

AWS COST · SAMPLE DELIVERABLE

Architecture Changes and Guardrails

Week 5 of 6: Architecture and guardrails · prepared for
Seatlark, Inc.

Prepared by Charles Sieg · charles.sieg@vantalect.com · vantalect.com

June 12, 2026 · AWS Cost Analysis & Optimization: The Optimization (six weeks)

SAMPLE DELIVERABLE

This is a sample deliverable. Seatlark, Inc. is a fictional company: every account, resource, finding, figure, and person in this report is simulated to show the form and depth of a Vantalect engagement. AWS prices are list prices from the AWS Price List API retrieved on October 13 and 16, 2026, and AWS Business Support+ pricing is from the AWS Support pricing page on October 16, 2026.

Contents

- Summary 2
- What happened this week 4
 - S3–3 was reversed..... 5
- The guardrail register 6
 - Guardrail ownership..... 7
- Budgets and anomaly monitors 7
- Tag policy: four keys 9
- Storage defaults: a Config rule and an SCP 10
- Instance Scheduler as the non–production default 10
- The wave runbook 11
 - Sizing: use over time, not allocation 11
 - MGN cleanup: finalize, disconnect, archive 12
 - The checklist 13
- Cost avoidance on the remaining waves..... 14
- What next week does 16
- The exact mechanism..... 17
 - Budgets and anomaly monitors 17
 - The tag policy..... 18
 - The storage rule and the SCP 20
 - Instance Scheduler..... 21
 - Log–group retention 22
 - The VPC template 23
 - The sizing sheet 24
 - MGN cleanup 25

Summary

Week 5 counted no savings, and that is by design. The production freeze ran from June 4 to June 14, 2026, so no production change went in this week, and the 20 changes in place landed by June 3, 2026. They are estimated at **\$155,076 a year** and are measured on your Cost and Usage Report

next week. This week built what keeps the bill down and what lands the remaining waves right: **10 guardrails**, each with an owner and a pull request ([The guardrail register](#)), and a wave runbook ([The wave runbook](#)).

The guardrails matter because the next five waves add \$71,000 a month to your plan, and the patterns that found this engagement’s savings are the ones that will recur in them: allocation-sized instances, gp2 and io1 volumes, no log retention, three NAT gateways in every VPC, and migration servers nobody retired. We priced what the guardrails and the runbook should prevent as **\$100,000 to \$170,000 a year** at the end-state run rate ([Cost avoidance on the remaining waves](#)). That figure is **not measured, not counted toward the gate, and not a saving on the bill**. There is no "before" on the Cost and Usage Report for a workload that has not moved, so nothing in this report adds it to a saving, a total, or a multiple.

Two things happened this week that change the arithmetic of the guarantee, and both are in this report. Diego Saldana’s team reversed S3–3 because white-label partners hotlink rendition URLs on archived event pages ([S3–3 was reversed](#)); it was approved at the readout, so it counts at its \$3,109 estimate, and it is the only reason the counted figure will exceed the measured one. And wave 6 cut over on May 30, 2026, which raised the bill by \$2,600 a month (5.0% of the May bill); that is growth, not a saving, and the migration plan priced it. Wave 6 was lifted before the runbook existed, so it is the last wave that carries allocation-sized servers by default. Next week’s measurement is per change, so the rise neither hides nor inflates any measured saving.

Week 5 in numbers. The last row is a model of the future, not a result, and is not part of any total.

Week 5 at a glance	Result
Production changes this week	0
Savings counted this week	\$0
Changes in place, all landed by Jun 3 (estimated, measured in week 6)	20 (\$155,076 a year)
Guardrails built and merged	10
Modeled avoidance on waves 7 to 11, a year (not measured, not counted toward the gate, and not a saving on the bill)	\$100,000 to \$170,000

Decision for the sponsor

Renee Castellano: make the wave runbook's two steps, the sizing sheet and the MGN cleanup, required attachments on every cutover change ticket from wave 7 (July to August 2026). It is a process decision, not a purchase, and it is the largest lever in the avoidance model ([the levers table](#)). Nothing else in this report needs your call. Once you say yes, Owen Pradhan owns the sizing step and Beth Morrow owns the cleanup step ([Guardrail ownership](#)). Your decision is recorded in the handover report.

What happened this week

The week ran June 8 to June 12, inside the freeze. The freeze exists to protect the summer on-sales, so the rule for the week was simple: build everything that waits for no window, and leave production alone.

The week—one asks, answered. Both asks from the baseline report that were still open at the readout have an answer, and the sponsor's decision is the one item waiting on you.

- **Confirm which reading of the \$1.5 million is right (assumption A1).** Lydia Brandt answered on June 9, 2026: the \$1.5 million a year is the end-state annual run rate, the first reading, so the plan's end state stands and the other reading is a sensitivity only.
- **Send the colo allocation for each landed workload.** Lydia Brandt answered on June 2, 2026: the business case allocates the data center's cost to waves 6 to 11 (the plan's wave figures) and to nothing before wave 6: it holds no allocation for any single workload, so the by-workload comparison stays AWS-only and the by-wave comparison starts with wave 6.
- **The sponsor's decision** on the runbook's two required steps is the box at the top of this report. It was asked on June 12, 2026.

Allowed during the freeze:

- Budgets, anomaly monitors, and Config rules (they observe and change nothing a workload runs).
- Anything in a non-production or sandbox account.
- Runbooks, templates, and pull requests that wait for a window.

Not allowed during the freeze:

- Any change to what a production workload runs, how it launches, or how it is tagged.

- A policy that could refuse a production launch during an on-sale.

