Sarcouncil Journal of Engineering and Computer Sciences

ISSN(Online): 2945-3585

Volume- 05| Issue- 10| 2026

Research Article **Received:** 01-09-2026 | **Accepted:** 25-09-2026 | **Published:** 10-10-2026

Hitless Firmware and Software Upgrades in Cloud and AI Infrastructure: Architectural Taxonomy, State Preservation, and High-Availability Engineering A Comprehensive Survey of In-Service Maintenance, Hardware State Retainers, and Production Resilience Across Hyperscale Datacenters

Ishaan Gupta

Google Inc

Abstract: Hyperscale cloud and distributed artificial intelligence (AI) infrastructures enforce five-nines (99.999%) availability, permitting ≤ 5.26 minutes of annual node downtime. In gang-scheduled AI clusters scaling to 32,768 accelerators, a single node reboot stalls synchronous AllReduce collective barriers, desynchronizes parameters, aborts training, and strands ~30 MWh of energy. Consequently, engineers cannot afford downtime for security patching, microcode remediation, or driver upgrades. This paper presents an architectural survey of hitless firmware and software upgrades across cloud and AI datacenters. We formalize a five-tier taxonomy spanning silicon microcode, peripheral controllers, accelerators, operating systems, and userspace data planes. We analyze state preservation via PCIe SerDes Link L0 keep-alive, Completion Timeout Range C extensions, dynamic ftrace livepatching, and SCM_RIGHTS socket migration. We evaluate critical edge cases, including voltage regulator load-line transients (dI/dt) and IEEE 1588 PTP holdover. Finally, we examine Hardware-in-the-Loop chaos validation, canary pipelines, and empirical deployments across Google Jupiter OCS, Titanium, Andromeda 2.2, and Cloud SQL.

Keywords: Hitless Firmware, Software, AI, Architectural Taxonomy.

INTRODUCTION

Hyperscale cloud and frontier AI platforms operate as continuous utilities [Barroso, L. A. *et al*., 2019]. Cloud Service Level Agreements guarantee five-nines (99.999%) availability, permitting under 5.3 minutes of annual node downtime. In distributed AI training, availability constraints are even stricter: Large Language Models require gang-scheduling across clusters of up to 32,768 GPUs or TPUs linked by high-bandwidth fabrics [Narayanan, D. *et al*., 2021; Jouppi, N. *et al*., 2023].

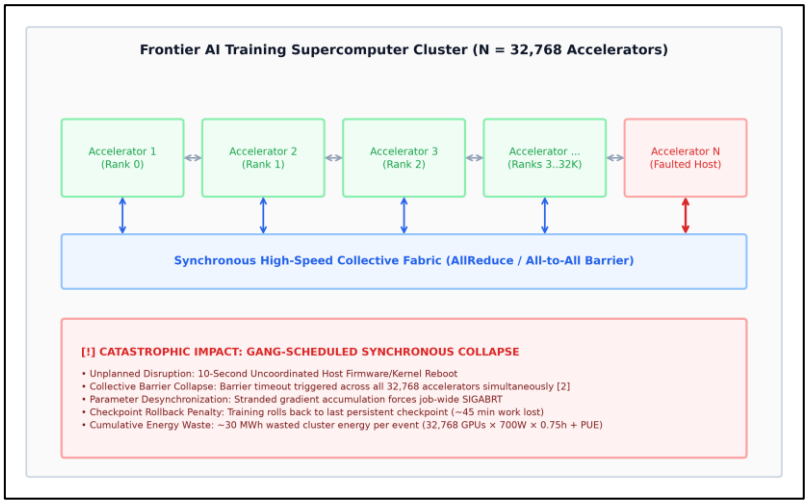


Figure 1: Catastrophic cascading impact of a single-node disruption on a gang-scheduled AI training collective.

As illustrated in Figure 1, these workloads rely on synchronous collective communication (Ring-AllReduce and All-to-All) where all 32,768 ranks must reach identical barriers before parameter updates proceed [Narayanan, D. *et al*., 2021]. If a single node resets for a 10-second reboot, the collective barrier collapses [Narayanan, D. *et al*., 2021]. Stranded workers trigger timeouts, gradients desynchronize, and the job aborts, forcing a rollback to the last checkpoint, discarding ~45 minutes of computation and stranding ~30 MWh of energy [Narayanan, D. *et al*., 2021].

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

***Corresponding Author:** Ishaan Gupta

DOI- https://doi.org/10.5281/zenodo.24005928

Because scheduled maintenance windows are obsolete in multi-tenant clouds and catastrophic to distributed AI jobs, **hitless upgrades**—patching, rebooting, and transitioning firmware and software in-situ without dropping sessions, interrupting I/O, or tearing down virtual contexts—are an essential architectural requirement.

THE MAINTENANCE DILEMMA FOR INFRASTRUCTURE ENGINEERS

Infrastructure engineers face conflicting imperatives: maintaining five-nines uptime while deploying security patches, microcode errata, and performance updates [Barroso, L. A. *et al*., 2019; Intel, I. (64). 2023]. Vulnerabilities in firmware, processor microcode (Spectre, Downfall), and hypervisor kernels demand rapid patching [Intel, I. (64). 2023; Arnold, J., & Kaashoek, M. F. 2009]. Simultaneously, modern server nodes have evolved into complex heterogeneous systems [Barroso, L. A. *et al*., 2019; Firestone, D. *et al*., 2018; OpenBMC Project. 2022].

