Containerization's Power Use Overhead in Video Streaming

Etienne-Victor Depasquale
Department of Communications and Computer Engineering
University of Malta
Msida, Malta
e-mail: edepa@ieee.org

Saviour Zammit
Department of Communications and Computer Engineering
University of Malta
Msida, Malta
e-mail: saviour.zammit@um.edu.mt

Abstract— **Containerization of a service enables live migration and, thereby, consolidation of running service instances onto as few host platforms as possible. However, containerization's operational overhead must be investigated to determine overall viability. One dimension of this overhead is that of power use, and this is investigated here. An architecture for a video cache service at the edge of a Communications Service Provider's (CSP) network in the metropolitan area is designed, and a scaled version is implemented in a laboratory environment. A comparison is made between power used while streaming videos in both native and containerized modes of operation. Containerization is found to incur a low power overhead while streaming video, compared with streaming video from ffmpeg running directly on the host operating system. Power use is measured using hardware instrumentation and with PowerTOP, a software power meter. Limits on the latter's accuracy have been observed.**

Keywords - containers; power; video; streaming; implementation model.

I. INTRODUCTION

Content Delivery Networks (CDNs) are overlay networks that are key to controlling the growth in demand for bandwidth in long-haul communications links. By distributing content to caches in geographical regions of the world where customers are located, the number of times which a single item of content crosses long-haul links between the content origin's region and the customer's region, is reduced to just one. In turn, the content is distributed several times to customers in the region. While the function of the CDN, from the customer's perspective, is that of reducing latency and avoiding buffer underrun, the control of bandwidth growth is a function that has a strategic role in the stability of world-wide communication. The CDN's role in bandwidth control continues to gain attention [1]; a variety of CDN implementations has been investigated [2][3] and surveyed [4][5] and generalized surveys are of ongoing interest [6][7]. The importance of the CDN seems to grant sufficient ground for study of the impact of its point of presence (PoP) on the information and communication technology of its environs.

This study seeks to compare power use in containerized deployment of the media server in a CDN PoP. It focuses on the power use of the media server as it processes a representative set of tasks. The media selected for study is video (henceforth, the media server will be referred to as the video server), and two reasons support this choice. Video dominates traffic, whether in the access, aggregation, metro-core, or long-haul. Moreover, some of the tasks, such as transcoding, are processor-intensive and serve to indicate the power capacity required to support CDN PoPs.

The rest of this paper is structured as follows. In Section II, the objective is stated. In Section III, the implementation model is presented. This supports reproduction of the test environment. In Section IV, the method is elaborated upon. Section V presents the results and Section VI supports interpretation through analysis of these results. Section VII draws a succinct conclusion on the impact that containerization of a video service has on power use overhead.

II. OBJECTIVE

An overhead is expected in the containerized implementation, and its ***quantification*** is sought. The objective can be articulated in terms of a comparison between two types of deployment:

- power use in a computer system that runs the service within containers, with
- power use in a computer system that runs the service directly on the operating system.

Quantification is sought in order to control a tradeoff between native and containerized deployment. The tradeoff may be succinctly summarized as one of greater operating power per unit (physical host) versus potential for lower number of operating units (physical hosts). The following sections elaborate on this summary.

III. IMPLEMENTATION MODEL

An edge cache of a video streaming service is deployed. A high-level view of the implementational model is shown in Figures 1 and 2.

- Figure 1 shows an implementation that is easily portable to a cloud-native infrastructure (henceforth referred to as the cloud-native implementation), and
- Figure 2 shows an implementation that is a hybrid of physical (the video server) and virtual network functions (the switch).

The cloud-native implementation uses containers to host the video server. Both implementations host a virtual layer 2 switch in the intermediate node.

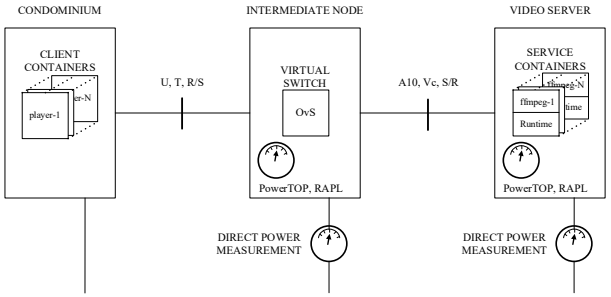

Figure 1. Physical topology of the video streaming service, deployed in containers. Video Server located in local exchange or Access Node (AN); Intermediate Note located in street cabinet (subtended AN [8]).

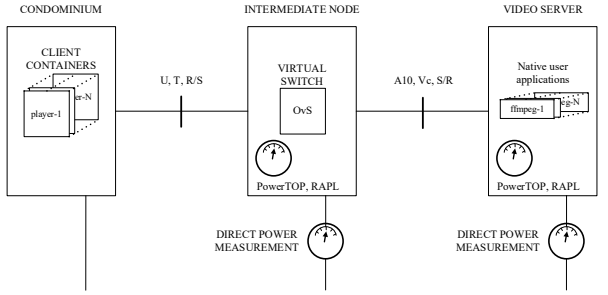

Figure 2. Physical topology of the video streaming service, deployed on a host operating system.

