OpenVMS hypervised

as-is

artedi e.U.
Agenda – I

• so what …is Hypervision?
• and why is it needed/used?
• Types of Hypervision
• why not OpenVMS?
• Processor Architectures
Agenda - II

- what else ...is needed?
- Emulation, bringing things together
- Types of Emulators
- who does it for VAX and Alpha (VMS)?
- Example, more detailed
- Life Demo
Hypervise, divide and conquer

- hyper … ancient greek for 'over'
- videre … latin word for 'to see'
- Hyper+visor … the 'overseer' / surveillant
- controlled distribution by Monitor Machines → VMMs
Reasons for hypervision

- Money $\rightarrow$ optimized use of HW expenses made
- Resources $\rightarrow$ optimized use of HW infrastructure
- Hardware evolves rapidly $\rightarrow$ miniaturization
- Clock rates stall $\rightarrow$ rejected heat
- Parallelism has its limits $\rightarrow$ synchronisation issues
General Models

- Divide in Groups (Virtual Systems → OSes)
  - which use their own groups (Processors/Interfaces)
- Two types of Hypervisors/VMMs
  - Type-1 native or Bare Metal
  - Type-2 hosted
Hypervisor, Type-1 Bare Metal

- doesn‘t need an underlying Software Infrastructure (Host OS)
- Kernel which supports HW Infrastructure + Management Interface
- allows controlled distribution of given Infrastructure
- is used by diverse higher level Infrastructure aka Guest OS Systems (like OpenVMS)
Hypervisors Type-1, practical use

- Hardware (X86 most likely)
- 8 Cores
- the Hypervisor
- 1st Virtual Machine (VM)
- 2 Cores assigned
- OS installed
- others to follow
Hypervisor, Type-2 hosted

- needs supporting Software Infrastructure → fully-fledged OS
- does share HW Investment and electrical Power resources
- has to share with other 'normal' applications
- is fully dependent on hosting OS
  - Patches
  - Security Updates
  - License prolongations
  - ...

Maier 2016
Hypervisors Type-2, practical use

- Hardware (X86 most likely)
- 8 Cores
- the Host OS (grabs Cores)
- the Hypervisor (an application)
- 1st Virtual Machine (VM)
- one Cores assigned
- OS installed
- others to follow
Why not for OpenVMS?

- most of the Hypervisors use x86 (AMD64, iE64, x64)
- VMS runs on VAX (CISC), Alpha (RISC) and Integrity (EPIC)
- Hypervisors neither translate nor emulate
- Type-2 Integrity based Hypervisors running on HP-UX (Integrity Virtual Machine)
- most customers want x86
Types of Processors

- **CISC ... Complex Instruction Set Computing**
  - PDP-11 (16bit), VAX (32 bit), x86

- **RISC ... Reduced Instruction Set Computing**
  - Alpha, MIPS, PA-RISC, Power(OC), SPARC, ARM (64bit)

- **EPIC ... Explicit Parallel Instruction Computing**
  - Integrity (Itanium) (64bit)

- **Hybrids ... CISC to RISC (Emulators ?)**
  - x86 starting with Pentium Pro
Emulate

- *aemulare…lat.word for 'to imitate'*
- System/Software which imitates another System…in some aspects
- Imitation by Transformation
- Emulators transform translated code for
  - different Processors with different 'vocabulary'
- Emulators are the supporting pillars for bridging
  - different Architectures with different Infrastructures
- transparent to the upper Layers (Operating Systems)
Emulators, types of

• Hardware Emulators
• ICE …In Circuit Emulator
• Printer Emulator → HP-PCL
• Terminal Emulator
• Processor Emulator
  • Alpha, VAX
  • HP 3000
  • SUN Sparc
Emulators, practical use

- original Hardware
- original Operating System + Apps (OS+)
- new Hardware
- Emulator
- transfer OS (still original)
- discard old Hardware
hype\textsuperscript{v}is\textsuperscript{e} \rightarrow emula\textsuperscript{t}e \rightarrow transla\textsuperscript{t}e

• Hypervisors
  • divert a given Infrastructure and assign it to upper Layers

• Emulators
  • ‘mask’ the underlying Infrastructure

• Translators
  • make HighLevelLanguages readable to the Processor
Hypervisor

Example
Desires; to fulfil

• keep OpenVMS based Applications alive
  • highly customized
  • high value
  • Mission Critical
  • Continuity Issue

• reduce Operating Expenses
  • converge Infrastructure
  • reduce space requirements
  • reduce power consumption
  • reduce costs (and complexity) of Hardware Maintenance
Obstacles; to overcome

