GPU SMITH
COMPUTE ENGINEERING

GPU SMITH / RESEARCH & PRACTICAL GUIDANCE

NVIDIA PAIR Local Inference Router: Enterprise Pilot Gate

> Plan and build private AI compute with GPU Smith. We connect workload requirements with GPU hardware, networking, deployment and operating decisions so your infrastructure fits the work it must perform.

Published by GPU Smith · 2026-11-02 · nvidia pair · nvidia personal ai router · local inference router · private ai · ollama load balancing · lm studio · enterprise ai pilot · dgx spark · rtx pro · ai governance

Original article: https://gpusmith.com/articles/en/nvidia-pair-enterprise-pilot-gate-local-inference-router



In this article

1. Executive Summary
2. Introduction and Background
3. Key Changes: What the PAIR Release Introduces
4. Threat Model and Trust Boundaries
5. Implementation Considerations and Process Changes
6. Comparison: PAIR, a Single Inference Server and an Enterprise Gateway
7. Data Analysis and Evidence
8. Go/No-Go Gate, Rollback and Unpair Plan
9. Implications and Future Directions
10. Frequently Asked Questions (FAQs)
11. Conclusion

Executive Summary

NVIDIA Personal AI Router (**PAIR**) is a request router for local inference engines, not a pooled GPU cluster. NVIDIA published the launch post on **3 September 2026** (developer.nvidia.com), and the project's GitHub release page lists **version 1.0.0** as the latest build as of the 2 November 2026 retrieval date (github.com). Each request is sent to one system, and PAIR does not combine GPU memory, join GPUs or split a model (build.nvidia.com). Adding machines raises concurrency but "does not make an individual request faster" (docs.nvidia.com).

For enterprise endpoint and AI platform teams, the useful question is whether a heterogeneous set of owned workstations can serve parallel agent calls under acceptable controls. The vendor documentation lists the boundaries that decide this. The scheduler does not consider GPU model, available memory or measured latency (docs.nvidia.com). Node hostname, hardware inventory and utilization are readable by anything on the same subnet that asks (docs.nvidia.com). Pairing starts with a six-digit PIN that the project's own security policy calls low entropy (github.com), and mTLS (mutual Transport Layer Security) protects cluster channels only (github.com). Engines, model catalogs and update systems are separate third-party boundaries (github.com).

The recommendation of this report is a conditional go for a **controlled pilot** on an isolated VLAN (virtual local area network) with host firewalls, approved models and synthetic or low-classification prompts. Sensitive prompts should wait for evidence: a ports and flows capture, a failure-injection grid, and per-node latency and correctness records. Community support is best-effort and maintainers do not guarantee a response or resolution time (github.com), so the pilot owner carries the operational risk.

The economics should be measured, not assumed. NVIDIA's own demonstration reported **18 minutes** on one laptop and **8 minutes 48 seconds** across three devices (developer.nvidia.com), a ratio of about **2.05**, and NVIDIA states that it is a configuration-specific demo, not a general benchmark (developer.nvidia.com). At the U.S. average commercial electricity price of **14.47 cents** per kilowatt-hour for August 2026 (www.eia.gov),

under an assumed 200 W per node, the energy difference for such a run is under half a cent, so administration and risk, not power, decide whether the pilot pays. PAIR must also not be described as air-gapped: the playbook requires internet access for installer, engine and model downloads (build.nvidia.com).

Introduction and Background

PAIR sits in front of existing **inference engines**. At launch, press coverage reported Ollama and LM Studio as the supported backends (tbreak.com), and the 1.0.0 release added managed llama.cpp support (github.com). Applications keep their usual endpoints because PAIR takes over the default engine ports and moves the engines aside (docs.nvidia.com). The result is a "virtual inference router": a client talks to a local proxy and the proxy chooses which paired machine runs the request.

Public coverage of this product is largely consumer-oriented, for example a report describing it as a free beta that turns compatible Macs, RTX PCs and DGX Spark systems into a local AI cluster (tbreak.com). This report takes a different view and treats PAIR as a candidate component of a **private AI** estate, which is the lens of an enterprise reviewer asking what a local inference router can safely be allowed to do.