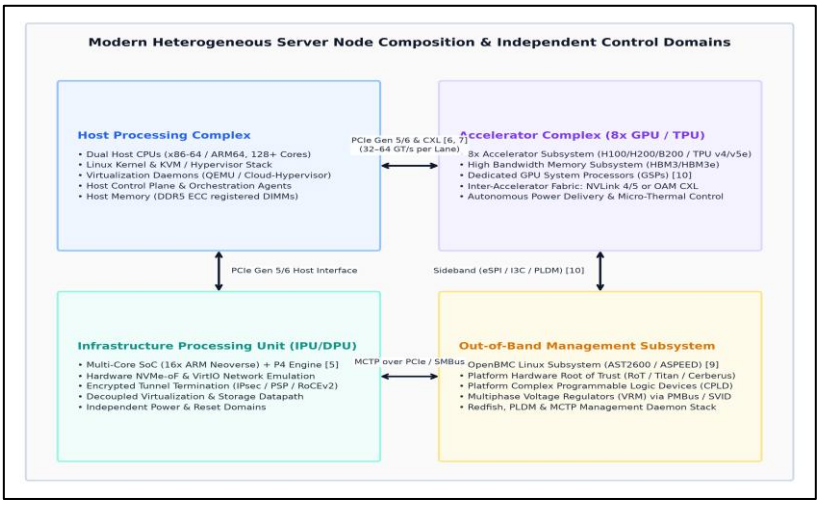


Figure 2: Architectural schematic of a modern heterogeneous server node highlighting independent control domains.

As structured in Figure 2, a modern node integrates multiple autonomous execution domains:

- **Host Processing Complex:** Dual x86-64/ARM64 CPUs running the Linux kernel, KVM, and virtualization daemons [Zhang, X. *et al*., 2019; Tatashin, P. *et al*., 2023].
- **Accelerator Subsystem:** 8-way GPU/TPU complexes with HBM3e, GPU System Processors (GSPs) [NVIDIA Corp. 2022], and scale-up fabrics (NVLink, ICI) [Jouppi, N. *et al*., 2023].
- **Infrastructure Processing Unit (IPU/DPU):** Multi-core SoCs executing P4 data planes, NVMe-oF, and encryption [Firestone, D. *et al*., 2018].
- **Out-of-Band (OOB) Subsystem:** Baseboard Management Controllers (OpenBMC) [OpenBMC Project. 2022], CPLDs, and Voltage Regulator Modules (VRMs).

These subsystems communicate over PCIe Gen 5/6, CXL [PCI-SIG. 2022; CXL Consortium. 2023], and sideband buses (eSPI, I3C) [NVIDIA Corp. 2022]. Cold reboots trigger cascading link drops, Machine Check Exceptions (MCEs), and VM crashes [Zhang, X. *et al*., 2019; PCI-SIG. 2022], necessitating autonomous in-service mutation.

ARCHITECTURAL TAXONOMY OF HITLESS UPGRADES

To systematically achieve zero-downtime maintenance, the infrastructure stack is structured into a five-tier hierarchy [Zhang, X. *et al*., 2019; PCI-SIG. 2022; Hicks, M., & Nettles, S. 200Firestone, D. *et al*., 2018], summarized in Table 1.

Table 1: Multi-tier architectural taxonomy of hitless upgrade mechanisms across infrastructure layers.

Layer 0	Silicon & Microcode	CPU Microcode, Patch RAM	Dynamic MSR Patching [Intel, I. (64). 2023]	10 µs - 2 ms	Register retention, pipeline drain [Intel, I. (64). 2023]

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

Layer 1	Peripheral Controllers	BMC, CPLD, Retimers, VRMs	Dual-Bank Flash Swap, Warm kexec [OpenBMC Project. 2022]	0 ms (Host Transparent)	Shadow register latching, isolated execution [OpenBMC Project. 2022]
Layer 2	High-Speed Accelerators	GPU/TPU Firmware, SmartNICs	Subsystem Reset (Link L0), FLR [PCI-SIG. 2022; NVIDIA Corp. 2022;	15 - 50 ms (Warm); 0.2 - 1.5 s (Deep)	SerDes L0 keep-alive, CTO Range C, DMA quiescing [PCI-SIG. 2022]
Layer 3	OS & Hypervisor	Host Kernel, KVM, Drivers	Dynamic ftrace Livepatch [Arnold, J., & Kaashoek, M. F. 2009; 13], Fast VMM Update [Zhang, X. *et al*., 2019; 14]	< 100 µs (Patch); 150 ms - 1.5 s (VMM)	Ftrace trampoline [Poimboeuf , J. 2016], reserved HugeTLB handoff [Zhang, X. *et al*., 2019; Tatashin, P. *et al*., 2023]
Layer 4	Userspace Data Plane	SDN Switches, Edge Proxies	Socket Handoff (SCM_RIGHTS), POSIX shm	< 5 ms	Dual-process handoff, shared memory session hydration

