UNIX Portability

Jim Lanciani - Manager
OpenVMS Security, Application Integration and Network Labs
September 27th 2005

Agenda

- What is UNIX Portability?
- UP features in V8.2
- Future UP features
- POSIX pathname processing and symbolic links
- UP success story

What is UNIX Portability?

What is UNIX Portability?

- An initiative to ease the porting of applications from UNIX and Linux to OpenVMS
- New UNIX-like APIs and utilities
- Integrated operating system support for UNIX features
- Heavily utilized by HP to develop layered products for OpenVMS
- Used by HP to improve Java performance

Goals of UNIX Portability

- 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 or interoperability issues
- Make the cost of porting to OpenVMS equal or comparable to porting from one "flavor" of UNIX to another (HP-UX, Solaris, AIX, Linux)
- Retain native OpenVMS behavior as the default
- Support ODS-5
- Support Alpha and Itanium

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

UP features in V8.2

UP features in V8.2

- C RTL features
- GNV utilities
- Other features

C RTL features in V8.2

- File pointer locking APIs
  - flockfile(), ftrylockfile(), funlockfile()
- Device information retrieval APIs
  - statvfs(), fstatvfs()
- Socket creation API
  - socketpair()
- Stream I/O for pipes
- File status flag operations
  - F_SETGL and F_GETGL in fcntl()
- Standard stat structure
- 64-bit glob() support

GNV utilities in V8.2

- GNUtar
  - Archive and back up files
- patch
  - Apply a set of differences to a source file
- du
  - Display amount of disk space used by the contents of a directory
- env, printenv
  - Display environment variables

GNV utilities in V8.2 (continued)

- file
  - Display file type
- su (SuperUser)
  - Become a different user
- vim (“vi improved”)
  - Invoke a common UNIX text editor
- which
  - Find an executable in the current user’s path
- Improvements to cc, gcc, ld

Other UP features in V8.2

- Pshared (process-shared) objects
  - Pshared mutexes
  - Pshared condition variables
- NFS support

Future UP Features

Future UP Features

- Byte-range locking *
- Posix Pathnames *
- Symbolic Links *
- Shared-stream I/O
- Semaphores

* OpenVMS V8.3

Byte-range locking

- Coordinate shared access to a file
  - Can lock a select region of a file
- Implemented in the C RTL
  - fcntl() API
  - F_GETLK, F_SETLK, F_SETLKW options
- Can be used against any file type
  - Accessed via a file descriptor
- Functional across processes and across a cluster
- Locks are advisory; to be effective, processes must agree to cooperate

POSIX pathname processing and symbolic links

What is a symbolic link?

- A symbolic link is a directory entry that associates a name with a text string
- The text string is interpreted as a POSIX pathname when accessed by certain services
- It is implemented on OpenVMS as a file of organization SPECIAL and type SYMBOLIC_LINK

Example of symbolic link creation

- New DCL qualifier /SYMLINK for CREATE
- Example:
  - $ CREATE/SYMLINK="a/b.txt" link_to_b.txt
  - $ DIR/DATE link_to_b.txt

```
Directory DKB0:[TEST]

LINK_TO_B.TXT -> a/b.txt
                              8-MAR-2005  16:46:45.88
```

Example of symbolic link access

- Now we can type the file through the link:

  -
    ```
    $ type link_to_b.txt
    This is a text file
    $
    ```

- In this example, RMS noticed the input file was a symbolic link, read its contents and interpreted those contents as a POSIX pathname

RMS support for symbolic links

- sys$open() opens the target file pointed to by the symbolic link
- sys$create() creates the file pointed to by the symbolic link
- sys$search() returns the DVI and FID of the target file; DID is zero; resultant name is that of the symbolic link and not the target file
- Setting flag NAML$V_OPEN_SPECIAL cause sys$open() and sys$search() to not follow the symbolic link

C RTL support for symbolic links

- Six newly documented APIs:
  - symlink() -- create a symbolic link
  - readlink() -- read the contents of a symbolic link
  - unlink() -- delete a symbolic link
  - realpath() -- return a direct pathname from the root
  - lchown() -- change the owner of a symbolic link
  - lstat() – return attributes of a symbolic link
- Other APIs that accept pathnames recognize symbolic links

POSIX pathname processing

- To provide a consistent programming environment, developers must be able to use POSIX pathnames through OpenVMS interfaces such as the C RTL and system services
- Other standard POSIX features (system-wide root, mount points, current working directory, version limits) must also be provided
- Rules must be devised to deal with the differences between POSIX pathnames and Open VMS file names

POSIX pathnames for RMS and DCL

- Issue: The POSIX name-separator '/' character has a different meaning to DCL (as a qualifier indicator)
- Quoting a pathname allows us to pass the pathname through DCL (since quoted strings are already allowed in DCL for DECnet)
- Adding a prefix to the pathname allows RMS to recognize the string as a POSIX pathname
- Format: "^UP^pathname"
- Example: a/b.txt becomes "^UP^a/b.txt"
- NOTE: The C RTL and GNV will accept pure POSIX pathnames (no need for the ^UP^ quoted format

DCL POSIX pathname example

```
$ cc "^UP^a/hello.c" /obj="^UP^a/hello.obj"
$ lin /exe="^UP^a/hello.exe"  "^UP^a/hello.obj"
$ dir [.a]hello.*

Directory DKB0:[TEST.A]

HELLO.C;1           HELLO.EXE;1          HELLO.OBJ;1

Total of 3 files.
$
```

System-wide root

- Available for use with POSIX pathnames
- New ROOT keyword for SET
- One root allowed per system
- Example:

```
$ sho def
 DKB0:[TEST]
 $ set root dkb0:[test]
 %SET-I-PSXROOSET, system POSIX root set to
DKB0:[TEST]
 $ show root
 DKB0:[TEST]
 $
 $ type "^UP^/a/b.txt"
 This is a text file
```

Mount points

- Allows the crossing of volumes from the root
- New mnt and umnt utilities
- Example:

```
$ dir dkb100:[newtest]
 NEWDIR.DIR;1
 Total of 1 file.
 $ mnt dkb100:[newtest]  /a/mnt
 $ dir DKB0:[TEST.A.MNT]
 NEWDIR.DIR;1
 Total of 1 file
 $
```

Can I use this now?

- Yes, you can!
- An SDK is available for V8.2 that contains the images you need to use symbolic links and POSIX pathname processing
- Available through OpenVMS field test distribution website
- Please review the documentation contained in the SDK
- Send email to openvmsfieldtestops@hp.com and ask to be included in the Symlinks field test effort

UNIX Portability Roadmap

2005 | 2006 | 2007 | 2008 | 2009

OpenVMS V8.2
CRTL
- File Lock APIs
- statvfs/fstatvfs
- Stnd stat struct

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

OpenVMS V8.3

- Symbolic Links
- Byte range locking
- GNV update

OpenVMS V8.x

- Semaphores
- Shared write for stream files
- ioctl()
- GNV update
- Shared memory APIs

OpenVMS V8.2
Symbolic Links SDK
- CRTL APIs
- DCL support
- NFS support (in future TCP/IP ECO)

UP Success Story

Stalker Software, Inc.
CommuniGate Pro
- *Advanced Communications Server*

- The top award winning, high performance messaging platform
- Provides data (email and Groupware) and Voice communications
- Secure & rock-solid scalability on HP Integrity OpenVMS Clustered platform
- Used by the largest Tier1 service Providers and Enterprise customers
- Multi-language support for Europe, Japan, China, and Latin America

Business needs

- Provide customers integrated voice and data communication system with high security, high availability & scalability
- Integrate with industry standard clients/front ends such as Outlook
- Provide customers low operations & administration costs

Solution

- Port CommuniGate Pro application to the HP OpenVMS operating environment

Customer benefits

- **Increased mobility with:**
  - One secure In-Box for email, calendar & voice
  - Infrastructure prepared for SoftPBX
- **Increased value:**
  - Industry standard UI
  - Low TCO compared to other, less secure messaging systems
  - 99.999% up time
  - 1Admin per 100K users

Security, performance, scaling, and clustering

"Stalker Software is the technology leader in messaging and collaboration solutions for a multitude of operating systems. Our CommuniGate Pro customers demand security, performance, scaling, and clustering for 5 nine uptime. OpenVMS on Integrity servers provides this exceptionally well. We worked with HP on the porting project of OpenVMS to Integrity servers, and found their UNIX to OpenVMS toolkit and the HP engineering team support to be the best we have seen in the industry. We already have several customers who are preparing implementations of the CommuniGate Pro Dynamic Cluster on OpenVMS for Integrity servers."

–Vladimir Butenko
CEO
Stalker Software, Inc.

Contact info

- Gaitan D'Antoni (OpenVMS Technical Directions)
  - Gaitan.dantoni@hp.com
- Paul Cerqua (UP Project Leader)
  - Paul.cerqua@hp.com
- John Ferguson (UP Program Manager)
  - John.ferguson@hp.com
- Jim Lanciani (UP Engineering Manager)
  - Jim.lanciani@hp.com

Thank you !!!

i n v e n t

Following are slides that provide details not covered in the abbreviated UP presentation

Example of symbolic link access

- Assume file being referenced does not exist:

  -
    ```
    $ type link_to_b.txt
     %TYPE-W-OPENIN, error opening
    DKB0:[TEST.A]LINK_TO_B.TXT; as input
    -RMS-E-FNF, file not found
    $
    ```
- Create the missing file through the link:

  -
    ```
    $ create link_to_b.txt
     This is a text file
     Exit
    ```

File system support for symbolic links

- No file system changes until V8.3
- Access subfunction will recognize a symbolic link and convert write access to read access
- Create subfunction will recognize a symbolic link and convert the create to a read access
- For wildcard directory search, lookup will match either a filename ending in ".DIR;1" or a filename which is a symbolic link

Current working directory

- Similar to OpenVMS default directory
- May not be a search list
- Must exist
- Example:

  -
    ```
    $ SET DEFAULT "^UP^/a/mnt"
    $ SHOW DEFAULT
    DKB100:[NEWDIR]
    $
    ```

C RTL and GNV

- The C RTL and GNV will accept pure POSIX pathnames (no need for the ^UP^ quoted format)
- DECC$POSIX_COMPLIANT_PATHNAMES controls how input pathnames are interpreted
- File versions are treated as on UNIX; only one version of a file will be deleted on a delete operation; no new version of a file is created on a create operation

File naming

- POSIX allows filenames “a” and “a.” in the same directory
- A file created through GNV and the C RTL that does not have a “.” or that ends in a “.” will have an additional “.” appended to its name to ensure uniqueness
- To allow POSIX filename “a.DIR” and directory “a” to co-exist in the same directory, GNV and the C RTL will append a “.” to a filename ending in “.DIR”

Shared-stream I/O

- Shared-write access to stream files
  - Standard UNIX I/O default
- Applicable across the nodes of a cluster
- Can be used on any stream file type
- Accessible via the C RTL
  - Set per file by an option on file open APIs
  - Set per process by C RTL feature switch

Semaphores

- Implemented via the C RTL
- POSIX semaphores
  - sem_open(), sem_post(), sem_wait(), etc.
- System V semaphores
  - semop(), semctl(), semget()
- Thread-aware