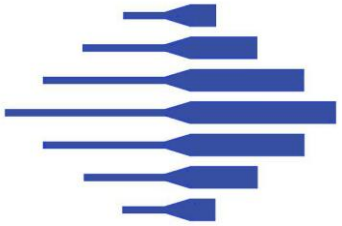
35 YEARS OF PERFORMANCE MANAGEMENT



VELOCITY SOFTWARE

Introduction to Container Technology Performance

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

Topics

Introduction to the requirements

Data collection technology

Container technology:

- Docker,
- Openshift / kubernetes
- Rancher / Kubernetes

Open Shift Performance

RHOS overhead analysis

Collecting container Performance data

Container View

z/OS zcx view

Why Performance Mgmt?

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

Black boxes do not have a "productive" future

- Vendors and customers would do better if understand this

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

Preview: zCX on z/OS?

YES, IT WORKS…. But SMF?

Install zCX with docker

- "https://velocitysoftware.com/zcximpl.html"

Install (snmp) instrumentation in container

And Yes, it works, and what is zCX?

- (Linux 5.4.0, Ubuntu Distribution for 390)

```
SNMP Server Configuration:
   Description:
      Linux 7f1752ffc4e3 5.4.0-146-generic
      #163-Ubuntu SMP Fri Mar 17 18:32:31 UTC 2023 s390x
  ObjectId: 01.03.06.01.04.01.BF.08.03.02.0A
```

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/containers 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

Pretty Dash boards are easy if the data is there

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 – CPU numbers "bogus"
  - Corrected with 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")

Data Collection Technologies

"Pull" or "Push"?

IBM Systems provide limited data (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/TLS (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 has provided some additional metrics
- Not well used

Prometheus support (vs Prometheus as the reporter)

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

Review standard snmp linux data

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 (is the data correct?)

Snmp implementation

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

Prometheus support (vs Prometheus as the reporter)

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

h

Linux Process Tree for Docker

```
Report: ESALNXC       LINUX Process Configuration Re
-------------------------------------------------
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
```

Linux Process Tree

- Shows all processes
- Docker-c "control"

Result:

- CPU by process
- CPU by container

Docker Configuration APIs

- **exposed via snmp**
- Image, container names, status
- Internal index, process ID, from “pid file”
- Create / start / finish date/time

```
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 Container APIs

If have container name, process ID, have CPU.

Need container index and name

By container, data validated against process table data

- User cpu, System CPU

```
Report: ESADOCK2      DOCKER Transaction Report
-------------------------------------------------------

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
```

Docker Storage Consumption (from process data)

Storage Resource consumption by container

- Storage use, paging, file activity

```
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 standard HOST MIB

```
Report: ESAHST2       LINUX HOST File Systems
-------------------------------------------------------------------------
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
```

Linux File Systems matched to containers

File systems by container

- File size/utilization
- Description, R/W, boot Flags

```
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

Our lab (currently)

Capture ratios for Linux processes is 100%

- Docker (sles12)
- Rancher by Suse
- RHOSCP by Redhat

```
Report: ESALNXV       LINUX Virtual Processor Analysis Report
-------------------------------------------------------------------------
Node/      VM          Node      <Linux Pct CPU> <Process  Data> Capture
 Name      ServerID    GroupID   Total Syst User Total Syst User   Ratio
--------- --------    -------- ----- ---- ---- ----- ---- ---- -------
00:15:00
rancha1    RANCHA1     TheUsrs    11.6  5.0  6.5  11.8  5.0  6.8   0.997
rancha2    RANCHA2     TheUsrs    11.6  4.9  6.6  11.8  4.9  6.9   0.999
ranchs1    RANCHS1     TheUsrs    19.4  6.0 13.4  19.4  6.0 13.4   1.000

rhoscp1    RHOSCP1     OpenShft   57.5 10.1 47.4  57.5 10.1 47.4   0.999
rhoscp2    RHOSCP2     OpenShft   46.8 10.1 36.7  46.8 10.1 36.7   1.000
rhoscp3    RHOSCP3     OpenShft   51.1  9.2 41.9  51.1  9.2 41.9   1.001

SLES12     SLES12      SUSE       24.1  0.8 23.3  24.1  0.8 23.3   1.002
```

VELOCITY SOFTWARE

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
- Snmp is extremely efficient in every comparison
- Snmp container installed on every node

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 Process Table

Standard Linux data provided for RHOS. "conmon" container manager