- **Layer 0 (Silicon Microcode):** MSR patching (e.g., IA32_BIOS_UPDT_TRIG) uses IPI rendezvous to synchronize cores, serialize pipelines, and update microcode in 10 µs - 2 ms without reboots [Intel, I. (64). 2023].
- **Layer 1 (Peripheral Management):** Firmware stages into alternate flash out-of-band; the BMC warm-boots via kexec while shadow latches maintain fan PWM and VRM power limits [OpenBMC Project. 2022].
- **Layer 2 (Accelerators & SmartNICs):** Updates coordinate host queue quiescing, SerDes L0 retention, and internal resets within millisecond windows [PCI-SIG. 2022; NVIDIA Corp. 2022].
- **Layer 3 (Host OS & Hypervisor):** Systems apply ftrace livepatching [Arnold, J., & Kaashoek, M. F. 2009; Poimboeuf , J. 2016] or fast kexec with Kernel Handover (KHO), preserving guest RAM across a 150 ms - 1.5 s reboot [Zhang, X. *et al*., 2019; Tatashin, P. *et al*., 2023].
- **Layer 4 (Userspace Data Plane):** Data-plane daemons transfer active socket descriptors and connection tables via UNIX domain sockets, avoiding TCP drops [IEEE. 2019].

HITLESS FIRMWARE UPGRADE ORCHESTRATION

Hitless firmware updates on PCIe/CXL endpoints follow a five-phase deterministic lifecycle, depicted in Figures 3 and 4.

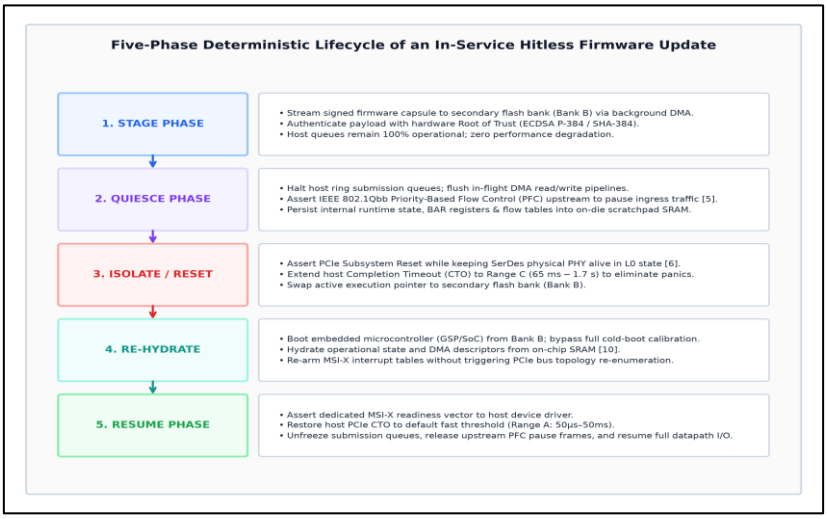


Figure 3: Five-phase lifecycle of an in-service hitless firmware update.

1. **Background Staging:** The driver streams the update capsule into inactive flash (Bank B) via background
2. DMA and verifies Root of Trust signatures (ECDSA/SHA-384) [Open Compute Project, 2022] without contending with active I/O.

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

3. **In-Band Quiescing:** The driver suspends ring submissions [Firestone, D. *et al.*, 2018], flushes in-flight DMAs, and asserts Ethernet PAUSE or PFC upstream [IEEE. 2019] to buffer transit packets.
4. **Subsystem Reset:** Unlike FLR (which resets only function state [PCI-SIG. 2022]), a vendor **Subsystem Reset** [6; NVIDIA Corp. 2022; reboots on-die cores while auxiliary power (V_aux) keeps SerDes PHYs in L0 transmitting Logical Idle flits [PCI-SIG. 2022]. The driver extends Completion Timeout (CTO) via DEVCTL2 to Range C (65 ms - 1.7 s) [PCI-SIG. 2022] to prevent MCE panics during microcontroller reboots.
5. **State Re-Hydration:** Firmware boots from Bank B, restoring BARs, MSI-X tables, and DMA pointers from on-chip SRAM without bus re-enumeration [PCI-SIG. 2022; NVIDIA Corp. 2022].
6. **Resume and Verification:** The endpoint asserts an MSI-X interrupt. The driver restores standard CTO limits, unfreezes queues, releases upstream flow control, and resumes full-speed I/O.

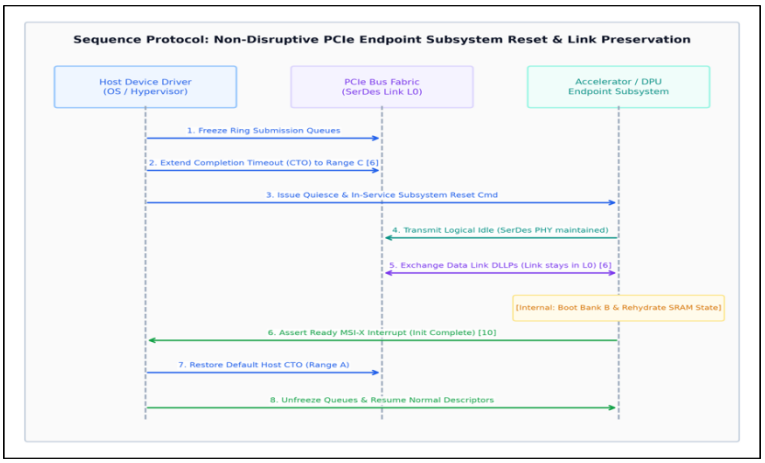


Figure 4: Sequence diagram of non-disruptive PCIe endpoint reset and link-preservation mechanics.