A. Hardware

The hardware used in this testbed consists of a set of three HPE (Hewlett Packard Enterprise) ProLiant BL460c Gen9 blade servers [9], hosted in an HPE c7000 blade enclosure. Connectivity between server and client blades is obtained through pass-through interconnect bay modules, patched with single-mode optic fibre cables. These latter modules support the goal of bypassing c7000 ecosystem interconnect-bay physical networking devices. Bypass is necessary to introduce separate, **virtual** switching hardware. The virtual switch is implemented on a third HPE Gen9 blade server. The links to the switch are of type 10GBASE-SR. The video server has a single Intel® Xeon® CPU E5-2640 v3 (2.60GHz) processor package. Dynamic Voltage and Frequency Scaling (DVFS) is under system firmware control.

B. Software

The software consists of:

- an FFmpeg [10] video server. This is representative of the access node at the edge of the metro-core network;
- a TSDuck [11] receiver. This is representative of end-user's video player, and is also used to measure received bitrate to ensure that Quality of Service (QoS) (see Section IV-C) is respected;
- the virtual switch software is Open vSwitch [12].

A minimalist operating system was selected for the video server, to support isolation and attribution in power measurements. While minimalist operating systems do not necessarily correlate with minimal noise in power measurement, it seems useful to reduce the number of possible sources from the outset. For this reason, Alpine Linux [13] Standard distribution version 3.19 was chosen.

The container system software selected is Docker [14]. Docker is a mature containerization platform and it is modular: the runtime daemon (containerd) supports other user interfaces apart from the Docker user interface (dockerd). For example, Kubernetes [15] can be used to manage containers created through the Docker Command-Line Interface (CLI).

IV. METHOD

A. Instrumentation

Near-real time measurement of power use can be obtained from two sources of instrumentation. The blade servers are equipped with a management processor (known as "integrated lights-out", or iLO) that logs a power measurement every 10 seconds and stores a 20-minute history that can be read through a Redfish®[16] - compliant RESTful (Representational State Transfer) Application Programming Interface (API). Selectivity in aggregate power use measurement is afforded by blade systems, since these separate power supply to the (blade) computer system from power supply to two major overhead power drains. Blade servers use blade chassis services for power supply (where ac – dc conversion losses occur) and cooling (where blowers use power as they ventilate from chassis front to chassis rear). Thus, measurement of power used by the blade server at the supply voltage rails is free of the problematic, variable contribution from overheads, and idle power can be measured to the accuracy afforded by these blade system power measurement instruments. The measurement datum is of integer type, obtained by truncation of the decimal part of the actual measurement. Moreover: since the iLO is not part of the System Under Test (SUT), it does not alter power measurement.

While the iLO provides an aggregate power measurement, process- and thread- level granularity is obtained through software power meters. Hardware extensions for power measurement are available in processor models that support the HoweverIntel Running Average Power Limit (RAPL) feature. PowerTOP [17] is software that enables this level of power attribution, and it is indeed capable of exploiting RAPL. This tool complements the aggregate power measurement obtained by blade sensor instrumentation. PowerTOP uses a top-down approach [18], (it divides the power measurement over a period amongst processes and threads in proportion to their core utilization) and precedes the measurement period by one of calibration (the utility was run in calibration mode for several hours before starting the first experiment) in which it obtains weighting parameters for the attribution process. Calibration is further refined with use, and PowerTOP saves its parametric refinement to persistent storage for future exploitation [19]. PowerTOP was used in its logging mode of operation, with 10 (ten) – second

averaging intervals. However, PowerTOP has several significant limitations, as follows: it only measures dynamic power, it does not capture all power use, and it increases the SUT's aggregate power use. These must be mitigated.

B. Baselining

It is necessary to distinguish power used by the video service from power used by other consumers. This requires measurement of static (/idle/leakage) power use. It is also necessary to distinguish between dynamic power used during video service operation time, from dynamic power used when the service is idle. In essence: service power use can be thought of as an amount added above that used by the operating system and system software, which in turn is added above that used to operate electronic components (leakage/static/idle) power. Hence, it is possible to perceive a baseline to which service power is added to obtain the total power. Formally:

$$P_{b_1}^{(video)} = P_{idle}^{f_1} + P_q^{(os)}$$

where $P_q^{(os)}$ is the **dynamic** power corresponding to the Operating System's (OS) operation **without** container system software and without running User Applications (UAs), and $P_{idle}^{f_1}$ is the **idle/leakage/static** power at the frequency f_1 at which the OS is quiescent.

A second baseline, $P_{b_2}^{(video)}$, is required to ensure experimental reproducibility.

$$P_{b_2}^{(video)} = P_{idle}^{f_2} + P_q^{(os+dockerd+containerd)}$$

Here, $P_q^{(os+dockerd+containerd)}$ is the **dynamic** power corresponding to the OS's operation **with** container system software but without running User Applications (UAs), and $P_{idle}^{f_2}$ is the **idle/leakage/static** power at the frequency f_2 at which the Operating System (OS) is quiescent. The state of quiescence is defined below (see IV-D-2).

C. Quality of Service

QoS is considered to be satisfied as long as there is sufficient capacity in the links to keep the overall average received bitrate of every video stream at or above the video file's overall bitrate.

D. Experiments

1) Test conditions

Video service will be delivered from both containerized and native deployments. The test conditions pertinent to the video server will be the following.

