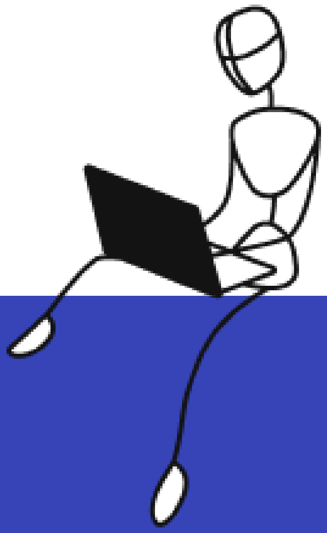
gart.



WHITEPAPER

Migration from AWS to Hetzner: Step-by-Step Guide to Cut Costs on 90%

Prepared by :

Gart Solutions

Issued :

October 2025

gartsolutions.com

info@gartsolutions.com

Understanding the Cost Problem with AWS

Ask anyone who's run production workloads on AWS long enough, and they'll tell you the same thing: your bill grows fast, often unpredictably.

That's not just due to increased usage — it's the nature of AWS billing itself. You're charged for compute, bandwidth, storage, read/write IOPS, managed services, data egress, and even DNS lookups.

What seems like a few cents here and there quickly snowballs into a **$10k+ monthly bill.**

In contrast, platforms like Hetzner offer flat-rate pricing, predictable billing, and high-performance VMs that don't nickel-and-dime you for every packet.

Let's break it down:

Common AWS Cost Traps:

- Data Egress Fees: AWS charges up to $0.09/GB for outbound traffic — Hetzner includes 20 TB free per VM, then only ~$1/TB after.
- Managed Services Premium: AWS RDS, ElastiCache, SQS, Lambda all add layers of cost and margin.
- Per-Request Billing: API Gateway, CloudWatch, S3, etc. charge per request — hard to predict and scale.
- Storage IOPS: EBS IOPS fees balloon costs for disk-intensive workloads.

A recent Data-Aces migration revealed **Hetzner cut costs by 70–80%** across compute and storage alone — and by over 90% in bandwidth fees. That's a massive savings opportunity.

Why Hetzner is a Serious AWS Alternative

Let’s be clear — Hetzner isn’t a one-size-fits-all replacement for AWS.

But for a huge subset of workloads, it outperforms AWS in price-performance, network throughput, and simplicity. If your app doesn’t depend heavily on AWS-native services like Lambda, Step Functions, or Aurora, you can replicate most of your stack on Hetzner with open-source tools and infrastructure-as-code.

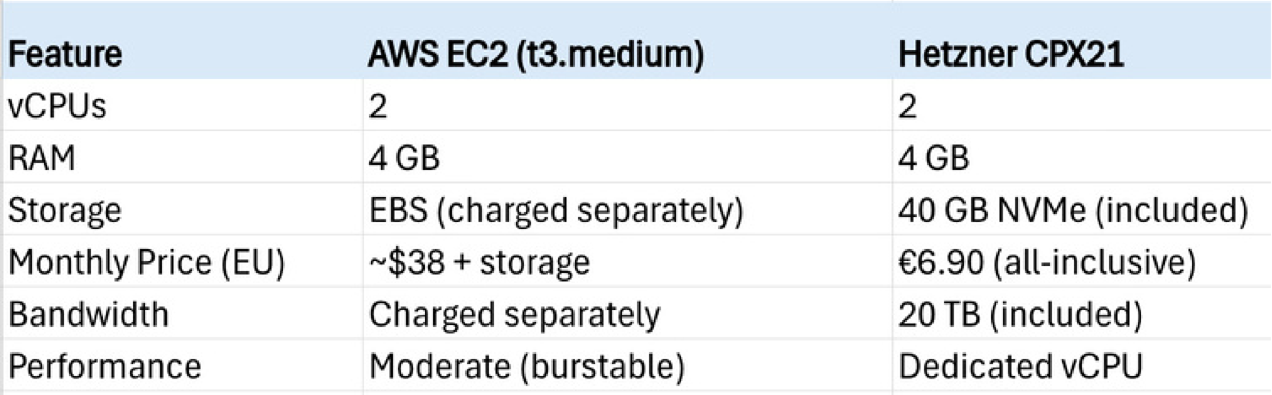
From Reddit threads to blog breakdowns, developers rave about:

- Raw performance per dollar (NVMe SSDs, modern CPUs)
- Flat pricing (no hidden fees, egress surprises, or “request taxes”)
- Network speed (1 Gbps and up)
- Dedicated or cloud servers (flex based on workload)

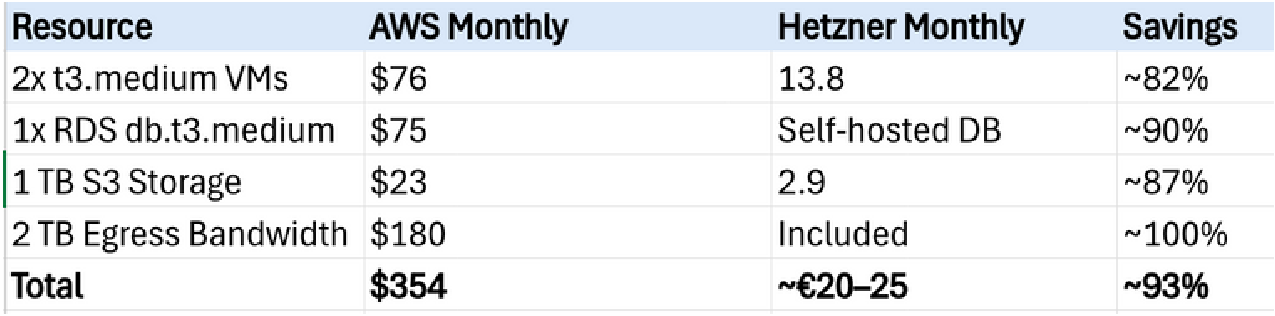
Drawbacks

- No AWS-native managed services (you run your own DBs, queues, etc)
- Fewer data centers (Germany, Finland, US)
- Less redundancy unless architected manually
- Somewhat manual networking (e.g. floating IPs, routing)

Performance & Pricing Table: AWS vs Hetzner



Feature	AWS EC2 (t3.medium)	Hetzner CPX21
vCPUs	2	2
RAM	4 GB	4 GB
Storage	EBS (charged separately)	40 GB NVMe (included)
Monthly Price (EU)	~$38 + storage	€6.90 (all-inclusive)
Bandwidth	Charged separately	20 TB (included)
Performance	Moderate (burstable)	Dedicated vCPU



Resource	AWS Monthly	Hetzner Monthly	Savings
2x t3.medium VMs	$76	13.8	~82%
1x RDS db.t3.medium	$75	Self-hosted DB	~90%
1 TB S3 Storage	$23	2.9	~87%
2 TB Egress Bandwidth	$180	Included	~100%
Total	**$354**	**~€20–25**	**~93%**

(Source: Hetzner Pricing Docs)

Pre-Migration Phase: Scoping, Auditing, Planning

Before you touch any server, scoping is everything. This is where your migration succeeds or fails. The first step? Take a full inventory of everything running in your AWS account.

Step 1: Inventory & Mapping

- EC2 instances
- EBS volumes
- S3 buckets
- IAM roles / users
- Security groups / firewalls
- RDS / Aurora / ElastiCache / Lambda
- CloudWatch / logs / alarms
- Route53 zones / DNS entries
- SQS, API Gateway, Cognito, and any AWS-native services

Use AWS Config, Cost Explorer, Inframap, or Gart's audit suite to pull this automatically.

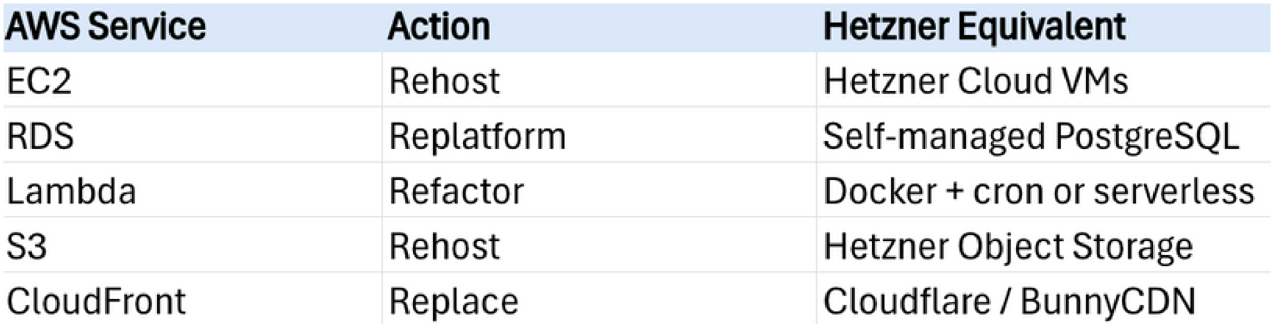
Step 2: Group by Priority & Risk

Make a matrix:

- Mission-critical vs low-risk
- Stateless vs stateful
- Easy to rehost vs tightly coupled to AWS

Step 3: Choose Migration Approach

- Lift & Shift: Fastest, but carries AWS inefficiencies
- Replatform: Move services but optimize architecture
- Refactor: Best long-term, requires dev investment



AWS Service	Action	Hetzner Equivalent
EC2	Rehost	Hetzner Cloud VMs
RDS	Replatform	Self-managed PostgreSQL
Lambda	Refactor	Docker + cron or serverless
S3	Rehost	Hetzner Object Storage
CloudFront	Replace	Cloudflare / BunnyCDN

Building Your Hetzner Foundation (Parallel Infra Setup)

Once your AWS audit is complete - it’s time to build the new home for your infrastructure on Hetzner. This is the "foundation-laying" phase — your Hetzner infrastructure will run in parallel to your AWS environment during testing and validation.

This approach ensures zero production downtime and gives you time to test everything end-to-end before pulling the plug on AWS.

Step 1: Set Up Hetzner Projects & Permissions

- Create a Hetzner Cloud account.
- Inside your project dashboard, create “Projects” for dev, staging, and production.
- Generate API tokens for use with Terraform/Ansible.
- Set up team member access and 2FA.

Step 2: Network Architecture Design

- Create private networks to isolate traffic.
- Assign Floating IPs for public-facing services (can move between servers like AWS Elastic IPs).
- Configure firewall rules (similar to AWS Security Groups).
- Plan subnet ranges, DNS, and NAT rules.

You can automate this with Terraform and manage firewall/IP routing declaratively.

Step 3: Provision VMs (Cloud and Dedicated)

- Hetzner’s CPX (shared) or CAX (dedicated core) instances offer excellent price-performance.
- Use CX11 instances for basic load balancing or cron jobs.
- Deploy via Hetzner Console or IaC tools (Terraform, Pulumi, etc).

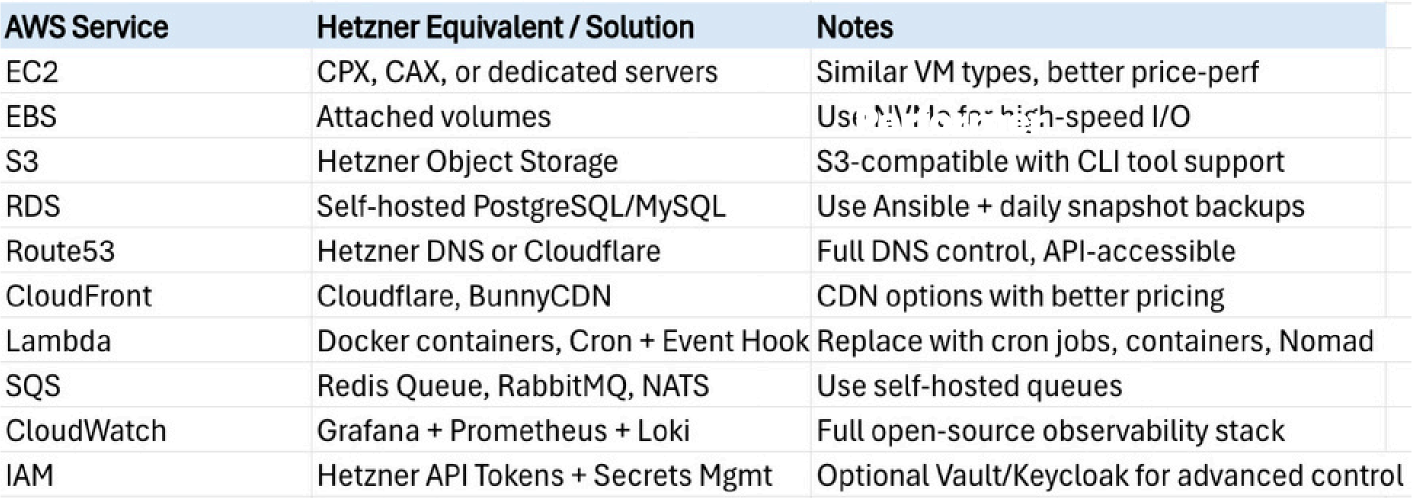
Step 4: Configure Base Images

- Choose OS: Ubuntu LTS, Debian, AlmaLinux, etc.
- Use cloud-init scripts or Ansible roles to install packages, users, monitoring, etc.
- Harden the image: SSH keys, UFW/iptables, fail2ban, etc.