Dates around week 5. A change needs seven full days of finalized data after it, so only changes made by June 8, 2026 can be measured on June 17, 2026; every counted change was in place by June 3, 2026.

Date	What happened	Counts toward the gate
May 30, 2026	Wave 6 cut over: partner reporting and the affiliate feed generator. The bill rose by \$2,600 a month	No: growth
June 2, 2026	R-1 and R-2, the production rightsizing, went in before the freeze	Yes, measured in week 6
June 3, 2026	C-1, the Compute Savings Plan, was bought in the management account. This was the last counted change	Yes, measured by its ARN
Jun 4 to Jun 14	Production freeze	
Jun 8 to Jun 12	The 10 guardrails merged; S3-3 reversed	No: nothing new in place
June 15, 2026	The first day after the freeze: the tag policy reaches the production accounts	
June 22, 2026	E-2b, the production search volumes from io1 to gp3, is scheduled	Not counted until it is in place and measured
June 23, 2026	The volume-type SCP reaches the production accounts	

S3-3 was reversed

S3-3, deleting 11 TB of regenerable image renditions for past events, was approved at the readout and is now declined by its owner, Diego Saldana, Director, Partner Engineering. His team found that white-label partners hotlink rendition URLs on archived event pages: deleting the renditions would break pages on some of the 140 partner sites. The renditions stay where they are, in S3 Standard.

The guarantee's terms cover this case: a change the client approved at the week-two readout and later declines "counts at the savings the roadmap estimated for it." So S3-3 counts at its estimate of \$3,109 a year and contributes \$0 to the measured figure, because it was never made. It is the only reason the counted figure exceeds the measured one, and next week's gate table shows both. The savings reopen if the partner pages stop linking to rendition URLs.

The guardrail register

The guardrail register, as of June 12, 2026. A guardrail is a small piece of automation with an owner that catches the expensive thing on the first day instead of the first quarterly review. All 10 are pull requests in seatLark-landing-zone; what each one does is in the mechanism section.

ID	Guardrail	Owner	Pull request, merged	Status	How to undo it
GR-1	A budget per account	Lydia Brandt	PR #419, Jun 8	Live in all 16 accounts	Delete the budget resources
GR-2	Cost Anomaly Detection per account	Lydia Brandt	PR #420, Jun 9	Live: 16 account monitors and 1 service monitor	Delete the monitor resources
GR-3	Tag policy: Team, Workload, Environment, MigrationWave	Beth Morrow	PR #421, Jun 9	Attached to 5 non-production and sandbox accounts; production on June 15	Detach the policy
GR-4	Config rule: no new gp2 or io1 volumes without an exception tag	Beth Morrow	PR #422, Jun 10	Live in all 16 accounts; detective only	Delete the rule
GR-5	SCP: refuse gp2 and io1 volumes without an exception tag	Owen Pradhan	PR #423, Jun 11	Attached to 5 non-production and sandbox accounts; production on June 23, after E-2b	Detach the policy
GR-6	Instance Scheduler as the non-production default	Beth Morrow	PR #424, Jun 11	Live in the 5 non-production and sandbox accounts	Remove the tag
GR-7	Log-group retention in the account baseline	Beth Morrow	PR #425, Jun 11	Live in all 16 accounts; new log groups	Delete the rule and function
GR-8	The VPC template	Owen Pradhan	PR #426, Jun 12	Merged; applies to every new VPC	Revert the pull request
GR-9	Wave runbook: size from use, not allocation	Owen Pradhan	PR #427, Jun 12	Adopted for wave 7, pending the sponsor's decision	Revise the runbook
GR-10	Wave runbook: MGN cleanup	Beth Morrow	PR #428, Jun 12	Adopted for wave 7, pending the sponsor's decision	Reinstall the replication agent

Guardrail ownership

A guardrail with no owner decays. Each one has a single owner who reads the alert, decides whether the cause is expected, and either fixes it or changes the threshold with a reason in the pull request. The same people attend the monthly cost review in the handover report.

- **Lydia Brandt, Vice President, Finance** (GR-1, GR-2). Sets and revisits budget limits and anomaly thresholds at the monthly review, and receives the immediate alerts.
- **Owen Pradhan, Vice President, Infrastructure** (GR-5, GR-8, GR-9). Owns the volume-type SCP, the VPC template, and the runbook's sizing step, and approves every exception tag.
- **Beth Morrow, Manager, Site Reliability Engineering** (GR-3, GR-4, GR-6, GR-7, GR-10). Maintains the tag policy, the storage rule, the scheduler, and the retention rule in the baseline, owns the runbook's MGN cleanup step, and reads the compliance reports.

The account owners in the budget tables ([below](#)) answer the alerts for their own accounts.

Budgets and anomaly monitors

Every account has a budget and an anomaly monitor, and every alert goes to the person who owns the account. A budget is the account's monthly cost with today's changes in place, plus 10%, rounded up to \$100. It alerts at 80% and 100% of actual cost and at 100% of forecast. The budgets use amortized cost, so the Savings Plan's benefit lands in the accounts that use it. The savings measurement is a different basis: it uses public On-Demand cost, so that the plan's discount is not counted twice (the measurement report explains why). Wave 6's \$2,600 a month is in the limit for partner-prod, where partner reporting runs.

One budget per workload account. Cost today is each account's monthly cost at list prices with the changes in place, before the Savings Plan. Wave 7 raises limits at cutover ([the checklist](#)).