1. Implementation
   a. During containerized operation, each video service process and the libraries on which it depends will be operated from a container. One service process serves one client.
   b. During native operation, a new instance of the video service process will be started for every new client.
2. Load unit: This will consist of the work required to process a workflow based upon a video with the following technical specifications:
   a. Overall bitrate = 457 kb/s, = video bitrate of 326 kb/s + audio bitrate of 127 kb/s + mp4 container metadata rate (overhead)
   b. Duration = 1h 32m 2.19s (5522.19 s), of which 30 minutes are played, starting at a randomly-selected point in the video.
   c. H.264 video codec, Main profile
      i. Resolution = 1280 x 720
      ii. Frame rate ≈ 23.98 frames/second (fps)
   d. Advanced Audio Coding (AAC) audio codec, Low Complexity profile
      i. Sampling rate = 44.1 kHz
   e. Client supports same video and audio codec; hence server does not need real-time transcoding.

2) Procedure

The power used by the video server is measured at progressively higher load levels. Two sets of experiments are carried out: the first set uses containerized video server instances and the second set uses native video server instances. A containerized service instance consists of a container carrying ffmpeg. A single container is created to deliver a single stream and is destroyed immediately thereafter. When the container is created, ffmpeg is executed and listens on a TCP port, through which it streams 30 minutes of video. A native service instance is a single instance of the ffmpeg process; it follows the same lifecycle as the containerized instance.

Management of operations is not trivial, even at the minimum load level, as it involves the following steps:

1. Reboot the video server, to obtain a common and reproducible initial state.
2. Wait until the video server quiesces. This is the time required for server power use to fall to the state where the iLO measurement persistently shows baseline 2 usage. Persistence was empirically found to be ascertained 20 minutes after rebooting.
3. Start the power meters for both total and dynamic power, for both the video server and the virtual switch.
4. Wait for a fifteen-minute interval, to capture behaviour before video streaming.
5. Instantiate and start a container carrying the ffmpeg listener, poised for real-time playback with randomized starting point and 30-minute play time.
6. Start a TSDuck client to connect to the container and measure the bitrate, averaged over 5-second intervals.
7. Once 30 minutes of video have been played, destroy the container.
8. Wait for a fifteen-minute interval, to capture behaviour after video streaming.

For several concurrent streams, steps 5 and 6 must be repeated for each one of the additional streams. For the native service instance, step 5 involves the ffmpeg process only and there is no equivalent to step 7.

It seems evident that manual management is highly prone to error and is therefore unsuitable. Automated management using Python scripts and Ansible [20] is employed to handle

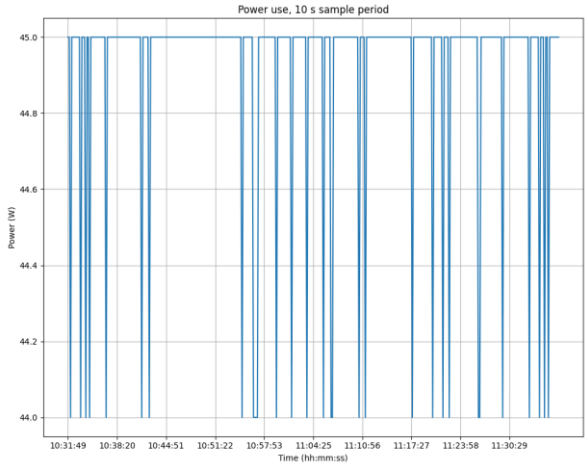

Figure 3. Power used by the video server, with a quiescent OS.

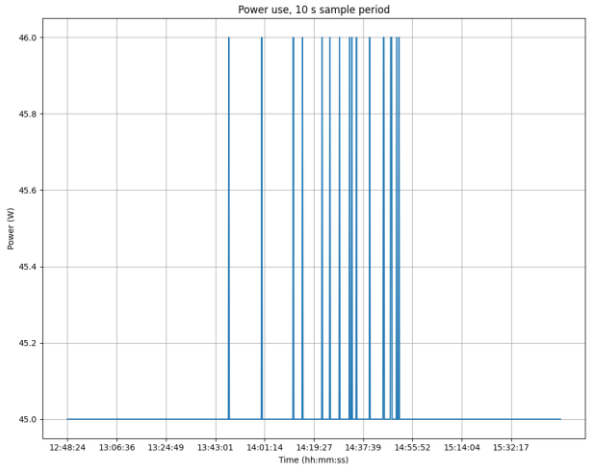

Figure 4. Baseline 2 video server aggregate power.

the **orchestration** of the various roles: power meters, container runtime managers and video clients. This enables the experiment to be scaled out to levels that are well beyond the physical limitations of a single human operator.

V. RESULTS

Denote:

- mean dynamic power measured by PowerTOP by $\overline{p_{dyn}^{(ptop)}}$
- mean total power measured by the iLO during a time period T_x by $\overline{p^{(iLO)}}([T_x])$.

A. Video server's baseline 1

Figure 3 shows the power used by the video server over an hour period of measurement, post-onset of quiescence. Since the iLO truncates decimals in $[n, n+1)$ to n, then the computation of the mean will count the incidences of 45 W and 44 W, and use them as weights to compute a lower limit to the range of values which the average can take. An upper limit is obtained by adding the maximum possible error (equal to 1W) and the mean of the possible range obtained by adding the mean error (0.5W) to the lower limit of the range. Using this premise, the mean power measured by the iLO, under the condition of a quiescent operating system (see Figure 3) is as follows:

$$P_{idle}^{f_1} + P_q^{(os)} = \overline{p^{(iLO)}}([10{:}31{:}49, 11{:}37{:}01]) = 45.4198\ \text{W} \cong 45.4\ \text{W}$$

B. Mitigation of PowerTOP's limitations

PowerTOP captures neither static nor dynamic power used by Hard Disk Drives (HDDs) and Solid-State Disks (SSDs); this was observed and confirmed through discussion with PowerTOP's developers [21]. Indeed, our experiments under baseline 1 conditions show that if PowerTOP is operated in logging mode with HDD as destination, iLO aggregate power use is more than 0.5 W greater on the SUT than the figure obtained while logging to a RAM disk. While logging to RAM disk (under baseline 1 conditions), average aggregate power use increases to 45.5W, compared with 45.4W (see Section V-A, above) when measurements are taken solely through use of the iLO's instrumentation.

