(Advanced) OpenVMS Performance Tips & Tricks

Guy Peleg
President
Maklee Engineering
guy.peleg@maklee.com

Place Yourself in the Hands of the Experts
Who we are

• What is Maklee?
  – US Based consulting firm operating all over the world.
  – Former members of various engineering groups at HP

• Among our customers are:
  – Verizon Wireless, Eli Lilly, AIG financial group, Volvo, M.O.L. America, ConEd, FDNY, France Telecom, IKEA, Navistar, Private Banks in Europe, Frankfurt Airport, ThyssenKrupp Steel, Tel-Aviv Stock Exchange, Hewlett Packard, Dow Jones Company, Bloomberg, NYSE and more...

• We specialize in:
  – Performance Tuning
  – Oracle & Oracle tuning (official Oracle Partner)
  – Platform migration
  – Custom Engineering

• Supported platforms: OpenVMS, HP-UX, Linux, Tru64, Solaris and AIX

Maklee provides guarantee for success for all projects
Basic Tuning Techniques

- OpenVMS V8.3-1H1
  - VMS831H1I_SYS-V0700
- SET RMS
- QUANTUM
- Resident Images
- Cache vs. No cache
- Fastpath
- Compile /optimize
- Hyper Threads
- PE data compression
- Gigabit Jumbo Frames
- SDA PRF
• “Keep looking below surface appearances. Don't shrink from doing so just because you might not like what you find.”

Colin Powell
• Virtual Address lookup IA64
  – CPU TB cache
  – VHPT
  – OpenVMS performs 3 level address translation walking the page tables.

• The VHPT is sized by SYSGEN parameter - VHPT_SIZE.

• Default value of 1 means allocate 32KB per CPU for the VHPT.
Oracle Batch job A

Elapsed Time in Minutes (less is better)

23% performance increase

VHPT = 1
VHPT = 2048
VHPT = 10000
• No good deed goes unpunished.

• High cost associated with invalidating large address space.
  – Oracle server process mapping large SGA

• May result in high MP Synch time during while invalidation is in progress.

• Processes may show up in RWSWP.

• Large VHPT not suitable for applications that frequently map large virtual address space for short period of time.

• In severe situations stop all CPUs on the system until condition clears up.
  – Shutting down Oracle database requires 25 minutes vs. 3.
Multiple Kernel Threads

- When running a threaded application, the threads manager creates one Kernel thread per CPU.
  - This happens regardless of the number of user threads in the application
- Kernel threads are execution engines for user threads
- The threads manager schedules user threads to run on an available kernel thread.
- Overhead is associated with managing multiple Kernel threads.
- A threaded application decides if multiple kernel threads should be enabled or disabled.

With kernel threads disabled – one execution engine is used for running the user threads.
Multiple Kernel Threads

- Evaluated the impact of disabling multiple kernel threads on a Java based benchmark.

- Single threaded Java program performing CPU intensive operation (encryption).

- SD32B, 32 CPUs, OpenVMS V8.3-1H1, Java 5.

- Used SET IMAGE to disable multiple kernel threads.
Encryption Test

<table>
<thead>
<tr>
<th>Accounting information:</th>
<th>MKT enabled</th>
<th>MKT disabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buffered I/O count:</td>
<td>103709</td>
<td>102399</td>
</tr>
<tr>
<td>Direct I/O count:</td>
<td>7279</td>
<td>7145</td>
</tr>
<tr>
<td>Page faults:</td>
<td>55739</td>
<td>52623</td>
</tr>
<tr>
<td>Peak working set size:</td>
<td>891216</td>
<td>841424</td>
</tr>
<tr>
<td>Peak virtual size:</td>
<td>2652928</td>
<td>2584064</td>
</tr>
<tr>
<td>Mounted volumes:</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Charged CPU time:</td>
<td>00:02:36.81</td>
<td>00:01:35.80</td>
</tr>
<tr>
<td>Elapsed time:</td>
<td>00:15:54.98</td>
<td>00:15:18.83</td>
</tr>
</tbody>
</table>

- 39% less CPU time
PC Sampling

Number of CPU cycles (in percent of total)
Less is better
CPU utilization during benchmarks. 15% reduction in CPU, no spikes!!

Java Garbage Collector
Data Encryption

• Business rules and data privacy regulations force more and more organizations to encrypt data stored on tapes.

• Starting with OpenVMS V8.3, OpenVMS can generate encrypted savesets.
  – OpenVMS supports various AES encryption algorithms, and various encryption key sizes.

• OpenVMS also supports the LTO-4 tape drive family.
  – LTO-4 tape drives support hardware encryption.

Which one would perform better?
encryption benchmark

- Customer benchmark comparing performance of:
  - Alphaserver ES80, 8 CPUs
  - EVA 8100
  - OpenVMS V8.3
  - 2gb fiber connection
  - LTO-3

  VS.

