OpenVMS Open Systems Update

Andy Goldstein

OpenVMS Engineering, HP

Unix Portability features in V8.3

Unix Portability features in V8.3

- C RTL features
- GNV utilities
- Other features

V8.3 C RTL Features

- Symbolic links and POSIX pathname processing
- Byte-range locking
- Encryption routines
- Other changes

POSIX Compliant Pathnames

POSIX pathname interpretation

- 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 OpenVMS 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"**

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
- Example:

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

System-wide root (continued)

- The name after the opening "/" in an absolute pathname must exist in the root directory
  - Example: For "/a", "a" must be a directory, file, symlink, etc. that resides in the root directory
- Logical names are **not** supported in V8.3

Mount points

- Allows the crossing of volumes from the root
- New mnt and umnt utilities (packaged with GNV)
- 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
    $

Current working directory

- Similar to OpenVMS default directory, but with important differences
  - If the default directory is a search list, the current working directory is the first directory in the list
  - The current working directory must exist
- Example:
  - $ SET DEFAULT “^UP^/a/mnt”

    $ SHOW DEFAULT

    DKB100:[NEWTEST]

    $

Current working directory (continued)

- Used with relative pathnames
  - Pathnames that do not begin with a "/"
  - Example: if relative pathname is "a/b", RMS will look for "a" in the current working directory

C RTL and GNV

- The C RTL and GNV will accept pure POSIX pathnames (no need for the ^UP^ quoted format)
- This is an alternative to the standard C RTL UNIX pathname features

C RTL and GNV (continued)

- DECC$POSIX_COMPLIANT_PATHNAMES controls how input pathnames are interpreted
  - 1 = UNIX-only
  - 2 = leans UNIX (unless pathname contains brackets or ends with a colon and passes sys$filescan())
  - 3 = leans OpenVMS (unless pathname contains a slash)
  - 4 = OpenVMS only
- Modes 1 and 4 are not recommended due to interactions with other shareable libraries and utilities

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”

Other support for POSIX-compliant pathnames

- Most DCL commands/utilities accept POSIX-compliant pathnames with the ^UP^ format
- POSIX-compliant pathnames with the ^UP^ format can be used in a linker options file
- Quotes within a POSIX-compliant pathname should be doubled to be recognized
  - Example: a”b is expressed as “^UP^a””b”

$parse() results with POSIX-compliant pathnames

- When presented with a POSIX-compliant pathname, sys$parse() returns the following results:
  - node is null
  - dev is "^UP^
  - dir is all characters following the dev string up to and including the final /
  - name is all characters following the dir up to and not including the final .
  - type is all characters from the final . up to and not including the closing "
  - Version is "
- Concatenating these components together results in a proper POSIX-compliant pathname

$parse() results with POSIX-compliant pathnames (continued)

- Example: given "^UP^/a/b/c.d.txt"
  - node is null
  - dev is "^UP^
  - dir is /a/b/
  - name is c.d
  - type is .txt
  - version is "

Devices and POSIX-compliant pathnames

- /dev/null is supported
  - New utility "special" will create a file called "null" in the "dev" directory under the root
  - File is a character special file (a file organization of "Special: character")
  - Upon encountering the file, RMS will redirect to the null device
- Other devices in /dev can be supported in a similar way, but we have no current plans to do so

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
- Symbolic links are also known as “symlinks”

Symlink Interpretation

- If a symbolic link contains an absolute pathname, the path is followed from the system root, regardless of the location of the symbolic link
- If a symbolic link contains a relative pathname, the pathname is relative to the directory in which the symbolic link was found
  - Example: if symbolic link “a” is in directory “b” and contains the path “c/d”, “c” is looked for in directory “b”
- A symbolic link is interpreted as either a directory or a file, depending on how it is referenced

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

- 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:
  - $ create [.a]b.txt

    This is a text file

    Exit

Example of symbolic link access (continued)

- 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

Alternate example

- Create a symbolic link
  - $ CREATE/SYMLINK="c/d.txt" link_to_d.txt
- Assuming the file being referenced does not exist, create the missing file through the link:
  - $ create link_to_d.txt

    This is another text file

    Exit
- Now we can type the newly created file:
  - $ type [.c]d.txt

    This is another text file

    $

Symbolic link to a directory

- Create a file
  - $ CREATE DKB0:[TEST2.B]X.TXT
  - This is a text file in another directory
  - ***Exit***
- Create a symbolic link to its directory
  - $ CREATE/SYMLINK="../test2" link_to_test2
- Reference the directory
  - $ TYPE [.link_to_test2.b]x.txt
  - This is a text file in another directory