C. Video server's baseline 2

The difference in average dynamic power is added to baseline 1, to obtain baseline 2:

$$\Delta\overline{p_{dyn}^{(ptop)}} = \overline{p_{dyn}^{(ptop)}}(baseline\_2) - \overline{p_{dyn}^{(ptop)}}(baseline\_1) = 0.7727 - 0.1851 = 0.5876\ \text{W}$$

$$\therefore P_{b_2}^{(video)} = P_{idle}^{f_2} + P_q^{(os+dockerd+containerd)} = P_{idle}^{f_1} + P_q^{(os)} + \Delta\overline{p_{dyn}^{(ptop)}} = 45.4 + 0.5876 \cong 45.99\ \text{W}$$

This is consistent with the graphical summarization of iLO measurements shown in Figure 4. This baseline, notably the graph of power against time, is essential in obtaining a reproducible starting state for all video service operation experiments.

D. Orchestration of containerized streaming

Results from running experiments on 1, 2, 5, 10, 20, 40 and 80 instances are presented. The result items consist of:

1. Mean aggregate power use (iLO instrumentation). Due to the integer type of the measurement, actual average iLO power use can lie in the range of ± 0.5 W of the reported result.
2. Mean dynamic power use (PowerTOP instrumentation). Dynamic power data is added to baseline 1 and the sum is plotted on the same Cartesian axes as the total power data.

PowerTOP was used to attribute dynamic power to processes, and these were sorted in descending order. Graphical representations of the power used were produced too. These results are presented in the Github online repository at [22]. Measurements of received stream bitrates are also available in this repository.

1) Single instance

Table I shows the mean power use; Figure 5 shows PowerTOP's measurements offset by baseline 1 and laid over the iLO's measurements. Time is shown in the format hh:mm:ss, where hh, mm and ss stand for hour-of-day, minutes in the hour and seconds in the minute, respectively. The larger post-operation (post-op) average power is due to activity undertaken by an instance of containerd (the container runtime) after the container is destroyed (post-ops). However, well after operations end, the iLO's measurements return to the baseline 2 profile. Pre-operations (pre-ops), both meters (iLO and PowerTOP) are in good agreement (PowerTOP's measurements would all be rounded down to 45W). Moreover, the average power used during operations as estimated by PowerTOP is 46.99 W (baseline_1, = 45.4, + 1.5940), whereas the iLO estimates 47.03W. The ten-second averages' dissimilarity increases during and post-operations but is still good. Notably, the spike in power use at the beginning and end of operations is captured by both meters, albeit not being measurements of the same magnitude.

2) Two instances

Table II and Figure 6 show the results pertinent to two containerized video server instances. As is the case with the single instance, for pre-ops and post-ops, both meters are in good agreement (the spike at about 09:29:00 is probably due to HDD input/output operations while loading PowerTOP). During operations, the average total power estimated by PowerTOP is 48.02 W (baseline_1 + 2.6162), whereas the iLO estimates 47.06W. The discrepancy is an overestimate by about 1W.

An interpretation of the discrepancy between operating period averages is visible in the graph (Figure 6) showing real time measurements. When the iLO measures 46W, the actual value is in the range [46,47), and the rate of change between 46W and 47W is much larger than the single-instance case.

TABLE I. MEAN POWER USE – SINGLE SERVICE INSTANCE

Power type	Description	Avg[a] (W)
$\overline{p^{(iLO)}}[14:47:05,15:03:00]$	Before starting the service instance	45.65
$\overline{p^{(iLO)}}[15:03:00,15:33:05]$	During the service instance's operation	47.03
$\overline{p^{(iLO)}}[15:33:05,15:52:17]$	After the service instance ended	46.17
$\overline{p_{dyn}^{(ptop)}}[14:48:17,15:03:00]$	Mean dynamic power before service instance operation	0.8593
$\overline{p_{dyn}^{(ptop)}}[15:03:00,15:33:05]$	Mean dynamic power during service instance operation	1.5940

a. Average.

PowerTOP's real time measurements are consistently higher than 47W, revealing that several of the 10-second measurement intervals are in certain disagreement, albeit small (< 2/46, i.e., < 5%).