Workload account	Owner	Cost today	Monthly budget
marketplace-prod	Aaron Feld	\$13,072	\$14,400
partner-prod	Diego Saldana	\$5,921	\$6,600
marketplace-staging	Aaron Feld	\$2,189	\$2,500
sandbox-eng	Beth Morrow	\$1,281	\$1,500
content-prod	Kara Lindqvist	\$1,152	\$1,300
marketplace-dev	Aaron Feld	\$1,037	\$1,200
data-prod	Sanjay Iyer	\$941	\$1,100
partner-nonprod	Diego Saldana	\$586	\$700
perf-test	Aaron Feld	\$384	\$500
Subtotal, 9 accounts		\$26,563	\$29,800

One budget per platform account. The management account's cost includes AWS Business Support+. All 16 budgets add up to \$46,900 a month.

Platform account	Owner	Cost today	Monthly budget
sl-network	Owen Pradhan	\$7,712	\$8,500
sl-management	Owen Pradhan	\$2,754	\$3,100
sl-shared-services	Beth Morrow	\$2,421	\$2,700
sl-backup	Beth Morrow	\$798	\$900
sl-migration	Owen Pradhan	\$644	\$800
sl-log-archive	Beth Morrow	\$509	\$600
sl-security-tooling	Beth Morrow	\$440	\$500
Subtotal, 7 accounts		\$15,277	\$17,100

The anomaly monitors are one per account plus one by service. Each owner gets a daily summary of anomalies with an impact of \$100 or more, and the on-call topic gets an immediate alert at \$1,000, about 1.9% of the monthly bill. The code is in [Budgets and anomaly monitors](#).

Tag policy: four keys

Budgets and monitors route by account, but the bill is also cut by team, workload, and migration wave, and only a tag does that. The tag policy fixes four keys: Team, Workload, Environment, and MigrationWave. It fixes the spelling of all four and the allowed values of three. A tag policy checks a tag when someone applies it; it does not require the tag to exist, which is why the SCP and the runbook carry the requirement ([The tag policy](#)).

The tag keys and their values, from the resources in the estate today.

Key	Allowed values	Why
Team	data-etl, fulfillment-content, inventory-ingest, inventory-search, marketplace-api, marketplace-web, partner-api, platform-sre	Showback and the budget owner's first question: whose is this
Workload	Free text, key enforced. Today: bastion, ci-runners, connect-affiliate-api, etl-workers, exchange-api, exchange-web, listing-ingestion, load-generation, sandbox, search-cluster, sessions, tooling	Unit costs per workload in the monthly review
Environment	management, nonprod, prod, sandbox, security, shared	The scheduler, the SCP scope, and the Multi-AZ rule
MigrationWave	wave-0, wave-1, wave-2, wave-3, wave-4, wave-5, wave-6, wave-7, wave-8, wave-9, wave-10, wave-11, none	The cost of each wave against its plan

The policy is attached to the 5 non-production and sandbox accounts now and to the 4 production accounts on June 15, 2026. Enforcement can refuse an out-of-policy tag operation, and a refused tag on a production Auto Scaling group during an on-sale is exactly the surprise the freeze rules out.

Storage defaults: a Config rule and an SCP

Every volume the migration created was gp2, and the search cluster's production volumes are io1, matched to the colo SAN. E-1 and E-2a fixed what existed. These two guardrails stop the next wave from recreating it.

- **The Config rule (GR-4) is the detective half.** It marks any gp2 or io1 volume without the `StorageException` tag non-compliant, in every account, and changes nothing. Today it lists 3 volumes: `vol-mkt-search-01-data`, `vol-mkt-search-02-data`, `vol-mkt-search-03-data`, the production search cluster's io1 volumes that E-2b (\$10,800 a year) converts on June 22, 2026.
- **The SCP (GR-5) is the preventive half.** It refuses to create a gp2 or io1 volume unless the request carries the exception tag. It is attached to the 5 non-production and sandbox accounts now, and to the production accounts on June 23, 2026, the day after E-2b, so it cannot refuse a production launch during the freeze.
- **Why both.** A service control policy does not apply to service-linked roles, so an Auto Scaling group launching from an old launch template is not stopped by the SCP. The Config rule sees that volume within minutes, and its owner hears about it the same day.

The exception tag is a request for a decision, not a loophole: Owen Pradhan approves each one, and the monthly review lists them. A volume that needs more than gp3's 80,000 IOPS or 2,000 MiB/s is an io2 Block Express decision made in the wave's design review. The SCP does not block io2.

Instance Scheduler as the non-production default

S-1 put a weekday schedule on the development, partner non-production, and perf-test hosts. This week it became the default instead of a one-time change. The default schedule is `weekday-12h: 12 hours, Mon-Fri, 07:00 to 19:00 America/New_York` (the scheduler's period ends at 18:59, the last minute before 19:00). It is the schedule S-1 defined, against the Instance Scheduler hub stack, and this week the remote stack that reads it went into all 5 non-production and sandbox accounts, with a managed Config rule that marks any instance there without a `Schedule` tag.

18 instances in those accounts are schedulable today. The perf-test fleet is the exception to the weekday window: it sits at zero between rehearsals and scales out for each one. The estimate assumed 96 hours a month of rehearsals, and the measurement report shows what June actually

used. Staging is in the five accounts but its hosts carry the tag value none: S-2 was declined, because staging runs the overnight on-sale rehearsals and partner acceptance tests. Any owner can opt out the same way, with a line in the monthly review.

The wave runbook

The runbook is a checklist every remaining wave follows, in seatlark-landing-zone. Its two largest steps came straight from this engagement: size from use, and clean up what the migration tool leaves behind.

Sizing: use over time, not allocation

MGN launches every server at the size of its VMware allocation, so eight vCPUs and 32 GiB become an m5.2xlarge whether the server uses them or not. The runbook replaces that with a sizing sheet built from 30 days of vCenter statistics: the 95th-percentile CPU and memory use, and the busiest two hours in the window. The rule for the target size is short. Halve the instance only if, on the smaller size, the 95th percentile stays at or under 60% of CPU and the busiest hours stay under 70%.

