd} \)
  
  is the encrypted password, as returned by crypt(3).

- \( \text{-s shell} \)
  
  specifies the name of the user's new login shell. Setting this field to blank causes the system to select the default login shell.

- \( \text{-u uid [ -o]} \)
  
  specifies the numerical value of the user's ID. This value shall be unique, unless the \( \text{-o} \) option is used. The value shall be non-negative. Any files which the user owns and which are located in the directory tree rooted at the user's home directory will have the file user ID changed automatically. Files outside of the user's home directory shall be altered manually.
xargs

Name
xargs — build and execute command lines from standard input

Description
xargs is as specified in ISO POSIX (2003), but with differences as listed below.

Differences
-E has unspecified behavior.
-I has unspecified behavior.
-L has unspecified behavior.
IV. Execution Environment
Chapter 4. File System Hierarchy

An LSB conforming implementation shall provide the mandatory portions of the filesystem hierarchy specified in the Filesystem Hierarchy Standard (FHS), together with any additional requirements made in this specification.

An LSB conforming application shall conform to the Filesystem Hierarchy Standard.

The FHS allows many components or subsystems to be optional. An application shall check for the existence of an optional component before using it, and should behave in a reasonable manner if the optional component is not present.

The FHS requirement to locate the operating system kernel in either / or /boot does not apply if the operating system kernel does not exist as a file in the filesystem.

The FHS specifies certain behaviors for a variety of commands if they are present (for example, ping or python). However, LSB applications shall not rely on any commands beyond those specified by the LSB. The mere existence of a command may not be used as an indication that the command behaves in any particular way.

The following directories or links need not be present: /etc/X11 /usr/bin/X11 /usr/lib/X11 /proc

4.1. /dev

The following shall exist under /dev. Other devices may also exist in /dev. Device names may exist as symbolic links to other device nodes located in /dev or subdirectories of /dev. There is no requirement concerning major/minor number values.

/dev/null

An infinite data source and data sink. Data written to this device shall be discarded. Reads from this device shall always return end-of-file (EOF).

/dev/zero

This device is a source of zeroed out data. All data written to this device shall be discarded. A read from this device shall always return the requested number of bytes, each initialized to the value '\0'.

/dev/tty

In each process, a synonym for the controlling terminal associated with the process group of that process, if any. All reads and writes to this device shall behave as if the actual controlling terminal device had been opened.
Chapter 5. Additional Recommendations

5.1. Minimal granted Directory and File permissions

In this Chapter "System" means an "LSB conforming implementation" and "application" means an "LSB conforming (third party vendor) application".

The system shall grant to the application read and execute permissions on files needed to use all system interfaces (ABIs) required by the LSB specification.

5.2. Recommendations for applications on ownership and permissions

5.2.1. Directory Write Permissions

The application should not depend on having directory write permission outside /tmp, /var/tmp, invoking user's home directory and /var/opt/package, (where package is the name of the application package).

The application should not depend on owning these directories.

For these directories the application should be able to work with directory write permissions restricted by the S_ISVTXT bit (otherwise known as the "sticky bit").

5.2.2. File Write Permissions

The application should not depend on file write permission on files not owned by the user it runs under with the exception of its personal inbox /var/mail/username.

5.2.3. File Read and execute Permissions

The application should not depend on having read permission to every file and directory.

5.2.4. Suid and Sgid Permissions

The application should not depend on the set user ID or set group ID (the S_ISUID or S_ISGID permissions of a file not packaged with the application. Instead, the distribution is responsible for assuming that all system commands have the required permissions and work correctly.

Rationale

In order to implement common security policies it is strongly advisable for applications to use the minimum set of security attributes necessary for correct operation. Applications that require substantial appropriate privilege are likely to cause problems with such security policies.
5.2.5. Privileged users

In general, applications should not depend on running as a privileged user. This specification uses the term "appropriate privilege" throughout to identify operations that cannot be achieved without some special granting of additional privilege.

Applications that have a reason to run with appropriate privilege should outline this reason clearly in their documentation. Users of the application should be informed, that "this application demands security privileges, which could interfere with system security".

The application should not contain binary-only software that requires being run with appropriate privilege, as this makes security auditing harder or even impossible.

5.2.6. Changing permissions

The application shall not change permissions of files and directories that do not belong to its own package. Should an application require that certain files and directories not directly belonging to the package have a particular ownership, the application shall document this requirement, and may fail during installation if the permissions on these files is inappropriate.

5.2.7. Removable Media (Cdrom, Floppy, etc.)

Applications that expect to be runnable from removable media should not depend on logging in as a privileged user, and should be prepared to deal with a restrictive environment. Examples of such restrictions could be default mount options that disable set-user/group-ID attributes, disabling block or character-special files on the medium, or remapping the user and group IDs of files away from any privileged value.

Rationale

System vendors and local system administrators want to run applications from removable media, but want the possibility to control what the application can do.

5.2.8. Installable applications

Where the installation of an application needs additional privileges, it must clearly document all files and system databases that are modified outside of those in /opt/pkg-name and /var/opt/pkg-name, other than those that may be updated by system logging or auditing activities.

Without this, the local system administrator would have to blindly trust a piece of software, particularly with respect to its security.
Chapter 6. Additional Behaviors

6.1. Mandatory Optional Behaviors

This section specifies behaviors in which there is optional behavior in one of the standards on which the LSB relies, and where the LSB requires a specific behavior.

The LSB does not require the kernel to be Linux; the set of mandated options reflects current existing practice, but may be modified in future releases.

LSB conforming implementations shall support the following options defined within the *ISO POSIX (2003)*:

- `_POSIX_FSYNC`
- `_POSIX_MAPPED_FILES`
- `_POSIX_MEMLOCK`
- `_POSIX_MEMLOCK_RANGE`
- `_POSIX_MEMORY_PROTECTION`
- `_POSIX_PRIORITY_SCHEDULING`
- `_POSIX_REALTIME_SIGNALS`
- `_POSIX_THREAD_ATTR_STACKADDR`
- `_POSIX_THREAD_ATTR_STACKSIZE`
- `_POSIX_THREAD_PROCESS_SHARED`
- `_POSIX_THREAD_SAFE_FUNCTIONS`
- `_POSIX_THREADS`
- `_XOPEN_UNIX`

The `opendir()` function shall consume a file descriptor in the same fashion as `open`, and therefore may fail with `EMFILE` or `ENFILE`.

The `START` and `STOP` termios characters shall be changeable, as described as optional behavior in the "General Terminal Interface" section of the *ISO POSIX (2003).*

The `access()` function function shall fail with `errno` set to `EINVAL` if the `amode` argument contains bits other than those set by the bitwise inclusive OR of `R_OK`, `W_OK`, `X_OK` and `F_OK`.

The `link()` function shall require access to the existing file in order to succeed, as described as optional behavior in the *ISO POSIX (2003).*

Calling `unlink()` on a directory shall fail. Calling `link()` specifying a directory as the first argument shall fail. See also `unlink`.

Linux allows `rename()` on a directory without having write access, but the LSB does not require this.

6.1.1. Special Requirements

LSB conforming systems shall enforce certain special additional restrictions above and beyond those required by ISO POSIX (2003).
These additional restrictions are required in order to support the testing and certification programs associated with the LSB. In each case, these are values that defined macros must not have; conforming applications that use these values shall trigger a failure in the interface that is otherwise described as a "may fail".

The fcntl() function shall treat the "cmd" value -1 as invalid.

The whence value -1 shall be an invalid value for the lseek(), fseek() and fcntl() functions.

The value -5 shall be an invalid signal number.

If the sigaddset() or sigdelset() functions are passed an invalid signal number, they shall return with EINVAL. Implementations are only required to enforce this requirement for signal numbers which are specified to be invalid by this specification (such as the -5 mentioned above).

The mode value -1 to the access() function shall be treated as invalid.

A value of -1 shall be an invalid "_PC_..." value for pathconf().

A value of -1 shall be an invalid "_SC_..." value for sysconf().

The nl_item value -1 shall be invalid for nl_langinfo.

The value -1 shall be an invalid "_CS_..." value for confstr().

The value "z" shall be an invalid mode argument to popen().
Chapter 7. Localization

In order to install a message catalog, the installation procedure shall supply the message catalog in a format readable by the `msgfmt` utility, which shall be invoked to compile the message catalog into an appropriate binary format on the target system.

**Rationale**

The original intent was to allow an application to contain the binary GNU MO format files. However, the format of these files is not officially stable, hence it is necessary to compile these catalogs on the target system. These binary catalogs may differ from architecture to architecture as well.

The resulting binary message catalog shall be located in the package's private area under `/opt`, and the application may use `bindtextdomain()` to specify this location.

Implementations shall support the POSIX and C locales as specified in the ISO POSIX (2003).

### 7.1. Regular Expressions

Utilities that process regular expressions shall support Basic Regular Expressions and Extended Regular Expressions as specified in ISO POSIX (2003), with the following exceptions:

- Range expression (such as `[a-z]`) can be based on code point order instead of collating element order.
- Equivalence class expression (such as `[=a=]`) and multi-character collating element expression (such as `[.ch.]`) are optional.
- Handling of a multi-character collating element is optional.

This affects at least the following utilities: `grep` (grep) (including `egrep`), `sed` (sed), and `awk` (awk).

### 7.2. Pattern Matching Notation

Utilities that perform filename pattern matching (also known as Filename Globbing) shall do it as specified in ISO POSIX (2003), Pattern Matching Notation, with the following exceptions:

- Pattern bracket expressions (such as `[a-z]`) can be based on code point order instead of collating element order.
- Equivalence class expression (such as `[=a=]`) and multi-character collating element expression (such as `[.ch.]`) are optional.
- Handling of a multi-character collating element is optional.

This affects at least the following utilities: `cpio` (cpio), `find` (find), `ls` (ls) and `tar` (tar).
V. System Initialization
Chapter 8. System Initialization

8.1. Cron Jobs

In addition to the individual user crontab files specified by ISO POSIX (2003) stored under /var/spool/cron, the process that executes scheduled commands shall also process the following additional crontab files: /etc/crontab, /etc/cron.d/* The installation of a package shall not modify the configuration file /etc/crontab.

If a package wishes to install a job that has to be executed periodically, it shall place a file in one of the following directories:

/etc/cron.daily
/etc/cron.weekly
/etc/cron.monthly

As these directory names suggest, the files within them are executed on a daily, weekly, or monthly basis, respectively, under the control of an entry in one of the system crontab files. See below for the rules concerning the names of files in these directories.

It is recommended that files installed in any of these directories be scripts (e.g. shell scripts, Perl scripts, etc.) so that they may be modified by the local system administrator.

The scripts in these directories should check if all necessary programs are installed before they try to execute them. Otherwise, problems will arise if a package is removed (but not purged), since the configuration files are kept on the system in this situation.

If a certain job has to be executed at a different frequency (e.g. more frequently than daily), the package shall install a file /etc/cron.d/cron-name tagged as a configuration file. This file uses the same syntax as /etc/crontab and is processed by the system automatically.

To avoid namespace conflicts in the /etc/cron.* directories, the filenames used by LSB-compliant packages in /etc/cron.daily, /etc/cron.weekly, /etc/cron.monthly, or /etc/cron.d shall come from a managed namespace. These filenames may be assigned using one of the following methods:

• Assigned namespace. This namespace consists of names which only use the character set [a-z0-9]. In order to avoid conflicts these cron script names shall be reserved through the Linux Assigned Names and Numbers Authority (LANANA). Information about the LANANA may be found at www.lanana.org (http://www.lanana.org).

Commonly used names shall be reserved in advance; developers for projects should be encouraged reserve names from LANANA, so that each distribution can use the same name, and to avoid conflicts with other projects.

• Hierarchical namespace. This namespace consists of script names of the form: [hier1]-[hier2]-...-[name], where name is again taken from the character set [a-z0-9], and where there may be one or more [hier-n] components. [hier1] may either be an LSB provider name assigned by the LANANA, or it may be owners' DNS name in lower case, with at least one ' . e.g. "debian.org", "staroffice.sun.com", etc. The LSB provider name assigned by LANANA shall only consist of the ASCII characters [a-z0-9].

• Reserved namespace. This namespace consists of script names which begin with the character '_', and is reserved for distribution use only. This namespace should be used for core packages only.
8.2. Init Script Actions

Init files provided by LSB applications shall accept one argument, saying what to do:

- **start**: start the service
- **stop**: stop the service
- **restart**: stop and restart the service if the service is already running, otherwise start the service
- **try-restart**: restart the service if the service is already running
- **reload**: cause the configuration of the service to be reloaded without actually stopping and restarting the service
- **force-reload**: cause the configuration to be reloaded if the service supports this, otherwise restart the service if it is running
- **status**: print the current status of the service

The start, stop, restart, force-reload, and status commands shall be supported by all init files; the reload and the try-restart options are optional. Other init script actions may be defined by the init script.

Init files shall ensure that they will behave sensibly if invoked with start when the service is already running, or with stop when it isn’t, and that they don’t kill unfortunately-named user processes. The best way to achieve this is to use the init-script functions provided by `/lib/lsb/init-functions`.

If a service reloads its configuration automatically (as in the case of cron, for example), the reload option of the init file shall behave as if the configuration has been reloaded successfully. The restart, try-restart, reload and force-reload action may be atomic; i.e. if a service is known not to be operational after a restart or reload, the script may return an error without any further action.

These executable files shall not fail obscurely when the configuration files remain but the package has been removed, as the default in [the packaging system] is to leave configuration files on the system after the package has been removed. Only when it is executed with the [purge] option will [the packaging system] remove configuration files. Therefore, you should include a test statement at the top of the file, like this:

```
test -f program-executed-later-in-file || exit 5
```

or take the equivalent action if the init file is not a shell script.

If the status command is given, the init script will return the following exit status codes.

- **0**: program is running or service is OK
- **1**: program is dead and /var/run pid file exists
- **2**: program is dead and /var/lock lock file exists
- **3**: program is not running
- **4**: program or service status is unknown
- **5-99**: reserved for future LSB use
- **100-149**: reserved for distribution use
- **150-199**: reserved for application use
- **200-254**: reserved

In the case of init script commands other than "status" (i.e., "start", "stop", "restart", "try-restart", "reload", and "force-reload"), the init script shall return an exit status of zero if the action described by the argument has been successful. Otherwise, the exit status shall be non-zero, as defined below. In addition to straightforward success, the following situations are also to be considered successful:
Chapter 8. System Initialization

- restarting a service (instead of reloading it) with the "force-reload" argument
- running "start" on a service already running
- running "stop" on a service already stopped or not running
- running "restart" on a service already stopped or not running
- running "try-restart" on a service already stopped or not running

In case of an error, while processing any init script action except for "status", the init script shall print an error message and return one of the following non-zero exit status codes.

1. generic or unspecified error (current practice)
2. invalid or excess argument(s)
3. unimplemented feature (for example, "reload")
4. user had insufficient privilege
5. program is not installed
6. program is not configured
7. program is not running
8-99 reserved for future LSB use
100-149 reserved for distribution use
150-199 reserved for application use
200-254 reserved

Error and status messages should be printed with the logging functions such as log_failure_msg and so on. Scripts may write to standard error or standard output, but implementations need not present text written to standard error/output to the user or do anything else with it.

