SMAC Strategy

Exploring the Frontier of AI, Cybersecurity and Cloud

Follow me on: Neeraj Kumar (LinkedIn) | **https://smacstrategy.com**

100+ Checklist for Cloud Cost Optimization

Cloud cost optimization is the practice of systematically managing and reducing cloud spending without sacrificing agility or performance. Inefficiencies like oversized resources, idle workloads, unmanaged SaaS etc. can quickly inflate cloud costs which can be controlled by some proven strategies. I have the reasons for cloud cost overrun and solutions in my blog: Cloud Cost Optimization: 10 Overrun Causes and 10 Solutions

Following 100+ checklists can help organization optimize their costs and get the strategic advantage of cloud.

Below Are The 100+ Controls For Cloud Cost Optimizations

✅ Real-Time Monitoring and Proactive Alerting

1. Use AI-powered anomaly detection for cloud spend.
2. Set up real-time alerts on sudden spend spikes.
3. Monitor API call volumes and costs.
4. Track cost anomalies by service, region, and SKU.
5. Set alerts on unexpected use of premium services.
6. Set up Slack/Teams integrations for real-time cost alerts.
7. Aggregate cost visibility across all cloud platforms.
8. Use multi-cloud management tools (e.g., CloudHealth, Apptio Cloudability).
9. Set up automated cost alerts and notifications.
10. Use Cloud Discovery Tools to find Idle tools
11. Implement organization-wide mandatory tagging policies.
12. Mandatory tagging at the time of resource creation.
13. Regularly audit untagged or mis-tagged resources.
14. Use centralized dashboards for cross-account cost visibility.
15. Consolidate billing for multi-account environments
16. Implement organization-wide mandatory tagging policies.
17. Mandatory tagging at the time of resource creation.
18. Regularly audit untagged or mis-tagged resources.
19. Use centralized dashboards for cross-account cost visibility.
20. Consolidate billing for multi-account environments

✅ Strong Governance and Policy Enforcement

21. Establish chargeback/showback models for departments.
22. Use free or trial tiers carefully to avoid unplanned costs; be aware of unexepcted or forgotten renewals
23. Establish a cross-functional FinOps team or Cloud COE.
24. Use IAM policies to control resource provisioning permissions.
25. Negotiate enterprise agreements centrally for discounts.
26. Avoid redundant services across different clouds.
27. Monitor and manage SaaS spend in addition to IaaS and PaaS.

28. Audit and investigate SaaS or marketplace charges.
29. Set KPIs for cost efficiency, anomaly resolution, and budget adherence.
30. Benchmark against industry peers for cloud efficiency.
31. Align cloud spending with business KPIs (e.g., cost per transaction, per customer).
32. Implement policy-as-code to enforce cloud cost governance. (e.g. OCI Policies, Open Policy Agent, Azure Policy, Organization Policy Service etc.)
33. Create and enforce budget limits at the team or project level.
34. Define approved service catalogs and instance types.
35. Use budgeting tools (e.g. OCI Budget Alerts (part of Cost Management), OCI Budgets & Alarms, GCP Cost Anomaly Detection, AWS Cost Anomaly Detection, AWS Budgets, GCP Budgets & Alerts, AWS Budgets etc.)
36. Perform monthly cost variance analysis.
37. Investigate and remediate unexpected large bills promptly.
38. Use cost forecasting tools (e.g. AI/ML based tools) for proactive budget planning.
39. Track cost by business unit, project, environment, team ownership and accountability
40. Use labels or tags to track cost by feature or service.

✅ Cloud Native Architecture for Optimal Cost

41. Refactor legacy monolithic apps to microservices. They are cost effective to scale.
42. Use containerized architectures. They increase the server utilization
43. Use serverless architectures. They may decrease the server runtime.
44. Use auto-scaling for workloads with variable demand.
45. Implement efficient DR/HA setups based on workload criticality to avoid unnecessary redundancy which increases cost.

✅ Wherever Possible Automation Over Manual

46. Infrastructure as a Code (e.g. Terraform scripts scripts) should be preferred above manual provisioning to avoid inaccuracies and enforce policies.
47. Turn off idle or low-usage resources automatically.
48. Enforce best practices in CI/CD pipelines (e.g., cost estimation with third party tools, terraform with mandatory tagging etc.).
49. Automate resource lifecycle and decommissioning. (e.g. expired data deleted)
50. Use configuration management tools to standardize deployments.