- no Code Migration, due to
  - Source Code missing
  - Application Vendor doesn’t sell Migration
  - Application Vendor does sell, but is way to expensive
  - still risk of incompatibilities

- Hardware Infrastructure to maintain
  - Storage Arrays
  - TapeDrives
  - FC-Switches to connect to
Expectations to be met

- no software changes
- reliable & secure
- support modern infrastructure
- easy to manage
- performance
Solution; first choice → Bare Metal

- get new Hardware
- get and install Hypervisor
- hook emulated Systems onto
- transfer Systems currently in use
  - copy disk content
- continue with original SW (OS+Apps)
Bare Metal

- Windows
- Linux
- other Hypervisor

{ no cost, no maintenance, no sync time delays

- Alpha/VAX Reliability
- Single Point of Contact
Solutions; other Options (Type-2)

- use existing Hardware
- use existing Hypervisor
- install vtServer onto it (one to many)
- create virtual Alphas and VAXes (one to many)
- Install System(s)
- live migration
Solutions; and another one

- use existing Hardware
- use existing Hypervisor
- host other guests
- install vtServer as another guest
- create virtual Alphas and VAXes (one to many)
- Install System(s)
Sizing; CPU

• # Cores
  • virtual CPU = 1,5 Host CPU-cores

• Intel/ AMD

• Generation (Haswell)

• Frequency, the more the better (3GHz +)

• Hyperthreading
Sizing; Memory

per virtual Alpha/VAX:

• Virt.Mem + 25% +1 GB

\[ 1024 + (1024 \times 0.25) + 1024 = 2304 \]

• safety valve
Infrastructure; Transparency
Infrastructure; Flexibility
Networking; Options

- **Alpha**
  - DE435
  - DE450
  - DE500
  - DE600

- **VAX**
  - DELQA
  - DEQNA
  - SGEC
  - DEMNA

- Combine links
- VLAN support
- Bonding support
Licensing

- licensed via LicenseKeys
- presented via
  - License Dongles connected to USB Port
  - Network (TCP/IP) Link
- limited Timeframe or perpetual
- redundant License Keys to overcome SPOF
Licensing; distributed
<table>
<thead>
<tr>
<th>Product</th>
<th>Licensing</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlphaStation 200, 250, 255, DEC3000</td>
<td>vtAlpha-AS</td>
</tr>
<tr>
<td>AlphaServer 300, 400</td>
<td></td>
</tr>
<tr>
<td>AlphaServer 800, 1000</td>
<td>vtAlpha-BS</td>
</tr>
<tr>
<td>AlphaStation 500, 600, DPW, XP900, XP1000</td>
<td></td>
</tr>
<tr>
<td>AlphaServer 2000, 2100</td>
<td>vtAlpha-CS</td>
</tr>
<tr>
<td>AlphaServer 4000, 4100</td>
<td></td>
</tr>
<tr>
<td>AlphaServer DS10, DS15</td>
<td></td>
</tr>
<tr>
<td>AlphaServer DS20, DS25</td>
<td></td>
</tr>
<tr>
<td>AlphaServer 1200</td>
<td></td>
</tr>
<tr>
<td>AlphaServer ES40, ES45, ES47</td>
<td>vtAlpha-ES</td>
</tr>
<tr>
<td>AlphaServer GS80, GS160, GS320</td>
<td>vtAlpha-GS (Q2/2015)</td>
</tr>
</tbody>
</table>
# vtVAX, Licensing

<table>
<thead>
<tr>
<th>Model</th>
<th>Licensing Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAXstation II, GPX, 2000, VAXserver 3600/3900</td>
<td>vtVAXstation</td>
</tr>
<tr>
<td>MicroVAX II, 2000</td>
<td>vtVAX-128 (+ AC)</td>
</tr>
<tr>
<td>MicroVAX 3100 - 3900</td>
<td>vtVAX-256 (+ AC)</td>
</tr>
<tr>
<td>VAX 4000</td>
<td>vtVAX-512 (+ AC)</td>
</tr>
<tr>
<td>MicroVAX 3100</td>
<td>vtVAX-7000</td>
</tr>
<tr>
<td>VAX 4000</td>
<td></td>
</tr>
<tr>
<td>VAX 6000</td>
<td></td>
</tr>
<tr>
<td>VAX 6000 1 – 6 CPU, up to 3.5 GB memory</td>
<td></td>
</tr>
<tr>
<td>VAX 7000 1 – 6 CPU, up to 3.5 GB memory</td>
<td></td>
</tr>
</tbody>
</table>
Life Demo

- vtAlpha OVMS 7.3-2
- vtAlpha OVMS 8.4-2

Virtual Switch
still beats
Sieve of Eratosthenes

\[ \begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{array} \]