CloudYali

AWS Cost Optimization Checklist

30+ actionable items to reduce your AWS bill by 15-60%

> This checklist covers the highest-impact AWS cost optimizations based on $250M+ in analyzed cloud spend. Each item includes typical savings and how to identify the opportunity in your environment.

Quick Wins: Start Here

These optimizations take less than an hour and deliver immediate savings.

Optimization	Typical Savings	Effort
GP2 → GP3 migration	~20%	Low
io1/io2 → GP3 migration	~60-87%	Low
S3 Intelligent-Tiering	Up to 65%	Low
DynamoDB Standard-IA	~60%	Low
CloudFront compression	~65-80%	Low
Delete orphaned EBS volumes	100%	Low
Stop dev instances off-hours	~65%	Medium
Delete duplicate CloudTrail trails	100%	Low
Delete idle NAT Gateways	100%	Low
Disable unused FSR	100%	Low

1. Elastic Block Store (EBS)

☐ Migrate GP2 volumes to GP3

~20% savings on storage costs

How to spot: Run `aws ec2 describe-volumes --filters Name=volume-type,Values=gp2` — any results are costing you 20% more than necessary.

☐ Migrate io1/io2 volumes to GP3 where IOPS allow

~60-87% savings per volume

How to spot: Check CloudWatch for actual IOPS usage. If peak IOPS < 16,000, GP3 can handle it at a fraction of the cost.

☐ Delete unattached EBS volumes

100% of orphaned volume costs

How to spot: Filter volumes by "available" state in the console. These are attached to nothing and burning money.

☐ Delete old snapshots (90+ days)

$0.05/GB-month

How to spot: Sort snapshots by creation date. Anything older than your retention policy is waste.

☐ Disable unused Fast Snapshot Restore (FSR)

Hidden hourly charges

How to spot: Check each snapshot's FSR settings. If you're not using instant restores, you're paying for nothing.

2. Simple Storage Service (S3)

☐ Enable S3 Intelligent-Tiering on buckets with unknown access patterns

Up to 65% savings on infrequently accessed data

How to spot: Use S3 Storage Lens to identify buckets with mixed or declining access patterns.

☐ Set lifecycle policies to expire old object versions

Varies by versioning depth

How to spot: Check bucket versioning settings. Old versions pile up silently — we've seen buckets where versions were 10x the size of current objects.

☐ Delete incomplete multipart uploads

100% of partial upload costs

How to spot: Use S3 Storage Lens or set a lifecycle rule to auto-expire incomplete uploads after 7 days.

3. Elastic Compute Cloud (EC2)

☐ Right-size underutilized instances

~20-50% savings

How to spot: Check CloudWatch CPU and memory metrics. Consistent utilization below 40% means you're paying for capacity you don't use.

☐ Use Graviton (ARM) instances where compatible

Up to ~40% better price-performance

How to spot: Any Linux workload without x86-specific dependencies is a candidate. Start with non-production.

☐ Migrate previous-gen instances to current-gen

~10-30% better performance/cost

How to spot: Look for instance types like m4, c4, r4, t2. Current gen costs less and performs better.

☐ Stop idle dev/test instances off-hours

~65% savings

How to spot: Tag instances by environment. If CPU is near-zero evenings and weekends, automate stop/start schedules.

4. Networking & Content Delivery

☐ Enable CloudFront compression for text assets

~65-80% bandwidth reduction

How to spot: Check your CloudFront distribution settings. If "Compress Objects Automatically" is off, you're transferring bloated files.

☐ Set up VPC endpoints for AWS service traffic

Eliminates NAT Gateway data processing fees

How to spot: High NAT Gateway charges in Cost Explorer. S3 and DynamoDB endpoints are free and cut transfer costs immediately.

☐ Remove idle NAT Gateways

100% of idle NAT costs

How to spot: Check NAT Gateway metrics for BytesOutToDestination. Near-zero over 30+ days means it's unused.

☐ Delete unused Elastic IPs

$3.60/month per idle EIP

How to spot: Filter EIPs by association status. Unassociated = wasted money.

☐ Delete idle load balancers

$16-22/month base cost per ALB/NLB

How to spot: Check RequestCount and ActiveConnectionCount. Zero traffic for 30+ days = candidate for deletion.

5. Database Services

☐ Migrate RDS io1/io2 storage to GP3

~60-87% storage cost reduction

How to spot: Same logic as EBS, check actual IOPS in CloudWatch vs. provisioned.

☐ Right-size over-provisioned RDS instances

~20-50% savings

How to spot: Check CPU and FreeableMemory in CloudWatch. Consistent low utilization means you're paying for unused capacity.

☐ Delete idle RDS/Aurora clusters

100% of cluster costs

How to spot: Check DatabaseConnections metric. Zero connections over weeks means a forgotten database burning money.

☐ Enable DynamoDB Standard-IA for cold tables

~60% storage savings

How to spot: Check table metrics for read/write patterns. Tables accessed less than 20% of the time are candidates.

☐ Switch to Aurora I/O-Optimized for I/O-heavy workloads

~30-40% cost reduction

How to spot: If I/O charges exceed 25% of your Aurora bill, I/O-Optimized pricing likely saves money.

6. Lambda & Serverless

☐ Right-size Lambda memory allocation

~15-40% runtime cost reduction

How to spot: Use AWS Lambda Power Tuning tool. Many functions run faster AND cheaper with different memory settings.

☐ Use Graviton (ARM) for Lambda functions

Up to ~34% better price-performance

How to spot: Any function without x86-specific dependencies. Change architecture in function config — one setting.

☐ Remove unused Provisioned Concurrency

$0.015/GB-hour when idle

How to spot: Check ProvisionedConcurrencyUtilization metric. Low utilization means you're paying for warm capacity you don't use.

7. Monitoring & Governance

☐ Delete duplicate CloudTrail trails

100% of duplicate trail costs

How to spot: List trails across all regions. Multiple trails logging the same events to different buckets is common and wasteful.

☐ Optimize CloudWatch log retention

Varies by volume

How to spot: Check log group retention settings. The default is "never expire" but most logs lose value after 30-90 days.

☐ Delete unused CloudWatch alarms

$0.10/alarm/month

How to spot: Check alarm state history. Alarms that haven't changed state in months are monitoring nothing useful.

8. Commonly Overlooked Services

☐ Delete idle OpenSearch clusters

100% of idle cluster costs

How to spot: Check indexing rate and search rate metrics. Near-zero activity over weeks means an unused cluster.

☐ Stop idle SageMaker notebooks

100% of idle notebook costs

How to spot: Notebooks running for days with no kernel activity. These are often forgotten after experiments.

☐ Delete idle QuickSight users

$18-24/user/month

How to spot: Check user activity in QuickSight admin. Admin Users who haven't logged in for 30+ days are wasting licenses.

Implementation Strategy

1. **Start with quick wins** — GP2 to GP3, enable compression, delete orphaned resources
2. **Build monitoring** — Set up CloudYali Cost Reports and budgets before making changes
3. **Tag everything** — Good tagging enables good cost allocation and easier optimization
4. **Automate where possible** — Scheduled instance stops, lifecycle policies, automated rightsizing
5. **Review regularly** — Cloud environments change; make cost review a monthly habit

---

The Bottom Line

These 30+ optimizations can reduce your AWS bill by 15-60%, depending on your current state. Most organizations achieve 20-30% savings within the first few months of focused optimization efforts.

Get more cloud cost optimization tips at **cloudyali.io/blogs**