In cloud infrastructure management, understanding how CPU resources are allocated and billed can make the difference between a cost-effective deployment and hidden cloud waste. Terms like **CPU entitlement**, **sustained allocation**, and **shared core limits** often come up in cost reviews—but their implications vary significantly across cloud providers and service types. In this post, we'll unpack these concepts, explain why they matter for always-on small services, and highlight how you can use observability and trusted tools like AWS Compute Optimizer and Azure Advisor to make confident cost comparisons that reflect real-world usage—not averages.

CPU Entitlement: The Concept and Its Importance

CPU entitlement refers to the amount of CPU capacity that a virtual machine or container is allocated or guaranteed by a cloud provider within a given time frame. It's not simply the number of virtual CPUs (vCPUs) assigned to an instance but a measure of how much CPU time your workload actually receives and can sustain.

Why does this matter? Because cloud pricing and performance are tied closely to these guarantees:

- **Over-provisioning leads to waste:** Assigning instances with more vCPUs than your workload uses inflates costs unnecessarily.
- **Under-provisioning harms performance:** Choosing sizes without considering CPU entitlement can cause throttling, delays, and poor user experience.
- **Shared CPU environments add complexity:** Workloads that run on shared CPU cores face different bottlenecks and performance dynamics than dedicated CPU instances.

In essence, CPU entitlement helps you understand *how much CPU you're actually getting consistently*, beyond just what the instance type nominally offers.

Always-On Small Services and Hidden Cloud Waste

Many teams run **always-on small services**—for instance, monitoring agents, lightweight APIs, or background workers. These often consume a small slice of CPU but persist 24/7. If you pick instance sizes or configurations based https://computingforgeeks.com/shared-cpu-cloud-waste-migration-guide/ on average CPU usage alone, you risk locking in hidden overheads:

- *Unused capacity charging:* Providers charge for the full vCPU count regardless of how idle your service might be most of the time.
- *Ignoring CPU burst capacity behavior:* Small instances with shared CPUs rely on bursting mechanisms (CPU credits in AWS T series, for example) but those are often misunderstood or mismeasured.
- *Myth of idle efficiency:* Just because your CPU averages 10% usage over an hour doesn't mean you can safely pick a smaller instance—it might be spiking to 80% at P95 or P99.

Consequently, the difference between what you *think* you need and your CPU entitlement can mean substantial wasted spends.

Shared CPU Definitions Differ by Provider

The cloud providers categorize shared CPU and entitlement in different ways, which is crucial for cost and performance comparisons:

AWS

AWS offers shared-core instances (e.g., T3, T4g) where instances operate on physical cores shared among tenants but have a system of CPU credits that accumulate when idle and are spent during bursts. Their *CPU entitlement* relates heavily to available credits; sustained high CPU use without sufficient credits results in throttling.

AWS Compute Optimizer helps identify when your workload is either over-provisioned or under-provisioned relative to observed CPU utilization, but the calibration depends on measuring usage against entitlement, not just raw vCPU count.

Azure

Azure's burstable VM series (e.g., B-series) work similarly, but the metric and limits are different. The burst limit describes how long and how often you can exceed base CPU performance, with base levels guaranteed constantly. Azure Advisor analyzes these patterns and recommends VM sizes that better fit workload burstiness and entitlement.

Google Cloud

Google's shared-core offerings also use a baseline CPU allocation with bursting allowed up to 100% of a core. However, it's critical to distinguish between allocated CPU (what you pay for) and CPU usage, which may be constrained or throttled depending on load.

Each provider's unique definition of shared CPU and entitlement means you can't simply compare instance family types or vCPU counts at face value across clouds.

Measure Peaks with the Right Observation Window

One of the biggest mistakes in CPU entitlement analysis is focusing on average CPU percentages over long intervals—this masks critical short-term spikes that affect performance. To avoid this pitfall, you need to:

- **Align observation windows with workload characteristics:** If your workload experiences bursts of a few seconds or minutes, aggregate CPU data over those windows, not averages across hours.
- **Record high-frequency metrics:** Use sampling intervals of 1 minute or less to capture CPU spikes accurately.