3) Five, ten, twenty, forty and eighty instances

The results for five (Table III, Figure 7), ten (Table IV, Figure 8), twenty (Table V, Figure 9), forty (Table VI, Figure 10) and eighty instances (Table VII, Figure 11) are shown below. Conditions pre-operations are similar, but PowerTOP's average error estimation increases as power use increases. The numbers shown in the list are PowerTOP's estimate vs iLO's maximum estimate, for N instances (Ni):

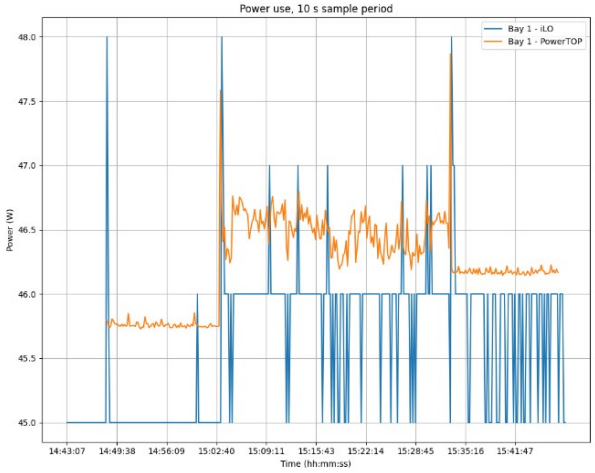

Figure 5. One instance. Video server's power use during containerized service operation. Baseline 1 added to powertop measurements.

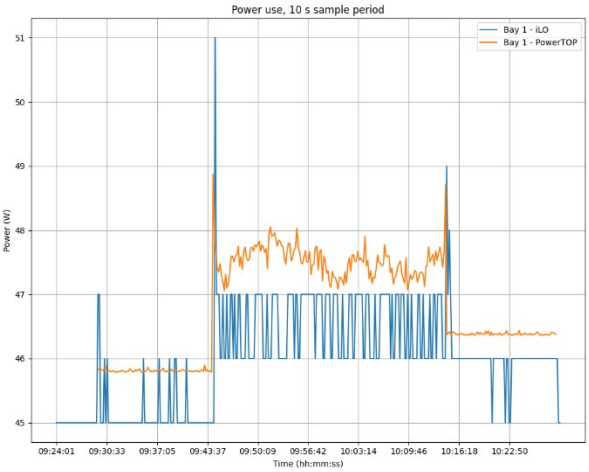

Figure 6. Two instances. Video server's power use during containerized service operation. Baseline 1 added to powertop measurements.

TABLE II. MEAN POWER USE – TWO SERVICE INSTANCES

Power type	Description	Avg (W)
$\overline{p^{(iLO)}}[09:24:01,09:44:27]$	Before starting the service instance	45.60
$\overline{p^{(iLO)}}[09:44:27,10:14:37]$	During the service instance's operation	47.06
$\overline{p^{(iLO)}}[10:14:37,10:29:23]$	After the service instance ended	46.12
$\overline{p_{dyn}^{(ptop)}}[09:29:27,09:44:27]$	Mean dynamic power before the service instances' operation	0.9693
$\overline{p_{dyn}^{(ptop)}}[09:44:27,10:14:37]$	Mean dynamic power during the service instances' operation	2.6162

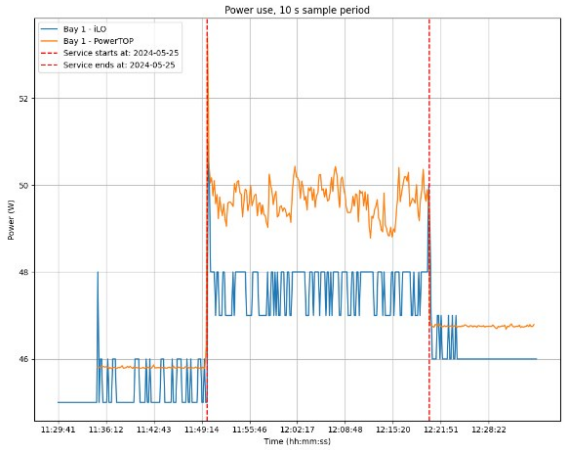

Figure 7. Five instances, containerized operations, baseline 1.

- 5i: 50.14 vs 48.66W
- 10i: 54.38 vs 50.10W
- 20i: 60.77 vs 51.74W
- 40i: 64.59 vs 53.90W
- 80i: 60.92 vs 56.80W

Inspection of the online supplementary data on process – level power attribution suggests that PowerTOP overestimates across all processes on our test platform.

TABLE III. MEAN POWER USE – FIVE SERVICE INSTANCES

Power type	Description	Avg, (W)
$\overline{p^{(iLO)}}[11:29:41,11:49:59]$	Before starting the service instance	45.79
$\overline{p^{(iLO)}}[11:49:59,12:20:15]$	During the service instance's operation	48.16
$\overline{p_{dyn}^{(ptop)}}[11:35:00,11:49:59]$	Mean dynamic power before service instances' operation	0.9970
$\overline{p_{dyn}^{(ptop)}}[11:49:59,12:20:15]$	Mean dynamic power during the service instances' operation	4.7421

TABLE IV. MEAN POWER USE – TEN SERVICE INSTANCES

Power type	Description	Avg. (W)
$\overline{p^{(iLO)}}[15:06:49,15:27:03]$	Before starting the service instance	45.60
$\overline{p^{(iLO)}}[15:27:03,15:57:27]$	During the service instance's operation	49.60
$\overline{p_{dyn}^{(ptop)}}[15:12:03,15:27:03]$	Mean dynamic power before service instances' operation	0.8759
$\overline{p_{dyn}^{(ptop)}}[15:27:03,15:57:27]$	Mean dynamic power during the service instances' operation	8.9781

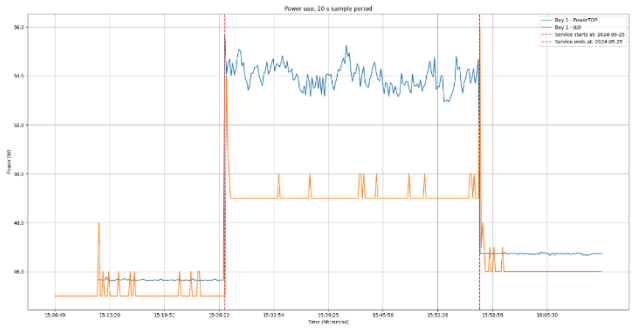

Figure 8. Ten instances, containerized operations, baseline 1.