```
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
```

VELOCITY SOFTWARE

CPU by Process

By CPU Consumption by process. Just the active processes

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

CPU realtime by Process

By CPU Consumption For " NODE RHOS*", sorted by cpu

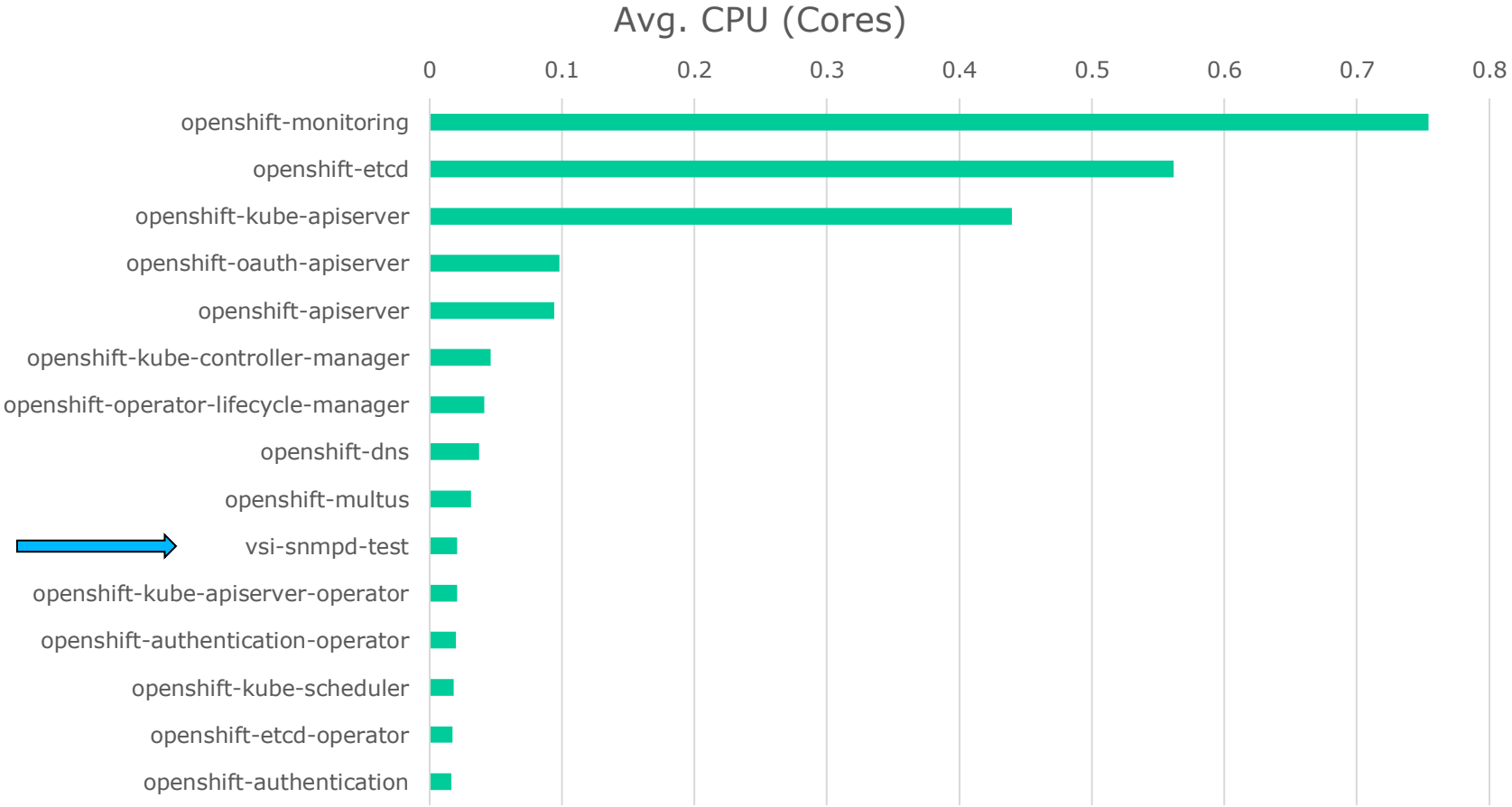
Screen: ESALNXP Velocity Software - VSIVM4 ESAMON 5.134 03/03 :44-22:45
<--1 of 4 VSI Linux Percent Usage by Process NODE RHOSCP*
-Process Ident---> <-----CPU Percents----->

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?



Configuration: Rancher

Rancher not “closed”, run snmp native or container

Container Configuration includes

- Container index, name,
- Pod index, name

