### **Netflix (Tech Blog)**

* Zuul @ the Edge — Netflix’in API ağ geçidi, istek yönlendirme, devre kesme, rate-limiting, çok bölge işlemleri. ([Discord](https://discord.com/blog/how-discord-stores-trillions-of-messages?utm_source=chatgpt.com))
* EVCache: Netflix’te cache katmanını ölçeklemek (çok bölgeli, milyonlarca RPS). ([Discord](https://discord.com/blog/how-discord-handles-two-and-half-million-concurrent-voice-users-using-webrtc?utm_source=chatgpt.com))
* “Performance Under Load” serisi — prod’daki yük altında performans ve kapasite yaklaşımı. ([Discord](https://discord.com/blog/how-discord-indexes-trillions-of-messages?utm_source=chatgpt.com))

### **Discord (Resmî Blog)**

* How Discord Stores Trillions of Messages — depolama, indeksleme, çok kiracılı mimari. ([The Cloudflare Blog](https://blog.cloudflare.com/he-il/rearchitecting-workers-kv-for-redundancy/?utm_source=chatgpt.com))
* Scaling Video Infrastructure — canlı medya/RTC altyapısını ölçekleme. ([The Cloudflare Blog](https://blog.cloudflare.com/rich-complex-rules-for-advanced-load-balancing/?utm_source=chatgpt.com))
* How Discord Scaled Voice — ses yığınını dünya çapında büyütme. ([The Cloudflare Blog](https://blog.cloudflare.com/webrtc-turn-using-anycast/?utm_source=chatgpt.com))

### **Meta/Facebook (Engineering)**

* TAO: Facebook’un sosyal graf verisi için dağıtık veri katmanı. ([Medium](https://medium.com/%40ilakk2023/400-live-system-design-interview-simulation-design-problem-build-twitter-x-e7bec4268d4c?utm_source=chatgpt.com))
* Scaling Memcache at Facebook — cache katmanı, tutarlılık ve kapasite. ([Medium](https://medium.com/%40karan99/system-design-twitter-793ab06c9355?utm_source=chatgpt.com))

### **Twitter/X (Engineering)**

* Observability at Twitter — yüksek trafik sistemlerde gözlemlenebilirlik mimarisi. ([techblog.netflix.com](https://techblog.netflix.com/2016/03/caching-for-global-netflix.html?utm_source=chatgpt.com))
* Building a Real-Time Analytics Stack at Twitter — gerçek zamanlı veri boru hattı. ([techblog.netflix.com](https://techblog.netflix.com/2012/11/hystrix.html?utm_source=chatgpt.com))

### **Uber (Engineering)**

* Cadence (şimdi Temporal’ın atası) — yüksek hacimli workflow/oryantasyon. ([engineering.linkedin.com](https://engineering.linkedin.com/distributed-systems/log-what-every-software-engineer-should-know-about-real-time-datas-unifying?utm_source=chatgpt.com))
* M3: Uber’in büyük ölçekli metrik platformu. ([engineering.linkedin.com](https://engineering.linkedin.com/metrics/scaling-collection-self-service-metrics?utm_source=chatgpt.com))

### **LinkedIn (Engineering)**

* Kafka benchmark: “2 Million Writes/sec (3 ucuz makinede)” — log/mesajlaşma omurgası. ([engineering.linkedin.com](https://engineering.linkedin.com/kafka/benchmarking-apache-kafka-2-million-writes-second-three-cheap-machines?utm_source=chatgpt.com))
* Running Kafka at Scale — LinkedIn’de Kafka’nın üretim kullanımı. ([engineering.linkedin.com](https://engineering.linkedin.com/kafka/running-kafka-scale?utm_source=chatgpt.com))
* Samza’yı ölçekli işletmek — LinkedIn’in gerçek-zamanlı stream işleme altyapısı. ([engineering.linkedin.com](https://engineering.linkedin.com/samza/operating-apache-samza-scale?utm_source=chatgpt.com))

### **Cloudflare (Engineering Blog)**

* “Millions of HTTP Requests per Second” — uçta (edge) ölçeklenebilirlik, anycast, L4/L7. ([Amazon Web Services, Inc.](https://aws.amazon.com/builders-library/using-load-shedding-to-avoid-overload/?utm_source=chatgpt.com))
* Anycast ve ağ mimarisi yazıları (BGP/anycast ile küresel dağıtım). ([Amazon Web Services, Inc.](https://aws.amazon.com/builders-library/workload-isolation-using-shuffle-sharding/?utm_source=chatgpt.com))