Step 5: Attach Volumes & Object Storage

- Add additional volumes (like AWS EBS) for DB/data separation.
- Hetzner’s Object Storage is S3-compatible — migrate S3 data using:
  - rclone
  - s3cmd
  - awscli sync
  - MinIO client (mc) for metadata-preserving sync

AWS-to-Hetzner Service Mapping Table



AWS Service	Hetzner Equivalent / Solution	Notes
EC2	CPX, CAX, or dedicated servers	Similar VM types, better price-perf
EBS	Attached volumes	Use NVMe for high-speed I/O
S3	Hetzner Object Storage	S3-compatible with CLI tool support
RDS	Self-hosted PostgreSQL/MySQL	Use Ansible + daily snapshot backups
Route53	Hetzner DNS or Cloudflare	Full DNS control, API-accessible
CloudFront	Cloudflare, BunnyCDN	CDN options with better pricing
Lambda	Docker containers, Cron + Event Hook	Replace with cron jobs, containers, Nomad
SQS	Redis Queue, RabbitMQ, NATS	Use self-hosted queues
CloudWatch	Grafana + Prometheus + Loki	Full open-source observability stack
IAM	Hetzner API Tokens + Secrets Mgmt	Optional Vault/Keycloak for advanced control

Terraform & Ansible Sample Modules from Gart Solutions

Gart Solutions offers prebuilt Terraform modules to rapidly spin up:

- CPX clusters with autoscaling
- Load balancers with floating IP failover
- VPC subnets, routes, firewalls
- Volume attachment & formatting
- Monitoring/alerting stacks with Grafana/Loki

Likewise, our Ansible roles cover:

- Base system hardening
- Database provisioning + HA
- Monitoring agents
- Backups and snapshot rotation
- App deployment with zero-downtime rolling updates

These automation tools make your Hetzner setup repeatable, testable, and version-controlled — just like AWS CloudFormation, but fully portable.

Data Migration & Syncing Live Systems (Minimize Downtime)

With your infrastructure now mirrored on Hetzner, it's time to move data from AWS.

This is a critical stage. You want data consistency, minimal downtime, and fallback options.

Step 1: Filesystem & App Data Migration
Use rsync or scp to migrate from EC2 instances or EBS volumes

Run multiple passes:

1. Initial bulk sync (takes longest)
2. Second pass (faster, syncs deltas)
3. Final sync during cutover window

For large disks or bootable images:

- Snapshot EBS volumes
- Attach to an EC2 instance
- Mount, compress, transfer to Hetzner

Step 2: S3 to Hetzner Object Storage
Use rclone with S3-compatible endpoints:

Step 3: Database Sync
If you used AWS RDS:

- Enable logical replication (e.g. pglogical, wal2json)
- Start streaming to your Hetzner DB replica
- During final cutover:
    - Pause writes
    - Apply last WAL logs
    - Promote Hetzner DB

For small DBs:

- Use pg_dump → scp → pg_restore

Step 4: Queue/Cache Warmup

- Flush Redis dumps to disk → copy to Hetzner
- Use queue draining (e.g. Amazon SQS → RabbitMQ) to empty queues before switchover

Step 5: Dual Writes (Advanced)
For zero-downtime:

- Implement temporary dual writes to AWS and Hetzner DBs
- Flip reads to Hetzner once validated
- Decommission AWS after 100% success

This requires app logic or service mesh — Gart Solutions has a dual-write orchestration template using sidecars and feature flags.

DNS Switchover, Load Balancing & Cutover

With your infrastructure now mirrored on Hetzner, it’s time to move data from AWS.

This is a critical stage. You want data consistency, minimal downtime, and fallback options.