The rule applied to the marketplace API tier (R-1) and the web tier. The web tier’s 41% is its April on-sale peak; the smaller size would run it at 82%, which is why it keeps its size.

Tier	Candidate size	CPU at p95 on it	CPU at theFits busiest hour on it	Result
Marketplace API: 6 x m5.2xlarge (8 vCPUs), p95 9%, busiest 29%	m5.large	36%	116%no	Too hot
Marketplace API: 6 x m5.2xlarge (8 vCPUs), p95 9%, busiest 29%	m5.xlarge	18%	58%yes	Halve to m5.xlarge
Marketplace API: 6 x m5.2xlarge (8 vCPUs), p95 9%, busiest 29%	m5.2xlarge	9%	29%yes	Fits, but a smaller size fits
Marketplace web: 6 x m5.2xlarge (8 vCPUs), p95 41%, busiest 41%	m5.large	164%	164%no	Too hot
Marketplace web: 6 x m5.2xlarge (8 vCPUs), p95 41%, busiest 41%	m5.xlarge	82%	82%no	Too hot
Marketplace web: 6 x m5.2xlarge (8 vCPUs), p95 41%, busiest 41%	m5.2xlarge	41%	41%yes	Keep m5.2xlarge

The sheet comes from a short PowerCLI script, in [The sizing sheet](#). vCenter keeps its 30-day history as two-hour averages, so a short spike inside one of them is invisible. Setting the busiest-hour limit below 100% allows for that on a back-office server; it is not enough for a tier customers use during an on-sale. Compute Optimizer's default 14-day window is not a substitute: it missed the April on-sale entirely, which is why R-3, the ingestion workers, is still a load-test question.

Customer-facing tiers need more than the sheet. A customer-facing tier is sized only with on-sale peak data (the busiest on-sale window at one-minute resolution) or a load-test result on the candidate size, attached to the cutover ticket; the sizing sheet alone never sizes it. This covers the web and API tiers above and, in wave 7, orders and checkout, which is 31% of the remaining planned cost and the largest share of the avoidance model. The runbook does not size wave 7's tiers from a vCenter average, whatever the sheet shows.

MGN cleanup: finalize, disconnect, archive

W-2 found 14 replication servers and their staging disks still running for source servers whose waves had finished months earlier, \$661 a month and the most common lift-and-shift leak. The runbook ends every wave with three actions, and a check:

1. **Finalize the cutover** once the application owner signs off, which moves the source server to the cutover state.
2. **After a 14-day hold, disconnect** each source server from the service. Replication stops and the service terminates the replication resources it created.
3. **Mark it archived**, so it no longer appears in the active list.
4. **Check the migration account.** The instances and volumes left in it must match the active source servers, and nothing else.

The Cost and Usage Report shows no AWS Application Migration Service source-server charge for these servers in May: MGN bills a source server by the hour only while it replicates, and none of the 14 has replicated in 30 or more days (their colo sources were powered off after cutover). What still bills is the replication servers and their staging disks, which W-2 removes. The commands are in [MGN cleanup](#).

The checklist

The wave runbook's checklist. Every step names the finding it answers and the guardrail that backs it, so a skipped step is caught by something other than memory.

When	Step and the evidence it leaves	Answers	Owner
Before launch	Size from 30 days of vCenter use: 95th-percentile CPU and memory, plus the busiest hour. Evidence: sizing sheet attached to the cutover ticket.	R-1, R-2, R-3, R-4; GR-9	Wave lead
Before launch	For a customer-facing tier, attach the busiest on-sale window at one-minute resolution or a load-test result on the candidate size; a two-hour vCenter average never sizes it. Evidence: peak data or load-test run attached to the cutover ticket.	R-1, R-3; GR-9	Wave lead
Before launch	Set every volume to gp3 with explicit IOPS and throughput in the launch template; io2 Block Express only through the design review. Evidence: launch template reviewed in the pull request.	E-1, E-2a, E-2b; GR-4, GR-5	Wave lead
Before launch	Tag the launch template: Team, Workload, Environment, MigrationWave. Evidence: compliance report clean for the account.	Tag policy; GR-3	Wave lead
Before launch	Non-production hosts carry the Schedule tag; Multi-AZ only on production databases. Evidence: config rule reports none missing.	S-1, D-1; GR-6	Workload owner
Before launch	Build the VPC from the template: gateway endpoints, and one NAT gateway outside production. Evidence: stack outputs list both endpoints.	N-1, N-2; GR-8	Network owner
Before launch	Applications log at info; the account baseline sets retention; the shared agent configuration keeps host-level metrics only. Evidence: log groups show retention; metric count checked.	L-1, L-2, L-4; GR-7	Workload owner
At cutover	Add or raise the account's budget to the sized monthly cost plus ten percent. Evidence: budget stack updated in the same pull request.	Budgets; GR-1	Finance
At cutover	Delete test-cutover volumes and rehearsal load balancers when the test closes. Evidence: nothing detached or idle left in the account.	W-1, W-3; GR-10	Wave lead
After cutover	Finalize the cutover, disconnect each source server from the service, and mark it archived. Evidence: active source servers match the migration account.	W-2; GR-10	Site reliability

When	Step and the evidence it leaves	Answers	Owner
After cutover	Daily snapshot policies carry an expiry; none is created without one. Evidence: policy list has no policy without a retention count.	W-4; GR-10	Site reliability
After cutover	Thirty days later, compare the sizing sheet with CloudWatch and record the difference. Evidence: entry in the monthly cost review.	R-1, R-2; GR-9	Wave lead

Cost avoidance on the remaining waves

Not a saving

Everything in this section is not measured, not counted toward the gate, and not a saving on the bill. There is no "before" on the Cost and Usage Report for a workload that has not moved, so the guarantee cannot count it, and this engagement does not. It is a model of what the guardrails and the runbook should prevent, against Seatlark's own plan.

