The PowerNap Server Architecture

DAVID MEISNER, The University of Michigan
BRIAN T. GOLD, Oracle Labs
THOMAS F. WENISCH, The University of Michigan

Data center power consumption is growing to unprecedented levels: the EPA estimates U.S. data centers will consume 100 billion kilowatt hours annually by 2011. Much of this energy is wasted in idle systems: in typical deployments, server utilization is below 30%, but idle servers still consume 60% of their peak power draw. Typical idle periods—though frequent—last seconds or less, confounding simple energy-conservation approaches.

In this article, we propose PowerNap, an energy-conservation approach where the entire system transitions rapidly between a high-performance active state and a near-zero-power idle state in response to instantaneous load. Rather than requiring fine-grained power-performance states and complex load-proportional operation from individual system components, PowerNap instead calls for minimizing idle power and transition time, which are simpler optimization goals. Based on the PowerNap concept, we develop requirements and outline mechanisms to eliminate idle power waste in enterprise blade servers. Because PowerNap operates in low-efficiency regions of current blade center power supplies, we introduce the Redundant Array for Inexpensive Load Sharing (RAILS), a power provisioning approach that provides high conversion efficiency across the entire range of PowerNap's power demands. Using utilization traces collected from enterprise-scale commercial deployments, we demonstrate that, together, PowerNap and RAILS reduce average server power consumption by 74%.

Categories and Subject Descriptors: C.5.5 [**Computer System Implementation**]: Servers

General Terms: Design, Measurement

Additional Key Words and Phrases: Power management, servers

ACM Reference Format:
Meisner, D., Gold, B. T., and Wenisch, T. F. 2011. The PowerNap server architecture. ACM Trans. Comput. Syst. 29, 1, Article 3 (February 2011), 24 pages.
DOI = 10.1145/1925109.1925112 http://doi.acm.org/10.1145/1925109.1925112

1. INTRODUCTION

Data center power consumption is undergoing alarming growth. By 2011, U.S. data centers will consume 100 billion kWh at a cost of $7.4 billion per year [U.S. EPA 2007b]. Unfortunately, much of this energy is wasted by systems that are idle. At idle, current servers still draw about 60% of peak power [Barroso and Hölzle 2007;

This article expands on D. Meisner, B. T. Gold, and T. F. Wenisch, PowerNap: Eliminating server idle power, in *Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems*.
This work was supported by an equipment grant from Intel, grants from Google, and NSF grant CCF-0811320.
Authors' addresses: D. Meisner and T. F. Wenisch, Computer Science and Engineering Department, University of Michigan, 2260 Hayward Street, Ann Arbor, MI 48109; email: {meisner, twenisch}@umich.edu; B. T. Gold, Oracle Labs, 16 Network Circle, Menlo Park, CA 94025; email: brian.gold@oracle.com.

DOI 10.1145/1925109.1925112 http://doi.acm.org/10.1145/1925109.1925112

Fan et al. 2007; Lefurgy et al. 2007]. In typical data centers, average utilization is only 20–30% [Barroso and Hölzle 2007; Bohrer et al. 2002]. Low utilization is endemic to data center operation: strict service-level agreements force operators to provision for redundant operation under peak load. Idle-energy waste is compounded by losses in the power delivery and cooling infrastructure, which increase power consumption requirements by as much as 50–100% [Moore et al. 2005].

Ideally, we would like to simply turn idle systems off. Unfortunately, a large fraction of servers exhibit frequent but brief bursts of activity [Bash and Forman 2007; Bohrer et al. 2002]. Moreover, user demand often varies rapidly and/or unpredictably, making dynamic consolidation and system shutdown difficult. Our analysis shows that server workloads, especially interactive services, exhibit frequent idle periods of less than one second, which cannot be exploited by existing mechanisms.

Concern over idle-energy waste has prompted calls for a fundamental redesign of each computer system component to consume energy in proportion to utilization [Barroso and Hölzle 2007]. Processor dynamic frequency and voltage scaling (DVFS) exemplifies the energy-proportional concept, providing up to cubic energy savings under reduced load. Unfortunately, processors account for an ever-shrinking fraction of total server power, only 25% in current systems [Fan et al. 2007; Laudon 2006; Lefurgy et al. 2007], and controlling DVFS remains an active research topic [Miyoshi et al. 2002; Wu et al. 2005]. Other subsystems incur many fixed power overheads when active and do not yet offer energy-proportional operation.

We propose an alternative energy-conservation approach, called *PowerNap*, that is attuned to server utilization patterns. With PowerNap, we design the entire system to transition rapidly between a high-performance active state and a minimal-power nap state in response to instantaneous load. Rather than requiring components that provide fine-grain power-performance trade-offs, PowerNap simplifies the system designer's task to focus on two optimization goals: (1) optimizing energy efficiency while napping, and (2) minimizing transition time into and out of the low-power nap state.

Based on the PowerNap concept, we develop requirements and outline mechanisms to eliminate idle power waste in a high-density blade server system. Whereas many mechanisms required by PowerNap are available in existing server components, one critical subsystem of current blade chassis falls short of meeting PowerNap's energy-efficiency requirements: the power conversion system. PowerNap reduces total ensemble power consumption when all blades are napping to only 6% of the peak when all are active. Power supplies are notoriously inefficient at low loads, typically providing conversion efficiency below 70% under 20% load [ECOS and EPR 2008]. These losses undermine PowerNap's energy efficiency.

Directly improving power supply efficiency implies a substantial cost premium. Instead, we introduce the Redundant Array for Inexpensive Load Sharing (RAILS), a power provisioning approach where power draw is shared over an array of low-capacity power supply units (PSUs) built with commodity components. The key innovation of RAILS is to size individual power modules such that the power delivery solution operates at high efficiency across the entire range of PowerNap's power demands. In addition, RAILS provides N + 1 redundancy, graceful compute capacity degradation in the face of multiple power module failures, and reduced component costs relative to conventional enterprise-class power systems. Through modeling and analysis of actual data center workload traces, we demonstrate the following.

—*Analysis of Idle/Busy Intervals in Actual Data Centers.* We analyze utilization traces from production servers and data centers to determine the distribution of idle and active periods. Though interactive servers spend over 60% of their time idle, most idle intervals are under one second.

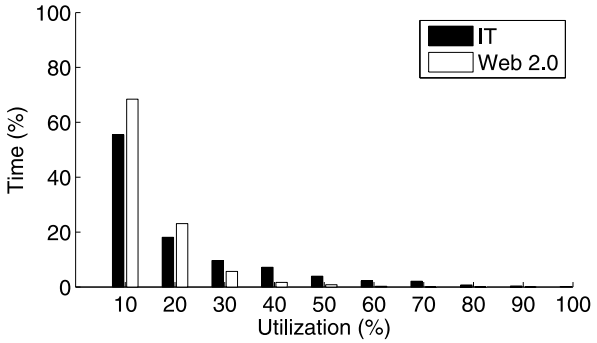


Fig. 1. Server utilization histogram. Real data centers are under 20% utilized.

Table I. Enterprise Data Center Utilization Traces

Workload	Avg. Utilization	Description
Web 2.0	7.4%	"Web 2.0" application servers
IT	14.2%	Enterprise IT Infrastructure apps

—*Energy Efficiency and Response Time Bounds.* Through queuing analysis, we establish bounds on PowerNap's energy efficiency and response time impact. Using our models, we determine that PowerNap is effective if state transition time is below 10 ms, and incurs no overheads below 1 ms. Furthermore, we show that PowerNap provides greater energy efficiency and lower response time than solutions based on DVFS.

—*Experimental Validation of Response Time Impact.* By instrumenting a kernel to emulate PowerNap's transition delays, we validate the response time predictions of our analytic model, confirming that neither CPU caching effects, nor 1 ms transitions significantly impact workload response time.

—*Efficient PowerNap Power Provisioning with RAILS.* Our analysis of commercial data center workload traces demonstrates that RAILS improves average power conversion efficiency from 68% to 86% in PowerNap-enabled servers.

2. UNDERSTANDING SERVER UTILIZATION

It has been well-established in the research literature that the average server utilization of data centers is low, often below 30% [Bash and Forman 2007; Bohrer et al. 2002; Fan et al. 2007]. In facilities that provide interactive services (e.g., transaction processing, file servers, Web 2.0), average utilization is often even worse, sometimes as low as 10% [Bohrer et al. 2002]. Figure 1 depicts a histogram of utilization for two production workloads from enterprise-scale commercial deployments. Table I describes the workloads running on these servers. We derive this data from utilization traces collected over many days, aggregated over more than 120 severs (production utilization traces were provided courtesy of HP Labs). The most striking feature of this data is that the servers spend the vast majority of time under 10% utilization.

Low utilization creates an energy efficiency challenge because conventional servers are notoriously inefficient at low loads. Although power-saving features like clock gating and dynamic voltage and frequency scaling (DVFS) greatly reduce processor power consumption in under-utilized systems, present-day servers still dissipate about 60% as much power when idle as when fully loaded [Chase et al. 2001; Fan et al. 2007; Lefurgy et al. 2007]. Processors often account for only a quarter of system power; main memory and cooling fans contribute larger fractions [Lefurgy et al. 2003].

