CHECKLIST

# Azure VM Scaling Cost Optimization Checklist

A practical checklist for deciding when to scale Azure virtual machines vertically, horizontally, or by schedule, with validation steps to reduce spend without increasing operational risk.

## Azure VM Scaling Strategies for Cost Optimization

## Checklist

Use this checklist to decide whether a VM should be resized, split into more instances, or scheduled down outside active hours.

## 1) Confirm the workload profile

- [ ] Identify whether the workload is stateful, stateless, or mixed.
- [ ] Record the primary use case: database, application server, worker, batch, admin tool, or other.
- [ ] Document business-critical periods, peak windows, and idle periods.
- [ ] Note any dependencies on local disk, local sessions, or single-node operation.

## 2) Gather enough evidence

- [ ] Review at least 14 to 30 days of monitoring data.

- [ ] Collect CPU, memory, disk latency, disk queue depth, IOPS, and network throughput.
- [ ] Compare averages with peaks, not averages alone.
- [ ] Check whether incidents, retries, or slow requests align with resource bursts.
- [ ] Identify the bottleneck that actually affects users.

## 3) Decide whether vertical scaling fits

- [ ] Confirm the application can tolerate a resize or restart.
- [ ] Verify that CPU is not the only metric driving the decision.
- [ ] Check memory pressure, paging, storage limits, and network limits.
- [ ] Confirm the target VM size has enough headroom for peak demand.
- [ ] Validate that resizing will not reduce disk or network capability below acceptable levels.
- [ ] Estimate savings after including any operational downtime or maintenance risk.

## 4) Decide whether horizontal scaling fits

- [ ] Confirm the workload can run across multiple instances.
- [ ] Verify stateless request handling or externalized session storage.
- [ ] Check that load balancing, health probes, and deployment automation are in place.
- [ ] Confirm shared storage or replication design is acceptable.
- [ ] Estimate the cost of extra instances plus any supporting services.
- [ ] Ensure failure of one instance will not take down the service.

## 5) Consider schedule-based scaling

- [ ] Identify predictable peaks and troughs such as office hours, batch windows, or weekends.

- [ ] Determine whether non-production systems can be stopped or reduced outside active hours.
- [ ] Check whether start/stop automation is approved and monitored.
- [ ] Confirm that patching, backups, and maintenance tasks still run successfully.
- [ ] Validate that any scheduled shutdowns do not affect critical jobs or alerts.

## 6) Compare total cost, not just VM size

- [ ] Include load balancers, gateways, queueing layers, storage, replication, and monitoring in the cost model.
- [ ] Compare baseline cost, peak cost, and average monthly cost.
- [ ] Confirm that a smaller VM will not shift bottlenecks to another tier.
- [ ] Review licensing, support, and operational overhead if applicable.
- [ ] Document the expected savings and the assumptions behind them.

## 7) Validate in non-production first

- [ ] Reproduce the change in a test or staging environment.
- [ ] Use the same monitoring, alert thresholds, and test workload pattern where possible.
- [ ] Confirm application behavior during startup, failover, and peak load.
- [ ] Test rollback before the production change.
- [ ] Record the expected success criteria in advance.

## 8) Prepare a production rollout plan

- [ ] Choose a maintenance window or low-risk observation period.
- [ ] Define who approves the change and who can roll it back.
- [ ] Create a rollback path with clear timing and decision thresholds.

- [ ] Notify affected teams before the change.
- [ ] Make sure backups, snapshots, or recovery options are current.

## 9) Observe after the change

- [ ] Monitor CPU, memory, disk latency, and network throughput immediately after the change.
- [ ] Watch application response times, error rates, and retry counts.
- [ ] Confirm autoscaling or instance distribution behaves as expected.
- [ ] Check for new bottlenecks outside the VM, such as storage or database contention.
- [ ] Keep an observation window long enough to cover the next peak period.

## 10) Review whether the change was successful

- [ ] Compare actual savings against the original estimate.
- [ ] Confirm user experience remained stable or improved.
- [ ] Verify no new operational risk was introduced.
- [ ] Decide whether the new size or scale pattern should be treated as the baseline.
- [ ] Capture lessons learned for the next resizing decision.

## Quick decision guide

- Choose vertical scaling when the workload is single-node, stateful, or hard to split, and a resize can remove excess capacity safely.
- Choose horizontal scaling when the workload is stateless or can be made stateless, and demand varies enough to benefit from elasticity.
- Choose schedule-based scaling when usage is predictable and systems can be reduced or stopped during idle periods.

- Choose a mixed approach when the best result is a right-sized baseline plus extra capacity only during peaks.

## Minimum production checklist before approval

- [ ] Workload type identified
- [ ] Monitoring data reviewed
- [ ] Bottleneck confirmed
- [ ] Cost model updated
- [ ] Non-production test completed
- [ ] Rollback plan documented
- [ ] Owners notified
- [ ] Post-change observation plan defined

## Notes

A cheap VM that causes incidents is not cost optimization. The goal is to match compute supply to real demand while keeping the application stable, supportable, and recoverable.