Intro

Surya Lankalapalli · Software Engineer, Microsoft

Databases Resiliency: Active-Active DNS Strategies for DevOps and Failover

The Problem Nobody Talks About

Your database replication is solid. Your failover automation is tested. But when an outage hits, **DNS becomes the bottleneck nobody planned for.**

Stale Records

TTLs keep clients pointing at dead endpoints

Propagation Lag

Single-provider updates take too long to reach all resolvers

Operational Complexity

Manual DNS updates during incidents introduce human error

What This Session Covers

01

DNS Architecture

Why Active-Active multi-provider DNS outperforms traditional models

02

Database Integration

How DNS maps to replication, failover, and read/write routing

03

Practical Patterns

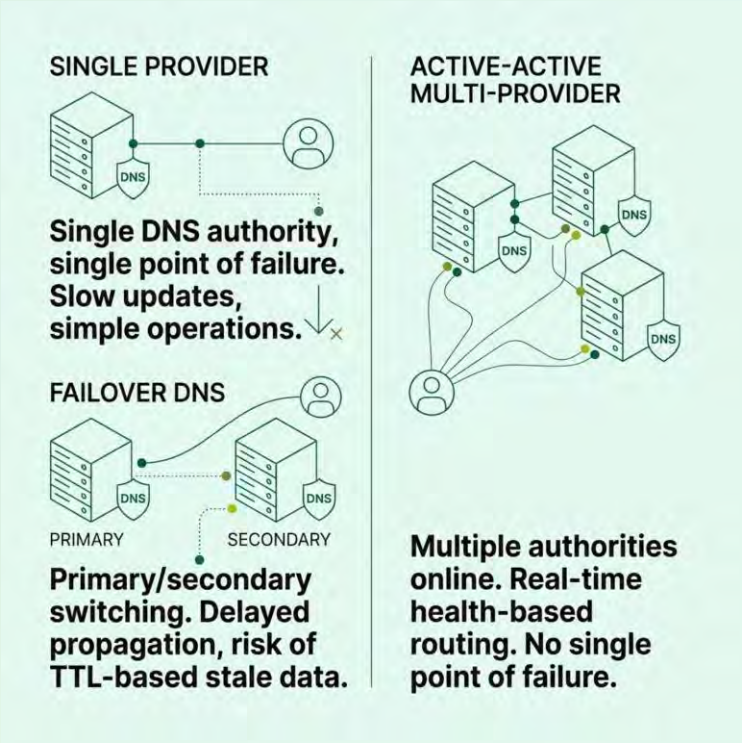
Config sync, health checks, and recovery time reduction

04

Observability

Monitoring distributed DNS and database state in lockstep

Traditional DNS Models and Their Limits



Each generation of DNS architecture trades simplicity for resilience. The jump to Active-Active is where database availability meaningfully improves.

⚠ Single-provider DNS is a silent dependency that can outlast your database failover by minutes - long enough to lose transactions and violate SLAs.

Active-Active DNS

Two or more DNS providers serve authoritative responses **simultaneously** - not in standby, not in sequence.

No Single Point of Failure

If one provider degrades, the other continues serving records without any intervention

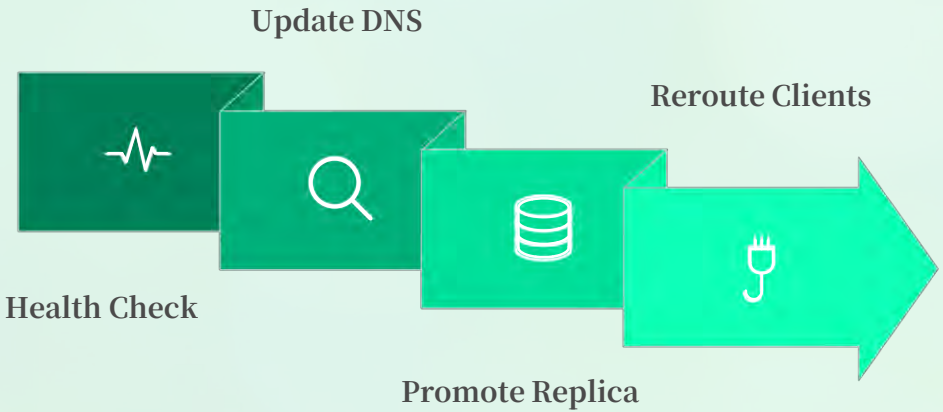
Real-Time Health Routing

Health checks drive record changes - traffic shifts away from unhealthy endpoints automatically

Global Distribution

Each provider's anycast network reduces latency for geographically distributed database clients

How DNS Integrates With Database Replication



DNS is not just a naming layer - it is the traffic control plane that determines which database node clients reach. Aligning DNS update timing with replica promotion is essential to avoid split-brain and dirty reads during failover.

