Boot Camp Highlights

Hans Bachner
b.it.co IT Consulting GmbH
Boot Camp Highlights

- ~170 participants
- ~25 first-timers

- pre-conference seminar
  - 2 tracks, 12 modules
- 83 sessions in 3 days
  - up to 5 tracks
- 2 keynotes / day
- Partner Roundhouse
- OpenVMS 40th anniversary celebration
Boot Camp Highlights

VSI Technical Update (Eddie Orcutt)

VSI Technical Achievements To Date

- Layered Product Releases: 65
- Open Source Releases: 12
- OpenVMS Releases: 6
- VSI Defect Repairs: 515
- New Features Since V7.3-2: 179
Boot Camp Highlights

VSI Technical Update (Eddie Orcutt)

![Bar Chart: Test Hours Per VSI OpenVMS Version]

- TK5BRY IA64
- V8.4-2L1
- V8.4-2
- V8.4-IH1

Next Release
Boot Camp Highlights

VSI Technical Update (Eddie Orcutt)

- x86 Hardware Roadmap
  - V9.0 – rackmount HPE G9/G10, Dell 12G/13G/14G
  - V9.2 – blades
  - HBA, NIC roadmap

- Storage Roadmap
  - Dell/EMC, Hitachi, HPE, Kove, Pure

(subject to change)
Boot Camp Highlights

x86 – Further Adventures in Booting and Dumping (Richard Bishop, Gary Newsted)

• UEFI based, graphical boot manager
• uses UEFI boot drivers to load a pre-populated memory disk from disk, CD-ROM, PXE, http, ...
• OpenVMS always boots from this memory disk!
• never ever write boot drivers, no more primitive file system so IPB.EXE can load what it needs from the system disk
x86 – Further Adventures in Booting and Dumping (Richard Bishop, Gary Newsted)

• dumping memory to the dump file currently uses the boot drivers & primitive file system
• on x86, at boot time a second VMS instance (dump kernel) will be loaded but not booted
• BUGCHECK will boot this instance, dump file can be written with standard VMS runtime drivers
VSI OpenVMS Alpha (Eddie Orcutt)

- why?
  - provide direct path to x86
  - provide OpenVMS parity on Alpha, Itanium and x86 when released
VSI OpenVMS Alpha (Eddie Orcutt)

- why a Performance Build (V8.4-2L2)?
  - EV6 provides additional instructions (BWX, FIX, ...)
    - BWX used in 10,027 out of 16,150 modules
    - FIX used in 994 out of 16,150 modules
  - leverage hints for Out of Order Execution
  - leverage Load Hit / Miss Prediction
  - Cache Management
  - and more...
Boot Camp Highlights

VSI OpenVMS Alpha Performance Build (V8.4-2L2)

• also useful with emulators
• CHARON-AXP (Stromasys) and vtAlpha (AVTware) emulate the EV6 instruction set
  • performance gain visible in emulators as well
  • other emulators not checked
Boot Camp Highlights

Alpha emulators

• Stromasys and AVTware were Boot Camp sponsors
• AVTware presented their support for solution for running vtAlpha and vtVAX in the Cloud
• Stromasys introduced the new CHARON-AXP+
Boot Camp Highlights

Alpha emulators

“Everybody has a testing environment. Some people are lucky enough to have a totally separate environment to run production in."

(unknown)
Boot Camp Highlights

Using virtualized Systems to Improve Preparedness and Operation (Robert Gezelter)

• duplicating production hardware is expensive
• training scenarios require setup and reproducible situations
• emulators are easy to rerun / roll back
• allow one to safely experiment
• train operators to handle rare, but critical cases
  • normal reboot, emergency reboot, ...
Password Management Enhancements (Doug Gordon)

- some sites have password (policy) requirements that OpenVMS can’t handle today
- password controls are scattered over various places
  - SYSUAF, SYSGEN, logical names
- new capabilities (via logical names):
  - minimum password lifetime
  - complexity rules
Boot Camp Highlights

Passwords – Best Security Practices Update (Lorin Ricker)

• new “Digital Identity Guidelines” from NIST (June 2017)
  • [https://pages.nist.gov/800-63-3/sp800-63b.html#appA](https://pages.nist.gov/800-63-3/sp800-63b.html#appA)

• suggestions:
  • use /FLAG=PWDMIX
  • use /PWDLIFETIME="0" (indefinitely!)
  • use /PWDMIN=8 for non-privileged users, 10..12 for privileged
  • consider secondary password for SYSTEM and other accounts with high visibility, use personal accounts
Boot Camp Highlights

Keith Parris: several sessions on performance and high availability

- Maximizing the Performance of your Cluster Interconnect
- Using Locking Performance Data to Solve Cluster Problems
- Best Practices for Multi-Site and DT OpenVMS Clusters
- Volume Shadowing Best Practices
Boot Camp Highlights

A history of technology

VMS: The First 40 Years (Andy Goldstein)

• 25-Oct-1977: OpenVMS and the VAX-11/780 are announced

• lots of technical details, stories, anecdotes, ...
VAXA Committee

- Gordon Bell
- Peter Conklin
- Dave Cutler
- Bill Demmer
- Tom Hastings
- Richie Lary
- Dave Rogers
- Steve Rothman
- Bill Strecker, chief architect
Initial VMS Design Team

By November, 1975...