RMS support for symbolic links

- Directory path follows symbolic links
- sys$open() operates on the target file pointed to by the symbolic link
- sys$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

Other support for symbolic links

- All DCL commands/utilities that accept filenames also accept symbolic links
  - Most follow the symbolic link by default
  - Exceptions to this are BACKUP, DELETE, PURGE, RENAME
  - Options available on COPY, DIRECTORY, DUMP, SET FILE
  - /SYMLINK operates on the symbolic link itself
- New lexical functions F$READLINK and F$SYMLINK_ATTRIBUTES are available
- Pre-V8.3 versions of OpenVMS will treat a symbolic link as a file of organization SPECIAL and will not follow the link

Other V8.3 C RTL Features

Byte-range locking

- 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
- 4 GB limit
- New privileged image: DECC$SHRP

Encryption routines

- crypt()
  - Encrypts an input string
  - Note: algorithm has nothing to do with OpenVMS password encryption
- setkey()
  - Sets an encoding key to be used with encrypt()
- encrypt()
  - Encrypts an array in-place using key generated by setkey()
  - Note: algorithm has nothing to do with OpenVMS password encryption

Other C RTL changes

- fchmod()
  - New function that changes the mode of a file that is specified with a file descriptor
- confstr()
  - Existing function that returns system configuration information
  - New symbols supported include:
    - _CS_MACHINE_IDENT
    - _CS_PARTITION_IDENT
    - _CS_MACHINE_SERIAL

Other OpenVMS V8.3 UNIX Portability Features

GNV utilities

- Better handling of pipes and subprocesses in bash
- Behavior of cc is now more UNIX-like, particularly for certain config scripts

Other features

- NFS improvements
- Perl has been updated and made POSIX-aware (Note: this kit is not in general release yet)
- terminate() can now be called from any thread in a C++ program (previously, terminate() was only callable from the default thread)

References

- OpenVMS web site:
  http://www.hp.com/go/openvms
- UNIX Portability
  http://h71000.ww7.hp.com/portability/index.html
- CRTL Reference Manual
  http://h71000.ww7.hp.com/doc/83final/5763/5763pro.html
- GNV Page – OpenVMS
  http://h71000.ww7.hp.com/portability/GNV.html
- GNV Sourceforge
  http://gnv.sourceforge.net/

Contact info

- Mike Boucher (C RTL Project Leader)
  - Michael.R.Boucher@hp.com
- Gaitan D'Antoni (OpenVMS Technical Director)
  - Gaitan.Dantoni@hp.com
- Leo Demers (UP Program Manager)
  - Leo.Demers@hp.com
- Karl Puder (GNV)
  - Karl.Puder@hp.com

TCP/IP Services in OpenVMS

Supported Versions & ECO's

OpenVMS VAX V7.3	TCPIP V5.3 ECO 4
OpenVMS Alpha V7.3-2	TCPIP V5.4 ECO 6 (ECO 7 coming soon)
OpenVMS Alpha V8.2 OpenVMS Integrity V8.2-1	TCPIP V5.5 ECO 2 or TCPIP V5.6 ECO 1
OpenVMS V8.3 (Alpha and Integrity)	TCPIP V5.6 ECO 2 (TCPIP V5.5 unsupported)

TCP/IP Services V5.6

TCP/IP Version 5.6

- Shipped with OpenVMS 8.3
- OpenVMS Alpha and Integrity
- NFS server returns on Integrity
- NFS client TCP transport
- DNS/BIND 9 resolver and v9.3 server
- DNSsec
- NFS symbolic links
- NTP security update including SSL, AutoKey
- SMTP multi-domain zone
- SSH upgrade with Kerberos
- IPv6 support for printing
- FTP performance boost for VMS Plus
- Updates to TCPIP$CONFIG (Interface menu)
- Improved management utilities (such as ifconfig)
- PPP serial-line support returns

Post V5.6 – IPsec Support (as an EAK)

BIND 9 Resolver and Server

- BIND 9.3.1 for resolver and server
  - Resolver in TCPIP V5.5 was based on BIND 8
  - Server in TCPIP V5.5 was based on BIND 9.2.1
- BIND resolver
  - Lookups over IPv6
  - New ASCII configuration file (supplements existing one)
  - Improved thread support in getaddrinfo() and getnameinfo()
- BIND server
  - Includes critical updates to DNSSEC (signed zones)
  - Aligns DNSSEC with current RFCs and industry practice