Step 1: Filesystem & App Data Migration
Use rsync or scp to migrate from EC2 instances or EBS volumes

Run multiple passes:

1. Initial bulk sync (takes longest)
2. Second pass (faster, syncs deltas)
3. Final sync during cutover window

For large disks or bootable images:

- Snapshot EBS volumes
- Attach to an EC2 instance
- Mount, compress, transfer to Hetzner

Step 2: S3 to Hetzner Object Storage
Use rclone with S3-compatible endpoints:

Step 3: Database Sync
If you used AWS RDS:

- Enable logical replication (e.g. pglogical, wal2json)
- Start streaming to your Hetzner DB replica
- During final cutover:
  - Pause writes
  - Apply last WAL logs
  - Promote Hetzner DB

For small DBs:

- Use pg_dump → scp → pg_restore

Step 4: Queue/Cache Warmup

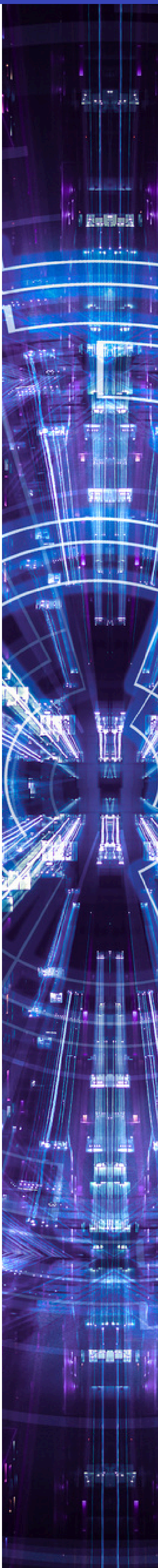
- Flush Redis dumps to disk → copy to Hetzner
- Use queue draining (e.g. Amazon SQS → RabbitMQ) to empty queues before switchover

Step 5: Dual Writes (Advanced)
For zero-downtime:

- Implement temporary dual writes to AWS and Hetzner DBs
- Flip reads to Hetzner once validated
- Decommission AWS after 100% success

This requires app logic or service mesh — Gart Solutions has a dual-write orchestration template using sidecars and feature flags.

Post-Migration Optimization: Monitoring, Backups, Security



Congratulations — your traffic is now flowing through Hetzner!
But don’t pop the champagne yet.
You’ve migrated. Now it’s time to optimize and operationalize.

Monitoring Setup

- Use Prometheus for metrics
- Grafana dashboards
- Loki for log aggregation
- Optional: UptimeRobot, StatusCake for synthetic monitoring

Gart offers plug-and-play dashboards for all Hetzner components.

Backups & DR

- Snapshot databases nightly with pg_basebackup or Ansible roles
- Backup volumes using ZFS snapshots or Hetzner backup volumes
- Sync Object Storage to 3rd-party for redundancy (e.g. Wasabi)

Security

- Harden SSH (disable password login, rotate keys)
- Run regular updates with unattended-upgrades
- Enable UFW / iptables with strict rules
- Use auditd, Fail2ban, and rootkit scanners

Gart Solutions offers a security hardening playbook based on ISO 27001 practices.

Compliance

- Enable data locality (Germany / Finland / US)
- Document controls for GDPR, HIPAA, SOC2 as needed
- Encrypt everything: at-rest, in-transit

Common Migration Pitfalls and How to Avoid Them

Here are some of the most common pitfalls, based on real-world migrations and how to avoid them (especially with Gart Solutions at your side):

1. Underestimating Operational Overhead

AWS handles a lot for you — patching, backups, scaling, metrics, alarms. On Hetzner, you own the ops.

Avoid it by:

- Using Infrastructure as Code (IaC) to replicate environments
- Automating with Ansible, Terraform, Prometheus, and Grafana
- Letting Gart Solutions set up self-healing and auto-recovery tools

2. Floating IP Mismanagement

Unlike AWS's Elastic IPs, Hetzner's Floating IPs require manual reassignment during failover or deployment — a common source of downtime.

Fix it with:

- Scripts for automated failover (Gart provides these prebuilt)
- Health-check monitors that trigger reassignment

3. Lack of Load Testing

You might find Hetzner faster on paper — but if your app wasn't built for 1Gbps NICs, NVMe disks, or specific CPU types, you might hit bottlenecks.

Test early with:

- iperf for network throughput
- fio for disk I/O
- ab, wrk, or k6 for HTTP performance

4. DNS TTL Neglect

Switching DNS during migration without properly lowering TTL leads to inconsistent user routing — and can delay cutovers by hours.