- Dave Cutler, project leader
- Andy Goldstein
- Roger Gourd, manager
- Roger Heinen
- Dick Hustvedt
- Hank Levy
- Peter Lipman
- Trev Porter
Early Development

- Sept 1975  SRM Rev 1
- April 1976  April Task Force
- June-Aug  Detailed software design
- Sept 1976  Hardware simulator and initial system kernel
- April 1977  DCL and file system
- June 1977  Breadboard and first VMS timesharing
780 Prototype Power On

- October 25, 1977
- VAX-11/780
- VMS V1.0 Announced

- VMS V1.0 Shipped
- DECnet Phase II
- FORTRAN IV
- Up to 64 MB Memory
VMS Technology Highlights

- OpenVMS Calling Standard
- VMSclusters
- Symmetric Multiprocessing
- The Alpha Port
- OpenVMS Galaxy
- The Itanium Port
OpenVMS Calling Standard

- Fortran Module
- C Module
- Pascal Module
- Fortran I/O Library
- Common Math Library
- Pascal I/O Library
- VMS System Services
VMSClusters

VMS Becomes a Distributed Operating System

VMS Node

Storage Ctrl

VMS Node

Storage Ctrl

VMS Node
Clusters: The Lock Manager

Fully distributed implementation with no permanent master

- Distributed directory identifies master for a resource
- Lock ownership recorded by master and lock holders
- Master is the node with the most activity
- Automatic reconfiguration on node failure
The RISC Advantage

1-cpu Relative Throughput (Log Scale)

- AlphaServer 6400 5/360
  SPECint_rate95=64
  SPEC_int92=6551

- DEC 7000-610
  SPEC_int92=3180

- VAX 11/780
  SPEC_int92=23.717

- VAX 6000.410
  SPEC_int92=275e

- VAX 6000.210
  SPEC_int92=57e

- AlphaServer 6400 5/625
  SPECint_rate95=165

Port to Alpha

Rewrite:
- CPU support
- Boot code
- Some drivers
- Low level memory management
- Exception handling
- Math RTL

Keep:
- All major interfaces
Soft Partitioning – OpenVMS Galaxy

VMS instances cooperate to partition the hardware

- CPU and IOP assignment
- Memory allocation
- Shared memory
VMS instances cooperate to partition the hardware

- Resources can be reallocated
Port to Itanium

- Another 64-bit architecture, but...
- Different register conventions
- Intel calling standard
- Different privileged architecture
  - No PALcode
  - Different console / boot procedure
  - Different interrupt architecture
  - Different synchronization primitives
Port to Itanium

• Fortunately...
• 4 access modes
• Compatible memory protection features
• Memory atomicity no worse than Alpha
Port to Itanium

• Rewrite
  – CPU support
  – Boot code

• New
  – Interrupt & exception delivery in software
  – Emulation of interlocked instructions (queues, etc.)
  – EFI partition on system disk

• Redesign
  – Calling standard and condition handling
  – Object and executable file format
Port to Itanium

- Recompile
- 95% of base OS code recompiled without change
  - Biggest problem was “IF ALPHA” conditionals
- Binary translator also available
  - Even supports VAX translated images!
How to Build an Evolvable System

It begins at the beginning

• Start with a team of grownups
• Design with care
• Keep the team small
  – Initial VMS architecture came from 3 people
  – Entire VMS V1 team was 24 people
• Keep the pressure up
  – The first known “fact” about VMS was the schedule
  – Beware of creeping elegance
Modularity in VMS

- Dynamically loaded modules for all configuration dependent components
- Huge number of system models and devices supported over the life of the system
- Any VMS system disk will boot on any configuration of a particular architecture
- New hardware is supported with minimal effect on the rest of the system
Maintain Design Integrity

Causes of “software rot”

• Lack of design understanding
• Quick and dirty changes
• Changes that compromise the original design
• Functional extension without extending the original design
• Duplication of function
• Runaway complexity
Ready For the Next Adventure…

- X86 port
- New file system
- Ongoing modernization
- ...