Read/Write Routing at the DNS Layer

Write Endpoint

Single CNAME or weighted record always pointing to the current primary. Updated immediately on promotion.

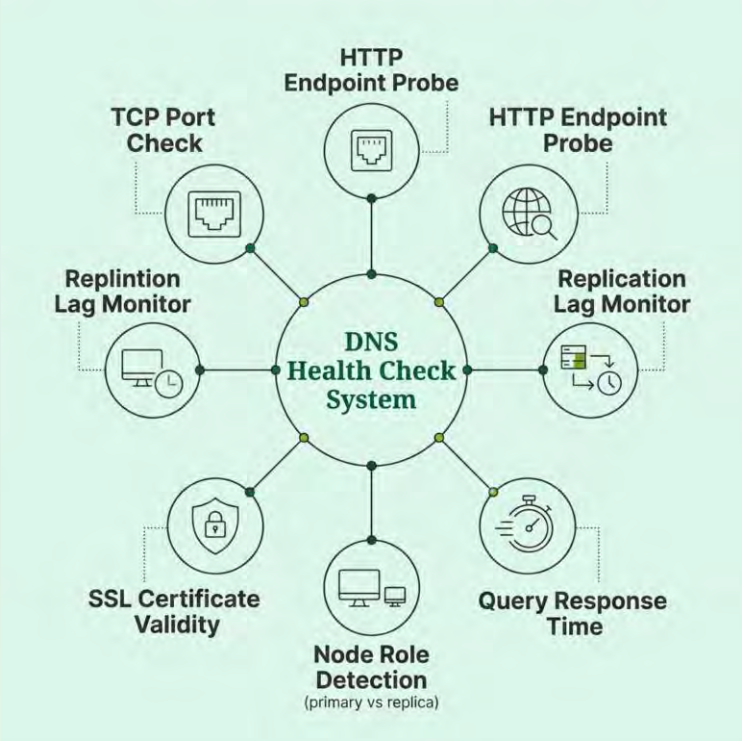
Read Endpoint

Round-robin or latency-weighted record across healthy replicas. Scales reads without application changes.

Regional Endpoint

Geo-routing directs clients to the nearest healthy replica, reducing cross-region read latency.

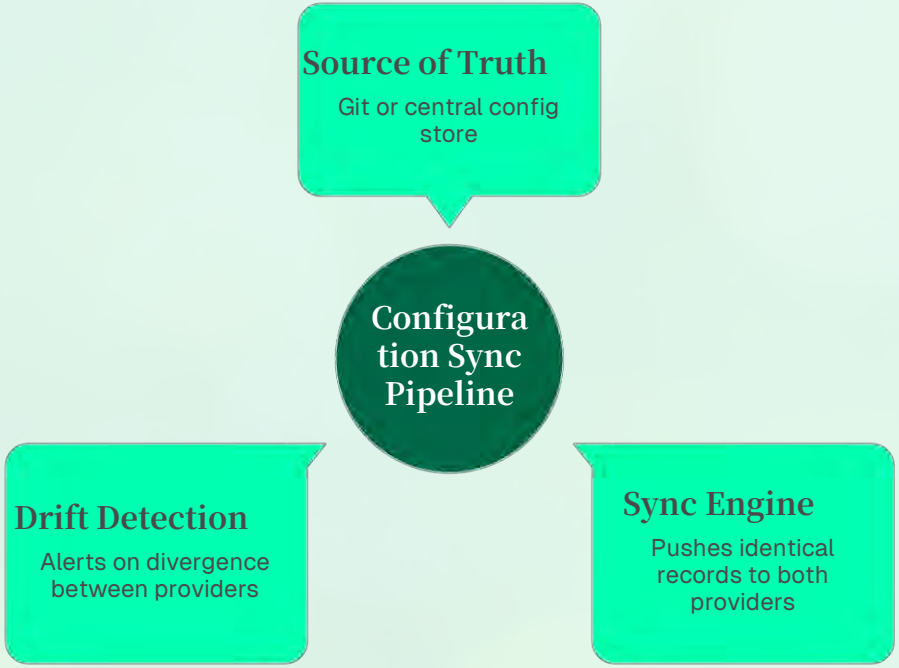
Health Checks: The Engine of Automated Failover



Active-Active DNS failover is only as fast and reliable as the health checks driving it. Checks must validate **application-level** database health, not just network reachability.