  - 4P/8C BL870c
  - EVA 8100
  - OpenVMS V8.3-1H1
  - 4 gb fiber connection
  - LTO-4
CPU utilization during backup
AlphaServer ES80 Throughput

LTO-3 throughput MB/sec
More is better
BL870 Throughput

LTO-4 throughput MB/sec
More is better
Data Encryption

• Always enable hardware data compression.

• Hardware encryption outperforms software encryption.

• Stronger encryption keys require more CPU resources.

• Depending on the storage sub system, the /IO_LOAD qualifier may improve performance of backup operations.

• The MSL tape library allows distributing backup across more physical tapes, increasing the throughput of the backup operation.
Mount

• More and more systems use large number of disk volumes.
  – Overcome VMS limitation of 1TB per volume.

• It is not unheard of to encounter systems with 100+ volumes.

• Mounting volumes is slow !

• Try parallelising the mount operation.
  – SPAWN is the easiest way
  – Writing a program calling $MOUNT is the fastest way
Mount

- BL870c.
- OpenVMS V8.3-1H1.
- DT cluster spread over 2 sites, 5KM apart.
- Booting the system required 15 minutes.
  - Sequentially mounting 100 shadow sets.
- After changing the startup script to mount all volumes in parallel, the system now boots in 1.5 minutes.
SYS$IO_PERFORM

- One of VMS’s best kept secrets.

- SYS$IO_PERFORM starts a fast I/O operation.
  - Developed as an alternative to SYS$QIO.

- Shortcut into the core of the I/O subsystem.
Fast Copy

- Fast I/O is the fastest way to copy data from one disk to another.

- The following shows the results of various tests copying 500MB file from one disk to another:

<table>
<thead>
<tr>
<th>CONFIG</th>
<th>METHOD</th>
<th>~DIO</th>
<th>CPU</th>
<th>ELAPSED</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVA3000</td>
<td>COPY</td>
<td>16156</td>
<td>02.71</td>
<td>00:12.94</td>
</tr>
<tr>
<td>LP9802</td>
<td>BACKUP</td>
<td>31288</td>
<td>00.72</td>
<td>00:14.06</td>
</tr>
<tr>
<td></td>
<td>BACK/BLO=65024</td>
<td>15911</td>
<td>00.35</td>
<td>00:12.49</td>
</tr>
<tr>
<td>GS1280</td>
<td>CONVERT</td>
<td>23529</td>
<td>08.97</td>
<td>00:15.37</td>
</tr>
<tr>
<td>VMS V8.2</td>
<td>FAST_IO_COPY</td>
<td>7845</td>
<td>00.22</td>
<td>00:07.34</td>
</tr>
</tbody>
</table>

Contact me off-line for a copy of the fast copy program.
C Vs. C++

• The C compiler uses a backend (code generator) provided by HP.

• The C++ compiler uses a backend provided by Intel
  – The C++ compiler knows how to use the Itanium advanced loads and speculative loads.
  – Allows the compiler to hoist fetches out of loops AND move fetches before stores that might impact them
  – Can be huge performance win for certain applications.

• Try compiling CPU intensive C routines with the C++ compiler.
  – We’ve seen ranges from the C compiler is 20% faster to Intel’s compiler is twice as fast.

YMMV !!
TCP/IP I/O post processing

• TCP/IP interrupts are handled by one CPU.
  – SHOW FASTPATH will display the TCP/IP CPU.

• Saturating the TCP/IP CPU will limit the throughput of the application.

• Busy systems with heavy TCP/IP traffic should enable local I/O post-processing for TCP/IP.
  – I/O post processing will be performed on the CPU issued the I/O vs. the TCP/IP CPU.
  – Off loads the TCP/IP CPU.

• To enable local I/O post-processing for TCPIP
  – sysconfig –r net ovms_unit_status = 2147483648
  – Add to sysconfig.tab

Watch out for the PPE feature in TCP/IP V5.7
TCP/IP FTP

• Use the following logical names to speed up FTP transfers:
  – TCPIP$FTP_FILE_ALQ
  – TCPIP$FTP_FILE_DEQ
  – TCPIP$FTP_WNDSIZ

• Logical names may be set system wide or only for specific processes.
Oracle RAC

- Oracle RAC startup/shutdown is very slow.
  - Rx7640, OpenVMS V8.3-1H1, Oracle 10gR2 requires 29 minutes to start 10 RAC databases under CRS.
  - CRS startup and shutdown is serialized.
  - Try parallelizing the startup using PIPE
    * Disable automatic startup of all databases
    * PIPE start db1 | start db2 | start db3 ...

- The system in question now starts all 10 databases in 4.5 minutes.
Data Pump

- Data pump is the fastest way to export/import data.

- Data pump creates multiple threads lowering elapsed time required for export/import to complete.

- Data pump on OpenVMS will gradually slow down as the dump file grows.

