
# *zVRM*
# *The Velocity Resource Manager*

Velocity Software Inc.
196-D Castro Street
Mountain View CA 94041
650-964-8867

Velocity Software GmbH
Max-Joseph-Str. 5
D-68167 Mannheim
Germany
+49 (0)621 373844

Barton Robinson,
barton@velocitysoftware.com

*If you can't measure it, I'm just not interested….*

The point of zVRM: Dynamically size Linux servers to meet current workload requirements

Velocity Software's mission:

- Enhancing z/VM Platform Acceptance

Agenda:

- Processor cache: CPU Case Study
- Memory options
- CPU options
- Cpuplugd issues
- VMRM issues
- zVRM Overview

# Case Study: Why sizing matters

- Spike in CPU,
- Performance impact
- Business impact
- Linux Guideline review

Multiple LPARs show same problem, CPU "Core" spikes
Assigned time is at LPAR level, from HMC

```
Report: ESALPARS     Logical Partition Summary
-------------------------------------------------------------
          <--Complex--> <--------Logical Partition------->
          Phys Dispatch                  Virt CPU  <%Assigned>
Time      CPUs    Slice Name      Nbr CPUs Type Total  Ovhd
--------  ---- -------- -------- --- ---- ---- -----  ----
14:01:00    62  Dynamic Totals:   00   36 IFL   2301  22.3
                        LNXPLX    15   18 IFL   1176  12.2
                        LXAPLX    0A   18 IFL   1125  10.1
-------------------------------------------------------------
14:08:00    62  Dynamic Totals:   00   36 IFL   3599  15.6
                        LNXPLX    15   18 IFL   1799   4.1
                        LXAPLX    0A   18 IFL   1799  11.5
```

Spike in Thread CPU very visible, heavy impact on applications
What would cause such a spike? Double in 2 minutes?

Report: ESASSUM Subsystem Activity

| | <---Users----> | | | Transactions | | <Processor> | |
|---|---|---|---|---|---|---|---|
| | <-avg number-> | | | Per | Avg. | Utilization | |
| Time | On | Actv | In Q | Minute | Resp | Total | Virt. |
| 14:01:00 | 95 | 70 | 191 | 88.0 | 0.030 | 1741 | 1632 |
| 14:02:00 | 95 | 69 | 191 | 89.0 | 0.028 | 1666 | 1559 |
| 14:03:00 | 95 | 71 | 193 | 89.0 | 0.030 | 1727 | 1622 |
| 14:04:00 | 95 | 69 | 194 | 86.0 | 0.035 | 1886 | 1787 |
| 14:05:00 | 95 | 73 | 189 | 81.0 | 0.029 | 2414 | 2298 |
| 14:06:00 | 95 | 70 | 189 | 83.0 | 0.192 | 3590 | 3491 |
| 14:07:00 | 95 | 70 | 188 | 83.0 | 0.083 | 3591 | 3494 |

We hear about "knee in curve"? Processor cache is close?
What is important? Real work (instructions)
CPU Doubles, CPI > doubles, work goes down
What would cause such a spike?

```
Report: ESAMFC          MainFrame Cache Magnitudes
-------------------------------------------------
                <CPU Busy><-------Processor------>
                <percent> Speed/<-Rate/Sec->
Time       CPU Totl User Hertz Cycles Instr Ratio
System:        1741 1632 5200M  90.7G 50.2G 1.808
System:        1666 1559 5200M  86.8G 50.5G 1.717
System:        1727 1622 5200M  89.9G 52.2G 1.724
System:        1886 1787 5200M  98.2G 55.9G 1.757
System:        2414 2298 5200M   126G 50.2G 2.504
System:        3590 3491 5200M   187G 41.1G 4.541
System:        3591 3494 5200M   187G 42.9G 4.349
```

**Linux, the perfect storm.** Does L1/L2 cache last till next dispatch?

- Dispatches Linux Servers 1000's of times per second,
- Dispatches per second per thread: 10,000
- Dispatches per second by core: 20,000
- Average dispatch: 25 microseconds, 125K cycles, 70k Instr

| CPU | <VMDBK Steals | --> Dispatcher pty | <-From Nesting Long Paths | Same | NL1 | NL2 |
|---|---|---|---|---|---|---|
| 0 | 837.8 | 0 | 12672.5 | 571 | 267 | 0 |
| 1 | 523.6 | 0 | 9663.9 | 330 | 194 | 0 |
| 2 | 663.1 | 0 | 9370.8 | 381 | 282 | 0 |
| 3 | 506.8 | 0 | 7953.3 | 278 | 229 | 0 |
| 4 | 651.3 | 0 | 8687.8 | 379 | 272 | 0 |
| 5 | 517.2 | 0 | 7721.4 | 281 | 236 | 0 |
| 6 | 678.3 | 0 | 11241.3 | 361 | 317 | 0 |
| 7 | 569.0 | 0 | 10316.2 | 276 | 293 | 0 |
| 8 | 984.4 | 0 | 11196.9 | 642 | 343 | 0 |