HITLESS SOFTWARE UPGRADE ORCHESTRATION

At operating system and userspace tiers, hitless updates ensure zero disruption to virtual machines, containers, and data planes [Zhang, X. *et al.*, 2019; Hicks, M., & Nettles, S. 2005; Arnold, J., & Kaashoek, M. F. 2009].

Operating System Live Patching (livepatch)

Dynamic kernel live patching modifies running kernel code without rebooting [Hicks, M., & Nettles, S. 2005; Arnold, J., & Kaashoek, M. F. 2009; Poimboeuf , J. 2016] using CONFIG_LIVEPATCH built on ftrace [Poimboeuf , J. 2016].

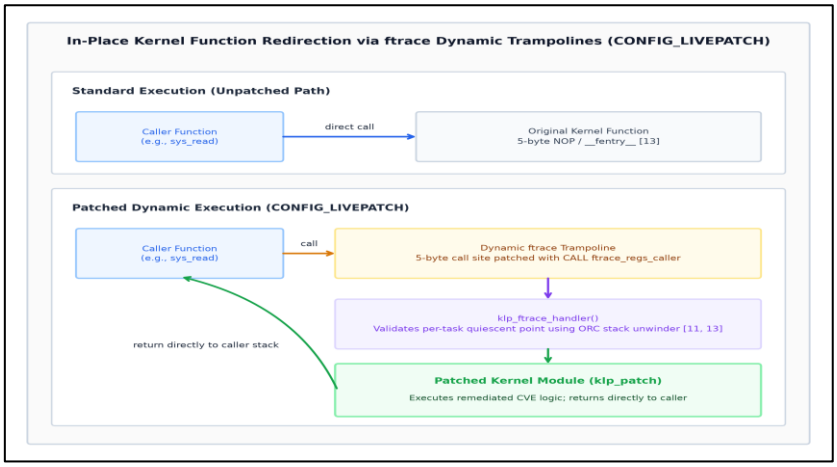


Figure 5: In-place kernel function redirection using ftrace dynamic trampolines.

As illustrated in Figure 5, compilers reserve a 5-byte __fentry__ call site at each function prologue [Poimboeuf , J. 2016]. Applying a patch redirects execution via an ftrace trampoline (klp_ftrace_handler()) to the patched function [Poimboeuf , J. 2016]. The ORC stack unwinder

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

verifies per-task consistency under Dynamic Software Updating (DSU) rules [Hicks, M., & Nettles, S. 2005; Arnold, J., & Kaashoek, M. F. 2009]: tasks switch to the new function only when quiescent outside the target stack frame, preventing lock inversions and state corruption.

Fast Kernel and Hypervisor Handoff via kexec and KHO

When updates modify core ABIs, live patching is insufficient [Zhang, X. *et al*., 2019; Tatashin, P. *et al*., 2023]. Hyperscalers deploy fast kexec handoffs coupled with Kernel Handover (KHO) [Zhang, X. *et al*., 2019; Tatashin, P. *et al*., 2023]:

- **Memory Decoupling:** Guest memory resides in persistent HugeTLB pools or memfd_create files, isolating guest RAM from host kernel state [Zhang, X. *et al*., 2019; Tatashin, P. *et al*., 2023].
- **State Serialization:** Hypervisors serialize vCPU states, interrupt configurations, and IOMMU tables (IOMMUFD) into preserved memory reservations (memblock=) [Zhang, X. *et al*., 2019; Tatashin, P. *et al*., 2023].
- **Fast Boot:** The host boots the new kernel via kexec_file_load() in 150 ms - 1.5 s, bypassing UEFI [Zhang, X. *et al*., 2019; Tatashin, P. *et al*., 2023]. The new hypervisor rebinds preserved HugeTLB pages, keeping VM blackout well within TCP retry thresholds.

Userspace Data-Plane Upgrades: Zero-Drop Socket Migration

Software data planes (Envoy, Open vSwitch) upgrade without dropping client connections [Firestone, D. *et al*., 2018].

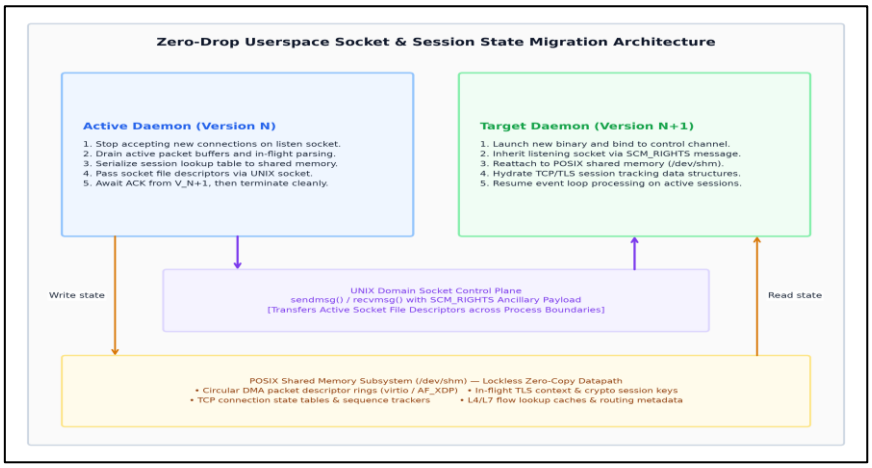


Figure 6: Userspace socket and session migration using SCM_RIGHTS and POSIX shared memory.