Since init files may be run manually by a system administrator with non-standard environment variable values for PATH, USER, LOGNAME, etc. init files shall not depend on the values of these environment variables. They should set them to some known/default values if they are needed.

8.3. Comment Conventions for Init Scripts

LSB applications which need to execute script(s) at bootup and/or shutdown may provide one or more init.d files. These files are installed by the install_initd program described below, which copies it into a standard directory and makes whatever other adjustments (creation of symlinks, creation of entries in a database, etc.) are necessary so that the script can be run at boot-time.1

In the init.d file, information about the shell script shall be delimited by the lines "### BEGIN INIT INFO" and "### END INIT INFO". These delimiter lines may contain trailing whitespace, which shall be ignored. Inside this block there shall be lines of the form "# {keyword}: [arg1] [arg2] ...". (All lines inside this block start with a hash ('#) character in the first column, so that shell treats them as comments.) There shall be exactly one space character between "#" and the keyword.2 The following keywords, with their arguments are defined in this specification:

```
# Provides: boot_facility_1 [ boot_facility_2 ...
# Required-Start: boot_facility_1 [ boot_facility_2 ...
# Required-Stop: boot_facility_1 [ boot_facility_2 ...
# Should-Start: boot_facility_1 [ boot_facility_2 ...
# Should-Stop: boot_facility_1 [ boot_facility_2 ...
# Default-Start: run_level_1 [ run_level_2 ...
# Default-Stop: run_level_1 [ run_level_2 ...
# Short-Description: short_description
```
Chapter 8. System Initialization

310

# Description: multiline_description

Additional keywords may be defined in future LSB specifications. Distributions may define local extensions by using the prefix "X-[distribution name]" --- for example, "X-RedHat-foobardecl", or "X-Debian-xyzzydecl".

An init.d shell script may declare using the "Required-Start: " header that it shall not be run until certain boot facilities are provided. This information is used by the installation tool or the boot-time boot-script execution facility to assure that init scripts are run in the correct order. When an init script is run with a "start" argument, the boot facility or facilities specified in the "Provides" header shall be considered present, and hence init scripts which require those boot facilities would then be eligible to be run. When an init script is run with a "stop" argument, the boot facilities specified in the "Provides" header are considered no longer present. There are naming conventions for boot facilities and system facilities, as described in a following section.

Similarly, the "Required-Stop:" header defines which facilities shall still be available during the shutdown of that service. Hence, the init script system should avoid stopping shell scripts which provide those facilities until this shell script is stopped.

The "Should-Start:" header defines which facilities if present should be started before this service. This allows for weak dependencies which do not cause the service to fail if a facility is not available. But may cause reduced functionality of the service. Compliant applications should not rely on the existence of this feature.

The "Should-Stop:" header defines which facilities should be still available during the shutdown of that service.

The "Default-Start" and "Default-Stop" headers define which run levels should by default run the script with a start or stop argument, respectively, to start or stop the services controlled by the init script. 3

The "Short-Description" and "Description" header fields are used to provide text which describes the actions of the init script. The "short_description" shall be a relatively short, pithy description of the init script, where as the "multiline_description" can be a much longer piece of text that may span multiple lines. In a multiline description, each continuation line shall begin with a '#' followed by tab character or a '#' followed by at least two space characters. The multiline description is terminated by the first line that does not match this criteria.

The comment conventions described in this section are only required for use by LSB-compliant applications; system init scripts as provided by LSB-compliant run-time environments are not required to use the scheme outlined here.

8.4. Installation and Removal of init.d Files

An init.d file is installed in /etc/init.d (which may be a symlink to another location). This can be done by the package installer. See Script Names>. During the package's postinstall script, the program "/usr/lib/lsb/install_initd" configures the distribution's boot script system to call the package's init.d file at the appropriate time. 4

The install_initd program takes a single argument, the pathname to the /etc/init.d file. For example:

/usr/lib/lsb/install_initd /etc/init.d/example.com-coffee

The install_initd program shall return an exit status of zero if the init.d file has been successfully installed or if the the init.d file was already installed. If the required boot facilities cannot be fulfilled an exit status of one shall be returned and the init.d file shall not be installed.

When a software package is removed, the package's preuninstall script shall call /usr/lib/lsb/remove_initd and pass the pathname to the /etc/init.d file. The package manager is still responsible for removing the /etc/init.d file; the remove_initd program is provided in case the distribution needs to clean up any other modifications in the distribution's boot script system that might have been made by the install_initd program. For example:
The `remove_initd` program shall return an exit status of zero if the init.d file has been successfully removed or if the the init.d file is not installed. If another init.d file which depends on a boot facility provided by this init.d file is installed, an exit status of one shall be returned and the init.d file shall remained installed.

There should be a tool available to the user (e.g., RedHat's `chkconfig`) which can be used by the system administrator to easily manipulate at which init levels a particular init.d script is started or stopped. This specification currently does not specify such an interface, however.

### 8.5. Run Levels

The following run levels are specified for use by the "Default-Start:" and "Default-Stop:" specifiers as defined by the section Comment Conventions for Init Scripts>. Many LSB run-time environments commonly use these run level definitions, and in the absence of other considerations, providers of run-time environments are strongly encouraged to follow this convention to provide consistency for system administrators who need to work with multiple distributions. However, it is not required that LSB-compliant run-time environments use these run levels; the distribution-provided `install_initd` script may map the run levels specified below to whatever distribution-specified run levels are most appropriate.

<table>
<thead>
<tr>
<th>Run Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>halt</td>
</tr>
<tr>
<td>1</td>
<td>single user mode</td>
</tr>
<tr>
<td>2</td>
<td>multiuser with no network services exported</td>
</tr>
<tr>
<td>3</td>
<td>normal/full multiuser</td>
</tr>
<tr>
<td>4</td>
<td>reserved for local use, default is normal/full multiuser</td>
</tr>
<tr>
<td>5</td>
<td>multiuser with xdm or equivalent</td>
</tr>
<tr>
<td>6</td>
<td>reboot</td>
</tr>
</tbody>
</table>

### 8.6. Facility Names

Boot facilities are used to indicate dependencies in init scripts, as defined in a previous section. Facility names that begin with a dollar sign ('$') are system facility names, defined by the LSB, and SHALL be provided by distributions. LSB applications shall not provide facilities that begin with a dollar sign. This document defines the following facility names:

- `$local_fs`: all local filesystems are mounted
- `$network`: low level networking (ethernet card; may imply PCMCIA running)
- `$named`: daemons which may provide hostname resolution (if present) are running
- `$portmap`: daemons providing SunRPC/ONCRPC portmapping service (if present) are running
- `$remote_fs`: all remote filesystems are mounted
- `$syslog`: system logger is operational
- `$time`: the system time has been set

Other (non-system) facilities may be defined by other LSB applications. These facilities shall be named using the same conventions defined for naming init.d script names. Commonly, the facility provided by an LSB application init.d script will have the same name as the name assigned to the init.d script.
8.7. Script Names

Since the init.d scripts shall live in a single directory, they shall come from a single namespace. Three means of assigning names from this namespace are available:

- Assigned namespace. This namespace consists of names which only use the character set [a-z0-9]. This space is desirable for scripts which system administrators may often wish to run manually: e.g., "/etc/init.d/named restart" In order to avoid conflicts these init.d names shall be reserved through the Linux Assigned Names and Numbers Authority (LANANA). Information about the LANANA may be found at www.lanana.org (http://www.lanana.org).

  Commonly used names shall be reserved in advance; developers for projects should be encouraged to reserve names from LANANA, so that each distribution can use the same name, and to avoid conflicts with other projects.

- Hierarchical namespace. This namespace consists of scripts names which look like this: [hier1]-[hier2]-...-[name], where name is again taken the character set [a-z0-9], and where there may be one or more [hier-n] components. [hier1] may either be an LSB provider name assigned by the LANANA, or it may be owners' DNS name in lower case, with at least one '.' (e.g., "debian.org", "staroffice.sun.com"). The LSB provider name assigned by LANANA shall only consist of the ASCII characters [a-z0-9].

- Reserved namespace. This namespace consists of script names which begin with the character '_', and is reserved for distribution use only. This namespace should be used for core packages only, and in general use of this namespace is highly discouraged.

In general, if a package or some system function is likely to be used on multiple systems, the package developers or the distribution SHOULD get a registered name through LANANA, and distributions should strive to use the same name whenever possible. For applications which may not be "core" or may not be commonly installed, the hierarchical namespace may be more appropriate. An advantage to the hierarchical namespace is that there is no need to consult with the LANANA before obtaining an assigned name.

Short names are highly desirable, since many system administrators like to use them to manually start and stop services. Given this, they should be standardized on a per-package basis. This is the rationale behind having a LANANA organization to assign these names. The LANANA may be called upon to handle other namespace issues, such as package/prerequisites naming (which is essential to making prerequisites to work correctly).

8.8. Init Script Functions

Each LSB-compliant init.d script shall source the file /lib/lsb/init-functions. This file shall cause the following shell script commands to be defined. This can be done either by adding a directory to the PATH variable which defines these commands, or by defining sh aliases. While the distribution-provided aliases may choose to use shell extensions (at the distribution's option), the LSB init.d files themselves should only depend in shell features as defined by the LSB.

The `start_daemon`, `killproc` and `pidofproc` functions shall use this algorithm for determining the status and the pid(s) of the specified program. They shall read the pidfile specified or otherwise /var/run/basename.pid and use the pid(s) herein when determining whether a program is running. The method used to determine the status is implementation defined, but should allow for non-binary programs. Compliant implementations may use other mechanisms besides those based on pidfiles, unless the -p pidfile option has been used. Compliant applications should not rely on such mechanisms and should always use a pidfile. When a program is stopped, it should delete its pidfile. Multiple pid(s) shall be separated by a single space in the pidfile and in the output of `pidofproc`.

```
start_daemon [-f] [-n nicelevel] [-p pidfile] pathname
```

This runs the specified program as a daemon.
start_daemon shall check if the program is already running using the algorithm given above. If so, it shall not start another copy of the daemon unless the \texttt{-f} option is given. The \texttt{-n} option specifies a nice level. See nice(1). start_daemon should return the LSB defined exit status codes. It shall return 0 if the program has been successfully started or is running and not 0 otherwise.

killproc [-p pidfile] pathname [signal]

This stops the specified program. The program is found using the algorithm given above. If a signal is specified, using the \texttt{-signal\_name} or \texttt{-signal\_number} syntaxes as specified by the \texttt{kill} command, the program is sent that signal. Otherwise, a SIGTERM followed by a SIGKILL after some number of seconds shall be sent. If a program has been terminated, the pidfile should be removed if the terminated process has not already done so. Compliant applications may use the basename instead of the pathname. killproc should return the LSB defined exit status codes. If called without a signal, it shall return 0 if the program has been stopped or is not running and not 0 otherwise. If a signal is given, it shall return 0 only if the program is running.

pidofproc [-p pidfile] pathname

This function returns one or more pid(s) for a particular daemon using the algorithm given above. Only pids of running processes should be returned. Compliant applications may use the basename instead of the pathname. pidofproc should return the LSB defined exit status codes for "status". It shall return 0 if the program is running and not 0 otherwise.

log_success_msg "message"

This requests the distribution to print a success message. The message should be relatively short; no more than 60 characters is highly desirable.

log_failure_msg "message"

This requests the distribution to print a failure message. The message should be relatively short; no more than 60 characters is highly desirable.

log_warning_msg "message"

This requests the distribution to print a warning message. The message should be relatively short; no more than 60 characters is highly desirable.

Notes

1. This specification does not require, but is designed to allow, the development of a system which runs boot scripts in parallel. Hence, enforced-serialization of scripts is avoided unless it is explicitly necessary.

2. More than one space, or a tab character, indicates the continuation line.

3. For example, if you want a service to run in runlevels 3, 4, and 5 (only), specify "Default-Start: 3 4 5" and "Default-Stop: 0 1 2 6".
4. For example, `install_initd` might create symbolic links in `/etc/rc2.d` and other such directories which point to the files in `/etc/init.d` (or it might update a database, or some other mechanism). The init.d files themselves should already be in `/etc/init.d` before running `install_initd`.

5. The dollar sign does not indicate variable expansion as in many Linux utilities. Starting a facility name with a dollar sign is merely a way of dividing the namespace between the system and applications.

6. For example, daemons to query DNS, NIS+, or LDAP

7. as defined in RFC 1833

8. In some LSB run-time environments, filesystems such as `/usr` may be remote. Many applications that require `$local_fs` will probably require also require `$remote_fs`

9. i.e., using a network-based time program such as ntp or rdate, or via the hardware Real Time Clock

10. This note is only informative. Commonly used methods check either for the existence of the `/proc/pid` directory or use `/proc/pid/exe` and `/proc/pid/cmdline`. Relying only on `/proc/pid/exe` is discouraged since this results in a not-running status for daemons that are written in a script language.
VI. Users & Groups
Chapter 9. Users & Groups

9.1. User and Group Database

The format of the User and Group databases is not specified. Programs may only read these databases using the provided API. Changes to these databases should be made using the provided commands.

9.2. User & Group Names

Below is a table of required mnemonic user and group names. This specification makes no attempt to numerically assign uid or gid numbers. The exception is the uid and gid for "root" which are equal to 0.

<table>
<thead>
<tr>
<th>User</th>
<th>Group</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>root</td>
<td>root</td>
<td>Administrative user with all appropriate privileges</td>
</tr>
<tr>
<td>bin</td>
<td>bin</td>
<td>Legacy UID/GID&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td>daemon</td>
<td>daemon</td>
<td>Legacy UID/GID&lt;sup&gt;b&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

Notes:

a. The 'bin' UID/GID is included for compatibility with legacy applications. New applications should no longer use the 'bin' UID/GID.
b. The 'daemon' UID/GID was used as an unprivileged UID/GID for daemons to execute under in order to limit their access to the system. Generally daemons should now run under individual UID/GIDs in order to further partition daemons from one another.

Below is a table of optional mnemonic user and group names. This specification makes no attempt to numerically assign uid or gid numbers. If the username exists on a system, then they should be in the suggested corresponding group. These user and group names are for use by distributions, not by applications.

<table>
<thead>
<tr>
<th>User</th>
<th>Group</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>adm</td>
<td>adm</td>
<td>Administrative special privileges</td>
</tr>
<tr>
<td>lp</td>
<td>lp</td>
<td>Printer special privileges</td>
</tr>
<tr>
<td>sync</td>
<td>sync</td>
<td>Login to sync the system</td>
</tr>
<tr>
<td>shutdown</td>
<td>shutdown</td>
<td>Login to shutdown the system</td>
</tr>
<tr>
<td>halt</td>
<td>halt</td>
<td>Login to halt the system</td>
</tr>
<tr>
<td>mail</td>
<td>mail</td>
<td>Mail special privileges</td>
</tr>
</tbody>
</table>
The differences in numeric values of the uids and gids between systems on a network can be reconciled via NIS, rdist(1), rsync(1), or ugidd(8). Only a minimum working set of “user names” and their corresponding “user groups” are required. Applications cannot assume non system user or group names will be defined.

Applications cannot assume any policy for the default umask or the default directory permissions a user may have. Applications should enforce user only file permissions on private files such as mailboxes. The location of the users home directory is also not defined by policy other than the recommendations of the FHS and shall be obtained by the *pwnam(3) calls.