**Linux, the perfect storm.**

- Dispatches Linux Servers 1000's of times per second,
- Linux balances across VCPU (New data exposed)

```
<Dispatch>
<BY SERVER>
<Rate/Sec>
Disp Waits
---- -----
5426  5426
 35K 35167
5084  5084
4419  4419
 13K 13225
3865  3865
5745  5745
```

```
UserID    <---CPU time-->  <Dispatch>
 CPUvadd      <-Percent->  <Rate    >
          Cnt TOT   Virt   /Second
-------- --- ----- -----  -------
14:01:00   0  1687  1632   139K
ZLA081     8 153.6 152.0   5426
 CPU-00      20.30 19.64   1207
 CPU-01      17.14 17.00    583
 CPU-02      11.72 11.61    539
 CPU-03      21.69 21.54    634
 CPU-04      21.00 20.86    644
 CPU-05      22.73 22.58    679
 CPU-06      21.78 21.64    571
 CPU-07      17.27 17.15    570
```

# CPU Case Study

Cache performance directly related to processor speed

- Better cache utilization –> faster processor
- Faster process -> lower utilization
- Lower utilization -> need less engines

Processor cache tuning – reduce L1 cache miss?

- All components of instruction required in L1 Cache to execute (instruction, data)
- L1 Cache Miss increases the cycles required per instruction

How to pollute the cache? (and require more cycles per instruction)

- Dispatch new work frequently to rebuild L1 cache every time
- Multiple CPUs doing same work (on different cores)

# CPU Case Study

Too many VCPU

- Shared memory in multiple caches
- Work moved from vcpu to vcpu -> core to core
- Overhead in Linux, in z/VM
- Result:  Reduces cache effectiveness
- And the Diagnose 44 and 9C impact

Observations:

- Dispatch time 25 micro seconds
- Dispatch time slice default is 10 milliseconds

Tuning and configuration guidelines for Linux

- SET SRM DSPSLICE 1MS  (minimum, to protect impact)
- Minimize vcpu so one dispatch does more work

Oversized servers pollute the cache

Linux “round robins” through

```
Report: ESAUSRC          User Configuration
<---CPU---> <Storage>
 <Count>      <VM Size>
 Def  On  Mode  Dflt   Max
 ---  --  ----  ----  ----
   4   4   ESA   32G   32G
   4   4   ESA   32G   32G
   8   8   ESA   96G   96G
   8   8   ESA   96G   96G
   4   4   ESA   70G   70G
   4   4   ESA   70G   70G
   4   4   ESA   28G   28G
   4   4   ESA   28G   28G
   4   4   ESA   20G   20G
```

# The "Large Server" Problem

## Servers moving from x86 oversized

- Typically more (inexpensive) storage on x86
- More (less efficient) processors on x86
- Education and trust in z

## Why large virtual machines?

- Intel servers require and outage to add resources
- Intel hardware is much less expensive
- The standard is to oversize everything

## Cookie cutter virtual machines make life easier

- Cloning is easier
- Requires little planning
- Easy to provide "small, medium and large"

# The Result

## Expensive Real storage is overcommitted

- Workloads variable
- Idle servers consume storage
- Storage requirement larger

## Multiple processors result is spinlocks

- The more vCPUs, the more spinlocks (DIAG 44, 9C)
- Spinlocks cause both system overhead and delays
- Overhead and higher CPI results in more IFLs

# The Problem

## Outages and CritSits

- Managing very large virtual machines with overcommit
- Overcommit of 100GB servers requires technology
- Expanded storage buffer of 20% no longer supported
- IBR, replacement for ExStore limited to 5%, default 3%
- When Linux touches a page, z/VM must back it forever

## Tuning Guideline – Set IBR to max 5% and prepage

- Ensure storage available for new work

## Cookie cutter servers

- 85gb
- 16 VCPU
- 100+ servers

## What should server size be?

- What should CPU busy be?
- What should free storage be?

# User Configuration: ESAUSRC (B)

Report: ESAUSRC User Configuration

| UserID | ClassID | Account Code | > CPU Type | <-------SHARE-------> <Normal> Rel | <--MAX-> Abs | Typ | Shre | Lim -it | <---CPU-us> <Count> Def | On | Qck MDsp | <-MDC> NO FS | NO INS | <Sto <VM Dflt |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LNXS1J2 | Other | LNXS1J2 | IFL | 1600 | . | . | . | . | 16 | 16 | Y | N | N | 85G |
| LNXS1J4 | Other | LNXS1J4 | IFL | 1600 | . | . | . | . | 16 | 16 | Y | N | N | 85G |
| LNXS2J2 | Other | LNXS2J2 | IFL | 1600 | . | . | . | . | 16 | 16 | Y | N | N | 85G |
| LNXS3J2 | Other | LNXS3J2 | IFL | 1600 | . | . | . | . | 16 | 16 | Y | N | N | 85G |