- Use the parallel=n feature to guarantee single dump file does not grow beyond 1GB.
SDA> prf load
PRF$DEBUG load status = 00000001
SDA> prf start pc/ind=21E004DA
PC Sampling started...
SDA> prf start collect
SDA>
Now run the application:

$ r prime
  ELAPSED:  0 00:00:24.16  CPU: 0:00:24.06  BUFIO: 0  DIRIO: 0  FAULTS: 0
$

•  To look at the collected data:

    SDA> prf show collect
<table>
<thead>
<tr>
<th>Start VA</th>
<th>End VA</th>
<th>Image</th>
<th>Count</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFFF802.11F00000</td>
<td>FFFFF802.11F01FFF</td>
<td>PRIME</td>
<td>305113</td>
<td>99.85%</td>
</tr>
<tr>
<td>FFFFF802.A1000000</td>
<td>FFFFF802.A1015FFF</td>
<td>Kernel Promote VA</td>
<td>1</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.80000000</td>
<td>FFFFFF.F.80000FFF</td>
<td>SYS$PUBLIC_VECTORS</td>
<td>2</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.80000100</td>
<td>FFFFFF.F.800111FF</td>
<td>SYS$BASE_IMAGE</td>
<td>2</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.80011200</td>
<td>FFFFFF.F.800651FF</td>
<td>SYS$PLATFORM_SUPPORT</td>
<td>258</td>
<td>0.08%</td>
</tr>
<tr>
<td>FFFFFF.F.800A0000</td>
<td>FFFFFF.F.801DD6FF</td>
<td>SYSTEM_PRIMITIVES</td>
<td>88</td>
<td>0.03%</td>
</tr>
<tr>
<td>FFFFFF.F.801DD700</td>
<td>FFFFFF.F.80243BFF</td>
<td>SYSTEM_SYNCHRONIZATION_MIN</td>
<td>9</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.80254600</td>
<td>FFFFFF.F.8026EFFF</td>
<td>SYS$EIDRIVER.EXE</td>
<td>5</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.8026F000</td>
<td>FFFFFF.F.802895FF</td>
<td>SYS$LAN.EXE</td>
<td>2</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.80289600</td>
<td>FFFFFF.F.802BA1FF</td>
<td>SYS$LAN_CSMACD.EXE</td>
<td>2</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.80440E00</td>
<td>FFFFFF.F.8052B2FF</td>
<td>IO_ROUTINES</td>
<td>1</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.8053A600</td>
<td>FFFFFF.F.80670DFF</td>
<td>PROCESS_MANAGEMENT</td>
<td>7</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.80670E00</td>
<td>FFFFFF.F.807759FF</td>
<td>SYS$VM</td>
<td>11</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.80779500</td>
<td>FFFFFF.F.807C76FF</td>
<td>LOCKING</td>
<td>1</td>
<td>0.00%</td>
</tr>
<tr>
<td>FFFFFF.F.807C7700</td>
<td>FFFFFF.F.807F9CFF</td>
<td>MESSAGE_ROUTINES</td>
<td>1</td>
<td>0.00%</td>
</tr>
</tbody>
</table>
```plaintext
<table>
<thead>
<tr>
<th>PC</th>
<th>Count</th>
<th>Rate</th>
<th>Symbolization</th>
<th>Module</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFFFF802.11F00170</td>
<td>63410</td>
<td>20.07%</td>
<td>PRIME+10170</td>
<td>PRIME</td>
<td>00010170</td>
</tr>
<tr>
<td>FFFFFF802.11F00190</td>
<td>6138</td>
<td>2.01%</td>
<td>PRIME+10190</td>
<td>PRIME</td>
<td>00010190</td>
</tr>
<tr>
<td>FFFFFF802.11F001A0</td>
<td>6761</td>
<td>2.21%</td>
<td>PRIME+101A0</td>
<td>PRIME</td>
<td>000101A0</td>
</tr>
<tr>
<td>FFFFFF802.11F00200</td>
<td>6296</td>
<td>2.06%</td>
<td>PRIME+10200</td>
<td>PRIME</td>
<td>00010200</td>
</tr>
<tr>
<td>FFFFFF802.11F00220</td>
<td>8102</td>
<td>2.65%</td>
<td>PRIME+10220</td>
<td>PRIME</td>
<td>00010220</td>
</tr>
<tr>
<td>FFFFFF802.11F00290</td>
<td>6804</td>
<td>2.23%</td>
<td>PRIME+10290</td>
<td>PRIME</td>
<td>00010290</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[GENERATE_PRIME+00000170 / GENERATE_PRIME+00000170]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[GENERATE_PRIME+00000190 / GENERATE_PRIME+00000190]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[GENERATE_PRIME+000001A0 / GENERATE_PRIME+000001A0]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[GENERATE_PRIME+00000200 / GENERATE_PRIME+00000200]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[GENERATE_PRIME+00000220 / GENERATE_PRIME+00000220]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
Would never have been possible without the gracious & expert assistance of:

Christian Moser & Norman Lastovica