### 9.3. UID Ranges

The system UIDs from 0 to 99 should be statically allocated by the system, and shall not be created by applications. The system UIDs from 100 to 499 should be reserved for dynamic allocation by system administrators and post install scripts using useradd(1).

### 9.4. Rationale

The purpose of specifying optional users and groups is to reduce the potential for name conflicts between applications and distributions.
Appendix A. Alphabetical Listing of Interfaces

A.1. libc

The behaviour of the interfaces in this library is specified by the following Standards.

Large File Support
this specification
SUSv2
SVID Issue 3
SVID Issue 4

Table A-1. libc Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_Exit(GLIBC_2.1.1)</td>
<td>getrlimit(GLIBC_2.1.1)</td>
</tr>
<tr>
<td>_IO_feof(GLIBC_2.0)</td>
<td>getrlimit64(GLIBC_2.0)</td>
</tr>
<tr>
<td>_IO_getc(GLIBC_2.0)</td>
<td>getusage(GLIBC_2.0)</td>
</tr>
<tr>
<td>_IO_putc(GLIBC_2.0)</td>
<td>getservbyname(GLIBC_2.0)</td>
</tr>
<tr>
<td>_IO_puts(GLIBC_2.0)</td>
<td>getservbyport(GLIBC_2.0)</td>
</tr>
<tr>
<td>__assert_fail(GLIBC_2.0)</td>
<td>getservent(GLIBC_2.0)</td>
</tr>
<tr>
<td>__ctype_b_loc[1]</td>
<td>getsid()[1]</td>
</tr>
<tr>
<td>__ctype_get_mb_cur_max(GLIBC_2.0)[1]</td>
<td>getssockopt()[1]</td>
</tr>
<tr>
<td>__ctype_tolower_loc[1]</td>
<td>getsubopt()[1]</td>
</tr>
<tr>
<td>__ctypetoupper_loc[1]</td>
<td>siginterrupt()[1]</td>
</tr>
<tr>
<td>__errno_location(GLIBC_2.0)</td>
<td>sigPersistent()</td>
</tr>
<tr>
<td>__fxstat(GLIBC_2.0)</td>
<td>getutent(GLIBC_2.0)</td>
</tr>
<tr>
<td>__fxstat64(GLIBC_2.0)</td>
<td>getutent_r(GLIBC_2.0)</td>
</tr>
<tr>
<td>__getpagesize(GLIBC_2.0)</td>
<td>getutxent(GLIBC_2.0)</td>
</tr>
<tr>
<td>__getpgid(GLIBC_2.0)</td>
<td>getutxline()</td>
</tr>
<tr>
<td>__h_errno_location[1]</td>
<td>getw()</td>
</tr>
</tbody>
</table>
| __isinf[1]                                    | sigqueue()[

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>__ctype_b_loc[1]</td>
<td>getsid()[1]</td>
</tr>
<tr>
<td>__ctype_get_mb_cur_max(GLIBC_2.0)[1]</td>
<td>getssockopt()[1]</td>
</tr>
<tr>
<td>__ctype_tolower_loc[1]</td>
<td>getsubopt()[1]</td>
</tr>
<tr>
<td>__ctypetoupper_loc[1]</td>
<td>siginterrupt()[1]</td>
</tr>
<tr>
<td>__errno_location(GLIBC_2.0)</td>
<td>sigPersistent()</td>
</tr>
<tr>
<td>__fxstat(GLIBC_2.0)</td>
<td>getutent(GLIBC_2.0)</td>
</tr>
<tr>
<td>__fxstat64(GLIBC_2.0)</td>
<td>getutent_r(GLIBC_2.0)</td>
</tr>
<tr>
<td>__getpagesize(GLIBC_2.0)</td>
<td>getutxent(GLIBC_2.0)</td>
</tr>
<tr>
<td>__getpgid(GLIBC_2.0)</td>
<td>getutxline()</td>
</tr>
<tr>
<td>__h_errno_location[1]</td>
<td>getw()</td>
</tr>
</tbody>
</table>
| __isinf[1]                                    | sigqueue()[