This report was prepared by GPU Smith, an independent engineering firm working on private AI infrastructure and enterprise GPU procurement. The publisher does not sell PAIR and has no cloud capacity of its own, but it may procure GPU hardware for clients through its channel, so readers should weigh the hardware discussion accordingly. No benchmark was run for this report. Every number below comes from a dated primary source or from arithmetic shown in the text, and any example built on assumed inputs is labeled as hypothetical. The product pages were read on 2 November 2026, and where the launch-era description differs from the current pages (notably the version number and engine list), the current pages are used.

Two facts frame everything that follows. First, discovery grants nothing, and "seeing a machine grants it nothing" (docs.nvidia.com), which moves the trust decision to pairing. Second, the project is licensed under Apache License 2.0 (github.com), but the engines and models used with it may carry separate terms (github.com), so the legal and supply-chain review cannot stop at PAIR itself.

Key Changes: What the PAIR Release Introduces

Request routing, not pooled memory

The central design statement is documented on NVIDIA's playbook: PAIR sends each request to one system and does not combine GPU memory or join GPUs (build.nvidia.com). A request therefore stays on the one node that received it. The documentation also states that PAIR does not pool GPU memory or make several GPUs act as one larger GPU (docs.nvidia.com).

Three consequences follow for capacity planning. A model that does not fit in one node's memory is not made runnable by pairing. A single long generation takes as long as it would on the chosen node. Throughput gains appear only when many independent requests arrive together, so highly sequential tasks and workloads dominated by one long call gain little. Pairing is also not a model distribution mechanism: nodes do not share models, so a node can serve a request only for a model it already holds (docs.nvidia.com), and PAIR itself does not store or serve models (docs.nvidia.com).

Supported nodes, engines and model placement

The launch post lists RTX PRO workstation GPUs from the Turing architecture and newer among supported hardware (developer.nvidia.com). The playbook lists Windows 11, Linux and macOS 13 or later (build.nvidia.com), and Windows on ARM is marked experimental (github.com). Enterprise reviewers should separate three classes of node in the eligibility ledger.

- **Discrete-GPU workstations.** The RTX PRO 6000 Blackwell Workstation Edition lists **96 GB** of GDDR7 memory with ECC (error-correcting code) (www.nvidia.com) and **1,792 GB/s** of memory bandwidth (www.nvidia.com), with a maximum power consumption specified at **600 W** (www.nvidia.com).
- **Unified-memory systems.** DGX Spark lists **128 GB** of unified system memory (docs.nvidia.com) and a 64 GB variant sold only through OEM partners (www.nvidia.com).
- **Consumer and laptop hardware.** Mixed fleets are acceptable in a lab, but an enterprise ledger should record each device's management and patch status before it is paired.

Unified memory is a reporting hazard. PAIR's Known Issues page says it counts only memory dedicated to the GPU, so unified-memory systems can show understated memory (docs.nvidia.com). The same page states that routing does not consider VRAM at all (docs.nvidia.com), so the display gap affects what an operator sees, not the routing decision, but it removes a signal an operator might otherwise use for sizing.

Model placement is manual. PAIR's model-eligibility gate matches Ollama names after normalizing the implicit latest tag, while LM Studio model identifiers must match exactly (docs.nvidia.com). If no advertised owner is routable, the proxy answers locally without sending the request to an engine (docs.nvidia.com). Ollama exposes a content digest per model in its listing API (docs.ollama.com), which is the practical field for an approved-model allowlist, because a matching name on two nodes does not prove identical weights.

Architecture, ports and data flow

Table 1 summarizes the listeners and the protection the vendor documents for each, using the default installation.