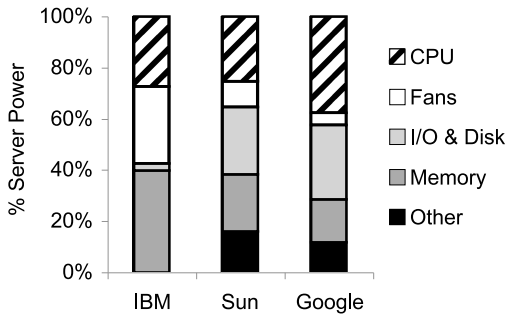


Fig. 2. Server power breakdown. No single component dominates total system power.

Figure 2 reproduces typical server power breakdowns for the IBM p670 [Lefurgy et al. 2003], Sun UltraSparc T2000 [Laudon 2006], and a generic server specified by Google [Fan et al. 2007], respectively.

Given the poor efficiency of under-utilized servers, one obvious approach to improve overall energy efficiency is to increase average server utilization. The recent trend towards server consolidation [Padala et al. 2007] is partly motivated by this objective. By moving services to virtual machines, several services can be time-multiplexed on a single physical server. Consolidation allows the total number of physical servers to be reduced, thereby reducing idle inefficiency. With the availability of live migration, where virtual machines can be transferred among physical hosts during operation without service interruption, it has become possible to operate clusters where servers are brought online and shut down automatically in response to coarse-grain changes in load (e.g., diurnal patterns).

However, dynamic server consolidation cannot eliminate idle energy waste, for several reasons. First, current dynamic consolidation solutions adapt cluster size over 10's of minutes. However, load changes can be far more rapid, particularly when precipitated by an external event (e.g., web server traffic at the end of a World Cup match). For interactive services, peak loads often exceed the average by more than a factor of three [Bohrer et al. 2002]. Second, concerns over performance isolation, service robustness, redundancy, hardware configuration conflicts, and security often preclude consolidation of mission-critical services. Third, the software architectures of some data center workloads preclude cluster resizing. For example, both Web Search [Barroso et al. 2003] and memcached [Memcached 2010] distribute their data sets over an entire cluster, typically without replication, to allow user queries to be processed within tight latency constraints. Under this architecture, there is no straight-forward way to resize a cluster in response to load variation. Though industry trends suggest that consolidation approaches can increase utilization from the 5% to 10% range that is not uncommon today, it is unlikely that utilization above 30%–50% can be achieved for even highly-tuned interactive services.

2.1 Frequent Brief Utilization

Clearly, eliminating server idle power waste is critical to improving data center energy efficiency. Engineers have been successful in reducing idle power in mobile platforms, such as cell phones and laptops. However, servers pose a fundamentally different challenge than these platforms. The key observation underlying our work is that, although servers have low utilization, their activity occurs in frequent, brief bursts. As a result, they appear to be under a constant, light load.

Table II. Fine-Grain Utilization Traces

Workload	Full System Idle	Avg. Interval		Description
		Busy	Idle	
Cluster	36%	3.25 s	1.8 s	600-node scientific computing cluster
DNS	83%	194 ms	923 ms	Department DNS and DHCP server
Mail	45%	115 ms	94 ms	Department POP and SMTP servers
Shell	68%	51 ms	108 ms	Interactive shell and IMAP support
Web	74%	38 ms	106 ms	Department web server
Backup	78%	31 ms	108 ms	Continuous incremental backup server

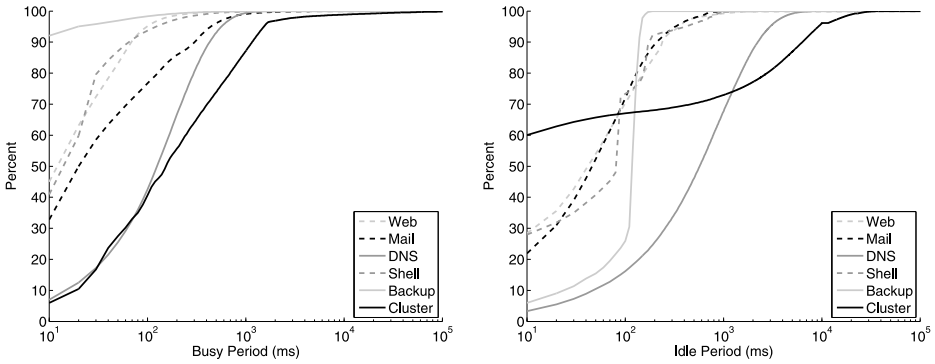


Fig. 3. Busy and idle period cumulative distributions.

To investigate the time scale of servers' idle and busy periods, we have instrumented a series of interactive and batch processing servers to collect utilization traces at 10 ms granularity. To our knowledge, our study is the first to report server utilization data measured at such fine granularity. We classify an interval as busy or idle based on how the OS scheduler accounted the period in its utilization tracking. The traces were collected over a period of a week from seven departmental IT servers and a scientific computing cluster comprising over 600 servers. We present the mean idle and busy period lengths, percent full-system idle time and a brief description of each trace in Table II.

Figure 3 shows the cumulative distribution for the busy and idle period lengths in each trace (i.e., the vertical axis reflects the fraction of the count of idle periods of a given length or shorter). The key result of our traces is that the vast majority of idle periods are shorter than 1s, with mean lengths in the 100's of milliseconds. Busy periods are even shorter, typically only 10's of milliseconds.

DNS and Mail tend to exhibit the densest activity periods, as both of these frequently handle batch-like tasks (e.g., DNS zone transfers). The Mail server also exhibits the highest utilization among the departmental servers. The Web workload experiences the most frequent transitions between busy and idle, as only minimal processing is required to serve the frequent requests for static web pages. The Shell and Backup servers exhibit the largest variation in busy periods. For Shell, this variation arises because users occasionally run long, interactive jobs, whereas for Backup, the length of incremental backup tasks varies with the size of recent file modifications.

At the opposite extreme, the scientific computing cluster exhibits comparatively high utilization (in line with the results reported in Fan et al. [2007] and Bash and Forman [2007]) and an enormous variation in job lengths, from sub-second activities to jobs that run for days. Though the queue of jobs submitted to this cluster is rarely

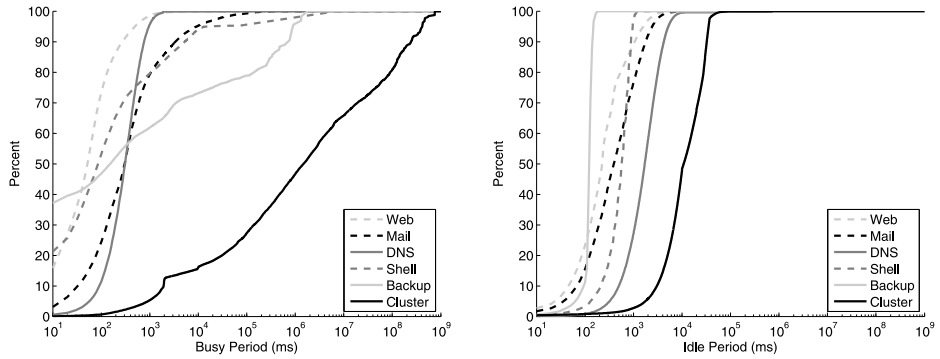


Fig. 4. Busy and idle period weighted cumulative distributions.

empty, because some machines in this pool are dedicated for specific job classes and users, there are also many long idle periods.

The cumulative distribution of busy and idle periods provides insight into the *frequency* of idle and busy events. However, it does not illustrate where the *time* is spent at each time scale. Figure 4 provides the *weighted* CDFs of idle and busy periods; these graphs show the cumulative fraction of idle time that occurs in intervals shorter than the horizontal axis value (i.e., the vertical axis reflects the total time, rather than the count, as in Figure 3, of idle periods). This representation demonstrates the presence of infrequent but long idle and active periods. For example, the fact that the Cluster workload spends a significant amount of time in infrequent, long jobs is immediately clear. More importantly, we can see that the majority of idle time occurs in intervals of up to 100 ms. Even though Figure 3 suggests that most idle periods last 1–10 ms, Figure 4 shows that the majority of time is spent in slightly longer idle intervals.

Our fine-grain utilization traces do not exhaustively represent the space of data center workloads. In particular, with the exception of the Cluster workload, we have specifically focused on interactive services, which present substantial power management challenges because of their latency constraints. The average utilization levels we observe for these workloads qualitatively match the behavior seen in the customer-provided traces of Figure 1 and reports from other sources [Bash and Forman 2007; Bohrer et al. 2002; Fan et al. 2007]. Data centers also often run batch-oriented scientific and data intensive (e.g., MapReduce) tasks, which are more similar to our Cluster workload. Such workloads typically have looser latency constraints and are more amenable to consolidation and scheduling-based approaches, which increase average utilization and coalesce idle periods.

2.2 Existing Energy-Conservation Techniques

Although support for sleep states is widespread in handheld, laptop and desktop machines, these states are rarely used in current server systems. Unfortunately, the high restart latency typical of current sleep states renders them unacceptable for interactive services; current laptops and desktops require several seconds to suspend using operating system interfaces (e.g., ACPI). The specifications for these sleep states were not developed to enable millisecond-scale sleep transitions; they transition the power state of individual devices sequentially through elaborate driver APIs that introduce numerous overheads. Moreover, they are not software-transparent: mode switches can have numerous side effects, for example closing active network connections.

