Designing for Platform Resilience in an Unpredictable World

June 2026

IBM

NetActuate

Table of Contents

Executive Summary

Major cloud and infrastructure outages are becoming more common and their causes more diverse. Platform engineering teams must assume failures will occur and design resilience across multiple layers to reduce blast radius and accelerate recovery.

This paper examines three complementary approaches to multi-layer resilience:

- **BGP Anycast at the networking layer** determines how traffic reaches the nearest viable entry point
- **Authoritative DNS at the mapping layer** determines which endpoints are eligible to be returned
- **Traffic steering at the decision layer** applies decision logic to select among eligible endpoints using performance, health, and business context

This paper explains how each layer works, the role each plays in delivering resilience, and how coordinated behavior across layers further protects availability and performance.

Developed by experts in Domain Name System (DNS) architecture and traffic steering at IBM NS1 and networking experts at NetActuate, this paper will help you design resilience across your entire stack.

While the approaches in this paper are vendor-agnostic, we draw on real operational patterns from global Anycast networks and DNS-based traffic steering platforms used by large service providers and enterprises to reduce blast radius and speed recovery.

Modern outages illustrate why a comprehensive, multi-layer approach matters. A network event may isolate an entire region. A DNS control-plane failure may prevent traffic steering changes. A content delivery network (CDN) or cloud provider disruption may make a large portion of capacity unavailable. At the application layer, a bad deploy or resource exhaustion can produce partial service degradation that routing layers cannot detect. Each of these failure modes requires a different mitigation path.

Recent outages reveal rising complexity and risk

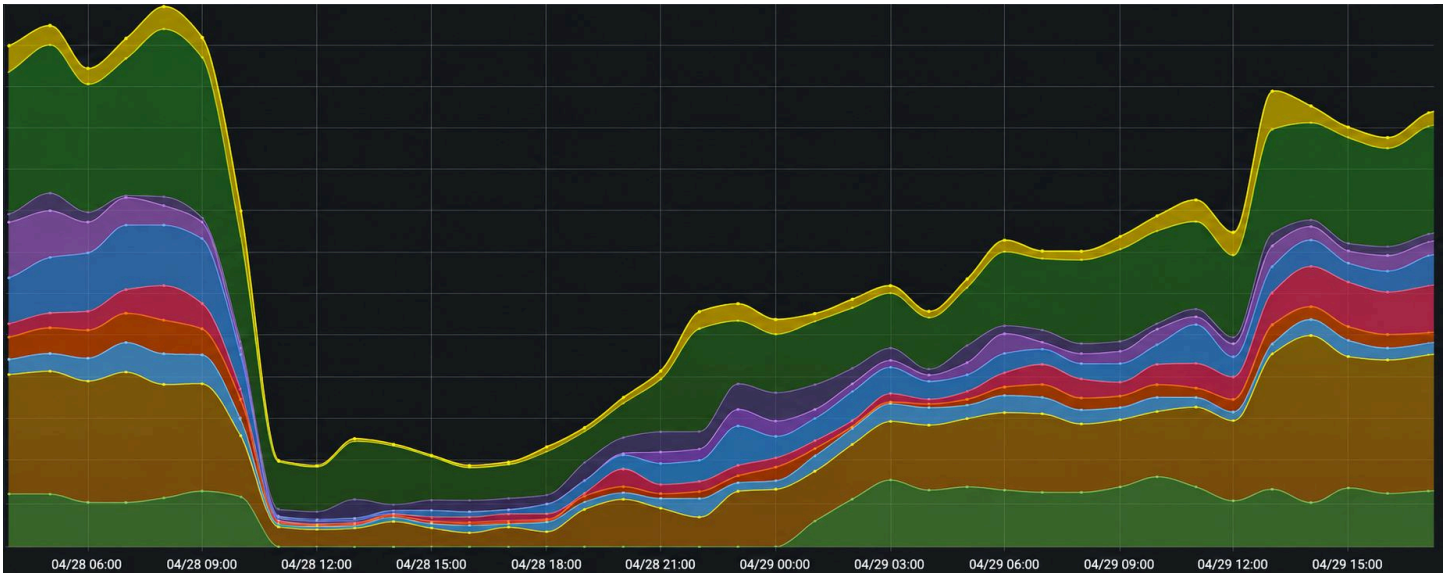
In ISACA's State of Cybersecurity 2025 report, 63% of the 3,800 cybersecurity professionals surveyed cited the complex threat landscape as their leading stressor.[1] IMF Managing Director Kristalina Georgieva echoed this sentiment: "My advice to policymakers in this new global environment is think of the unthinkable and prepare for it."[2] Recent geopolitical instability and regional conflicts have underscored that digital infrastructure is a real-world target, alongside energy and transportation. Enterprise leaders must design platforms that remain available when unexpected regional disruptions impact connectivity, cloud capacity, or dependencies.[3, 4]

A clear example of this came in April 2025, when the Iberian Peninsula blackout demonstrated how failures outside the traditional network stack can cascade directly into internet outages. A sudden grid disturbance caused a rapid loss of power across much of Spain and Portugal, knocking millions offline as data centers, IXPs, and access networks lost electricity simultaneously.