318
## Appendix A. Alphabetical Listing of Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>__isinff</code>[1]</td>
<td>getwc()[1]</td>
<td>sigreturn()[1]</td>
</tr>
<tr>
<td><code>__isinfl</code>[1]</td>
<td>getwchar()[1]</td>
<td>sigset()[1]</td>
</tr>
<tr>
<td><code>__isnan</code>[1]</td>
<td>getwd()[1]</td>
<td>sigstack()[1]</td>
</tr>
<tr>
<td><code>__isnanf</code>[1]</td>
<td>glob()[1]</td>
<td>sigsuspend()[1]</td>
</tr>
<tr>
<td><code>__isnanl</code>[1]</td>
<td>glob64()[1]</td>
<td>sigstimesd()[1]</td>
</tr>
<tr>
<td><code>__libc_current_sigrtmax</code>[GLIBC_2.1][1]</td>
<td>globfree(GLIBC_2.1)[1]</td>
<td>sigwait(GLIBC_2.1)[1]</td>
</tr>
<tr>
<td><code>__libc_current_sigrtmin</code>[GLIBC_2.1][1]</td>
<td>globfree64(GLIBC_2.1)[1]</td>
<td>sigwaitinfo(GLIBC_2.1)[1]</td>
</tr>
<tr>
<td><code>__libc_start_main</code>[GLIBC_2.0][1]</td>
<td>gntime(GLIBC_2.0)[1]</td>
<td>sleep(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__lxstat</code>[GLIBC_2.0][1]</td>
<td>gntime_r(GLIBC_2.0)[1]</td>
<td>snprintf(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__lxstat64</code>[GLIBC_2.2][1]</td>
<td>grantpt(GLIBC_2.2)[1]</td>
<td>socket(GLIBC_2.2)[1]</td>
</tr>
<tr>
<td><code>__mempcpy</code>[GLIBC_2.0][1]</td>
<td>hcreate(GLIBC_2.0)[1]</td>
<td>socketpair(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__rawmemchr</code>[GLIBC_2.1][1]</td>
<td>hdestroy(GLIBC_2.1)[1]</td>
<td>setsockopt(GLIBC_2.1)[1]</td>
</tr>
<tr>
<td><code>__register_atfork</code>[1]</td>
<td>hsearch()[1]</td>
<td>rand()[1]</td>
</tr>
<tr>
<td><code>__sigsetjmp</code>[GLIBC_2.0][1]</td>
<td>htonl(GLIBC_2.0)[1]</td>
<td>srand48(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__stpcpy</code>[GLIBC_2.0][1]</td>
<td>htons(GLIBC_2.0)[1]</td>
<td>random(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__strtod_internal</code>[GLIBC_2.0][1]</td>
<td>iconv(GLIBC_2.0)[1]</td>
<td>sscanf(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__strtof internal</code>[GLIBC_2.0][1]</td>
<td>iconv_close(GLIBC_2.0)[1]</td>
<td>statvfs(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__strtok_r</code>[GLIBC_2.0][1]</td>
<td>icnv_open(GLIBC_2.0)[1]</td>
<td>statvfs64[1]</td>
</tr>
<tr>
<td><code>__strtodl Internal</code>[GLIBC_2.0][1]</td>
<td>icnv_openr(GLIBC_2.0)[1]</td>
<td>stime(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__strtol_internal</code>[GLIBC_2.0][1]</td>
<td>imaxabs(GLIBC_2.0)[1]</td>
<td>stpcpy(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__strtold_internal</code>[GLIBC_2.0][1]</td>
<td>index(GLIBC_2.0)[1]</td>
<td>strlen(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__strtoall internal</code>[GLIBC_2.0][1]</td>
<td>inet_addr(GLIBC_2.0)[1]</td>
<td>strcspn(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__strouall internal</code>[GLIBC_2.0][1]</td>
<td>inet_n6t(GLIBC_2.0)[1]</td>
<td>strcasestr(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__sysconf</code>[GLIBC_2.2][1]</td>
<td>inet_ntp[1]</td>
<td>strcat(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__sysv_signal</code>[GLIBC_2.0][1]</td>
<td>initgroups(GLIBC_2.0)[1]</td>
<td>strncmp(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__wctod_internal</code>[GLIBC_2.0][1]</td>
<td>initstate(GLIBC_2.0)[1]</td>
<td>strcoll(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__wctof_internal</code>[GLIBC_2.0][1]</td>
<td>insue(GLIBC_2.0)[1]</td>
<td>strcpy(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td><code>__wcstol_internal</code>[GLIBC_2.0][1]</td>
<td>ioctl(GLIBC_2.0)[1]</td>
<td>strspn(GLIBC_2.0)[1]</td>
</tr>
</tbody>
</table>
### Appendix A. Alphabetical Listing of Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>__wctold_internal(GLIBC_2.0)[1]</td>
<td>isalnum(GLIBC_2.0)[1]</td>
<td>strdup(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>__wctoul_internal(GLIBC_2.0)[1]</td>
<td>isalpha(GLIBC_2.0)[1]</td>
<td>strerror(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>__xmlnod(GLIBC_2.0)[1]</td>
<td>isascii(GLIBC_2.0)[1]</td>
<td>strerror_r(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>__xstat(GLIBC_2.0)[1]</td>
<td>isatty(GLIBC_2.0)[1]</td>
<td>strerror(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>__xstat64(GLIBC_2.2)[1]</td>
<td>isblank(GLIBC_2.2)[1]</td>
<td>strerror(GLIBC_2.2)[1]</td>
</tr>
<tr>
<td>_exit(GLIBC_2.0)[1]</td>
<td>iscntrl(GLIBC_2.0)[1]</td>
<td>strerror(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>_obstack_begin(GLIBC_2.0)[1]</td>
<td>isdigit(GLIBC_2.0)[1]</td>
<td>strlen(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>_obstack_newchunk(GLIBC_2.0)[1]</td>
<td>isgraph(GLIBC_2.0)[1]</td>
<td>strncasecmp(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>_tolower(GLIBC_2.0)[1]</td>
<td>isprint(GLIBC_2.0)[1]</td>
<td>strncmp(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>_toupper(GLIBC_2.0)[1]</td>
<td>isspace(GLIBC_2.0)[1]</td>
<td>strncpy(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>a64l(GLIBC_2.0)[1]</td>
<td>isupper(GLIBC_2.0)[1]</td>
<td>strndup(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>abort(GLIBC_2.0)[1]</td>
<td>iswalpha(GLIBC_2.0)[1]</td>
<td>strpos(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>abs(GLIBC_2.0)[1]</td>
<td>iswblank(GLIBC_2.0)[1]</td>
<td>strftime(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>accept(GLIBC_2.0)[1]</td>
<td>iswcntrl(GLIBC_2.0)[1]</td>
<td>strftime(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>access(GLIBC_2.0)[1]</td>
<td>iswctype(GLIBC_2.0)[1]</td>
<td>strsep(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>acct(GLIBC_2.0)[1]</td>
<td>iswdigit(GLIBC_2.0)[1]</td>
<td>strsignal(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>adjtime(GLIBC_2.0)[1]</td>
<td>iswgraph(GLIBC_2.0)[1]</td>
<td>strspn(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>alarm(GLIBC_2.0)[1]</td>
<td>iswlower(GLIBC_2.0)[1]</td>
<td>strtok(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>asctime(GLIBC_2.0)[1]</td>
<td>iswupper(GLIBC_2.0)[1]</td>
<td>strtok(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>asctime_r(GLIBC_2.0)[1]</td>
<td>iswpunct(GLIBC_2.0)[1]</td>
<td>strtol(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>asprintf(GLIBC_2.0)[1]</td>
<td>iswxdigit(GLIBC_2.0)[1]</td>
<td>strtold(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>atol(GLIBC_2.0)[1]</td>
<td>isxdigit(GLIBC_2.0)[1]</td>
<td>strtol(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>atoll[1]</td>
<td>isxdigit()[1]</td>
<td>strtol()[1]</td>
</tr>
<tr>
<td>authnone_create(GLIBC_2.0)[1]</td>
<td>jrand48(GLIBC_2.0)[1]</td>
<td>strtol(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>basename(GLIBC_2.0)[1]</td>
<td>key_decryptsession(GLIBC_2.0)[1]</td>
<td>strtoq(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>bcmp(GLIBC_2.0)[1]</td>
<td>kill(GLIBC_2.0)[1]</td>
<td>strtoull(GLIBC_2.0)[1]</td>
</tr>
</tbody>
</table>
## Appendix A. Alphabetical Listing of Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>bcopy(GLIBC_2.0)</td>
<td>killpg(GLIBC_2.0)</td>
<td>strtoull(GLIBC_2.0)</td>
</tr>
<tr>
<td>bind(GLIBC_2.0)</td>
<td>l64a(GLIBC_2.0)</td>
<td>strtoumax(GLIBC_2.0)</td>
</tr>
<tr>
<td>bind_textdomain_codeset</td>
<td>labs()</td>
<td>strtouq()</td>
</tr>
<tr>
<td>bindresvport(GLIBC_2.0)</td>
<td>lchown(GLIBC_2.0)</td>
<td>strverscmp(GLIBC_2.0)</td>
</tr>
<tr>
<td>bindtextdomain(GLIBC_2.0)</td>
<td>lcong48(GLIBC_2.0)</td>
<td>strxfrm(GLIBC_2.0)</td>
</tr>
<tr>
<td>brk(GLIBC_2.0)</td>
<td>ldiv(GLIBC_2.0)</td>
<td>svc_getreqset(GLIBC_2.0)</td>
</tr>
<tr>
<td>bsd_signal(GLIBC_2.0)</td>
<td>lfind(GLIBC_2.0)</td>
<td>svc_register(GLIBC_2.0)</td>
</tr>
<tr>
<td>bsearch(GLIBC_2.0)</td>
<td>link(GLIBC_2.0)</td>
<td>svc_run(GLIBC_2.0)</td>
</tr>
<tr>
<td>btiowc(GLIBC_2.0)</td>
<td>listen(GLIBC_2.0)</td>
<td>svc_sendreply(GLIBC_2.0)</td>
</tr>
<tr>
<td>bzero(GLIBC_2.0)</td>
<td>llabs(GLIBC_2.0)</td>
<td>svcerr_auth(GLIBC_2.0)</td>
</tr>
<tr>
<td>calloc(GLIBC_2.0)</td>
<td>lldiv(GLIBC_2.0)</td>
<td>svcerr_decode(GLIBC_2.0)</td>
</tr>
<tr>
<td>catclose(GLIBC_2.0)</td>
<td>localeconv(GLIBC_2.0)</td>
<td>svcerr_nooproc(GLIBC_2.0)</td>
</tr>
<tr>
<td>catopen(GLIBC_2.0)</td>
<td>localtime(GLIBC_2.0)</td>
<td>svcerr_nooprog(GLIBC_2.0)</td>
</tr>
<tr>
<td>catgets(GLIBC_2.0)</td>
<td>localtime_r(GLIBC_2.0)</td>
<td>svcerr_progvers(GLIBC_2.0)</td>
</tr>
<tr>
<td>cfgetispeed(GLIBC_2.0)</td>
<td>lockf(GLIBC_2.0)</td>
<td>svcerr_systemerr(GLIBC_2.0)</td>
</tr>
<tr>
<td>cfgetospeed(GLIBC_2.0)</td>
<td>lockf64(GLIBC_2.0)</td>
<td>svcerr_weakauth(GLIBC_2.0)</td>
</tr>
<tr>
<td>cfmakeraw(GLIBC_2.0)</td>
<td>longjmp(GLIBC_2.0)</td>
<td>svctcp_create(GLIBC_2.0)</td>
</tr>
<tr>
<td>cfsetispeed(GLIBC_2.0)</td>
<td>lrand48(GLIBC_2.0)</td>
<td>svcudp_create(GLIBC_2.0)</td>
</tr>
<tr>
<td>cfsetospeed(GLIBC_2.0)</td>
<td>lseek(GLIBC_2.0)</td>
<td>swab(GLIBC_2.0)</td>
</tr>
<tr>
<td>cfsetspeed(GLIBC_2.0)</td>
<td>lseek64(GLIBC_2.0)</td>
<td>swapcontext(GLIBC_2.0)</td>
</tr>
<tr>
<td>cfsetvspeed(GLIBC_2.0)</td>
<td>makecontext(GLIBC_2.0)</td>
<td>swprintf(GLIBC_2.0)</td>
</tr>
<tr>
<td>chmod(GLIBC_2.0)</td>
<td>malloc(GLIBC_2.0)</td>
<td>syscall(GLIBC_2.0)</td>
</tr>
<tr>
<td>chdir(GLIBC_2.0)</td>
<td>mblen(GLIBC_2.0)</td>
<td>sync(GLIBC_2.0)</td>
</tr>
<tr>
<td>chroot(GLIBC_2.0)</td>
<td>mbrlen(GLIBC_2.0)</td>
<td>syncf(GLIBC_2.0)</td>
</tr>
<tr>
<td>clearerr(GLIBC_2.0)</td>
<td>mbslen(GLIBC_2.0)</td>
<td>syslog(GLIBC_2.0)</td>
</tr>
<tr>
<td>clnt_create(GLIBC_2.0)</td>
<td>mbrtowc(GLIBC_2.0)</td>
<td>syslog(GLIBC_2.0)</td>
</tr>
<tr>
<td>clnt_pcreateerror(GLIBC_2.0)</td>
<td>mbsinit(GLIBC_2.0)</td>
<td>system(GLIBC_2.0)</td>
</tr>
<tr>
<td>clnt_permno(GLIBC_2.0)</td>
<td>mbsrtowcs(GLIBC_2.0)</td>
<td>tcdrain(GLIBC_2.0)</td>
</tr>
<tr>
<td>clnt_perror(GLIBC_2.0)</td>
<td>mbsrtowcs(GLIBC_2.0)</td>
<td>tcflow(GLIBC_2.0)</td>
</tr>
<tr>
<td>clnt_spcreateerror(GLIBC_2.0)</td>
<td>mbstowcs(GLIBC_2.0)</td>
<td>tcflush(GLIBC_2.0)</td>
</tr>
<tr>
<td>clnt_sperrno(GLIBC_2.0)</td>
<td>mbtowc(GLIBC_2.0)</td>
<td>tcgetattr(GLIBC_2.0)</td>
</tr>
<tr>
<td>Function</td>
<td>Version</td>
<td>Function</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>clnt_sperror</td>
<td>GLIBC_2.0</td>
<td>memccpy</td>
</tr>
<tr>
<td>clock</td>
<td>GLIBC_2.0</td>
<td>memchr</td>
</tr>
<tr>
<td>close</td>
<td>GLIBC_2.0</td>
<td>memcmp</td>
</tr>
<tr>
<td>closedir</td>
<td>GLIBC_2.0</td>
<td>memcpy</td>
</tr>
<tr>
<td>closelog</td>
<td>GLIBC_2.0</td>
<td>memmove</td>
</tr>
<tr>
<td>confstr</td>
<td>GLIBC_2.0</td>
<td>memmove</td>
</tr>
<tr>
<td>connect</td>
<td>GLIBC_2.0</td>
<td>memrchr</td>
</tr>
<tr>
<td>creat</td>
<td>GLIBC_2.0</td>
<td>memset</td>
</tr>
<tr>
<td>creat64</td>
<td>GLIBC_2.1</td>
<td>mkdir</td>
</tr>
<tr>
<td>ctermid</td>
<td>GLIBC_2.0</td>
<td>mkfifo</td>
</tr>
<tr>
<td>ctime</td>
<td>GLIBC_2.0</td>
<td>mkstemp</td>
</tr>
<tr>
<td>ctime_r</td>
<td>GLIBC_2.0</td>
<td>mkstemp64</td>
</tr>
<tr>
<td>cuserid</td>
<td>GLIBC_2.0</td>
<td>mktemp</td>
</tr>
<tr>
<td>daemon</td>
<td>GLIBC_2.0</td>
<td>mktemp64</td>
</tr>
<tr>
<td>dcgettext</td>
<td>GLIBC_2.0</td>
<td>mlock</td>
</tr>
<tr>
<td>dcgettext</td>
<td>GLIBC_2.0</td>
<td>mlockall</td>
</tr>
<tr>
<td>difftime</td>
<td>GLIBC_2.0</td>
<td>malloc</td>
</tr>
<tr>
<td>dirname</td>
<td>GLIBC_2.0</td>
<td>mprotect</td>
</tr>
<tr>
<td>div</td>
<td>GLIBC_2.0</td>
<td>mrand48</td>
</tr>
<tr>
<td>dngettext</td>
<td>GLIBC_2.0</td>
<td>msgctl</td>
</tr>
<tr>
<td>drand48</td>
<td>GLIBC_2.0</td>
<td>msgget</td>
</tr>
<tr>
<td>dup</td>
<td>GLIBC_2.0</td>
<td>msgsrecv</td>
</tr>
<tr>
<td>dup2</td>
<td>GLIBC_2.0</td>
<td>msgsnd</td>
</tr>
<tr>
<td>ecvt</td>
<td>GLIBC_2.0</td>
<td>msync</td>
</tr>
<tr>
<td>endgrent</td>
<td>GLIBC_2.0</td>
<td>munlock</td>
</tr>
<tr>
<td>endnetent</td>
<td>GLIBC_2.0</td>
<td>munlockall</td>
</tr>
<tr>
<td>endprotoent</td>
<td>GLIBC_2.0</td>
<td>munmap</td>
</tr>
<tr>
<td>endpwent</td>
<td>GLIBC_2.0</td>
<td>nanosleep</td>
</tr>
<tr>
<td>endserverent</td>
<td>GLIBC_2.0</td>
<td>nftw</td>
</tr>
<tr>
<td>endutent</td>
<td>GLIBC_2.0</td>
<td>nftw64</td>
</tr>
</tbody>
</table>
### Appendix A. Alphabetical Listing of Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>endutxent(GLIBC_2.1)</td>
<td>ngettext(GLIBC_2.0)</td>
</tr>
<tr>
<td>erand48(GLIBC_2.0)</td>
<td><code>nice(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>eran</code></td>
<td><code>nl_langinfo(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>error(GLIBC_2.0)</code></td>
<td><code>nrand48(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>errorx(GLIBC_2.0)</code></td>
<td><code>ntohl(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>execl(GLIBC_2.0)</code></td>
<td><code>ntohs(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>execle(GLIBC_2.0)</code></td>
<td><code>obstack_free(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>execvp(GLIBC_2.0)</code></td>
<td><code>open(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>execve(GLIBC_2.0)</code></td>
<td><code>open64(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>execvep(GLIBC_2.0)</code></td>
<td><code>opendir(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>execvp(GLIBC_2.0)</code></td>
<td><code>openlog(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>exit(GLIBC_2.0)</code></td>
<td><code>pathconf(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>fchdir(GLIBC_2.0)</code></td>
<td><code>pause(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>fchown(GLIBC_2.0)</code></td>
<td><code>pclose(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>fclose(GLIBC_2.0)</code></td>
<td><code>perror(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>fcntl(GLIBC_2.0)</code></td>
<td><code>perror(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>fflush(GLIBC_2.0)</code></td>
<td><code>putenv(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>fflush_unlocked(GLIBC_2.0)</code></td>
<td><code>putenv(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>ffs(GLIBC_2.0)</code></td>
<td><code>pmap_getport(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>fgets(GLIBC_2.0)</code></td>
<td><code>pmap_set(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>fgetwc(GLIBC_2.2)</code></td>
<td><code>pmap_unset(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>fgetwc_unlocked(GLIBC_2.2)</code></td>
<td><code>putenv(GLIBC_2.2)</code></td>
</tr>
<tr>
<td><code>fgetwc</code></td>
<td><code>putenv(GLIBC_2.2)</code></td>
</tr>
<tr>
<td><code>fflush_unlocked(GLIBC_2.2)</code></td>
<td><code>puts(GLIBC_2.2)</code></td>
</tr>
<tr>
<td>Function</td>
<td>Version</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
</tr>
<tr>
<td>fgetws(GLIBC_2.2)</td>
<td>[1]</td>
</tr>
<tr>
<td>fileno(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>flock(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>flockfile(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>fmtmsg(GLIBC_2.1)</td>
<td>[1]</td>
</tr>
<tr>
<td>fnmatch(GLIBC_2.2.3)</td>
<td>[1]</td>
</tr>
<tr>
<td>fopen(GLIBC_2.1)</td>
<td>[1]</td>
</tr>
<tr>
<td>fopen64(GLIBC_2.1)</td>
<td>[1]</td>
</tr>
<tr>
<td>fork(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>fpathconf(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>fprintf(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>fputc(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>fputs(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>fputwc(GLIBC_2.2)</td>
<td>[1]</td>
</tr>
<tr>
<td>fputws(GLIBC_2.2)</td>
<td>[1]</td>
</tr>
<tr>
<td>fscanf(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>fseek(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>fseeko(GLIBC_2.1)</td>
<td>[1]</td>
</tr>
<tr>
<td>fseeko64(GLIBC_2.1)</td>
<td>[1]</td>
</tr>
<tr>
<td>fsetpos(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>fsetpos64(GLIBC_2.1)</td>
<td>[1]</td>
</tr>
<tr>
<td>fstatvfs(GLIBC_2.1)</td>
<td>[1]</td>
</tr>
<tr>
<td>fstatvfs64(GLIBC_2.1)</td>
<td>[1]</td>
</tr>
<tr>
<td>fsync(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>ftell(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>ftell64(GLIBC_2.0)</td>
<td>[1]</td>
</tr>
<tr>
<td>ftello(GLIBC_2.1)</td>
<td>[1]</td>
</tr>
</tbody>
</table>
### Appendix A. Alphabetical Listing of Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ftello64(GLIBC_2.1)[1]</td>
<td>rmdir(GLIBC_2.1)[1]</td>
<td>wcstold(GLIBC_2.1)[1]</td>
</tr>
<tr>
<td>fttime(GLIBC_2.0)[1]</td>
<td>sbbrk(GLIBC_2.0)[1]</td>
<td>wcstoll(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>ftok(GLIBC_2.0)[1]</td>
<td>scanf(GLIBC_2.0)[1]</td>
<td>wcstombs(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>fttruncatel(GLIBC_2.0)[1]</td>
<td>sched_get_priority_max(GLIBC_2.0)[1]</td>
<td>wcstoq(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>fttruncatel64(GLIBC_2.1)[1]</td>
<td>sched_get_priority_min(GLIBC_2.1)[1]</td>
<td>wcstoul(GLIBC_2.1)[1]</td>
</tr>
<tr>
<td>fttrylockfile(GLIBC_2.0)[1]</td>
<td>sched_getparam(GLIBC_2.0)[1]</td>
<td>wcstoull(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>ftw(GLIBC_2.0)[1]</td>
<td>sched_getscheduler(GLIBC_2.0)[1]</td>
<td>wcstoumax(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>ftw64(GLIBC_2.1)[1]</td>
<td>sched_rr_get_interval(GLIBC_2.1)[1]</td>
<td>wcstouq(GLIBC_2.2)[1]</td>
</tr>
<tr>
<td>funlockfile(GLIBC_2.0)[1]</td>
<td>sched_getscheduler(GLIBC_2.1)[1]</td>
<td>wcswcs(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>fwide(GLIBC_2.2)[1]</td>
<td>sched_setscheduler(GLIBC_2.2)[1]</td>
<td>wcswidth(GLIBC_2.2)[1]</td>
</tr>
<tr>
<td>fwrite(GLIBC_2.2)[1]</td>
<td>sched_yield(GLIBC_2.2)[1]</td>
<td>wcxfrm(GLIBC_2.2)[1]</td>
</tr>
<tr>
<td>fwrite(GLIBC_2.0)[1]</td>
<td>seed48(GLIBC_2.0)[1]</td>
<td>wctob(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>fwrite(GLIBC_2.0)[1]</td>
<td>seekdir(GLIBC_2.2)[1]</td>
<td>wctomb(GLIBC_2.2)[1]</td>
</tr>
<tr>
<td>gcvt(GLIBC_2.0)[1]</td>
<td>selectd(GLIBC_2.0)[1]</td>
<td>wctrans(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getaddrinfo[1]</td>
<td>semctl(GLIBC_2.0)[1]</td>
<td>wctype(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getc(GLIBC_2.0)[1]</td>
<td>semop(GLIBC_2.0)[1]</td>
<td>wmemchr(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>geteck_unlocked(GLIBC_2.0)[1]</td>
<td>send(GLIBC_2.0)[1]</td>
<td>wmemcmp(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>geteck(GLIBC_2.0)[1]</td>
<td>sendmsg(GLIBC_2.0)[1]</td>
<td>wmemcpy(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>geteck_unlocked(GLIBC_2.0)[1]</td>
<td>sendto(GLIBC_2.0)[1]</td>
<td>wmemmove(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getcontext(GLIBC_2.1)[1]</td>
<td>setbuf(GLIBC_2.1)[1]</td>
<td>wmemset(GLIBC_2.1)[1]</td>
</tr>
<tr>
<td>getcwd(GLIBC_2.0)[1]</td>
<td>setbuffer(GLIBC_2.0)[1]</td>
<td>wordexp(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getdate(GLIBC_2.1)[1]</td>
<td>setcontext(GLIBC_2.1)[1]</td>
<td>wordfree(GLIBC_2.1)[1]</td>
</tr>
<tr>
<td>getdomainname(GLIBC_2.0)[1]</td>
<td>setdomainname[1]</td>
<td>wprintf(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getegid(GLIBC_2.0)[1]</td>
<td>setegid(GLIBC_2.0)[1]</td>
<td>write(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>geteuid(GLIBC_2.0)[1]</td>
<td>seteuid(GLIBC_2.0)[1]</td>
<td>writev(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getgid(GLIBC_2.0)[1]</td>
<td>setgid(GLIBC_2.0)[1]</td>
<td>wscanf(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getgid(GLIBC_2.0)[1]</td>
<td>xdr_accepted_reply(GLIBC_2.0)[1]</td>
<td>xdr_array(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>gettext(GLIBC_2.0)[1]</td>
<td>setgetent(GLIBC_2.0)[1]</td>
<td>xdr_array(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>Function</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>getgrgid(GLIBC_2.0)[1]</td>
<td>setgroups(GLIBC_2.0)[1]</td>
<td>xdr_bool(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getgrgid_r(GLIBC_2.0)[1]</td>
<td>sethostid(GLIBC_2.0)[1]</td>
<td>xdr_bytes(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getgrnam(GLIBC_2.0)[1]</td>
<td>sethostname(GLIBC_2.0)[1]</td>
<td>xdr_callhdr(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getgrnam_r(GLIBC_2.0)[1]</td>
<td>settimert(GLIBC_2.0)[1]</td>
<td>xdr_callmsg(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getgroups(GLIBC_2.0)[1]</td>
<td>setlocale(GLIBC_2.0)[1]</td>
<td>xdr_char(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>gethostbyaddr(GLIBC_2.0)[1]</td>
<td>setlogmask(GLIBC_2.0)[1]</td>
<td>xdr_double(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>gethostbyname(GLIBC_2.0)[1]</td>
<td>setnetent(GLIBC_2.0)[1]</td>
<td>xdr_enum(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>gethostid(GLIBC_2.0)[1]</td>
<td>setpgid(GLIBC_2.0)[1]</td>
<td>xdr_float(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>gethostname(GLIBC_2.0)[1]</td>
<td>setpgrp(GLIBC_2.0)[1]</td>
<td>xdr_free(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getitimer(GLIBC_2.0)[1]</td>
<td>setpriority(GLIBC_2.0)[1]</td>
<td>xdr_int(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getloadavg(GLIBC_2.2)[1]</td>
<td>setprotoent(GLIBC_2.2)[1]</td>
<td>xdr_long(GLIBC_2.2)[1]</td>
</tr>
<tr>
<td>getlogin(GLIBC_2.0)[1]</td>
<td>setpwent(GLIBC_2.0)[1]</td>
<td>xdrOpaque(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getnetbyaddr(GLIBC_2.0)[1]</td>
<td>setregid()[1]</td>
<td>xdrOpaque_auth()[1]</td>
</tr>
<tr>
<td>getopt(GLIBC_2.0)[1]</td>
<td>setteruid(GLIBC_2.0)[1]</td>
<td>xdr_pointer(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getopt_long(GLIBC_2.0)[1]</td>
<td>setrlimit(GLIBC_2.0)[1]</td>
<td>xdr_reference(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getopt_long_only(GLIBC_2.0)[1]</td>
<td>setrlimit64[1]</td>
<td>xdr_rejected_reply(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getpagesize(GLIBC_2.0)[1]</td>
<td>setservent(GLIBC_2.0)[1]</td>
<td>xdr_replymsg(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getpeername(GLIBC_2.0)[1]</td>
<td>setsockopt(GLIBC_2.0)[1]</td>
<td>xdr_short(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getpgid(GLIBC_2.0)[1]</td>
<td>setstate(GLIBC_2.0)[1]</td>
<td>xdr_u_char(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getppid(GLIBC_2.0)[1]</td>
<td>settid(GLIBC_2.0)[1]</td>
<td>xdr_u_int(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getppid(GLIBC_2.0)[1]</td>
<td>setutent(GLIBC_2.0)[1]</td>
<td>xdr_u_long(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getpriority(GLIBC_2.0)[1]</td>
<td>setuxtent(GLIBC_2.0)[1]</td>
<td>xdr_u_short(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getpriority(GLIBC_2.0)[1]</td>
<td>setvbuf(GLIBC_2.0)[1]</td>
<td>xdr_union(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getprotobyname(GLIBC_2.0)[1]</td>
<td>shmat(GLIBC_2.0)[1]</td>
<td>xdr_vector(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getprotobynumber(GLIBC_2.0)[1]</td>
<td>shmatl(GLIBC_2.0)[1]</td>
<td>xdr_void(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getproootent(GLIBC_2.0)[1]</td>
<td>shmdt(GLIBC_2.0)[1]</td>
<td>xdr_wrapstring(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getpwent(GLIBC_2.0)[1]</td>
<td>shmget(GLIBC_2.0)[1]</td>
<td>xdrmem_create(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getpwnam(GLIBC_2.0)[1]</td>
<td>shutdown(GLIBC_2.0)[1]</td>
<td>xdrrec_create(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getpwnam_r(GLIBC_2.0)[1]</td>
<td>sigaction(GLIBC_2.0)[1]</td>
<td>xdrrec_eof(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>getpwuid(GLIBC_2.0)[1]</td>
<td>sigaddset(GLIBC_2.0)[1]</td>
<td></td>
</tr>
</tbody>
</table>
Appendix A. Alphabetical Listing of Interfaces

