Jim Brannan (jim.b@quali.com)

Abstract

Documents metrics implementation for environments using Raritan PDUs  
Updated: 21 Sept 2017

Power Metrics   
w/Raritan PDUs

[Integrates with BI Solution]

Contents

[Power Management 1](#_Toc493746135)

[Power Metrics 1](#_Toc493746136)

[Table Definitions 4](#_Toc493746137)

[PDU\_VIEW 4](#_Toc493746138)

[OUTLETS\_VIEW 5](#_Toc493746139)

[TotalPeakPower View 5](#_Toc493746140)

[Top10totalPeakPower\_VIEW 6](#_Toc493746141)

[Sisense Integration 7](#_Toc493746142)

Power Metrics w/Raritan PDUs

# Power Management

Resource Power Management is automated as part of CloudShell’s OOB setup and teardown orchestration. Power commands are also available to users. Whether a device is powered off at the end of a sandbox reservation is controlled by an attributes specifying the number of minutes to look-ahead for the next usage.

Although there are other benefits, the most common reason to control power is for energy conservation (power consumption and air conditioning cost reductions). So how can this savings be measured? Easy if the PDU provides power metrics for inlets and outlets. This solution accommodates the use case where not all the PDU outlets are dedicated to CloudShell managed resources.

This project covers the use case when Raritan PDUs are used in a master/slave pair per rack. This in-turn provides power on, off, and cycle commands for the resources. These commands are used by automation in setup and teardown. Raritan PDUs offer a rich set of real-time power measurements. Hence we only need to archive the right data, hourly, to enjoy significant savings.

Some devices must not be under power management, such as L2 Fabric switches or the COE GW to Cisco. Safest solution is to not map their power port in CloudShell. There is however a resource attribute for Power Management, a Boolean. The power control functions honors this setting so as to not power on/off a critical device.

# Power Metrics

All user reserveable devices and some of the infrastructure devices should be provisioned in racks and powered by Raritan PDUs. The power metrics solution depends on continued use of these PDUs or a compatible PDU from Raritan. At present the metrics assumes 3-phase input to the PDUs. A master/slave pair of PDUs per rack provides adequate outlet counts as well as the ability to feed each off a different AC Supply for redundancy. This use case was developed for an environment where the devices have power cord connections to both the master and the slave, however that is not required.

The Raritan Shell must be installed and the PDUs auto-loaded. Each resource then has its power connections mapped to one or more PDU outlets.

Power metrics are captured every hour. A windows task scheduler task runs the python script from a .bat file.

C:\CloudShell\PDU Metrics\raritan\power-metrics.py

**If this script is fired manually, the data logged off-schedule should be deleted from   
the database (SQL).**

The login credentials are read from CloudShell attributes of each PDU, as it is processed.

The metrics collector, in python, requires the Raritan libraries.

Inlet and outlet data is posted to a database, QualiEnv, by the python script. For performance, it uses the bulk data retrieval capabilities of the Raritan API.

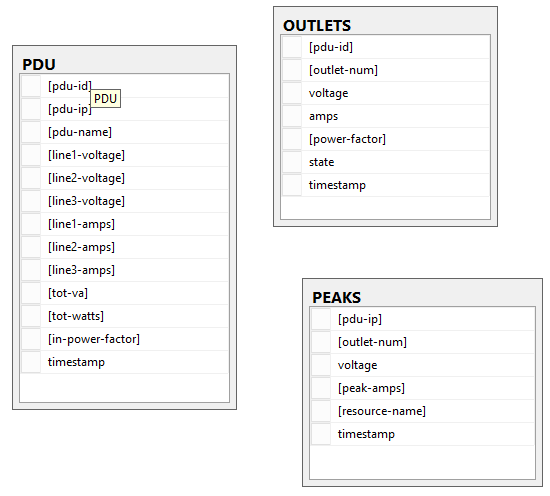
The data also includes a table of the all-time peak amperage read for each outlet. Multiple Views were created to aid reporting. There are two sources of data:

* CloudShell data is read to obtain the name of the resource connected to each outlet.
* The input and outlet power data (voltage, current, power factor) is read from each PDU

The data can be used for

* Management Reports illustrating power savings achieved by powering off devices not in near-term use. This is easily obtained by comparing the sum of all the outlet peak readings versus real-time actual.
* Determining gaps in deployment data (such as an outlet drawing power yet we have no record of a device connected to the outlet)
* DB user metrics created for this use.