Port or channel	Purpose	Documented protection	Pilot control
11434, 1234, 8080	Application-facing proxies (Ollama-style, OpenAI-style, llama.cpp)	Plaintext HTTP, loopback only (docs.nvidia.com)	Confirm with a port scan from a second VLAN host that no plaintext request is answered
14318	Node hardware and telemetry	Hostname, hardware inventory and utilization readable by anything on the subnet (docs.nvidia.com)	Isolated VLAN, host firewall allow-list of paired peers only
14321	Pairing and cluster membership	Plaintext, authenticated only by the PIN (docs.nvidia.com)	Open pairing windows only during a supervised change, then close
14322, 14323	Model list for peers; cluster-scoped remote engine control	Cluster-gated: caller must present a certificate pinned for a current member (docs.nvidia.com)	Review the member list against the asset inventory

Port or channel	Purpose	Documented protection	Pilot control
mDNS (multicast DNS)	Discovery announcements	Announcements carry no built-in trust model; mDNS assumes cooperating participants (www.rfc-editor.org)	Treat as advertisement only; pairing is the trust decision
Engine ports	Ollama or LM Studio behind the proxy	Ollama binds to loopback by default (docs.ollama.com)	Verify that no engine is separately bound to a LAN address

The table groups the documented ports, and the default install uses further node-to-node ports for error synchronization and workload propagation. The table shows that the sensitive prompt path (loopback in, mTLS between nodes) is better protected than the metadata path, and that the weakest surfaces are the ones a rogue host on the subnet can reach: telemetry, pairing and discovery. The security policy states that network segmentation and host firewall rules remain the operator's responsibility (github.com).

The certificate model is trust on first pairing, with cluster-gated endpoints accepting only a caller whose certificate is pinned for a current member (Table 1). The pages reviewed document certificate deletion on reset or purge (github.com), but they do not document certificate expiry, rotation or revocation lists. That is an open question for the vendor, not evidence of absence.

Scheduler: load signals and omissions

The scheduler ranks nodes by total pending work across all engines (docs.nvidia.com). *Table 2* lists what it uses and what it ignores, together with the test that exposes each omission.

Signal	Used?	Documented behavior	Acceptance test
Pending job count	Yes	Every job counts as one unit, so a three-token completion and a long generation weigh the same (docs.nvidia.com)	Mix short and long jobs, record skew per node
GPU utilization	Coarsely	Busiest-GPU utilization maps to pressure units at 40%, 70% and 85% (docs.nvidia.com); multi-GPU nodes use the maximum across GPUs	Load one GPU of a multi-GPU node and observe routing
GPU model, memory, latency	No	The scheduler does not consider GPU model, available memory or measured latency	Pair a fast and a slow node, compare p95 per node
Model load state	No	Known to PAIR but not used, so a request can go to a cold node (docs.nvidia.com)	Warm one node, unload on another, send identical requests
Direct engine and CPU-only work	Partly	Direct work can raise pressure, but CPU-only work and queued demand stay invisible	Run a local job outside PAIR on one node
Telemetry age	Fallback	Missing or older-than-ten-second telemetry counts as a neutral pressure of 1 (docs.nvidia.com)	Block port 14318 on one node, observe placement

The pattern is capacity blindness. A node with a small GPU and a node with a large GPU look identical if each holds one pending job, so a faster node does not attract proportionally more work. Manual selection is a proxy capability that neither the desktop application nor the terminal interface uses (docs.nvidia.com), so the

"manual-node fallback" in a pilot plan means a per-client direct endpoint, not a PAIR feature. The project roadmap lists more scheduler variables and KV cache-aware scheduling as future items, with the explicit caveat that it is not a commitment to delivery (github.com).

Threat Model and Trust Boundaries

This section applies generic controls to documented behavior. It makes no claim that any defect exists, and the vendor's own security policy says its statements are not claims that every deployment is secure (github.com).

Trusted subnet and zero trust

The product assumes a trusted local network. NIST SP 800-207 states that all communication is secured regardless of network location (nvlpubs.nist.gov) and that trust should not be automatically granted based on the device (nvlpubs.nist.gov). Read against that standard, a flat office network is a weak trust boundary for a router whose node telemetry is readable on the subnet. The mitigations are network segmentation and host firewalls, consistent with SP 800-53 control SC-7, which requires external connections only through managed interfaces (doi.org), and CM-7, which requires prohibiting or restricting unneeded ports, protocols and services (doi.org). The CISA and NSA joint guidance on deploying AI systems recommends security protections at the boundary between the IT environment and the AI system (media.defense.gov).

