PowerVC 1.3.1

This video assumes:
- You already know PowerVC version 1.3
  - So I'll cover the new features

If you have not used PowerVC:
  Watch this YouTube video
  - Power Systems Virtual User Group
  - Session 52 – January 2016 roughly 90 minutes
  - Name PowerVC 1.3.0
  - https://www.youtube.com/watch?v=yOTuavd0dc0
    ~660 views so far plus the live audience on the day

→ My PowerVC YouTube videos have 16,000++ views in 2016 & growing
IBM PowerVC 1.3.1

Announced: April 2016
GA Released: 17th June 2016

Comes in two editions
1. PowerVC Standard Edition
2. Cloud PowerVC Manager Edition

PowerVC 1.3.1 Redbook available
– Already pretty good
– Draft under rapid writing & reviews until released ~August 2016

Nigel’s 1.3.1 Announcement Highlights

IBM Cloud PowerVC Manager V1.3.1 includes NEW THINGS:
1. AIX VM - Active Memory Expansion
2. Dynamic Resource Optimiser – can be scheduled
3. Remove a VM but keep the storage “just in case”
4. Internal database moved from DB2 to MariaDB
5. Novalink improvements [not covered in this session]
6. Project & user role admin for separating resources: VMs, Disks, Images
7. Hitachi Storage Provider – Postponed
8. IBM Cloud Manager merging in to PowerVC
   – Self-service user has limited actions
   – User role “self-service”
   – Deploy Templates from their project(s) only
   – Policy based with or without admin approval process
   – VM expirations - admin or auto remove
   – Metering data that can be used for chargeback (REST API)

—

Comes in two editions
8. IBM Cloud Manager merging in to PowerVC
   – Self-service user has limited actions
   – User role “self-service”
   – Deploy Templates from their project(s) only
   – Policy based with or without admin approval process
   – VM expirations - admin or auto remove
   – Metering data that can be used for chargeback (REST API)

—

Cloud PowerVC Manager Edition adds to the Standard Edition

—
1 Active Memory Expansion

AIX only feature
Compresses memory for efficient use & performance

Switch on at the HMC + set the Expansion Factor
EF 1.0 = on but not active yet
EF 2.5 = 4GB looks like 10GB
Active Memory Expansion

- Configuration → Compute Templates
Advanced → Active Memory Expansion

```
Active Memory Expansion:
- Enable AVE
- AVE Expansion Factor: 25
```

Advanced → Simplified Remote Restart

```
Miscellaneous:
- Enable virtual machine remote restart
- Compatibility mode:
  - default
- Availability priority (0-255):
  - 127
```
Simplified Remote Restart (SRR) Update

- SRR v1 HMC 8.4 + current FW & VIOS in Dec 2015
  - LPM (static) from a dead machine + reactivate LPAR
  - Via HMC command line and/or PowerVC
  - Wrinkle: the dead machine must be a zombie (FSP alive)

- SSR v2 HMC 8.5 + current FW & VIOS in June 2016
  - By popular demand → it doesn’t need that FSP
  - Via HMC command line add the --noconnection option
  - Wrinkle: If it fails try the other HMC!
  - Wrinkle: PowerVC needs a small update to 1.3.1

- Also larger concurrency = 32 SRR at a time

Shared Processor Pool → SW license control

Check your software vendor supports CPU Pools for licensing
2 Dynamic Resource Optimizer (DRO)

New in 2015:
PowerVC using LPM to load balance servers or Enterprise Mobile CPU core license balancing

Now with scheduling periods

Dynamic Resource Optimizer (DRO)
- As before DRO is a Host Group feature
- Scheduling set-up here
Dynamic Resource Optimizer (DRO)

- Enable and …
- “Add Time Frame”

- Days of the week + Hours of the day
- Can have many periods

- Lets you decide when LPM or Enterprise Mobile cores changed

Dynamic Resource Optimizer (DRO)

- Example midday to 6 pm weekdays and any time at the weekend
- Not weekday morning = our busy time
- Click a period and you can adjust it (Edit)

Welcome Feedback on how you see this being used!
- Blobs are active periods
- Bar is now
Dynamic Resource Optimizer (DRO)

- Go to a particular Host's Details
- DRO actions when above the 70% threshold (settable)

3 Storage Life Cycle
Improved Lifecycle Control of Data Volumes

VM delete release CPU & RAM + removed the disks

Now you choose which disks to permanently destroy

- Example policy 1: Keep for 2 months as “dumb” users often want them back tomorrow!
- Example policy 2: Keep them to archive & destroy that evening