Best practice:

- Lower TTL to 30s 48 hours before migration
- Use tools like dnschecker.org and dig +trace to confirm propagation

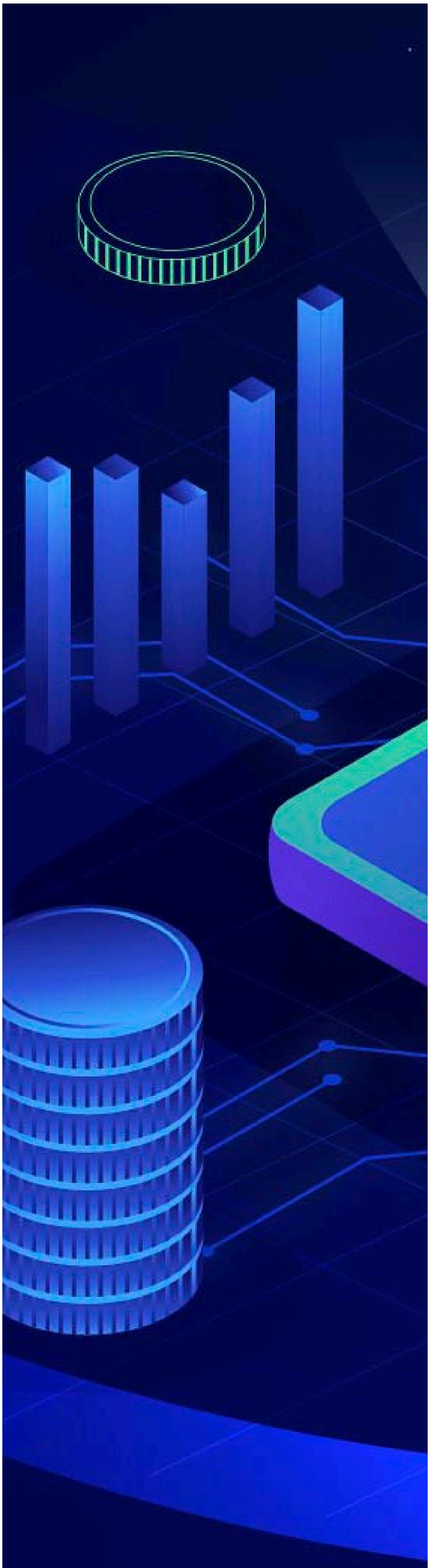
5. Security Loopholes Post-Migration

Many teams forget to recreate IAM-equivalent controls or audit their firewall policies — leaving ports exposed.

Solution:

- Use Hetzner's firewall rules + OS-level firewalls (UFW/iptables)
- Run external scans (Nessus, OpenVAS)
- Have Gart Solutions implement a full security hardening pass.

How to Decide If Hetzner is Right for You



Hetzner isn't ideal for everyone. Before you commit, assess your workload's nature and your team's comfort level.

When Hetzner Is a Great Fit

✅ Predictable workloads (SaaS, backend APIs, databases)