- Check interval and failure threshold determine your effective RTO
- Replication lag thresholds prevent promoting stale replicas
- Node role detection avoids routing writes to read-only replicas

Synchronizing Configuration Across Providers



Config drift between DNS providers is a silent reliability killer. Treating DNS records as code - stored in version control, deployed via CI/CD pipelines, and continuously validated - eliminates manual inconsistencies and ensures both providers stay in lockstep.

LABELS

OPERATIONAL PATTERN

Reducing Recovery Time Objectives

TTL Strategy

Lower TTLs (30–60 seconds) on database endpoints reduce the window where clients cache stale records. Balance this against resolver load and provider limits.

Pre-Warming Records

Keep replica endpoint records live and healthy at all times - even before a failover is needed. Avoids cold-start propagation delays during incidents.

Negative Caching

Configure SOA minimum TTLs to limit how long NXDOMAIN responses are cached by resolvers, preventing false "not found" states during record transitions.

Automation First

Manual DNS changes during incidents are too slow. Every failover path should be scripted, tested, and triggered automatically by health check state changes.

Observability Across DNS and Database State

Effective incident response requires correlating DNS resolution data with database health metrics in a single view.

DNS Query Metrics

Track resolution success rates, latency per provider, and record change propagation times

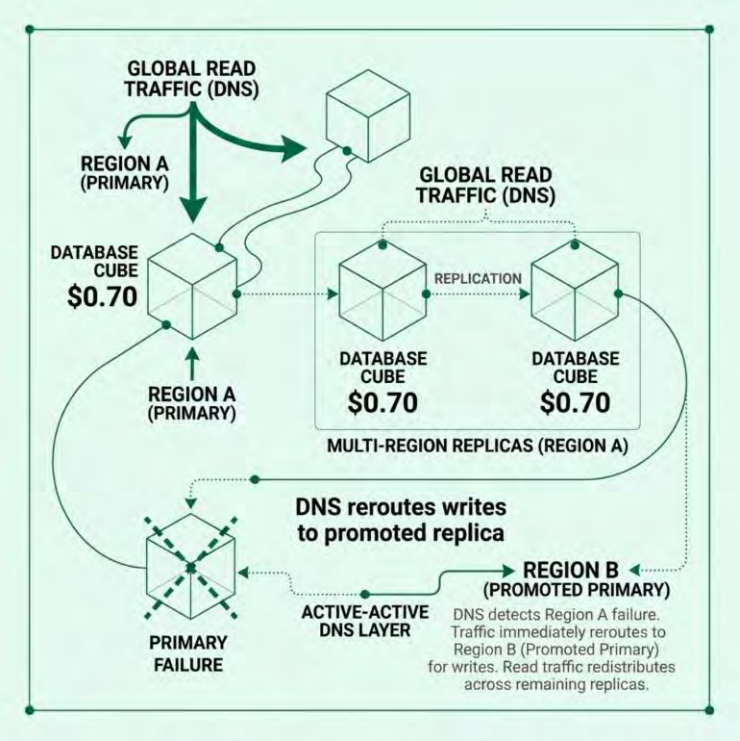
Replication Lag Visibility

Expose replica lag as a metric that both your database monitoring and DNS health check systems can consume

Failover Event Tracing

Log every DNS record change with timestamp, triggering health check state, and database node role at time of change

Global Failover: Multi-Region Considerations



Write Fencing

Ensure only one region accepts writes post-failover. DNS alone cannot prevent split-brain - coordinate with your database's own fencing mechanism.

Latency vs. Consistency

Geo-routing optimizes read latency but introduces replication lag. Define acceptable lag thresholds explicitly in your SLOs.

Pitfalls to Avoid

1

Assuming Low TTL = Fast Failover

Resolver caching behavior varies. Test actual propagation times in your environment, not just theoretical TTL values.

2

Health Checks That Are Too Shallow

A TCP connect check passes even on a read-only replica. Validate database role and replication state explicitly.

3

Ignoring Provider API Rate Limits

Automated failover scripts that hammer provider APIs during incidents can be throttled. Build in backoff and queuing.

4

No Runbook for DNS Failures

Active-Active reduces DNS as a single point of failure, but providers themselves can degrade. Have a manual fallback documented.

Key Takeaways

DNS is a first-class reliability concern

Treat it with the same rigor as replication topology and failover automation

Active-Active eliminates the DNS single point of failure

Multiple providers serving simultaneously means no provider outage can block database traffic routing

Health checks must reflect database reality

Surface replication lag, node role, and query health - not just network connectivity

Automate every failover path

Manual DNS changes during incidents are too slow and too error-prone to be part of your RTO

Thank You

Surya Lankalapalli
Software Engineer · Microsoft

Questions? Let's talk about DNS architecture, database failover patterns, or anything in between.