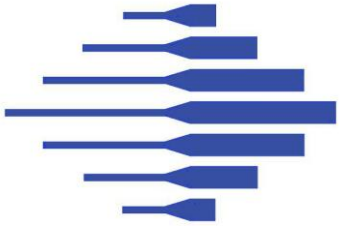
35 YEARS OF PERFORMANCE MANAGEMENT



VELOCITY SOFTWARE

Introduction to RedHat OpenShift Performance

Barton Robinson, CTO
Velocity Software, Inc.
Barton@velocitySoftware.com

Topics

- Why performance measurement is important
- Some history of Linux performance management
- Data collection technology
- Kubernetes, Docker Analysis
- Open Shift Performance
- RHOS overhead analysis
- Collecting RHOS Performance data
- Container View

Why Performance Mgmt?

"If you can't measure it, I'm just not interested"

Black boxes do not have a "productive" future

When, not "IF" an application misses objectives

- What is the solution?

Production requires some level of performance

- Requires planning – with input data
- Requires real time feedback
- Large environments require operational support

Performance Mgmt is a Process

Performance Analysis

- Understanding system, application performance
- Resolving current performance issues (z/VM, Linux, network)
- 0

Operational Alerts

- Supporting 100's/1000's of servers in many locations
- Defining and automating operational support

Capacity Planning

- Providing input to the financial acquisition process

Accounting / Charge back

- Building a financial model for resource billing
- Who is consuming the resource?

Performance management can NOT be the performance problem

Providing performance management requires:

- Accuracy,
- scalability,
- Long term data for trend analysis
- extensible
- Minimize complexity
- Ease of use, support
- Modernization

Data Accuracy History

Correct data

- Linux in virtualized environments was very wrong (bogomips?)
  - Required prorate technology to correct
  - "stealtime" implemented, but often misunderstood
- Linux in SMT environment – many numbers "bogus"
  - Corrected with "Velocity" prorate technology
- Containers use resource, how much?
- Does Managing containers take (a LOT) of resource?

Capture ratios (is the data valid?)

- Do we know where our resources are being utilized?
- Compare data from multiple sources (HMC, z/VM, Linux, etc)
- ("http://VelocitySoftware.com/handouts/capture.html")

Costs of Linux Performance Management

Operational cost of running agents

- "2% per server costs 1 IFL per 50 servers"
- Velocity targets less than .1% (point one percent) of ONE processor with one minute data collection per Linux server
- (One current installation complains about 20 ifls for agent….)

Data Accuracy not easy

- Virtualized CPU (SMT) accounting must be normalized

Capture ratios

- Data must be complete,
- Capture ratio normally at 100% to the process level

Data Collection Technologies

"Pull" or "Push"?

System provided (CP Monitor, SMF)

Snmp is a significant source of data

- .1% of one CPU with one minute granularity collecting everything
- "Pull" technology, may pull 1000's of metrics with little overhead
- Secure with SSL (V3)
- Docker has APIs to collect container information
- Velocity Software's snmp mib collects container information