The data base tables are as follows:



The PDU table is all about the INLET data and total consumption.  
The OUTLET table is all about the power drawn by resources.  
The PEAKS table retains the all-time highest reading for each outlet/resource.

To facilitate use of the data, SQL Views are used to access the tables. The views are particularly important in the presentation of the time-stamp of the data. We wish to report hourly however it takes time to extract the data from the PDUs. So the views present a version of the time-stamp with the minutes and seconds reverted to 0. The OUTLETS\_VIEW provides the actual and the special version of the time-stamp.

Reporting is done using Sisense. A few views of the data have been created to simplify reporting.

# Table Definitions

CREATE TABLE [dbo].[OUTLETS](

[pdu-id] [int] NOT NULL,

[outlet-num] [int] NOT NULL,

[voltage] [decimal](6, 2) NOT NULL,

[amps] [decimal](6, 2) NOT NULL,

[power-factor] [decimal](4, 3) NOT NULL,

[state] [varchar](4) NOT NULL,

[timestamp] [datetime] NOT NULL

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[PDU](

[pdu-id] [int] IDENTITY(1,1) NOT NULL,

[pdu-ip] [varchar](22) NOT NULL,

[pdu-name] [varchar](80) NOT NULL,

[line1-voltage] [decimal](6, 2) NOT NULL,

[line2-voltage] [decimal](6, 2) NOT NULL,

[line3-voltage] [decimal](6, 2) NOT NULL,

[line1-amps] [decimal](6, 2) NOT NULL,

[line2-amps] [decimal](6, 2) NOT NULL,

[line3-amps] [decimal](6, 2) NOT NULL,

[tot-va] [decimal](8, 1) NOT NULL,

[tot-watts] [decimal](8, 1) NOT NULL,

[in-power-factor] [decimal](4, 3) NOT NULL,

[timestamp] [datetime] NOT NULL

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[PEAKS](

[pdu-ip] [varchar](28) NOT NULL,

[outlet-num] [int] NOT NULL,

[voltage] [decimal](6, 2) NOT NULL,

[peak-amps] [decimal](6, 2) NULL,

[resource-name] [varchar](80) NULL,

[timestamp] [datetime] NOT NULL

) ON [PRIMARY]

GO

# Views

## PDU\_VIEW

CREATE VIEW [dbo].[PDU\_VIEW] as

(SELECT [pdu-ip], [pdu-name], [line1-voltage], [line2-voltage], [line3-voltage],

[line1-amps], [line2-amps], [line3-amps],

[tot-va], [tot-watts], [in-power-factor],

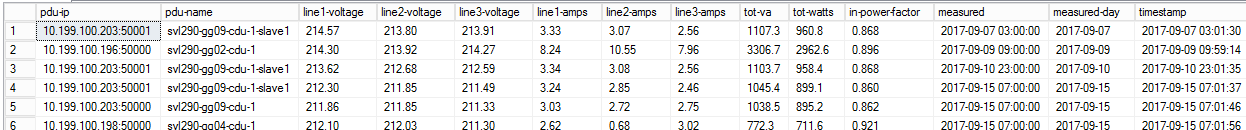
convert(varchar, (DATEADD(hh, DATEDIFF(hh,0,[timestamp]), 0)), 120) as measured,

convert(date, [timestamp]) as [measured-day],

convert(varchar, [timestamp], 120) as [timestamp]

FROM PDU

)



Example use: For a chart of total VA versus time, you’d simply chart the tot-va column against time. Using the inlet data makes quick work of solution-wide metrics. Be careful if you have a mix of resources in the racks that ate not part of the CloudShell based solution.