Rogue discovery, the PIN and pairing

Multicast DNS (mDNS) .local names are meaningful only on the link where they originate (www.rfc-editor.org). RFC 6762 notes that conflict handling assumes cooperating participants (www.rfc-editor.org) and points to IPsec or DNSSEC signatures where hosts cannot be trusted (www.rfc-editor.org). DNS-based service discovery adds no specific security of its own (www.rfc-editor.org). PAIR's documentation likewise treats discovery as finding candidates only, as noted in the introduction.

Pairing is where trust begins. A six-digit PIN has 10^6 (one million) possible values. NIST SP 800-63B-4 specifies randomly generated out-of-band secrets of at least six decimal digits (pages.nist.gov) but only together with rate limiting of consecutive failures (pages.nist.gov) and a cap of no more than 100 failures (pages.nist.gov). At that cap, a blind guesser against a uniformly random PIN succeeds with probability 100 in 1,000,000, or **0.01%**. For a per-invitation PIN, the documentation says a PIN belongs to one invitation attempt (docs.nvidia.com) and that a mistyped PIN ends that attempt. If an attacker could not simply start new invitations, that would reduce blind guessing to roughly **one chance in a million** per invitation, but that figure is an assumption, not a documented guarantee: the same page calls retrying "cheap", and the pages reviewed do not state a retry limit or lockout, so that remains a verification item. The playbook describes the PIN as a short-lived setup code, not a long-term credential (build.nvidia.com). Mutual TLS itself is optional in the protocol: TLS 1.3 omits the certificate request when client authentication is not desired (www.rfc-editor.org), and NIST SP 800-52 describes client authentication as optional (nvlpubs.nist.gov), so enforcement on each cluster endpoint should be verified by test rather than assumed. The practical control is procedural: pair only during supervised change windows, on the isolated VLAN, between machines on the asset inventory.

Endpoint and local-user risk

The security policy states that the PIN does not protect a node before pairing and does not compensate for a compromised paired member (github.com). Certificate storage and operating-system account protection are part of the trust boundary (github.com). A paired node's trust is therefore only as strong as its operating-system account controls, and zero trust tenet 5 expects the enterprise to monitor the integrity and security posture of all owned assets (nvlpubs.nist.gov). Members that are no longer authorized should be removed, as a standing joiner-mover-leaver step.

NIST's AI RMF (Risk Management Framework) lists third-party software, hardware and data as a distinct measurement challenge (nvlpubs.nist.gov) and organizes governance into four functions, Govern, Map, Measure and Manage (nvlpubs.nist.gov), and its governance category expects mechanisms to inventory AI systems (nvlpubs.nist.gov). The framework says its actions are not a checklist (nvlpubs.nist.gov), so the control list in this report is a pilot instrument, not a compliance mapping.

Engines, model downloads and updates

The documentation states plainly that "local-first" is not the same as nothing leaving the network, and that engines, model catalogs and update systems may contact external services (github.com). The specifics matter for classification:

- **Model downloads.** Ollama pulls models from the internet (docs.ollama.com), and LM Studio makes network requests such as to huggingface.co (lmstudio.ai).
- **Engine installs and updates.** PAIR does not currently offer to update an engine because it does not yet detect newer versions (docs.nvidia.com). LM Studio's app updater makes a network request to check for updates (lmstudio.ai).
- **PAIR updates.** Released installers are signed (github.com), while builds made from source are unsigned and check no updates (github.com).
- **Offline modes.** LM Studio's core functions do not require the internet (lmstudio.ai), and llama.cpp's server has an offline mode that blocks network access (github.com).

OWASP's supply-chain entry recommends using models only from verifiable sources with integrity checks (genai.owasp.org) and notes that on-device LLMs increase the attack surface and supply-chain risk (genai.owasp.org). The CISA and NSA guidance advises evaluating the supply chain for external models (media.defense.gov), not running newly acquired models right away in the enterprise environment (media.defense.gov), and creating hashes of each model release (media.defense.gov). NIST's generative AI profile similarly flags non-transparent or untraceable integration of third-party components as a value-chain risk (nvlpubs.nist.gov), and SP 800-53 SR-3 calls for a process to identify and address supply chain weaknesses (doi.org). A pilot should therefore import approved models through a staging host, record digests, and block egress from the pilot VLAN except to an internal mirror.

