Accuracy of Energy Model Calibration with IPMI

Richard Kavanagh, Django Armstrong and Karim Djemame
School of Computing, University of Leeds, Leeds, LS2 9JT, UK
Email: {scsrek,scsdja,scskd}@leeds.ac.uk

***Abstract*—Energy consumption in Cloud computing is a significant issue and affects aspects such as the cost of energy, cooling in the data center and the environmental impact of cloud data centers. Monitoring and prediction provides the groundwork for improving the energy efficiency of data centers. This monitoring however is required to be fast and efficient without unnecessary overhead. It is also required to scale to the size of a data center where measurement through directly attached Watt meters is unrealistic. This therefore requires models that translate resource utilisation into the power consumed by a physical host. These models require calibrating and are hence subject to error. We discuss the causes of error within these models, focusing upon the use of IPMI in order to gather this data. We make recommendations on ways to mitigate this error without overly complicating the underlying model. The final result of these models is a Watt meter emulator that can provide values for power consumption from hosts in the data center, with an average error of 0.20W.**

I. Introduction

Energy efficiency in Cloud computing is fast becoming a primary concern of Cloud providers. Cloud computing is undergoing rapid adoption which is consequently giving rise to a dramatic increase in energy consumption and its associated cost. Data centers consequentially are placing an ever increasing importance on attempts to save on energy consumption. Accurate and timely information regarding power consumption is hence important in establishing ways to mitigate both the energy consumed and the overall cost.

To this end we present tools that enable the measurement of the energy efficiency of service deployments in Cloud environments, focusing on energy modelling, profiling capabilities and upon calibration of models. The monitoring tool is capable of modelling, measuring and reporting on energy efficiency for both billing and reporting purposes. This tooling has the ability to utilise various data sources such as baseboard management controllers (BMCs) that have various sensors for reporting on the physical hosts. These sensors include measurements for the energy and power consumed of a physical host and are able to report this using the Intelligent Platform Management Interface (IPMI). These sensors are however subject to error and cannot be practically substituted with more accurate Watt meters on a per machine basis. These sensor inaccuracies diminish the overall accuracy and usefulness of our models that are used to attribute power consumption to virtual machines (VMs). We therefore investigate strategies to mitigate this error and improve the reporting accuracy of our tools. The eventual aim of this tooling is to assist developers in understanding and minimising their overall energy consumption, including in practical situations where sensor accuracy may be limited.

This paper's main contributions are:

- recommendations on how to calibrate energy models, with the aim of reducing error.
- a comparison IPMI gathered power measurements vs Watt meter measurements with discussion on the impacts on accuracy.
- discussion regarding actual error in energy models and a demonstration on how to report this error.
- we illustrate the use of segmented linear regression as a means to overcome non-linearity in power vs CPU utilisation which avoids over fitting calibration data with high order polynomials.

The remaining structure of the paper is as follows: The following section covers the related work. In Section III we discuss energy modelling within our framework and the allocation of power consumption to virtual machines (VMs). Followed by a discussion of the key points to create good calibration data for such models in Section IV. We then perform and evaluation in Section V discussing the accuracies of IPMI based power sensors and how they might be utilised to calibrate a model. We finally conclude and discuss future work in Section VI.

II. Related Work

The characterisation of the resources is an important step in regards to accurate energy predictions for software usage. This gives rise to profiling and testing frameworks such as JouleUnit [1] that enable the profiling of hardware systems in order to understand their power consumption profiles. In order to utilise profiling in a distributed system monitoring frameworks such as Zabbix [2] or other frameworks [3] may be used. Kwapi [4] is the most closely related monitoring tool to our own work, in that it focuses on power and energy monitoring, however the focus of our framework is upon extending this to both VMs and applications.

Data for resource's power consumption is principally obtained either by direct measurement [5] or inferred via software and physical performance counters [1], [6]. Direct measurement obtains the wall power [6] value via the use of Watt meters [5], providing an aggregation of the current power usage of a physical resource [7]. Performance counters [8]–[10] are a non-invasive means of determining energy usage, by utilising performance counters located within the CPU and Operating System. Wall power measurements have the advantage of accuracy but require the specialist physical hardware to be attached into the infrastructure, while the performance counters are indirect measures of power consumption and requires a model to derive an estimate of the energy consumed. IPMI offers, the potential for direct measurement of power consumption based on sensor's integrated into the physical host, thus it is non-invasive like performance counters and offers the potential for high accuracy as well, although this accuracy is not always realised, thus models are required.

In order to determine VM or host energy usage various frameworks have been developed. The majority of cases use linear models [6], [8], [11], [12], which we show is not always representative of what actually occurs in real systems. Schubert et al. [13] remark how easy it is to get calibration wrong with such models especially when averaging or aggregation is used. In most cases linear models have provided power estimates with a high degree of accuracy for VMs and their underlying resources, usually within 3W of the actual value or within 5% error. Additive models such as [6], [11] utilise load characteristics for each of the major physical components such as CPU, disk and network, each of which is considered separately and summed together. In these cases idle power consumption is treated as an additional model parameter that is simply added to the other load characteristics. There are others that use a bias mechanism [8] or where each parameters importance is learned [10]. The use of performance counters can also differ amongst existing models, such as physical meters being only needed during an initial training phase followed by the use of counters post training [14].

The second concern after profiling a physical host's power consumption is to determine its future energy consumption, which can then be used to guide both the deployment and operation of VMs. Estimating future energy consumption requires an understanding of the VMs workload over time. This can include CPU load prediction in models such as LiRCUP [12] which is aimed at assisting in the maintenance of service level agreements and others [15] that search for workload patterns. Workload prediction has enjoyed a lot of attention with a particular focus on the cloud property of the scaling of resources and the maintenance of QoS parameters [16]–[18]. Workload prediction in Clouds has also been seen as a means to plan future workloads so that physical hosts may be switched off when not required [19], but may also be used to as a basis of the prediction of future power consumption. This work focuses on errors introduced by measurements during calibration, but for long term predictions, accurate estimates of workload are very important.

III. Energy Modelling

Energy modelling has several key functions within a data center. The first is the discovery of the amount of energy consumed where it cannot be directly measured and the second regards the adaptation to these values in regards to mitigating the energy consumed. These models are realised within our monitoring framework as the IaaS Energy modeller and the Watt Meter emulator [20], [21].

The IaaS Energy Modeller has three main roles, the first is at deployment time when the VM Manager utilises power consumption predictions for the placement of VMs. The second is at operation time when the VMs are monitored and this information is utilised to aid adaptation. The third covers the aspect of billing and monitoring, ensuring energy usage can be monitored and potentially charged for. In each case the IaaS Energy Modeller is required to attribute power consumption to both existing VMs and those that are scheduled to be deployed.

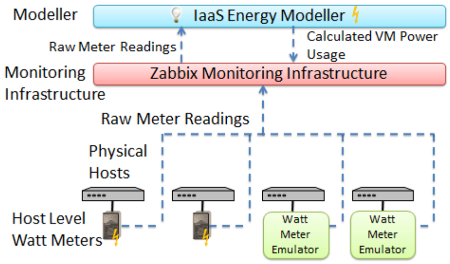

Fig. 1: Energy Modelling in the IaaS Energy Modeller

The energy monitoring in the IaaS Layer is shown in Figure 1. At the lowest level the monitoring utilises Watt meters [22] that are attached to the physical host machines. The data from these meters is published in Zabbix [2]. The values for host power consumption is then read by the IaaS Energy Modeller. The Energy Modeller's main role is to assign energy consumption values to a VM from the values obtained at host level. This is needed because energy consumption associated with VMs is not a directly measurable concept. Rules therefore establish how the host energy consumption is assigned to VMs. The host energy consumption can be fractioned out in one of several ways, within the Energy Modeller, which is discussed below:

CPU Utilisation Only: This uses CPU utilisation data for each VM and assigns the energy usage by the ratio produced by the utilisation data. (Available for: Historic, Current, Predictions). This is described in the Equation 1 where VM_P_x is the named VM's power consumption, $Host\_P$ is the measured host power consumption. VM_Util_x is the named VMs CPU utilisation, VM_Count is the count of VMs on the host machine. VM_Util_y is the CPU utilisation of a member of the set of VMs on the named host.

$$VM\_P_x = Host\_P \times \frac{VM\_Util_x}{\sum_{y=1}^{VM\_Count} VM\_Util_y} \qquad (1)$$

CPU Utilisation and Idle Energy Usage: Idle energy consumption of a host can also be considered. Using training data the idle energy of a host is calculated. This is evenly distributed among the VMs that are running upon the host machine. The remaining energy is then allocated in a similar fashion to the CPU Utilisation only mechanism. (Available for: Historic, Current, Predictions). This is described in Equation 2 where $Host\_Idle$ is the host's measured idle power consumption. This provides an advantage over the first method in that a VM is more appropriately allocated power consumption values and prevents it from using no power while it is inactive.

$$VM\_P_x = Host\_Idle + (Host_P - Host\_Idle) \times \frac{VM\_Util_x}{\sum_{y=1}^{VM\_Count} VM\_Util_y} \tag{2}$$

Evenly Shared: In the case of predictions CPU utilisation is clearly not easy to estimate, thus predicted power consumption can instead be evenly fractioned amongst VMs that are on the host machine. The default for predictions is to share out power consumption evenly as per Equation 3, this is chosen as it relies less upon forecasting individual VM workloads and is hence favourable given the potential inaccuracies. A slight variation also exists which counts the CPU cores allocated to each of the VMs and allocating power based upon this count (Equation 4). Equations 3 and 4 describe this even sharing rules where $Host\_Predicted$ is the amount of power that the host on which the named VM resides is estimated to utilise This value is derived from an average of the most recent measurements. VM_VCPU_x is the amount of virtual CPUs allocated to the named VM while VM_VCPU_y is the amount of virtual CPUs allocated to a VMs on the named host.

$$VM\_P_x = Host\_Predicted \times \frac{1}{VM\_Count} \tag{3}$$

$$VM\_P_x = Host\_Predicted \times \frac{VM\_VCPU_x}{\sum_{y=1}^{VM\_Count} VM\_VCPU_y} \tag{4}$$

The default method chosen on the IaaS Energy Modeller is Equation 2 for current and historic values and 3 for predictions. Once the Energy Modeller has assigned energy values to a given VM it then writes these values to disk, which are then reported back to the monitoring infrastructure, thus providing VM level power consumption values to the PaaS layer.

The Watt meter emulator is a tool for dealing that prevents the need for having Watt meters attached to every physical host in the data center, thus enabling monitoring at scale. It utilises recent utilisation information and an energy calibration model to decide what the current power consumption of a physical host is. In doing so it removes the requirement for attaching Watt meters to every physical host.

IV. Data Collection and Calibration

In order to gather sensor data there are two principle sources for monitoring infrastructures such Zabbix to collect data from. The first is reporting data from the operating system, which can utilise special structures such as /proc/ on Linux. The second is to use more specialist hardware such as baseboard management controllers (BMCs) and standardised interfaces. This can include aspects such as CPU performance counters as well as standardised interfaces such as IPMI. IPMI allows for sensors that are integrated in current generation server hardware to be accessed over a common API. The sensors that have traditionally been used to remotely manage and monitor larger clusters of physical machines and are starting to include power sensing capabilities. The integration of IPMI with the Zabbix monitoring infrastructure can be achieved through the use of `libopenipmi (v2.0.21)`, a library for interfacing to a large range of vendor specific BMC devices, with Zabbix. This enables the sensor data to be periodically scraped and stored.

The data gathered by these sensors can the be utilised to generate a model, that can calculate the power consumed based on utilisation. Errors in the values reported by the models that drive the energy modeller and Watt meter emulator can occur at two stages. The first is the calibration phase and the second is at operation time.

The calibration phase results in an inaccurate model that does not correctly represent the relationship between load and power consumption. This can occur for several different reasons:

Unsynchronized metric update intervals for different metric types: This could occur when measuring CPU utilisation and power together. For calibration to be accurate it requires the measurements to be perfectly synced or for the utilisation to remain stable during a measurement phase, so that both measurements represent the physical host's true state.

Measurement arrival latency (Monitoring infrastructure overhead): Differing on the above case, where synchronisation issues may occur, this is caused by the inherent delays in taking a measurement, transferring the value across a network and recording it in the monitoring infrastructure. This effects the detection of the start and end of periods of induced load. This can be mitigated by performing the calibration run locally without the use of a full monitoring infrastructure, such as Zabbix, Ganglia etc. This however will only work during the calibration and will not work during normal operations. Locally monitoring load will however have the side effect of measuring a small amount of load induced by itself.

Averaging and time windows of measurement's values: Measurements arrive with a given polling interval, however measurements such as CPU load also have a time window in which the measurement was taken e.g. over the last minute. This averaging causes errors in the model and requires the CPU utilisation measurement window to be made as small as possible. One alternative is for measurements used in the calibration dataset to only start to be taken after load has been induced for a time that is longer than the length of the averaging period. The former option is simpler but requires custom scripts in the case of the Zabbix monitoring environment.

Update interval of a sensor's reported value Sensors such as power measurements taken over IPMI update slower than the interval at which the baseboard can be queried. Thus rapid polling of the interface can result in the previously reported value been reported again, without prospect of change. Hence the poll interval should not exceed this update interval. In the case of IPMI power values polling is hence restricted to every 5 seconds in this paper's experimentation.

This therefore provides the basis of several recommendations which we implement in this paper that should be followed while calibrating an energy model:

- to use metrics that represent the physical host in its most recent state, which we call spot metrics and tend to avoid averaging and representing long periods of time
- that load should be induced followed by waiting a set period of time for the values to stabilise and then taking measurements. A further addition to this is to detect plateaus in the measured values and only using congruent data points, which can be used as a mechanism to determine how long to wait before accepting measurements as being valid.
- to take measurements locally thus avoiding monitoring system overheads including network delays.

In contrast to calibration time, delays in the arrival time of measurement data or purposefully averaging recent utilisation data at operation time does not matter as much and in some cases is useful. Using a longer time window for a measurement can be used to generate a smoothing effect on the data at the cost of responsiveness and overall accuracy. It mitigates change in values and slows response times but it can also avoid rapid fluctuating estimated power consumption values upon which a decision about deployments may be based.

V. Evaluation

The evaluation performed in this section focuses upon the evaluation of the accuracy of BMC devices and attached sensors accessed via IPMI for the purpose of measuring power consumption of servers within a Cloud based infrastructure. The aim of the experimentation is to explore the suitability of the built in power measuring functionality for measuring the power consumption of an application or a virtual machine.

The experimentation follows the energy modeller's calibration process which involves inducing load at selected present values onto a physical host and measuring the power consumption that the load causes.

A. Experimental Setup

The experimentation was performed on a Cloud testbed, that uses Open Nebula 4.10.2 [23] and Zabbix 2.4.4 [2] for monitoring. The physical host that was measured is a Dell PowerEdge R430 Server commodity server that is monitored through IPMI. The physical host tested has two 2.4GHz Intel Xeon E5-2630 v3 CPUs with 128GB of RAM, a 120GB SSD hard disk and an iDRAC Port Card that is IPMI 2.0 compliant. For the purpose of creating a baseline to compare IPMI based power meter values a WattsUp Meter Pro [22] is attached, with an accuracy of +/- 1.5%. The readings from the Watts Up meter were taken every second and reported to Zabbix and from the IPMI sensor it was every 5 seconds. In post processing the values reported by IPMI were interpolation, in order to compare data to the Watt meter. The IPMI sensor uses an inbuilt time window of 60 seconds. Zabbix was installed on a separate server as to the host undergoing measurement as to avoid unnecessary additional load. The physical host used network attached storage (NAS) that was used for VM images. This NAS was backed by a PowerEdge R730xd server with an Intel Xeon E5-2603 v3 CPU, with 64GB of RAM, 48TB Hard disk space with an additional 400GB SSDs for caching with a 4Gb bonded network connection.

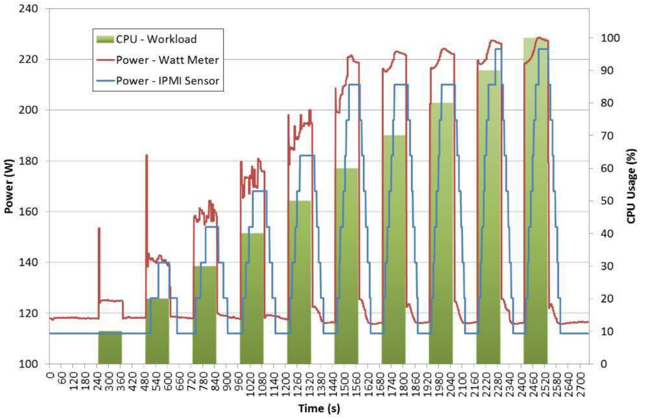

Fig. 2: Trace of IPMI and Watt meter measurements with incrementing CPU load

The load induced on the physical hosts ranges from 0% CPU usage upto 100% in increments of 10%. In order to generate this load a tool called `Stress` [24] is used, along with `cpulimit` and `taskset`. In order to generate full load 32 threads were launched and then mapped using `taskset` to the CPU cores on the physical host. `cpulimit` was used to set the intended load and at each interval of induced load, it was induced for 120 seconds. In order to represent a realistic setup for the physical host the CPU scheduling governor was set to the default option of on demand and hyper-threading was enabled with all sleep states been available.

B. Results & Discussion

In Figure 2 the overall trace of the calibration run is shown. It shows multiple measurements for each set CPU utilisation level been gathered via IPMI and the Watt meter along with the CPU load induced on the physical host. The Watt meter at the start of some periods of induced load especially at 10% and 20% CPU load shows spikes, before the load settles. This is in contrast to the IPMI sensor that is unable to detect any change in power consumption at 10% CPU load. This is due to the granularity of the sensor. It exhibits only 9 distinct values bands within the measurement range used (112W - 224W in 14W increments). The initial measured idle is 117W while at 10% load it is 124W and with only 7W difference this is undetectable using IPMI.

IPMI undergoes averaging, which results in the peak associated with IPMI been offset to the right of the Watt meter's reported values. This suggests that if accurate calibration is desired that these values should only be used after the average window has passed while sustained consistent load is in effect. The IPMI power values also under report the power consumption by seemingly only rounding down towards the last permissible increment.

At 60% CPU utilisation and above we notice that the system's power consumption becomes capped at around 228W,

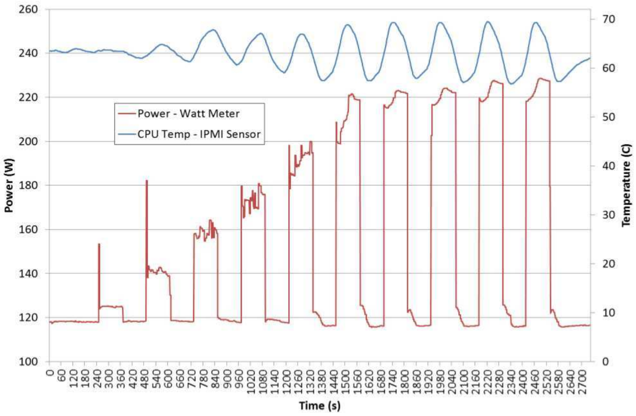

Fig. 3: Trace of Watt meter and temperature measurements with incrementing CPU load

after this point we speculate the CPU is throttled to meet it's TDP (Thermal Design Power). It can therefore be seen that a purely linear model as seen in much of the literature does not apply in the context of our machine.

In Figure 3, we examine the effect of temperature measured by IPMI on the power consumption to examine the high than expected variance in power during the sustained 120 second workload. The correlation between CPU load and CPU temperature can clearly be seen. The temperature at the start of our experiment before any load is induced starts at 63°C, yet lowers to 53°C at the lowest point during our experiment, which occurs soon after a load period has completed and is a result of the fans cooling the CPU past its normal idle temperature. At 50% CPU utilisation and above in our test setup, the power consumption as reported by the Watt meter shows an initial slope and then a tail in which the power consumption doesn't immediately drop down to idle once the load has finished. We speculate that this is the effect of Ohm's law and the increased resistance caused by the higher operating temperature of the CPU, in addition to the power consumption induced by the fans as part of the increased requirement for cooling. Thus as the CPU further heats at the start of a load period an initial slop is created due to heating and the increase in fan speed. The power consumption stabilises and then at the end of the load period drops, yet the remaining additional heat takes time to dissipate, thus causing the tail.

In Figure 4 we show the CPU load and power consumption calibration data processed from the raw data (shown in Figure 2) where all the data points over the 120 seconds of each workload are averaged. Standard deviation is illustrated via vertical error bars. This data is used in estimating power consumption from CPU load. We see that IPMI consistently under reports the power consumption and also the overall energy consumed. The error is also larger particularly when the CPU load is higher. This error is due to the averaging window that the IPMI device is using when taking measurements. It can also be seen as in Figure 2 how at 10% CPU utilisation that IPMI doesn't register the change in power consumption.

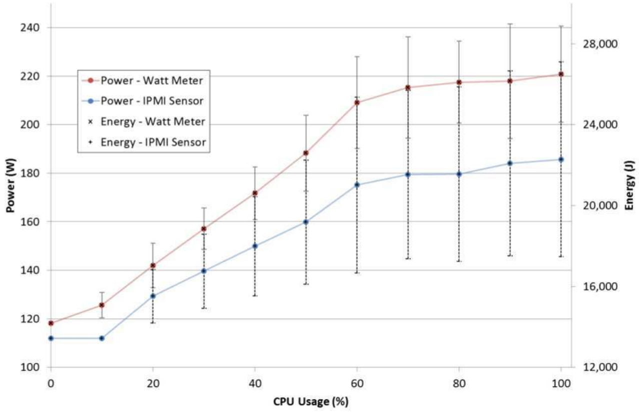

Fig. 4: CPU load vs power and energy consumption

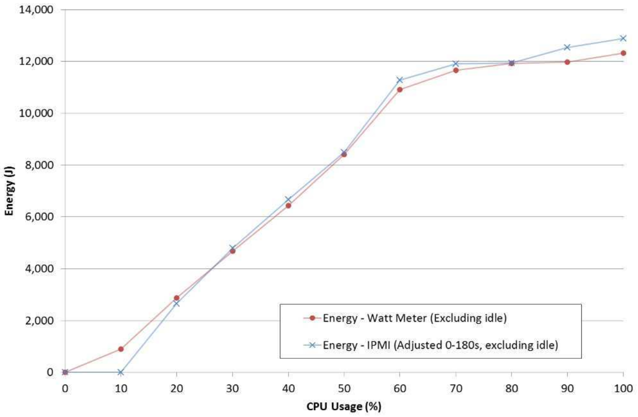

Fig. 5: CPU load vs energy consumption adjusting to compensate for idle host power consumption

We can however see that the lines although offset follow the same trend.

Figure 5 shows the effect of making two adjustments, that means calibration data obtained by IPMI more closely matches the data obtained from the Watt meter. Firstly we remove the idle power consumption of the server thus we only consider the additional energy consumption of the application and secondly we increase the window size for the IPMI measurements from 120 seconds to 180 seconds. This takes account of the entire averaging window used by IPMI which is fixed at 60 seconds. After these changes we can see that the two lines nearly directly correlate, with the Watt meter and the IPMI sensor closely agreeing in the range 20-80 % CPU utilisation but with slightly more error at the high and low ends. The application of these two simple rules thus illustrates how IPMI can be used to produce a similar result to an actual Watt meter, albeit for the energy consumption of a physical host, VM or application.

To derive the current power consumption of an application from the model is more useful than its energy consumption alone. Figure 6 demonstrates how this can be achieved. We show a graph of calibration data for power consumption vs CPU utilisation along with confidence intervals of 95% for the

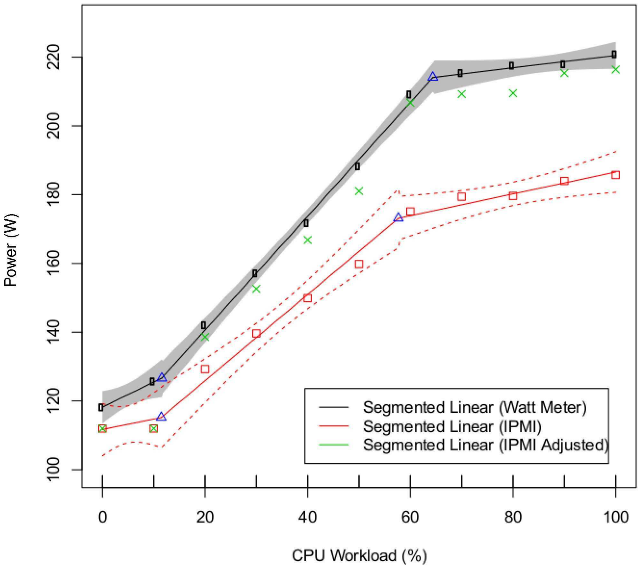

Fig. 6: CPU load vs power and energy consumption - Compensating for inaccuracies in IPMI measured values

	Multiple R^2	Adjusted R^2
Watt Meter Segmented	0.9989	0.978
Watt Meter Linear	0.9358	0.9287
IPMI Segmented	0.9946	0.9891
IPMI Linear	0.9417	0.9352
IPMI Adjusted Segmented	0.9928	0.9857
IPMI Adjusted Linear	0.9285	0.9206

TABLE I: The fit data for both linear regression and segmented linear regression

Watt meter and IPMI results. The adjusted IPMI confidence intervals are very similar and thus excluded to avoid overly filling the graph. The fit was generated in R using segmented linear regression. We additionally show IPMI gathered data after adjustments. We can see how IPMI without processing under reports the power consumption and that the correct answer is reported by the Watt meter. IPMI can be used to get a closer answer to the Watt meter by ignoring the first 60 seconds of datapoints. This works as the averaging window used by IPMI will no longer reflect a period of time before the load was induced and measurements will only reflect the CPU at the load specified. Once this is done the IPMI calibration line fits much more closely to the Watt meter's line. This means in the context of calibration that the load should be induced for at least the length of the averaging window, in order to get a decent calibration. The R^2 values for the fitted lines are shown in Table I. Once this model has been constructed using the IPMI data, CPU counters can then be used in conjunction with the model generated in order to get rapid and accurate values for the power consumption.

The remaining focus of this section, is to access the validity of the changes made to the calibration data in the context of IPMI through analysing the accuracy of the power and energy predictions made from a less synthetic workload. We create a VM on the host with 32GB RAM and 32 virtual cores. This gives the VM the possibility of using all physical cores of the host machine. We then use the Phoronix testsuite [25] as a means of inducing a workload. The benchmarking suite then runs for an hour inducing load on the system, with the resultant trace shown in Figure 7. Figure 7 shows the use of the Watt meter emulator with the results from three different calibration datasets. These datasets having been gathered via a Watt meter, by IPMI and via IPMI with the same adjustments as used in Figure 6. It clearly shows how the estimated power consumption for the adjusted IPMI more closely matches the Watt meter generated calibration data's trace. The average error and absolute average error for this trace is shown in Table II, for Watt Meter calibrated (WM), IPMI, and adjusted IPMI (IPMI-adj).

	WM	IPMI	IPMI-adj
Average error (W)	-0.20	-18.35	-6.50
Average absolute error (W)	15.68	21.88	18.92
Average error/idle power	-0.17%	-15.75%	-5.58%
Absolute average error/idle power	13.46%	18.78%	16.24%

TABLE II: Error between Watt meter reading and the model generated estimate of power consumption

We can see in terms of estimating energy consumption of an application the adjustments made to IPMI have made a substantial improvement to the average error (11.85W or 10.17%). Thus over time the estimation of energy consumption will be far more accurate. In considering the absolute error it can be seen while the model used to estimate the actual power consumption has errors a reduction in the error from IPMI alone is also realised (2.96W or 2.54%). This demonstrates how a single power value may have inaccuracies but for the overall energy consumption it will eventually converge to the real value in the context of this workload. The difference in error between IPMI and the Watt meter remains, principally as a result of the lack of resolution of the IPMI based power sensors, having eliminated averaging issues during the calibration run. This can only be resolved by hardware vendor based improvements of these power sensors. Until this improvement is realized, this leaves our models and careful calibration as the only solution for gaining reliable estimates of current power consumption. Models such as ours will retain their usefulness for the prediction of future power consumption of a given workload.

Finally we illustrate the error's associated with the trace as shown in Figure 8. The deviation from the actual power consumption for each estimated power value is calculated and shown on the x axis, while on the y axis the count of how many estimates with that error are shown. Therefore the more estimates that are close to zero Watts of error the better the prediction of power consumption and the more symmetrical a distribution is for error the more accurate the prediction for energy consumption will be.

Figure 8 shows how the IPMI based calibration data biggest peak has a slight offset from 0 underestimating the power consumed. The adjusted IPMI makes an improvement on this

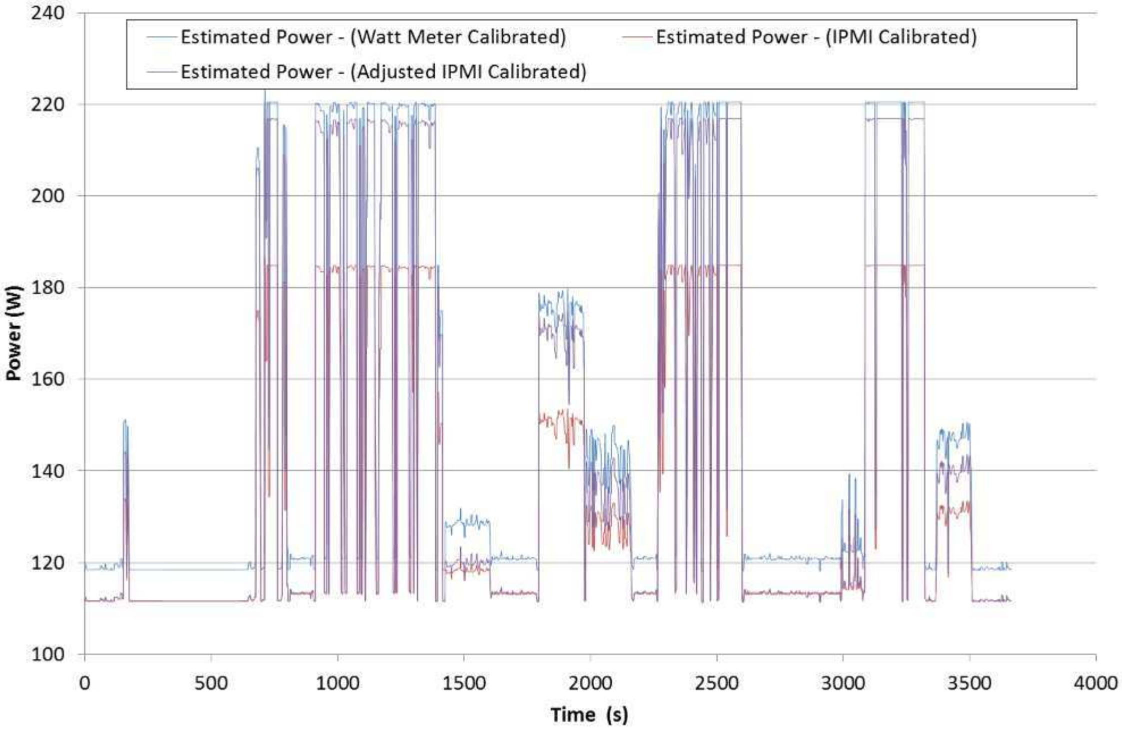

Fig. 7: Trace of a workload induced by the Phoronix testsuite (CPU)

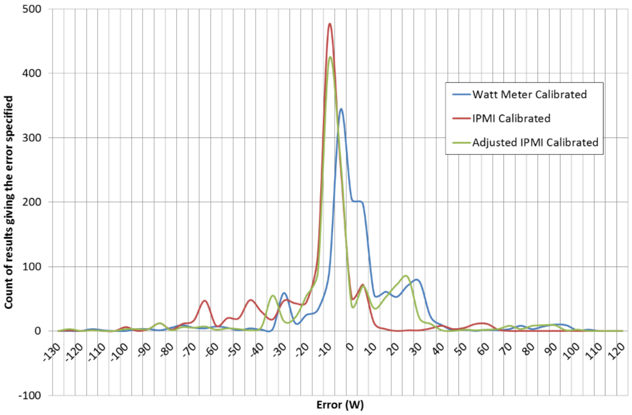

Fig. 8: A distribution of errors in the model's accuracy as compared to the power meter reading

with a peak centred closer to 0W of error. The Watt meter based calibration performed best in that its average error was -0.20W. Aside from observing the proximity to the ideal of centring around zero Watts of error, we can see other errors shown. These tend to result from transition periods between distinctly different levels of CPU load and timing issues between the different types of metric values been gathered given that measurements were stored in a real distributed system.

VI. Conclusion and Future Work

In conclusion we have shown how IPMI although relatively inaccurate can be used in various specialised scenarios. These include showing the energy consumption due to additional load of an application if datapoints after the load has ended are taken into account, due to the effect of an averaging window used by the IPMI device. IPMI can be further used as part of calibration of a host's power model if calibration runs with a continuous load take longer than the averaging window, with the initial datapoints been discounted. This gives rise to the possibility of calibrating power models for large data centers, even though the IPMI measurement equipment has not achieved a high level of accuracy. These power models thus serve two purposes, the first is that they can be used to predict future power consumption, by estimating workload. The second is that for the time being they can be used to make more rapid estimates of power than the readily available measurement equipment allows, given the access to the faster more accurate measurements of CPU load. In the future we intend to refine the accuracy of our calibration in an automated fashion by performing a search for the CPU load that causes a transition in the IPMI's reported power consumption. In addition our main focus will be on workload prediction that will allow predictions of applications power consumption to be used in areas such as VM scheduling and billing and SLAs based upon energy consumption. Additionally, we plan to explore automating the selection of the regression model used within the calibration process to better fit the characteristics of machines that do not exhibit a linear trend. Finally, we will introduce further metrics to increase the accuracy of our model, so that it no longer focuses solely upon the CPU thus better accounting for the type of workload being executed.

Acknowledgment

This work has received support through the EU 7th Framework Programme for research, technological development and demonstration under Grant Agreement no 610874 as part of the ASCETiC project.

References

[1] C. Wilke, S. Götz, and S. Richly, "JouleUnit: A Generic Framework for Software Energy Profiling and Testing," in *Proceedings of the 2013 Workshop on Green in/by Software Engineering*, ser. GIBSE '13. New York, NY, USA: ACM, 2013, pp. 9–14.

[2] ZABBIX SIA, "Homepage of Zabbix:: An Enterprise-Class Open Source Distributed Monitoring Solution," 2014. [Online]. Available: http://www.zabbix.com/

[3] K. Fatema, V. C. Emeakaroha, P. D. Healy, J. P. Morrison, and T. Lynn, "A survey of Cloud monitoring tools: Taxonomy, capabilities and objectives," *Journal of Parallel and Distributed Computing*, vol. 74, no. 10, pp. 2918–2933, 2014.

[4] F. Rossigneux, L. Lefevre, J.-P. Gelas, and M. Dias De Assuncao, "A Generic and Extensible Framework for Monitoring Energy Consumption of OpenStack Clouds," in *The 4th IEEE International Conference on Sustainable Computing and Communications (Sustaincom 2014)*, Sydney, Australia, Dec. 2014.

[5] GEMBIRD Deutschland GmbH, "EGM-PWM-LAN data sheet," http://gmb.nl/Repository/6736/EGM-PWM-LAN_manual—7f3db9f9-65f1-4508-a986-90915709e544.pdf, 2013.

[6] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya, "Virtual Machine Power Metering and Provisioning," in *Proceedings of the 1st ACM Symposium on Cloud Computing*, ser. SoCC '10. New York, NY, USA: ACM, 2010, pp. 39–50.

[7] G. G. Castañé, A. Núñez, P. Llopis, and J. Carretero, "E-mc2: A formal framework for energy modelling in cloud computing," *Simulation Modelling Practice and Theory*, vol. 39, no. 0, pp. 56–75, 2013.

[8] A. E. H. Bohra and V. Chaudhary, "VMeter: Power modelling for virtualized clouds," in *Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*, Apr. 2010, pp. 1–8.

[9] M. Hähnel, B. Döbel, M. Völp, and H. Härtig, "Measuring Energy Consumption for Short Code Paths Using RAPL," *SIGMETRICS Perform. Eval. Rev.*, vol. 40, no. 3, pp. 13–17, Jan. 2012.

[10] H. Yang, Q. Zhao, Z. Luan, and D. Qian, "iMeter: An integrated {VM} power model based on performance profiling," *Future Generation Computer Systems*, vol. 36, no. 0, pp. 267–286, 2014.

[11] J. W. Smith, A. Khajeh-Hosseini, J. S. Ward, and I. Sommerville, "CloudMonitor: Profiling Power Usage," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, Jun. 2012, pp. 947–948.

[12] F. Farahnakian, P. Liljeberg, and J. Plosila, "LiRCUP: Linear Regression Based CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Centers," in *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on*, Sep. 2013, pp. 357–364.

[13] S. Schubert, D. Kostic, W. Zwaenepoel, and K. G. Shin, "Profiling Software for Energy Consumption," pp. 515–522, 2012.

[14] Z. Jiang, C. Lu, and Y. Cai, "VPower: Metering power consumption of VM," *2013 IEEE 4th International Conference on Software Engineering and Service Science*, pp. 483–486, May 2013.

[15] A. Khan, X. Yan, S. Tao, and N. Anerousis, "Workload characterization and prediction in the cloud: A multiple time series approach," pp. 1287–1294, 2012.

[16] N. Roy, A. Dubey, and A. Gokhale, "Efficient Autoscaling in the Cloud Using Predictive Models for Workload Forecasting," pp. 500–507, 2011.

[17] S. Islam, J. Keung, K. Lee, and A. Liu, "Empirical prediction models for adaptive resource provisioning in the cloud," *Future Generation Computer Systems*, vol. 28, no. 1, pp. 155–162, Jan. 2012.

[18] A. Quiroz, H. Kim, M. Parashar, N. Gnanasambandam, and N. Sharma, "Towards autonomic workload provisioning for enterprise Grids and clouds," pp. 50–57, 2009.

[19] J. J. Prevost, K. Nagothu, M. Jamshidi, and B. Kelley, "Optimal calculation overhead for energy efficient cloud workload prediction," pp. 741–747, 2014.

[20] K. Djemame, D. Armstrong, R. E. Kavanagh, A. J. Ferrer, D. G. Perez, D. R. Antona, J.-C. Deprez, C. Ponsard, D. Ortiz, M. Macías, J. Guitart, F. Lordan, J. Ejarque, R. Sirvent, R. M. Badia, M. Kammer, O. Kao, E. Agiatzidou, A. Dimakis, C. Courcoubetis, and L. Blasi, "Energy efficiency embedded service lifecycle: Towards an energy efficient cloud computing architecture," in *Proceedings of the Energy Efficient Systems (EES'2014) Workshop, 2nd International Conference on ICT for Sustainability 2014*, vol. 1203. Stockholm, Sweden: CEUR Workshop Proceedings, Aug 2014, p. 1–6, http://ceur-ws.org/Vol-1203/EES-paper1.pdf.

[21] R. Kavanagh, D. Armstrong, and K. Djemame, "Towards an Energy-Aware Cloud Architecture for Smart Grids," in *12th International Conference on Economics of Grids, Clouds, Systems, and Services*, Cluj-Napoca, Romania, 2015, pp. 1–14.

[22] Electronic Education Devices, "WattsUp? Watt Meters." [Online]. Available: https://www.wattsupmeters.com/secure/index.php

[23] OpenNebula Systems, "Open Nebula: Flexible Enterprise Cloud Made Simple," 2016. [Online]. Available: http://opennebula.org/

[24] A. Waterland, "Stress Project," 2014. [Online]. Available: http://people.seas.harvard.edu/{∼}apw/stress/

[25] Phoronix Media, "Phoronix Test Suite - Linux Testing & Benchmarking Platform, Automated Testing, Open-Source Benchmarking," 2016. [Online]. Available: http://www.phoronix-test-suite.com/