Fastly Built the Fastest Cache Invalidation in the Industry. The Authority to Invalidate Still Lives in One Place.

by Nick Clark | Published March 19, 2026 | PDF

Fastly's distributed multicast purge system achieves sub-150 millisecond global cache invalidation, a genuine engineering achievement in propagation speed. But propagation speed does not resolve the governance problem: every purge originates from a centrally authenticated API call, meaning the authority over cache state remains singular even as the execution mechanism is distributed. Scoped cache governance requires local invalidation authority, not faster central relay.

Fastly's cache invalidation is genuinely fast. Their documentation states a global average purge time of 150 milliseconds, frequently faster. Their engineering blog describes a decade of work building a distributed multicast system so that when a purge request arrives, cache servers broadcast it to each other directly rather than routing everything through a central coordinator. The result is near-simultaneous invalidation across a global network at a speed that competitors have not matched.

That engineering achievement is real. It is also orthogonal to the structural question this article is about.

What Fastly solved, and what it didn't

Fastly's distributed purging solves a propagation problem: how do you get an invalidation signal to every cache node in the world as fast as possible? Their answer — intercept the signal at the edge and use bimodal multicast to distribute it peer-to-peer among cache servers — is clever and it works. Their own blog is candid about why: a centralized coordinator for purge distribution introduced a single point of failure and higher latency. The distributed approach eliminates that bottleneck.

But the propagation mechanism and the governance of invalidation are two different things. Fastly solved propagation. The authority to issue a purge request still flows from a single source: api.fastly.com.

Every purge in Fastly's system originates as an API call authenticated against Fastly's control plane. You hit the Fastly API, authenticated with a Fastly API token, and the purge propagates from there. The rate limit is 100,000 purges per customer per hour, enforced by Fastly's infrastructure. The service configuration that determines what can be purged and how lives in Fastly's control panel. The VCL or Compute code that governs caching behavior is deployed through Fastly's configuration system.

Fastly has not distributed authority over cache state. They have distributed the mechanism for propagating changes to cache state once that authority has been exercised. The edge nodes are fast executors of centrally-originated decisions. They are not governors of their own cache.

Why the distinction matters

The practical consequences of this distinction show up in three specific places.

Scoped cache policy. Every cache object on Fastly is governed by a single service configuration. Surrogate keys allow fine-grained purging of tagged groups of objects, which is genuinely useful. But the policy governing what those keys mean, who can purge them, and under what conditions is defined in the service configuration and enforced by Fastly's control plane. There is no mechanism for a region of the cache to hold its own policy about what it serves and when its content becomes stale. The cache has rules; it does not have governance.

Jurisdictional isolation. A content policy change that applies to one jurisdiction — a regulatory requirement to stop serving certain content in a specific region, for example — propagates through the same purge mechanism as any other change. The purge system does not have a concept of scoped authority. Fastly's infrastructure is globally coherent by design: a purge issued at the API level propagates globally. Regional behavior differences must be implemented through routing logic and VCL conditionals, not through governance that is structurally local to the region.

Structural adaptation. When traffic patterns shift significantly — a region grows large, a new geography emerges as important, a service needs to be restructured — those decisions happen through Fastly's configuration and deployment system. The cache nodes adapt to instructions. They do not self-govern adaptation. The edge executes; the center decides.

What changes when cache governance is local

The alternative is not a slower purge system. Speed of propagation and location of authority are independent properties. The structural alternative is a cache layer where each scope holds its own governance: the nodes responsible for a region of content hold the policy for that region, cache mutations are proposed through those nodes and validated by local consensus, and the invalidation decision is made by the nodes governing the affected scope rather than by an API call that originates upstream.

In this model, Fastly's propagation mechanism — fast, distributed, peer-to-peer broadcast — is not displaced. It is the right answer to the propagation problem. What changes is the origin of the signal. Instead of an API call authenticated against a central control plane, the invalidation originates from an anchor-governed mutation proposal: a change to a content scope proposed by the nodes governing it, validated through local quorum, and propagated through the network from there. The authority is local. The propagation is fast.