PowerVC 1.3.1
Install & Upgrade
List prices

- PowerVC Standard – additional features
- IBM Cloud PowerVC Manager
  - Name decided by committee!
  - Massive feature jump and more to come in Q4
  - Effective merge IBM Cloud Manager into PowerVC

<table>
<thead>
<tr>
<th>Power Systems Tiers</th>
<th>PowerVC Standard Edition License</th>
<th>IBM Cloud PowerVC Manager License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large</td>
<td>$160</td>
<td>$460</td>
</tr>
<tr>
<td>Medium</td>
<td>$120</td>
<td>$320</td>
</tr>
<tr>
<td>Small</td>
<td>$80</td>
<td>$160</td>
</tr>
</tbody>
</table>

- Price per core managed by PowerVC
- Confirmed by the Product Manager April 2016

Installation PowerVC 1.3.1

1. You install RHEL 7
2. Single media ➔ DVD or large file (700MB)
3. Unzip the .tgz file
4. Untar the .tar file
5. cd to the directory that tar created
6. -s option decides what gets installed:
   -s `standard` or -s `powerkvm` or -s `cloud_powervm`

7. Pre-test readiness ./install -s `cloud_powervm` -t
8. Install with ./install -s `cloud_powervm`
9. Takes about 30 minutes on fast disks

- If upgrading still use ./install ➔ it finds the older release
- If running PowerVC Standard Edition you can upgrade later to Cloud PowerVC Manager Edition
Install

I prefer

RHEL 7.1 on POWER8
Can be BE or LE
but can be x86_64 LE

Hints:

- On network – doh!
- Red Hat repositories connected up
- Install DVD repo

- Support is “RHEL7.1 or later”, so you can upgrade RHEL 7.1 & even use RHEL 7.2

DB2 to MariaDB

- Installing PowerVC 1.3.1
  - MariaDB installed automatically behind the covers

- If Upgrading to PowerVC 1.3.1
  - the DB2 database is moved automatically to MariaDB

- If I did not tell you then you may not have noticed !

```
# ps -ef | grep -i mariadb
mysql  3464  2961  0 Jun13  00:22:22 /usr/libexec/mysqld --basedir=/usr --
datadir=/var/opt/ibm/powervc/db --plugin-dir=/usr/lib64/mysql/plugin --log-
error=/var/log/mariadb/mariadb.log --open-files-limit=8192 --pid-
file=/var/run/mariadb/mariadb.pid --socket=/var/lib/mysql/mysql.sock --port=50110
```

```
[root@vm17 yum.repos.d]# rpm -qa | grep -i mariadb
mariadb-server-5.5.47-1.el7_2.ppc64
mariadb-ibs-5.5.47-1.el7_2.ppc64
mariadb-5.5.47-1.el7_2.ppc64
```
Projects & User Roles

Project & user roles for separating resources: VMs, Disks, Images

Why?
- Previously, one project “ibm-default” & one PowerVC super “admin role” owns everything & connected to the Linux user group “admin”
  - Likewise for user group deployer & viewer but less authority

- Now, different people (or teams) own & control their subset of PowerVC resources

- Makes for
  - Simpler management of larger PowerVC set-ups
  - Isolation of resources = reduced mistakes / multi-tenancy
Project and roles – “Old School”

- PowerVC 1.3.0 (& older) already knew about projects
  - EVERYTHING was in project: ibm-default
  - EVERYTHING access by admin Role via admin Linux user group

- Images to deploy

- Virtual Machines

- Admin Users also see
  - Host machines
  - Storage providers
  - Disk volume

Project and roles – new way

- Created three projects
  - Image/Deploy-Template
  - Virtual Machine

- Admin Users also see
  - Host machines
  - Storage providers
  - Disk volume
Project and roles – new way

- Manage a HMC → visible to admin user in all 3 projects
  - Users see only their project
    - Image/Deploy-Template
    - Virtual Machine
  - Admin Users also see
    - Host machines
    - Storage providers
    - Disk volume

Project and roles – new way

- Next manage the host Power machines → visible in all projects
  - Users see only their project
    - Image/Deploy-Template
    - Virtual Machine
  - Admin Users also see
    - Host machines
    - Storage providers
    - Disk volume
Project and roles – new way

- Manage Storage Providers visible from all 3 projects
- Shared Storage Pool found when managing VM using SSP