✅ Steady traffic with known patterns

✅ Full control over infra > abstracted services

✅ Ops team ready for deeper ownership

✅ You're cost-sensitive and margin-focused

When to Think Twice

⚠️ You rely heavily on AWS services like Lambda, Step Functions, or Aurora

⚠️ Your traffic is highly bursty and unpredictable

⚠️ You need compliance coverage in dozens of regions

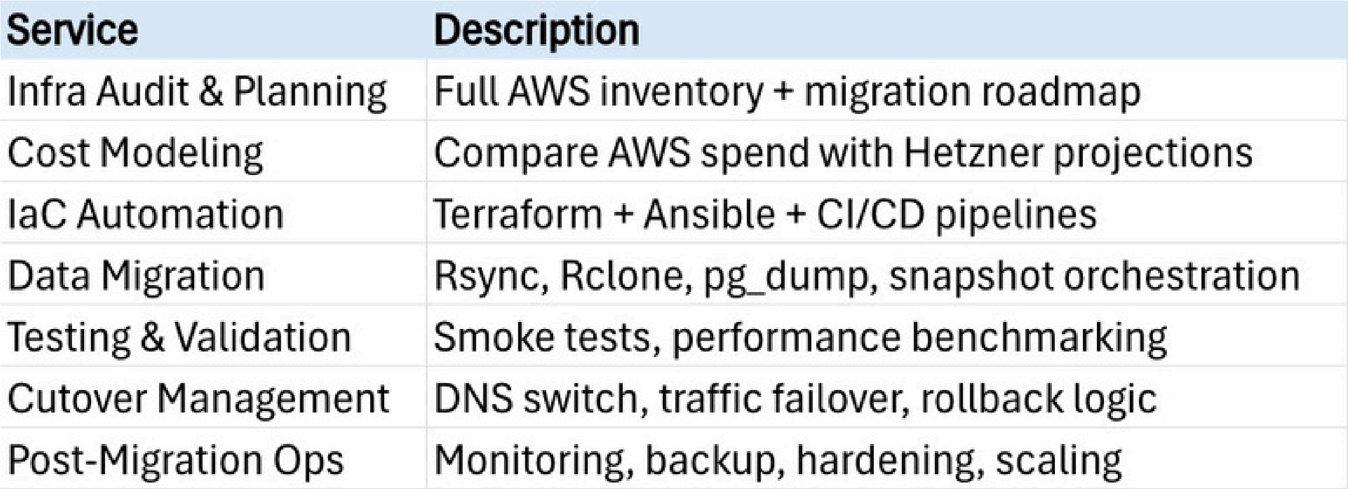
⚠️ You have no DevOps capacity in-house

Gart Solutions offers a free assessment to help you decide, with:

- Cost comparisons
- Feature parity maps
- Migration effort scoring

What Makes Gart Solutions the Ideal Migration Partner

Gart Solutions isn’t just another provider. We’ve led dozens of successful migrations from AWS, GCP, and Azure to Hetzner and other independent providers.



Service	Description
Infra Audit & Planning	Full AWS inventory + migration roadmap
Cost Modeling	Compare AWS spend with Hetzner projections
IaC Automation	Terraform + Ansible + CI/CD pipelines
Data Migration	Rsync, Rclone, pg_dump, snapshot orchestration
Testing & Validation	Smoke tests, performance benchmarking
Cutover Management	DNS switch, traffic failover, rollback logic
Post-Migration Ops	Monitoring, backup, hardening, scaling

We bring repeatable playbooks, real-world experience, and DevOps automation to deliver zero-drama migrations that stick.

We also:

- Train your team
- Monitor your systems
- Stay involved long after the switch

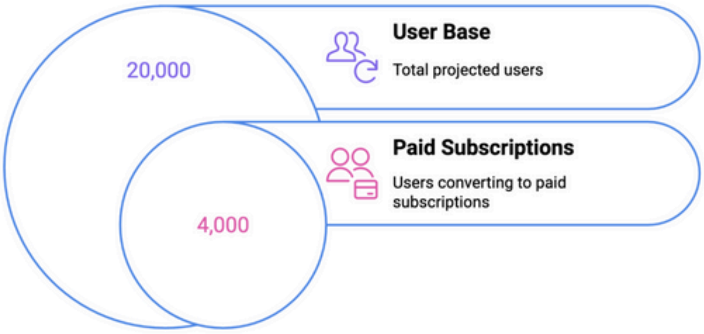
When you partner with Gart, you’re not just getting cheaper infrastructure. You’re getting infrastructure that works.

Case Study

Why Hetzner is a Game-Changer for High-Traffic Workloads

Client background

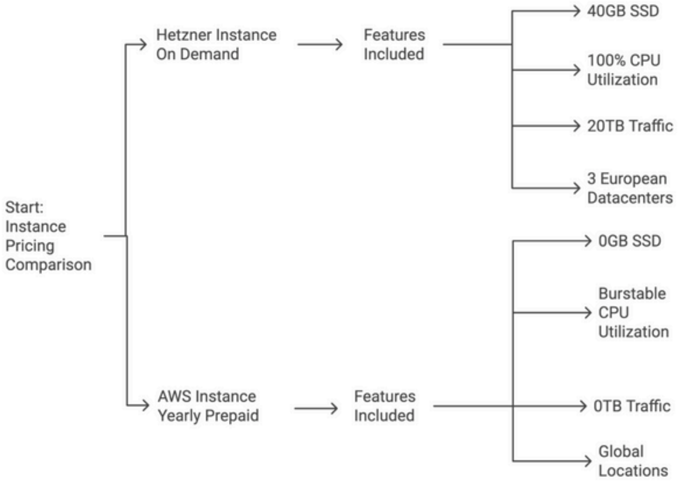
An iGaming company planned to host a platform expecting 20,000 users, offering free trials with game downloads (500MB–1GB each) and targeting 4,000 paid subscribers.



Business challenge

On AWS, traffic costs from 16,000 free users (up to 10GB each) made the business model unsustainable. Projected $80,000 revenue wouldn't cover AWS egress fees.