```
Report: ESAK8S1       Kubernetes Configuration Report
-------------------------------------------------------------------------
Linux    <---OpenShift Pod Configuration-->  <--------Container Configuration
Node/                                                    <--Process Ide
Time     PodName                 PodIndex      Name                  ProcessID Pro
-------- ----------------------  ------------  --------------------  --------- ---
06/22/23
00:15:00
rancha1
         rke2-canal-5n722        b815f6bd4ba1  calico-node                3725
                                               kube-flannel               3763 fla
         cert-manager-webhook    c317030510d4  cert-manager               2432 web
         rancher-7c5dbf46fc-z    02ad1e9d7318  rancher                    3497
         rke2-coredns-rke2-co    22a8cbf9d51f  coredns                    3349 cor
         rke2-ingress-nginx-c    31a310f130ed  rke2-ingress-nginx-c       3555
         rancher-webhook-577b    5a81e5c27074  rancher-webhook            3105 web
         kube-proxy-rancher-a    545065ae69ff  kube-proxy                 1697 kub
```

Configuration: RHOS

Container Configuration RHOS much larger (snmp in container)…

```
Report: ESAK8S1       Kubernetes Configuration Report                  Velocity Sof
Monitor initialized: 06/22/23 at 00:00:00 on 8562 serial 040F78     First record
-------------------------------------------------------------------------------
Linux     <---OpenShift Pod Configuartion-->  <----------Container Configuration
Node/                                                          <--Process Ide
Time      PodName                  PodIndex      Name                 ProcessID Pro
--------  -----------------------  ------------  -------------------- --------- ---
00:15:00
rhoscp1
          insights-operator-7f     ba92ef4e1b29  insights-operator         13075 ins
          multus-admission-con     bbb779ebae39  kube-rbac-proxy          14520 kub
                                                 multus-admission-con     12865 web
          etcd-rhoscp1.vsi1.ve     c6088570034c  etcd                      2276 etc
                                                 etcd-metrics              2389 etc
                                                 etcd-readyz               2941 clu
                                                 etcd-health-monitor       3594 clu
          prometheus-operator-     c7875e16a183  prometheus-operator-     11955 pro
          kube-state-metrics-5     d5e9800a6a6c  kube-state-metrics       11658 kub
                                                 kube-rbac-proxy-main     12519 kub
                                                 kube-rbac-proxy-self     13456 kub
          prometheus-k8s-1         45f9f5becfa0  prometheus               14548 pro
                                                 thanos-sidecar           15191 tha
                                                 prometheus-proxy         15372 oau
                                                 kube-rbac-proxy-than     15931 kub
                                                 kube-rbac-proxy          15508 kub
                                                 config-reloader          14820 pro
          packageserver-5f99c6     662518f1bc49  packageserver            13044 pac
          vsi-snmpd-vk5vd          7e583397ff6e  vsi-snmpd                19285 snm
          multus-9fj4w             8e5c35c1b612  kube-multus               4922 /en
```

VELOCITY SOFTWARE

CPU by container (rancher)

By CPU Consumption by container

```
Report: ESAK8S2       Kubernetes Resource Utilization Report          V
Monitor initialized: 06/22/23 at 00:00:00 on 8562 serial 040F78       F
-------------------------------------------------------------------------

NODE/                        <---Container-->  <--Container CPU------->
Time/ PodName                <--Process ID-->  <------CPU Percents---->
Date   ContainerName          ProcID ProcName   Tot  sys user syst usrt
----------------------       ------ --------  ---- ---- ---- ---- ----
00:15:00
rancha1
   rke2-canal-5n722
    calico-node               3725 runsvdir  3.70 0.21 0.12 0.73 2.64
   rancher-7c5dbf46fc-z
    rancher                   3497 tini      0.60 0.15 0.36 0.07 0.03
rancha2
   rke2-canal-xl77b
    calico-node               3333 runsvdir  3.68 0.21 0.13 0.72 2.63
ranchs1
   etcd-rancher-server
    etcd                      1965 etcd      1.36 0.53 0.82    0    0
   kube-apiserver-ranch
    kube-apiserver            2010 kube-api  4.27 0.58 3.69    0    0
   rke2-canal-mdccx
   calico-node                2581 runsvdir  3.48 0.20 0.14 0.69 2.46
```

CPU by container (rhos)

By CPU Consumption by container for RHOS