✅ Compute Resource Right-Sizing and Optimization

51. Right-size VMs and databases regularly based on utilization data.
52. Implement predictive scaling for seasonal workloads.
53. Schedule shutdowns for dev/test/QA environments outside business hours.
54. Review and clean up idle databases.
55. Optimize database storage and indexes regularly to reduces data size, improves query efficiency, and lower storage and I/O costs
56. Use Edge and CDN services to reduce compute and bandwidth needs.
57. Optimize cloud licensing strategies (e.g., bring-your-own-license).
58. Use spot instances for fault-tolerant and non-critical workloads.
59. Use Convertible RIs if workloads need flexibility.
60. Monitor and adjust existing commitments regularly for reserved instances or saving plans
61. Leverage spot and preemptible instances for non-critical jobs.
62. Leverage spot and preemptible instances for non-critical jobs.
63. Use Reserved Capacity or Committed Use Discounts (Reserved Instances or Savings Plans) for predictable usage.
64. Avoid on-demand pricing for long-running workloads.
65. Try to leverage Bring your own license during migration whereever possible

66. Migrate to serverless for event-driven or intermittent workloads.

✅ Storage and Data Lifecycle Optimization

67. Enforce limits on log retention periods.
68. Use storage reservations for consistent storage use.
69. Decommission orphaned volumes,and snapshots.
70. Apply data lifecycle management policies for storage tiering.
71. Move cold data to archival storage
72. Compress data before storing to save space.
73. Deduplicate data to avoid redundant storage.
74. Delete unnecessary snapshots, backups, and logs.
75. Use intelligent-tiering storage classes where applicable.
76. Use object versioning only when necessary.
77. Implement backup and retention policies based on data value.
78. Regularly review observability/logging configurations to reduce unnecessary log retention.

✅ Network and Data Transfer Optimization

79. Design apps to minimize data transfer between regions or clouds reducing data egress and inter-region transfer costs.
80. Minimize inter-region and cross-cloud data transfers.
81. Use direct connect, interconnect, or ExpressRoute for efficient connectivity.
82. Decommission orphaned IPs
83. Review and clean up idle load balancers and network interfaces.
84. Use CDN caching to minimize repeated data transfers.
85. Optimize CDN cache control and content delivery rules to reduce egress.

✅ Serverless and Function Optimization

86. Right-size memory allocations by matching memory to actual needs
87. Benchmark and optimize execution time to lower duration costs by improving code efficiency
88. Use asynchronous patterns and caching to minimize idle wait time and repeated executions
89. Set concurrency limits to prevent sudden burst costs from uncontrolled concurrency
90. Throttle at API Gateway or event source to control invocation rate to avoid event storms
91. Reducing cold starts to lowers latency and avoids extra costs from provisioned concurrency
92. Audit event triggers and sources to prevent unnecessary or recursive invocations
93. Implement event filters to reduce unnecessary invocations
94. Optimize API Gateway configurations to switch to lower-cost HTTP APIs where possible
95. Monitor and tune log retention settings to reduce logging and observability tool costs
96. Consolidate function logic to reduce the number of function calls and simplify architecture
97. Use cost dashboards for serverless services to enable visibility and accountability for serverless costs
98. Tag functions and allocate costs to teams/projects to drive cost ownership and transparency
99. Use batch processing where feasible to aggregate requests to reduce invocation overhead

✅ Container Workload Optimization

100. Use container rightsizing tools (e.g., Kubecost, CAST AI).
101. Optimize Kubernetes autoscaling policies (HPA, VPA, Cluster Autoscaler).
102. Use bin-packing techniques to maximize node utilization in Kubernetes.
103. Automate idle cluster and node management in Kubernetes.

✅ Unified Multi-Cloud Cost Management

104. Use unified dashboards for multi-cloud reporting.
105. Use multi-cloud dashboards and reporting tools to centralize cost visibility across clouds and avoid blind spots.
106. Enforce consistent tagging standards across clouds to Ensure cross-cloud cost allocation and accountability.
107. Minimize cross-cloud data transfers to avoid unnecessary egress and inter-cloud data movement fees.
108. Avoid service redundancies across clouds to prevent paying twice for the same type of services in different clouds.
109. Use cloud-agnostic management platforms to standardize governance, policy enforcement, and automation across clouds.
110. Monitor SaaS and marketplace usage across clouds to Identify and manage hidden or duplicate SaaS costs.
111. Use cloud-native anomaly detection tools per cloud to enable cost anomaly detection in each cloud (AWS, Azure, GCP, OCI).
112. Implement FinOps practices centrally for all clouds; Establish a unified FinOps team and processes across the organization.

✅ Cultural Shift Towards FinOps and Cost Awareness

113. Educate engineering and finance teams on cloud cost drivers.
114. Run regular FinOps workshops and knowledge sharing sessions.
115. Foster a culture of continuous improvement in cost optimization.
116. Continuously educate new hires on cloud cost best practices as part of onboarding.
117. Educate developers on serverless pricing models to promote cost-aware development practices

Share with Developers, FinOps Enthusiast, Cloud, Technology and Business Leaders!!

May Your Cloud Be Flexible and Inexpensive !!!

For More such insights on AI, Cybersecurity and Cloud follow me on:

LinkedIn: https://www.linkedin.com/in/neerajkmr47/

Website: https://smacstrategy.com

NEERAJ KUMAR
smacstrategy.com
linkedin.com/in/neerajkmr47/