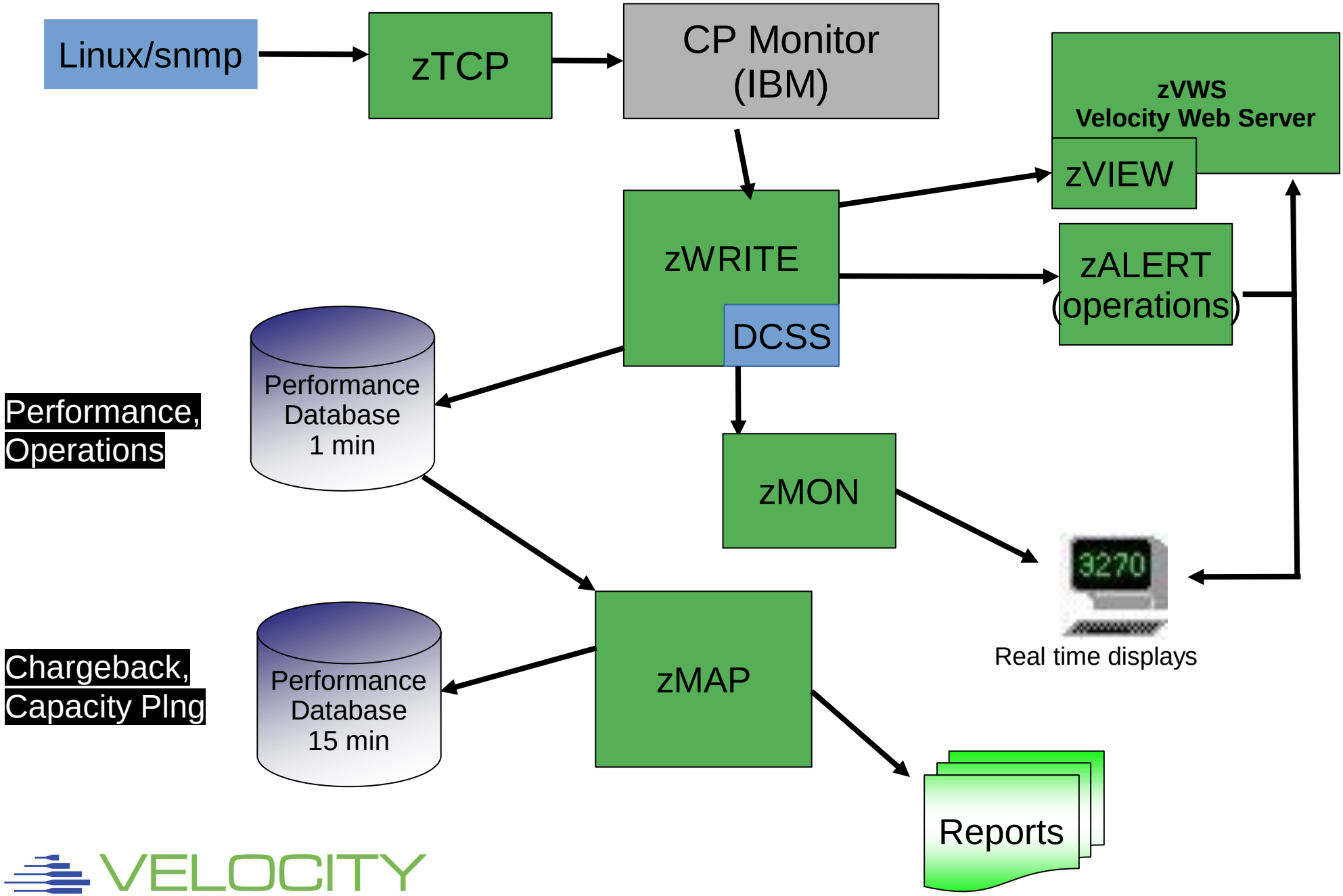
collectd

- Opensource.
- "Push" technology, suitable for secure technology.
- IBM as provided some additional metrics
- Not well used

Prometheus support (vs Prometheus as the reporter)

- Large (megabytes vs "k") data
- Seems expensive to process

zVPS Architecture (35 years)



High Linux CPU capture ratio

```
Report: ESALNXV      LINUX Virtual Processor Analysis Report
-------------------------------------------------------------------------
Node/     VM       <Linux Pct CPU> <Process  Data> Capture  Prorate
 Name     ServerID Total Syst User Total Syst User   Ratio  Factor
-------- -------- ----- ---- ---- ----- ---- ---- ------- ------
10:03:00
NEALE1    LNEALE1  100.0 11.4 88.6 100.2 11.5 88.7   1.002  1.000
```

```
Report: ESALNXP      LINUX HOST Process Statistics Report
-------------------------------------------------------------
node/     <-Process Ident-> Nice <------CPU Percents---->
 Name      ID    PPID   GRP  Valu  Tot  sys user syst usrt
--------- ----- ----- ----- ---- ---- ---- ---- ---- ----
10:03:00
NEALE1        0     0     0    0  100 0.43 3.35 11.0 85.4
 kswapd0    100     1     1    0 0.12 0.12    0    0    0
 snmpd     1013     1  1012  -10 0.13 0.03 0.10    0    0
 sh        3653  3652 30124    0 52.7    0    0 9.37 43.3
 gmake     9751  9750 30124    0 43.4 0.02 0.02 1.37 42.0
 sh       10129  9751 30124    0 0.02 0.02    0    0    0
 sh       10130 10129 30124    0 0.63 0.03 0.23 0.28 0.08
 cc1      10307 10306 30124    0 3.12 0.18 2.93    0    0
 rpmbuild 30124 16382 30124    0 0.07 0.03 0.03    0    0
 sh       30125 30124 30124    0 0.02    0 0.02    0    0
 gmake    30126 30125 30124    0 0.02    0 0.02    0    0
```

```
Report: ESALNXC      LINUX Process Conf
-----------------------------------------
Node/        <-Process Ident-> <-----Pr
 Name         ID    PPID   GRP  Path
------------ ----- ----- ----- --------
NEALE1
init             1     0     0 init [3]
migratio         2     1     0 migratio
ksoftirq         3     1     0 ksoftirq
events/0         4     1     0 events/0
 khelper         5     4     0 khelper
 kblockd/        6     4     0 kblockd/
 cio            41     4     0 cio
 cio_noti       42     4     0 cio_noti
 kslowcrw       43     4     0 kslowcrw
 appldata       96     4     0 appldata
 aio/0         101     4     0 aio/0
 pdflush      5266     4     0 pdflush
 pdflush     26647     4     0 pdflush
kswapd0        100     1     1 kswapd0
kmcheck        158     1     1 kmcheck
syslogd        976     1   976 /sbin/sy
klogd          979     1   979 /sbin/kl
snmpd         1013     1  1012 snmpd
portmap       1030     1  1030 /sbin/po
rpciod        1034     1     1 rpciod
lockd         1035     1     1 lockd
sshd          1072     1  1072 /usr/sbi
 sshd        16272  1072 16272 sshd: bu
 sshd        16288  1072 16288 sshd: bu
  sshd       16290 16288 16288 sshd: bu
   bash      16291 16290 16291 bash
    python   16312 16291 16291 python
     do-bui  16313 16312 16291 /bin/sh
      bb_do  16382 16313 16291 /usr/bin
       rpmb  16415 16382 16415 rpmbuild
       rpmb  30124 16382 30124 rpmbuildc
```

VELOCITY SOFTWARE

What is a container?

