Tradeoff Analysis and Edge Case

Routing Saturation

In large AWS and hybrid networks, TGW route tables can fill up, which risks outages or connectivity failures. I handle this by segmenting route tables by environment (Dev, QA, Prod) so each is kept lean and focused. Whenever possible, I summarize routes. I've considered alternatives like Cloud WAN or multiple TGWs for future scale, but for this org, environment-based segmentation is the most maintainable. I also monitor route utilization and set up alerts, so we're never surprised.

DNS Failures

DNS is a critical dependency, so I deploy HA Route 53 Resolver endpoints in each region and test failover regularly. I also keep on-prem caching resolvers as backup, so that even if a central Resolver fails, workloads can still resolve critical addresses. I debated more complex DNS patterns, but chose this for a balance between resilience and simplicity. I always set up logging and monitoring for DNS errors, so we can catch problems early.

Metric Latency

In my setup, CloudWatch Agents buffer and forward telemetry at the edge, and I use Kinesis/Firehose to decouple ingestion from storage. By putting logging in a dedicated VPC, I keep telemetry from competing with application traffic. There's some extra cost and operational overhead, but it's worth it for reliability and compliance. I run periodic latency checks and set up fallback local retention for especially critical logs.

3rd-party SaaS Connectivity

Connecting securely to SaaS is always a challenge, so I use AWS PrivateLink to keep the traffic private and restrict access to only authorized endpoints. I lock down egress with strict route tables and DNS policies, so nothing can slip out by mistake. Some SaaS vendors require extra coordination for PrivateLink, but I'd prefer that up-front complexity than risk compliance issues. I regularly review all endpoint policies to keep things tight as the SaaS landscape evolves.