Recent server processors include CPU throttling solutions (e.g. Intel Speedstep, AMD Cool'n'Quiet) to reduce the large overhead of light loads. These processors use

DVFS to reduce their operating frequency linearly while gaining cubic power savings. DVFS relies on operating system support to tune processor frequency to instantaneous load. In Linux, the kernel continues lowering frequency until it observes ∼20% idle time. Improving DVFS control algorithms remains an active research area [Miyoshi et al. 2002; Wu et al. 2005]. Nonetheless, DVFS can be highly effective in reducing CPU power. However, as Figure 2 shows, CPUs account for a small portion of total system power.

Several research proposals have sought to exploit idle periods of individual memory banks to conserve memory power [Delaluz et al. 2001; Diniz et al. 2007; Huang et al. 2005; Lebeck et al. 2000]. During execution, if a particular bank is predicted/detected to be idle, it is transitioned to a low-power mode and re-activated upon a subsequent access. These approaches conserve memory energy during execution at a small penalty in performance, for example, one study of desktop/engineering applications reports that using RDRAM's nap mode cuts DRAM energy 60% to 85% for a few percent performance loss [Lebeck et al. 2000]. However, a common conclusion across several studies is that the deepest-available low-power modes (power-down in RDRAM and self-refresh in DDR DRAM) cannot be used effectively because of performance overheads of frequent mode transitions that delay many memory accesses. In contrast, because PowerNap transitions the entire memory system between states on system-level active/idle transitions, it can leverage the deeper sleep modes, which further reduce memory power by more than an order-of-magnitude relative to nap.

Energy proportional computing [Fan et al. 2007] seeks to extend the success of DVFS to the entire system. In this scheme, each system component is redesigned to consume energy in proportion to utilization. In an energy-proportional system, explicit power management is unnecessary, as power consumption varies naturally with utilization. However, as many components incur fixed power overheads when active (e.g., clock power on synchronous memory busses, leakage power in CPUs, etc.) designing energy-proportional subsystems remains a research challenge.

Energy-proportional operation can be approximated with non-energy-proportional systems through dynamic virtual machine consolidation over a large server ensemble [Tolia et al. 2008]. However, such approaches do not address the performance isolation concerns of dynamic consolidation and operate at coarse time scales (minutes). Hence, they cannot exploit the brief idle periods found in servers.

3. POWERNAP

Although servers spend most of their time idle, conventional energy-conservation techniques are unable to exploit these brief idle periods. Hence, we propose an approach to power management that enables the entire system to transition rapidly into and out of a low-power state where all activity is suspended until new work arrives. We call our approach *PowerNap*.

Figure 5 illustrates the PowerNap concept. Each time the server exhausts all pending work, it transitions to the nap state. In this state, nearly all system components enter sleep modes, which are already available in many components (see Section 5). While in the nap state, power consumption is low, but no processing can occur. System components that signal the arrival of new work, expiration of a software timer, or environmental changes, remain partially powered. When new work arrives, the system wakes and transitions back to the active state. When the work is complete, the system returns to the nap state.

PowerNap is simpler than many other energy conservation schemes because it requires system components to support only two operating modes: an active mode that provides maximum performance and a nap mode that minimizes power draw. For many devices, providing a low-power nap mode is far easier than providing multiple

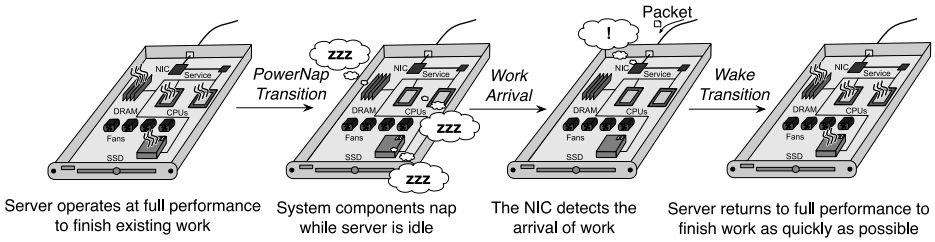


Fig. 5. PowerNap.

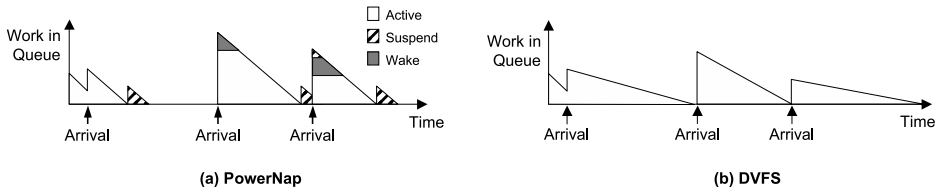


Fig. 6. PowerNap and DVFS analytic models.

active modes that trade performance for power savings. Any level of activity often implies fixed power overheads (e.g., bus clock switching, power distribution losses, leakage power, mechanical components, etc.) We outline mechanisms required to implement PowerNap in Section 5.

3.1 PowerNap Performance and Power Model

To assess PowerNap's potential, we develop a queuing model that relates its key performance measures—power consumption and response time penalty—to workload parameters and PowerNap implementation characteristics. We contrast PowerNap with a model of the upper-bound energy-savings possible with DVFS. The goal of our model is threefold: (1) to gain insight into PowerNap behavior, (2) to derive requirements for PowerNap implementations, and (3) to contrast PowerNap and DVFS.

We model both PowerNap and DVFS under the assumption that each seeks to minimize the energy required to serve the offered load. Hence, both schemes provide identical throughput (matching the offered load) but differ in response time and energy consumption.

PowerNap Model. We model PowerNap as an M/G/1 queuing system with arrival rate λ, and a generalized service time distribution with known first and second moments $E[S]$ and $E[S^2]$. Figure 6(a) shows the work in the queue for three job arrivals. Note that, in this context, work also includes time spent in the wake and suspend states. Average server utilization is given by $\rho = \lambda E[S]$. To model the effects of PowerNap suspend and wake transitions, we extend the conventional M/G/1 model with an exceptional first service time [Welch 1964]. We assume PowerNap transitions are symmetric with latency T_t. Service of the first job in each busy period is delayed by an initial setup time I. The setup time includes the wake transition and may include the remaining portion of a suspend transition as shown for the rightmost arrival in Figure 6(a). Hence, for an arrival x time units from the start of the preceding idle period, the initial setup time is given by:

$$I = \begin{cases} 2T_t - x & \text{if } 0 \le x < T_t \\ T_t & \text{if } x \ge T_t. \end{cases}$$

The first and second moments $E[I]$ and $E[I^2]$ are:

$$\begin{aligned} E[I] &= \int_0^\infty I\lambda e^{-\lambda x}dx = 2T_t + \frac{1}{\lambda}e^{-\lambda T_t} - \frac{1}{\lambda} \\ E[I^2] &= \int_0^\infty I^2\lambda e^{-\lambda x}dx \\ &= 4T_t^2 - 2T_t^2 e^{-\lambda T_t} - \left(\frac{4T_t}{\lambda} + \frac{2}{\lambda^2}\right)\left[1 - (1+\lambda T_t)e^{-\lambda T_t}\right]. \end{aligned}$$

We compute average power as

$$P_{avg} = P_{nap} \cdot Pr(\text{nap}) + P_{max}(1 - Pr(\text{nap})),$$

where the fraction of time spent napping $Pr(\text{nap})$ is given by the ratio of the expected length of each nap period $E[N]$ to the expected busy-idle cycle length $E[C]$:

$$\begin{aligned} Pr(\text{nap}) &= \frac{\int_0^{T_t}(0)\lambda e^{-\lambda t}dt + \int_{T_t}^\infty (t - T_t)\lambda e^{-\lambda t}dt}{\frac{E[S]+E[I]}{1-\lambda E[S]} + \frac{1}{\lambda}} \\ &= \frac{e^{-\lambda T_t}\left(1 - \lambda E[S]\right)}{1 + \lambda E[I]}. \end{aligned}$$

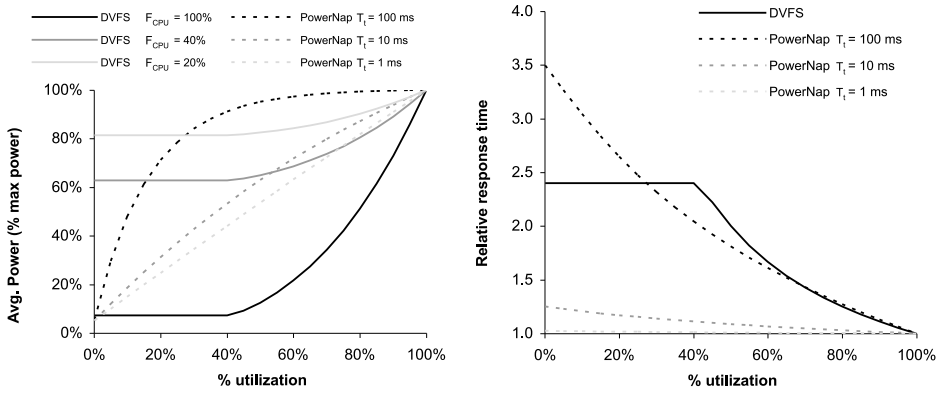
The response time for an M/G/1 server with exceptional first service is due to Welch [1964]:

$$E[R] = \frac{\lambda E[S^2]}{2(1-\lambda E[S])} + \frac{2E[I] + \lambda E[I^2]}{2(1+\lambda E[I])} + E[S].$$

Note that the first term of $E[R]$ is the Pollaczek-Khinchin formula for the expected queuing delay in a standard M/G/1 queue, the second term is additional residual delay caused by the initial setup time I, and the final term is the expected service time $E[S]$. The second term vanishes when $T_t = 0$.