The practical consequences are concrete. A regulatory requirement that applies to a single region is handled by the anchors governing that region's scope, not by a global purge that the operator then has to scope manually through VCL. A cache scope that grows beyond its current structure can split into child scopes governed by the relevant anchors, without a service configuration change deployed through a central control panel. A content mutation that affects one semantic scope does not require touching the governance infrastructure of any other scope.

The remaining gap

Fastly is arguably the most serious CDN engineering organization in the industry. The decision to build distributed purge propagation rather than rely on a centralized coordinator was the right call. Their documentation even names the problem directly: centralized coordination introduces a single point of failure and higher latency.

The next step in that same direction — distributing not just the propagation mechanism but the authority that originates invalidation decisions — is what the adaptive index provides. Each scope in the namespace is anchor-governed. Mutations to cache state within a scope are validated by the nodes responsible for that scope. The control plane does not disappear. It distributes into the namespace itself.

Adaptive Indexing All 21 steps →

Resolution without global consensus. Anchor-governed self-organization.

Patent
US 19/326,036 · published

Primary Technical Disclosure
○ The Adaptive Index: A Scalable Foundation for Decentralized Systems
Secondary Technical
○ Anchor-Governed Hierarchical Nesting: Recursive Semantic Containers at Unlimited Depth○ Entropy-Triggered Index Splitting: Deterministic Partitioning Under Mutation Load○ Dormant Index Merging: Recursive Consolidation of Low-Entropy Subindices○ Elastic Anchor Group Management: Governance That Scales With Criticality○ Trust-Weighted Quorum Voting: Consensus Where Weight Reflects Earned Trust○ Asynchronous Consensus Coordination: Offline Vote Completion With Reconciliation○ Best-Match Alias Querying: Longest-Match Resolution With Stepwise Delegation○ Action-Typed Aliases: Behavioral Intent Embedded in the Namespace○ UID Persistence Through Alias Mutation: Stable Identity Across Structural Change○ Lineage-Preserving Structural Mutation: Cryptographic History Through Every Change○ Proximity-Based Routing With Trust Scoring: Dynamic Path Selection in Decentralized Networks○ Dynamic Device Hash for Pseudonymous Authentication: Volatile Identity Without Stored Credentials○ On-Demand Adaptive Caching: Cache Instances That Follow Usage, Not Configuration○ Predictive Cache Prefetching: Forecasting Models That Proactively Instantiate Caches○ Contextual Access Enforcement: Policy Graphs Evaluated With Real-Time Telemetry○ Mutation Router With Contextual Signals: Policy-Aware Propagation Path Selection○ Impact Simulation During Mutation Staging: Pre-Execution Analysis of Proposed Changes○ DNS Bidirectional Fallback: Hybrid Resolution With Legacy DNS Compatibility○ Asset Versioning as First-Class Metadata: Version Entries Under UIDs With Lineage Tracking○ Telemetry-Driven Topology Mutation: Autonomous Network Reconfiguration From Operational Data
Applications (General)
○ Applying Adaptive Indexes to Legacy Decentralized Systems○ Why Edge Platforms Still Depend on a Central Authority○ Supply Chain Tracking Through Governed Namespace Resolution○ Social Media Platforms Without Central Namespace Authority○ Healthcare Data Federation Through Scoped Governance○ Government Identity Infrastructure at Scale○ Financial Market Data With Governed Resolution○ Gaming and Metaverse Namespace Governance
Applications (Specific)