Containers are not magic

Each container managed by a “container manager”

- “virtual” Processes inside container are mapped to “real”
- File systems are remapped
- Container “limited to what it can see”’

RHOS Data Collection Technologies

RHOS Servers are "closed"

- Not open to using snmp directly
- No collectd
- Pushes data to prometheus

Snmp implementation:

- Create a container with snmp
- Install container
- Snmp in container has access to data

Prometheus support (vs Prometheus as the reporter)

- Collect very large data segments (megabytes vs "k")
- Parse data in low level language

Linux Process Tree

```
Report: ESALNXC      LINUX Process Configuration Re
Monitor initialized: 03/01/23 at 21:30:01 on 8562 s
-------------------------------------------------
Node/        <-------Process Ident-------> Appl
 Name          ID      PPID   GRP     Appl Name
------------ ------- ------- ----- ------- --------
21:32:00
DOCKER
 systemd           1       0     1   61176 systemd
  dockerd       1218       1  1218    1218 dockerd
   docker-c     1339    1218  1339    1218 dockerd
    docker-c   32289    1339 32289    1218 dockerd
     httpd     32312   32289 32312   32312 bouncer1
      httpd    32370   32312 32312   32312 bouncer1
      httpd    32371   32312 32312   32312 bouncer1
      httpd    32372   32312 32312   32312 bouncer1
    docker-c   32476    1339 32476    1218 dockerd
     httpd     32498   32476 32498   32498 bouncer2 <--
      httpd    32553   32498 32498   32498 bouncer2
      httpd    32554   32498 32498   32498 bouncer2
      httpd    32555   32498 32498   32498 bouncer2
    docker-c   32663    1339 32663    1218 dockerd
     httpd     32685   32663 32685   32685 bouncer3
      httpd    32740   32685 32685   32685 bouncer3
      httpd    32741   32685 32685   32685 bouncer3
      httpd    32742   32685 32685   32685 bouncer3
    docker-c   32853    1339 32683    1218 dockerd
     httpd     32872   32853 32664   32872 bouncer4
      httpd    32923   32872 32664   32872 bouncer4
      httpd    32924   32872 32664   32872 bouncer4
      httpd    32925   32872 32664   32872 bouncer4
```

h

Customer request for Docker management

Linux Process Tree

- Shows all processes
- Docker-c “control”

Result:

- Containers measurable
- Processes measurable

VELOCITY SOFTWARE

Docker Container APIs

Docker Configuration APIs

- **exposed via snmp - VSI MIB**
- Image, container names, status
- Internal index, process ID
- Create / start / finish date/time

```
Report: ESADOCK1      DOCKER Configuration Report
-------------------------------------------------------------------
Time /    <----------Container Configuration------------> Status
Node      Index          ImageName  ContName          Procid
--------  -------------  ---------  ----------------  ------  ------
11:25:00
sles12
          a2e334b2a401   stresscpu  stress4            36448  runn
          6dd81b413dfa   stresscpu  stress3            36360  runn
          e96592194411   stresscpu  stress2            36219  runn
          ab0a179c44c9   stresscpu  stress1            36133  runn
          afc4f3819380   hello-worl hardcore_nash          0  exit
          f57e0f5fd61c   hello-worl pedantic_lamarr        0  exit
```

Docker CPU Consumption APIs (exposed via snmp - VSI MIB)

By container, data validated against process table data

- User cpu
- System CPU

```
Report: ESADOCK2      DOCKER Transaction Report
Monitor initialized: 03/07/23 at 11:23:56 on 8562 seria
-----------------------------------------------------

Node      <---Container---------> Interval <CPU Percent>
          Index         Name       Seconds  User   System
--------  ------------ ---------- -------  -----  ------
11:25:00
sles12    a2e334b2a401 stress4        60.5     8.7       0
          6dd81b413dfa stress3        60.5     8.1     0.0
          e96592194411 stress2        60.5     5.2     0.0
          ab0a179c44c9 stress1        60.5     5.8     0.0
          afc4f3819380 hardcore_n     60.5       0       0
          f57e0f5fd61c pedantic_l     60.5       0       0
```

VELOCITY SOFTWARE

Docker Storage Consumption (exposed via snmp - VSI MIB)

Storage Resource consumption by container

- Storage use, paging, file activity (From DOCKER API)

```
Report: ESADOCK2      DOCKER Trans              Velocity Software Corpor
-------------------------------------------------------------------------
                                        <----------Storage in "MB"---------
Node      <---Container--------->  Current                    <Anonymous>
          Index         Name        use    max    cache   rss  inact  activ
--------  ------------  ----------  ----  -----  ------  ----  -----  -----
11:25:00
sles12    a2e334b2a401  stress4     4.19   4.53    3.70  0.12   0.11   0.01
          6dd81b413dfa  stress3     3.96  11.64    3.70  0.12   0.11   0.01
          e96592194411  stress2     4.23   5.32    3.70  0.13   0.11   0.01
          ab0a179c44c9  stress1     4.02  11.62    3.70  0.12   0.11   0.01
          afc4f3819380  hardcore_n     0      0       0     0      0      0
          f57e0f5fd61c  pedantic_l     0      0       0     0      0      0
```

Docker File Systems

Filesystems have "link" to container, from HOST MIB

```
Report: ESAHST2       LINUX HOST            Velocity Software Corporate
Monitor initialized: 03/07/23 at040F78      First record analyzed: 03/07/2
-------------------------------------------------------------------------
NODE/              <-Utilization-> -Storage------>
Time/              <MegaByte>  Pct
Date       Index   Size  Used Full  Description
-------- -------  ----  ---- ----  ------------------------
sles12
              76   389     0    0  /run/user/0
           25208  64.0     0    0  /var/lib/docker/containers/ab0a179c44c9
           25209  1946   0.0  0.0  /var/lib/docker/containers/ab0a179c44c9
           25211  64.0     0    0  /var/lib/docker/containers/e96592194411
           25212  1946   0.0  0.0  /var/lib/docker/containers/e96592194411
           25214  64.0     0    0  /var/lib/docker/containers/6dd81b413dfa
           25215  1946   0.0  0.0  /var/lib/docker/containers/6dd81b413dfa
           25217  64.0     0    0  /var/lib/docker/containers/a2e334b2a401
           25218  1946   0.0  0.0  /var/lib/docker/containers/a2e334b2a401
```

Docker File Systems (exposed via snmp - VSI MIB)

File systems by container

- File size/utilization (not process table)
- Description, R/W, boot Flags

```
Report: ESADOCK3      DOCKER File System Report
-------------------------------------------------------
NODE/Container   <-Utilization->                    <----
Time/     FileSys <MegaByte>  Pct          R/W  Boot Alloc
Date      Index   Size  Used Full  Errors  R/W  Flag Units
-------- -----    ----  ---- ----  ------  ---  ---- -----
11:25:00
sles12
 stress4
          25217     64     0    0       0   No    No  4096
          25218   1946     0  0.0       0   No    No  4096
 stress3
          25214     64     0    0       0   No    No  4096
          25215   1946     0  0.0       0   No    No  4096
 stress2
          25211     64     0    0       0   No    No  4096
          25212   1946     0  0.0       0   No    No  4096
```

VELOCITY SOFTWARE

Openshift Performance ?

Openshift servers are “closed”

No netsnmp, no collectd, only Prometheus provided

Velocity Software provides “vsi-snmp” container with fully enabled snmp

Performance management for Openshift should provide:

- **Data at the container level**
- Resource consumption at container level
- Number active containers
- Understanding of CPU consumption

Prometheus agents part of openshift HTTP interface

- Seem “expensive”

Consuming Prometheus data is also an option?

- http interface
- “blob” of data

Openshift performance data source

Data sources critical to performance management

Snmp installed in a container

- **Snmp is the most efficient method of collecting performance data!!!**
- Full access to systemwide metrics
- Additional metrics collected to align process data with containers
  - (1.3.6.1.4.1.15601.31.1) (12 bytes per metric)
- Container available on Velocity Software download site
- Snmp is extremely efficient in every comparison

Prometheus agents part of openshift (not efficient)

- HTTP interface
- "blob of data" (100k/pod, multiple megabytes per request) (yes, really "blob")
- https://github.com/google/cadvisor/blob/master/docs/storage/prometheus.md#prometheus-container-metrics
- Limited metrics
- Significant overhead in parsing every single metric of "blob"

Prometheus "blob" interface

Generic implementation, parsing not too difficult, just expensive

Each metric tagged in blob:

container=""
id="/kubepods.slice/kubepods-burstable.slice/kubepods-burstable-pod81ba15c5_32a2
image=""
name=""
namespace="openshift-monitoring"
pod="prometheus-k8s-1"} 65382.32 1677698845481
container_cpu_system_seconds_tota

And for comparison

container=""
id="/kubepods.slice/kubepods-besteffort.slice/kubepods-besteffort-pod953ec259_31
image=""
name=""
namespace="vsi-snmpd-test"
pod="vsi-snmpd-test-nzz4b"} 86.61 1677698849149 container_cpu_system_seconds_tot

OpenShift snmp Implementation

Velocity mib for OpenShift provides:

- ProcessID – matches to Linux process table
- Container name: “vsi-snmpd”
- Pod name: “vsi-snmpd-test-9td44”
- Namespace: “vsi-snmpd-test”
- Pod ID

VSI Container has one active process, snmpd with velocity mib

- (Kubernetes keeps both a pidfile and a config.json file with container info)
- New mib aligns container info to process table
- `"KUBERNETES_PORT_443_TCP_PORT=443",`
- `"MARKETPLACE_OPERATOR_METRICS_SERVICE_PORT_HTTPS_METRICS=8081`
- `"REDHAT_OPERATORS_SERVICE_PORT_GRPC=50051",`

Vsi-snmpd Container is installed on ALL nodes

Velocity Software implemented openshift with 3 nodes

VELOCITY SOFTWARE

OpenShift Process Table

By CPU Consumption (ESALNXP NODE RHOS*), sorted by cpu

```
Report: ESALNXC       LINUX Process
Monitor initialized: 03/08/23 at 0
--------------------------------
Node/          <-------Process Ident
 Name            ID       PPID   GRP
------------ ------- ------- -----
rhoscp1
 systemd           1       0     1
  systemd-       919       1   919
  systemd-       963       1   963
  auditd        1060       1  1060
  dbus-dae      1097       1  1097
  crio          1929       1  1929
  kubelet       1970       1  1970
  conmon        2387       1  2387
   kube-sch     2442    2387  2442
  conmon        2520       1  2520
   cluster-     2559    2520  2559
  conmon        2582       1  2582
   cluster-     2600    2582  2600
  conmon        2677       1  2677
   cluster-     2703    2677  2703
  conmon        8396       1  8396
   promethe     8455    8396  8455
  conmon       10850       1 10850
   grafana-    11073   10850 11073
  conmon     1509339       1  2011
   snmpd     1509401 1509339  2073
```

CPU by Process

By CPU Consumption (ESALNXP)

Report: ESALNXP LINUX HOST Process Statistics Report

node/ Name	<Process Ident> ID	PPID	Nice Valu	PRTY Valu	<------CPU Percents----> Tot	sys	user	syst	usrt
07:02:00									
rhoscp1	0	0	0	0	139	19.9	113	3.74	2.87
systemd	1	0	0	20	1.62	0.57	1.05	0	0
crio	1929	1	0	20	6.48	0.15	0.58	3.24	2.51
kubelet	1970	1	0	20	10.5	3.06	7.47	0	0
etcd	3078	3061	0	20	15.4	5.91	9.5	0	0
etcd	3131	3113	0	20	1.67	0.87	0.80	0	0
cluster-	3959	3944	0	20	2.49	0.23	2.25	0	0
cluster-	5072	4989	0	20	2.02	0.35	1.67	0	0
promethe	8455	8396	0	20	64.1	3.47	60.6	0	0
openshif	11221	11167	0	20	2.72	0.32	2.41	0	0
kube-api	1245253	1245136	0	20	16.7	1.77	15.0	0	0
oauth-ap	1830274	1830210	0	20	3.29	0.25	3.04	0	0
rhoscp2	0	0	0	0	84.2	18.5	59.4	3.88	2.52
systemd	1	0	0	20	2.15	0.67	1.18	0.22	0.08
crio	2056	1	0	20	5.85	0.18	0.58	3.10	1.98
kubelet	2091	1	0	20	12.5	3.72	8.81	0	0
etcd	3185	3169	0	20	14.3	5.33	9.00	0	0
etcd	3230	3210	0	20	1.35	0.68	0.67	0	0
cluster-	3277	3258	0	20	2.50	0.23	2.27	0	0
cluster-	9096	9068	0	20	1.47	0.22	1.25	0	0
cluster-	10370	10281	0	20	1.02	0.20	0.82	0	0
openshif	10951	10905	0	20	2.22	0.22	2.00	0	0
cluster-	11973	11886	0	20	1.50	0.23	1.27	0	0

OpenShift Capture Ratio

Is there complete container capture? 97% first interval is good

```
Report: ESALNXV       LINUX Virtual Processor Analysis Report
Monitor initialized: 03/08/23 at 07:00:01 on 8562 serial 040F78
-------------------------------------------------------------------------
Node/      VM          Node      <Linux Pct CPU> <Process  Data> Capture
 Name      ServerID    GroupID   Total Syst User Total Syst User   Ratio
--------- --------    -------- ----- ---- ---- ----- ---- ---- -------
07:02:00
rhoscp1   RHOSCP1     OpenShif 143.3 23.6  120 139.4 23.6  116   0.973
rhoscp2   RHOSCP2     OpenShif  87.9 22.7 65.2  84.2 22.4 61.9   0.959
rhoscp3   RHOSCP3     OpenShif 126.0 20.0  106 126.0 20.0  106   1.000
```

CPU realtime by Process

By CPU Consumption (ESALNXP NODE RHOS*), sorted by cpu

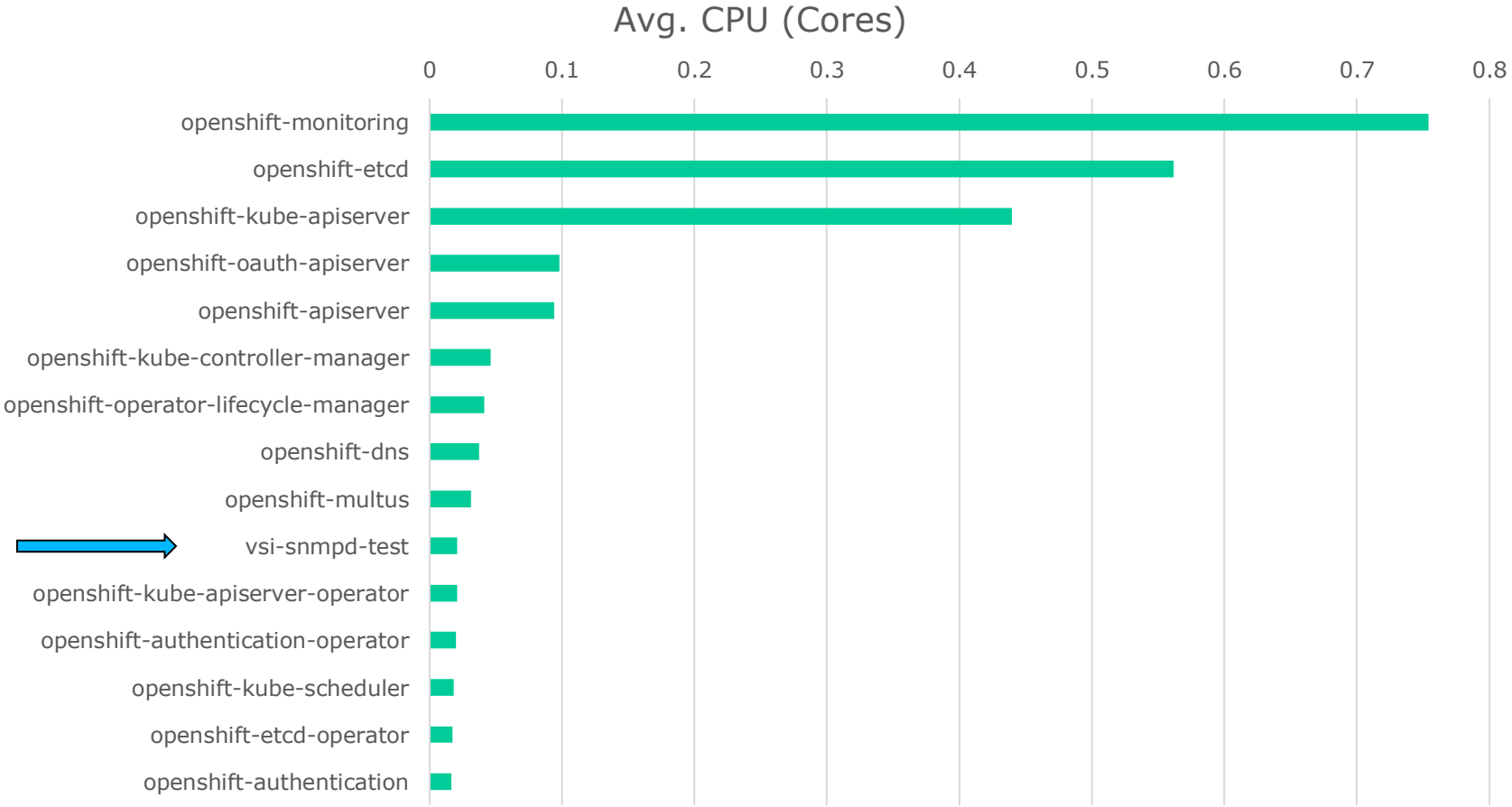
Screen: ESALNXP Velocity Software - VSIVM4 ESAMON 5.134 03/03 :44-22:45
<--1 of 4 VSI Linux Percent Usage by Process NODE RHOSCP*
-Process Ident---> <-----CPU Percents----->

Time	Node	Name	ID	PPID	GRP	**Tot**	sys	user	syst	usrt
22:45:00	rhoscp1	*Totals*	0	0	0	84.5	12.1	67.5	2.9	2.1
	rhoscp3	*Totals*	0	0	0	73.2	9.4	60.7	1.8	1.3
	rhoscp2	*Totals*	0	0	0	53.7	10.4	33.7	2.2	7.5
	rhoscp1	**prometheus**	8455	8396	8455	**38.6**	1.6	37.0	0	0
	rhoscp3	prometheus	7807	7735	7807	**38.4**	1.5	36.9	0	0
	rhoscp1	etcd	3078	3061	3078	10.1	3.9	6.1	0	0
	rhoscp2	kube-apiserv	1464486	1464276	0	8.4	0.9	7.5	0	0
	rhoscp3	etcd	4129	4099	4129	8.2	3.1	5.1	0	0
	rhoscp1	kube-apiserv	1245253	1245136	65488	7.9	0.8	7.1	0	0
	rhoscp2	etcd	3185	3169	3185	7.7	2.9	4.8	0	0
		systemd	1	0	1	7.3	0.4	0.7	0.2	6.0
		kubelet	2091	1	2091	7.3	2.3	5.1	0	0
	rhoscp1	kubelet	1970	1	1970	6.9	2.0	4.9	0	0
	rhoscp3	kube-apiserv	1609641	1609516	36652	6.7	0.7	6.0	0	0
		kubelet	1880	1	1880	5.3	1.6	3.7	0	0
	rhoscp1	crio	1929	1	1929	4.7	0.1	0.3	2.5	1.8
	rhoscp2	crio	2056	1	2056	3.2	0.1	0.3	1.6	1.2
	rhoscp3	crio	1839	1	1839	2.8	0.1	0.2	1.5	1.1
	rhoscp1	oauth-apiser	1830274	1830210	60802	2.1	0.1	2.0	0	0
	rhoscp2	oauth-apiser	694375	694352	0	1.9	0.1	1.8	0	0
	rhoscp3	oauth-apiser	634417	634398	44593	1.9	0.1	1.8	0	0
	rhoscp1	openshift-ap	11221	11167	11221	1.8	0.2	1.6	0	0
		cluster-etcd	3959	3944	3959	1.6	0.2	1.5	0	0
	rhoscp2	openshift-ap	10951	10905	10951	1.6	0.1	1.5	0	0
		cluster-etcd	3277	3258	3277	1.5	0.1	1.3	0	0

OpenShift CPU Consumption

Openshift overhead by component over **three** servers/nodes

- Monitoring seems “excessive”?
- IBM provides 3 IFLs at no charge? But other software?



OpenShift CPU by container

By CPU Consumption by container

```
Report: ESAOPN1       OPENSHIFT Configuration Report                    Veloc
Monitor initialized: 03/07/23 at 11:23:56 on 8562 serial 040F78         First
-------------------------------------------------------------------------------
Linux     <Container Process Info> <------CPU Percents------>  <--Contai
Node                <-Process ID>                             <Alocatio
         Name      Process  Parent  Tot  sys user syst usrtr   Container
-------- -------- -------- ------- ----- ----- ---- ---- ----  ---------
11:25:00
rhoscp1   *Totals*       0       0   135 18.5  110 3.99 2.91
         conmon       2387       1
          kube-sch     2442    2387  0.48 0.07 0.42    0    0 kube-schedu
         conmon       3061       1
          etcd         3078    3061  15.5 6.11 9.37    0    0 etcd
         conmon       3113       1
          etcd         3131    3113  1.93 1.02 0.92    0    0 etcd-metric
         conmon       3944       1
          cluster-     3959    3944  2.55 0.23 2.31    0    0 etcd-health

         conmon       8396       1
          promethe     8455    8396  67.9 3.00 64.9    0    0 prometheus

         conmon      11167       1
          openshif    11221   11167  2.66 0.27 2.40    0    0 openshift-a

         conmon    1830210       1
          oauth-ap 1830274 1830210  3.23 0.20 3.03    0    0 oauth-apise

         conmon    2855315       1
          snmpd    2855331 2855315  0.52 0.17 0.35    0    0 vsi-snmpd
```

VELOCITY SOFTWARE

Openshift Filesystems

Filesystems identified by “pod” index (ESAHST2)

There seems to be too many filesystems as compared to containers…

rhoscp1

11	11K	0	0	Available memory
33	64.0	0	0	/dev
37	11K	0	0	/sys/fs/cgroup
50	64.0	0	0	/dev/shm
51	11K	59.1	0.6	/etc/resolv.conf
52	11K	59.1	0.6	/etc/hostname
53	11K	59.1	0.6	/run/.containerenv
54	191K	111K	58.4	/host
55	191K	111K	58.4	/host/etc
56	191K	111K	58.4	/host/usr
57	191K	111K	58.4	/host/sysroot
60	11K	0	0	/host/sys/fs/cgroup
81	11K	0.1	0.0	/host/dev/shm
85	11K	59.1	0.6	/host/run
86	64.0	0	0	/host/run/containers/tainers/fff97b0e03
87	64.0	0	0	/host/run/containers/tainers/**ea7d7321ef**
94	64.0	0	0	/host/run/containers/tainers/e6bb584ca6
98	64.0	0	0	/host/run/containers/tainers/72dbc2b40e
102	64.0	0	0	/host/run/containers/tainers/99429d8c6d
106	64.0	0	0	/host/run/containers/tainers/96edf02945
110	64.0	0	0	/host/run/containers/tainers/8bcc905ffd
114	64.0	0	0	/host/run/containers/tainers/90fa992328
118	64.0	0	0	/host/run/containers/tainers/e6b24703fa
119	64.0	0	0	/host/run/containers/tainers/69b5cf639a
123	64.0	0	0	/host/run/containers/tainers/9aa9f6cb8d

CPU by Component/Function

Other components?