| getpwuid_r(GLIBC_2.0)[1] | sigaltstack(GLIBC_2.0)[1] |

**Table A-2. libc Data Interfaces**

| __daylightIDL_STD_46_LSB | __timezoneIDL_STD_46_LSB | __sys_errlistIDL_STD_46_LSB |
| _environIDL_STD_46_LSB | __tznameIDL_STD_46_LSB |

**A.2. libcrypt**

The behaviour of the interfaces in this library is specified by the following Standards.


**Table A-3. libcrypt Function Interfaces**

| crypt(GLIBC_2.0)[1] | encrypt(GLIBC_2.0)[1] | setkey(GLIBC_2.0)[1] |

**A.3. libdl**

The behaviour of the interfaces in this library is specified by the following Standards.

this specification


**Table A-4. libdl Function Interfaces**

| dladdr(GLIBC_2.0)[1] | dlerror(GLIBC_2.0)[1] | dlerror(GLIBC_2.0)[1] |
| dclose(GLIBC_2.0)[1] | dlopen(GLIBC_2.0)[1] |

**A.4. libm**

The behaviour of the interfaces in this library is specified by the following Standards.

ISO C (1999)

SUSv2


**Table A-5. libm Function Interfaces**

<p>| acos(GLIBC_2.0)[1] | csinh(GLIBC_2.0)[1] | log(GLIBC_2.0)[1] |
| acosf(GLIBC_2.0)[1] | csinl(GLIBC_2.0)[1] | log10(GLIBC_2.0)[1] |
| acosh(GLIBC_2.0)[1] | csqrt(GLIBC_2.0)[1] | log10f[1] |
| acoshf(GLIBC_2.0)[1] | csqrtf(GLIBC_2.0)[1] | log10l[1] |
| acoshl(GLIBC_2.0)[1] | csqrtl(GLIBC_2.0)[1] | log1p(GLIBC_2.0)[1] |
| acosl(GLIBC_2.0)[1] | ctan(GLIBC_2.0)[1] | logb(GLIBC_2.0)[1] |</p>
<table>
<thead>
<tr>
<th>func</th>
<th>func</th>
<th>func</th>
</tr>
</thead>
<tbody>
<tr>
<td>asin(GLIBC_2.0)</td>
<td>ctanf(GLIBC_2.0)</td>
<td>logf[1]</td>
</tr>
<tr>
<td>asinf(GLIBC_2.0)</td>
<td>ctanh(GLIBC_2.0)</td>
<td>logl[1]</td>
</tr>
<tr>
<td>asinh(GLIBC_2.0)</td>
<td>ctanhf(GLIBC_2.0)</td>
<td>lrint(GLIBC_2.0)</td>
</tr>
<tr>
<td>asinhf(GLIBC_2.0)</td>
<td>ctanhl(GLIBC_2.0)</td>
<td>lrint(GLIBC_2.0)</td>
</tr>
<tr>
<td>asinhl(GLIBC_2.0)</td>
<td>ctanl(GLIBC_2.0)</td>
<td>lrintl(GLIBC_2.0)</td>
</tr>
<tr>
<td>asinl(GLIBC_2.0)</td>
<td>dremf(GLIBC_2.0)</td>
<td>lround(GLIBC_2.0)</td>
</tr>
<tr>
<td>atan(GLIBC_2.0)</td>
<td>dreml(GLIBC_2.0)</td>
<td>lroundf(GLIBC_2.0)</td>
</tr>
<tr>
<td>atan2(GLIBC_2.0)</td>
<td>erf(GLIBC_2.0)</td>
<td>lroundl(GLIBC_2.0)</td>
</tr>
<tr>
<td>atan2f(GLIBC_2.0)</td>
<td>erfc(GLIBC_2.0)</td>
<td>matherr(GLIBC_2.0)</td>
</tr>
<tr>
<td>atan2l(GLIBC_2.0)</td>
<td>erfcf(GLIBC_2.0)</td>
<td>modf(GLIBC_2.0)</td>
</tr>
<tr>
<td>atanf(GLIBC_2.0)</td>
<td>erfcf(GLIBC_2.0)</td>
<td>modff(GLIBC_2.0)</td>
</tr>
<tr>
<td>atanh(GLIBC_2.0)</td>
<td>erfl(GLIBC_2.0)</td>
<td>nan(GLIBC_2.0)</td>
</tr>
<tr>
<td>atanhf(GLIBC_2.0)</td>
<td>exp(GLIBC_2.0)</td>
<td>nanf(GLIBC_2.0)</td>
</tr>
<tr>
<td>atanhf(GLIBC_2.0)</td>
<td>expf(GLIBC_2.0)</td>
<td>nanl(GLIBC_2.0)</td>
</tr>
<tr>
<td>cabs(GLIBC_2.1)</td>
<td>expm1(GLIBC_2.1)</td>
<td>nextafter(GLIBC_2.1)</td>
</tr>
<tr>
<td>cabsf(GLIBC_2.1)</td>
<td>expm1f(GLIBC_2.1)</td>
<td>nextafterf(GLIBC_2.1)</td>
</tr>
<tr>
<td>cabsl(GLIBC_2.1)</td>
<td>expm1l(GLIBC_2.1)</td>
<td>nextafterl(GLIBC_2.1)</td>
</tr>
<tr>
<td>casin(GLIBC_2.1)</td>
<td>feclexcept(GLIBC_2.1)</td>
<td>nexttoward(GLIBC_2.1)</td>
</tr>
<tr>
<td>casinf(GLIBC_2.1)</td>
<td>feclexceptf(GLIBC_2.1)</td>
<td>nexttowardf(GLIBC_2.1)</td>
</tr>
<tr>
<td>casinh(GLIBC_2.1)</td>
<td>feclexceptl(GLIBC_2.1)</td>
<td>nexttowardl(GLIBC_2.1)</td>
</tr>
<tr>
<td>casinhf(GLIBC_2.1)</td>
<td>fegetenv(GLIBC_2.1)</td>
<td>pow(GLIBC_2.1)</td>
</tr>
<tr>
<td>carg(GLIBC_2.1)</td>
<td>fegetenvflag(GLIBC_2.1)</td>
<td>pow10(GLIBC_2.1)</td>
</tr>
<tr>
<td>cargf(GLIBC_2.1)</td>
<td>fegetenvflag(GLIBC_2.1)</td>
<td>pow10f(GLIBC_2.1)</td>
</tr>
<tr>
<td>cargl(GLIBC_2.1)</td>
<td>fegetround(GLIBC_2.1)</td>
<td>pow10f(GLIBC_2.1)</td>
</tr>
<tr>
<td>casin(GLIBC_2.1)</td>
<td>feholdexcept(GLIBC_2.1)</td>
<td>pow10l(GLIBC_2.1)</td>
</tr>
<tr>
<td>casinf(GLIBC_2.1)</td>
<td>feraiseexcept(GLIBC_2.1)</td>
<td>powf(GLIBC_2.1)</td>
</tr>
<tr>
<td>casinh(GLIBC_2.1)</td>
<td>fegetenv(GLIBC_2.1)</td>
<td>powl(GLIBC_2.1)</td>
</tr>
<tr>
<td>casinhf(GLIBC_2.1)</td>
<td>fegetexceptflag(GLIBC_2.1)</td>
<td>remainder(GLIBC_2.1)</td>
</tr>
</tbody>
</table>
### Appendix A. Alphabetical Listing of Interfaces

<table>
<thead>
<tr>
<th>casinh(GLIBC_2.1)</th>
<th>fesetround(GLIBC_2.1)</th>
<th>remainderf(GLIBC_2.1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>casin(GLIBC_2.1)</td>
<td>fetestexcept(GLIBC_2.1)</td>
<td>remainderl(GLIBC_2.1)</td>
</tr>
<tr>
<td>catan(GLIBC_2.1)</td>
<td>feupdateenv(GLIBC_2.1)</td>
<td>remquo(GLIBC_2.1)</td>
</tr>
<tr>
<td>catanf(GLIBC_2.1)</td>
<td>finitef(GLIBC_2.1)</td>
<td>remquof(GLIBC_2.1)</td>
</tr>
<tr>
<td>catanh(GLIBC_2.1)</td>
<td>finite(GLIBC_2.1)</td>
<td>remquol(GLIBC_2.1)</td>
</tr>
<tr>
<td>catanhf(GLIBC_2.1)</td>
<td>fmax(GLIBC_2.1)</td>
<td>round(GLIBC_2.0)</td>
</tr>
<tr>
<td>catanhl(GLIBC_2.1)</td>
<td>fma(GLIBC_2.0)</td>
<td>roundf(GLIBC_2.0)</td>
</tr>
<tr>
<td>catanl(GLIBC_2.1)</td>
<td>fmaf(GLIBC_2.0)</td>
<td>roundl(GLIBC_2.0)</td>
</tr>
<tr>
<td>cbrtf(GLIBC_2.0)</td>
<td>fmaxf(GLIBC_2.1)</td>
<td>scalb(GLIBC_2.1)</td>
</tr>
<tr>
<td>cbrtl(GLIBC_2.0)</td>
<td>fmaxl(GLIBC_2.1)</td>
<td>scalbl(GLIBC_2.1)</td>
</tr>
<tr>
<td>cbrt(GLIBC_2.0)</td>
<td>fmaxf(GLIBC_2.1)</td>
<td>scalb(GLIBC_2.1)</td>
</tr>
<tr>
<td>cbrtl(GLIBC_2.0)</td>
<td>fmaxl(GLIBC_2.1)</td>
<td>scalbl(GLIBC_2.1)</td>
</tr>
<tr>
<td>cbrtf(GLIBC_2.0)</td>
<td>fmaxf(GLIBC_2.1)</td>
<td>scalb(GLIBC_2.1)</td>
</tr>
<tr>
<td>cbrtl(GLIBC_2.0)</td>
<td>fmaxl(GLIBC_2.1)</td>
<td>scalbl(GLIBC_2.1)</td>
</tr>
<tr>
<td>cbrtf(GLIBC_2.0)</td>
<td>fmaxf(GLIBC_2.1)</td>
<td>scalb(GLIBC_2.1)</td>
</tr>
<tr>
<td>cbrtl(GLIBC_2.0)</td>
<td>fmaxl(GLIBC_2.1)</td>
<td>scalbl(GLIBC_2.1)</td>
</tr>
<tr>
<td>cbrtf(GLIBC_2.0)</td>
<td>fmaxf(GLIBC_2.1)</td>
<td>scalb(GLIBC_2.1)</td>
</tr>
<tr>
<td>cbrtl(GLIBC_2.0)</td>
<td>fmaxl(GLIBC_2.1)</td>
<td>scalbl(GLIBC_2.1)</td>
</tr>
<tr>
<td>ceill(GLIBC_2.0)</td>
<td>fmaxl(GLIBC_2.1)</td>
<td>scalblnl(GLIBC_2.0)</td>
</tr>
<tr>
<td>ceilf(GLIBC_2.0)</td>
<td>fmod(GLIBC_2.0)</td>
<td>scalbnf(GLIBC_2.1)</td>
</tr>
<tr>
<td>ceilf(GLIBC_2.0)</td>
<td>fmodf(GLIBC_2.1)</td>
<td>significand(GLIBC_2.1)</td>
</tr>
<tr>
<td>ceilf(GLIBC_2.0)</td>
<td>fmodf(GLIBC_2.1)</td>
<td>significandf(GLIBC_2.1)</td>
</tr>
<tr>
<td>cexp(GLIBC_2.1)</td>
<td>frexp(GLIBC_2.1)</td>
<td>sin(GLIBC_2.1)</td>
</tr>
<tr>
<td>cexpf(GLIBC_2.1)</td>
<td>frexpf(GLIBC_2.1)</td>
<td>sincos(GLIBC_2.1)</td>
</tr>
<tr>
<td>cexpl(GLIBC_2.1)</td>
<td>frexpl(GLIBC_2.1)</td>
<td>sincosf(GLIBC_2.1)</td>
</tr>
<tr>
<td>cimag(GLIBC_2.1)</td>
<td>gamma(GLIBC_2.1)</td>
<td>sincosf(GLIBC_2.1)</td>
</tr>
<tr>
<td>cimagf(GLIBC_2.1)</td>
<td>gammaf(GLIBC_2.1)</td>
<td>sincosf(GLIBC_2.1)</td>
</tr>
<tr>
<td>cimagl(GLIBC_2.1)</td>
<td>gammaf(GLIBC_2.1)</td>
<td>sincosf(GLIBC_2.1)</td>
</tr>
<tr>
<td>clog(GLIBC_2.1)</td>
<td>hypot(GLIBC_2.1)</td>
<td>sinh(GLIBC_2.1)</td>
</tr>
<tr>
<td>clog10(GLIBC_2.1)</td>
<td>hypotf(GLIBC_2.1)</td>
<td>sinh(GLIBC_2.1)</td>
</tr>
<tr>
<td>clog10f(GLIBC_2.1)</td>
<td>hypotf(GLIBC_2.1)</td>
<td>sinh(GLIBC_2.1)</td>
</tr>
<tr>
<td>clog10l(GLIBC_2.1)</td>
<td>hypotl(GLIBC_2.1)</td>
<td>sinh(GLIBC_2.1)</td>
</tr>
<tr>
<td>clogf(GLIBC_2.1)</td>
<td>ilogb(GLIBC_2.1)</td>
<td>sinh(GLIBC_2.1)</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>clogl(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>ilogbf(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>conj(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>ilogbl(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>conjf(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>j0(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>conjl(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>j0f(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>copysign(GLIBC_2.0)</code></td>
<td>GLIBC_2.0</td>
<td><code>j0l(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>copysignf(GLIBC_2.0)</code></td>
<td>GLIBC_2.0</td>
<td><code>j1(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>copysignl(GLIBC_2.0)</code></td>
<td>GLIBC_2.0</td>
<td><code>j1f(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>cos(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>j1l(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>cosf(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>jn(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>cosh(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>jnl(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>coshf(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>ldexp(GLIBC_2.0)</code></td>
</tr>
<tr>
<td><code>coshl(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>ldexpl(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>cproj(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>lgamma(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>cprojf(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>lgammaf(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>cprojl(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>lgammalr(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>creal(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>lgammal(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>crealf(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>llrintf(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>creall(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>llrintl(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>csin(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>llrintf(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>csinf(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>llrintl(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>csinh(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>llroundf(GLIBC_2.1)</code></td>
</tr>
<tr>
<td><code>csinhf(GLIBC_2.1)</code></td>
<td>GLIBC_2.1</td>
<td><code>llroundl(GLIBC_2.1)</code></td>
</tr>
</tbody>
</table>