DVFS Model. Rather than model a real DVFS frequency control algorithm, we instead model the upper bound of energy savings possible with DVFS. For each job arrival, we scale instantaneous frequency f to stretch the job to fill any idle time until the next job arrival, as illustrated in Figure 6(b), which gives $E[f] = f_{max}\rho$. This scheme maximizes power savings, but cannot be implemented in practice because it requires knowledge of future arrival times. We base power savings estimates on the theoretical formulation of processor dynamic power consumption $P_{CPU} = \frac{1}{2}CV^2Af$. We assume C and A are fixed, and choose the optimal f for each job within the range $f_{min} < f < f_{max}$. We impose a lower bound $f_{min} = f_{max}/2.4$ to prevent response time from growing asymptotically when utilization is low. We chose a factor of 2.4 between f_{min} and f_{max} based on the frequency range provided by a 2.4-GHz AMD Athlon. We assume voltage scales linearly with frequency (i.e., $V = V_{max}(f/f_{max})$), which is optimistic with respect to current DVFS implementations. Finally, as DVFS only reduces the CPU's contribution to system power, we include a parameter F_{CPU} to control the fraction of total system power affected by DVFS. Under these assumptions, average power P_{avg} is given by:

$$P_{avg} = P_{max}(1 - F_{CPU}(\frac{E[f]}{f_{max}})^3).$$



(a) Power Scaling (b) Response Time Scaling

Fig. 7. PowerNap and DVFS power and response time scaling.

Response time is given by:

$$E[R] = E\left[\frac{R_{base}}{f/f_{max}}\right],$$

where R_{base} is the response time without DVFS.

3.2 Analysis

Power Savings. Figure 7(a) shows the average power (as a fraction of peak) required under PowerNap and DVFS as a function of utilization. For DVFS, we show power savings for three values of F_{CPU}. F_{CPU} = 100% represents the upper bound if DVFS were applicable to all system power. 20% $< F_{CPU} <$ 40% bound the typical range in current servers. For PowerNap, we construct the graphs with $E[s]$ = 38 ms and $E[s^2] = 3.7E[s]$, which are both estimated from the observed busy period distribution in our Web trace. We assume P_{nap} is 5% of P_{max}. We vary λ to adjust utilization, and present results for three values of T_t: 1 ms, 10 ms, and 100 ms. We expect 10 ms to be a conservative estimate for achievable PowerNap transition time. For transition times below 1 ms, transition time becomes negligible and the power savings from PowerNap varies linearly with utilization for all workloads. We discuss transition times further in Section 5.

When F_{CPU} is high, DVFS clearly outperforms PowerNap, as it provides cubic power savings while PowerNap's savings are at best linear in utilization. However, for realistic values of F_{CPU} and transition times in our expected range ($T_t \leq$ 10 ms), PowerNap's savings rapidly overtake DVFS. As transition time increases, the break-even point between DVFS and PowerNap shifts towards lower utilization. Even for a transition time of 100 ms, PowerNap can provide substantial energy savings when utilization is below 20%.

Response Time. In Figure 7(b), we compare the response time impact of DVFS and PowerNap. The vertical axis shows response time normalized to a system without power management (i.e., that always operates at f_{max}). For DVFS, response time grows rapidly when the gap between job arrivals is large, and reaches the f_{min} floor below 40% utilization. DVFS response time penalty is independent of F_{CPU}, and is bounded at 2.4 by the ratio of f_{max}/f_{min}. For PowerNap, the response time penalty is negligible if T_t

Table III. Per-Workload Energy Savings and Response Time Penalty

Workload	PowerNap		DVFS	
	Energy Savings	Δ Latency	Energy Savings	Δ Latency
Cluster	34%	0.2%	18%	156%
DNS	77%	5.1%	23%	240%
Mail	35%	11%	21%	181%
Shell	55%	13%	23%	240%
Web	59%	13%	23%	240%
Backup	61%	7.6%	23%	240%

is small relative to average service time E[S], which we expect to be the common case (i.e., most jobs last longer than 10 ms). However, if T_t is significant relative to E[S], the PowerNap response time penalty grows as utilization shrinks. When utilization is high, the server is rarely idle and few jobs are delayed by transitions. As utilization drops, the additional delay seen by each job converges to T_t (i.e., every job must wait for wake-up).

Per-Workload Energy Savings. Finally, we report the energy savings under simulated PowerNap and DVFS schemes for our workload traces. Because these traces only contain busy and idle periods, and not individual job arrivals, we cannot estimate response time impact. For each workload, we perform a trace-based simulation that assumes busy periods will start at the same time, independent of the current PowerNap state (i.e., new work still arrives during wake or suspend transitions). We assume a PowerNap transition time of 10 ms and nap power at 5% of active power, which we believe to be conservative estimates (see Section 5). For DVFS, we assume F_{CPU} = 25%. Table III shows the results of these simulations. All workloads except Mail and Cluster hit the DVFS frequency floor, and, hence, achieve a 23% energy savings. In all cases, PowerNap achieves greater energy savings. Additionally, we extracted the average arrival rate (assuming a Poisson arrival process) and compared the results in Table III with the M/G/1 model of Pr(nap) previously derived. We found that for these traces, the analytic model was within 2% of our simulated results in all cases. When arrivals are more deterministic (e.g., Backup) than the exponential we assume, the model slightly overestimates PowerNap savings. For more variable arrival processes (e.g., Shell), the model underestimates the energy savings.

3.3 Implementation Requirements

Based on the results of our analytic model, we identify two key PowerNap implementation requirements:

Fast Transitions. Our model demonstrates that transition speed is the dominant factor in determining both the power savings potential and response time impact of PowerNap. Our results show that transition time must be less than one tenth of average busy period length. Although a 10-ms transition speed is sufficient to obtain significant savings, 1ms transitions are necessary for PowerNap's overheads to become negligible. To achieve these transition periods, a PowerNap implementation must preserve volatile system state (e.g., memory) while napping—mass storage devices transfer rates are insufficient to transfer multiple GB of memory state in milliseconds.

Minimizing Power Draw in Nap State. Given the low utilization in most enterprise deployments, servers will spend a majority of time in the nap state, making PowerNap's power requirements the key factor affecting average system power. Hence, it is critical to minimize the power draw of napping system components. As a result

Benchmark	Relative Response Time
SPECweb	1.01
SPECpower	1.00

Fig. 8. Cache effect.

of eliminating idle power, PowerNap drastically increases the range between the minimum and maximum power demands on a blade chassis. Existing blade-chassis power-conversion systems are inefficient in the common case, where all blades are napping. Hence, to maximize PowerNap potential, we must re-architect the blade chassis power subsystem to increase its efficiency at low loads.

Although PowerNap requires system-wide modifications, it demands only two states from each subsystem: active and nap states. Hence, implementing PowerNap is substantially simpler than developing energy-proportional components. Because no computation occurs while napping, many fixed power draws, such as clocks and leakage power, can be conserved.

4. EMULATING POWERNAP TRANSITION PERFORMANCE IMPACT

The PowerNap architecture can impact application response time in two ways: transitions in and out of the nap state delay responses and some processors may flush on-chip caches when transitioning. To investigate these effects in greater detail, we have instrumented a Linux kernel to insert transition delays and flush CPU caches when exiting from idle, emulating PowerNap's performance impact. Using this emulation, we have examined PowerNap's impact on the response time of a web serving benchmark.

4.1 Cache Effects

The static power of processor caches consumes a large and potentially growing fraction of overall CPU power budget, particularly when idle. Accordingly, sleep modes available in some CPUs may turn off caches, flushing their contents. The ACPI standard leaves it unspecified whether cache contents are preserved during ACPI sleep states, and implementations vary across vendors and processor generations. We wish to characterize the performance impact of discarding cache contents during PowerNap transitions, to determine if it is important for PowerNap to use only cache-state-preserving sleep modes.

To produce the effect of flushing the cache, we instrument the kernel to issue the x86 `WBINVD` instruction (which writes back and then invalidates the entire contents of CPU caches [Intel 2009]) when emulating a PowerNap transition. We have tested our modified kernel using a microbenchmark that strides over L1 and L2-sized data structures to confirm that the `WBINVD` instruction discards the contents of both the L1 and L2 caches. We test the effect of flushing the cache each time the server becomes idle (i.e., upon entry to the OS idle loop) for the SPECweb and SPECpower benchmarks [Standard Performance Evaluation Corporation 2005, 2008]. Figure 8 shows that the average response time for these benchmarks does not change appreciably as the cold-start cache effect is small relative to the average response time.