## OUTLETS\_VIEW

CREATE VIEW [dbo].[OUTLETS\_VIEW] as

(SELECT OUTLETS.[pdu-id],

[pdu-name],

OUTLETS.[outlet-num],

PEAKS.[resource-name],

OUTLETS.[voltage],

OUTLETS.[amps],

OUTLETS.[power-factor],

OUTLETS.[state],

OUTLETS.[voltage] \* OUTLETS.[amps] as VA,

convert(varchar, (DATEADD(hh, DATEDIFF(hh,0,OUTLETS.[timestamp]), 0)), 120) as measured,

convert(datetime, OUTLETS.[timestamp]) as measuredDT,

convert(date, OUTLETS.[timestamp]) as [measured-day],

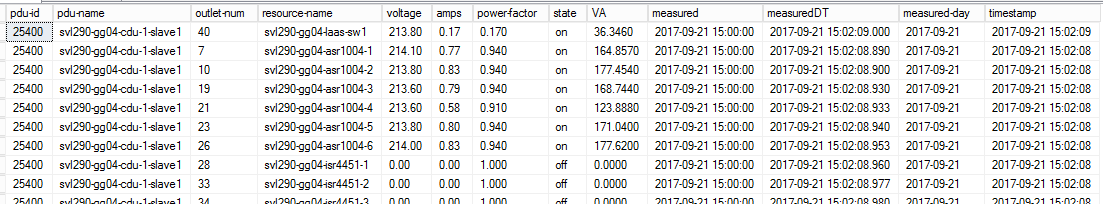
convert(varchar, OUTLETS.[timestamp], 120) as [timestamp]

FROM OUTLETS

JOIN PDU on OUTLETS.[pdu-id] = PDU.[pdu-id]

JOIN PEAKS on PDU.[pdu-ip] = PEAKS.[pdu-ip] AND OUTLETS.[outlet-num] = PEAKS.[outlet-num]

WHERE PEAKS.[resource-name] <> 'none')



Example use: determine which resources are the big consumers. Are they over configured? Can they be replaced with lower power units to save cost? Or what about any with LOW power factors? Is the 6880 one of these? Graph VA over time. Etc.

## TotalPeakPower View

This view returns all-time-high power use data per outlet. Use this to find the big consumers. Note that this is NOT filtering out resource-name=’none’. Link to other tables in Sisense to grab other resource meta-data.

CREATE VIEW [dbo].[TotalPeakPower] as

(SELECT [resource-name],

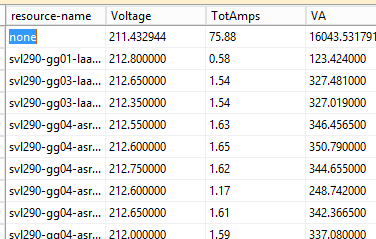
AVG([voltage]) as Voltage,

SUM([peak-amps]) as TotAmps,

(AVG([voltage]) \* SUM([peak-amps])) as VA

FROM PEAKS

Group By PEAKS.[resource-name])



## Top10totalPeakPower\_VIEW

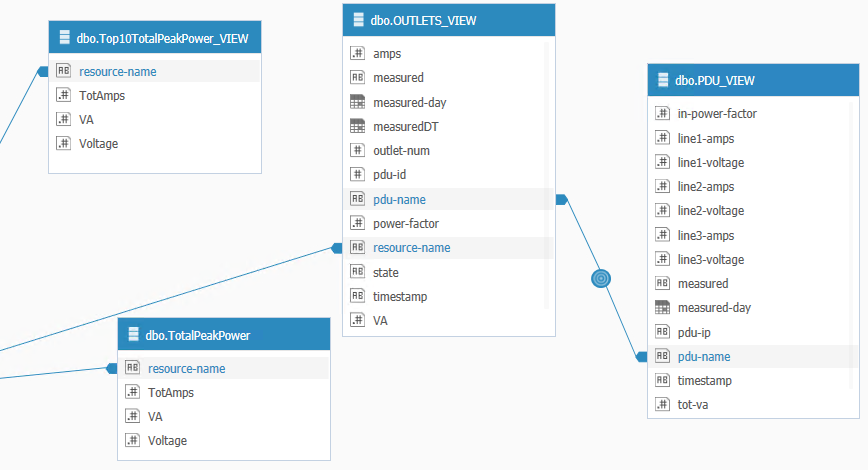
Sort of unnecessary as it is easy to do in Sisense, but this view was created anyway. This is the same as the view above except it returns just the top 10 consumers by power.

CREATE VIEW [dbo].[Top10TotalPeakPower\_VIEW] AS

select TOP 10 \* from TotalPeakPower order by TotAmps desc

# Sisense Integration

The ElastiCube should have all 4 views added and linked to other tables.



The resource-name links go to ResourceLog.ResourceName.