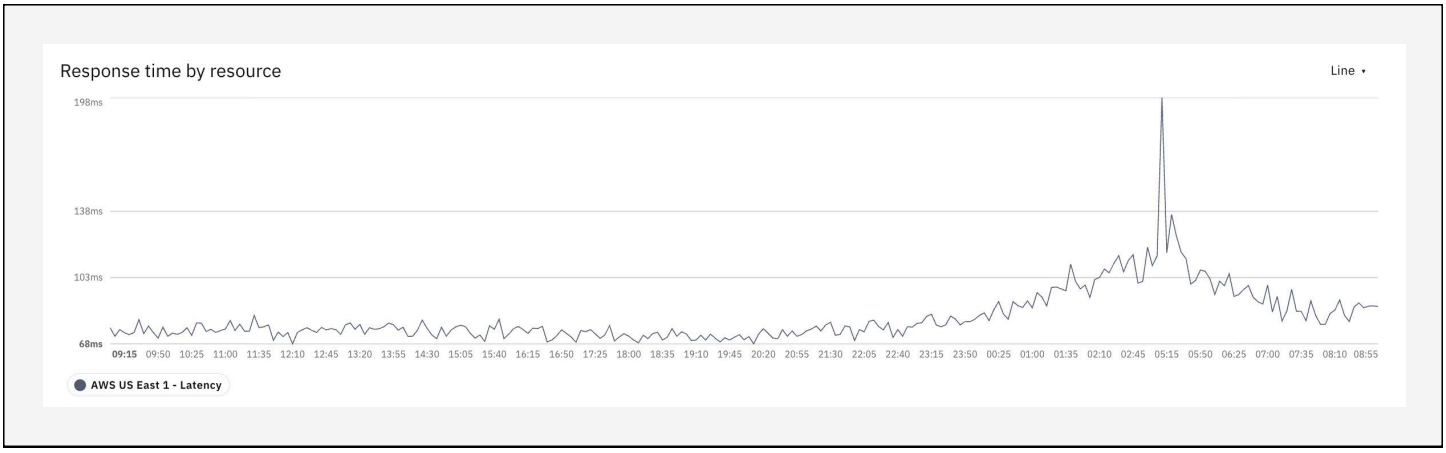
Even well-engineered telecom infrastructure ultimately depends on power availability; once backup systems are exhausted or unevenly distributed, connectivity degrades quickly across regions.

As Madrid and the surrounding region experienced rolling failures, NetActuate's global Anycast platform automatically redirected traffic to nodes in Paris and London, maintaining service availability for users even when large parts of the grid were offline.[5]

In another tangible example, the October 2025 AWS US-EAST-1 outage illustrated how failures within a hyperscale cloud provider can rapidly propagate across large portions of the internet. A defect in internal DNS automation supporting DynamoDB disrupted service discovery in the Northern Virginia region, triggering cascading failures across dozens of core AWS services, including EC2, Lambda, and IAM. Because US-EAST-1 hosts a significant share of global workloads, the disruption rippled outward to thousands of dependent platforms—from messaging apps and gaming services to financial and enterprise systems—demonstrating how tightly coupled cloud control planes and application stacks have become.[6]



NetActuate's view of internet traffic disruption during the Iberian power outage



IBM NS1's community Real User Monitoring (RUM) data capturing latency spikes to AWS US EAST-1 during the service outage (times in EDT)

Inadvertent, malicious, or suspicious undersea cable disruption presents another risk. Hundreds of such cables, which carry over 99% of intercontinental data traffic, are disrupted or cut hundreds of times every year by fishing vessels or dredging. But on November 17, 2024, a data cable connecting Sweden and Lithuania and another connecting Finland and Germany were damaged in a suspicious incident. In parts of the world with redundant cables, these incidents rarely result in complete outages, though they do increase latency and packet loss.[7]

These incidents highlight a key lesson: the combination of high threat level and multivariate causes of outages means that resilience must be designed across multiple layers of the stack. Network routing, DNS, and traffic steering each play a critical role in determining whether and how failures propagate and how quickly traffic can be rerouted and recovered.

BGP Anycast: The Resilient Network Foundation

With BGP Anycast, multiple geographically distributed systems share the same IP address, enabling the network to direct users to the most appropriate service instance based on network and infrastructure conditions.[8]

Because Anycast relies on the same foundational protocols that underpin the Internet, its behavior is governed by well-established standards. This reduces dependence on proprietary control planes and helps avoid vendor lock-in.

Anycast announces the same IP prefix from multiple locations using the Border Gateway Protocol (BGP), which governs interdomain routing across the global internet. Each participating site—typically a point of presence (POP) in a data center—advertises the identical prefix to upstream providers and peers. Routers across the internet then apply their normal BGP decision processes to determine the "best" path to that prefix.

In practice, this typically results in users being directed to the topologically closest or lowest-cost location. BGP path selection is a largely standardized process and experienced Anycast network operators know how to work with these standards to achieve the desired routing results for their service.

Because routing decisions are made dynamically by BGP, Anycast improves operational resilience. If connectivity to a region degrades—whether due to a fiber cut, transit provider failure, or localized infrastructure issue—traffic shifts to other sites. Operators can also intentionally withdraw announcements from a location during maintenance or incidents, causing traffic to flow elsewhere. Using BGP Communities, network tuning, and vigilant monitoring, Anycast reduces latency and improves application availability.

From network behavior to application design

Anycast is transparent to clients—they simply connect to a single IP address or hostname regardless of their location. Behind the scenes, though, that single address corresponds to multiple, dozens, or hundreds of distributed service instances across multiple regions.

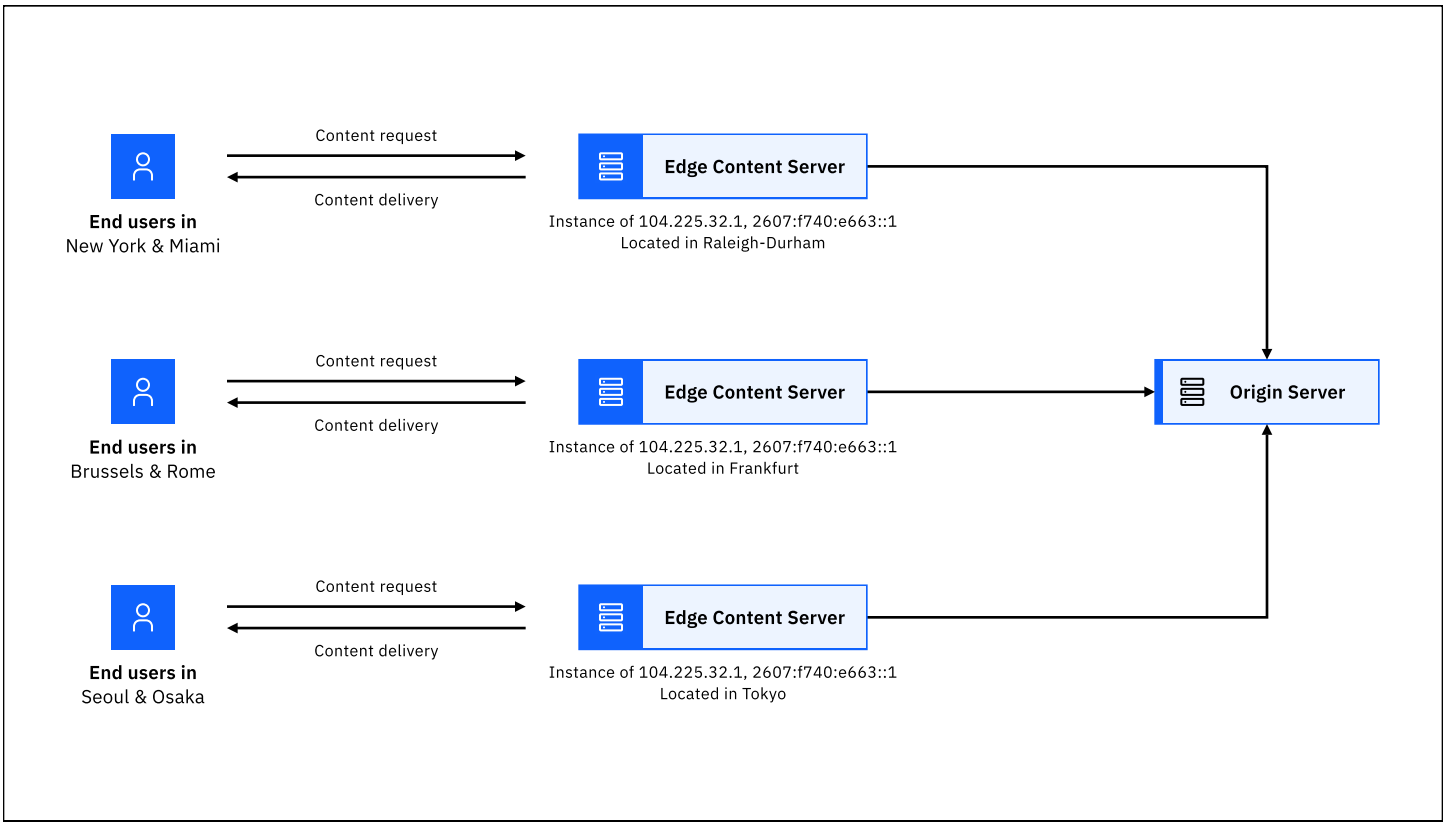
Successful Anycast deployments require careful operational practices. Prefix ownership, autonomous system numbers (ASNs), routing policy, and route validation mechanisms such as RPKI all play important roles in ensuring that announcements are trusted and consistently propagated across the internet.

Additionally, since Anycast routes users to the nearest available instance rather than a specific backend, applications must tolerate requests arriving from different locations. Stateless services, replicated data stores, and distributed caching are common architectural patterns used to support this model.

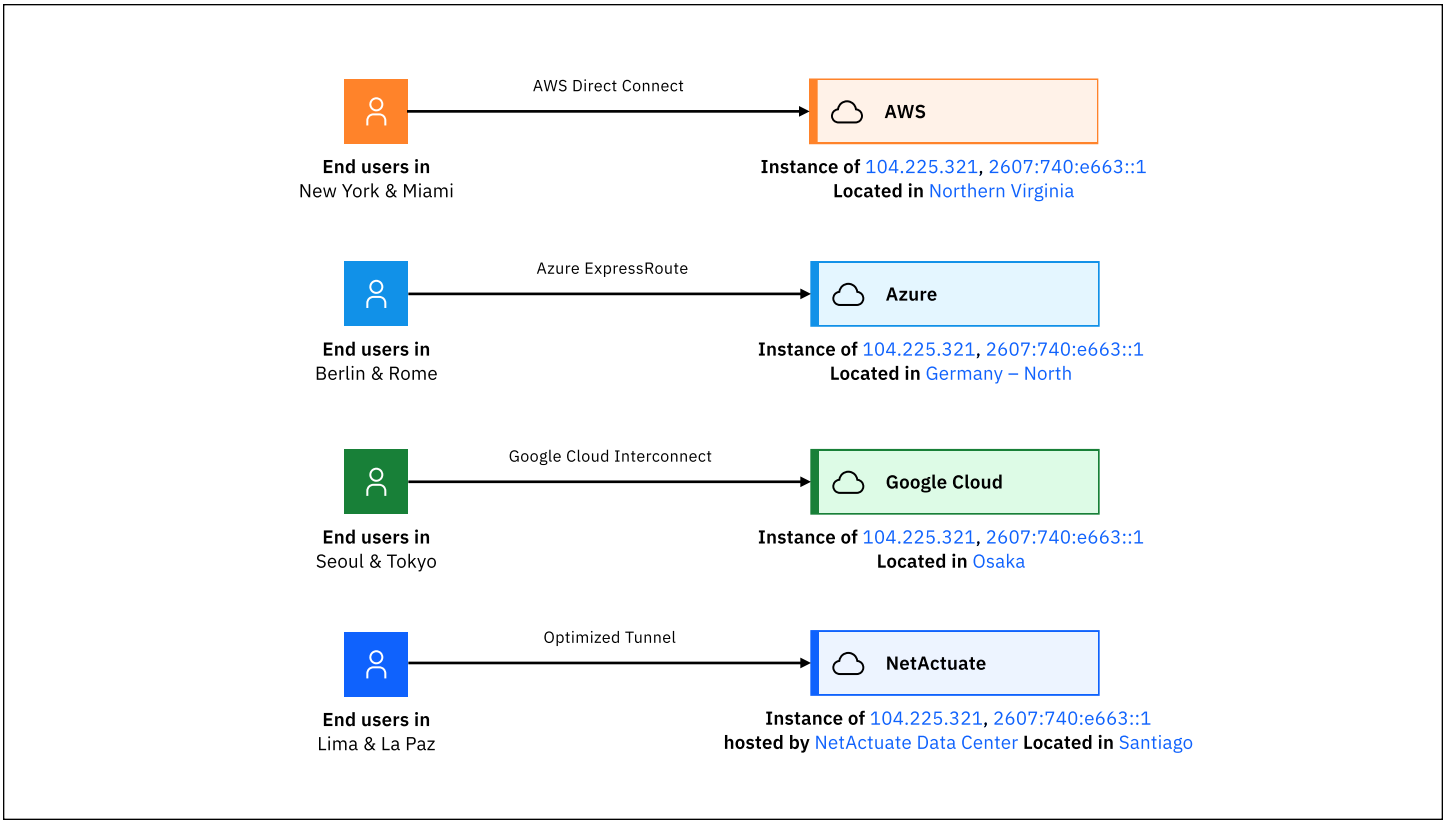
Single address, distributed system

Many critical internet services rely on Anycast to achieve global reach and resilience. For example, the global DNS infrastructure—including root servers, top-level domain registries, and public recursive resolvers—relies heavily on Anycast to distribute query traffic and place capacity close to end users. Public DNS resolvers such as Quad9's "9.9.9.9" or Cloudflare's "1.1.1.1" represent single IP addresses to billions of devices, but in reality those addresses are served simultaneously from many data centers around the world. The routing system ensures that queries are typically answered by a nearby instance.

CDNs, security platforms, and cloud load-balancing services follow similar patterns. Edge nodes in multiple regions advertise the same Anycast addresses, allowing users to connect to nearby ingress points. Once traffic enters a provider's network, additional layers of routing and policy—such as private backbones, application-layer load balancing, or service meshes—can determine how requests are handled internally. In this way, Anycast often serves as the Layer-3 entry point to a much larger distributed system.



Anycast for CDNs



Anycast for Global Load Balancers

Anycast complements other forms of load balancing and traffic management. BGP determines which site receives the initial connection, routing to the best available POP, and then traffic needs to be distributed within each location using traditional mechanisms such as ECMP, L4 load balancers, or application-level routing. In many deployments, Anycast acts as the outermost routing layer, while regional infrastructure handles session management, caching, and service orchestration.

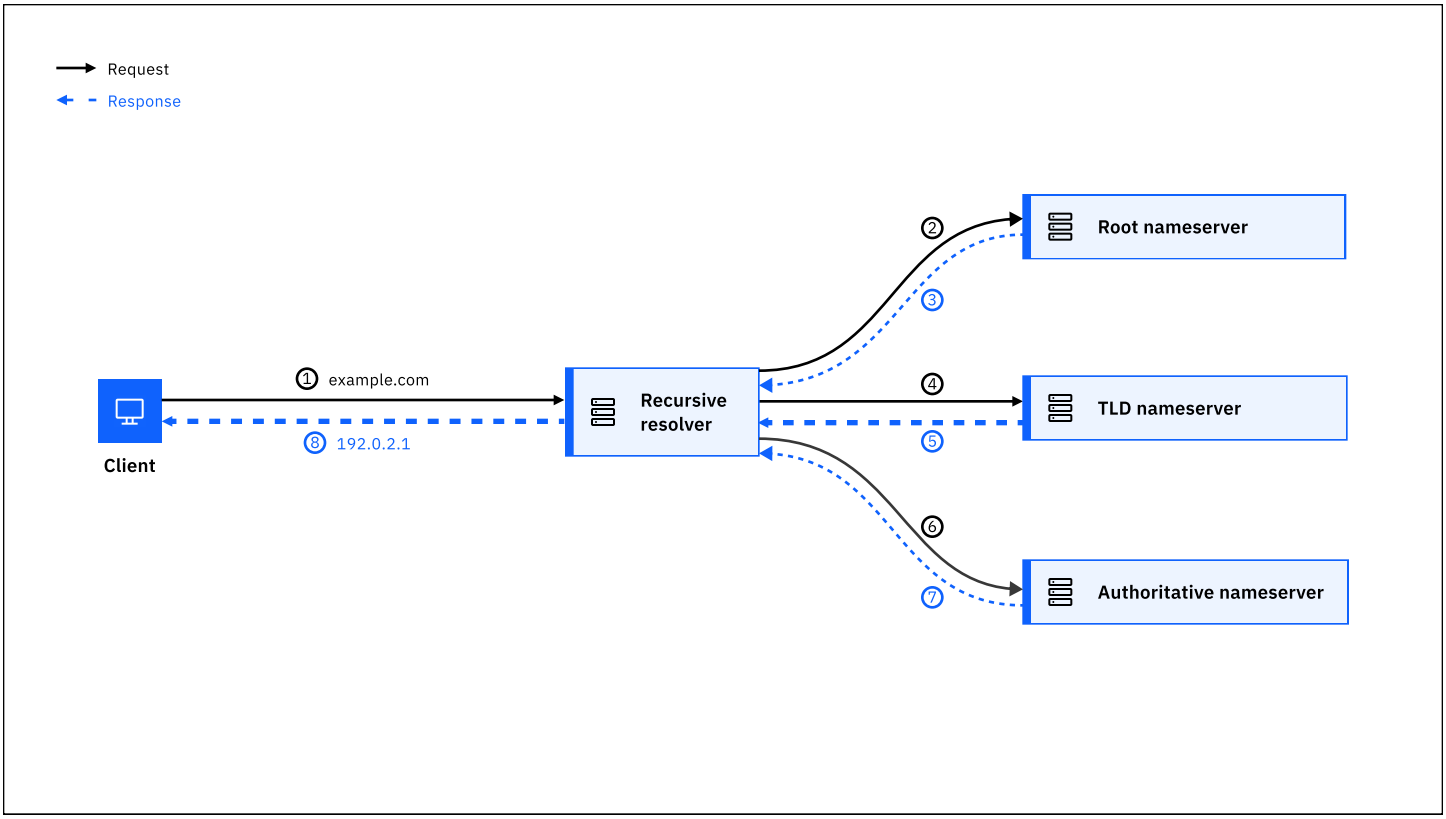
Anycast provides a powerful primitive: a single, globally reachable entry point backed by distributed infrastructure. By shifting some aspects of traffic steering into the routing layer, operators can reduce latency, distribute load across regions, and improve resilience against localized failures.

Authoritative DNS: The Mapping Layer

Anycast provides the resilient network foundation for global services and applications, which complements the mapping layer provided by authoritative DNS. This layer determines where users can go.

DNS has been a core and critical part of the internet path since the 1980s, when it replaced manual host files with a distributed, hierarchical system capable of scaling with a rapidly growing network of connected machines and servers. For decades, authoritative DNS has served as the source of truth for how users find applications and services.

DNS maps easy-to-remember names to the IP (Internet Protocol) addresses that the routing layer needs. Authoritative DNS answers a simple question: which endpoints should the user be sent to?



DNS query resolution

Unlike recursive resolvers, which retrieve and cache answers on behalf of clients, authoritative DNS servers publish the records that operators control for a public audience.

That distinction is especially important in resilient architectures. Because authoritative DNS sits before and outside the application's data path, responding to queries from recursive resolvers, it can influence user traffic before any connection is established. This makes it a powerful tried and tested mechanism for delivering answers.

Limits of DNS

Because DNS answers are cached by resolvers, operating systems, browsers, and applications, failover depends not only on configured TTLs but also on real-world resolver behavior and internet conditions. As a result, some changes propagate quickly, while others do not. DNS, therefore, should not be treated as a substitute for network reachability.

Instead, it works best as a policy-driven control plane for steering new connections and shifting demand over seconds to minutes.

Operating DNS for resilience

To be effective, authoritative DNS must be operated with the same rigor as any other resilience layer. Health signals must be reliable, and fallback policies must be safe and automated. Secondary DNS for redundancy, zone transfers, DNSSEC, and emergency drills should all be tested in advance. In today’s world, authoritative DNS is no longer just a lookup system; it’s a mechanism to implement changes, respond to queries quickly and meet the needs of the modern internet infrastructure.

DNS-based traffic management operates outside the application data path and serves as a control-plane alternative to network level routing.

This model is particularly well suited to multi-cloud, hybrid, and multi-CDN environments, where organizations need predictable, provider-agnostic control over application entry points across regions and platforms.

Layered resilience can be achieved through routing-layer mechanisms, DNS-layer control, or a combination—with the right mix determined by how failures occur and how quickly control must adapt. Anycast can preserve reachability, while DNS can apply intent and deterministic control. By aligning fast, infrastructure-driven failover with app-aware traffic management, organizations can reduce blast radius, avoid single points of failure, and withstand outages.

By aligning fast, infrastructure-driven failover with application-aware traffic management, organizations can reduce blast radius, avoid single points of failure, and better withstand diverse outage scenarios.

Traffic Steering: The Decision Layer

If authoritative DNS provides the framework for directing users to the right environment, traffic steering turns that framework into live, intelligent, and dynamic operational decisions. The idea is not new. Historically, much of this function was handled inside the enterprise perimeter by appliance-based load balancers and application delivery controllers deployed in front of application servers or at datacenter ingress. Hardware-based appliances such as F5 or NetScaler, first introduced in the late 1990s, gave operators increasingly sophisticated control over session handling, health monitoring, and local traffic distribution.

These systems were highly effective for balancing traffic across servers and enforcing policy within a site or regional footprint, but that control generally began only after traffic had already been routed to a particular network, datacenter, or ingress point.

In practice, DNS-based steering effectiveness varies widely by platform. The difference isn't whether a provider can return multiple answers—it's whether operators can express deterministic policy; ingest real telemetry on health, performance, and user experience; and enforce safe fallbacks when signals degrade.

From load balancing to global traffic control

As internet-scale applications expanded across multiple data centers, clouds, and CDNs, a newer model emerged: often using managed authoritative DNS as a globally distributed traffic-control plane.

Since modern applications are distributed across multiple clouds, CDNs, regions, and service providers, user experience can vary widely depending on latency, congestion, provider health, and application capacity. DNS-based traffic steering emerged to manage that complexity by giving operators a way to shape traffic flows intentionally.

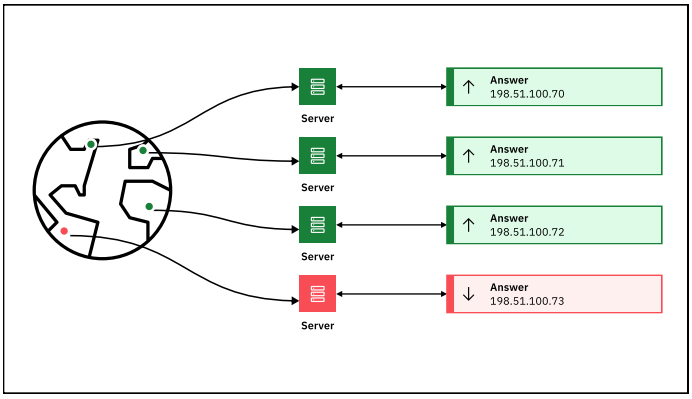
Defining traffic steering

At its core, traffic steering answers a practical question: when more than one viable destination exists, which one should receive this user's request right now? In simple architectures, that decision may be based on geography or round-robin distribution. In more advanced environments, it may depend on real-time health checks, latency measurements, regional capacity, application error rates, or business rules.

- Anycast determines how traffic reaches the nearest viable entry point at the network layer.
- Authoritative DNS determines which answers are eligible to be returned based on predetermined policies.
- Traffic steering applies more granular decision logic to select among

This distinction matters in practice because modern outages are rarely absolute. More often, systems degrade unevenly. One CDN may be reachable but slow in a specific geography. One cloud region may be healthy for most workloads but unable to serve workloads with high throughput. A service may appear “up” while still delivering a poor user experience. Rather than answering “is this endpoint available?”, traffic steering answers “is this endpoint the best choice at the moment, for this user, under these conditions?”

A platform team may want users in Western Europe to prefer one cloud region under normal conditions, shift to another provider if latency rises beyond a threshold, route certain countries to a specific environment for compliance reasons, or drain a degraded CDN without changing application code. These are not routing decisions in the pure BGP sense; they are business and application policy decisions executed through traffic steering at the naming layer. DNS-based steering is especially valuable in multi-cloud, hybrid, and multi-CDN architectures, where deterministic control matters as much as simple reachability.



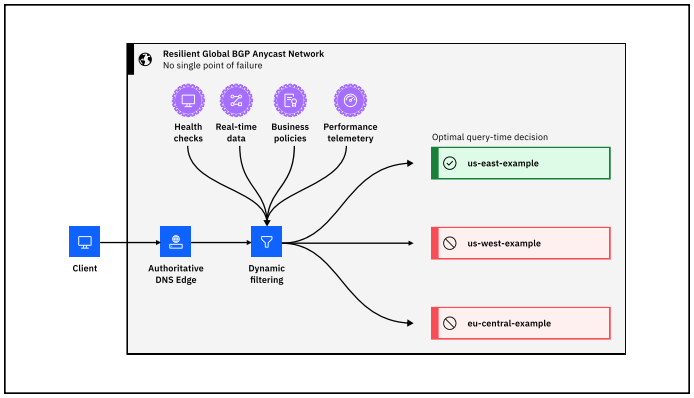
Traffic steering for distributed services

How traffic steering makes decisions

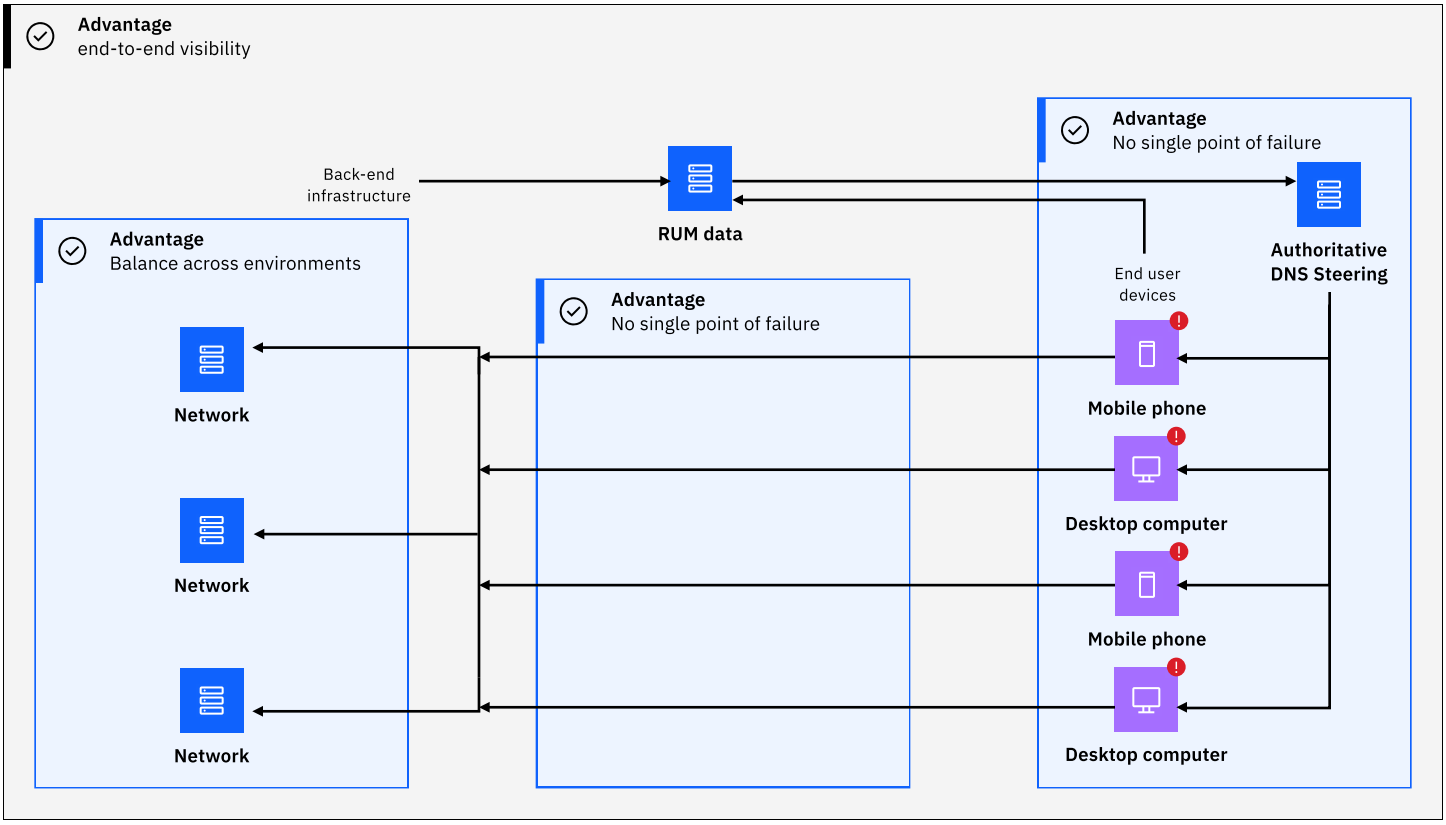
In practice, traffic steering typically combines signals:

- Health checks identify hard failures such as an unreachable endpoint or a provider outage.
- Performance telemetry measures relative quality, such as latency.
- Real-user measurements provide a direct view of end-user experience across networks and geographies.
- Operational data feeds add internal context, such as regional saturation, or application-specific limits.

Together, these inputs allow steering policies to be both adaptive and controlled. Operators can define thresholds for when traffic should move, set priorities among providers, and ensure that changes occur in a deterministic way.



Optimized traffic decisions based on dynamic input signals and business context



Balancing across delivery networks with Real User Monitoring (RUM) signals.

Traffic steering in multi-provider environments

Traffic steering becomes a necessity in multi-CDN, multi-cloud, and hybrid environments. When a platform depends on multiple delivery or compute providers, the challenge is no longer just redundancy; it is deciding how to use that redundancy. During an incident, steering policies can shift traffic away from a degraded provider, drain a region, or prioritize stable endpoints while preserving a controlled user experience. In such conditions, traffic steering is a continuous optimization mechanism.

Like DNS more broadly, traffic steering does not operate at the granularity of every packet or every transaction. It is a control function applied at the level of connection initiation, request routing, or client selection, depending on the architecture. Thus, its effectiveness depends on policy design, telemetry quality, and the speed at which clients or resolvers adopt new decisions.

A goal of traffic steering should not be to chase every momentary fluctuation in metrics. Good steering systems avoid thrashing by using thresholds, dampening, safe defaults, and clearly defined fallback paths.

For resilient global platforms, traffic steering is best viewed as the layer that translates resilience objectives into action. With Anycast providing reachability and authoritative DNS providing a framework, traffic steering provides decision quality.

It is the mechanism that lets organizations express intent such as “send users to the best-performing CDN,” “drain this region before it fails,” or “shift traffic only if user experience falls below a defined threshold.” This capability reduces blast radius, improves recovery flexibility, and enables businesses to adapt to both sudden outages and degradations.

Designing the Layers to Work Together

In resilient architectures, each layer does the job it's designed for. Not every resilient platform requires every layer discussed in this paper. Effective designs select the combination of network, DNS, and traffic steering mechanisms that best match their failure modes, control requirements, and operational maturity. In many large-scale architectures these layers are combined, but they can also be applied independently depending on which problems a platform needs to solve.

When deployed together, a resilient stack emerges from the interaction of these layers. Network routing, DNS policy, and traffic steering each operate on different time scales and failure domains. This, in turn, means that the rest of the platform must be ready for the resulting load shifts. A failover site cannot be sized only for its steady-state footprint. Where full duplication is inefficient, operators need strategies such as excess headroom or backhaul to stronger regions.

DNS and traffic steering then provide the control needed to turn reachability into usable service continuity. If a network event shifts users into a different ingress region, authoritative DNS and traffic steering policies can still return endpoints that match geography, policy, and deployment intent.

This is especially important in partial-failure scenarios: the endpoint that is reachable is not always the endpoint that should receive more traffic. A provider may be up but degraded, a region may be healthy but near saturation, or an application may be passing basic health checks while delivering poor user experience. In those cases, traffic steering should make stable, threshold-based decisions using layered inputs such as health checks, performance data, and operational feeds.

Coordinated behavior under stress often includes mechanisms such as Anycast to maintain ingress continuity. DNS can preserve deterministic control, including during telemetry or decisioning degradation, when designed with safe defaults and fallbacks. Traffic steering adapts to demand without introducing volatility, so that the application platform can tolerate the consequences:

- Sudden regional load increases
- Asymmetric paths
- Stale DNS answers
- Degraded dependencies

Organizations that do this well test how the layers interact, define fallback behavior in advance, and ensure that no single control-plane decision can create a larger failure. That is what turns three separate resilience mechanisms into a true resilience stack.

SREs and Platform Engineers may consult the accompanying SRE Checklist for a comprehensive and practical inventory of actions to keep globally-distributed applications available and performant.

SRE Checklist for Multi-Layer Resilience

This checklist is intended for SREs, platform engineers, and network architects responsible for operating globally distributed services. It is organized by **failure domains and control layers**, reflecting how outages propagate through modern systems.

1. Failure Awareness and Blast Radius Mapping ☐

Objective: Understand where failures can occur and how far they can spread.

Document common outage classes relevant to your stack:
- Hyperscaler or cloud control-plane failures
- Backbone or regional network outages (e.g., fiber cuts, power outages, routing leaks, conflict zones)
- DNS resolution or authoritative service failures
- CDN or edge service disruptions
- Application-layer failures (resource exhaustion, misconfiguration, bad deploys)

For each outage class, identify:
- Expected blast radius (global, regional, zonal, service-specific)
- Mean time-to-detection (MTTD)
- Mean time-to-recovery/mitigation (MTTR/MTTM)

Validate that the critical user journey can survive the loss of any single dependency, or explicitly document where it can't and add a mitigation (fallback, bypass, manual override, or reduced-mode).

2. Network-Layer Resilience (Routing and Reachability) ☐

Objective: Ensure basic reachability survives large-scale or upstream failures.

Use geographically distributed ingress points where possible.

Ensure routing decisions are resilient to:
- Regional network failures
- Backbone congestion or instability
- DDoS or other volumetric attacks (distribution, upstream mitigation, and capacity headroom)

Confirm that ingress services are:
- Stateless or safely replicated
- Able to tolerate asymmetric routing or explicitly enforce symmetry (and confirm it still holds during failover)

Test failure scenarios where:
- Entire regions are withdrawn
- Paths reconverge unpredictably

Measure and understand routing convergence behavior and limitations.

3. DNS and Control-Plane Resilience ☐

Objective: Maintain predictable, policy-driven control over traffic during partial or upstream failures.

Ensure DNS routing logic can:

- Detect upstream endpoint or provider failures.
- Incorporate health, performance, and policy signals.
- Define “safe defaults” for DNS decisioning degradation: define what answer set DNS should return when upstream telemetry or feeds are missing (e.g., last-known-good, static priority order, or cost-based default) and use DNS-based traffic steering ordering to ensure deterministic fallback paths.
- Prefer “graceful degradation” in the control plane: if advanced steering inputs fail (Real-user monitoring/RUM, custom data integration feeds), ensure records still resolve using fallback policies (e.g., geo, failover, or load-balance) placed later in the chain, for cascading failover options.
- Avoid a single point of failure in authoritative DNS for critical services. Implement multi-authority redundancy using an approach aligned to both your operating model and the control-plane features you require, such as secondary DNS with zone transfer, dual-primary (active/active) publishing, or synchronized authoritative architectures.
- Continuously validate multi-authority consistency and recovery readiness. Monitor for drift (e.g., version, serial, or config mismatch) and maintain a tested rollback path to a known-good state.

Testing your DNS posture -- Validate TTLs and real-world resolver behavior align with:

- Failover speed requirements (test change uptake, not just TTL on paper)
- Operational risk tolerance

4. Load Balancing and Traffic Steering ☐

Objective: Control where traffic goes and why—not just whether it flows.

Confirm traffic steering supports:

- Multi-region architectures
- Multi-cloud or hybrid deployments
- Multi-CDN strategies (if applicable)

Ensure routing decisions can be:

- Policy-driven (e.g., cost, geography, compliance)
- Performance-aware (e.g., latency, error rates)
- Health-aware (e.g., dependency-level failures)

Avoid static routing assumptions during failure conditions.

Validate that traffic steering changes can be executed safely under load.

Automate steering changes safely:

- Manage steering policies via API and/or Terraform with Ansible to reduce manual incident risk and to support repeatable rollback.
- Pre-stage “incidents” such as “CDN A global failure,” “Region X drain,” “Cost-optimized,” and switch via automation or runbooks rather than editing in-console under stress.
- Use client-side decisioning where appropriate to enable dynamic CDN switching closer to the user, as an additional steering lever beyond server-side DNS behavior.

Multi-CDN/multi-cloud steering patterns

- Steer across CDNs and clouds using RUM signals to inform routing decisions, directing traffic to a better-performing CDN, not just the closest provider on a map.
- Use RUM with health checks and data feeds for layered decisions: RUM evaluates user experience, health checks catch hard failures, data feeds represent business logic or app constraints (e.g., capacity, error rate, buffering, cost).
- Define "thresholds," such as "only move traffic if latency degrades by X%" or "only shift if availability falls below Y%," and then enforce via automated policies.
- Prepare for manual overrides and changes for on-the-fly decision making when necessary.

5. Application-Layer Resilience ☐

Objective: Prevent localized failures from escalating into global incidents.

Ensure applications degrade gracefully:

- Feature flagging
- Read-only or reduced functionality modes

Validate dependency isolation:

- Timeouts and circuit breakers
- Bulkhead or cell-based patterns for shared resources

6. Layer Interaction and Dependency Testing ☐

Objective: Verify that resilience mechanisms work together, not just in isolation.

Run failure drills that combine:

- Network failures and application stress
- DNS failover and regional capacity loss

Validate that:

- Network-layer failover does not overload healthy regions
- DNS-layer decisions align with actual application capacity

Confirm autoscaling behavior under:

- Sudden traffic shifts
- Failover-induced load spikes

Verify DNS cache behavior and how quickly resolvers pick it up (per region or ISP), then tune TTL and incident approach accordingly.

Identify and break incident-critical circular dependencies between:

- DNS
- Control planes (including identity and automation)
- Monitoring or alerting systems

Confirm autoscaling behavior under:
- Sudden traffic shifts
- Failover-induced load spikes

Test misconfiguration scenarios (e.g., bad deploys, bad flags, bad config pushes).

A/B test or use blue/green deployments with sufficient TTLs to adequately respond to application health.

Partner with application and platform SRE teams to identify relevant regional and global health metrics for global load balancing.

Design for controlled brownouts (e.g., when an app enters reduced-mode, flip a steering policy that prioritizes stable endpoints or reduces traffic to impacted regions until recovery).

7. Observability and Signal Quality ☐

Objective: Detect failures quickly and act with confidence.

Monitor from multiple perspectives:
- End-user experience
- Network reachability
- Application health

Avoid single-source observability dependencies.

Prioritize alerts on user impact and retain component-level alerts for early warning and faster diagnosis.

Design telemetry to degrade gracefully and remain accessible via independent paths, such as external probes and out-of-band checks, and validate what still works during major incidents.

8. Operational Readiness and Runbooks ☐

Objective: Reduce cognitive load during high-stress incidents.

Maintain clear runbooks for:
- Regional evacuation
- Provider failover
- DNS or routing overrides

Ensure on-call engineers understand:
- Which layers fail automatically
- Which require manual intervention

Regularly rehearse incident scenarios involving:
- Large blast radius events
- Partial, ambiguous failures

9. Continuous Review

Objective: Treat resilience as an ongoing practice.

Review architecture after every significant incident.

Reassess assumptions as providers, traffic patterns, and dependencies change.

Periodically validate that “resilient by design” still holds true in practice.

About IBM NS1 Connect and NetActuate

IBM NS1 Connect offers premium, authoritative DNS and advanced traffic steering to deliver the high-performance, reliable, secure network connectivity that businesses need to meet increasingly sophisticated customer expectations. NS1 Connect leverages its global anycast network to provide the massive capacity and scale needed to keep users reliably connected across the world. An API-first architecture empowers teams to embrace automation and streamline DNS management. Enterprises with complex network infrastructures can take performance to the next level with sophisticated traffic steering capabilities and real-time reporting on DNS observability data.

Learn more about IBM NS1 Connect →

Request a live demo →

NetActuate delivers edge infrastructure and network solutions in over 45 locations worldwide, enabling customers to take control of their workloads anywhere with low latency, resiliency, and security built in. Their Open Network Edge (ONE) IaaS built on open source software and standards provides everything needed to architect cloud environments and extensible infrastructure. Choose between VMs, Kubernetes, cloud, colocation, bare metal, and storage infrastructure—all with Anycast connectivity. They offer 24x7 support, expert consulting, and flexible, open solutions engineered for scalability and performance.

Learn more about NetActuate →

Schedule a call →

IBM

NetActuate

1. What's the state of cybersecurity in the coming year?, ISACA, 2025
2. IMF's Georgieva Warns Middle East Conflict Could Push Global Inflation Higher, Reuters, 2026
3. Nation-State Threats, CISA
4. Banking, payments services disrupted after Amazon UAE data centers hit in drone strikes, CNBC Mar 2026
5. How the Iberian Power Outage Tested Network Resiliency — and Why Anycast Matters, NetActuate Mar 2025
6. Summary of the Amazon DynamoDB Service Disruption in the Northern Virginia (US-EAST-1) Region, Amazon, Oct 2025
7. Law Can't Stop Submarine Cable Sabotage. Russia And China Know It., Feb 2025
8. Apply Anycast Best Practices for Resilient & Performant Global Applications, NetActuate Nov 2025

IBM Corporation
New Orchard Road
Armonk, NY 10504

Produced in the United States of America
June 2026

IBM

NetActuate