**Table A-6. libm Data Interfaces**

<table>
<thead>
<tr>
<th>Function</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>signgam</td>
<td>ID_STD_46_SUSV3</td>
</tr>
</tbody>
</table>
### A.5. libncurses

The behaviour of the interfaces in this library is specified by the following Standards.

**X/Open Curses**

Table A-7. libncurses Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Function</td>
<td>Function</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------</td>
<td></td>
</tr>
<tr>
<td>hline[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>prefresh[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wcursyncup[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>idcok[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>printw[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wdelch[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>idlok[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>putp[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wdelete[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>immedok[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>putwin[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wechochar[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>inch[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>qiflush[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>werase[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>inchnstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>raw[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wget[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>inchstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>redrawwin[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wgetnstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>init_color[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>refresh[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wgetstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>init_pair[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>reset_prog_mode[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>whline[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>initscr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>reset_shell_mode[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winch[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>innstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>resetty[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winchstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>insch[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>restartterm[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winchstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>insdelln[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ripoffline[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winnstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>insertln[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>savetty[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winsch[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>insnstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>scanw[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winsdelln[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>insstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>scr_dump[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winsertln[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>instr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>scr_init[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winsnstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>intrflush[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>scr_restore[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winsstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>is_linetouched[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>scr_set[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>winstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>is_wintouched[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>scr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wmove[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>isendwin[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>scroll[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wnoutrefresh[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>keyname[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>scrolllok[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wprint[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>keypad[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>set_curterm[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wredraw[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>killchar[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>set_term[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wrefresh[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>leaveok[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>setscrreg[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wscanw[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>longname[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>setupterm[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wscrl[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>meta[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slk_attr_set[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wsset[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>move[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slk_attr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wstandend[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mvaddch[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slk_attr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wstandout[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mvaddchnstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slk_attrset[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wsyncdown[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mvaddchstr[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slk_clear[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wsycup[1]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table A-8. libncurses Data Interfaces

<table>
<thead>
<tr>
<th>COLORS.ID_STD_46_SUS_46_CURSES</th>
<th>LINES.ID_STD_46_SUS_46_CURSES</th>
<th>curscr.ID_STD_46_SUS_46_CURSES</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLOR_PAIRS.ID_STD_46_SUS_46_CURSES</td>
<td>acs_map.ID_STD_46_SUS_46_CURSES</td>
<td>stdscr.ID_STD_46_SUS_46_CURSES</td>
</tr>
<tr>
<td>COLS.ID_STD_46_SUS_46_CURSES</td>
<td>cur_term.ID_STD_46_SUS_46_CURSES</td>
<td></td>
</tr>
</tbody>
</table>

**A.6. libpam**

The behaviour of the interfaces in this library is specified by the following Standards.

Table A-9. libpam Function Interfaces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>pam_close_session[1]</td>
<td>pam_open_session[1]</td>
<td></td>
</tr>
<tr>
<td>pam_end[1]</td>
<td>pam_set_item[1]</td>
<td></td>
</tr>
</tbody>
</table>

**A.7. libpthread**

The behaviour of the interfaces in this library is specified by the following Standards.

Large File Support

Table A-10. libpthread Function Interfaces

<table>
<thead>
<tr>
<th>_pthread_cleanup_pop[1]</th>
<th>pthread_create()[1]</th>
<th>pthread_rwlock_trywrlock()[1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>_pthread_cleanup_push[1]</td>
<td>pthread_detach()[1]</td>
<td>pthread_rwlock_unlock()[1]</td>
</tr>
<tr>
<td>pread(GLIBC_2.1)[1]</td>
<td>pthread_equal(GLIBC_2.1)[1]</td>
<td>pthread_rwlock_wrlock(GLIBC_2.1)[1]</td>
</tr>
</tbody>
</table>
### Appendix A. Alphabetical Listing of Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>GLIBC Version</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pread64</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthreads</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_rwlockattr_destroy</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_attr_destroy</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_getspecific</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_rwlockattr_getpshared</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_attr_getdetachstate</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_join</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_rwlockattr_init</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_attr_getguardsize</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_getguardsize</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_attr_getschedparam</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_getschedparam</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_attr_getstackaddr</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_attr_getstacksize</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_attr_init</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_getspecific</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_key_create</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_key_delete</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_self</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_attr_getstackaddr</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_attr_getstacksize</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_attr_init</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutex_destroy</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutex_init</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutex_lock</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutex_unlock</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutex_trylock</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutexattr_destroy</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutexattr_getpshared</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutexattr_getguardsize</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutexattr_init</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutexattr_setpshared</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutexattr_setguardsize</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutexattr_settype</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_setcancelstate</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_setcanceltype</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_setconcurrency</code></td>
<td>GLIBC 2.0</td>
<td></td>
</tr>
<tr>
<td><code>pthread_setspecific</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_sigmask</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_testcancel</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_rwlockattr_destroy</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_rwlockattr_getpshared</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_rwlockattr_init</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_rwlockattr_setpshared</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pwrite</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pwrite64</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_close</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_destroy</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_getvalue</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_init</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_open</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_post</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_timedwait</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_timedwait</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_cancel</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>pthread_mutexattr_gettype</code></td>
<td>GLIBC 2.1</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_close</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_destroy</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_getvalue</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_init</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_open</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
<tr>
<td><code>sem_post</code></td>
<td>GLIBC 2.0</td>
<td>[1]</td>
</tr>
</tbody>
</table>
Appendix A. Alphabetical Listing of Interfaces

<table>
<thead>
<tr>
<th>pthread_cond_wait(GLIBC_2.0)[1]</th>
<th>pthread_rwlock_init(GLIBC_2.0)[1]</th>
<th>sem_timedwait(GLIBC_2.0)[1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>pthread_condattr_destroy(GLIBC_2.0)[1]</td>
<td>pthread_rwlock_rdlock(GLIBC_2.0)[1]</td>
<td>sem_trywait(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>pthread_condattr_init(GLIBC_2.0)[1]</td>
<td>pthread_rwlock_timedwrlock[1]</td>
<td>sem_wait(GLIBC_2.0)[1]</td>
</tr>
<tr>
<td>pthread_condattr_setpshared[1]</td>
<td>pthread_rwlock_tryrdlock()[1]</td>
<td></td>
</tr>
</tbody>
</table>

A.8. libutil

The behaviour of the interfaces in this library is specified by the following Standards.

Table A-11. libutil Function Interfaces

<table>
<thead>
<tr>
<th>forkpty(GLIBC_2.0)[1]</th>
<th>login_tty(GLIBC_2.0)[1]</th>
<th>logwtmp(GLIBC_2.0)[1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>login(GLIBC_2.0)[1]</td>
<td>logout(GLIBC_2.0)[1]</td>
<td>openpty(GLIBC_2.0)[1]</td>
</tr>
</tbody>
</table>

A.9. libz

The behaviour of the interfaces in this library is specified by the following Standards.

Table A-12. libz Function Interfaces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>------------------------</td>
<td>------------</td>
<td>-----------</td>
</tr>
<tr>
<td>get_crc_table[1]</td>
<td>gzseek[1]</td>
<td></td>
</tr>
<tr>
<td>gzclose[1]</td>
<td>gzsetparams[1]</td>
<td></td>
</tr>
</tbody>
</table>
Linux Packaging Specification
Linux Packaging Specification
Table of Contents

I. Package Format and Installation

1. Software Installation
   1.1. Package File Format
   1.1.1. Lead Section
   1.1.2. Header Structure
       1.1.2.1. Header Record
       1.1.2.2. Index Record
           1.1.2.2.1. Index Type Values
           1.1.2.2.2. Index Tag Values
       1.1.2.3. Header Store
   1.1.3. Signature Section
   1.1.4. Header Section
       1.1.4.1. Package Information
       1.1.4.2. Installation Information
       1.1.4.3. File Information
       1.1.4.4. Dependency Information
           1.1.4.4.1. Package Dependency Values
           1.1.4.4.2. Package Dependencies Attributes
       1.1.4.5. Other Information
   1.1.5. Payload Section
   1.2. Package Script Restrictions
   1.3. Package Tools
   1.4. Package Naming
   1.5. Package Dependencies
   1.6. Package Architecture Considerations

II. Package Script Restrictions
List of Tables

1-1. RPM File Format.................................................................................................................. 1
1-2. Signature Format .................................................................................................................. 2
1-3. Index Type values.................................................................................................................. 3
1-4. Header Private Tag Values.................................................................................................... 4
1-5. Signature Tag Values ........................................................................................................... 5
1-6. Signature Digest Tag Values ................................................................................................ 6
1-7. Signature Signing Tag Values ............................................................................................... 6
1-8. Package Info Tag Values....................................................................................................... 7
1-9. Installation Tag Values.......................................................................................................... 9
1-10. File Info Tag Values ......................................................................................................... 10
1-11. Package Dependency Tag Values ..................................................................................... 12
1-12. Index Type values................................................................................................................ 14
1-13. Package Dependency Attributes ....................................................................................... 15
1-14. Other Tag Values ................................................................................................................ 15
1-15. CPIO File Format .............................................................................................................. 17
I. Package Format and Installation
Chapter 1. Software Installation

Applications shall either be packaged in the RPM packaging format as defined in this specification, or supply an installer which is LSB conforming (for example, calls LSB commands and utilities).\(^1\)

Distributions shall provide a mechanism for installing applications in this packaging format with some restrictions listed below.\(^2\)

1.1. Package File Format

An RPM format file consists of 4 sections, the Lead, Signature, Header, and the Payload. All values are stored in network byte order.

Table 1-1. RPM File Format

<table>
<thead>
<tr>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lead</td>
</tr>
<tr>
<td>Signature</td>
</tr>
<tr>
<td>Header</td>
</tr>
<tr>
<td>Payload</td>
</tr>
</tbody>
</table>

These 4 sections shall exist in the order specified.

The lead section is used to identify the package file.

The signature section is used to verify the integrity, and optionally, the authenticity of the majority of the package file.

The header section contains all available information about the package. Entries such as the package's name, version, and file list, are contained in the header.

The payload section holds the files to be install.

1.1.1. Lead Section

```c
struct rpmlead {
    unsigned char magic[4];
    unsigned char major, minor;
    short type;
    short archnum;
    char name[66];
    short osnum;
    short signature_type;
    char reserved[16];
} ;
```

*magic*

Value identifying this file as an RPM format file. This value shall be "\355\253\356\333".
Chapter 1. Software Installation

1.1.2. Header Structure

The Header structure is used for both the Signature and Header Sections. A Header Structure consists of 3 parts, a Header record, followed by 1 or more Index records, followed by 0 or more bytes of data associated with the Index records. A Header structure shall be aligned to an 8 byte boundary.

Table 1-2. Signature Format

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Header Record</td>
</tr>
<tr>
<td>Array of Index Records</td>
</tr>
<tr>
<td>Store of Index Values</td>
</tr>
</tbody>
</table>

1.1.2.1. Header Record

```c
struct rpmheader {
    unsigned char magic[4];
    unsigned char reserved[4];
    int nindex;
    int hsize;
};
```
Chapter 1. Software Installation

`magic`
Value identifying this record as an RPM header record. This value shall be "\216\255\350\001".

`reserved`
Reserved space. This value shall be "\000\000\000\000".

`nindex`
The number of Index Records that follow this Header Record. There should be at least 1 Index Record.

`hsize`
The size in bytes of the storage area for the data pointed to by the Index Records.

### 1.1.2.2. Index Record

```
struct rpmhdrindex {
    int tag;
    int type;
    int offset;
    int count;
} ;
```

`tag`
Value identifying the purpose of the data associated with this Index Record. This value of this field is dependent on the context in which the Index Record is used, and is defined below and in later sections.

`type`
Value identifying the type of the data associated with this Index Record. The possible `type` values are defined below.