Solution



Gart Solutions recommended migrating to Hetzner Private Cloud, ensuring predictable, low-cost infrastructure optimized for heavy traffic workloads.

Result

- Infrastructure costs slashed
- Business model became profitable & sustainable
- Supported high-traffic growth without AWS cost overhead

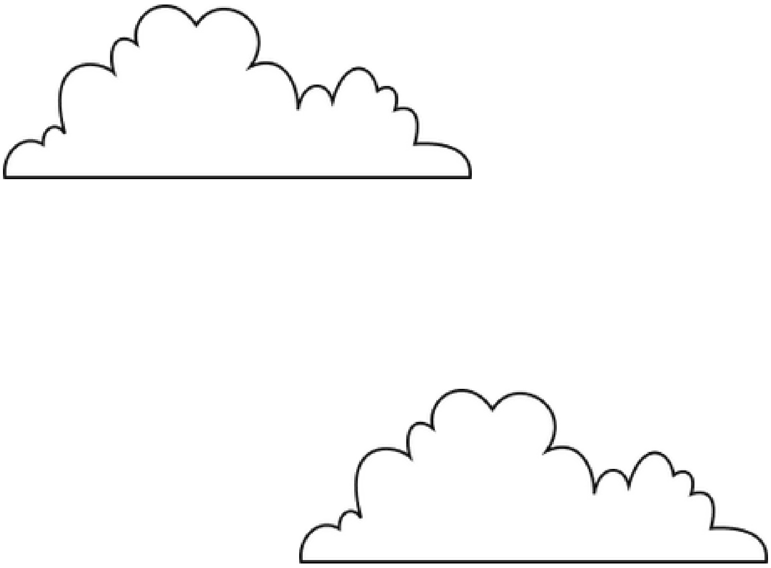
Contributors

Fedir Kompaniiets

Co-Founder at Gart Solutions, DevOps and Cloud Solutions Consultant

Roman Burdiuzha

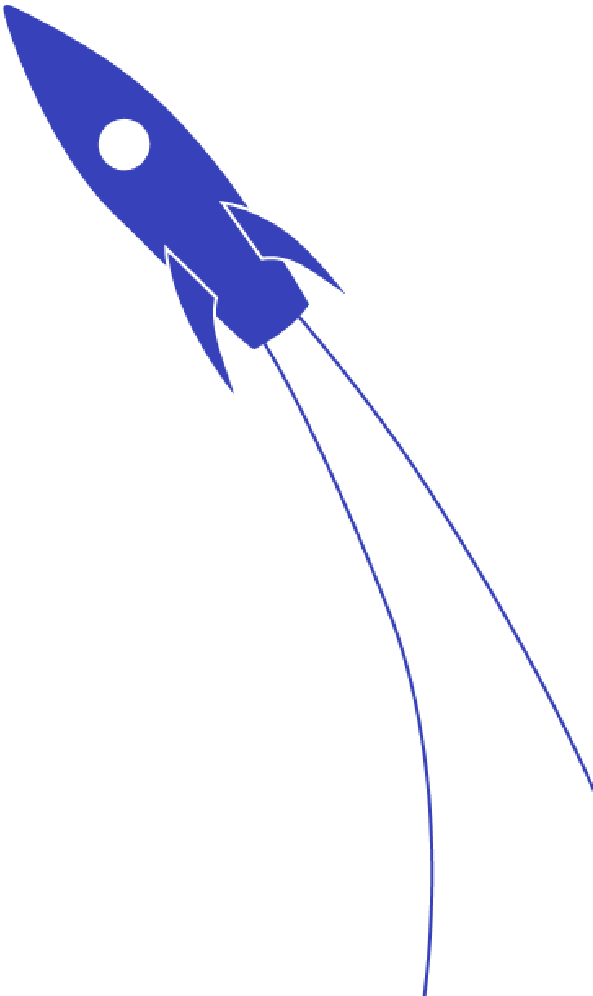
Cloud Architect, Co-Founder and CTO at Gart Solutions gart.



Get in touch with us!

Gart Solutions is a Cloud and Devops services agency that provides businesses with infrastructure setup, automatization, cloud migration, cloud native development, CI/CD, and more.

We work to solve your tech challenges on time and budget and provide infrastructure with the endurance it needs
to let you focus on what matters the most – growing business.



gartsolutions.com

info@gartsolutions.com