Your plan adds \$71,000 a month across waves 7 to 11, priced the way the first six waves were sized. The model applies to each wave the share of its planned cost that this engagement's findings cut from the first six waves, lever by lever, and the levers are the guardrails above.

The avoidance model's levers: not measured, not counted toward the gate, and not a saving on the bill. The total is a range of a plan, not an observation.

Lever, built this week	How it prevents cost	Guardrails	Answers	Share of a wave's planned cost	Monthly, on \$71,000
The wave runbook sizes from 30-day vCenter 95th-percentile use, not allocation	Rightsized at launch	GR-9	R-1, R-2, R-3, R-4	7.0% to 10.0%	\$4,970 to \$7,100
Instance Scheduler on by default in non-production accounts; Multi-AZ only in production	Non-production bills business hours	GR-6	S-1, S-2, D-1	2.0% to 4.0%	\$1,420 to \$2,840
An AWS Config rule and an SCP that refuse new gp2 and io1 volumes without an exception tag	Storage defaults	GR-4, GR-5	E-1, E-2a, E-2b	1.0% to 2.0%	\$710 to \$1,420
Log-group retention set by the account baseline; agent metrics from a shared configuration	Logging defaults	GR-7	L-1, L-2, L-3, L-4	1.0% to 2.0%	\$710 to \$1,420
The VPC template: S3 and DynamoDB gateway endpoints, one NAT gateway in non-production	Network defaults	GR-8	N-1, N-2	0.5% to 1.0%	\$355 to \$710
MGN cleanup in the cutover checklist (finalize, disconnect, archive)	No leftovers	GR-10	W-2	0.5% to 1.0%	\$355 to \$710
Total				12% to 20%	\$8,520 to \$14,200

At 12% to 20% of the \$71,000 a month the remaining waves add, avoidance at colo exit is **\$8,520 to \$14,200 a month, or \$100,000 to \$170,000 a year** (the monthly range times twelve, rounded to \$10,000), at the end-state run rate. It arrives wave by wave, and the first wave to use the runbook, wave 7, carries the most. Wave 7 is orders and checkout, a customer-facing tier, so its \$2,640 to \$4,400 a month is modeled only as far as peak data or a load test supports a smaller size ([Sizing: use over time, not allocation](#)); without one, the runbook keeps the allocated size and the wave avoids nothing.

Modeled avoidance by wave, a month: not measured, not counted toward the gate, and not a saving on the bill. Wave 6 is not here; it landed before the runbook existed.

WaveWhen	Planned monthly cost added	At 12%	At 20%
7July to August 2026	\$22,000	\$2,640	\$4,400
8October to November 2026	\$12,000	\$1,440	\$2,400
9January to March 2027	\$18,000	\$2,160	\$3,600
10March to May 2027	\$14,000	\$1,680	\$2,800
11May to June 2027	\$5,000	\$600	\$1,000
Total	\$71,000	\$8,520	\$14,200

How the range moves. Assumption A1, carried from the Analysis, reads the plan's end state as \$125,300 a month, \$1.5 million a year. Lydia Brandt confirmed that reading on June 9, 2026, so the range above stands. Had the \$1.5 million instead been the total spend over the next twelve months, the end state would be \$170,000 to \$200,000 a month and the remaining waves would add more than \$71,000, scaling the model by roughly 1.6 to 2.1 times; that reading is no longer in play, and the range carries the label above wherever it appears.

What is deliberately not claimed. Two larger numbers are in Seatlark's plan and are not avoidance. Retiring the Direct Connect ports at colo exit is up to \$3,285 a month, already in the plan. Moving the orders database off SQL Server Enterprise, a licensing and application decision, is beyond this engagement. Neither is in the model.

What next week does

Week 6 measures every change on the Cost and Usage Report, seven days before against seven days after, and annualizes it. The data runs through June 15, 2026, the last day the report will have finalized when the measurement runs on June 17, 2026. It then decides the second invoice (\$17,500) and hands the playbook over on June 18, 2026. The measurement counts only the 20 changes in place, plus S3-3 at its estimate; the guardrails and the avoidance model are not in it.

The exact mechanism

Everything here is in `seatlark-landing-zone` and deployed as CloudFormation. The Organizations-level pieces (the tag policy, the SCP, the monitors, and the budgets) live in one stack in the management account. The account-level pieces (the Config rule, the scheduler's remote stack, and the retention function) go out as a StackSet to the accounts in the table above. Placeholders in angle brackets are values the pull request fills from `accounts.yaml`.

Budgets and anomaly monitors

One budget and one monitor per account, created in the management account with a filter on the linked account, so a new account is a new pair of resources. The first account's pair is below; the other 15 differ only in the account ID, the name, the owner's address, and the limit from [the budget tables](#).