### **Pinterest (Engineering)**

* Scaling Cache Infrastructure at Pinterest — **150M+ RPS** cache katmanı (Memcached + mcrouter). ([Medium](https://medium.com/pinterest-engineering/scaling-cache-infrastructure-at-pinterest-422d6d294ece?utm_source=chatgpt.com))
* Manas arama altyapısının iki katmanlı fan-out mimarisi (migrasyon/kubenin ölçek etkileri). ([Medium](https://medium.com/pinterest-engineering/debugging-the-one-in-a-million-failure-migrating-pinterests-search-infrastructure-to-kubernetes-bef9af9dabf4?utm_source=chatgpt.com))

### **Slack (Engineering)**

* Migrating **Millions of Concurrent WebSockets** to Envoy — bağlantı uç katmanı. ([Engineering at Slack](https://slack.engineering/migrating-millions-of-concurrent-websockets-to-envoy/?utm_source=chatgpt.com))
* Slack’s Migration to a **Cellular Architecture** — blast radius sınırlandırma, yalıtım. ([Engineering at Slack](https://slack.engineering/slacks-migration-to-a-cellular-architecture/?utm_source=chatgpt.com))
* Load Testing with Koi Pond — mimari diyagram ve istek yolculuğu. ([Engineering at Slack](https://slack.engineering/load-testing-with-koi-pond/?utm_source=chatgpt.com))

### **Google (SRE Book — resmî)**

* Load Balancing at the Frontend / in the Datacenter & **Handling Overload** — Google’ın istek yönlendirme ve overload yönetimi pratikleri. ([sre.google](https://sre.google/sre-book/part-III-practices/?utm_source=chatgpt.com))
* Production Environment: **GSLB**, Stubby/gRPC, RPC mimarisi. ([sre.google](https://sre.google/sre-book/production-environment/?utm_source=chatgpt.com))

### **Amazon (The Builders’ Library — resmî)**

* **Timeouts, Retries, and Backoff with Jitter** — istemci/servis dayanıklılığı. ([Engineering at Meta](https://engineering.fb.com/2013/06/25/core-infra/tao-the-power-of-the-graph/?utm_source=chatgpt.com))
* **Avoiding Overload** — shed/reject stratejileri, koruma devreleri. ([Engineering at Meta](https://engineering.fb.com/2013/04/15/core-infra/scaling-memcache-at-facebook/?utm_source=chatgpt.com))
* **Shuffle Sharding** — izolasyon ve blast radius küçültme. ([Engineering at Meta](https://engineering.fb.com/2025/04/28/security/how-meta-understands-data-at-scale/?utm_source=chatgpt.com))

# **Netflix**

* Announcing EVCache — *distributed in-memory cache (memcached-based) used at Netflix*.  
  <https://techblog.netflix.com/2013/02/announcing-evcache-distributed-in.html>. ([techblog.netflix.com](https://techblog.netflix.com/2013/02/announcing-evcache-distributed-in.html?utm_source=chatgpt.com))
* Pushy to the Limit: Evolving Netflix’s WebSocket proxy for the future — *WebSocket/long-lived-connection ölçeklendirme (Pushy)*.  
  <https://netflixtechblog.com/pushy-to-the-limit-evolving-netflixs-websocket-proxy-for-the-future-b468bc0ff658>. ([netflixtechblog.com](https://netflixtechblog.com/pushy-to-the-limit-evolving-netflixs-websocket-proxy-for-the-future-b468bc0ff658?utm_source=chatgpt.com))
* Open Sourcing Zuul 2 / Zuul (Edge Gateway) — *API gateway / edge yük dengeleme ve connection yönetimi*.  
  <https://netflixtechblog.com/open-sourcing-zuul-2-82ea476cb2b3>. ([netflixtechblog.com](https://netflixtechblog.com/open-sourcing-zuul-2-82ea476cb2b3?utm_source=chatgpt.com))

# **Discord (resmî Discord mühendislik blogu)**

* How Discord Stores Trillions of Messages — *mesaj depolama evrimi: Mongo → Cassandra → Scylla; nedenler, tasarım kararları*.  
  <https://discord.com/blog/how-discord-stores-trillions-of-messages>. ([Discord](https://discord.com/blog/how-discord-stores-trillions-of-messages?utm_source=chatgpt.com))
* How Discord handles push request bursts of over a million per minute (GenStage) — *push/back-pressure ve load-shedding anlatımı*.  
  <https://discord.com/blog/how-discord-handles-push-request-bursts-of-over-a-million-per-minute-with-elixirs-genstage>. ([Discord](https://discord.com/blog/how-discord-handles-push-request-bursts-of-over-a-million-per-minute-with-elixirs-genstage?utm_source=chatgpt.com))
* How Discord Handles Two and Half Million Concurrent Voice Users using WebRTC — *RTC/medya ölçekleme*.  
  <https://discord.com/blog/how-discord-handles-two-and-half-million-concurrent-voice-users-using-webrtc>. ([Discord](https://discord.com/blog/how-discord-handles-two-and-half-million-concurrent-voice-users-using-webrtc?utm_source=chatgpt.com))

# **Meta / Facebook (resmî yayın ve konferans makaleleri)**

* TAO: Facebook’s Distributed Data Store for the Social Graph — *graf-veri mağazası, read-optimized, yüksek-işlem hacimleri için tasarım* (research/Usenix).  
  <https://www.usenix.org/system/files/conference/atc13/atc13-bronson.pdf>. ([USENIX](https://www.usenix.org/system/files/conference/atc13/atc13-bronson.pdf?utm_source=chatgpt.com))  
   (Ayrıca özet:<https://research.facebook.com/publications/tao-facebooks-distributed-data-store-for-the-social-graph/>). ([Facebook Research](https://research.facebook.com/publications/tao-facebooks-distributed-data-store-for-the-social-graph/?utm_source=chatgpt.com))
* Scaling Memcache at Facebook — *memcached ölçekleme, mcrouter vs tasarım kararları (USENIX / Meta mühendislik yazısı)*.  
  <https://www.usenix.org/system/files/conference/nsdi13/nsdi13-final170_update.pdf>. ([USENIX](https://www.usenix.org/system/files/conference/nsdi13/nsdi13-final170_update.pdf?utm_source=chatgpt.com))  
   (Engineering blog versiyonu:<https://engineering.fb.com/2013/04/15/core-infra/scaling-memcache-at-facebook/>). ([Engineering at Meta](https://engineering.fb.com/2013/04/15/core-infra/scaling-memcache-at-facebook/?utm_source=chatgpt.com))

# **Twitter / X (resmî mühendislik bloğu)**

* Manhattan — *Twitter’ın büyük ölçekli, çok kiracılı, gerçek zamanlı veri deposu (Manhattan).*<https://blog.x.com/engineering/en_us/a/2014/manhattan-our-real-time-multi-tenant-distributed-database-for-twitter-scale>. ([blog.x.com](https://blog.x.com/engineering/en_us/a/2014/manhattan-our-real-time-multi-tenant-distributed-database-for-twitter-scale?utm_source=chatgpt.com))
* Open Sourcing Heron — *Heron: Twitter’ın yüksek hacimli gerçek zamanlı stream işleme moturu (Storm’un yerini alacak yapı).*<https://blog.x.com/engineering/en_us/topics/open-source/2016/open-sourcing-twitter-heron>. ([blog.x.com](https://blog.x.com/engineering/en_us/topics/open-source/2016/open-sourcing-twitter-heron?utm_source=chatgpt.com))
* Building DistributedLog — *yüksek performanslı replikalı log servisi (Twitter’ın notları).*<https://blog.x.com/engineering/en_us/topics/infrastructure/2015/building-distributedlog-twitter-s-high-performance-replicated-log-servic>. ([blog.x.com](https://blog.x.com/engineering/en_us/topics/infrastructure/2015/building-distributedlog-twitter-s-high-performance-replicated-log-servic?utm_source=chatgpt.com))

# **Uber (resmî blog / engineering)**

* Cadence (Uber’s workflow orchestration) — *yüksek hacimli workflow’ları, multi-tenant görevleri ölçeklendirme biçimi.*<https://www.uber.com/blog/open-source-orchestration-tool-cadence-overview/>. ([Uber](https://www.uber.com/blog/open-source-orchestration-tool-cadence-overview/?utm_source=chatgpt.com))
* M3: Uber’s large-scale metrics platform — *ölçekli metrik ingest/serving (Prometheus ölçeklendirme için M3).*<https://www.uber.com/blog/m3/>. ([Uber](https://www.uber.com/blog/m3/?utm_source=chatgpt.com))
* Optimizing M3: How Uber halved metrics ingestion latency — *M3 performans iyileştirmeleri örneği.*<https://www.uber.com/blog/optimizing-m3/>. ([Uber](https://www.uber.com/blog/optimizing-m3/?utm_source=chatgpt.com))

# **LinkedIn (resmî engineering)**

* Apache Samza: LinkedIn’s real-time stream processing framework — *Samza’nın LinkedIn’de nasıl kullanıldığı (stream processing at scale).*<https://engineering.linkedin.com/data-streams/apache-samza-linkedins-real-time-stream-processing-framework>. ([engineering.linkedin.com](https://engineering.linkedin.com/data-streams/apache-samza-linkedins-real-time-stream-processing-framework?utm_source=chatgpt.com))

# **Cloudflare (resmî blog)**

* Cloudflare mitigates record-breaking 71 million request-per-second DDoS — *edge/anycast ve otomatik mitigasyon örneği (çok yüksek RPS hikâyesi).*<https://blog.cloudflare.com/cloudflare-mitigates-record-breaking-71-million-request-per-second-ddos-attack/>. ([The Cloudflare Blog](https://blog.cloudflare.com/cloudflare-mitigates-record-breaking-71-million-request-per-second-ddos-attack/?utm_source=chatgpt.com))
* A Brief Primer on Anycast — *global anycast ile isteklerin uçta dağıtılması.*<https://blog.cloudflare.com/a-brief-anycast-primer/>. ([The Cloudflare Blog](https://blog.cloudflare.com/a-brief-anycast-primer/?utm_source=chatgpt.com))
* How we make sense of 700M events/sec — *Cloudflare’ın veri pipeline/telemetry ölçeklendirmesi (events/sec örnekleri).*<https://blog.cloudflare.com/how-we-make-sense-of-too-much-data/>. ([The Cloudflare Blog](https://blog.cloudflare.com/how-we-make-sense-of-too-much-data/?utm_source=chatgpt.com))

# **Slack (resmî engineering)**

* Slack’s Migration to a Cellular Architecture — *blast-radius azaltma, hücresel/topology ile yüksek kullanılabilirlik.*<https://slack.engineering/slacks-migration-to-a-cellular-architecture/>. ([Engineering at Slack](https://slack.engineering/slacks-migration-to-a-cellular-architecture/?utm_source=chatgpt.com))

# **Pinterest (resmî Pinterest Engineering / Medium)**

* Scaling Cache Infrastructure at Pinterest — *mcrouter + memcached ölçeklendirme uygulamaları (Pinterest mühendisliği).*<https://medium.com/pinterest-engineering/scaling-cache-infrastructure-at-pinterest-422d6d294ece>. ([Medium](https://medium.com/pinterest-engineering/scaling-cache-infrastructure-at-pinterest-422d6d294ece?utm_source=chatgpt.com))
* Improving distributed caching performance and efficiency at Pinterest — *cache optimizasyonları.*<https://medium.com/pinterest-engineering/improving-distributed-caching-performance-and-efficiency-at-pinterest-92484b5fe39b>. ([Medium](https://medium.com/pinterest-engineering/improving-distributed-caching-performance-and-efficiency-at-pinterest-92484b5fe39b?utm_source=chatgpt.com))

# **Amazon (Builders’ Library — resmî, mimari pratikler)**

* Timeouts, retries and backoff with jitter — *istemci/servis dayanıklılık desenleri (retry + jitter vb.).*<https://aws.amazon.com/builders-library/timeouts-retries-and-backoff-with-jitter/>. ([Amazon Web Services, Inc.](https://aws.amazon.com/builders-library/timeouts-retries-and-backoff-with-jitter/?utm_source=chatgpt.com))
* Using load shedding to avoid overload — *yük atma (load-shedding) stratejileri.*<https://aws.amazon.com/builders-library/using-load-shedding-to-avoid-overload-in-distributed-systems-by-putting-the-smaller-service-in-control/>. ([Amazon Web Services, Inc.](https://aws.amazon.com/builders-library/using-load-shedding-to-avoid-overload/?utm_source=chatgpt.com))

# **Google (Site Reliability Engineering — resmi SRE Book)**

* Load Balancing at the Frontend (SRE Book) — *datacenterlar arası yük dengeleme ve ölçek.*<https://sre.google/sre-book/load-balancing-frontend/>. ([sre.google](https://sre.google/sre-book/load-balancing-frontend/?utm_source=chatgpt.com))
* Handling Overload (SRE Book) — *overload yönetimi, servis reddi/serving degraded responses.*<https://sre.google/sre-book/handling-overload/>. ([sre.google](https://sre.google/sre-book/handling-overload/?utm_source=chatgpt.com))