4.2 Transition Latency

We further investigate the impact of PowerNap transition time to understand how various values of T_t affect a workload. To emulate a wide spectrum of delays, we instrument the Linux kernel to artificially insert delays when exiting the idle loop. The instrumentation tracks the time since the end of the last job such that the delay

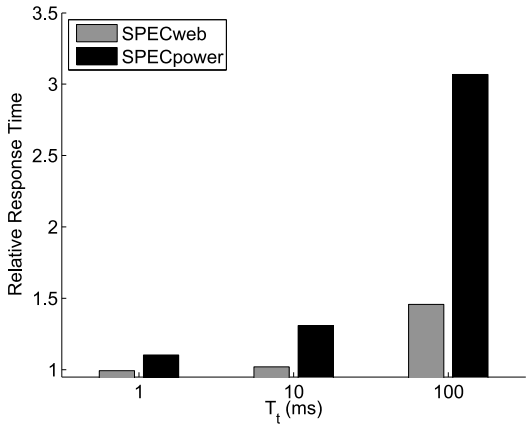


Fig. 9. Emulating PowerNap.

is I as described in the model in Section 3 (i.e., it accounts for both sleep and wake transitions, falling between T_t and $2T_t$).

Figure 9 reports the average response time of SPECweb and SPECpower for a T_t of 1, 10 and 100 ms including cache flush effects. Our measurements confirm the model predictions, showing that a 100 ms transition time has a considerable response time impact. However, a T_t of 10 ms results in tolerable delay and 1ms incurs a negligible performance impact. Furthermore, we see that SPECpower is more sensitive to transition latency because of it's shorter average service time.

5. POWERNAP MECHANISMS

We outline the design of a PowerNap-enabled blade server system and enumerate required implementation mechanisms. PowerNap requires nap support in all hardware subsystems that have nonnegligible idle power draws, and software/firmware support to identify and maximize idle periods and manage state transitions.

5.1 Hardware Mechanisms

At the component level, the sleep states required by PowerNap are already available in many products, particularly those targeted to mobile devices. However, few of these mechanisms are exploited in existing servers, and some are omitted in current-generation server-class components. Moreover, the operating system APIs that control sleep/wake transitions in current desktops and laptops introduce enormous overheads that dominate the transition latency, making them inapplicable for PowerNap.

For each hardware subsystem, we identify existing mechanisms or outline requirements for new mechanisms necessary to implement PowerNap. Furthermore, we provide estimates of power dissipation while napping and transition speed. We summarize these estimates, along with our sources, in Table IV. Our estimates for a "Typical Blade" are based on HP's c-series half-height blade designs; our PowerNap power estimate assumes a two-CPU system with eight DRAM DIMMs.

Processor: ACPI S3 "Sleep" State. The ACPI standard defines the S3 "Sleep" state for processors that is intended to allow low-latency transitions. Although the ACPI standard does not specify power or performance requirements, some implementations of S3 are ideal for PowerNap. For example, in Intel's mobile processor line, S3 preserves last-level cache state and consumes only 3.4W [Intel 2007]. These processors

Table IV. Component Power Consumption

Component	Power			Transition	Sources
	Active	Idle	Nap		
CPU chip	80–150W	12–20W	3.4W	30 μs	[Intel 2007] [Intel 2005]
DRAM DIMM	3.5–5W	1.8–2.5W	0.2W	$< 1\mu$s	[Micron 2004] [Hynix 2008]
NIC	0.7W	0.3W	0.3W	no trans.	[SMSC 2008]
SSD	1W	0.4W	0.4W	no trans.	[Samsung 2008]
Fan	10–15W	1–3W	-	independent	[Leigh and Ranganathan 2007]
PSU	50–60W	25–35W	0.5W	300 μs	[National Semiconductor 2002]
Typical Blade	450W	270W	10.4W	300 μs	

require approximately 30 μs for PLL stabilization to transition from sleep back to active execution [Intel 2005].

If S3 is unavailable, clock gating can provide substantial energy savings. For example, Intel's Xeon 5400-series power requirements drop from 80W to 16W upon executing a halt instruction [Intel 2008]. From this state, resuming execution requires only nanosecond-scale delays.

DRAM: Self-Refresh. DRAM is typically the second-most power-hungry system component when active. However, several recent DRAM specifications feature an operating mode, called self-refresh, where the DRAM is isolated from the memory controller and autonomously refreshes DRAM content. In this mode, the memory bus clock and PLLs are disabled, as are most of the DRAM interface circuitry. Self-refresh saves more than an order of magnitude of power. For example, a 2GB SODIMM (designed for laptops) with a peak power draw above 5W uses only 202mW of power during self-refresh [Micron 2004]. Transitions into and out of self-refresh can be completed in less than a microsecond [Hynix 2008].

Mass Storage: Solid State Disks. Solid state disks draw negligible power when idle, and, hence, do not need to transition to a sleep state for PowerNap. A recent 64-GB Samsung SSD consumes only 0.32W while idle [Samsung 2008].

Network Interface: Wake-on-LAN. The key responsibility PowerNap demands of the network interface card (NIC) is to wake the system upon arrival of a packet. Existing NICs already provide support for Wake-on-LAN to perform this function. Current implementations of Wake-on-LAN provide a mode to wake on any physical activity. This mode forms a basis for PowerNap support. Current NICs consume only 400mW while in this mode [SMSC 2008].

Environmental Monitoring and Service Processors: PowerNap Transition Management. Servers typically include additional circuitry for environmental monitoring, remote management (e.g., remote power on), power capping, power regulation, and other functionality. These components typically manage ACPI state transitions and would coordinate PowerNap transitions. A typical service processor draws less than 10mW when idle.

Fans: Variable Speed Operation. Fans are a dominant power consumer in many recent servers. Modern servers employ variable-speed fans where cooling capacity is constantly tuned based on observed temperature or power draw. Fan power requirements typically grow cubically with average power. Thus, PowerNap's average power savings yield massive reductions in fan power requirements. In most blade designs, cooling systems are centralized in the blade chassis, amortizing their energy cost over many blades. Because thermal conduction progresses at drastically different timescales than

PowerNap's transition frequency, chassis-level fan control is independent of PowerNap state (i.e., fans may continue operating during nap and may spin down during active operation depending on temperature conditions).

Power Provisioning: RAILS. PowerNap fundamentally alters the range of currents over which a blade chassis must efficiently supply power. In Section 6, we explain why conventional power delivery schemes are unable to provide efficient AC to DC conversion over this range, and present RAILS, our power conversion solution.

5.2 Software Mechanisms

Existing software support for sleep modes in desktop and laptop (e.g., ACPI) fails to meet the needs of PowerNap in several ways. First, system-wide sleep transitions are exceedingly slow (often requiring seconds) because individual devices are transitioned among modes sequentially through elaborate driver interfaces. To achieve acceptable transition latencies, devices must transition in parallel without complex operating system interactions. Second, existing APIs contain complexity and features (e.g., support for multiple power modes and per-device state management) that are not needed for PowerNap and introduce unnecessary overheads. Third, current state transitions are not software-transparent—most operating systems notify applications prior to a state change and have numerous visible side-effects (e.g., closing active network connections). Finally, these APIs do not provide adequate mechanisms to schedule the system to wake from sleep at a specific time in the future.

For schemes like PowerNap, the periodic timer interrupt used by legacy OS kernels to track the passage of time and implement software timers poses a challenge. As the timer interrupt is triggered every 1ms, conventional OS time keeping precludes the use of PowerNap. The periodic clock tick also poses a challenge for idle-power conservation on laptops and for virtualization platforms that consolidate hundreds of OS images on a single hardware platform. Hence, the Linux kernel has recently been enhanced to support "tickless" operation, where the periodic timer interrupt is eschewed in favor of hardware timers for scheduling and time keeping [Siddha et al. 2007]. PowerNap depends on a kernel that provides tickless operation.

PowerNap's effectiveness increases with longer idle periods and less frequent state transitions. Some existing hardware devices (e.g., legacy keyboard controllers) require polling to detect input events. Current operating systems often perform maintenance tasks (e.g., flushing disk buffers, zeroing memory) when the OS detects significant idle periods. These maintenance tasks may interact poorly with PowerNap and can induce additional state transitions. However, efforts are already underway (e.g., as described in Siddha et al. [2007]) to redesign device drivers and improve background task scheduling.

6. RAILS

AC to DC conversion losses in computer systems have recently become a major concern, leading to a variety of research proposals [Hölzle and Weihl 2006; Leigh and Ranganathan 2007], product announcements (e.g., HP's Blade System c7000), and standardization efforts [ECOS and EPR 2008] to improve power supply efficiency. The concern is particularly acute in data centers, where each watt wasted in the power delivery infrastructure implies even more loss in cooling. Because PowerNap's power draw is substantially lower than the idle power in conventional servers, PowerNap demands conversion efficiency over a wide power range, from as few as 300W to as much as 7.2 kW in a fully populated enclosure.

In this section, we discuss why existing power solutions are inadequate for PowerNap and present RAILS, our power solution. RAILS provides high conversion

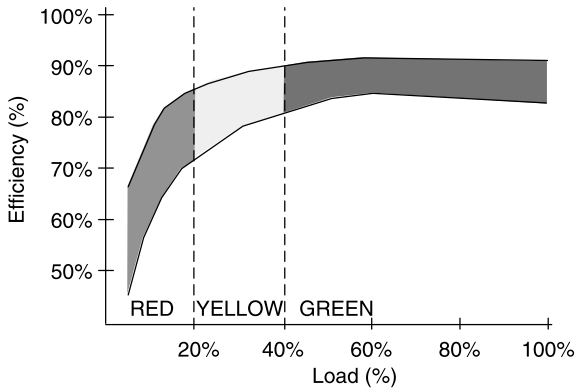


Fig. 10. Power supply efficiency.

efficiency across PowerNap's power demand spectrum, provides N + 1 redundancy, allows for graceful degradation of compute capacity when PSUs fail, and minimizes costs by using commodity PSUs in an efficient arrangement.