```
Report: ESAK8S2       Kubernetes Resource Utilization Report
-------------------------------------------------------------------------
NODE/                          <---Container-->  <--Container CPU------->
Time/ PodName                  <--Process ID-->  <------CPU Percents---->
Date   ContainerName            ProcID ProcName   Tot  sys user syst usrt
-----------------------  ------ --------  ---- ---- ---- ---- ----
00:15:00
rhoscp1
   etcd-rhoscp1.vsi1.ve
    etcd                         2276 etcd      7.94 2.81 5.13    0    0
    etcd-metrics                 2389 etcd      1.31 0.69 0.62    0    0
    etcd-readyz                  2941 cluster-  0.79 0.07 0.72    0    0
    etcd-health-monitor          3594 cluster-  1.33 0.11 1.22    0    0
   kube-controller-mana
    kube-controller-mana         8999 cluster-  0.61 0.11 0.49    0    0
   apiserver-d84c8f947-
    oauth-apiserver             24092 oauth-ap  1.81 0.08 1.72    0    0
   apiserver-5d795f8cd7
    openshift-apiserver         12688 openshif  1.79 0.13 1.65    0    0
   prometheus-k8s-1
    prometheus                  14548 promethe  13.4 0.45 12.9    0    0
   authentication-opera
    authentication-opera        13103 authenti  1.01 0.16 0.85    0    0
   packageserver-5f99c6
    packageserver               13044 package-  0.95 0.12 0.83    0    0
   kube-apiserver-rhosc
    kube-apiserver             574916 watch-te  10.3 0.79 9.48    0    0
   kube-controller-mana
    kube-controller-mana        22941 kube-con  0.78 0.21 0.57    0    0
```

VELOCITY SOFTWARE

Openshift Filesystems

Filesystems identified by “pod” index (ESAHST2)

Can relate file system to pod index

Report: ESAK8S3 Kubernetes File System Report

NODE/<-------POD---------> Time/ Date	PODNAME	PODIindex	<-------Filesystem----------> FileSys Index	<MegaByte> Size	Used	Pct Full	Errors
00:15:00							
rancha1							
	rke2-canal-5	b815f6bd4ba1	102	64	0	0	0
	cert-manager	c317030510d4	92	64	0	0	0
	rancher-7c5d	02ad1e9d7318	123	64	0	0	0
	rke2-coredns	22a8cbf9d51f	117	64	0	0	0
	rke2-ingress	31a310f130ed	128	64	0	0	0
	rancher-webh	5a81e5c27074	111	64	0	0	0
	kube-proxy-r	545065ae69ff	82	64	0	0	0
rhoscp1							
	oauth-opensh	a2b6aef3d0d0	215	64	0	0	0
	node-ca-69g4	a729126cc8f7	112	64	0	0	0
	insights-ope	ba92ef4e1b29	259	64	0	0	0
	multus-admis	bbb779ebae39	262	64	0	0	0
	etcd-rhoscp1	c6088570034c	90	64	0	0	0
	prometheus-o	c7875e16a183	164	64	0	0	0
	machine-conf	dbb8e11763fb	155	64	0	0	0
	network-metr	ddcf7234b038	237	64	0	0	0
	kube-state-m	d5e9800a6a6c	187	64	0	0	0

CPU by Component/Function

Other components?

- Totals for all RHOS "OpenShif" servers
- CPU time as measured by Linux – correct???

```
Report: ESALNXA       LINUX HOST Application Report
-------------------------------------------------------------------
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
```

CPU by Component/Function

Some "better news" from z/VM based measurements

- CPU numbers are traditional, measured by Linux (Thread time)
- Virtual Machine with SMT "Prorate" are lower
- IBM SMT numbers do not match reality…. (Same in z/OS SMF)
- 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 data