YAML

```
# seatlark-landing-zone/finops/budgets.yaml (management account)
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  BudgetMarketplaceprod:
    Type: AWS::Budgets::Budget
    Properties:
      Budget:
        BudgetName: acct-marketplace-prod
        BudgetType: COST
        TimeUnit: MONTHLY
        BudgetLimit: {Amount: 14400, Unit: USD}
        CostTypes: {UseAmortized: true}
        CostFilters:
          LinkedAccount: ["<marketplace-prod-account-id>"]
      NotificationsWithSubscribers:
        - Notification: {NotificationType: ACTUAL, ComparisonOperator: GREATER_THAN, Threshold:
80, ThresholdType: PERCENTAGE}
          Subscribers: [{SubscriptionType: EMAIL, Address: "<marketplace-owner-email>"}]
        - Notification: {NotificationType: ACTUAL, ComparisonOperator: GREATER_THAN, Threshold:
100, ThresholdType: PERCENTAGE}
          Subscribers: [{SubscriptionType: EMAIL, Address: "<marketplace-owner-email>"}]
        - Notification: {NotificationType: FORECASTED, ComparisonOperator: GREATER_THAN,
Threshold: 100, ThresholdType: PERCENTAGE}
          Subscribers: [{SubscriptionType: EMAIL, Address: "<marketplace-owner-email>"}]
  MonitorMarketplaceprod:
    Type: AWS::CE::AnomalyMonitor
    Properties:
      MonitorName: acct-marketplace-prod
      MonitorType: CUSTOM
      MonitorSpecification: '{"Dimensions":{"Key":"LINKED_ACCOUNT","Values":["<marketplace-
prod-account-id>"],"MatchOptions":["EQUALS"]}}'
  DailyMarketplaceprod:
```

```

Type: AWS::CE::AnomalySubscription
Properties:
  SubscriptionName: daily-marketplace-prod
  Frequency: DAILY
  MonitorArnList: [!Ref MonitorMarketplaceprod]
  Subscribers: [{Type: EMAIL, Address: "<marketplace-owner-email>"}]
  ThresholdExpression: '{"Dimensions":
{"Key":"ANOMALY_TOTAL_IMPACT_ABSOLUTE","MatchOptions":["GREATER_THAN_OR_EQUAL"],"Values":
["100"]}}'
  ServiceMonitor:
    Type: AWS::CE::AnomalyMonitor
    Properties: {MonitorName: by-service, MonitorType: DIMENSIONAL, MonitorDimension: SERVICE}
ImmediateAlert:
  Type: AWS::CE::AnomalySubscription
  Properties:
    SubscriptionName: immediate-over-1000
    Frequency: IMMEDIATE
    MonitorArnList: [!Ref ServiceMonitor]
    Subscribers: [{Type: SNS, Address: "<on-call-sns-topic-arn>"}]
    ThresholdExpression: '{"Dimensions":
{"Key":"ANOMALY_TOTAL_IMPACT_ABSOLUTE","MatchOptions":["GREATER_THAN_OR_EQUAL"],"Values":
["1000"]}}'

```

An immediate alert can only go to an SNS topic, which fans out to the on-call channel.

The tag policy

An Organizations tag policy fixes the key spelling and the allowed values and enforces them on the services that support it. Create it at the organization root, attach it to the accounts in the table, and then activate the keys as cost allocation tags so they appear on the bill.

JSON

```
{
  "tags": {
    "Team": {
      "tag_key": {"@@assign": "Team"},
      "tag_value": {"@@assign": ["data-etl", "fulfillment-content", "inventory-ingest",
"inventory-search", "marketplace-api", "marketplace-web", "partner-api", "platform-sre"]},
      "enforced_for": {"@@assign": ["ec2:instance", "ec2:volume", "rds:db", "s3:bucket"]}
    },
    "Workload": {
      "tag_key": {"@@assign": "Workload"},
      "enforced_for": {"@@assign": ["ec2:instance", "ec2:volume", "rds:db", "s3:bucket"]}
    },
    "Environment": {
      "tag_key": {"@@assign": "Environment"},
      "tag_value": {"@@assign": ["management", "nonprod", "prod", "sandbox", "security",
"shared"]},
      "enforced_for": {"@@assign": ["ec2:instance", "ec2:volume", "rds:db", "s3:bucket"]}
    },
    "MigrationWave": {
      "tag_key": {"@@assign": "MigrationWave"},
      "tag_value": {"@@assign": ["wave-0", "wave-1", "wave-2", "wave-3", "wave-4", "wave-5",
"wave-6", "wave-7", "wave-8", "wave-9", "wave-10", "wave-11", "none"]},
      "enforced_for": {"@@assign": ["ec2:instance", "ec2:volume", "rds:db", "s3:bucket"]}
    }
  }
}
```

BASH

```
aws organizations enable-policy-type --root-id <root-id> --policy-type TAG_POLICY
aws organizations create-policy --name cost-allocation-tags --type TAG_POLICY \
  --description "Team, Workload, Environment, MigrationWave" --content file://tag-policy.json
aws organizations attach-policy --policy-id <policy-id> --target-id <marketplace-staging-
account-id>
aws organizations attach-policy --policy-id <policy-id> --target-id <marketplace-dev-account-
id>
aws organizations attach-policy --policy-id <policy-id> --target-id <partner-nonprod-account-
id>
aws organizations attach-policy --policy-id <policy-id> --target-id <perf-test-account-id>
aws organizations attach-policy --policy-id <policy-id> --target-id <sandbox-eng-account-id>
aws ce update-cost-allocation-tags-status --cost-allocation-tags-status \
  TagKey=Team,Status=Active \
  TagKey=Workload,Status=Active \
  TagKey=Environment,Status=Active \
  TagKey=MigrationWave,Status=Active
aws resourcegroupstaggingapi start-report-creation --s3-bucket <report-bucket>
```

The compliance report from the last command lists every resource that breaks the policy in every account, production included, and it changes nothing. That list is the retrofit list for the production accounts before the policy reaches them on June 15, 2026.