- By server/node
- Group all containers “conmon”
- Crio, kubelet part of openshift
- Systemd are root functions

```
Report: ESALNXA       LINUX HOST Application Report
Monitor initialized: 03/07/23 at 11:23:56 on 8562 seri
-------------------------------------------------------
Node/     Process/     ID      <---Processor Percent--->
Date      Application                 <Process><Children>
Time      name                 Total sys  user syst usrt
--------  -----------  ------  ----- ---- ---- ---- ----
rhoscp1   *Totals*          0 135.5 18.5  110  4.0  2.9
          conmon      2986625 116.3 14.4  101  0.5  0.4
          crio           1929  6.56  0.1  0.4  3.5  2.5
          kernel            1  0.37  0.4    0    0    0
          kubelet        1970 10.37  2.9  7.5    0  0.0
          ovs-vswi       1299  0.47  0.3  0.2    0    0
          systemd           0  1.35  0.5  0.9    0    0
```

CPU by Component/Function

Other components?

- Totals for all RHOS “OpenShif” servers
- Note this shows CPU time as measured by Linux

```
Report: ESALNXA       LINUX HOST Application Report
Monitor initialized: 03/07/23 at 11:23:56 on 8562 ser
-------------------------------------------------------
Node/     Process/    ID      <---Processor Percent--->
Date      Application                 <Process><Children>
Time      name                Total sys  user syst usrt
--------- ----------- ------ ----- ---- ---- ---- ----
11:25:00
OpenShif  *Totals*         0 351.0 53.4  278 11.8  7.9
          conmon           0 293.1 39.9  251  1.5  1.1
          crio             0 18.68  0.4  1.3 10.3  6.8
          kernel           0  1.37  1.4    0    0    0
          kubelet          0 31.91  9.5 22.4  0.0  0.0
          ovs-vswi         0  1.27  0.7  0.6    0    0
          ovsdb-se         0  0.15  0.0  0.1    0    0
          systemd          0  4.50  1.5  3.0    0    0
```

VELOCITY SOFTWARE

CPU by Component/Function

Some "better news" from z/VM based measurements

- CPU numbers are traditional, measured by Linux
- Virtual Machine with SMT are lower
- IBM SMT numbers do not match,
- customer "correctly" complain that chargeback is broken.

```
Report: ESAUSP5        User SMT CPU Consumption Analys
-------------------------------------------------------
           <------CPU Percent Consumed    (Total)---->
UserID     <Traditional> <MT-Equivalent> <IBM Prorate>
/Class      Total  Virt    Total  Virtual  Total Virtual
-------- ------ -----    -----  -------  ----- -------
07:02:00 414.9   408.0   322.7    317.3   239.7   235.8

  ***User Class Analysis***
OpenShif 355.0   350.3   276.0    272.3   204.9   202.2

  ***Top User Analysis***
RHOSCP1  142.4   140.8   110.1    108.9   82.93   82.01
RHOSCP3  125.2   123.8   97.38    96.34   72.35   71.60
RHOSCP2  86.79   85.04   68.00    66.64   49.31   48.30
```

CPU by Component/Function

Some even "better news"

- CPU numbers are traditional, measured by Linux
- **VSI Prorated** based on HMC and MFC data

```
Report: ESAUSP5       User SMT CPU Consumption Analysis
Monitor initialized: 03/08/23 at 07:00:01 on 8562 serial 040F78
-------------------------------------------------------------------
          <------CPU Percent Consumed    (Total)---->  <-TOTAL CPU-->
UserID    <Traditional> <MT-Equivalent> <IBM Prorate> <VSI Prorated>
/Class    Total  Virt    Total  Virtual  Total Virtual Total  Virtual
-------- ----- -----    -----  -------  ----- ------- -----  --------
07:02:00 414.9  408.0   322.7    317.3  239.7   235.8  208.2    204.7
 ***User Class Analysis***
OpenShif 355.0  350.3   276.0    272.3  204.9   202.2  178.1    175.7
 ***Top User Analysis***
RHOSCP1  142.4  140.8   110.1    108.9  82.93   82.01  71.43    70.65
RHOSCP3  125.2  123.8   97.38    96.34  72.35   71.60  62.80    62.14
RHOSCP2  86.79  85.04   68.00    66.64  49.31   48.30  43.55    42.67
```

OpenShift Components

Prometheus – openshift monitoring

- Monitoring
- time series database
- Alert manager

etcd

Kube-apiserv

marketplace

crio

Controller manager server (conman)

API Server (think smapi)

- Openshift-apiserver
- Oauth-apiser

Cluster-kube-sc - scheduler

HAProxy – high availability load balancing

Kubelet

Dns – obvious…

CMO: Cluster Monitoring Operator

Node-exporter agent – one per node

Thanos Querier – aggregates and deduplicates metrics

Grafana – dashboards for analyzing and visualizing

Summary

OpenShift has the data for:

- Performance Analysis
- Operational Alerts
- Capacity Planning
- Chargeback

Data collection

- Inexpensive
- Validated
- Measurable by container

Thank you. Happy 35th Birthday Velocity Software

VELOCITY SOFTWARE

Mark the dates: Columbus, Ohio

- VM Workshop June 22-24
- Velocity Software performance workshop (June 20-21)