TABLE V. MEAN POWER USE – TWENTY SERVICE INSTANCES

Power type	Description	Avg. (W)
$\overline{p^{(iLO)}}[17:56:58,18:17:08]$	Before starting the service instance	45.76
$\overline{p^{(iLO)}}[18:17:08,18:48:00]$	During the service instance's operation	51.24
$\overline{p_{dyn}^{(ptop)}}[18:02:08,18:17:08]$	Mean dynamic power before service instances' operation	0.8913
$\overline{p_{dyn}^{(ptop)}}[18:17:08,18:48:00]$	Mean dynamic power during the service instances' operation	15.3720

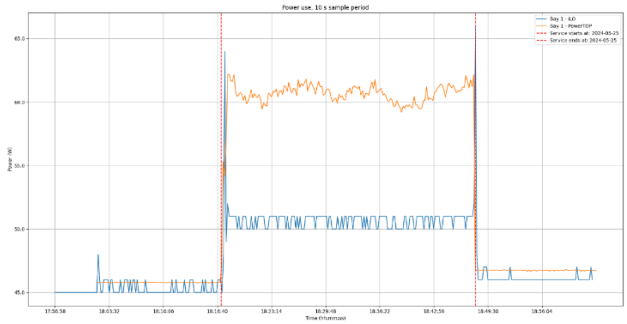

Figure 9. Twenty instances, containerized operations, baseline 1.

TABLE VI. MEAN POWER USE – FORTY SERVICE INSTANCES

Power type	Description	Avg. (W)
$\overline{p^{(iLO)}}[12:57:37,13:17:40]$	Before starting the service instance	45.56
$\overline{p^{(iLO)}}[13:17:40,13:49:15]$	During the service instance's operation	53.40
$\overline{p_{dyn}^{(ptop)}}[13:02:42,13:17:40]$	Mean dynamic power before service instances' operation	0.7206
$\overline{p_{dyn}^{(ptop)}}[13:17:40,13:49:15]$	Mean dynamic power during the service instances' operation	19.1873

E. Orchestation of native streaming

A similar set of experiments was run for native video servers. The results are available in the online repository, and are structured in the same manner as that used in Section V-D.

TABLE VII. MEAN POWER USE – EIGHTY SERVICE INSTANCES

Power type	Description	Avg. (W)
$\overline{p^{(iLO)}}[18:19:21,18:39:20]$	Before starting the service instance	45.53
$\overline{p^{(iLO)}}[18:39:30,19:13:53]$	During the service instance's operation	56.30
$\overline{p_{dyn}^{(ptop)}}[18:24:31,18:39:30]$	Mean dynamic power before service instances' operation	0.7435
$\overline{p_{dyn}^{(ptop)}}[18:39:30,19:13:53]$	Mean dynamic power during the service instances' operation	15.5243

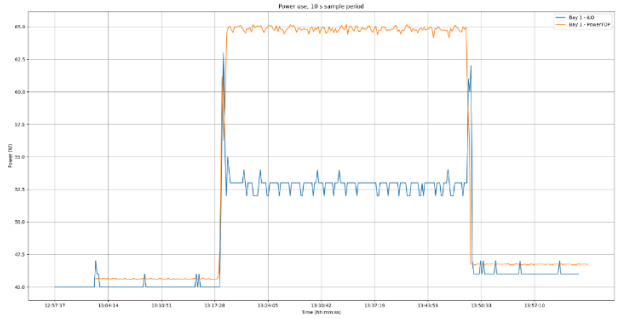

Figure 10. Forty instances, containerized operations, baseline 1.

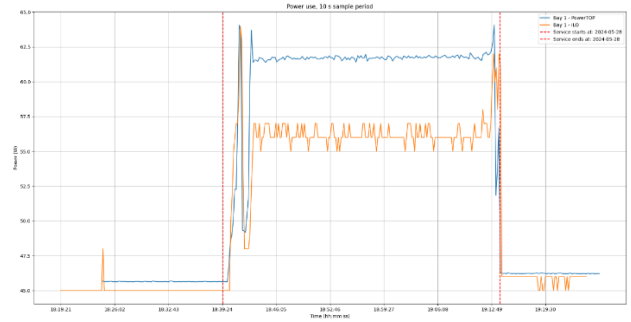

Figure 11. Eighty instances, containerized operations, baseline 1.

VI. ANALYSIS

Various characterizations of power use are considered and plotted in Figure 12. In the notation shown below, the (n) symbol indicates dependence of power used on number of streaming containers.

1. total power during operations, $P_{ops}^{iLO}(n)$, and
2. differential total power, where the difference is between operations and quiescence, $P_{ops}^{iLO}(n) - P_{q}^{iLO}$.

Figure 12 illustrates the results in graphical form. The top row of graphs compares total power and differential total power, respectively, for containerized and native operations. The bottom row shows the difference between total power and differential total power. The non-monotonic behaviour seen in the bottom row is due to the error introduced by the rounding of iLO instrumentation.

Dynamic power measurements as a function of streaming videos are not shown in Figure 12, as PowerTOP's measurements do not produce consistent, intelligible results on our platform. Estimates are insufficiently accurate. PowerTOP is capable of capturing power change behaviour (see, notably, Figure 11), but it requires further development before its estimates can be used for quantitative analysis.

VII. CONCLUSIONS

The objective set out in Section II was to quantify the overhead incurred by operating the video service containerized, instead of as an application running directly on the host operating system (native operation). An access network of the Active Ethernet type was constructed and a video cache deployed in an access node to stream videos to the access node's service area. An implementation model describing the access network was included.

The results obtained have shown that the overhead is negligible and that the benefit of running the video source in a container comes at little cost. The possibility of consolidating video streaming containers can be pursued with confidence.

No discernable cause for concern was found in the power measurement instrumentation embedded in the HPE Gen9 platform. Documentation on interfacing with the Integrated Lights-Out (iLO) server management was readily available. For detail beyond typical interest, HPE readily divulged information on this tool when contacted for help, including, for example, the method used to round the power measurement into an integer [23].

On the other hand, PowerTOP's accuracy poses a problem. The various graphs of power against time have shown that it captures changes well, but significantly overestimates them. In the light of these errors, works that have investigated containerization's overhead with the use of this tool (e.g., [24]) may need to be reviewed for the implications of inaccuracies introduced by the tool, perhaps by using external, physical power meters to calibrate PowerTOP's measurement.

Baselines have been obtained for both the video server and the virtual switch. In particular, P_{b2}^{video} has been found useful in obtaining a reproducible starting point for experiments; to a lesser extent, P_{bq}^{video} has been found useful in providing an offset for power obtained through tools that measure dynamic power. This segues well into an observation that merits particular attention. Even with 80 concurrent streams, the static power has dwarfed the dynamic power. The importance of this observation pertains to the importance of the benefit of containerization as an enabler of consolidation of physical hosts. It can readily be stated that the overhead incurred in providing the **service framework** of containerization poses no obstacle to exploration of exploitation of this benefit.

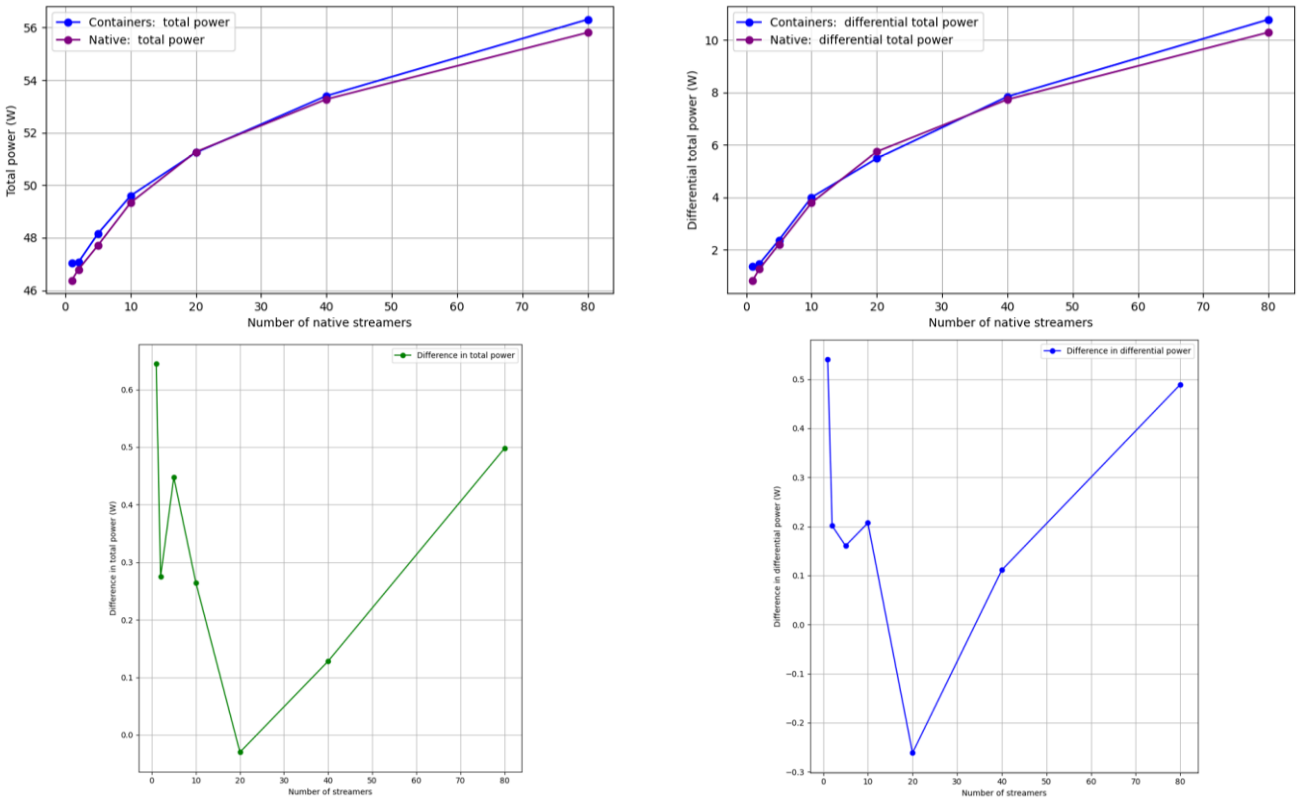

Figure 12. Comparison: native vs containerized streaming. Clockwise from top left: $P_{ops}^{iLO}(n)$, $P_{ops}^{iLO}(n) - P_{q}^{iLO}$, $P_{ops}^{iLO}(n_{cont}) - P_{ops}^{iLO}(n_{native})$ and $\left(P_{ops}^{iLO}(n_{cont}) - P_{q}^{iLO}{}_{cont}\right) - \left(P_{ops}^{iLO}(n_{native}) - P_{q}^{iLO}{}_{native}\right)$.

REFERENCES

[1] A. Teker, A. H. Örnek, and B. Canberk, "Network Bandwidth Usage Forecast in Content Delivery Networks", in *2020 International Conference on Broadband Communications for Next Generation Networks and Multimedia Applications (CoBCom)*, Jul. 2020, pp. 1–6. doi: 10.1109/CoBCom49975.2020.9174180.

[2] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "FemtoCaching: Wireless Content Delivery Through Distributed Caching Helpers", *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 8402–8413, Dec. 2013, doi: 10.1109/TIT.2013.2281606.

[3] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. M. Leung, "Cache in the air: exploiting content caching and delivery techniques for 5G systems", *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 131–139, Feb. 2014, doi: 10.1109/MCOM.2014.6736753.

[4] S. Wang *et al.*, "A Survey on Mobile Edge Networks: Convergence of Computing, Caching and Communications", *IEEE Access*, vol. 5, pp. 6757–6779, 2017, doi: 10.1109/ACCESS.2017.2685434.

[5] C. Mouradian *et al.*, "A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges", *IEEE Commun. Surv. Tutor.*, vol. 20, no. 1, pp. 416–464, 2018, doi: 10.1109/COMST.2017.2771153.

[6] J. Zhao, P. Liang, W. Liufu, and Z. Fan, "Recent Developments in Content Delivery Network: A Survey", in *Parallel Architectures, Algorithms and Programming*, H. Shen and Y. Sang, Eds., Singapore: Springer, 2020, pp. 98–106. doi: 10.1007/978-981-15-2767-8_9.

[7] B. Zolfaghari *et al.*, "Content Delivery Networks: State of the Art, Trends, and Future Roadmap", *ACM Comput. Surv.*, vol. 53, no. 2, p. 34:1-34:34, Apr. 2020, doi: 10.1145/3380613.

[8] "Multi-service Broadband Network Architecture and Nodal Requirements". TR-178 Issue 2, Broadband Forum, Sep. 2017. Accessed: Oct. 4, 2024. [Online]. Available: https://www.broadband-forum.org/technical/download/TR-178_Issue-2.pdf

[9] Hewlett Packard Enterprise, "HPE ProLiant BL460c Gen9 Server Blade", PSNow. Accessed: Jun. 17, 2024. [Online]. Available: https://www.hpe.com/psnow/doc/c04347343

[10] "FFmpeg". Accessed: Jun. 17, 2024. [Online]. Available: https://ffmpeg.org/

[11] "TSDuck". Accessed: Jun. 17, 2024. [Online]. Available: https://tsduck.io/

[12] "Open vSwitch". Accessed: Jun. 17, 2024. [Online]. Available: https://www.openvswitch.org/

[13] "index | Alpine Linux". Accessed: Jun. 17, 2024. [Online]. Available: https://alpinelinux.org/

[14] "Docker: Accelerated Container Application Development". Accessed: Jun. 17, 2024. [Online]. Available: https://www.docker.com/

[15] “Kubernetes Documentation”, Accessed: Sep. 03, 2019. [Online]. Available: https://kubernetes.io/docs/home/
[16] DMTF Redfish Forum, “Redfish Specification”. Apr. 03, 2024. Accessed: Jun. 17, 2024. [Online]. Available: https://www.dmtf.org/sites/default/files/standards/documents/DSP0266_1.20.1.pdf
[17] “Powertop – ArchWiki”. Accessed: Feb. 20, 2024. [Online]. Available: https://wiki.archlinux.org/title/powertop
[18] A. van de Ven, private communication, “PowerTOP running average interval and display update interval”, Apr. 08, 2024.
[19] A. van de Ven, private communication, “PowerTOP running average interval and display update interval (update)”, Apr. 08, 2024.
[20] Ansible Community Documentation, “User Guide”. Accessed: Jun. 17, 2024. [Online]. Available: https://docs.ansible.com/ansible/latest/user_guide/index.html
[21] A. van de Ven, private communication, “PowerTOP running average interval and display update interval (2nd update)”, Apr. 30, 2024.
[22] E.V. Depasquale, “Video Streaming Power Use,” *GitHub Repository,* Oct. 26, 2024 [Online]. Available: https://github.com/edepa/video_streaming_power_use.
[23] J. Sultana, private communication, “Power measurement - Gen9”, May 24, 2024.
[24] R. Bolla, R. Bruschi, F. Davoli, C. Lombardo, and N. S. Martinelli, “Analyzing the Power Consumption in Cloud-Native 5/6G Ecosystems”, in *2023 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2023, pp. 611–617. doi: 10.1109/ICCWorkshops57953.2023.10283755.