`offset`
Location in the Store of the data associated with this Index Record. This value should between 0 and the value contained in the `hsize` of the Header Structure.

`count`
Size of the data associated with this Index Record. The `count` is the number of elements whose size is defined by the type of this Record.

### 1.1.2.2.1. Index Type Values

The possible values for the `type` field are defined in this table.

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</th>
<th>Size (in bytes)</th>
<th>Alignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPM_NULL_TYPE</td>
<td>0</td>
<td>Not Implemented.</td>
<td></td>
</tr>
<tr>
<td>RPM_CHAR_TYPE</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Chapter 1. Software Installation

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</th>
<th>Size (in bytes)</th>
<th>Alignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPM_INT8_TYPE</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>RPM_INT16_TYPE</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>RPM_INT32_TYPE</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>RPM_INT64_TYPE</td>
<td>5</td>
<td>Reserved.</td>
<td></td>
</tr>
<tr>
<td>RPM_STRING_TYPE</td>
<td>6</td>
<td>variable, NUL terminated</td>
<td>1</td>
</tr>
<tr>
<td>RPM_BIN_TYPE</td>
<td>7</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>RPM_STRING_ARRAY_TYPE</td>
<td>8</td>
<td>Variable, sequence of NUL terminated strings</td>
<td>1</td>
</tr>
<tr>
<td>RPM_I18NSTRING_TYPE</td>
<td>9</td>
<td>variable, sequence of NUL terminated strings</td>
<td>1</td>
</tr>
</tbody>
</table>

The string arrays specified for entries of type RPM_STRING_ARRAY_TYPE and RPM_I18NSTRING_TYPE are vectors of strings in a contiguous block of memory, each element separated from its neighbors by a NUL character.

Index records with type RPM_I18NSTRING_TYPE shall always have a count of 1. The array entries in an index of type RPM_I18NSTRING_TYPE correspond to the locale names contained in the RPMTAG_HDRI18NTABLE index.

1.1.2.2. Index Tag Values

Some values are designated as header private, and may appear in any header structure. These are defined here. Additional values are defined in later sections.

Table 1-4. Header Private Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_HEADERSIGNATURES</td>
<td>62</td>
<td>BIN</td>
<td>16</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_HEADERIMMUTABLE</td>
<td>63</td>
<td>BIN</td>
<td>16</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_HDRI18NTABLE</td>
<td>100</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
</tbody>
</table>

RPMTAG_HEADERSIGNATURES

The signature tag differentiates a signature header from a metadata header, and identifies the original contents of the signature header.

RPMTAG_HEADERIMMUTABLE

This tag contains an index record which specifies the portion of the Header Record which was used for the calculation of a signature. This data shall be preserved or any header-only signature will be invalidated.
Chapter 1. Software Installation

RPMTAG_HEADERI18NTABLE
Contains a list of locales for which strings are provided in other parts of the package.
Not all Index records defined here will be present in all packages. Each tag value has a status which is defined here.
Required
This Index Record shall be present.
Optional
This Index Record may be present.
Deprecated
This Index Record should not be present.
Obsolete
This Index Record shall not be present.
Reserved
This Index Record shall not be present.

1.1.2.3. Header Store
The header store contains the values specified by the Index structures. These values are aligned according to their type and padding is used if needed. The store is located immediately following the Index structures.

1.1.3. Signature Section
The Signature section is implemented using the Header structure. The signature section defines the following additional tag values which may be used in the Index structures.
These values exist to provide additional information about the rest of the package.

Table 1-5. Signature Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIGTAG_SIGSIZE</td>
<td>1000</td>
<td>INT32</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>SIGTAG_PAYLOADSIZE</td>
<td>1007</td>
<td>INT32</td>
<td>1</td>
<td>Optional</td>
</tr>
</tbody>
</table>

SIGTAG_SIGSIZE
This tag specifies the combined size of the Header and Payload sections.

SIGTAG_PAYLOADSIZE
This tag specifies the uncompressed size of the Payload archive, including the cpio headers.
These values exist to ensure the integrity of the rest of the package.
Table 1-6. Signature Digest Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIGTAG_MD5</td>
<td>1004</td>
<td>BIN</td>
<td>16</td>
<td>Required</td>
</tr>
<tr>
<td>SIGTAG_SHa1HEADER</td>
<td>1010</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
</tbody>
</table>

SIGTAG_MD5
This tag specifies the 128-bit MD5 checksum of the combined Header and Archive sections.

SIGTAG_SHa1HEADER
This index contains the SHA1 checksum of the entire Header Section, including the Header Record, Index Records and Header store.

These values exist to provide authentication of the package.

Table 1-7. Signature Signing Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIGTAG_PGP</td>
<td>1002</td>
<td>BIN</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>SIGTAG_GPG</td>
<td>1005</td>
<td>BIN</td>
<td>65</td>
<td>Optional</td>
</tr>
<tr>
<td>SIGTAG_DSAHEADER</td>
<td>1011</td>
<td>BIN</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>SIGTAG_RSAHEADER</td>
<td>1012</td>
<td>BIN</td>
<td>1</td>
<td>Optional</td>
</tr>
</tbody>
</table>

SIGTAG_PGP
This tag specifies the RSA signature of the combined Header and Payload sections. The data is formatted as a Version 3 Signature Packet as specified in RFC 2440: OpenPGP Message Format.

SIGTAG_GPG
The tag contains the DSA signature of the combined Header and Payload sections. The data is formatted as a Version 3 Signature Packet as specified in RFC 2440: OpenPGP Message Format.

SIGTAG_DSAHEADER
The tag contains the DSA signature of the Header section. The data is formatted as a Version 3 Signature Packet as specified in RFC 2440: OpenPGP Message Format. If this tag is present, then the SIGTAG_GPG tag shall also be present.

SIGTAG_RSAHEADER
The tag contains the RSA signature of the Header section. The data is formatted as a Version 3 Signature Packet as specified in RFC 2440: OpenPGP Message Format. If this tag is present, then the SIGTAG_PGP shall also be present.
1.1.4. Header Section

The Header section is implemented using the Header structure. The Header section defines the following additional tag values which may be used in the Index structures.

1.1.4.1. Package Information

The following tag values are used to indicate information that describes the package as a whole.

Table 1-8. Package Info Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_NAME</td>
<td>1000</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_VERSION</td>
<td>1001</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_RELEASE</td>
<td>1002</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_SUMMARY</td>
<td>1004</td>
<td>I18NSTRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_DESCRIPTION</td>
<td>1005</td>
<td>I18NSTRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_SIZE</td>
<td>1009</td>
<td>INT32</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_LICENSE</td>
<td>1014</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_GROUP</td>
<td>1016</td>
<td>I18NSTRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_OS</td>
<td>1021</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_ARCH</td>
<td>1022</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_SOURCE</td>
<td>1044</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_ARCHIVESIZE</td>
<td>1046</td>
<td>INT32</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_RPMVERSION</td>
<td>1064</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_COOKIELIVE</td>
<td>1094</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_PAYLOADFORMAT</td>
<td>1124</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_PAYLOADADCOMPRESSOR</td>
<td>1125</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>Name</td>
<td>Tag Value</td>
<td>Type</td>
<td>Count</td>
<td>Status</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------</td>
<td>----------</td>
<td>-------</td>
<td>-----------</td>
</tr>
<tr>
<td>RPMTAG_PAYLOAD</td>
<td>1126</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>ADFLAGS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This tag specifies the name of the package.

RPMTAG_VERSION

This tag specifies the version of the package.

RPMTAG_RELEASE

This tag specifies the release of the package.

RPMTAG_SUMMARY

This tag specifies the summary description of the package. The summary value pointed to by this index record contains a one line description of the package.

RPMTAG_DESCRIPTION

This tag specifies the description of the package. The description value pointed to by this index record contains a full description of the package.

RPMTAG_SIZE

This tag specifies the sum of the sizes of the regular files in the archive.

RPMTAG_LICENSE

This tag specifies the license which applies to this package.

RPMTAG_GROUP

This tag specifies the administrative group to which this package belongs.

RPMTAG_OS

This tag specifies the OS of the package. The OS value pointed to by this index record shall be "linux".

RPMTAG_ARCH

This tag specifies the architecture of the package. The architecture value pointed to by this index record is defined in architecture specific LSB specification.

RPMTAG_SOURCERPM

This tag specifies the name of the source RPM

RPMTAG_ARCHIVESIZE

This tag specifies the uncompressed size of the Payload archive, including the cpio headers.
Chapter 1. Software Installation

RPMTAG_RPMVERSION
This tag indicates the version of RPM tool used to build this package. The value is unused.

RPMTAG_COOKIE
This tag contains an opaque string whose contents are undefined.

RPMTAG_PAYLOADFORMAT
This tag specifies the format of the Archive section. The format value pointed to by this index record shall be 'cpio'.

RPMTAG_PAYLOADCOMPRESSOR
This tag specifies the compression used on the Archive section. The compression value pointed to by this index record shall be 'gzip'.

RPMTAG_PAYLOADFLAGS
This tag indicates the compression level used for the Payload. This value shall always be '9'.

1.1.4.2. Installation Information
The following tag values are used to provide information needed during the installation of the package.

### Table 1-9. Installation Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_PREIN</td>
<td>1023</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_POSTIN</td>
<td>1024</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_PREUN</td>
<td>1025</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_POSTUN</td>
<td>1026</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_PREINPROG</td>
<td>1085</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_POSTINPROG</td>
<td>1086</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_PREUNPROG</td>
<td>1087</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_POSTINNPROG</td>
<td>1088</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
</tbody>
</table>

RPMTAG_PREIN
This tag specifies the preinstall scriptlet.

RPMTAG_POSTIN
This tag specifies the postinstall scriptlet.
Chapter 1. Software Installation

RPMTAG_PREUN

This tag specifies the preuninstall scriptlet.

RPMTAG_POSTUN

This tag specifies the postuninstall scriptlet.

RPMTAG_PREINPROG

This tag specifies the name of the interpreter to which the preinstall scriptlet will be passed. The interpreter pointed to by this index record shall be '/bin/sh'.

RPMTAG_POSTINPROG

This tag specifies the name of the interpreter to which the postinstall scriptlet will be passed. The interpreter pointed to by this index record shall be '/bin/sh'.

RPMTAG_PREUNPROG

This tag specifies the name of the interpreter to which the preuninstall scriptlet will be passed. The interpreter pointed to by this index record shall be '/bin/sh'.

RPMTAG_POSTUNPROG

This program specifies the name of the interpreter to which the postuninstall scriptlet will be passed. The interpreter pointed to by this index record shall be '/bin/sh'.

1.1.4.3. File Information

The following tag values are used to provide information about the files in the payload. This information is provided in the header to allow more efficient access of the information.

Table 1-10. File Info Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_OLDFILES</td>
<td>1027</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_FILESIZE</td>
<td>1028</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILEMODE</td>
<td>1030</td>
<td>INT16</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILERDEV</td>
<td>1033</td>
<td>INT16</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILETMY</td>
<td>1034</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILEMD5</td>
<td>1035</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILEVER</td>
<td>1036</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
</tbody>
</table>
## Chapter 1. Software Installation

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>NKTOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPMTAG_FILEFLAGS</td>
<td>1037</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILEUSERNAMES</td>
<td>1039</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILEGROUPNAMES</td>
<td>1040</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILEDEVICES</td>
<td>1095</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILEINODES</td>
<td>1096</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILELANGUAGES</td>
<td>1097</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_DIRINDEXES</td>
<td>1116</td>
<td>INT32</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_BASENAMES</td>
<td>1117</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_DIRNAMES</td>
<td>1118</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
</tbody>
</table>

RPMTAG_OLDFILENAMES
This tag specifies the filenames when not in a compressed format as determined by the absence of rpmlib(CompressedFileNames) in the RPMTAG_REQUIRENAME index.

RPMTAG_FILESIZES
This tag specifies the size of each file in the archive.

RPMTAG_FILEMODES
This tag specifies the mode of each file in the archive.

RPMTAG_FILERDEVIFS
This tag specifies the device number from which the file was copied.

RPMTAG_FILEMTIMES
This tag specifies the modification time in seconds since the epoch of each file in the archive.

RPMTAG_FILEMD5S
This tag specifies the ASCII representation of the MD5 sum of the corresponding file contents. This value is empty if the corresponding archive entry is not a regular file.
Chapter 1. Software Installation

RPMTAG_FILELINKTOS
The target for a symlink, otherwise NULL.

RPMTAG_FILEFLAGS
This tag specifies the bit(s) to classify and control how files are to be installed.

RPMTAG_FILEUSERNAME
This tag specifies the owner of the corresponding file.

RPMTAG_FILEGROUPNAME
This tag specifies the of the corresponding file.

RPMTAG_FILEDEVICES
This tag specifies the 16 bit device number from which the file was copied.

RPMTAG_FILEINODES
This tag specifies the inode value from the original file on the build host.

RPMTAG_FILELANGS
This tag specifies a per-file locale marker used to install only locale specific subsets of files when the package is installed.

RPMTAG_DIRINDEXES
This tag specifies the index into the array provided by the RPMTAG_DIRNAMES Index which contains the directory name for the corresponding filename.

RPMTAG_BASENAMES
This tag specifies the base portion of the corresponding filename.

RPMTAG_DIRNAMES
This tag specifies the directory portion of the corresponding filename. Each directory name shall contain a trailing '/'.

One of RPMTAG_OLDFILENAMES or the tuple RPMTAG_DIRINDEXES,RPMTAG_BASENAMES,RPMTAG_DIRNAMES shall be present, but not both.

1.1.4.4. Dependency Information

The following tag values are used to provide information about interdependencies between packages.

Table 1-11. Package Dependency Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_PROVIDENAME</td>
<td>1047</td>
<td>STRING_ARRAY</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_REQUI</td>
<td>1048</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
</tbody>
</table>
## Chapter 1. Software Installation

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>REFLAGS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPMTAG_REQUIRENAME</td>
<td></td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_REQUIREVERSION</td>
<td></td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_CONFLICTFLAGS</td>
<td></td>
<td>INT32</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_CONFLICTNAME</td>
<td></td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_CONFLICTVERSION</td>
<td></td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_OBSOLETENAME</td>
<td></td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_PROVIDEFLAGS</td>
<td></td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_PROVIDEVERSION</td>
<td></td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_OBSOLETEFLAGS</td>
<td></td>
<td>INT32</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_OBSOLETEVERSION</td>
<td></td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
</tbody>
</table>

RPMTAG_PROVIDENAME
This tag indicates the name of the dependency provided by this package.

RPMTAG_REQUIREFLAGS
Bits(s) to specify the dependency range and context.

RPMTAG_REQUIRENAME
This tag indicates the dependencies for this package.

RPMTAG_REQUIREVERSION
This tag indicates the versions associated with the values found in the RPMTAG_REQUIRENAME Index.

RPMTAG_CONFLICTFLAGS
Bits(s) to specify the conflict range and context.

RPMTAG_CONFLICTNAME
This tag indicates the conflicting dependencies for this package.
Chapter 1. Software Installation

RPMTAG_CONFLICTVERSION
This tag indicates the versions associated with the values found in the RPMTAG_CONFLICTNAME Index.

RPMTAG_OBSOLETENAME
This tag indicates the obsoleted dependencies for this package.

RPMTAG_PROVIDEFLAGS
Bits(s) to specify the conflict range and context.