Prompt and log retention

The vendor states that no PAIR component writes prompts or responses to logs, as a hard rule and not a setting (docs.nvidia.com). The statement is scoped to PAIR components, so the retention behavior of the engines, client applications and agent frameworks must be reviewed separately. OWASP lists confidential

business data and security credentials among the sensitive information a model deployment can disclose (genai.owasp.org) and recommends least-privilege access (genai.owasp.org). CISA and NSA treat model weights, outputs and logs as sensitive AI information to protect (media.defense.gov) and recommend logs covering inputs, outputs and errors with automated alerts (media.defense.gov). Because PAIR does not write prompts, any prompt-level audit record has to be created deliberately at the client or gateway layer. NIST SP 800-92 notes that long retention periods (months or years) need archival handling (nvlpubs.nist.gov). SP 800-53 AU-2 expects the organization to identify loggable event types (doi.org), and AU-11 expects retention to support after-the-fact incident analysis (doi.org). The pilot must decide, per data class, which layer owns the audit record.

The matrix below is a recommended data-classification gate, derived from the controls above.

- **Public or synthetic data.** Allowed in the pilot from day one on the isolated VLAN.
- **Internal, non-sensitive prompts.** Allowed after the ports capture and the firewall allow-list are accepted.
- **Confidential business data.** Allowed only after failure injection, model digest pinning and a decision on client-side audit logging.
- **Regulated or credential-bearing data.** Not routed through PAIR during the pilot, because certificate lifecycle and PIN retry limits are undocumented.

Implementation Considerations and Process Changes

Pilot topology

Use a dedicated VLAN containing only pilot nodes, an administrator jump host and a staging mirror for models. Block inbound traffic from other VLANs and block all egress except the mirror. Allow mDNS inside the VLAN and the documented peer ports only between node addresses. Leave the proxy ports on loopback, because applications are expected to run on a PAIR node. The playbook states that PAIR accepts requests only from the local system (build.nvidia.com). That is a security property, but it means shared-service patterns need a separate gateway process on a node, which then becomes part of the trust boundary.

Pin every node to the same PAIR release, since mixed versions are untested and can misbehave (docs.nvidia.com). A rollout is therefore a coordinated change.

Functional tests and failure injection

The test grid below should be run on the pilot VLAN and recorded with the serving node for each request, using an independent client-side record in addition to any record kept by PAIR.

- **Routing eligibility.** Request a model held by one node only. Expect no balancing, because nodes do not share models. Request an absent model and expect the local error.
- **Exact model identity.** Pull "the same" model on two nodes, compare digests (docs.ollama.com), and send a fixed prompt set at temperature zero to compare outputs for correctness drift.

- **Cold versus warm load.** PAIR does not warm models. Ollama keeps a model for **5 minutes** by default (docs.ollama.com), LM Studio JIT-loaded models default to a **60 minute** TTL (lmstudio.ai), and PAIR-managed llama.cpp models sleep after five idle minutes (docs.nvidia.com). Measure first-token latency after each idle period.
- **Failover.** Power off a node mid-run. The documentation says a node is dropped only after twelve consecutive misses, **about a minute** (docs.nvidia.com), so expect a window where requests may still be directed to a dead node, and measure it.
- **Stuck service.** A service that exits is restarted, but nothing detects one that is running and wedged (docs.nvidia.com). The documentation adds that a hang is not detected. Build an external canary request per node.
- **Stale telemetry.** Block the telemetry port and confirm the node is scored at neutral pressure.
- **Queue saturation.** Ollama defaults to one parallel request per model (docs.ollama.com) and queues up to **512 requests** (docs.ollama.com) before returning HTTP 503 (docs.ollama.com). Test beyond that depth.
- **Manual-node fallback.** Because PAIR's front ends do not use manual pinning, verify that clients can fall back to a direct, firewalled engine endpoint on a chosen node.