As diagrammed in Figure 6, Version N stops accepting new connections while Version N+1 initializes. Version N transfers listening and established socket file descriptors across a UNIX domain socket via sendmsg() with SCM_RIGHTS ancillary messages. Circular DMA rings, connection tables, and TLS session keys pass via POSIX shared memory (/dev/shm), allowing Version N+1 to resume packet processing in < 5 ms.

DESIGN NUANCES, LATENCIES, AND HARDWARE EDGE CASES

Deploying hitless updates requires resolving microarchitectural, electrical, and clock synchronization failure modes [Barroso, L. A. *et al*., 2019; PCI-SIG. 2022; IEEE. 2019], detailed in Table 2.

Table 2: Critical failure modes, latency thresholds, and mitigation mechanisms during hitless transitions.

PCIe Completion Timeout (CTO)	MMIO read issued during endpoint reset [PCI-SIG. 2022]	10 µs - 50 ms	Kernel MCE, Host Crash	Extend DEVCTL2 CTO to Range C (65 ms - 1.7 s) [PCI-SIG. 2022]; hold MMIO reads
AER Bad TLP Generation	Residual bytes in SerDes FIFO [PCI-SIG. 2022]	< 1 µs	Link drops to Recovery; unregisters	Hardware FIFO flush synchronization before reset [PCI-SIG. 2022]
VRM Voltage Droop Shutdown	High dI/dt transient unfreezing queues	1 - 10 µs	Undervoltage lockout (UVLO)	Staged queue release; progressive DVFS clock

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

			trips	ramp
PTP Clock Drift	Network PHY timestamping freezes [IEEE. 2019]	5 - 20 ms freeze; drift >100 ns after >100 ms	PLL servo filters starve; transactions stall	Hardware holdover synthesizers maintain ticks; OCXO/TCXO bounds drift [IEEE. 2019]
State Schema Incompatibility	Memory layout differs across revisions [Hicks, M., & Nettles, S. 200Firestone, D. *et al*., 2018]	N/A	Memory corruption upon rollback	Versioned serialization (FlatBuffers) with 2-phase commit

State Drift and Atomic Rollback

To prevent "reverse state drift"—where a failed Version N+1 partially corrupts state before rollback [Hicks, M., & Nettles, S. 200Firestone, D. *et al*., 2018]—handoffs use versioned FlatBuffers schemas governed by a two-phase commit: Version N serializes state into shared memory with a digest; Version N+1 validates integrity via self-tests; and only upon passing does Version N+1 advance an atomic pointer in battery-backed SRAM. Failures trigger immediate fallback to untouched Version N state.

Electrical Voltage Transients and Power Delivery Dynamics

High-end 700W–1000W accelerators operate below 0.85V, drawing 800A - 1200A [Jouppi, N. *et al*., 2023; NVIDIA Corp. 2022;. Quiescing drops current below 100A; unfreezing queues surges current back toward 1000A, producing slew rates (dI/dt) exceeding 100 A/µs. Transient voltage droop (ΔV) across parasitic impedances is governed by:

$$\Delta V \approx I_{step} \cdot R_{loadline} + L_{parasitic} \cdot dI/dt$$

If ΔV breaches ± 30 mV tolerances, Undervoltage Lockout (UVLO) trips, causing chassis shutdown. Firmware coordinates with VRMs to enforce staged queue release and progressive DVFS ramping to bound dI/dt.

Clock Synchronization (PTP/IEEE 1588) Drift and Local Oscillator Holdover

Distributed databases and AI fabrics rely on PTP (IEEE 1588-2019) [IEEE. 2019] for sub-microsecond timestamps. When network controllers reset, PHY timestamping pauses for 5 - 20 ms [IEEE. 2019]. While TCXO/OCXO oscillators bound physical drift to < 1 ns over 20 ms ($\varepsilon \leq 100$ ns budget), missing timestamps starve PLL servo filters. Hardware holdover circuits synthesize reference ticks into the PLL until controller reboots complete.

VALIDATION AND VERIFICATION IN HARDWARE TESTBEDS

Because firmware defects cause fatal kernel panics or cluster-wide AI training aborts [Narayanan, D. *et al*., 2021; Zhang, X. *et al*., 201], hitless upgrades cannot be qualified on production infrastructure. Reliability engineering mandates **Hardware-in-the-Loop (HIL)** regression testbeds [Open Compute Project, 2022].

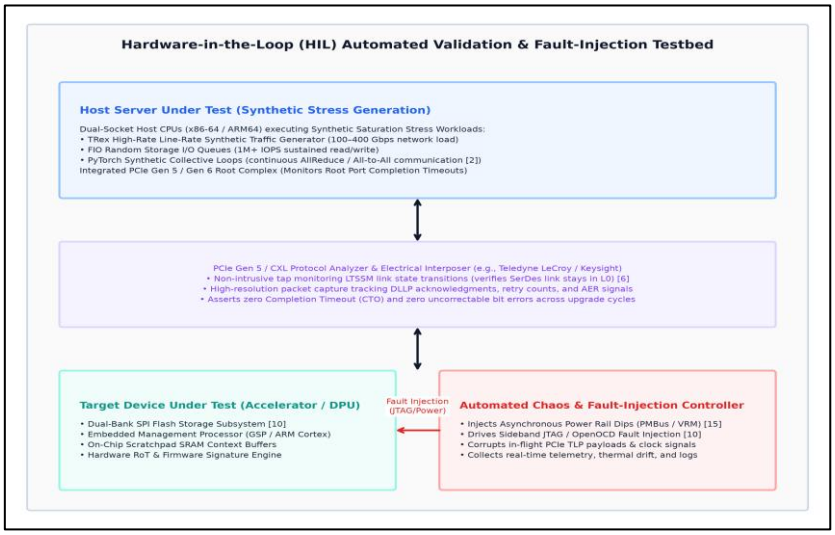


Figure 7: Structural layout of a Hardware-in-the-Loop (HIL) automated validation and fault-injection testbed.