RPMTAG_PROVIDEVERSION
This tag indicates the versions associated with the values found in the RPMTAG_PROVIDENAME Index.

RPMTAG_OBSOLETEFLAGS
Bits(s) to specify the conflict range and context.

RPMTAG_OBSOLETEVERSION
This tag indicates the versions associated with the values found in the RPMTAG_OBSOLETENAME Index.

1.1.4.4.1. Package Dependency Values
The package dependencies are stored in the RPMTAG_REQUIRENAME and RPMTAG_REQUIREVERSION index records.
The following values may be used.

Table 1-12. Index Type values

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>Meaning</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>lsb</td>
<td>2.0</td>
<td>Indicates this is an LSB conforming package.</td>
<td>Required</td>
</tr>
<tr>
<td>rpmlib(VersionedDependencies)</td>
<td>3.0.3-1</td>
<td>Indicates That the package contains RPMTAG_PROVIDENAME, RPMTAG_OBSOLETENAME or RPMTAG_PREREQ records that have a version associated with them.</td>
<td>Optional</td>
</tr>
<tr>
<td>rpmlib(PayloadFilesHavePrefix)</td>
<td>4.0-1</td>
<td>Indicates the filenames in the Archive have had &quot;.&quot; prepended to them.</td>
<td>Optional</td>
</tr>
<tr>
<td>rpmlib(CompressedFileNames)</td>
<td>3.0.4-1</td>
<td>Indicates that the filenames in the Payload are represented in the RPMTAG_DIRINDEXES, RPMTAG_DIRNAME and</td>
<td>Optional</td>
</tr>
</tbody>
</table>
1.1.4.4.2. Package Dependencies Attributes

The package dependency attributes are stored in the RPMTAG_REQUIREFLAGS, RPMTAG_PROVIDEFLAGS and RPMTAG_OBSOLETEFLAGS index records. The following values may be used.

Table 1-13. Package Dependency Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMSENSE_LESS</td>
<td>0x02</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_GREATER</td>
<td>0x04</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_EQUAL</td>
<td>0x08</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_PREREQ</td>
<td>0x40</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_INTERP</td>
<td>0x100</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_SCRIPT_PRE</td>
<td>0x200</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_SCRIPT_POST</td>
<td>0x400</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_SCRIPT_PREUN</td>
<td>0x800</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_SCRIPT_POSTUN</td>
<td>0x1000</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_RPMLIB</td>
<td>0x1000000</td>
<td></td>
</tr>
</tbody>
</table>

1.1.4.5. Other Information

The following tag values are also found in the Header section.

Table 1-14. Other Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_BUILD TIME</td>
<td>1006</td>
<td>INT32</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_BUILD HOST</td>
<td>1007</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_FILEWRITE</td>
<td>1045</td>
<td>INT32</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_CHANGE</td>
<td>1080</td>
<td>INT32</td>
<td></td>
<td>Optional</td>
</tr>
</tbody>
</table>
Chapter 1. Software Installation

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELOGTIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPMTAG_CHANGELOGNAME</td>
<td>1081</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_CHANGELOGTEXT</td>
<td>1082</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_OPTFLAGS</td>
<td>1122</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_RHNPLATFORM</td>
<td>1131</td>
<td>STRING</td>
<td>1</td>
<td>Deprecated</td>
</tr>
<tr>
<td>RPMTAG_PLATFORM</td>
<td>1132</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
</tbody>
</table>

RPMTAG_BUILDTIME

This tag specifies the time as seconds since the epoch at which the package was built.

RPMTAG_BUILDHOST

This tag specifies the on which which the package was built.

RPMTAG_FILEVERIFYFLAGS

This tag specifies the bit(s) to control how files are to be verified after install, specifying which checks should be performed.

RPMTAG_CHANGELOGTIME

This tag specifies the Unix time in seconds since the epoch associated with each entry in the Changelog file.

RPMTAG_CHANGELOGNAME

This tag specifies the name of who made a change to this package

RPMTAG_CHANGELOGTEXT

This tag specifies the changes associated with a changelog entry.

RPMTAG_OPTFLAGS

This tag indicates additional flags which may have been passed to the compiler when building this package.

RPMTAG_RHNPLATFORM

This tag contains an opaque string whose contents are undefined.

RPMTAG_PLATFORM

This tag contains an opaque string whose contents are undefined.
1.1.5. Payload Section

The Payload section contains a compressed cpio archive. The format of this section is defined by RFC 1952: GZIP File Format Specification.

When uncompressed, the cpio archive contains a sequence of records for each file. Each record contains a CPIO Header, Filename, Padding, and File Data.

<table>
<thead>
<tr>
<th>Table 1-15. CPIO File Format</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPIO Header</strong></td>
</tr>
<tr>
<td><strong>Filename</strong></td>
</tr>
<tr>
<td><strong>Padding</strong></td>
</tr>
<tr>
<td><strong>File data</strong></td>
</tr>
<tr>
<td><strong>Padding</strong></td>
</tr>
</tbody>
</table>

The CPIO Header uses the following header structure (sometimes referred to as "new ASCII" or "SVR4 cpio"). All numbers are stored as ASCII representations of their hexadecimal value with leading zeros as needed to fill the field. With the exception of `c_namesize` and the corresponding name string, and `c_checksum`, all information contained in the CPIO Header is also represented in the Header Section. The values in in the CPIO Header shall match the values contained in the Header Section.

```c
struct {
    char    c_magic[6];
    char    c_ino[8];
    char    c_mode[8];
    char    c_uid[8];
    char    c_gid[8];
    char    c_nlink[8];
    char    c_mtime[8];
    char    c_filesize[8];
    char    c_devmajor[8];
    char    c_devminor[8];
    char    c_rdevmajor[8];
    char    c_rdevminor[8];
    char    c_namesize[8];
    char    c_checksum[8];
};
```

`c_magic`

Value identifying this cpio format. This value shall be "070701".
Chapter 1. Software Installation

356 c_ino
357 This field contains the inode number from the filesystem from which the file was read. This field is ignored when
358 installing a package. This field shall match the corresponding value in the RPMTAG_FILEINODES index in the
359 Header section.

360 c_mode
361 Permission bits of the file. This is an ascii representation of the hexadecimal number representing the bit as
362 defined for the st_mode field of the stat structure defined for the stat function. This field shall match the
363 corresponding value in the RPMTAG_FILEMODES index in the Header section.

364 c_uid
365 Value identifying this owner of this file. This value matches the uid value of the corresponding user in the
366 RPMTAG_FILEUSERNAME as found on the system where this package was built. The username specified in
367 RPMTAG_FILEUSERNAME should take precedence when installing the package.

368 c_gid
369 Value identifying this group of this file. This value matches the gid value of the corresponding user in the
370 RPMTAG_FILEGROUPNAME as found on the system where this package was built. The groupname specified
371 in RPMTAG_FILEGROUPNAME should take precedence when installing the package.

372 c_nlink
373 Value identifying the number of links associated with this file. If the value is greater than 1, then this filename
374 will be linked to 1 or more files in this archive that has a matching value for the c_ino, c_devmajor and
375 c_devminor fields.

376 c_mtime
377 Value identifying the modification time of the file when it was read. This field shall match the corresponding
378 value in the RPMTAG_FILEMTIMES index in the Header section.

379 c_filesize
380 Value identifying the size of the file. This field shall match the corresponding value in the RPMTAG_FILESIZES
381 index in the Header section.

382 c_devmajor
383 The major number of the device containing the file system from which the file was read. With the exception of
384 processing files with c_nlink >1, this field is ignored when installing a package. This field shall match the
385 corresponding value in the RPMTAG_FILEDEVICES index in the Header section.

386 c_devminor
387 The minor number of the device containing the file system from which the file was read. With the exception of
388 processing files with c_nlink >1, this field is ignored when installing a package. This field shall match the
389 corresponding value in the RPMTAG_FILEDEVICES index in the Header section.
Chapter 1. Software Installation

\texttt{c\_rdevmajor}

The major number of the raw device containing the file system from which the file was read. This field is ignored when installing a package. This field shall match the corresponding value in the \texttt{RPMTAG\_RDEVS} index in the Header section.

\texttt{c\_rdevminor}

The minor number of the raw device containing the file system from which the file was read. This field is ignored when installing a package. This field shall match the corresponding value in the \texttt{RPMTAG\_RDEVS} index in the Header section.

\texttt{c\_namesize}

Value identifying the length of the filename, which is located immediately following the CPIO Header structure.

\texttt{c\_checksum}

Value containing the CRC checksum of the file data. This field is not used, and shall contain the value "00000000". This field is ignored when installing a package.

A record with the filename "TRAILER!!!" indicates the last record in the archive.

1.2. Package Script Restrictions

Scripts used as part of the package install and uninstall shall only use commands and interfaces that are specified by the LSB. All other commands are not guaranteed to be present, or to behave in expected ways.

Packages shall not use RPM triggers.

Packages shall not depend on the order in which scripts are executed (pre-install, pre-uninstall, &c), when doing an upgrade.

1.3. Package Tools

The LSB does not specify the interface to the tools used to manipulate LSB-conformant packages. Each conforming distribution shall provide documentation for installing LSB packages.

1.4. Package Naming

Packages supplied by distributions and applications must follow the following rules for the name field within the package. These rules are not required for the filename of the package file itself.\textsuperscript{1}

The following rules apply to the name field alone, not including any release or version.\textsuperscript{4}

- If the name begins with "lsb-" and contains no other hyphens, the name shall be assigned by the Linux Assigned Names and Numbers Authority (http://www.lanana.org) (LANANA), which shall maintain a registry of LSB names. The name may be registered by either a distribution or an application.

- If the package name begins with "lsb-" and contains more than one hyphen (for example "lsb-distro.example.com-database" or "lsb-gnome-gnumeric"), then the portion of the package name between first and second hyphens shall either be an LSB provider name assigned by the LANANA, or it may be one of the owners' fully-qualified domain names in lower case (e.g., "debian.org", "staroffice.sun.com"). The LSB provider...
name assigned by LANANA shall only consist of the ASCII characters [a-z0-9]. The provider name or domain
name may be either that of a distribution or an application.

- Package names containing no hyphens are reserved for use by distributions. Applications must not use such names.\(^5\)
- Package names which do not start with "lsb-" and which contain a hyphen are open to both distributions and
applications. Distributions may name packages in any part of this namespace. They are encouraged to use names
from one of the other namespaces available to them, but this is not required due to the large amount of current
practice to the contrary.\(^6\) Applications may name their packages this way, but only if the portion of the name before
the first hyphen is a provider name or registered domain name as described above.\(^7\) Note that package names in this
namespace are available to both the distribution and an application. Distributions and applications will need to
consider this potential for conflicts when deciding to use these names rather than the alternatives (such as names
starting with "lsb-").

### 1.5. Package Dependencies

Packages shall have a dependency that indicates which LSB modules are required. LSB module descriptions are dash
separated tuples containing the name 'lsb', the module name, and the architecture name. The following dependencies
may be used.

- **lsb-core-arch**
  
  This dependency is used to indicate that the application is dependent on features contained in the LSB-Core
  specification.

- **lsb-core-noarch**
  
  This dependency is used to indicate that the application is dependent on features contained in the LSB-Core
  specification and that the package does not contain any architecture specific files.

Packages shall not depend on other system-provided dependencies. They shall not depend on non-system-provided
dependencies unless those dependencies are fulfilled by packages which are part of the same application. A package
may only provide a virtual package name which is registered to that application.

Other modules in the LSB may supplement this list. The architecture specific dependencies are described in the
relevant architecture specific LSB.

### 1.6. Package Architecture Considerations

Packages which do not contain any architecture specific files must specify an architecture of **noarch**. A LSB runtime
environment must accept values **noarch**, or the value specified in the architecture specific supplement.

Additional specifications or restrictions may be found in the architecture specific LSB specification.

### Notes

1. Supplying an RPM format package is encouraged because it makes systems easier to manage. A future version of
   the LSB may require RPM, or specify a way for an installer to update a package database.
   Applications are also encouraged to uninstall cleanly.
Chapter 1. Software Installation

2. The distribution itself may use a different packaging format for its own packages, and of course it may use any available mechanism for installing the LSB-conformant packages.

3. For example, there are discrepancies among distributions concerning whether the name might be frobnicator-1.7-21-ppc32.rpm or frobnicator-1.7-21-powerpc32.rpm. The architecture aside, recommended practice is for the filename of the package file to match the name within the package.

4. For example, if the name with the release and version is frobnicator-1.7-21, the name part is frobnicator and falls under the rules for a name with no hyphens.

5. For example, "frobnicator".

6. For example, ssh-common, ssh-client, kernel-pcmcia, and the like. Possible alternative names include sshcommon, foolinux-ssh-common (where foolinux is registered to the distribution), or lsb-foolinux-ssh-common.

7. For example, if an application vendor has domain name visicalc.example.com and has registered visicalc as a provider name, they might name packages visicalc-base, visicalc.example.com-charting, and the like.
Free Documentation License
Free Documentation License
Table of Contents

A. GNU Free Documentation License ........................................................................................................ I
   A.1. PREAMBLE................................................................................................................................. 1
   A.2. APPLICABILITY AND DEFINITIONS...................................................................................... 1
   A.3. VERBATIM COPYING ............................................................................................................... 2
   A.4. COPYING IN QUANTITY .......................................................................................................... 2
   A.5. MODIFICATIONS ..................................................................................................................... 3
   A.6. COMBINING DOCUMENTS ....................................................................................................... 4
   A.7. COLLECTIONS OF DOCUMENTS ........................................................................................... 4
   A.8. AGGREGATION WITH INDEPENDENT WORKS ................................................................... 4
   A.9. TRANSLATION ......................................................................................................................... 5
   A.10. TERMINATION ....................................................................................................................... 5
   A.11. FUTURE REVISIONS OF THIS LICENSE ............................................................................. 5
   A.12. How to use this License for your documents ........................................................................ 5
Appendix A. GNU Free Documentation License

Version 1.1, March 2000

Copyright (C) 2000 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

A.1. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

A.2. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been
Appendix A. GNU Free Documentation License

designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not
"Transparent" is called "Opaque".
Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format,
LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML
designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and
edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not
generally available, and the machine-generated HTML produced by some word processors for output purposes only.
The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold,
legibly, the material this License requires to appear in the title page. For works in formats which do not have any title
page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the
beginning of the body of the text.

A.3. VERBATIM COPYING
You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that
this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced
in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical
measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may
accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow
the conditions in section 3.
You may also lend copies, under the same conditions stated above, and you may publicly display copies.

A.4. COPYING IN QUANTITY
If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires
Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover
Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify
you as the publisher of these copies. The front cover must present the full title with all words of the title equally
prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the
covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim
copying in other respects.
If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit
reasonably) on the actual cover, and continue the rest onto adjacent pages.
If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a
machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a
publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of
added material, which the general network-using public has access to download anonymously at no charge using
public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you
begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the
stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents
or retailers) of that edition to the public.
It is requested, but not required, that you contact the authors of the Document well before redistributing any large
number of copies, to give them a chance to provide you with an updated version of the Document.
A.5. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.
A.6. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

A.7. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

A.8. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.
Appendix A. GNU Free Documentation License

A.9. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

A.10. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

A.11. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

A.12. How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.