6.1 Power Supply Unit Background

Poor Efficiency at Low Loads. Although manufacturers often report only a single efficiency value, most PSUs do not have a constant efficiency across electrical load. A recent survey of server and desktop PSUs reported their efficiency across loads [ECOS and EPR 2008]. Figure 10 reproduces the range of efficiencies reported in that study. Though PSUs are often over 90% efficient at their optimal operating point (usually near 75% load), efficiency drops off rapidly below 40% load, sometimes dipping below 50% (i.e., >2W in for 1W out). We divide the operating efficiency of power supplies into three zones based on electrical load. Above 40% load, the PSUs operate in the "green" zone, where their efficiency is at or above 80%. In the 20–40% "yellow" zone, PSU efficiency begins to drop, but typically exceeds 70%. However, in the "red" zone below 20%, efficiency drops off precipitously.

Two factors cause servers to frequently operate in the "yellow" or "red" efficiency zones. First, servers are highly configurable, which leads to a large range of power requirements. The same server model might be sold with only one or as many as 20 disks installed, and the amount of installed DRAM might vary by a factor of 10. Furthermore, peripherals may be added after the system is assembled. To simplify ordering, upgrades, testing, and safety certification, manufacturers typically install a power supply rated to exceed the power requirements of the most extreme configuration. Second, servers are often configured with 2N redundant power supplies (i.e., twice as many as are required for a worst-case configuration). The redundant supplies typically share the electrical load to minimize PSU temperature and to ensure current flow remains uninterrupted if a PSU fails. However, the EPRI study [ECOS and EPR 2008] concluded that this load-sharing arrangement often shifts PSUs from "yellow"-zone to "red"-zone operation.

Recent Efficiency Improvements. A variety of recent initiatives seek to improve server power efficiency:

— *80+ Certification.* The EPA Energy Star program has defined the "80+" certification standard [U.S. EPA 2007a] to incentivize PSU manufacturers to improve efficiency at low loads. The 80+ incentive program is primarily targeted at the low-peak-power

desktop PSU market. 80+ supplies require considerably higher design complexity than conventional PSUs, which may pose a barrier to widespread adoption in the reliability-conscious server PSU market. Added circuit components and tighter tolerances add to the cost of the PSU. Furthermore, despite their name, the 80+ specification does not require energy efficiency above 80% across all loads, rather, only within the typical operating range of conventional systems. This specified efficiency range is not wide enough for PowerNap.

— *Single Voltage Supplies.* Unlike desktop machines, which require five different DC output voltages to support legacy components, server PSUs typically provide only a single DC output voltage, simplifying their design and improving reliability and efficiency [Hölzle and Weihl 2006]. Although PowerNap benefits from this feature, a single output voltage does not directly address inefficiency at low loads.

— *DC Distribution.* Recent research [Hölzle and Weihl 2006] has called for distributing DC power among data center racks, eliminating AC-to-DC conversion efficiency concerns at the blade enclosure level. However, the efficiency advantages of DC distribution are unclear [Rasmussen 2007] and deploying DC power will require multi-industry coordination.

— *Dynamic Load-Sharing.* Blade enclosures create a further opportunity to improve efficiency through dynamic load-sharing. HP's Dynamic Power Saver [Leigh and Ranganathan 2007] feature in the HP Blade Center c7000 employs up to six high-efficiency 2.2kW PSUs in a single enclosure, and dynamically varies the number of PSUs that are engaged, ensuring that all active supplies operate in their "green" zone while maintaining redundancy. Although HP's solution is ideal for the idle and peak power range of the c-class blades, it requires expensive PSUs and provides insufficient granularity for PowerNap.

While all these solutions improve efficiency for their target markets, none achieve all our goals of efficiency for PowerNap, redundancy, and low cost.

6.2 RAILS Design

We introduce a new power delivery solution tuned for PowerNap: the Redundant Array for Inexpensive Load Sharing (RAILS). The central idea of our scheme is to load- share over multiple inexpensive, small PSUs to provide the efficiency and reliability of larger, more expensive units. Through intelligent sizing and load-sharing, we ensure that active PSUs operate in their efficiency sweet spots. Our scheme provides 80+ efficiency and enterprise-class redundancy with commodity components.

RAILS targets three key objectives: (1) efficiency across the entire PowerNap dynamic power range; (2) N + 1 reliability and graceful degradation of compute capacity under multiple PSU failure; and (3) minimal cost.

Figure 11 illustrates RAILS. As in conventional blade enclosures, power is provided by multiple PSUs connected in parallel. A conventional load-sharing control circuit continuously monitors and controls the PSUs to ensure load is divided evenly among them. As in Dynamic Smart Power [Leigh and Ranganathan 2007], RAILS disables and electrically isolates PSUs that are not necessary to supply the load. However, our key departure from prior designs is in the granularity of the individual PSUs. We select PSUs from the economic sweet spot of the high-sales-volume market for low-wattage commodity supplies.

We choose a power supply granularity to satisfy two criteria: (1) A single supply must be operating in its "green" zone when all blades are napping. This criterion establishes an upper bound on the PSU capacity based on the minimum chassis power draw when all blades are napping. (2) Subject to this bound, we size PSUs to match the incremental power draw of activating a blade. Thus, as each blade awakens, one

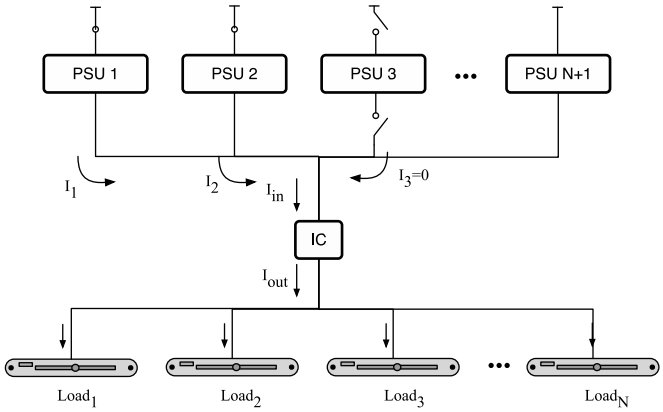


Fig. 11. RAILS PSU design.

additional PSU is brought on line. Because of intelligent sizing, each of these PSUs will operate in their optimal efficiency region. Whereas current blade servers use multi-kilowatt PSUs, a typical RAILS PSU might supply 500W.

RAILS meets its cost goals by incorporating high-volume commodity components. Although the form-factor of commodity PSUs may prove awkward for rack-mount blade enclosures, precluding the use of off-the-shelf PSUs, the power density of high-sales-volume PSUs differs little from high-end server supplies. Hence, with appropriate mechanical modifications, it is possible to pack RAILS PSUs in roughly the same physical volume as conventional blade enclosure power systems.

RAILS meets its reliability goals by providing fine-grain degradation of the system's peak power capacity as PSUs fail. In any N + 1 design, the first PSU failure does not affect compute capacity. However, in conventional blade enclosures, a subsequent failure may force shutdown of several (possibly all) blades. Multiple-failure tolerance typically requires 2N redundancy, which is expensive. In contrast, in RAILS, where PSU capacity is matched to the active power draw of a single blade, the second and subsequent failures each require the shutdown of only one blade.

6.3 Evaluation

We evaluate the power efficiency and cost of PowerNap with four power supply designs, commodity supplies ("Commodity"), high-efficiency 80+ supplies ("80+"), dynamic load sharing ("Dynamic"), and RAILS ("RAILS"). We evaluate all four designs in the context of a PowerNap-enabled blade system similar to HP's Blade Center c7000. We assume a fully populated chassis with 16 half-height blades. Each blade consumes 450W at peak, 270W at idle without PowerNap, and 10.4W in PowerNap (see Table IV). We assume the blade enclosure draws 270W (we neglect any variation in chassis power as a function of the number of active blades). The non-RAILS systems employ 4 2250W PSUs (sufficient to provide N + 1 redundancy). The RAILS design uses 17 500W PSUs. We assume the average efficiency characteristic from Figure 10 for commodity PSUs.

Cost. Server components are sold in relatively low volumes compared to desktop or embedded products, and thus, command premium prices. Some Internet companies (e.g., Google), have eschewed enterprise servers and instead assemble systems from commodity components to avoid these premiums. PSUs present another opportunity to capitalize on low-cost commodity components. Because desktop ATX PSUs are sold in massive volumes, their constituent components are cheap. A moderately-sized supply

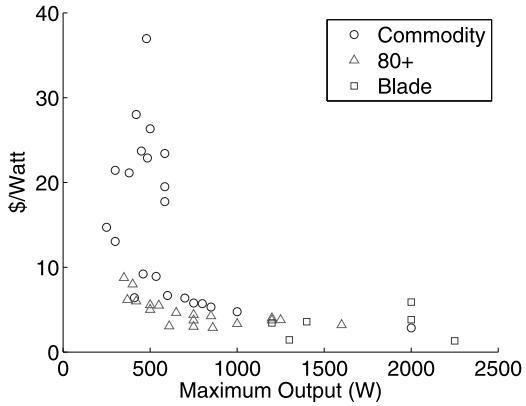


Fig. 12. Power supply pricing.