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

Enterprise HIL facilities replicate production CPU steppings, PCIe switch revisions, and optical transceivers [Open Compute Project, 2022]. Hardware interposers capture bus packets via protocol analyzers to verify LTSSM remains in L0 [PCI-SIG. 2022], while oscilloscopes ensure voltage droop stays within ± 5%. Test harnesses inject synthetic stress (AVX-512 loops, TRex 100–400 Gbps traffic, FIO 4K storms, PyTorch AllReduce rings [Narayanan, D. *et al*., 2021]) alongside chaos faults: PCIe bit errors triggering recovery loops [PCI-SIG. 2022], frame drops [Firestone, D. *et al*., 2018], and PMBus dips [NVIDIA Corp. 2022; IEEE. 2019]. Candidate releases must pass 10,000 automated cycles with zero dropped links or panics before production deployment.

DEPLOYMENT, CANARY ROLLOUTS, AND PRODUCTION TELEMETRY

Following laboratory qualification, production rollouts proceed through structured **Canary Rings** [Barroso, L. A. *et al*., 2019; 20] to constrain failure blast radiuses.

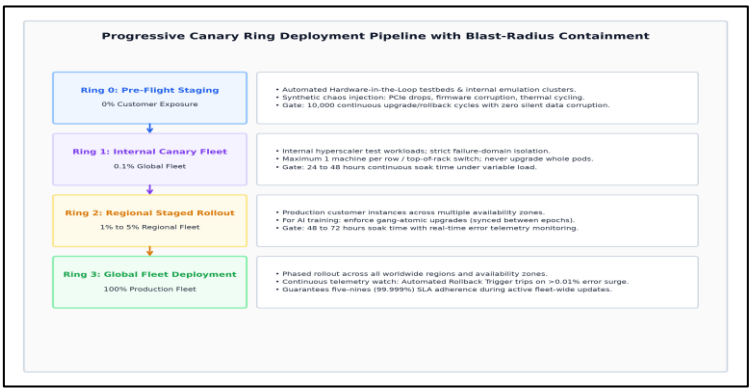


Figure 8: Progressive canary ring deployment pipeline ensuring blast-radius containment.

Deployment progresses through deterministic canary rings:

- **Ring 0 (Staging):** Internal testbeds and emulation clusters (0% customer exposure).
- **Ring 1 (Canary):** 0.1% of fleet, isolated to 1 host per rack/switch (24–48 hour soak).
- **Ring 2 (Regional):** 1% to 5% of fleet across availability zones (48–72 hour soak). For AI clusters, updates enforce **gang-atomic orchestration**: updates synchronize between training epochs across all ranks simultaneously to prevent barrier stalls [Narayanan, D. *et al*., 2021].
- **Ring 3 (Global):** Phased 100% rollout across worldwide data centers.

Continuous eBPF telemetry monitors kernel health, TCP retransmissions, PCIe Correctable Errors, and accelerator FLOPS. If metric deviations exceed 0.01% within 300 seconds, the orchestrator halts the rollout and triggers an atomic rollback.

HOLISTIC SYSTEMS DESIGN FOR DATA CENTER UPTIME

Hitless updates operate within a multi-tiered data center availability architecture [Barroso, L. A. *et al*., 2019; Zhang, X. *et al*., 2019; Clark, C. *et al*., 2005], illustrated in Figure 9.

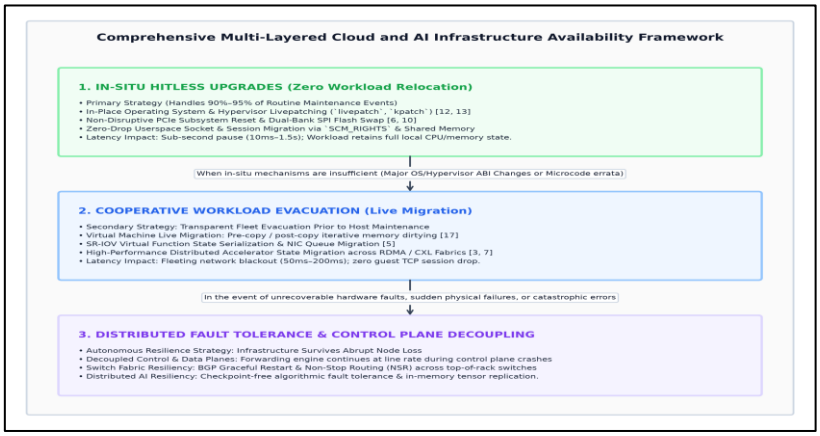


Figure 9: Layered availability framework ensuring five-nines uptime in cloud and AI infrastructure.