The storage rule and the SCP

The Config rule is a custom policy rule written in AWS CloudFormation Guard. It is non-compliant for a gp2 or io1 volume unless the volume carries the StorageException tag.

YAML

```
# seatlark-landing-zone/baseline/storage-rule.yaml (StackSet, every account)
Resources:
  VolumeTypeRule:
    Type: AWS::Config::ConfigRule
    Properties:
      ConfigRuleName: ebs-volume-gp3-or-excepted
      Description: gp2 and io1 volumes need the StorageException tag
      Scope:
        ComplianceResourceTypes: [AWS::EC2::Volume]
      Source:
        Owner: CUSTOM_POLICY
        SourceDetails:
          - EventSource: aws.config
            MessageType: ConfigurationItemChangeNotification
        CustomPolicyDetails:
          PolicyRuntime: guard-2.x.x
          PolicyText: |
            let blocked = ["gp2", "io1"]
            rule gp2_io1_need_exception when resourceType == "AWS::EC2::Volume" {
              configuration.volumeType NOT IN %blocked OR
              configuration.tags[ key == "StorageException" ] !empty
              <<gp2 and io1 volumes need the StorageException tag; use gp3>>
            }
          }
```

The SCP refuses the create call itself. `ec2:VolumeType` is the type being created, and the `Null` condition is true when the request carries no exception tag. Both `CreateVolume` and `RunInstances` are covered, because an instance launch creates its volumes from the block device mappings.

YAML

```
# seatlark-landing-zone/org/scp-volume-type.yaml (management account)
Resources:
  RefuseGp2AndIo1:
    Type: AWS::Organizations::Policy
    Properties:
      Name: refuse-gp2-io1-without-exception
      Type: SERVICE_CONTROL_POLICY
      Description: New gp2 and io1 volumes need the StorageException tag
      TargetIds:
        - <marketplace-staging-account-id>
        - <marketplace-dev-account-id>
        - <partner-nonprod-account-id>
        - <perf-test-account-id>
        - <sandbox-eng-account-id>
      Content:
        Version: "2012-10-17"
        Statement:
          - Sid: RefuseGp2AndIo1WithoutException
            Effect: Deny
            Action: [ec2:CreateVolume, ec2:RunInstances]
            Resource: arn:aws:ec2:*:*:volume/*
            Condition:
              StringEquals: {"ec2:VolumeType": [gp2, io1]}
              "Null": {"aws:RequestTag/StorageException": "true"}
```

At the production accounts, add their IDs to TargetIds in a pull request on June 23, 2026. Test it first in the sandbox account: launching a gp2 volume without the tag fails with an explicit deny, and the same call with `--tag-specifications`

'ResourceType=volume,Tags=[{Key=StorageException,Value=<ticket>}]' goes through.

Instance Scheduler

The schedule is defined once, as the `Custom::ServiceInstanceSchedule` resource `S-1` created against the Instance Scheduler hub stack (its `ServiceInstanceScheduleServiceToken` output is the resource's service token); each non-production and sandbox account runs the scheduler's remote stack, which reads it. An instance opts in with a tag, so the launch templates in `marketplace-infra` and `partner-infra` tag every instance with `Schedule: weekday-12h`, and a new non-production host is scheduled the day it is born. The rule below catches any host launched without it.

YAML

```
# seatlark-landing-zone/baseline/scheduler-rule.yaml (StackSet, non-production and sandbox accounts)
Resources:
  ScheduleTagRule:
    Type: AWS::Config::ConfigRule
    Properties:
      ConfigRuleName: instance-has-schedule-tag
      Scope: {ComplianceResourceTypes: [AWS::EC2::Instance]}
      Source: {Owner: AWS, SourceIdentifier: REQUIRED_TAGS}
      InputParameters: {tag1Key: Schedule}
```

The default schedule is 12 hours on Mon-Fri, 07:00 to 19:00 America/New_York; in the schedule's period, BeginTime is 07:00 and EndTime is 18:59. A host that needs another window gets a different schedule name in the pull request.

Log-group retention

The account baseline sets retention on every new log group the moment it is created. Application and flow groups get 30 days and groups under the audit prefix get 90. Security and audit logs that must outlive that are kept in S3, not in CloudWatch Logs.

YAML

```
# seatlark-landing-zone/baseline/log-retention.yaml (StackSet, every account)
Resources:
  RetentionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement: [{Effect: Allow, Principal: {Service: lambda.amazonaws.com}, Action:
sts:AssumeRole}]
      ManagedPolicyArns: [arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole]
      Policies:
        - PolicyName: set-retention
          PolicyDocument:
            Version: "2012-10-17"
            Statement: [{Effect: Allow, Action: ["logs:DescribeLogGroups",
"logs:PutRetentionPolicy"], Resource: "*"}]
  RetentionFunction:
    Type: AWS::Lambda::Function
    Properties:
      Runtime: python3.12
      Handler: index.handler
      Role: !GetAtt RetentionRole.Arn
      Code:
        ZipFile: |
          import boto3
          logs = boto3.client("logs")
          def handler(event, context):
            name = event["detail"]["requestParameters"]["logGroupName"]
```

```

        days = 90 if name.startswith("/seatlark/audit") else 30
        found = logs.describe_log_groups(logGroupNamePrefix=name)["logGroups"]
        if any(g["logGroupName"] == name and "retentionInDays" not in g for g in found):
            logs.put_retention_policy(logGroupName=name, retentionInDays=days)
NewLogGroupRule:
  Type: AWS::Events::Rule
  Properties:
    EventPattern:
      source: [aws.logs]
      detail-type: [AWS API Call via CloudTrail]
      detail: {eventSource: [logs.amazonaws.com], eventName: [CreateLogGroup]}
    Targets: [{Id: retention, Arn: !GetAtt RetentionFunction.Arn}]
RetentionPermission:
  Type: AWS::Lambda::Permission
  Properties:
    FunctionName: !Ref RetentionFunction
    Action: lambda:InvokeFunction
    Principal: events.amazonaws.com
    SourceArn: !GetAtt NewLogGroupRule.Arn

```

The event rule acts on log groups created after it is deployed; L-1 set retention on the groups that already existed. The AWS managed rule CW_LOGGROUP_RETENTION_PERIOD_CHECK, with a minimum of 30 days, is the backstop for a group the event rule missed.