Table V. Relative PSU Density

	microATX	ATX	Custom Blade
Density (Normalized W/vol.)	675.5	1000	1187

can be obtained at extremely low cost. Figure 12 shows a survey of PSU prices in Watts per dollar for a wide range of PSUs across market segments. Price per Watt increases rapidly with power delivery capacity. This rise can be attributed to the proportional increase in required size for power components such as inductors and capacitors. Also, the price of discrete power components grows with size and maximum current rating. Presently, the market sweet spot is around 500W supplies. Both 80+ and blade server PSUs are substantially more expensive than commodity parts. Because RAILS uses commodity PSUs with small maximum outputs, it takes advantage of PSU market economics, making RAILS far cheaper than proprietary blade PSUs.

Power Density. In data centers, rack space is at a premium, and, hence, the physical volume occupied by a blade enclosure is a key concern. RAILS drastically increases the number of distinct PSUs in the enclosure, but each PSU is individually smaller. To confirm the feasibility of RAILS, we have compared the highest power density available in commodity PSUs, which conform to one of several standard form-factors, with that of PSUs designed for blade centers, which may have arbitrary dimensions. Table V compares the power density of two commodity form factors with the power density of HP's c7000 PSUs. We report density in terms of Watts per unit volume normalized to the volume of one ATX power supply. The highly compact microATX form factor exhibits the worst power density—these units have been optimized for small dimensions but are employed in small form-factor devices that do not require high peak power. Though they are not designed for density, commodity ATX supplies are only 16% less dense than enterprise-class supplies. Furthermore, as RAILS requires only a single output voltage, eliminating the need for many of a standard ATX PSU's components, we conclude that RAILS PSUs fit within blade enclosure volumetric constraints.

Power Savings and Energy Efficiency. To evaluate each power system, we calculate expected power draw and conversion efficiency across blade ensemble utilizations. As noted in Section 2, low average utilization manifests as brief bursts of activity where a subset of blades draw near-peak power. The efficiency of each power delivery solution depends on how long blades are active and how many are simultaneously active. For each utilization, we construct a probability mass function for the number of

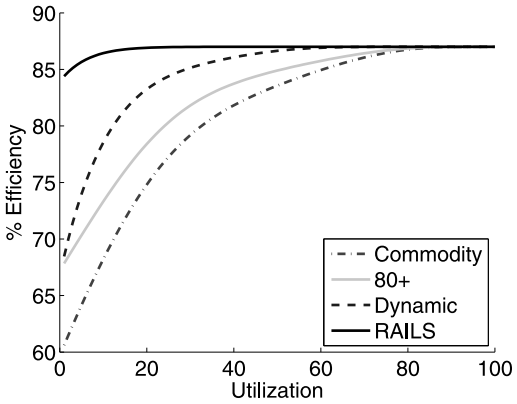


Fig. 13. Power delivery solution comparison.

simultaneously active blades, assuming utilization across blades is uncorrelated. Hence, the number of active blades follows a binomial distribution. From the distribution of active blades, we compute an expected power draw and determine conversion losses from the power supply's efficiency- versus-load curve. We obtain efficiency curves from the Energy Star Bronze 80+ specification [U.S. EPA 2007a] for 80+ PSUs and [ECOS and EPR 2008] for commodity PSUs.

Figure 13 compares the relative efficiency of PowerNap under each power delivery solution. Using commodity ("Commodity") or high efficiency ("80+") PSUs results in the lowest efficiency, as PowerNap's low power draw will operate these power supplies in the "Red" zone. RAILS ("RAILS") and Dynamic Load-Sharing ("Dynamic") both improve PSU performance because they increase average PSU load. RAILS outperforms all of the other options because its fine-grain sizing best matches PowerNap's requirements.

7. SHORTCOMINGS

PowerNap is able to provide near energy-proportional operation for workloads with characteristics similar to those we have studied (i.e., average service times near 100 ms). Though PowerNap is well suited for under-utilized services, there are a few potential shortcomings we enumerate in this section.

7.1 Multicore Servers

Current trends indicate that more and more cores will be integrated into a CPU. The workloads we analyzed were run on servers with only a few cores (i.e., 1–4). The model presented in Section 3 assumes a uniprocessor system; our queuing model predicts the performance for a single server (M/G/1) system. This queueing system is not appropriate for many-core servers, for which an M/G/k queue would be more appropriate. Unfortunately, it is not clear how to extend our model to an M/G/k system as Welch's derivation for exceptional first service [Welch 1964] does not apply; to date, the M/G/k variant remains analytically intractable.

However, even straight-forward analysis of multicore scaling suggests that PowerNap, or similar techniques that rely on full-system idleness, will grow increasingly difficult to apply if server software architectures do not change. Data center designers already find idle periods difficult to exploit with current multicore hardware [Weber 2010]. Current server workloads leverage multicore scaling through *weak scaling*, that is, they exploit additional cores by servicing additional, independent user

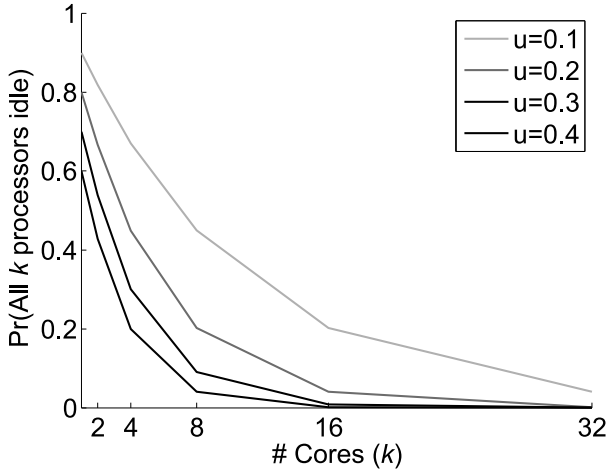


Fig. 14. M/M/k analysis of full-system idleness under weak scaling. Because idle periods do not align across cores, full system idleness rapidly vanishes. Smarter scheduling or new models of parallelism that enable strong scaling are needed to continue to exploit idleness.

requests. For example, if the number of cores doubles, roughly twice as much traffic can be directed to a single server, doubling throughput without increasing utilization. Unfortunately, because these requests are independent and their arrivals/completions are staggered, idle periods fail to align across cores, and PowerNap cannot be employed. Figure 14 illustrates this effect using a simple M/M/k analysis of a multicore server as the number of cores per socket is scaled. Even under only 10% utilization (u = 0.1), all cores in a 16-core system are concurrently idle less than 20% of the time.

To continue to gain the high-leverage power savings of PowerNap, we must either schedule jobs in an attempt to align idle periods, or rearchitect server software to leverage strong scaling. Prior work has proposed scheduling, using simple timeouts to control performance impact, to reduce the overhead of transitioning to/from idle low-power modes in single/dual-core CPUs [Amur et al. 2008; Elnozahy et al. 2003] and memory DIMMS [Pandey et al. 2006]. However, to recover a substantial fraction of idleness, these scheduling approaches require large delays which come at a steep response time penalty. Strong scaling, where multiple cores cooperate to reduce the latency of a single request, has the side-effect of aligning core busy and idle periods (if the parallelism is well-balanced), extending the applicability of PowerNap. However, rearchitecting services to leverage intra-request parallelism is challenging.

7.2 Highly Utilized Services

The PowerNap architecture provides excellent power savings with minimal latency penalty for lightly utilized servers. However, there are a few services that are highly utilized. Web search, rendering farms and batch processing, for example, have higher average utilization than the workloads we explore. Figure 7 shows that at high utilization PowerNap loses its advantage over throttling techniques such as DVFS. This change occurs because, as utilization approaches 100%, idle periods (and hence time spent in the nap state) become rare. Furthermore, at higher utilization DVFS will incur a significantly smaller latency penalty whereas PowerNap still incurs the same transition time. Therefore, we believe active low-power modes may be more appropriate for highly utilized services. For typical services where utilization is low, however, PowerNap remains a superior power management option.

Table VI. Power and Cost Comparison

	Web 2.0			Enterprise		
	Power	Efficiency	Power costs	Power	Efficiency	Power costs
Blade	6.4 kW	87%	$29k	6.6 kW	87%	$30k
PowerNap	1.9 kW	67%	$10k	2.6 kW	70%	$13k
PowerNap with RAILS	1.4 kW	86%	$6k	2.0 kW	86%	$9k

7.3 Reliability

It is important to consider that PowerNap may affect the reliability of server components. Because the system quickly transitions between power extremes, PowerNap may increase component wear. However, two insights suggest the increased stress may be limited. First, PowerNap does not rapidly modulate the operation of mechanical components. Fans and disks are both likely to suffer reduced lifetimes from frequent spin-up and spin-down; hence, PowerNap uses SSDs instead of disks and modulates fan speed independent of wake/nap transitions (in response to temperature instead). Second, the transition time we demand of most components is far longer than their designed capabilities. For instance, CPUs incur substantial power transitions (due to clock gating or HLT instructions) on the nano- and microsecond scale. PowerNap requires only millisecond-scale transitions. A rigorous study of the component-level reliability implications of PowerNap and RAILS is left to future work.

8. CONCLUSION

We presented *PowerNap*, a method for eliminating idle power in servers by quickly transitioning in and out of an ultra-low power state. We have constructed an analytic model to demonstrate that, for typical server workloads, PowerNap far exceeds DVFS's power savings potential with better response time. Because of PowerNap's unique power requirements, we introduced RAILS, a novel power delivery system that improves power conversion efficiency, provides graceful degradation in the event of PSU failures, and reduces costs.