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

The Role of Live Migration

When maintenance requires physical board replacement or CPU upgrades, **Live Migration** evacuates workloads transparently [Clark, C. *et al*., 2005; Cully, B. *et al*., 2008]. VM live migration iteratively pre-copies memory, concluding with a brief 10 - 50 ms handoff blackout [Clark, C. *et al*., 200Firestone, D. *et al*., 2018]. In contrast, migrating 8-way GPU/TPU nodes requires transferring 1.1 TB to 1.5 TB of volatile HBM3e state across the network [Narayanan, D. *et al*., 2021; Jouppi, N. *et al*., 2023; NVIDIA Corp. 2022;, introducing multi-second blackouts that collapse collective barriers. Consequently, in-situ hitless updates remain orders of magnitude more efficient than live migration for AI infrastructure [Zhang, X. *et al*., 2019].

Decoupled Control and Data Planes

High-availability systems enforce strict separation between control and data planes [Barroso, L. A. *et al*., 2019; Firestone, D. *et al*., 2018]. Forwarding engines (P4 DPUs, switch ASICs) forward packets autonomously at line rate while routing daemons reboot during software updates via **BGP Graceful Restart** and **Non-Stop Routing (NSR)**.

Comparative Analysis across Major Cloud Implementations

Table 3 compares high-availability engineering trade-offs across hyperscale architectures.

Table 3: Comparative evaluation of cloud high-availability design choices.

Scope	Firmware, Kernel, Drivers, Daemons [Zhang, X. *et al*., 2019; 6, 13]	Virtual Machines, Host OS [Clark, C. *et al*., 200Firestone, D. *et al*., 2018]	Massive Gang-Scheduled AI Jobs [Narayanan, D. *et al*., 2021]
Workload Pause	None (< 100 µs patch; 15 - 50 ms warm reset; 0.2 - 1.5 s deep flash) [Zhang, X. *et al*., 2019; 6]	Minimal (10 - 50 ms blackout) [Clark, C. *et al*., 2005; Firestone, D. *et al*., 2018]	Severe (Job terminates, rolls back) [Narayanan, D. *et al*., 2021]
Network Overhead	Zero network bandwidth required	Extreme (Streams full guest RAM over LAN) [Clark, C. *et al*., 2005; Firestone, D. *et al*., 2018]	High (Reads hundreds of TBs from storage) [Narayanan, D. *et al*., 2021]
Constraints	Dual-bank flash, PCIe Link L0 keep-alive [PCI-SIG. 2022]	Compatible CPU features on target [Clark, C. *et al*., 2005; Firestone, D. *et al*., 2018]	Homogeneous accelerator cluster [Narayanan, D. *et al*., 2021; Jouppi, N. *et al*., 2023]
Complexity	High firmware/driver engineering [PCI-SIG. 2022; NVIDIA Corp. 2022;	Moderate orchestration complexity [Clark, C. *et al*., 2005; Firestone, D. *et al*., 2018]	Low engineering; massive compute waste [Narayanan, D. *et al*., 2021]
Primary Use Case	Routine CVE patching, driver tuning [Intel, I. (64). 2023; Arnold, J., & Kaashoek, M. F. 2009]	Chassis hardware repairs, decommissioning [Barroso, L. A. *et al*., 2019]	Unrecoverable hardware failures [Narayanan, D. *et al*., 2021]

Empirical Production Case Studies in Hyperscale Infrastructure

To validate these architectural paradigms under production conditions, Table 4 synthesizes empirical deployments across Google Cloud's hyperscale infrastructure [Poutievski, L. *et al*., 2022; Google Cloud. 2024; Dal Degan, D. *et al*., 2020; Google Cloud, 2023], detailing upgrade primitives, disruption bounds, and scale metrics across optical fabrics, offload engines, SDN data planes, and enterprise databases.

Table 4: Empirical production deployments of hitless and in-service upgrades in hyperscale cloud infrastructure**.**

Google Jupiter Fabric (OCS) [Poutievski, L. *et al*., 2022]	Layer 1–2	Apollo OCS MEMS switching; dynamic SDN rerouting	Zero packet drop; 10% capacity slices upgraded	6 Pb/s fabric; **50x less downtime** vs. Clos; 40% less power [Poutievski, L. *et al*., 2022]
Google Titanium	Layer	Hardware offload	Near-zero	192 vCPUs, 1.5 TB DDR5, 200

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

& C4 VMs [Google Cloud. 2024]	2–3	engine; in-situ host OS patching & live migration	customer impact (< 30 ms live migration handoff)	Gbps; host updates without VM reboots [Google Cloud. 2024]
Google Andromeda 2.2 [Dal Degan, D. *et al*., 2020]	Layer 4	Shared memory queue handoff (shm); socket migration	< 5 ms queue delay; zero TCP drops	32 Gbps to 100 Gbps line-rate forwarding across 8-GPU clusters [Dal Degan, D. *et al*., 2020]
Google Cloud SQL In-Place [Google Cloud, 2023]	Layer 4	Zero-copy storage hard-linking (pg_upgrade --link); IP/volume preservation	Downtime cut from hours/days to minutes (engine restart)	Zero network data copying across multi-TB PostgreSQL/MySQL instances [Google Cloud, 2023]

CONCLUSION

In hyperscale cloud platforms and distributed AI clusters, five-nines availability constraints and gang-scheduled AllReduce synchronization render maintenance windows obsolete. Eliminating disruption requires a vertically integrated architecture:

1. *Silicon & Physical Layer:* Microcode serialization, dual-bank flash controllers, and PCIe SerDes L0 keep-alive.
2. *Host Kernel & OS:* CONFIG_LIVEPATCH ftrace trampolines for CVE fixes, alongside KHO fast kexec handoffs for hypervisor updates.
3. *Userspace Data Plane:* Zero-drop socket migration via SCM_RIGHTS and shared memory session hydration.
4. *Rigorous Verification:* HIL chaos testbeds enforcing 10,000 failure-free upgrade cycles, backed by canary rings and eBPF automated rollbacks.