Users see only their project
- Image/Deploy-Template
- Virtual Machine

Admin Users also see
- Host machines
- Storage providers
- Disk volume

Project and roles – new way

- Now you can “see” VM’s, images and virtual Disks

Users see only their project
- Image/Deploy-Template
- Virtual Machine

Admin Users also see
- Host machines
- Storage providers
- Disk volume
Project and roles – new way

- Now you can “see” VM’s, images and virtual Disks
- But these resources can only be managed in one Project

- Users see only their project
  - Image/Deploy-Template
  - Virtual Machine

- Admin Users also see
  - Host machines
  - Storage providers
  - Disk volume

---

Project and roles – new way

- Now you can “see” VM’s, images and virtual Disks
- But these resources can only be managed in one Project

- Users see only their project
  - Image/Deploy-Template
  - Virtual Machine

- Admin Users also see
  - Host machines
  - Storage providers
  - Disk volume
Project and roles – new way

- Now you can “see” VM’s, images and virtual Disks
- But these resources can only be managed in one Project

- Users see only their project
  - Image/Deploy-Template
  - Virtual Machine

- Admin Users also see
  - Host machines
  - Storage providers
  - Disk volume

Project & Role Management:

Now in PowerVC 1.3.1+
1. Use openstack command to create a project (once for each)
   - This is mostly giving it a name
   - The default project is “ibm-default”

2. Next create a Linux user: # adduser
3. Use openstack commands to
   - Assign the new user to one (or more) project with a specific role

- There are 9 roles defined: See Redbook section 3.8.2
  - Previous Roles: admin, deployer, viewer
  - Six new ones → variations on the above three
  - Self-service → only for Cloud PowerVC Manager Edition
But first lets do something simpler

- By default root user has the admin role for ibm-default project
  - Not a good idea to use root id for accessing PowerVC

- So for security reasons
  - Create a powervc “master” user: “powervc”
  - With admin role for the default project “ibm-default”

- This should help you understand projects

Create new “master” user for GUI access

```
# adduser powervc -p SECRET
#
# openstack role add --project ibm-default --user powervc admin
```

Missing parameter(s):
- Set a username with --os-username, OS_USERNAME, or auth.username
- Set an authentication URL, with --os-auth-url, OS_AUTH_URL or auth.auth_url
- Set a scope, such as a project or domain, set a project scope with --os-project-name, OS_PROJECT_NAME or auth.project_name, set a domain scope with --os-domain-name, OS_DOMAIN_NAME or auth.domain_name

Ugh!!!
REST API experts might recognise some of this as standard info needed to “talk” to openstack services
Create new “master” user for GUI access

```bash
# adduser powervc -p SECRET
#
# source /opt/ibm/powervc/powervcrc
#
# export OS_USERNAME=root
# export OS_PASSWORD= <ROOT_PASSWD>
#
# openstack role add --project ibm-default --user powervc admin
#
Now you can login as Linux user: powervc at the PowerVC GUI webpage as the “master” user

However, CLI PowerVC commands still need to be run as root as in /opt/ibm/powervc/bin/...
```

So what is in that “magic” file? /opt/ibm/powervc/powervcrc
- Automatically created at install/upgrade time

```bash
export OS_IDENTITY_API_VERSION=3
export OS_REGION_NAME=RegionOne
export OS_CACERT=/etc/pki/tls/certs/powervc.crt
export OS_PROJECT_DOMAIN_NAME=Default
export OS_PROJECT_NAME=ibm-default
export OS_TENANT_NAME=ibm-default
export OS_USER_DOMAIN_NAME=Default
export OS_USERNAME=
export OS_PASSWORD=
export OS_COMPUTE_API_VERSION=2.25
export OS_NETWORK_API_VERSION=2.0
export OS_IMAGE_API_VERSION=2
export OS_VOLUME_API_VERSION=2
```
Create new “master” user for GUI access

1. To let powervc user run PowerVC CLI commands
   - cp /opt/ibm/powervc/powervcrc /home/powervc/powervcrc
2. Set the OS_USERNAME & OS_PASSWORD for root user
   - vi /home/powervc/powervcrc
3. Make it read/write only for powervc
   - chmod 700 /home/powervc/powervcrc
   - chown powervc /home/powervc/powervcrc
4. Add it to .bash_profile
   - cd /home/powervc
   - echo source /home/powervc/powervcrc >> .bash_profile
5. Then log out and back in again