NFS Client TCP Support

- TCP transport for NFS (previously server-only)
  - Important for WAN access
  - Offers robust flow control and retransmission behavior
  - Friendly to tunneling and port forwarding

NFS Symbolic Link Support

- A symbolic link is simply a link to another file
- When accessed, the target file is used automatically
- Deletion of the link has no effect on target file
- Links can span disks and even systems with NFS support
- NFS server must be able to create and recognize links
- NFS client must properly create, detect and follow links
- Shipped with OpenVMS V8.3
  - More updates and refinements already underway

NTP Security Update

- Security updates from University of Delaware (UDel) NTPv4 (Version 4.2.0)
- NTP 4.2 AutoKey cryptography, using SSL
  - AutoKey is based on public key cryptography
  - Provides for secure server authentication, packet integrity, resistance against clogging and replay attacks, spoofing, and protection against masquerade.
  - Uses the OpenSSL crypto library
  - Detailed configuration steps in an Appendix of the Release Notes
  - Existing private key mechanism with MD5 remains available

SSH Upgrade with Kerberos Support

- Kerberos support is enabled for V5.6
- DCL help for SSH commands
- SFTP/SCP
  - Improved support for additional VMS file types
    - Most popular structures are now supported
    - No support yet for RMS Indexed files
      - (You can encapsulate them in a saveset or ZIP file)

TELNET Server Device Limit

- OpenVMS now supports large unit numbers
- Previous version (TCPIP V5.5) allowed units beyond 9999 for BG devices
- For V5.6, we added this support for TN devices

IPv6 Support for LPD and TELNETSYM

- Allows printer communication to use IPv6
- Needed for deployment of a mostly-IPv6 network

- Note: HP enterprise printers now support IPv6

Updated TCPIP$CONFIG (Interface Menu)

- Previous TCPIP$CONFIG.COM used outdated notion of cluster interfaces and one IP address per interface
- Improved configuration of multiple addresses
- Simplifies common task of changing IP address and/or hostname
- Additional information displayed to the user
- Manages both permanent database and active system
- Pseudo-interfaces continue to be stored internally

New Look of Interface & Address Menu

```
HP TCP/IP Services for OpenVMS Interface & Address Configuration Menu

Hostname Details:   Configured hostname=gryffindor-e0,  Active=gryffindor-e0

Configuration options:

  1   -  WE0 Menu (EWA0: Multimode 1000mbps)
  2   -  10.0.0.1/16       gryffindor-g0          Configured,Active

  3   -  BE0 Menu (EBA0: Unspecified 30000mbps)
  4   -  1.2.3.4/8         *noname*               Configured,Active

  5   -  IE0 Menu (EIA0: TwistedPair 100mbps)
  6   -  10.1.1.10/23      gryffindor-e0          Configured,Active

  7   -  IE1 Menu (EIB0: TwistedPair 100mbps)
  8   -  10.1.1.11/23      gryffindor-e1          Configured,Active
  9   -  10.1.1.10/23      gryffindor-e0          Configured,Active-Standby

  I   -  Information about your configuration

 [E]  -  Exit menu
```

Interface Menu

```
 HP TCP/IP Services for OpenVMS Interface WE0 Configuration Menu
 Configuration options:

         1  - Add a primary address on WE0
         2  - Add an alias address on WE0
         3  - Enable DHCP client to manage address on WE0

        [E] - Exit menu

Enter configuration option:
```

Address Menu

```
HP TCP/IP Services for OpenVMS Address Configuration Menu

      WE0 10.0.0.1/16 gryffindor-g0 Configured,Active WE0

 Configuration options:

         1  - Change address
         2  - Set "gryffindor-e0" as the default hostname
         3  - Delete from configuration database
         4  - Remove from live system
         5  - Add standby aliases to config database (for failSAFE IP)

        [E] - Exit menu

Enter configuration option:
```

TCP/IP Services for OpenVMS Pointers and Contacts

- HP OpenVMS home page:
  - http://www.hp.com/products/OpenVMS

- TCP/IP Contacts:
  - Product Management

    Leo.Demers@hp.com

  - Project Leader

    Mark.Hollinger@hp.com


invent