OpenVMS UNIX Application Portability Initiative

Brad McCusker
OpenVMS Engineering

Topics

- Introduction to UNIX Portability
- OpenVMS V7.3-2 Tips
- OpenVMS V8.2
- GNV
- Futures
- Summary

Various Terms – all mean the same…

- UNIX Portability is the original term used to describe this effort
- The effort quickly evolved to include:
  - Linux Interoperability
  - Linux Portability
  - Open Source Interoperability
  - Open Source Portability

All these are part of UNIX Portability

Unix Portability - Rationale

- Many ISVs develop applications for both OpenVMS and UNIX/Linux platforms
- Applications are (or can be) ported from UNIX/Linux platforms to OpenVMS
- Operators, programmers, users may be more familiar with *NIX-style interfaces, commands, utilities and tools

UNIX Portability - Goal

Provide a full set of UNIX interfaces and tools within OpenVMS

- In native, integrated fashion
- No layered emulator (e.g. old "POSIX for OpenVMS" product)
  - No performance issues
  - No interoperability issues

UNIX Portability - Benefits

- Easy portability of UNIX applications to OpenVMS
- Easy development of applications intended to run on both UNIX and OpenVMS
- No need to train UNIX-skilled personnel on OpenVMS

- OpenVMS will optionally be like a "UNIX flavor"
  - Cost of porting from UNIX to OpenVMS equal or comparable to porting from one "UNIX flavor" to another (e.g. from Solaris to Tru64)

But – I like VMS the way it is!!!

- Current VMS behavior is preserved
  - New UNIX Portability features typically need to be enabled
  - Defaults preserve existing behavior

- C Run Time Library: UNIX features are enabled via logical name switches
  - Old behavior is the default
    - Legacy behavior is preserved
  - Can also enable features via an API

Rollout…

- Started already with VMS V7.3-1, V7.3-2 …
  - GNV/BASH (Commands & Utilities)
  - C RTL
    - New UNIX APIs
    - Improved UNIX filename support
    - API for controlling feature switches
  - File system improvements
    - Mixed case file names, case sensitive compares
    - Time of last file access
    - Hard link improvements
    - Root directory support (pseudo – SYS$POSIX_ROOT)
  - Base VMS improvements
    - Extended DCL line length

UNIX Portability Roadmap

OpenVMS V8.2

Symbolic Links
- CRTL APIs
- DCL support
- NFS support

CRTL
- File Lock APIs
- statvfs/fstatvfs
- Stnd stat struct

GNV 1.6
- VI
- gnuTAR
- Continued configure and Make improvements

OpenVMS V8.3
- Byte range locking
- **Semaphores**
- **Asynch I/O APIs**
- GNV update

OpenVMS V8.x
- Binary Tree
- ioctl()
- UNIX Compliant Signals
- GNV update

OpenVMS UP future investigations
- Full function select()
- fork ()
- Shared memory APIs
- Unix 98 compliance

Note: Investigations are provided solely to inform what is being considered and should not be used as a deliverable commitment.

V7.3-2 Tips

UNIX Portability Features

V7.3-2 C RTL - Tips

- glob(), globfree()
  - New pattern matching APIs
  - Extended to allow for VMS style behavior
    - Controlled by feature switch – default is VMS behavior
    - Use VMS wild cards, not UNIX wildcards
    - Uses sys$search/sys$parse
    - No pattern matching
    - Returns VMS style specs
  - DECC$GLOB_UNIX_STYLE
    - Enables UNIX specific behavior
    - Use UNIX wildcards, return UNIX style specs

V7.3-2 C RTL Tips

- TCP/IP related enhancements
  - 64-bit pointer support in: sendmsg, recvmsg, freeaddrinfo, getaddrinfo
    - Previously, 32-bit implementation only
    - Now, dual implementation, 64 bit and 32 bit
      - Be careful with new 64 bit data structures
      - /POINTER_SIZE=LONG will not provide 64 bit data structure
      - Use 64 bit specific structure
      - See C RTL Ref Manual 1.10.4.3 “Functions With Two Implementations”

V7.3-2 C RTL Tips
New Feature Switch

- New switch – DECC$RENAME_ALLOW_DIR
  - rename() to directory is non-UNIX standard
    - But, it is VMS standard behavior
    - Example:
      - **`rename (file.ext,logname)`**
      - Where:
      - **`logname = [dir.subdir]`**
      - Results in:
      - **`[dir]subdir.ext`**
      - This happens because logname gets translated to a file because rename to a directory is not allowed
  - This switch restores the VMS behavior
      - **`rename (file.ext,logname)`** → **`[dir.subdir]file.ext`**

A word about rename()

- DECC$RENAME_NO_INHERIT should have been called RENAME_UNIX_COMPATIBLE
  - DECC$RENAME_NO_INHERIT causes UNIX compliant behaviors to be enforced –

- When DECC$RENAME_NO_INHERIT is enabled, DECC$RENAME_ALLOW_DIR is ignored.

7.3-2 CRTL - Tips
Enhanced access()

- access() enhanced to also check ACLs
  - DECC$ACL_ACCESS_CHECK
  - Uses $checkpro system service
  - Eventually need to add similar capability to stat() and other APIs – not done yet though

V8.2

Planned UNIX Portability Features

V8.2 New Features

- File Locking functions
  - X/Open file locking synchronization in threaded programs
  - flockfile(), ftrylockfile(), funlockfile()
  - clearerr_unlocked(), getc_unlocked(), getchar_unlocked(), feof_unlocked(), ferror_unlocked(), fgetc_unlocked(), fputc_unlocked(), putc_unlocked(), putchar_unlocked()
  - Integrated into all other stdio functions
- File-System Statistics functions
  - statvfs(), fstatvfs()

V8.2 New Features

- File Locking functions (con't)
  - Be careful - Brad said on previous slide:
    - "Integrated into all other stdio functions"
  - Mis-use of these new functions could result in deadlocks with the stdio functions!

V8.2 New Features

- File-System Statistics functions
  - statvfs(), fstatvfs()

V8.2 New Features

- Standard compliant stat structure
  - st_dev and st_rdev fields
    - Declared (char *) on VMS –
    - Declared int in X/Open
  - New feature test macro - _USE_STD_STAT
    - Provides the X/Open compliant definition of st_dev and st_rdev
    - Also provide st_blksize and st_blocks
      - X/Open fields that were previously not included
  - This fixes a very frequently seen porting problem

V8.2 New Features

- fcntl() function – added F_SETFL and F_GETFL
  - Two previously unimplemented function options
  - Set and get flie status flags…
- Stream oriented pipes
  - popen() creates a CR/LF oriented pipe
  - UNIX pipes do not have such record control
  - DECC$POPEN_NO_CRLF_REC_ATTR will cause popen() to create a UNIX style pipe
  - More compatible with UNIX expected behaviors

V8.2 New Features

- socketpair()
  - API to create a pair of connected sockets()
  - Requires underlying TCP/IP support
    - TCP/IP Services V5.5 required
- Performance enhancements for stat
  - Ongoing evaluation of C RTL performance,
  - Some minor improvements in stat – may be significant for file intensive applications

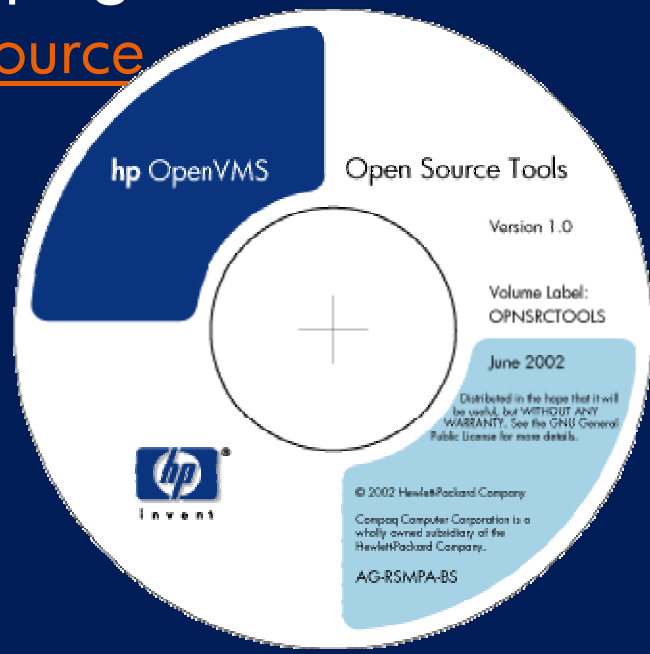
GNV

GNV – GNU's Not VMS

- Implementation BASH shell
- Shell Utilities
  - Simple commands: cat, ls, rm,
  - Application development utilities:
    - make
    - gawk
- Unix (like) environment
  - Root directory
  - Unix-style file specs
- HP Updates include:
  - ODS-5 file system support
  - Additional utilities ported and included
  - Packaged as a HP-branded PCSI kit

Finding GNV

- GNV project website
  - http://gnv.sourceforge.net
- OpenVMS website – OpenSource page
  - http://h71000.www7.hp.com/opensource
- OpenVMS kit – OpenSource CD
  - OpenVMS V7.3-1; V7.3-2…



Installing GNV

- PCSI product installation
  - Run the **PCSI-DCX_AXPEXE** file then:

```
$ PRODUCT INSTALL DEC-AXPVMS-GNV-V0105-004-1.PCSI
```

- Requires ODS-5 disk
  - If not system disk, specify target at install time:

```
$ PRODUCT INSTALL /DESTINATION=ddccuu: GNV
```

- Add startup file sys$startup:systartup_vms.com
  - **sys$startup:gnv$startup.com**
- Optionally, add login file to sys$manager:sylogin.com
  - **gnu:[lib]gnv_setup.com**
  - Or add it to your own login.com

Why ODS-5?

- Enables Unix-style filenaming
  - Also compatible with Microsoft Windows
- Filenames with funny characters, multiple dots

```
tar-1.13.25.tar.gz
This,Is@a#funny$filename%Dot.txt
```

- Directories with multiple dots
- Deep directories
- Optionally implements
  - Hard links
  - File access dates

Configuring

- User and/or data disks should also be ODS-5
  - **`SET VOLUME ddnnn:/STRUCTURE=5`**
- Enable hardlinks
  - **`SET VOLUME ddnnn:/VOLUME_CHAR=HARDLINKS`**
  - Not always necessary, but recommended
  - Required for some CONFIGURE scripts

GNV tips and tricks

- **$ SET PROCESS/PARSE_STYLE=EXTENDED**
  - Enables preservation of case in DCL commands
  - Also affects BASH
  - You'll need this for running CONFIGURE scripts

- **$ SET PROCESS/CASE=SENSITIVE**
  - If you really need it
  - Not generally needed
  - Enables full case sensitivity of file names
  - I don't recommend this, unless you need it
  - Some VMS applications won't take mixed/lower case files

GNV tips and tricks

- Define DECC$PIPE_BUFFER_SIZE 65000 to maximize pipe capabilities
  - Most configure scripts will need this set.
  - Notice we didn't say 65535?
    - There is an edge case bug we ought to fix some day

BASH Shell

- A shell is a CLI (Command Language Interpreter)
- BASH is the Bourne Again SHell
- Case sensitive (even if filenames aren't)
- Variables
- Conditionals (if… then… else…)
- Looping constructs (for, while, until)
- Program execution (no RUN command… just name the file to run)
- There is a BASH Reference Manual in the GNV kit
  - Its BASH V2. Software is V1.
  - Also, search the web for documentation.

GNV – Try it…

- To encourage you to try GNV, the next few slides provide some "ice breakers" to help ease you into using GNV and bash
  - How to start it
  - How to stop it
  - How to get help
  - Other important points

BASH – Introductory Commands

- Start BASH

```
$ bash
bash$
```

- $ HELP
  - More on "man" and help later

```
$ bash man
What manual page do you want?
bash$
```

- Exiting BASH
  - exit
  - ^D (crtl/d)
  - Not ^Z

```
bash$ exit
exit
$
```

BASH - HELP

- Where's the HELP command?
- Most commands support either -h or - -help

  `ls --help`
  - Give brief synopsis… Mostly good as a reminder of options
- Most commands have manual pages

  `man ls`
  - Detailed documentation
- man invokes "less"
  - Scroll with arrow buttons, space bar advances page
  - Use "q" to quit.

Interacting with VMS
Dealing with "$" in BASH

- Using VMS logical names, for example sys$login:

```
bash$ ls sys$login
ls: sys: no such file or directory
bash$
```

- $ character indicates variable substitution - Needs to be escape encoded – try this:

```
bash$ ls sys\$login
ls: sys$login: no such file or directory
bash$
```

- Getting closer – now it is looking for file or directory named "sys$login" – How do we get it to resolve the logical?

Interacting with VMS - logical names

- VMS path: **`DEV:[DIR.SUBDIR]FILE.EXT`**
  - Roughly equivalent to:
- UNIX path: **`/dev/dir/subdir/file.ext`**
- Logicals need to be the "device" to get interpreted:
  - "SYS$LOGIN" would be "/SYS\$LOGIN":

```
bash$ ls /sys\$login
file.txt  work
bash$
```

Interacting with VMS - logical names

- Kits used in future exercises are in **SYS$SYSDEVICE:[COMMON.KITS]** – how do we represent that in bash?

```
bash$ ls /sys\$sysdevice/common/kits/
barcode-0.98.tar.gz  cpio-2.5.tar.gz  m4-1.4.tar.gz
bash$
```

Interacting with VMS

- Can still use DCL!
  - BASH will pass unrecognized commands to DCL

```
bash$ dir UNXPRT\$DKA0:[USERS.USERX]

Directory UNXPRT$DKA0:[USERS.USERX]

.bash_history;1    file.txt;1      work.DIR;1

Total o 3 files.
bash$
```

Editors

- vi
  - GNV ships VITPU
    - TPU program that looks like vi
    - Warning: To exit:
      - To quit: `:q!`
      - To write output and exit: **<ESC>ZZ**
    - **<ESC>** is always a good idea
    - Not going to teach vi today!

- TPU/EDT
  - GNV is still VMS. You can always use VMS utilities

```
bash$ edit /tpu file.txt
```

GNV Updates

- New since VMS 7.3-2 shipped (available now):
  - gnutar
  - env, printenv
  - bug fixes – some critical to success of configure scripts
  - IPF port completed
    - GNV developer now develops on Itanium first, and ports back to Alpha
- Coming in late 2004
  - patch
  - file
  - vim (real vi editor instead of current TPU script)

GNV and IPF

- IPF Port complete – Negligible effort!
  - Available at E8.1 time frame
  - There is a release for IPF field test
  - There will be a release for V8.2.
- Development is now done on IPF first, and then ported back to Alpha.

Is GNV for real?

- From the GNV developers list (July 2003):
  - "GNV is working better and better. I could "./configure" and "make install" the following packages (sometimes with little hacks):
    - mktemp 1.5
    - hostinfo 2.2
    - patch 2.5.4
    - yacc 1.9.1
    - flex 2.5.4
    - bison 1.35"
  - "Recently I gave a try at making a few unix tools I had troubles building in the past, under the latest GNV bash, and got surprisingly further along, than in the past"

Futures

Symbolic Links (tentative)

- Initially planned V8.2
  - Probably V8.2+ (TBD)
- symlink(), readlink(), unlink(), lchown() & lstat() APIs
- POSIX pathname processing in RMS
  - Including POSIX root
  - Simplifies a complex part of C RTL
- NFS Integration
- DCL management commands and integration

V8.next features (tentative)

- Semaphore family of APIs
- Byte Range Locking
- Asynch I/O
- Other APIs TBD

Future Releases Contents (tentative)

- fork()
  - New system service $CLONE_PROCESS
  - IR complete
  - Functional Requirements in progress
  - Likely 8.4 for complete functionality
    - Some pieces in 8.3
  - Interested in customer/partner requirements

- UNIX I/O
  - aka “forkable-IO” or “shared stream files”
  - Parallel requirement for fork()

Do you port from UNIX to VMS?

- If so…We want to know…
- What can we do to make it easier?
  - Near term and long term
- Does lack of fork() impact you?
  - Interested in helping us define the requirements?
- Something else have a large impact?
- Not a developer?
  - Please forward this presentation to your developers

Contacts

- OpenVMS UNIX Portability Tech Lead:
  Paul.Cerqua@hp.com
- OpenVMS UNIX Portability Business Manager:
  Leo.Demers@hp.com

- http://h71000.www7.hp.com/portability/index.html


hp
invent