For example, if an always-on small service is mostly idle but spikes multiple times per hour for 30 seconds at 80% CPU, a naïve hourly average might show only 5–10%. Choosing an instance sizing based on that average leads to under-provisioning or throttling that costs you in latency or retries.

Use Percentiles and Spike Duration, Not Averages

Percentiles like the P95 or P99 are more meaningful than averages to understand sustained allocation needs:

- **P95 (95th percentile):** Tells you the CPU utilization value below which 95% of the data points fall—captures regular peak usage.
- **P99 (99th percentile):** Represents near-maximum short-term spikes—critical for sizing to avoid throttling.
- **Spike Duration:** How long CPU stays above certain thresholds. Short spikes might be tolerable on burstable hosts; longer spikes may require larger instance sizes.

Tools like AWS Compute Optimizer integrate these percentile metrics into recommendations by analyzing multi-day baselines, which are more reflective of sustained CPU entitlement, rather than just average usage.

Similarly, Azure Advisor takes detailed workload telemetry to recommend VM sizes aligned with observed burst patterns and CPU credit consumption.

Putting It All Together: A Practical CPU Entitlement Cost Comparison Framework

1. **Collect high-resolution CPU metrics for multiple observation windows:** Include 1-minute, 5-minute, and 1-hour intervals. Calculate CPU percentiles like P50, P90, P95, and P99.
2. **Understand provider-specific shared CPU behavior:** Document bursting mechanisms, CPU credit allocations or consumption, and baseline entitlement guarantees for your instances.
3. **Correlate CPU usage to actual CPU entitlement:** For shared CPU instances, CPU usage above entitlement indicates throttling or credit depletion.
4. **Use AWS Compute Optimizer and Azure Advisor recommendations as guides:** But always verify their assumptions and custom tailor updates to your workload's percentile analysis.

5. **Validate cost impact:** Include related storage and network egress costs—not just instance hour costs—to avoid hand-wavy cost estimates.
6. **Define rollback criteria before changes:** E.g., if after resizing, P99 CPU spikes exceed entitlement or latency and errors increase.

Summary: CPU Entitlement Considerations by Provider Provider Shared CPU Model CPU Credit / Burst Mechanism Key Observation Metrics Recommended Tools AWS Shared physical cores in T, M burst families CPU credits accumulate and spend during bursts; throttling upon depletion CPU usage percentiles (P95, P99); CPU credit balance tracking AWS Compute Optimizer, CloudWatch Azure B-series burstable VMs with base and burst CPU levels Burst duration & limits; CPU credits influence short term performance Burst credit consumption; CPU performance relative to base Azure Advisor, Monitor Google Cloud Shared cores with baseline CPU allotment and bursting to 100% No explicit credit system, but usage throttled beyond baseline High-resolution CPU usage with spike duration Cloud Monitoring

Final Thoughts

CPU entitlement is a critical but often overlooked factor in cloud cost optimization efforts. For always-on small services, misinterpreting CPU usage data and relying on averages can hide significant cloud waste. Different cloud providers implement shared CPU and burst models in distinct ways, making cross-provider cost comparisons challenging without deep understanding.

The solution is not just to pick instance sizes based on nominal vCPUs but to measure real-world CPU entitlement using proper observation windows, percentile metrics, and workload-specific spike duration analyses. When combined with the guidance of tools like AWS Compute Optimizer and Azure Advisor—used critically and with rollback criteria in place—you can substantively reduce waste and improve application performance.

Before your next sizing or cost review cycle, ask:

What do the P95 and P99 CPU usage percentiles look like? How long do spikes last? Are we incorporating storage and egress costs into our comparison? And finally, how does our CPU entitlement differ between cloud providers?

Answering these questions will steer your team away from cloud waste traps and toward smarter infrastructure investments.