Now powervc user can issue openstack commands

Now back to create project and setting user roles
Create a project

```
[root@vm19 ~]# openstack project create --description "Six Project" six
```

```
+--------------------------+-------------------+
| ID                       | Name              |
+--------------------------+-------------------+
| 138537788461455183a90a4b86142a65 | powervm |
| 45ce5455a81a4effa8af6cf5fd2b1e51 | ibm--default |
| 4ed9891daf44319eab573ed0b925e6 | six |
| aa0ed38f28154f0aaac3d60dec6b65 | service |
```

Create users and add them to a project

```
[root@vm19 ~]# adduser sixadmin -p 666 ↩ NOT good password
[root@vm19 ~]# adduser halfdozen -p 666 ↩ NOT good password
```

```
[root@vm19 ~]# openstack role add --project six --user sixadmin admin
[root@vm19 ~]# openstack role add --project six --user halfdozen admin
```

```
[root@vm19 ~]# openstack user list
```

```
+--------------------------+-------------------+
| ID                       | Name              |
+--------------------------+-------------------+
| ldd45dfca1d83b0e38       | powervc |
| 794d13683f9ab37b1d5      | sixadmin |
| 897d8369914c6109c4       | halfdozen |
```

Fixing a role — above halfdozen should have deployer role

```
[root@vm19 ~]# openstack role remove --project six --user halfdozen admin
[root@vm19 ~]# openstack role add --project six --user halfdozen deployer
```

WARNING: Not the “delete” sub-command
User role list

[root@vm19 ~]# openstack role list

+-----------------+----------------+
| Name            | ID              |
+-----------------+----------------+
| storage_manager | 0bd36ecc931549f5a0fe71a99c2a74e9 |
| viewer          | 135f12bbf7c449e195c47bdf4d6c216 |
| service         | 27547b439f9df709284b29525f4b921 |
| deployer_restricted | 4426e5bab864a12ab417d818ab229d |
| deployer        | 4bfe43c101949eaf3ae4fa2a921c6  |
| admin           | 72ad4c6b05d442cab7973a9d035a68b1 |
| image_manager   | f864d7e879164de583090c318e7c5302 |
| deployer        | fda52585536d476793d2764fa76c4042 |
+-----------------+----------------+

See the PowerVC Redbook for the full Role Descriptions

<table>
<thead>
<tr>
<th>admin</th>
<th>deployer*</th>
<th>vm_manager</th>
<th>vm_user</th>
</tr>
</thead>
<tbody>
<tr>
<td>deployer**</td>
<td>storage_manager</td>
<td></td>
<td></td>
</tr>
<tr>
<td>viewer</td>
<td>vm_manager</td>
<td></td>
<td></td>
</tr>
<tr>
<td>service**</td>
<td>storage_manager</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*deployer is deprecated (to be removed in later releases)
**service not for us
As sixadmin user

Rather boring as I have not discovered anything yet!

Change project but these is only one!

Can't change as sixadmin is only in one project
User with two projects can switch

4 Cloud PowerVC Manager Edition
IBM
Cloud Power Virtualization Center
Manager Edition

ICPVCME
Installing the Cloud Edition Features Changes

- New Role: self-service [mandatory]
- Manage Projects & User Roles [mandatory]
- Manage VM Deploy requests [optional]
- Manage VM Expiration [optional]
- You need at least 1 Deploy Image per Project [mandatory]
  - Currently these are not shared
- New Deploy-Templates resource [mandatory]
  - Deploy Image + Compute-Template
  - = one click Deploy with no questions for Self-Service users
- New power-cloud-config command for [mandatory]
  - Setting VM deploy, VM expire & Image Capture
- Metering usage stats [optional]
Don’t make my mistake!!

I crash & burn my PowerVC host so I then
  - Find all the HMC
  - Find the host machines
  - Find the VMs
  - Find the images

  - Now look for the new cloud / projects stuff

  - If you found everything its in the ibm-default project
  - There is nothing left for Cloud projects to manage!

  - So lets talk more about projects

My Projects and User roles

  - Super admin: powervc = admin for all projects
  - Project: six for POWER6 host group
    - Admin user: sixadmin
    - Deployer: halfdozen
    - Users: sixtus and sixtine and humble
  - Project: seven
    - Admin user: sevenadmin
    - Deployer: sevendep
    - Users: sevenself and humble
  - Project: eight
    - Admin user: eightadmin
    - Deployer: eightdep
    - Users: eightself and humble
How do we review the projects and role assignments?

- openstack project list
- openstack role list
- openstack user list
- openstack role assignment list
  
  - Oh dear! hexadecimal id’s → Yuck!

<table>
<thead>
<tr>
<th>Role</th>
<th>User</th>
<th>Group</th>
<th>Project</th>
<th>Domain</th>
<th>Inherited</th>
</tr>
</thead>
<tbody>
<tr>
<td>six</td>
<td>admin</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>eight</td>
<td>admin</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>eight</td>
<td>deploy</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>eight</td>
<td>self_service</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>seven</td>
<td>admin</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>seven</td>
<td>admin</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>seven</td>
<td>deploy</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>seven</td>
<td>self_service</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>six</td>
<td>admin</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>six</td>
<td>admin</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>six</td>
<td>deploy</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>six</td>
<td>self_service</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>six</td>
<td>self_service</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

"--names" can fix that but it is good to strip out some information we don’t need to see. Next slide

List Project + Role + User script:

```
openstack role assignment list -c Project -c Role -c User --names --f value | \
| grep -v @Service | sed 's/@Default//g' | \ 
| awk '{ printf "%-15s %-15s %-15s\n", $3, $1, $2 }' | sort
```

```
eight admin eightadmin
eight admin powervc
eight deploy eightdep
eight self_service eightself
seven admin powervc
seven admin sevenadmin
seven deploy sevendep
seven self_service humble
seven self_service sevenself
six admin powervc
six admin sixadmin
six deploy halfdozen
six self_service humble
six self_service sixtine
six self_service sixtus
```
powervc-config command - awkward

- Many PowerVC commands merging in this one
- This allows lots of settings like
  - PowerVC IP addresses
  - User web timeouts ← good
  - Max deploy image size
  - DNS domain defaults ← mandatory
  - Metering ← Cloud Edition more later

- See AIXpert blog
  - PowerVC 1.3.1 Cheat Sheet
    - for details on how to use this unhelpful command!
  - https://www.ibm.com/developerworks/community/blogs/aixpert/entry/PowerVC_1_3_1_Cheat_Sheet

powervc-cloud-config command - painful

- Projects with Self-Service users Settings
  - VM deploy needs Admin approval?
  - VM to have expiration date?
  - Image Capture limits?

- If not set then something’s do not appear on the GUI

- See AIXpert blog
  - PowerVC 1.3.1 Cheat Sheet
    - for details on how to use this command!
  - https://www.ibm.com/developerworks/community/blogs/aixpert/entry/PowerVC_1_3_1_Cheat_Sheet
powervc-cloud-config command - painful

- Need those powervcrc shell variables set

- This allows lot of settings like (my shorthand reminder)
  
  **default_request_wait_time**  Days waiting for VM approval before it’s automatically approved
  
  **deploy_approval_limit**  Number of VMs a user can have without needing approval
  
  **expired_resources_lifetime**  After a VM expired, the days before the VM is deleted
  
  **extension_approval_limit**  Number of VM user extensions before approval is needed
  
  **default_expiration_days**  Number for the default days before the VM is expired and a limit to the maximum days that can be requested
  
  **snapshot_approval_limit**  Number of VM Captures before approval needed

powervc-cloud-config command - painful

You must script this or you will get caught out

If OS_TENANT_NAME="ibm-default" then
  the --project six below is totally ignored!!

unset OS_TENANT_NAME
powervc-cloud-config --project six set-policy default_request_wait_time 5
powervc-cloud-config --project six set-policy deploy Approval_limit 3
powervc-cloud-config --project six set-policy expired_resources_lifetime 90
powervc-cloud-config --project six set-policy extension_approval_limit 4
powervc-cloud-config --project six set-policy default_expiration_days 30
powervc-cloud-config --project six set-policy snapshot_approval_limit 10
Project and User Control

- Linux user is allocated a role on specific project(s)
  - Admin full access
  - Self-service limited
  - Project Policy decides if VM needs approval

Project six

\[ \text{sixadmin} = \text{Admin with full access for project six} \]

\[ \text{ALL Resources:} \]
- Servers
- Networks
- Volumes
- Images
  - AIX 7.2 TL0 sp1
- ALL VMs
- Compute Template
**Project six**

sixadmin = Admin with full access for project six

ALL Resources:
- Servers
- Networks
- Volumes
- Images
  - AIX 7.2 TL0 sp1

ALL VMs
- Compute Template
  - # of CPU, GB RAM
  - Hosts group + settings

Deploy-Templates
- AIX 7.2 TL0 sp1

Self-service users

humble
vm1
vm2
vm3

sixtus
vm4
vm5

sixine
vm6

© Copyright IBM Corporation 2011
Project six

sixadmin = Admin with full access for project six

ALL Resources:
- Servers
- Networks
- Volumes
- Images
  - AIX 7.2 TL0 sp1
- ALL VMs
- Compute Template

Self-service users can’t see other users VMs

Deploy-Templates

AIX 7.2 TL0 sp1

Sixtus
vm4

Humble
vm1

Sixnine
vm6

Self-service users can only see and use their project’s Deploy-Templates

Self-service users can’t see other users VMs

Deploy-Templates

AIX 7.2 TL0 sp1

Sixtus
vm4

Humble
vm1

Sixnine
vm6

Self-service users can’t see other users VMs
If the Admin user makes a new Deploy-Template from the new capture all the self-service users can deploy it.
Users pay for their IT use?
- Multi-tenant or user department based charging

Currently, just raw stats - No invoice generation here!

User Metering via REST API

- Documented in the Manual
  - I found it very . . . challenging!
  - Zero information about what you get

- Data extracted via REST API via tools like these:
  - For scripting Curl (text) plus shell scripting
  - Python with JSON modules
  - GUI Firefox Rest Client
  - GUI Chrome Postman

- We need a clear worked example
User Metering via REST API

- From experiments
  - I use the curl command (text based) for scripts
  - It is already installed on PowerVC RHEL7.1+ server

- Data in JSON format
  - IMHO one the worst formats on the planet!
  - Extreme duplication in returned text 95% pointless
    - 50 timestamps & numbers ~1KB and 45KB file and on one line
    - The data is often in internal ID form that you then need to decode

- REST API + JSON = good for program access for you to develop a billing application!
User Metering via REST API

- **From experiments**
  - Project Data is taken every 10 minutes
    - It is configuration data
    - Project (admin) + User level (self-service)
  - Server Usage data is taken once a minute

- **Data available**
  - total_vcpu = Entitlement (not virtual CPUs)
  - total_memory = RAM in MB
  - total_volumes = Disk in GB (ignoring thin provisioning)
  - CPU Utilisation% per server (from HMC)

Sort of remotely login to PowerVC as a website

```
# export POWERVC=vm17.ache.com
# curl -l -X POST https://$POWERVC:5000/v3/auth/tokens
-H "Accept: application/json"
-H "Content-Type: application/json" -d @auth.json
```

Returns a TOKEN to use for the below GET data requests

Get a meter value, here Entitlement (called total_vcpu)
```
```

Similar for total_memory (RAM MB) and total_volumes (Diskspace GB)

Get Machine CPU use called samples
```
```

Then a few hours shell scripting to filter out the “noise” & for line by line CSV’s
```
2016-07-06T215810,1.5,sixtine
2016-07-06T215810,0.5,sixtus
2016-07-06T215810,6.0,sixadmin
2016-07-06T215810,2.5,humble
2016-07-06T215810,1.5,sixtine
```

And using awk to do a matrix inversion you get to usable CSV file (nmon format @)
```
Entitlement,sixtus,sixadmin,humble,sixtine
2016-07-06T215810,0.5,6.0,2.5,1.5
```

# export POWERVC=vm17.ache.com
# curl -l -X POST https://$POWERVC:5000/v3/auth/tokens
-H "Accept: application/json"
-H "Content-Type: application/json" -d @auth.json
Save events and data for 90 days = 2160 hours

Events are causes if stats pass a threshold that you set

```
# powervc-config metering event_ttl --set 2160 --unit hr
Setting event_time_to_live to 2160 hr
```

How much stats data you want to have stored

```
# powervc-config metering meter_ttl --set 2160 --unit hr
Setting metering_time_to_live to 2160 hr
```
DEMO
of Admin & Self-Service

Summary – Good time for Questions

IBM Cloud PowerVC Manager V1.3.1 includes NEW THINGS:

1. AIX VM - Active Memory Expansion ✓
2. Dynamic Resource Optimiser – can be scheduled ✓
3. Remove a VM but keep the storage “just in case” ✓
4. Internal database moved from DB2 to MariaDB ✓
5. Novalink improvements [not covered in this session]
6. Project & user role admin for separating resources: VMs, Disks, Images ✓
7. Hitachi Storage Provider → Postponed
8. IBM Cloud Manager merging in to PowerVC
   - Self-service user has limited actions ✓
   - User role “self-service” ✓
   - Deploy Templates from their project(s) only ✓
   - Policy based with or without admin approval process ✓
   - VM expirations - admin or auto remove ✓
   - Metering data that can be used for chargeback [REST API] ✓