Cookie cutter servrs

- 16 vCPU each (relative 100 / vCPU)
- 85GB each

# User CPU: ESAUSP2

```
Report: ESAUSP2          User Resource Rate Re
Monitor initialized: 06/12/23 at 00:00:00
-----------------------------------------
          <---CPU time---> <----Main Storag
UserID    <(Percent)> T:V -<Resident> Lock
/Class    Total  Virt  Rat Totl Activ  -ed
-------- ----- ----- ---- ---- ----- ----
LNXS3J2   222.4 221.2 1.01  20M 20.0M 1519
LNXS1J2   224.0 222.7 1.01  16M 15.8M 1506
LNXS3J2   226.5 224.6 1.01  19M 18.7M 1660
LNXS3J2   227.4 225.6 1.01  20M 20.0M 1745
LNXS1J4   227.4 226.1 1.01  17M 16.9M 1638
LNXS1J4   228.3 226.7 1.01  16M 15.7M 1687
LNXS1J4   241.8 240.9 1.00  16M 15.8M 1497
LNXS3J2   244.3 242.2 1.01  20M 20.0M 1620
LNXS1J2   245.2 244.2 1.00  16M 15.8M 1507
LNXS3J2   247.2 246.1 1.00  19M 18.7M 1523
LNXS3J2   248.5 247.4 1.00  20M 20.0M 1623
LNXS1J2   255.9 254.3 1.01  17M 16.8M 1555
LNXS1J4   276.5 274.5 1.01  16M 15.8M 1604
LINXS4J1 280.8 279.7 1.00 7.3M 7253K 1152
```

How much CPU is needed? (15 minute intervals)

- Sort CPU Utilization for whole day (cpu by thread)
- Pick largest samples - How many vCPU really needed?
- How much extra overhead in CPU management?

# Linux CPU Details: ESALNXS (Peak interval)

```
Report: ESALNXS        LINUX VSI System Analysis Report
-------------------------------------------------------

Node/     <---Load Numbers--> CPU <Processor Pct Util>
Time      Users Procs MaxProc NBR Total Syst User Idle
-------- ----- ----- ------- --- ----- ---- ---- ----
10:30:00
LNXS3J2       0   266       0 Tot 246.2 10.5  232 1077
                                  1  16.3  0.9 15.0 75.5
                                  2  16.6  0.8 15.5 77.7
                                  3  16.1  0.7 15.0 76.0
                                  4  16.1  0.7 15.1 79.0
                                  5  16.3  0.7 15.3 76.0
                                  6  15.6  0.7 14.7 75.1
                                  7  16.1  0.7 15.1 73.6
                                  8  15.9  0.6 15.0 72.0
                                  9  13.9  0.6 13.0 74.0
                                 10  14.4  0.6 13.6 73.4
                                 11  15.0  0.6 14.2 70.9
                                 12  15.4  0.7 14.5 69.7
                                 13  15.0  0.7 14.1 69.2
                                 14  15.2  0.7 14.3 67.7
                                 15  16.3  0.7 15.3 65.0
                                 16  14.8  0.6 14.0 63.0
```

Linux balances across all CPUs

- Spin locks are expensive
- More use of locks with more CPUs
- What if vCPU holds lock, but paged out?
- (or worse, engine parked….)

# User CPU: ESAUSP2

Report: ESAUSCP Virtual Machine vCPU Analysis

| UserID | <---CPU time--> | | | | | | | | <---Percent wait | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CPUvadd | | <-Percent-> | | <-SHARE--> | | CPU | <-Samples-> | | | | |
| | Cnt | TOT | Virt | Type | Value | TYPE | Total | In Q | Run | Sim | CPU |
| 10:30:00 | 0 | 1818 | 1807 | . | . | . | 298K | 143K | 12 | 0.1 | 24 |
| LNXS3J2 | 16 | 247.2 | 246.1 | REL | 1600 | IFL | 13440 | 12961 | 18 | 0.0 | 31 |
| CPU-00 | | 15.23 | 15.15 | REL | 100 | IFL | 840 | 836 | 17 | 0.1 | 31 |
| CPU-01 | | 15.95 | 15.88 | REL | 100 | IFL | 840 | 822 | 20 | 0 | 32 |
| CPU-02 | | 15.77 | 15.69 | REL | 100 | IFL | 840 | 821 | 18 | 0 | 31 |
| CPU-03 | | 15.87 | 15.80 | REL | 100 | IFL | 840 | 819 | 19 | 0 | 33 |
| CPU-04 | | 16.21 | 16.13 | REL | 100 | IFL | 840 | 813 | 16 | 0 | 31 |
| CPU-05 | | 15.57 | 15.50 | REL | 100 | IFL | 840 | 811 | 19 | 0 | 32 |
| CPU-06 | | 16.11 | 16.04 | REL | 100 | IFL | 840 | 819 | 17 | 0 | 30 |
| CPU-07 | | 15.70 | 15.63 | REL | 100 | IFL | 840 | 816 | 17 | 0 | 32 |
| CPU-08 | | 13.95 | 13.88 | REL | 100 | IFL | 840 | 814 | 16 | 0.1 | 31 |
| CPU-09 | | 14.47 | 14.41 | REL | 100 | IFL | 840 | 806 | 17 | 0 | 28 |
| CPU-10 | | 15.12 | 15.06 | REL | 100 | IFL | 840 | 809 | 17 | 0 | 31 |
| CPU-11 | | 15.52 | 15.46 | REL | 100 | IFL | 840 | 800 | 18 | 0 | 30 |
| CPU-12 | | 15.08 | 15.02 | REL | 100 | IFL | 840 | 814 | 18 | 0 | 31 |
| CPU-13 | | 15.35 | 15.28 | REL | 100 | IFL | 840 | 795 | 18 | 0.1 | 29 |
| CPU-14 | | 16.40 | 16.33 | REL | 100 | IFL | 840 | 804 | 18 | 0.1 | 32 |
| CPU-15 | | 14.92 | 14.86 | RE | | | | | | | |

Too many vCPU cause **spin locks** – wasted cpu

- Look at largest consumer during the day
- 16 vCPU, all waiting for CPU 30%
- Less than 15% busy
- Less CPUs means more work done per dispatch
- Linux balances CPU
- Cpuplugd not helping – needs zVRM

# Linux Configuration Guideline Summary

## Virtual machine size

- Minimize until some swap (swap out initialization pages)
- **Minimize vCPU counts to avoid overhead**

## Swapping

- swap to virtual disk
- Define 2 virtual disks,
  - One to meet the average requirement
  - Second one for overflow - Insurance
- Use DIAG driver instead of FBA
  - Reduces I/O by factor of 8

## Virtual processors

- **Minimize to meet the workload/application requirement**
- Ensure diag 9c, not 44

## Infrastructure costs

- Minimize – shared resource architecture

# Cpuplugd Operational Details

```
Report: ESAOPER       Operator/System Log
Monitor initialized:            at              on
10:15:00 LNXS3J2  vCPU stopped:       1
10:15:00 LNXS3J2  vCPU stopped:       2
10:15:00 LNXS3J2  vCPU stopped:       3
10:15:00 LNXS3J2  vCPU stopped:       4
10:15:00 LNXS3J2  vCPU stopped:       5
10:17:00 LNXS3J2  vCPU started:       1
10:17:00 LNXS3J2  vCPU started:       2
10:17:00 LNXS3J2  vCPU started:       3
10:17:00 LNXS3J2  vCPU started:       4
10:17:00 LNXS3J2  vCPU started:       5
10:17:00 LNXS3J2  vCPU started:       6
10:17:00 LNXS3J2  vCPU started:       7
10:17:00 LNXS3J2  vCPU started:       8
10:17:00 LNXS3J2  vCPU started:       9
10:17:00 LNXS3J2  vCPU started:      16
10:17:00 LNXS3J2  vCPU started:      17
10:17:00 LNXS3J2  vCPU started:      18
10:17:00 LNXS3J2  vCPU started:      19
10:17:00 LNXS3J2  vCPU started:      20
10:17:00 LNXS3J2  vCPU started:      21
10:29:00 LNXS3J2  vCPU stopped:      20
10:29:00 LNXS3J2  vCPU stopped:      21
10:30:00 LNXS3J2  vCPU started:      20
10:30:00 LNXS3J2  vCPU started:      21
10:36:00 LNXS3J2  vCPU stopped:      19
```

Operational changes are logged

- Evaluated at monitor start
- vCPU start/stops?

Cpuplugd at work

- Is it effective?

# Virtual Machine Diagnose Counts: ESAUSRD

Report: ESAUSRD Virtual Machine Diagnose Analysis

| UserID /ClassID | Total rate | <----------diag counts / second 000 | 004 | 008 | 00C | 010 | 014 | 024 | 044 | 09C | 0A0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 10:30:00 | 5820 | 0.1 | 0 | 1.1 | 0.2 | 0.1 | 0.0 | 0.1 | 3191 | 2441 | . |
| ***User Class Analysis*** | | | | | | | | | | | |
| LNXS1J2 | 499 | . | . | . | . | . | . | . | 201 | 298 | . |
| LNXS1J4 | 984 | . | . | . | . | . | . | . | 319 | 664 | . |
| LNXS2J2 | 487 | . | . | . | . | . | . | . | 149 | 337 | . |
| LNXS3J2 | 887 | . | . | . | . | . | . | . | 253 | 634 | . |
| LINXS1J2 | 94.2 | . | . | . | . | . | . | . | 87.0 | 6.5 | . |
| LINXS2J2 | 313 | . | | | | | | . | 301 | 11.1 | . |
| LINXS3J2 | 42.4 | . | | | | | | . | 39.8 | 2.2 | . |
| LINXS4J1 | 38.3 | . | | | | | | . | 34.3 | 3.5 | . |
| LINXS4J3 | 109 | . | | | | | | . | 93.6 | 14.9 | . |
| LINXS5J2 | 76.3 | . | | | | | | . | 61.8 | 14.0 | . |

Some Spin locks normal
- Changes?
- Diag 44 vs Diag 9C.

If owning process paged out,
- long delay

Impacted by vCPU count

# Virtual Machine Storage : ESAUCD2 (again)

```
Report: ESAUCD2       LINUX UCD Memory Analysis Report
-------------------------------------------------------------------
Node/     <-------------------------Storage Sizes (in MegaBytes)--
Time/     <--Real Storage--> <-----SWAP Storage----> Total <------
Date      Total  Avail Used  Total Avail Used  MIN   Avail CMM
-------- ------ ----- ----- ----- ----- ----- ----- ----- ------
10:30:00
LNXS1J2   85304 41209 44095  2810  2810     0  15.6 44019      0
LNXS1J4   85304 39480 45824  2810  2810     0  15.6 42290      0
LNXS2J2   85304 29881 55423  2810  2810     0  15.6 32691      0
LNXS3J2   85304 31377 53927  2810  2810     0  15.6 34187      0
```

Linux storage analysis ( “85 GByte”)

- Swap Unused
- Available storage 140GB
- More if Linux was slightly constrained
- CMM not being utilized

# "other" resource managers

## VMRM - IBM

- No feedback mechanism -> no insight into application requirements
- No storage metrics available
- Would arbitrarily take storage away from servers
- Servers crashed for lack of storage
- Relative shares set "ridiculous"….
- Many controls added for manual control

## Cpuplugd – opensource

- Each server individually manually configured
- Turning off vCPUs gives remaining vCPUs very high priority

# Some history

## Cooperative Memory Management (CMM1, z/VM 5.2)

- Provided command support for Linux to give up ram
- Builds the "CMM Balloon" and tells CP to re-use the storage
- Still available

## IBM's VMRM Cooperative Memory Management (2007)

- CP XAUTOLOG VMRMSVM
- Used CMM based on external sizing
- Zero ability to look inside Linux for "free storage"
- Attempts to utilize resulted in bad things
- Adjusted SHARES based on business requirements
- "I saw some relative shares of 1 which was a bit of shock"

## Collaborative Memory Management Assist (CMM2)

- Hardware assist, seemed too complicated

# zVRM Overview

## Centrally managed via zPRO

- By LPAR defaults
- By node group
- By node

## zVPS provides feedback and performance metrics

## CMM "balloon" used for storage management

- Small increments every interval
- Swapping causes immediate balloon pop
- Will minimize residency of stale storage

## CPU vary on / off

- Uses the zPRO command interface
- Threshold to ensure minimum vCPU counts
- Target utilization controlled by zVRM

# zVRM Controls

## CMM "balloon" used for storage management

- Small (defined) increments every interval
- Swapping causes immediate balloon pop
- Will minimize residency of stale storage
- Maintains target percent of available storage

## CPU vary on / off

- Uses the zPRO command interface
- Threshold to ensure minimum vCPU counts
- Target utilization controlled by zVRM
  - (higher for batch, lower for realtime)

# zVRM Functions

## Centralized control via zPRO interface

- One screen, all LPARs

## All data sourced on one minute interval

- Standard zVPS interval
- Decisions based on Linux metrics

## Storage:

- Reduces free storage incrementally

## CPU Counts online managed to CPU utilization

- Requires zPRO API

# Linux Server Requirements

## Storage / CMM

- 'modprobe cmm'

## CPU Command interface

- zPRO command interface as part of zPRO

# Control Parameters

## Control by server, by user class

- Parameter settings
- SMSG interface with same format

## ZVRM PARMS

- Provides default settings

## zPRO -> z/VM Administration -> zVRM

- Portal to manage zVRM for enterprise

## Note: zVRM runs on ALL managed LPARs

# Control Parameters

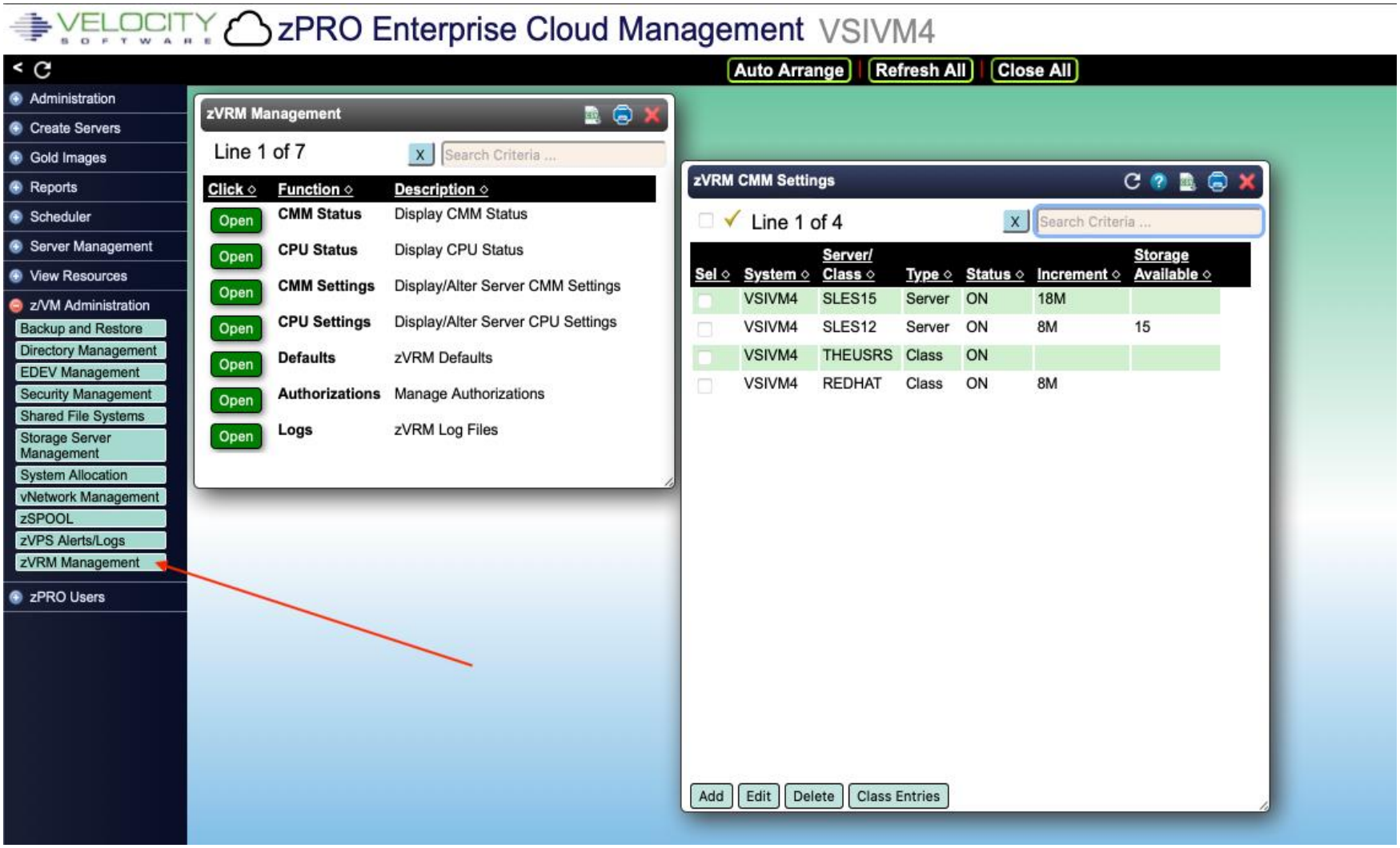
## Storage control:

- CMM DEFAULT ON INCREMENT 32M
- CMM DEFAULT STGAVAIL 20 ;Minimum storage available
- CMM SRVR SLES15 ON INCREMENT 64M
- CMM SRVR SLES12 ON INCREMENT 128M
- CMM CLAS THEUSRS OFF
- CMM CLAS SERVERS ON

## CPU Control

- CPU DEFAULT MINCPU 4
- CPU DEFAULT CPUPCT 25
- CPU SRVR sles15 ON MINCPU 2
- CPU SRVR sles15 ON CPUPCT 30

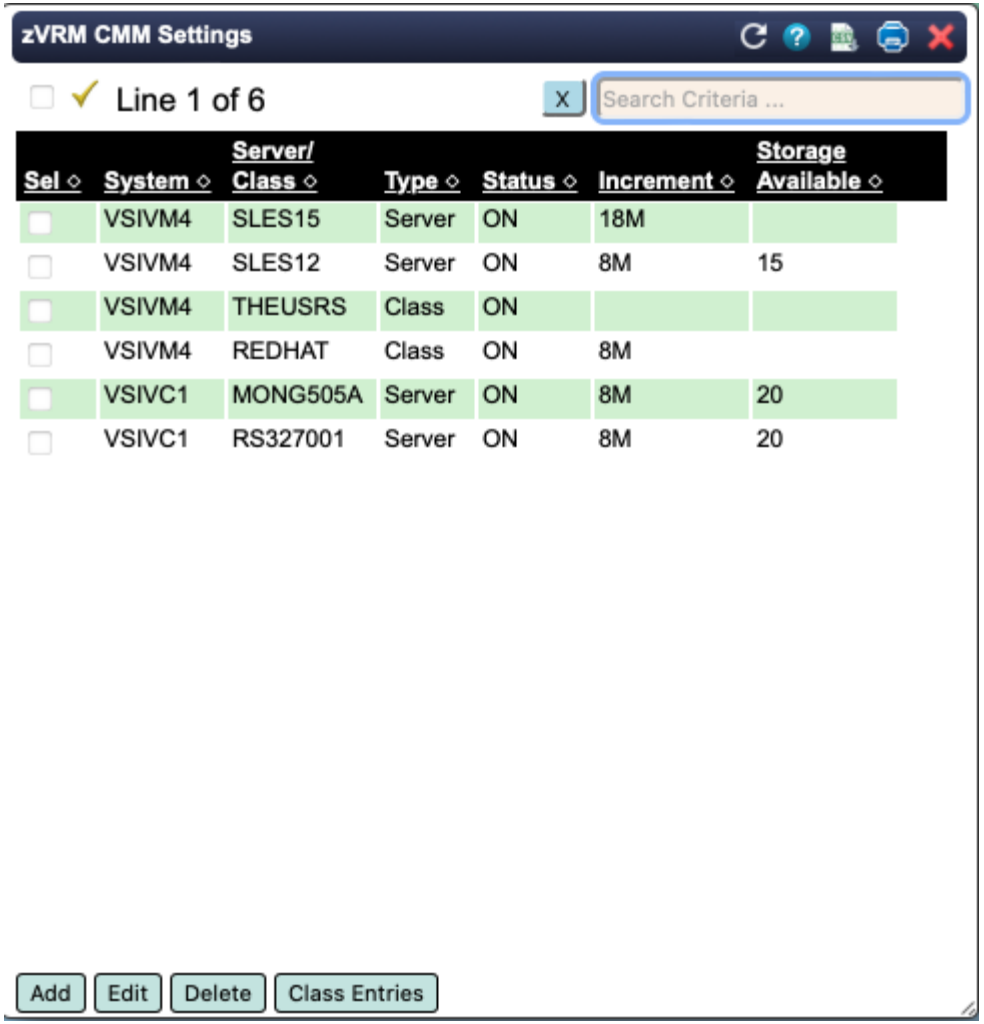
# zPRO Enterprise Portal for zVRM



# zPRO Enterprise Portal for zVRM

## Manage across LPARs

- (VSIVM4,VSIVC1)
- Manage classes
- Manage servers



# Management Requirements

## Don't drive a car without

- A speedometer….
- A gas gauge
- Headlights….

## zVPS provides Linux metrics (by server)

ESAUCD2 - VM4

**ESAUCD2 - LINUX UCD Memory Analysis Report - VM4**

| Time | Node/ Group | <Real Storage (MB)> Total | Avail | Used | <--SWAP Storage (MB)--> Total | Avail | Used | Total MIN | <-----Storage in Use (MB)-----> Avail | CMM | Buffer | Cache | Ovrhd | Shared | Error Message |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 10:51:00 | VPNs | 585.6 | 363.8 | 221.9 | 0 | 0 | 0 | 46.9 | 363.8 | 0.0 | 11.8 | 30.5 | 179.7 | 0 | |
| 10:51:00 | VMWARE | 995.6 | 181.0 | 814.7 | 4096 | 4058 | 37.8 | 15.6 | 4239 | 0.0 | 0.0 | 380.4 | 434.2 | 50.3 | |
| 10:51:00 | UBUNTU | 234.6 | 13.3 | 221.3 | 371.9 | 370.7 | 1.2 | 15.6 | 383.9 | 0 | 40.0 | 63.3 | 118.0 | 3.0 | |
| 10:51:00 | TheUsrs | 42544 | 6914 | 35629 | 8258 | 8242 | 15.5 | 140.6 | 15157 | 40.0 | 2673.9 | 24725 | 8231 | 605.8 | |
| 10:51:00 | SUSE | 23876 | 3203 | 20673 | 11139 | 9796 | 1343 | 93.8 | 12999 | 0.0 | 669.9 | 17436 | 2567 | 745.1 | |
| 10:51:00 | REDHAT | 3281.5 | 419.5 | 2861.9 | 15993 | 15842 | 150.7 | 78.1 | 16262 | 0.0 | 573.2 | 1270 | 1018 | 58.2 | |
| 10:51:00 | ORACLE | 996.8 | 17.7 | 979.1 | 123.9 | 55.0 | 69.0 | 15.6 | 72.7 | 0.0 | 263.4 | 559.2 | 156.5 | 0 | |
| 10:51:00 | OpenShft | 63405 | 6359 | 57046 | 0 | 0 | 0 | 46.9 | 6359 | 0.0 | 0.2 | 32298 | 24747 | 194.9 | |
| 10:51:00 | rhoscp3 | 21135 | 2544 | 18591 | 0 | 0 | 0 | 15.6 | 2544 | 0.0 | 0.1 | 10593 | 7998 | 63.8 | |
| 10:51:00 | rhoscp2 | 21135 | 2652 | 18483 | 0 | 0 | 0 | 15.6 | 2652 | 0.0 | 0.1 | 11492 | 6991 | 69.0 | |
| 10:51:00 | rhoscp1 | 21135 | 1163 | 19972 | 0 | 0 | 0 | 15.6 | 1163 | 0.0 | 0.1 | 10213 | 9758 | 62.1 | |

# Some history

## Velocity Resource Manager (VRM) requirements (2008)

- Application stability is of concern
- Dynamically react to current requirements

## Server size considerations – time to react

- Time to react based on size of server
- Time to react if workload grows fast

## Storage management for small servers difficult

- 4MB buffer (2007) is 16 pages per second for 1 minute
  - Reaction time required to be less than 60 seconds
  - VRM would ask Linux for storage metrics every 10 seconds
- 1GB available space is 4300 pages per second for 1 minute
- For 64GB server, ensure 4gb minimum free?

## Storage controls

- Free storage (percent)
- Increment size

CMM Status

Line 1 of 13

X Search Criteria ...

| Sel | System | Userid | Node | Storage Size(MB) | Storage Free(MB) | Storage CMM(MB) | Increment Size | Target Avail | Swap Full% | CMM Up | CMM Down | Status CMM | Status CPU |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | VSIVM4 | MONGO01 | mongo01 | 3849 | 1926 | 332 | 8 | 1924 | 8 | 1 | 0 | ON | OFF |
| | VSIVM4 | RANCHA1 | rancha1 | 3973 | 1534 | 0 | 79 | 993 | 0 | 1 | 0 | OFF | ON |
| | VSIVM4 | RANCHA2 | rancha2 | 3973 | 1703 | 0 | 79 | 993 | 0 | 1 | 0 | OFF | ON |
| | VSIVM4 | RANCHS1 | ranchs1 | 3973 | 992 | 0 | 79 | 993 | 0 | 1 | 0 | OFF | ON |
| | VSIVM4 | REDHAT6 | redhat6 | 492 | 8 | 0 | 8 | 123 | 0 | 0 | 0 | ON | OFF |
| | VSIVM4 | REDHAT6X | REDHAT6X | 996 | 16 | 0 | 8 | 249 | 22 | 0 | 0 | ON | OFF |
| | VSIVM4 | REDHAT75 | redhat75 | 988 | 429 | 0 | 8 | 247 | 0 | 1 | 0 | OFF | OFF |
| | VSIVM4 | REDHAT85 | REDHAT85 | 814 | 186 | 0 | 8 | 203 | 8 | 1 | 0 | OFF | OFF |
| | VSIVM4 | REDHAT9 | redhat01 | 970 | 144 | 0 | 8 | 242 | 0 | 0 | 0 | ON | OFF |
| | VSIVM4 | R75ORA18 | r75ora18 | 988 | 40 | 0 | 19 | 247 | 100 | 0 | 0 | ON | OFF |
| | VSIVM4 | SLES12 | SLES12 | 3892 | 131 | 0 | 16 | 583 | 100 | 0 | 0 | ON | ON |
| | VSIVM4 | SLES12X5 | sles12x5 | 1825 | 369 | 768 | 8 | 365 | 0 | 0 | 0 | ON | OFF |
| | VSIVM4 | SLES15 | SLES15 | 818 | 19 | 0 | 16 | 163 | 4 | 7 | 4 | ON | ON |

## vCPU controls

- Share controls by vCPU
- Target vCPU utilization

**CPU Status**

Line 1 of 13

X Search Criteria ...

| Sel | System | Userid | Node | CPU Util% | CPU Target | Current CPU | Defined CPU | Minimum CPU | SHARE /VCPU | SHARE VM |
|---|---|---|---|---|---|---|---|---|---|---|
| | VSIVM4 | MONGO01 | mongo01 | 0.7 | 30 | 1 | 1 | 2 | 100 | 100 |
| | VSIVM4 | RANCHA1 | rancha1 | 12.9 | 30 | 2 | 2 | 2 | 55 | 55 |
| | VSIVM4 | RANCHA2 | rancha2 | 13.2 | 30 | 2 | 2 | 2 | 55 | 100 |
| | VSIVM4 | RANCHS1 | ranchs1 | 22.3 | 30 | 2 | 2 | 2 | 55 | 100 |
| | VSIVM4 | REDHAT6 | redhat6 | 1.5 | 30 | 1 | 1 | 2 | 100 | 100 |
| | VSIVM4 | REDHAT6X | REDHAT6X | 0.5 | 30 | 1 | 1 | 2 | 100 | 100 |
| | VSIVM4 | REDHAT75 | redhat75 | 0.1 | 30 | 1 | 2 | 2 | 100 | 100 |
| | VSIVM4 | REDHAT85 | REDHAT85 | 0.2 | 30 | 1 | 1 | 2 | 100 | 100 |
| | VSIVM4 | REDHAT9 | redhat01 | 0.4 | 30 | 1 | 1 | 2 | 100 | 100 |
| | VSIVM4 | R75ORA18 | r75ora18 | 1.1 | 30 | 1 | 1 | 2 | 100 | 100 |
| | VSIVM4 | SLES12 | SLES12 | 25.6 | 20 | 1 | 1 | 4 | 100 | 100 |
| | VSIVM4 | SLES12X5 | sles12x5 | 0.1 | 18 | 1 | 1 | 1 | 100 | 100 |
| | VSIVM4 | SLES15 | SLES15 | 2.8 | 4 | 1 | 1 | 1 | 155 | 120 |

# zVRM Summary

## Cookie cutter servers manageable

- Storage / ram reduced to meet workload requirements
- vCPU counts managed to meet workload requirements
- Share settings, dispatch priorities managed

## Centralized management

- zPRO function
- By node class, node

## Optimize over commitment of resources

- Full feedback mechanisms
- Data driven decisions

# Recap and Thankyou

## zVRM

- Centralized resource management
- Will reduce memory requirements
- Will reduce CPU requirements
- Will make your machine faster
- Allows large "cookie cutter servers"
- Future opportunities

Questions and suggestions can be sent to 'barton@velocitysoftware.com'