```
Report: ESAUSP5        User SMT CPU Consumption Analysis
-------------------------------------------------------------------------
          <------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
```

Next Step: z/OS zCX

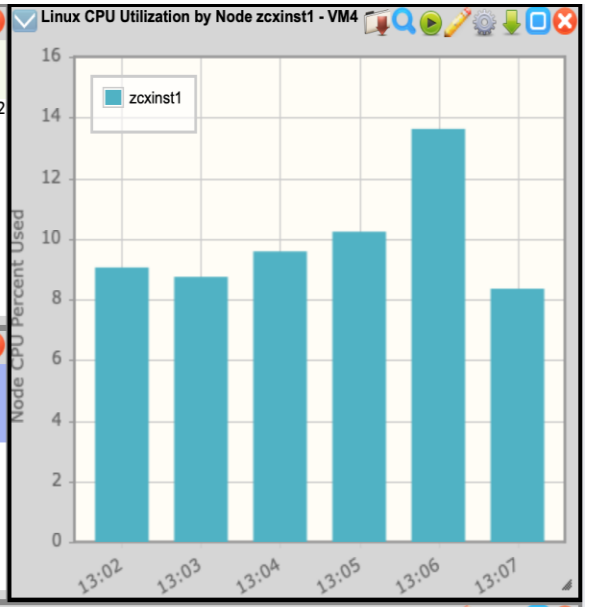
zCX, Openshift run on zip in z/OS address space
Looks like any other Linux server to us

ESALNXP - VSI Linux Percent Usage by Process - VM4

Time	Node	Name	Process Ident ID	PPID	GRP	CPU Percents Tot	sys	user	syst	usrt	nice prty valu	valu	Storage Metrics (MB) Size	RSS	Peak	Swap	Data	Stk	EXEC	Lib	Lck	PTbl	Faults min	maj
13:08:00	zcxinst1	*Totals*	0	0	0	10.7	6.2	2.7	1.5	0.3	0	0	24K	714	25K	306	2834	12.7	169	1981	336	11.6	17	0
13:08:00	zcxinst1	systemd	1	0	1	0.1	0.0	0.0	0	0	0	20	164	7.4	226	0.7	20.5	0.1	1.3	13.7	0	0.1	0	0
13:08:00	zcxinst1	ksoftirqd/0	10	2	0	0.1	0.1	0	0	0	0	20	0	0	0	0	0	0	0	0	0	0	0	0
13:08:00	zcxinst1	rcu_sched	11	2	0	0.1	0.1	0	0	0	0	20	0	0	0	0	0	0	0	0	0	0	0	0
13:08:00	zcxinst1	ksoftirqd/1	16	2	0	0.1	0.1	0	0	0	0	20	0	0	0	0	0	0	0	0	0	0	0	0
13:08:00	zcxinst1	dmcrypt_writ	603	2	0	0.0	0.0	0	0	0	0	20	0	0	0	0	0	0	0	0	0	0	0	0
13:08:00	zcxinst1	btrfs-transa	682	2	0	0.0	0.0	0	0	0	0	20	0	0	0	0	0	0	0	0	0	0	0	0
13:08:00	zcxinst1	systemd-jour	903	1	903	0.1	0.1	0.0	0	0	-1	19	89.1	29.7	111	0.1	18.7	0.1	0.1	13.6	0	0.2	2	0
13:08:00	zcxinst1	multipathd	1029	1	1029	0.1	0.1	0.0	0	0	0	-100	336	16.1	400	0	10.0	0.1	0.1	6.4	336	0.1	0	0
13:08:00	zcxinst1	zcxauthplugi	1083	1	1083	0.1	0.1	0.0	0	0	0	20	469	5.8	469	0.2	142	0.1	2.7	1.9	0	0.1	0	0
13:08:00	zcxinst1	containerd	1085	1	1085	1.1	0.7	0.4	0	0	0	20	1453	19.4	1517	3.4	134	0.1	25.6	1.9	0	0.2	0	0

ESALNXR - Linux Memory Analysis - VM4

Time	Node	Memory in Megabytes Total	Free	Cache Size	Activ	Anonymous Swap	Total	Activ	Inact	Kernel (MB) Stack	Slab Size	Buffers Reclm	Total	Dirty	WrtBk	vmalloc (MB) Total	Used	Page Chunk	table	Large Pages Usage (MB) Total	Free	Rsvd	Page Size
13:08:00	zcxinst1	1407	139.7	459.3	237.6	21.7	271.0	128.1	163.2	14.6	257.9	99.0	7.7	0.1	0	126.0	0.1	0	16.9	0	0	0	0
13:07:00	zcxinst1	1407	139.8	458.9	237.0	21.7	271.0	128.1	163.2	14.7	257.8	99.0	7.7	0.1	0	126.0	0.1	0	16.9	0	0	0	0
13:06:00	zcxinst1	1407	140.2	458.5	236.1	21.7	271.0	128.1	163.2	14.7	258.0	99.0	7.6	0.3	0	126.0	0.1	0	16.9	0	0	0	0