As fabrics embrace Compute Express Link (CXL 3.0) memory pooling [CXL Consortium. 2023] and Co-Packaged Optics [Jouppi, N. *et al*., 2023], hitless orchestration will expand across coherent memory domains, cementing in-service updates as the foundation of resilient infrastructure.

REFERENCES

1. Barroso, L. A., Hölzle, U., Ranganathan, P., & Martonosi, M. “The datacenter as a computer: Designing warehouse-scale machines.” *San Rafael, CA, USA: Morgan & Claypool*, (2019).
2. Narayanan, D. *et al*. “Memory-efficient pipeline parallelism for LLM training.” *SC '21*, (2021).
3. Jouppi, N., Kurian, G., Li, S., Ma, P., Nagarajan, R., Nai, L., & Patterson, D. A. "Tpu v4: An optically reconfigurable supercomputer for machine learning with hardware support for embeddings." *Proceedings of the 50th annual international symposium on computer architecture*. (2023).
4. Zhang, X., Zheng, X., Wang, Z., Li, Q., Fu, J., Zhang, Y., & Shen, Y. "Fast and scalable VMM live upgrade in large cloud infrastructure." *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. (2019).
5. Firestone, D., Putnam, A., Mundkur, S., Chiou, D., Dabagh, A., Andrewartha, M., & Greenberg, A. "Azure accelerated networking:{SmartNICs} in the public cloud." *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. (2018).
6. PCI-SIG. “PCI Express Base Specification Rev.” *6.0*, (2022).
7. CXL Consortium. “Compute Express Link Specification Rev.” *3.1*, (2023).
8. Intel, I. (64) "and IA-32 architectures software developer’s manual." *Volume 3A: System Programming Guide, Part* 1.64 (64): 64. (2023).
9. OpenBMC Project. “OpenBMC Architecture and Firmware Update Specification,” (2022).
10. NVIDIA Corp. “NVIDIA Open GPU Kernel Modules: GSP Architecture,” (2022).
11. Hicks, M., & Nettles, S. "Dynamic software updating." *ACM Transactions on Programming Languages and Systems (TOPLAS)* 27.6 (2005): 1049-1096.
12. Arnold, J., & Kaashoek, M. F. "Ksplice: Automatic rebootless kernel updates." *Proceedings of the 4th ACM European conference on Computer systems*. (2009).
13. Poimboeuf , J. “Live kernel patching in the Linux kernel.” *Linux Plumbers Conf.*, (2016).
14. Tatashin, P. *et al*. “Kernel Handover (KHO) and Live Update Orchestrator.” *Linux Plumbers Conf.*, 2023.

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

15. IEEE. *Standard for Precision Clock Synchronization (IEEE 1588-2019)*, (2019).
16. Open Compute Project. "OCP Baseline Security and Firmware Specifications." (2022).
17. Clark, C., Fraser, K., Hand, S., Hansen, J. G., Jul, E., Limpach, C., & Warfield, A. "Live migration of virtual machines." *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*. (2005).
18. Cully, B., Lefebvre, G., Meyer, D., Feeley, M., Hutchinson, N., & Warfield, A. "Remus: High availability via asynchronous virtual machine replication." *Proceedings of the 5th USENIX symposium on networked systems design and implementation*. (2008).
19. Chen, H., Chen, R., Zhang, F., Zang, B., & Yew, P. C. "Live updating operating systems using virtualization." *Proceedings of the 2nd international conference on Virtual execution environments*. (2006).
20. Verma, A., Pedrosa, L., Korupolu, M., Oppenheimer, D., Tune, E., & Wilkes, J. "Large-scale cluster management at Google with Borg." *Proceedings of the tenth european conference on computer systems*. (2015).
21. Poutievski, L., Mashayekhi, O., Ong, J., Singh, A., Tariq, M., Wang, R., & Vahdat, A. "Jupiter evolving: transforming google's datacenter network via optical circuit switches and software-defined networking." *Proceedings of the ACM SIGCOMM 2022 Conference*. (2022).
22. Google Cloud. "Titanium architecture and C4 machine series," (2024).
23. D. Dal Degan *et al*. "How Andromeda 2.2 enables high-throughput *VMs*," *Google Cloud*, (2020).
24. Google Cloud. "Cloud SQL in-place major version upgrade architecture," (2023).

Source of support: Nil; **Conflict of interest:** Nil.

Cite this article as:
Gupta, I. "Hitless Firmware and Software Upgrades in Cloud and AI Infrastructure: Architectural Taxonomy, State Preservation, and High-Availability Engineering A Comprehensive Survey of In-Service Maintenance, Hardware State Retainers, and Production Resilience Across Hyperscale Datacenters" *Sarcouncil Journal of Engineering and Computer Sciences* 5.10 (2026): pp 24-33.

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher