# Using SHPK

August 25, 2005

Version 1.1
HP-UX Servers

# Legal Notices

The information contained herein is subject to change without notice.

## Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

## U.S. Government License

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

## Trademark Notices

Red Hat® is a registered trademark of Red Hat, Inc.

Solaris® is the trademark or registered trademark of Sun Microsystems, Inc. in the United States and other countries.

Intel® and Itanium® are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. UNIX® is a registered trademark in the United States and other countries, licensed exclusively through The Open Group.

AMD® Opteron® is a trademark, or registered trademark of Advanced Micro Devices in the United States and/or other countries.

## Copyright Notice

# Contents

# About This Document

This document describes how to install and use the HP Solaris-to-HP-UX Porting Kit (SHPK) on HP-UX platforms. The document printing date indicates the documents current edition. The printing date will change when a new edition is printed. Minor changes may be made at reprint without changing the printing date. Document updates may be issued between editions to correct errors or document product changes. To ensure that you receive the updated or new editions, you should subscribe to the appropriate product support service. Please send your requests to rapid.migration@hp.com.

## Intended Audience

This document is intended for application developers responsible for porting their source code from Solaris-to-HP-UX. Developers are expected to have knowledge of operating system concepts and application development.

This document is not a tutorial.

## Document Organization

Using SHPK is divided into several chapters, and each contains information about using SHPK to migrate from Solaris-to-HP-UX. The appendixes area also included and contain supplemental information.

Chapter 1 SHPK Introduction Use this chapter to learn the summary of SHPK features.

Chapter 2 Installing SHPK Use this chapter to prepare for SHPK software installation on your HP-UX platform.

Chapter 3 Using SHPK - Migration Use this chapter to learn how to use SHPK to migrate from Solaris-to-HP-UX.

Chapter 4 Using SHPK Tools Use this chapter to learn how to use the tools that SHPK includes.

Chapter 5 Migration Considerations Use this chapter to learn more about the SHPK Tools.

## Typographic Conventions

This document uses the following conventions.

*sh_cc* A HP-UX manpage. In this example, sh_cc is the name of the manpage. On the Web and on the Instant Information CD, it may be a hot link to the manpage itself. From the HP-UX command line, you can enter man sh_cc to view the manpage. See man (1).

*Book Title* The title of a book. On the web and on the Instant Information CD, it may be a hot link to the book itself.

*Emphasis* Text that is emphasized.

**Bold** Text that is strongly emphasized.

**Bold** The defined use of an important word or phrase.

`ComputerOut` Text displayed by the computer.

`UserInput` Commands and other text that you type.

`Command` A command name or qualified command phrase.

`Variable` The name of a variable that you may replace in a command or function or information in a display that represents several possible values.

[ ] The contents are optional in formats and command descriptions. If the contents are a list separated by |, you must choose one of the items.

The contents are required in formats and command descriptions. If the contents are a list separated by |, you must choose one of the items.

... The preceding element may be repeated an arbitrary number of times.

| Separates items in a list of choices.

## Related Documents

Additional information about SHPK can be found at:

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

## HP Encourages Your Comments

HP encourages your comments concerning this document. We are truly committed to providing documentation that meets your needs.

Please send comments to: slpk.support@hp.com

Please include document title, and any comment, error found, or suggestion for improvement you have concerning this document. Also, include what we did right so we can incorporate it into other documents.

# 1 SHPK Introduction

This chapter provides an introduction to Solaris-to-HP-UX Porting Kit (SHPK) that enables you to migrate from Solaris-to-HP-UX. It discusses the following topics:

- Solaris-to-HP-UX Porting Kit
- SHPK Components

### Solaris-to-HP-UX Porting Kit Overview

Solaris-to-HP-UX Porting Kit (SHPK) enables you to port your source code from Solaris-to-HP-UX with ease. Most porting occurs in the C and C++ language environments.

SHPK significantly reduces the cost of migrating applications from Solaris-to-HP-UX. SHPK provides a migration path for Solaris applications with minimal porting overheads. This migration path makes the Solaris-to-HP-UX migration process less time-consuming and more affordable.

For more information, see Chapter 3, Using SHPK - Migration. In addition, see SHPK Components , for more information on SHPK components.

### SHPK Benefits

Following lists the benefits of using SHPK:

- Addressing toolset differences

  The compiler tools options in HP-UX compilers are different from those in Sun compilers. In addition, tools such as ln, mkdir support different options. SHPK provides driver programs that translate these options from Solaris-to-HP-UX.

- Addressing API differences

  Many APIs in HP-UX are different from those in Solaris, in terms of their usage and parameters. Some Solaris APIs have no equivalents in HP-UX. SHPK provides header files and API implementations that address these differences.

- Addressing makefile changes

  Migrating from Solaris-to-HP-UX requires manual modification of makefiles to address the differences in the build environment. SHPK provides make tools that minimize such changes.

### SHPK Components

SHPK is designed on the same lines as the Solaris-to-Linux Porting Toolkit.

Table 1: SHPK Components

| Component | Description |
|---|---|
| Migration Environment | Includes wrapper libraries and header files that bridge the difference in syntax between Solaris and HP-UX. They also include some new APIs that are currently not available on HP-UX. |
| Compiler and tools drivers | Compiler and linker drivers that map the Solaris options to their HP-UX equivalents, and call the corresponding binaries. |
| Make tools | Minimizes manual intervention during Solaris-to-HP-UX migration by editing the make files to address differences in the build environment. |
| Useful utilities | A set of useful utilities that help you retain the Solaris flavor. |

NOTE Use the Solaris-to-HP-UX Software Transition Kit to get a detailed assessment of the changes to be made. The Solaris-to-HP-UX STK includes a software wizard which scans the source code developed on the Solaris operating system and produces a report that highlights the compatibility issues with HP-UX. It also provides recommendation on how to resolve these issues. The Solaris-to-HP-UX STK includes technical reference documents as well as impact statements offering detailed porting information on major libraries, including libc, libsocket, libthread and libpthread.

NOTE See Chapter 4, Using SHPK Tools, for more information on tools that SHPK includes.

### Migration Environment

Libraries provide the missing components, during the runtime environment, for the applications to be ported. They also include the header files that the wrapper libraries use.

The following lists the libraries delivered with this release of SHPK:

- libwrapsc

  Wrapper for system calls

- libwrapstream

  Wrapper for network API functions

- libwrapstl

  Wrapper for threads

NOTE The SHPK libraries are usually available in /opt/shpk/lib/lp32 for 32-bit applications and in /opt/shpk/lib/lp64 for 64-bit applications.

For example, libraries for 32-bit applications are available as follows:

- /opt/shpk/lib/lp32/libwrapsc.a
- /opt/shpk/lib/lp32/libwrapstream.a
- /opt/shpk/lib/lp32/libwrapstl.a

# 2 Installing SHPK

This chapter describes how to install the Solaris-to-HP-UX Porting Kit (SHPK) on your system. The chapter also details the prerequisites for installing SHPK.

It discusses the following topics:

- Prerequisites
- Installation Steps

## Prerequisites

The following lists the prerequisites for installing SHPK:

- Operating system requirements

  This version of SHPK is supported on HP Servers running HP-UX 11iv2.

- Hardware requirements

  SHPK is supported on HP hardware, and on the following processors:

  – Itanium, PA-RISC

- Diskspace requirements

  You must have 8 MB of free diskspace on your system.

- Other requirements

  – You must install HP HP C/aCC Compiler Collection.

## Installation Steps

To install the SHPK software on your system, complete the following steps:

1. Run the following command to change your present working directory to the SHPK parent directory:

   ```
   $ cd /opt
   ```

2. Run the shell script and untar the SHPK tar ball, as in the following example:

   ```
   $ sh SHPK-1.1-HP-UX-B.11.23.ia64.tgz.sh
   ```

   ```
   $ tar -xvf SHPK-1.1-HP-UX-B.11.23.ia64.tar
   ```

Upon installation, a directory called shpk and a set of sub-directories under that directory are created on your system. These directories contain all SHPK tools, manual pages, and libraries.

NOTE Unless stated otherwise, the examples in this document assume that SHPK is installed in /opt/shpk.

### Changing the Default Path

To install SHPK in a different directory, complete the following steps:

1. Change your present working directory to a directory of your choice, by running the cd command. For example:

   ```
   $ cd /home/danny/shpk1.3
   ```

2. Run the shell script and untar the SHPK tar ball, as in the following example:

   ```
   $ sh SHPK-1.1-HP-UX-B.11.23.ia64.tgz.sh
   ```

   ```
   $ tar -xvf SHPK-1.1-HP-UX-B.11.23.ia64.tar
   ```

   Upon installation, a directory called shpk and a set of sub-directories under that directory are created under /home/danny/shpk1.3/ on your system.

3. Set an environment variable called SHPK_ROOT=/home/danny/shpk1.3/shpk

   This directory contains all SHPK tools and libraries. If you do not set the environment variable, SHPK uses the default path.

# 3 Using SHPK - Migration

This chapter includes information on how to migrate applications from Solaris-to-HP-UX using SHPK.

### Compiling an Application Using SHPK

1. Define the SHPK_ROOT environment variable (if needed), and include the location of the SHPK binaries in your PATH variable.

   For example:

   ```
   $ export SHPK_ROOT=/opt/shpk
   ```

   ```
   $ export PATH=$SHPK_ROOT/shpk/bin:$PATH
   ```

2. Compile your application with sh_make as follows:

   ```
   $ sh_make -f <makefile>
   ```

3. Check for compilation and linking errors while building your application with SHPK.

4. Upon Compilation, check whether a valid executable file has been generated.

# 4 Using SHPK Tools

SHPK includes useful tools, wrapper library of APIs and header files, and scanner tools to find and address porting issues. You can use the SHPK tools to map the Solaris functionality that is missing in the compiler toolchain and other utilities critical to a development environment.

The SHPK tools include the following:

- Compiler and linker drivers that map the Solaris options to equivalent HP-UX options and call the corresponding binaries.

The following lists the tools that SHPK includes:

- sh_cc

  A compiler driver that dynamically maps the Solaris options and paths to their HP-UX equivalents, and calls cc by passing the translated options.

  For more information, read the sh_cc manpage.

- sh_c++

  A compiler driver that dynamically maps the Solaris options and paths to their HP-UX equivalents, and calls aCC by passing the translated options.

  For more information, read the sh_c++ manpage.

- sh_ld

  A linker driver that dynamically maps the Solaris options and paths to their HP-UX equivalents, and triggers ld by passing the translated options.

  For more information, read the sh_ld manpage.

- sh_make

  A GNU Make version, modified to understand the behavior of Solaris make, which sets the internal macros like CXX and LD to point to sh_c++ and sh_ld. This transparently triggers the SHPK tools instead of aCC, ld directly (if not already set).

For more information, read the sh_make manpage.

### Environment Variables

The following section details the SHPK environment variables:

You can use certain environment variable settings to change the behavior of the drivers. These changes can be achieved as follows:

### SHPK_ROOT

Normally, SHPK tools are installed in /opt/shpk. SHPK_ROOT is an environment variable that allows you to set the path of the directory, in which the tools and libraries can reside.

If SHPK_ROOT is set, it defines the location, in which the tools are installed.

If SHPK_ROOT is not set, SHPK takes the default path (/opt/shpk). Example: If SHPK is installed in /home/opt/shpk, then the SHPK_ROOT must be set to /home/opt/shpk.

### SH_SHPKLIBS

SH_SHPKLIBS=ON|OFF

- SH_SHPKLIBS=ON: This is the default behavior. This setting is equivalent to passing the options -shpk_libs to the driver. They add -lwrapsc and -lwrapstream to the linker command line.
- SH_SHPKLIBS=OFF: This setting is equivalent to passing the option -noshpk_libs. Using this setting, does not add the wrapper libraries to the resultant command line.

Command line options to the driver will take priority over the environment variable settings. Thus, even if the environment variable SH_SHPKLIBS is set to OFF, if -shpk_libs option is passed to the driver, then -lwrapsc and -lwrapstream are added to the command line.

### SH_SHPKSTL and SH_SHPKPROC

SHPK consists of the following four libraries:

- libwrapsc.a
- libwrapstream.a
- libwrapstl.a
- libwrapproc.a (under design)

Compiler drivers include libwrapsc.a and libwrapstream.a by default in the link phase.

If libwrapstl.a and libproc.a need to be included, they either need to be added to the linker command line argument list or linker or the environment variables SH_SHPKSTL, SH_SHPKPROC need to be set to ON. The path of the libraries must be the absolute path.

### Compiler and Linker Drivers

The following drivers are provided to automate translation of Solaris options to their equivalent HP-UX options:

- sh_cc
- sh_c++
- sh_ld

These drivers operate with the help of mapping files that provide the mapping of Solaris options to their equivalent HP-UX options.

### Invoking Driver Programs

You can invoke these tools as:

```
sh_c++ [options] <commandline arguments passed to aCC>

sh_cc [options] <commandline arguments passed to cc>
```

Table 4-2 Compiler and Linker Driver Options

| Option | Description |
|---|---|
| -hp | After mapping the options the driver passes the resultant options to aCC compiler (This is the default behavior). |
| -gnu | After mapping the options the driver passes the resultant options to gcc/g++ compiler |
| -noshpk_libs | When this option is passed to the driver, it does not include the SHPK wrapper libraries to the resultant command line. |
| -report <filename> | When this option is passed to the driver, the driver programs register all the Solaris-to-HP-UX option mappings into the file <filename>. |

You can change the behavior of the drivers using certain environment variable settings as follows:

At startup, the driver program reads the map file and builds an internal table containing the mappings. Then for each option passed to it, if an equivalent mapping is present in the table, it replaces that option with the corresponding HP-UX option as specified in the map file. The driver program passes the Solaris option, which does not have a mapping in the map file, to the target compiler in its present form.

### Invoking sh_ld

You must invoke the linker(ld) driver sh_ld as:

```
sh_ld [options] <command line passed to ld>
```

The driver options for sh_ld are:

```
-report <filename>
```

When this option is passed to the driver, all Solaris-to-HP-UX option mappings are registered into the file <filename>.

The map file $(SHPK_HOME)/lib/sun2ld.map drives the option mapping for this driver. The process of mapping and discarding are similar to that of sh_cc and sh_c++.

NOTE Limitations of the Translation Process: When the source code, which is being ported, uses the c tags for the platforms such as:

```
#ifdef sun

<code1>

#elif hpux

<code2>

#else

<code3>

#endif
```

porting from Solaris-to-HP-UX executes <code1>, whereas compiling with SHPK tools or drivers lead to the execution of <code2> unless the -Dsun option is passed to the compiler drivers.

### Format for Mapping Files

SHPK uses a set of internal mapping files to do the commandline option conversion.

The option mapping in the mapping files is in the following form:

```
<solaris option> => <HP-UX option(s)>
```

Each line has one mapping, and a => forms the delimiter between the options.

An option with one or more arguments will be written in the following form:

```
-h $1 => -WI,$1
```

Here, $1 is an argument, which can be matched with any string without a white space. A single mapping can have any number of arguments. Options with white space in between will be formed by reading equivalent number of commandline arguments passed to the driver.

A mapping of the form:

```
<solaris option> =>
```

means that, there is no equivalent HP-UX option for the particular Solaris options, and any instance of this option in the command line is discarded.

The three standard mapping files used are:

- $(SHPK_ROOT)/lib/sun2hp.map used by sh_cc and sh_c++ to map the Solaris options to their equivalent aCC options
- $(SHPK_ROOT)/lib/sun2gnu.map used by sh_cc and sh_c++ to map the Solaris options to their equivalent gcc/g++ options
- $(SHPK_ROOT)/lib/sun2ld.map used by sh_ld to map the Solaris linker options to HP-UX linker options

### sh_make

The sh_make utility accepts a Solaris makefile and performs all the processing necessary to build the executable on HP-UX; sh_make also forces the inclusion of SHPK Migration Environment during the building of the application.

Usage:

```
sh_make [......]

sh_make --shpk-gnu [....]

sh_make --gnu [....]
```

Option Summary

-shpk-gnu By default, sh_make addresses the functionality differences between the deafault make utilities on Solaris and HP-UX and forces the inclusion of SHPK migration environment. The shpk-gnu option causes the makefile to be treated like a GNU makefile; with this option only the SHPK migration environment is included.

-gnu With this option, sh_make will behave exactly like the GNU make utility.

NOTE A source scanner tool - sh_src_scanner - is packaged along with the SHPK toolkit. HP recommends the usage of sh_src_scanner tool to obtain a quick and brief assessment of the changes required in the source code. Please read the sh_src_scanner man page for more details. For a detailed assessment of the changes required, use the Solaris-to-HP-UX STK.

### Other Utilities

The following commands provide features existing in Solaris, but not in HP-UX:

```
sh_rmdir [OPTION]... DIRECTORY...

sh_ln [OPTION]... TARGET [LINK\_NAME]

sh_ln [OPTION]... TARGET... DIRECTORY

sh_ln [OPTION]... --target-directory=DIRECTORY TARGET...
```

## 5 Migration Considerations

Before you migrate the application from Solaris-to-HP-UX, you need to be aware of the following:

- Differences in Library and System Calls
- Differences in Header Files
- Differences in Compiler Options
- Differences in Linker Options
- Differences in Makefiles
- Differences in the Utilities

NOTE SHPK takes care of these differences if you use SHPK tools for migration.

### Differences in Library and System Calls

The library and system calls available on Solaris are different from those available on HP-UX. This causes the linking step to fail during the application migration.

SHPK tools address this issue by providing a library of functions, which are present on Solaris but not on HP-UX. SHPK tools automatically use these libraries in the linking phase.

### Differences in Header Files

The differences in header files present on Solaris and HP-UX can cause compilation to fail on HP-UX. Some of the reasons for this failure are as follows:

- The header file present on Solaris may not be present on HP-UX.
- The prototype of the functions in the Solaris header file may not match with those present in HP-UX header files.
- The structure definitions, type definitions, and other attributes may not match in the Solaris and HP-UX header files.
- Although the structure or type definitions are similar, the field names in the structure may be different.

### Differences in Compiler Options

In the Solaris environment, the Forte compilers (cc and CC) are used extensively for application development. In the HP-UX environment, the HP aCC compiler collection is used. The command-line options are different for Forte and HP aCC Compilers. The pragmas too are different.

The SHPK tools sh_cc and sh_c++ handle the differences in the options between compilers on Solaris and HP-UX.

### Differences in Linker Options

In the Solaris environment, the native linker is used for creating an executable or a shared library. In the HP-UX enviroment, the native HP-UX linker is used. The command-line options are different for the native Solaris Linker and the the native HP-UX linker.

The SHPK tool sh_ld bridges the gap between the native linkers.

### Differences in Makefiles

The make and makefiles on Solaris are slightly different from the make and makefiles on HP-UX. The files have syntactic and semantic differences.

SHPK includes a very useful utility called sh_make. This utility bridges the differences in functionality between the default make tools on Solaris and HP-UX. By default, sh_make also includes the SHPK Migration Environment during the build process. To get the benefits provide by this tool, simply use sh_make instead of make or gmake on the HP-UX platform.