The VPC template

The landing zone used to stamp three NAT gateways and no endpoints into every VPC. The template now takes an environment parameter: gateway endpoints for S3 and DynamoDB in every VPC (they cost nothing), three NAT gateways in production, and one everywhere else. The 4 existing non-production VPCs already match, because N-1 and N-2 changed them; the template makes the next VPC match from the start.

YAML

```
# seatlark-landing-zone/network/vpc.yaml (excerpt)
Parameters:
  Environment: {Type: String, AllowedValues: [prod, nonprod, sandbox, shared]}
Conditions:
  IsProd: !Equals [!Ref Environment, prod]
Resources:
  S3Endpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      VpcId: !Ref Vpc
      VpcEndpointType: Gateway
      ServiceName: !Sub com.amazonaws.${AWS::Region}.s3
      RouteTableIds: [!Ref PrivateRouteA, !Ref PrivateRouteB, !Ref PrivateRouteC]
  NatA:
    Type: AWS::EC2::NatGateway
    Properties: {SubnetId: !Ref PublicSubnetA, AllocationId: !GetAtt EipA.AllocationId}
  NatB:
    Type: AWS::EC2::NatGateway
    Condition: IsProd
    Properties: {SubnetId: !Ref PublicSubnetB, AllocationId: !GetAtt EipB.AllocationId}
  DefaultRouteB:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteB
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !If [IsProd, !Ref NatB, !Ref NatA]
```

The DynamoDB endpoint is the same resource with the dynamodb service name. NatC, EipB, and EipC take the same condition as NatB, and DefaultRouteC follows DefaultRouteB. A production VPC keeps 3 NAT gateways so it survives the loss of a zone; every other VPC has 1. Preview a change with `aws cloudformation create-change-set` before you execute it.

The sizing sheet

The runbook's sizing sheet is one row per server, from the 30 days of vCenter statistics before the cutover ticket opens. It reads the two-hour averages vCenter keeps for the past month: the 95th percentile of the CPU and memory series, and the highest two-hour average as the busiest hour.

POWERSHELL

```
# seatlark-landing-zone/runbooks/sizing-sheet.ps1 -VmName <vm> [-Days 30]
param([Parameter(Mandatory)][string]$VmName, [int]$Days = 30)
$vm = Get-VM -Name $VmName
$start = (Get-Date).AddDays(-$Days)
function Get-P95([double[]]$v) { $s = $v | Sort-Object; $s[[math]::Ceiling(0.95 * $s.Count) - 1] }
$cpu = Get-Stat -Entity $vm -Stat cpu.usage.average -Start $start -IntervalSecs 7200 | ForEach-Object Value
$mem = Get-Stat -Entity $vm -Stat mem.usage.average -Start $start -IntervalSecs 7200 | ForEach-Object Value
[pscustomobject]@{
    Vm = $VmName; Vcpus = $vm.NumCpu; MemoryGiB = $vm.MemoryGB
    CpuP95Percent = Get-P95 $cpu; CpuBusiestPercent = ($cpu | Measure-Object -Maximum).Maximum
    MemP95Percent = Get-P95 $mem
} | Export-Csv -Append -NoTypeInfo sizing-sheet.csv
```

The sheet's CPU columns feed the rule in [the sizing table](#). For a customer-facing server, the sheet is not enough: add a column for the busiest on-sale window at one-minute resolution, from the application's own metrics, or attach the load-test result for the candidate size. The ticket is not complete without one of them.

MGN cleanup

Two commands finish a server's cutover and one retires it. The first runs at sign-off; the others run 14 days later, from the same list of source server IDs.

BASH

```
# at sign-off, per source server in the wave
aws mgn finalize-cutover --source-server-id <source-server-id>

# 14 days later: stop replication, then archive
while read -r id; do
    aws mgn disconnect-from-service --source-server-id "$id"
    aws mgn mark-as-archived --source-server-id "$id"
done < wave-source-servers.txt
```

Check the result from the migration account: the active source servers (`aws mgn describe-source-servers --filters isArchived=false`) should be exactly the servers still migrating, and the account should hold no instances or volumes beyond their replication servers and staging disks. Disconnecting does not touch test or cutover instances that a launch created, which is why the checklist deletes test-cutover volumes when the test closes.

AWS, Amazon Web Services, AWS Graviton, and the AWS service names are trademarks of Amazon.com, Inc. or its affiliates. Ansible is a trademark of Red Hat, Inc. Elasticsearch is a trademark of Elasticsearch B.V. PostgreSQL is a registered trademark of the PostgreSQL Community Association of Canada. MySQL is a trademark of Oracle Corporation and its affiliates. Redis is a trademark of Redis Ltd. Microsoft and SQL Server are trademarks of the Microsoft group of companies. VMware and vCenter are trademarks of Broadcom Inc. NetApp is a trademark of NetApp, Inc. Java is a trademark of Oracle Corporation and its affiliates. Vantalect is not affiliated with or endorsed by any of them. Seatlark, Inc. is fictional.