○ Cloudflare's Edge Has a Namespace Problem○ DNS Is 40 Years Old and Still Running the Internet○ ENS Solved the Wrong Half of the Naming Problem○ Handshake Decentralized the Root. Everything Below It Is Still Ungoverned.○ IPFS Solved Content Addressing. It Didn't Solve Naming, Persistence, or Governance.● Fastly Built the Fastest Cache Invalidation in the Industry. The Authority to Invalidate Still Lives in One Place.○ Akamai Built the Internet's Delivery Infrastructure. It Was Designed for a World That Needed Central Control.○ Bluesky Identified the Right Problem. The Architecture That Solves It Is the Adaptive Index.○ Consul's Service Catalog Is Brilliant Infrastructure. It Is Still a Central Registry.○ Istio Solved Programmable Traffic Policy. The Namespace That Routes Traffic Is Still Central.○ Unstoppable Domains Proved NFT Ownership Works. The Namespace Governance Model Is Still Unresolved.○ The Graph Built the Index Layer for Web3. The Index Itself Still Has a Governance Problem.○ Filecoin Proved Verifiable Storage. Discovery and Namespace Governance Are Still Unsolved.○ Arweave Made Data Permanent. It Has No Governance Model for What Permanent Data Means Over Time.○ Ceramic Built Mutable Data Streams for Web3. The Governance of Those Streams Is Still Not Local.○ Kubernetes Service Discovery Resolves Within Clusters. Cross-Cluster Namespace Is Central.○ Amazon Route 53 Is the Most Reliable DNS on Earth. It Is Still DNS Architecture.○ HashiCorp Nomad Distributes Scheduling. The Namespace That Organizes It Is Still Central.○ ZooKeeper Coordinates Distributed Systems. The Coordinator Is a Single Point of Authority.○ etcd Stores the State of Kubernetes. The State Store Has No Scoped Governance.○ Consul KV Distributes Configuration. The Distribution Authority Is Still Central.○ Raft Made Consensus Understandable. It Did Not Make Consensus Scope-Aware.○ Paxos Proved Consensus Is Possible. It Did Not Address Namespace Governance.○ Cosmos Tendermint Enabled Sovereign Blockchains. The Namespace Between Them Is Ungoverned.○ AWS Cloud Map Discovers Services. The Discovery Authority Lives in One Region's Control Plane.○ Azure Traffic Manager Routes Globally. The Routing Authority Is Centrally Defined.○ GCP Service Directory Centralizes Service Registration. Registration Is Not Governance.○ Netlify DNS Simplifies Deployment Routing. The Namespace Authority Is Still Netlify's.○ Vercel's Edge Network Executes at the Boundary. Routing Authority Does Not.○ Bunny CDN Delivers Content Globally. Cache Governance Is Still Central.○ KeyCDN Optimized Content Delivery. The Delivery Namespace Is Centrally Controlled.○ Limelight Networks Built Private Infrastructure for Delivery. The Namespace Governance Is Still Central.○ StackPath Combined CDN With Edge Computing. Namespace Authority Remained Central.○ Envoy Proxy Made Service Mesh Data Planes Programmable. The Control Plane Still Governs.○ NGINX Powers the Web's Reverse Proxy Layer. Its Configuration Is Statically Defined.○ Traefik Discovers Services Automatically. The Discovery Namespace Is Still External.○ Linkerd Simplified the Service Mesh. The Namespace It Meshes Is Still Kubernetes.○ Namecheap Made Domain Registration Accessible. Domain Governance Remains the Registrar Model.○ GoDaddy Registered More Domains Than Anyone. The Namespace Model Has Not Changed.○ DNSimple Made DNS Management Developer-Friendly. The Governance Model Is Still DNS.○ Datadog Observes Everything. The Namespace It Observes Has No Governed Structure.○ Grafana Unified Observability Visualization. The Data Namespace It Queries Has No Governed Structure.○ Prometheus Defined Cloud-Native Monitoring. Its Metric Namespace Has No Governance Layer.○ New Relic Pioneered APM. The Telemetry Namespace It Built Is Centrally Indexed.○ Splunk Indexes Machine Data at Scale. The Index Namespace Is Centrally Administered.
Adaptive Indexing overview →
AQ
deterministic
autonomy

Legal

Subject to one or more pending U.S. and international patent applications, see Patents for the current list and status. No license, express or implied, is granted. Any use requires a separate written agreement—see Licensing. Patent applications referenced on this site are pending. Claim scope, if any, is subject to examination and may issue in altered form or not at all. See Legal for terms and conditions.

Adaptive Query™ is a trademark of Nicholas Clark. U.S. federal registration is pending. federal registration. AQ™ , AQ Inside™ , Adaptive Index™ , Adaptive Network™ , Semantic Agent™ , @AQ™ , AQID™ , and Adaptive Coin™ are used as trademarks in connection with the Adaptive Query platform and brand. Other names may be trademarks of their respective owners.

Last updated: 2026-03-03

-
- Inventive Steps
- Licensing
- Patents
- Articles
- Legal
- Opportunities
- Sitemap

-
- nick@qu3ry.net
- 72 28 14 36 01

Invented by Nick Clark | Founding Investors: Devin Wilkie