13:05:00	zcxinst1	1407	132.2	457.7	235.0	21.7	280.6	137.7	163.2	14.7	258.1	99.0	7.6	0.1	0	126.0	0.1	0	17.2	0	0	0	0
13:04:00	zcxinst1	1407	141.5	457.4	234.6	21.7	270.9	128.1	163.2	14.7	258.1	98.9	7.5	0.2	0	126.0	0.1	0	17.0	0	0	0	0
13:03:00	zcxinst1	1407	142.0	457.2	234.2	21.7	271.0	128.0	163.2	14.7	258.0	98.9	7.5	0.0	0	126.0	0.1	0	16.9	0	0	0	0
13:02:00	zcxinst1	1407	142.0	456.9	233.6	21.7	271.0	128.1	163.2	14.6	258.0	98.9	7.5	0.2	0	126.0	0.1	0	16.9	0	0	0	0



ESALNXV - LINUX Virtual Processor Analysis Repo - VM4

Time	Node	VM ServerID	Node GroupID	Linux Pct Total	CPU Syst	User	Process Total	Data Syst	User	Capture Ratio	Prorate Factor	LPAR NVcpu	Name
13:08:00	zcxinst1	INST1	TheUsrs	10.7	7.7	3.0	10.7	7.7	3.0	1.000	1.000	2	ZOS25A
13:07:00	zcxinst1	INST1	TheUsrs	8.4	6.0	2.3	8.4	6.0	2.3	1.000	1.000	2	ZOS25A
13:06:00	zcxinst1	INST1	TheUsrs	13.6	9.3	4.4	13.6	9.3	4.4	1.000	1.000	2	ZOS25A
13:05:00	zcxinst1	INST1	TheUsrs	10.2	7.5	2.7	10.2	7.5	2.7	1.000	1.000	2	ZOS25A
13:04:00	zcxinst1	INST1	TheUsrs	9.6	6.8	2.8	9.6	6.8	2.8	1.000	1.000	2	ZOS25A
13:03:00	zcxinst1	INST1	TheUsrs	8.7	6.3	2.5	8.7	6.3	2.5	1.000	1.000	2	ZOS25A
13:02:00	zcxinst1	INST1	TheUsrs	9.1	6.5	2.5	9.0	6.5	2.4	0.991	1.000	2	ZOS25A

ESALNXF - VSI Disk File System Performance - VM4

Time	Node	Name	Type	Read I/O /Second	Sctrs	ms	Write I/O /Second	Sctrs	ms	I/O in Progress	Time(ms) <Per I/O> IOQ	I/O	Device Path
13:08:00	zcxinst1	dm-0	lvm	0	0	0	0	3.3	0	23.0	9.34	0	2.40 9.34 dm-0
13:08:00	zcxinst1	dm-1	lvm	0	0	0	0	3.3	0	23.0	40.7	0	2.65 40.7 dm-1
13:08:00	zcxinst1	dm-2	lvm	0.0	0	32.0	0	0.5	0	88.0	5.16	0	3.13 5.00 dm-2
13:08:00	zcxinst1	vda	disk	0	0	0	0	3.4	0.7	23.9	8.40	0	2.98 3.90 ccw-0.0.0001
13:08:00	zcxinst1	vda1	part	0	0	0	0	0.4	0.4	15.4	2.77	0	5.00 0.38 ccw-0.0.0001-part1
13:08:00	zcxinst1	vda2	part	0	0	0	0	3.0	0.3	25.2	9.21	0	2.68 4.41 ccw-0.0.0001-part2
13:08:00	zcxinst1	vdc	disk	0.0	0	32.0	1.00	0.3	0.2	144	4.68	0	5.00 0.50 ccw-0.0.0004
13:08:00	zcxinst1	vdc1	part	0.0	0	32.0	1.00	0.3	0.2	144	4.68	0	5.00 0.50 ccw-0.0.0004-part1
13:07:00	zcxinst1	dm-0	lvm	0	0	0	0	2.9	0	16.6	17.4	0	4.38 17.4 dm-0

VELOCITY SOFTWARE

zCX on z/OS?

YES, IT WORKS…. But SMF?

Install zCX with docker

- "https://velocitysoftware.com/zcximpl.html"

Install (snmp) instrumentation in container

And Yes, it works, and what is it?

- (Linux 5.4.0, Ubuntu Distribution for 390)