To conclude, we present a projection of the effectiveness of PowerNap with RAILS in real commercial deployments. We construct our projections using the commercial high-density server utilization traces described in Table I. Table VI presents the power requirements, energy-conversion efficiency and total power costs for three server configurations: an unmodified, modern blade center such as the HP c7000; a PowerNap-enabled system with large, conventional PSUs ("PowerNap"); and PowerNap with RAILS. The power costs include the estimated purchase price of the power delivery system (conventional high-wattage PSUs or RAILS), 3-year power costs assuming California's commercial rate of 11.15 cents/kWh [U.S. Official Information Administration 2008], and a cooling burden of 0.5W per 1W of IT equipment [Moore et al. 2005].

PowerNap yields a striking reduction in average power relative to Blade of nearly 70% for Web 2.0 servers. Improving the power system with RAILS shaves another 26%. Our total power cost estimates demonstrate the true value of PowerNap with RAILS: our solution provides power cost reductions of nearly 80% for Web 2.0 servers and 70% for Enterprise IT.

ACKNOWLEDGMENTS

The authors would like to thank Partha Ranganathan and HP Labs for the real- world data center utilization traces, Andrew Caird and the staff at the Michigan Academic Computer Center for assistance in collecting the Cluster utilization trace, Laura Falk for assistance in collecting the departmental server utilization

traces, Mor Harchol-Balter for her input on our queuing models, and the anonymous reviewers for their feedback.

REFERENCES

AMUR, H., NATHUJI, R., GHOSH, M., SCHWAN, K., AND LEE, H.-H. S. 2008. IdlePower: Application-aware management of processor idle states. In *Proceedings of the Workshop on Managed Many-Core Systems (MMCS'08)*.

BARROSO, L. A., DEAN, J., AND HÖLZLE, U. 2003. Web search for a planet: The google cluster architecture. *IEEE Micro. 23*, 2, 22–28.

BARROSO, L. A. AND HÖLZLE, U. 2007. The case for energy-proportional computing. *Computer 40*, 12.

BASH, C. AND FORMAN, G. 2007. Cool job allocation: Measuring the power savings of placing jobs at cooling-efficient locations in the data center. In *Proceedings of the USENIX Annual Technical Conference (USENIX'07)*.

BOHRER, P., ELNOZAHY, E., KELLER, T., KISTLER, M., LEFURGY, C., AND RAJAMONY, R. 2002. The case for power management in web servers. In *Power Aware Computing*.

CHASE, J., ANDERSON, D., THAKAR, P., AND VAHDAT, A. 2001. Managing energy and server resources in hosting centers. In *Proceedings of the 18th Symposium on Operating Systems Principles (SOSP'01)*.

DELALUZ, V., KANDEMIR, M., VIJAYKRISHNAN, N., SIVASUBRAMANIAM, A., AND IRWIN, M. 2001. DRAM energy management using software and hardware directed power mode control. In *Proceedings of the 7th International Symposium on High-Performance Computer Architecture (HPCA'01)*. IEEE Computer Society, 0159.

DINIZ, B., GUEDES, D., WAGNER MEIRA, J., AND BIANCHINI, R. 2007. Limiting the power consumption of main memory. In *Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA'07)*. ACM, New York, 290–301.

ECOS AND EPR. 2008. Efficient power supplies for data center. Tech. rep., ECOS and EPR.

ELNOZAHY, E. N., KISTLER, M., AND RAJAMONY, R. 2003. Energy conservation policies for web servers. *Proceedings of the USENIX Symposium on Internet Technologies and Systems*.

FAN, X., WEBER, W.-D., AND BARROSO, L. A. 2007. Power provisioning for a warehouse-sized computer. In *Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA'07)*. ACM, New York, 13–23.

HÖLZLE, U. AND WEIHL, B. 2006. High-efficiency power supplies for home computers and servers. Tech. rep., Google.

HUANG, H., SHIN, K. G., LEFURGY, C., AND KELLER, T. 2005. Improving energy efficiency by making DRAM less randomly accessed. In *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED'05)*. ACM, New York, 393–398.

HYNIX. 2008. Hynix-DDR2-1Gb.http://www.hynix.com/datasheet/pdf/HY5PS1G4(8.16)31A(L)FP(Rev0.7).pdf.

INTEL. 2005. Intel Pentium M processor with 2-MB L2 cache and 533-MHz front side bus. http://download.intel.com/support/processors/mobile/pm/sb/30526202.pdf.

INTEL. 2007. Intel Pentium dual-core mobile processor. ftp://download.intel.com/design/mobile/datashts/31651903.pdf.

INTEL. 2008. Quad-core Intel Xeon processor 5400 series. http://www.intel.com/Assets/PDF/prodbrief/xeon-5400.pdf.

INTEL. 2009. Intel 64 and ia-32 architectures software developer's manual volume 3b: System programming guide. http://www.intel.com/Assets/ja_JP/PDF/manual/253668.pdf.

LAUDON, J. 2006. UltraSPARC T1: A 32-threaded CMP for servers. Invited talk.

LEBECK, A. R., FAN, X., ZENG, H., AND ELLIS, C. 2000. Power aware page allocation. *SIGOPS Oper. Syst. Rev. 34*, 5, 105–116.

LEFURGY, C., RAJAMANI, K., RAWSON, F., FELTER, W., KISTLER, M., AND KELLER, T. W. 2003. Energy management for commercial servers. *IEEE Computer 36*, 12.

LEFURGY, C., WANG, X., AND WARE, M. 2007. Server-level power control. In *Proceedings of the 4th IEEE International Conference on Autonomic Computing (ICAC'07)*.

LEIGH, K. AND RANGANATHAN, P. 2007. Blades as a general-purpose infrastructure for future system architectures: Challenges and solutions. Tech. rep. HPL-2006-182, HPLABS.

MEMCACHED 2010. Memcached - a distributed memory object caching system. http://www.memcached.org/.

MICRON. 2004. DDR2 SDRAM SODIMM. http://download.micron.com./pdf/datasheets/modules/ddr2/HTF16C128_256x64H.pdf.

MIYOSHI, A., LEFURGY, C., HENSBERGEN, E. V., RAJAMONY, R., AND RAJKUMAR, R. 2002. Critical power slope: Understanding the runtime effects of frequency scaling. In *Proceedings of the 16th Annual ACM International Conference on Supercomputing (ICS'02)*.

MOORE, J., CHASE, J., RANGANATHAN, P., AND SHARMA, R. 2005. Making scheduling 'cool': Temperature-aware workload placement in data centers. In *Proceedings of the USENIX Annual Technical Conference (USENIX'05)*.

NATIONAL SEMICONDUCTOR. 2002. Introduction to power supplies. Tech. rep. AN-556, National Semiconductor.

PADALA, P., ZHU, X., WANF, Z., SINGHAL, S., AND SHIN, K. 2007. Performance evaluation of virtualization technologies for server consolidation. Tech. rep. HPL-2007-59, HP/LABS.

PANDEY, V., JIANG, W., ZHOU, Y., AND BIANCHINI, R. 2006. DMA-aware memory energy management. In *Proceedings of the 12th International Symposium on High-Performance Computer Architecture*.

RASMUSSEN, N. 2007. AC vs. DC power distribution for data centers. Tech. rep. #63, American Power Conversion.

SAMSUNG. 2008. SSD SATA 3.0Gbps 2.5 data sheet. http://www.samsung.com/global/system/business/semiconductor/product/2008/6/19/82925725_datasheet.pdf.

SIDDHA, S., PALLIPADI, V., AND VEN, A. V. D. 2007. Getting maximum mileage out of tickless. In *Proceedings of the LINUX Conference (LINUX'07)*.

SMSC. 2008. LAN9420/LAN9420i single-chip ethernet controller with HP Auto-MDIX support and PCI interface. http://www.smsc.com/media/Downloads_Public/Data_Sheets/9420.pdf.

STANDARD PERFORMANCE EVALUATION CORPORATION. 2005. Specweb2005. http://www.spec.org/web2005/.

STANDARD PERFORMANCE EVALUATION CORPORATION. 2008. Specpower2008. http://www.spec.org/power_ssj2008/.

TOLIA, N., WANG, Z., MARWAH, M., BASH, C., RANGANATHAN, P., AND ZHU, X. 2008. Delivering energy proportionality with non energy-proportional systems – optimizing the ensemble. In *Proceedings of the USENIX Workshop on Power Aware Computing and Systems (HotPower'08)*.

U.S. EPA. 2007a. Energy Star computer specification v. 4.0. Tech. rep., US EPA.

U.S. EPA. 2007b. Report to congress on server and data center energy efficiency. Tech. rep., US EPA.

U.S. OFFICIAL INFORMATION ADMINISTRATION. 2008. Average retail price of electricity to ultimate customers by end-use sector, by state.

WEBER, W.-D. 2010. Energy-saving approaches for warehouse-scale computing. ISCAS Keynote. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'10)*.

WELCH, P. D. 1964. On a generalized M/G/1 queuing process in which the first customer of each busy period receives exceptional service. *Oper. Res. 12*, 736–752.

WU, Q., JUANG, P., MARTONOSI, M., PEH, L., AND CLARK, D. 2005. Formal control techniques for power-performance management. *IEEE MICRO 5*.

Received March 2010; revised November 2010; accepted December 2010