```
SNMP Server Configuration:
   Description:
      Linux 7f1752ffc4e3 5.4.0-146-generic
      #163-Ubuntu SMP Fri Mar 17 18:32:31 UTC 2023 s390x
  ObjectId: 01.03.06.01.04.01.BF.08.03.02.0A
```

Standard snmp (network activity)

zCX Network configuration:

- two network interfaces,
- Ethernet, loopback
- Mac address even

```
NODE       Idx          Speed <-Status-> Up   <--------------------Interface--------------->
           Nbr MTU      (Est) Oper Admin Time MACAddress          Description Type
-------- --- ----- ----- ---- ----- ---- ----------------- --------- ---------------
zcxinst1     1 65536    10M   UP     UP     0 00:20:20:20:20:20 lo          Software LoopBack
            10  1492    10G   UP     UP     0 02:42:AC:11:00:05 eth0        ETHERNET-CSMACD
```

zCX Data Capture (DOCKER)

Linux process table – cpu by process (DOCKER)

Run some stresser processes (http load)

```
LINUX Virtual Processor Analysis Report
-------------------------------------------------------------------------
Node/      VM         Node      <Linux Pct CPU> <Process  Data> Capture
 Name      ServerID   GroupID   Total Syst User Total Syst User   Ratio
--------- --------   -------- ----- ---- ---- ----- ---- ---- -------
15:14:00
zcxinst1   INST1      TheUsers  88.4  5.0 83.4  88.4  5.0 83.4   1.000
```

Linux process table – cpu by process, Business as usual

Run some stresser processes (http load)

```
Report: ESALNXP       LINUX HOST Process Statistics Report
-------------------------------------------------------------------
node/      <Process Ident> Nice PRTY <------CPU Percents---->
 Name            ID    PPID Valu Valu  Tot  sys user syst usrt
--------- ------- ------- ---- ---- ---- ---- ---- ---- ----
15:14
zcxinst1          0       0    0    0 88.4 4.28 1.50 0.68 81.9
 systemd-       889       1   -1   19 0.45 0.17 0.28    0    0
 rsyslogd      1097       1    0   20 0.35 0.12 0.23    0    0
 dockerd       2732       1    0   20 3.12 2.73 0.38    0    0
 stresser      5518    5490    0   20 41.2    0    0 0.32 40.9
 stresser      5655    5630    0   20 41.3    0    0 0.30 41.0
 snmpd         5778    5752    0   20 0.17 0.13 0.03    0    0
 containe      5829       1    0   20 0.13 0.07 0.07    0    0
 httpd         5883    5856    0   20 0.15 0.10 0.05    0    0
 httpd         5884    5856    0   20 0.13 0.08 0.05    0    0
 httpd         5885    5856    0   20 0.13 0.08 0.05    0    0
 containe      6009       1    0   20 0.12 0.07 0.05    0    0
 httpd         6065    6032    0   20 0.12 0.08 0.03    0    0
 httpd         6066    6032    0   20 0.13 0.10 0.03    0    0
 httpd         7079    5856    0   20 0.12 0.08 0.03    0    0
```

Docker containers (configuration, CPU)

- Note docker assigns names if not provided

```
Report: ESADOCK1      DOCKER Configuration Report
-------------------------------------------------------------------
Time /    <-----------Container Configuration----------> Status
Node      ContainerName     ImageName  Index         Procid
--------  ----------------- ---------- ------------- ----- ------
15:14:00
zcxinst1
          clever_ramanujan  localhost/ 7f1752ffc4e3   5752 runn
          httpd1            httpd      2c9d7574ca87   6032 runn
          httpd2            httpd      1fca2a98a85e   5856 runn
          stress2           stresscpu  1c35d9534623   5518 runn
          stress1           stresscpu  aa927ba72ee6   5655 runn
          ibm_zcx_zos_ssh_  ibm_zcx_zo a3cfd896b4d7   3436 runn
```

Docker containers (configuration, CPU)

- Note docker assigns names if not provided – clever_ram is snmp
- NOTE: DOCKER API GIVES INCORRECT CPU
- Httpd containers getting hit many times a second
- Stress containers cpu intensive

```
Report: ESADOCK2      DOCKER Transaction Report
-------------------------------------------------------------------------

                                                          <--Container Me
Node      <---------Container------------> <CPU Percent> <---Storage met
          Name         Index         ProcID   User  System   Size RSS Peak
--------  ---------- ------------ -------  ----- ------  ---- ---- ----
15:14:00
zcxinst1 clever_ram 7f1752ffc4e3    5752    0.0     0.1  31.8  9.3 31.8
          httpd1     2c9d7574ca87    6032    0.1     0.3  3553  8.6 3617
          httpd2     1fca2a98a85e    5856    0.2     0.3  4736  7.7 4784
          stress2    1c35d9534623    5518   40.9     0.3   4.1  0.8  4.1
          stress1    aa927ba72ee6    5655   41.0     0.3   4.1  0.9  4.1
          ibm_zcx_zo a3cfd896b4d7    3436      0       0   0.9  0.0  0.9
```

From SMF 70 (we have one zip in SMT mode)

- V25A is our native z/OS LPAR
- ZIP is in SMT, there are two threads reported

```
Report: ZOSCPU          Z/OS CPU Report
-----------------------------------------
TIME/    <--CPU--> Sample <-CPU Utilization>

SYSID    ID    Type Count   Total   Wait Parked
------   ----  ---- -----   -----  ----- ------
15:12:00 - 15:13:00

V25A        0    GP     1     9.9   90.1      0
            2    GP     1     3.0   97.0      0
            4  zIIP     1    47.6   53.2      0
            5  zIIP     1    45.2   54.8      0
          ---  ---- -----   -----  ----- ------
          Tot    GP     2    12.9  187.1      0
          Tot  zIIP     2    92.8  108.1      0
```

From SMF 70 – LPAR "assigned time"

- One core assigned, two concurrent threads

```
Report: ZOSLPARS      Z/OS LPAR Summary Report
Monitor initialized: needinit at 15:12:00 on Z15S serial 0E0F78
-------------------------------------------------------------------
                                        <-------OnLine CPU---------->
TIME/     <--Logical Partition---> <-CPU--> <---%Assigned------>
Date      Name      ID Serial SYSID Type Cnt Total Virtual Entitl
--------  --------  -- ------ ----- ---- --- ----- ------- ------
15:12:00 - 15:13:00
          ZOSLP1    11 0E0F78  V25A GP     2  13.1    12.9   0.44
          ZOSLP1    11 0E0F78  V25A IIP    1  47.8    47.6   0.33
```

From SMF 30, only one user of ziip

- Why is “zip on CP”, and why “GP CPU percent”?
- OpenShift is not “free” in terms of GP requirements

z/OS Job/Step CPU/Resources

SYSID	<------JOB---------->			<--CPU Percents-				<---ZIP Pct--->			
	Name	JobID	Step Nbr	Total	STD	SRB	I T	Tot	Enc	Dep	on CP
15:13:00 - 15:14:00											
V25A	Totals		.	8.92	4.45	3.95		59	0	0.2	1.6
	INST1	STC05196	1	2.88	1.43	1.10		59	0	0.1	1.4
	ZOSMNVM2	STC04764	1	0.02	0.02	0		0	0	0	0
	ZOSMNVM4	STC04762	1	0.02	0.02	0		0	0	0	0

So, do we have accurate data?

From SMF 70 – LPAR "assigned time"

- One zip core assigned: 47.6% (smf70) (capacity planning/chargeback)
- two concurrent threads: 92.8% (smf70, 46% average thread))
- Job data zip time 59% (smf30)
- Linux (total thread time): 88.4% (snmp)

- Why is "zip on CP", and why "GP CPU percent"?

Thread time looks valid within 5%

Did IBM prorate the SMF30 CPU time for SMT? If so, again incorrect

Solution to get accurate (or at least much much better) cpu data:

- prorate the SMF30 ZIP to / SMF70 HMC Assigned
- Apply to Linux container CPU data

Summary

OpenShift / Kubernetes, Docker has the data for:

- Performance Analysis
- Operational Alerts
- Capacity Planning
- Chargeback

Data collection with snmp

- Inexpensive
- Validated
- Measurable by container
- z/OS data needs effective prorate technology

Thank you.