SLO test across homogeneous and heterogeneous nodes

Define the service level objective (SLO) before measuring. Google's SRE book warns that a simple average can obscure tail latencies (sre.google) and that tail effects are exacerbated at high load by queuing (sre.google), so the pilot should report percentiles per node. SLO tooling expects the same inputs of numerator, denominator and threshold (sre.google). OpenTelemetry's generative AI conventions define a time-to-first-token metric covering the queue and the prefill phase (raw.githubusercontent.com) and a time-per-output-token metric (raw.githubusercontent.com), which are suitable per-node attribution fields.

Run two arms: a homogeneous set (identical nodes) and a heterogeneous set (for example one RTX PRO workstation and one unified-memory system). Use the same prompt set, the same model digest and the same arrival pattern. For every request, record client send time, serving node, first-token latency, tokens per second, total time, HTTP status and an output-correctness flag. The comparison of interest is whether the heterogeneous arm's p95 shows slow-node skew, as the scheduler's omission of GPU model and latency predicts. Public benchmarks do not substitute: MLPerf, for instance, measures how fast systems process inputs with a trained model in defined scenarios (mlcommons.org) and notes that measured power is valid only for the accompanying benchmark (mlcommons.org).

Comparison: PAIR, a Single Inference Server and an Enterprise Gateway

Table 3 compares the three approaches on requirements. The gateway column describes capabilities of products documented by their authors, such as LiteLLM and the Kubernetes Gateway API Inference Extension, and is not a recommendation of any product.

Requirement	PAIR (routing across paired nodes)	Single inference server	Enterprise gateway or orchestrator
Capacity model	Job count and coarse GPU pressure	One engine queue; Ollama defaults to 1 parallel request per model (docs.ollama.com)	Can use queue depth and KV cache utilization reported by model servers (raw.githubusercontent.com)
Authentication	Plaintext ingress on loopback only; cluster endpoints gated by pinned certificates	Ollama binds loopback by default (docs.ollama.com); LM Studio requires no authentication by default (lmstudio.ai); llama.cpp key default is none (github.com)	Virtual keys for spend tracking and model-access control (docs.litellm.ai)
Identity and RBAC	No client identity model in the pages reviewed	Not documented in the pages reviewed; Ollama can be exposed through a proxy such as Nginx (docs.ollama.com)	Role-based access in the product; SSO beyond 5 users needs an enterprise license (docs.litellm.ai)
Health and failover	Eviction after about a minute; hangs undetected	No peer to fail over to (single point of failure)	Cooldowns, fallbacks, timeouts and retries (docs.litellm.ai); failing deployments removed from the pool (docs.litellm.ai)
Latency-aware routing	Not used	Not applicable	Latency-based strategy picks the lowest response time (docs.litellm.ai)
Governance and audit	PAIR logs exclude prompts; manual log collection	Engine-specific	Spend and access tracking are product features and need a database (docs.litellm.ai)
Deployment effort	Installer sets up background services and the desktop application on each node	One install	Kubernetes-native options exist (docs.vllm.ai) and the Inference Extension is an official Kubernetes project (gateway-api-inference-extension.sigs.k8s.io)

Read as requirements, the table separates use cases. A single server is the right baseline for a first control group, because it has the smallest attack surface and PAIR's gain must be measured against it. A gateway is the better fit when the requirement is identity-aware access, audit, quotas and latency-aware routing across a managed fleet, since the PAIR pages reviewed document none of those. PAIR's niche is narrow and real: opportunistic reuse of owned workstations for parallel, independent, tolerance-for-variance workloads, with an operator who accepts young, best-effort-supported tooling (launch coverage described a beta; version 1.0.0 is the current release). Some gateway scheduling strategies carry their own costs, for example LiteLLM states that usage-based routing is not recommended for production due to performance impacts (docs.litellm.ai), which shows that the gateway column also needs measurement, not assumption.

Data Analysis and Evidence

The numbers below are either quoted from primary sources or computed from them. Computations use stated inputs, and hypothetical inputs are labeled.

Vendor demonstration arithmetic. NVIDIA's reported five-subagent workload took 18 minutes on one laptop and 8 minutes 48 seconds on the three-device cluster, as cited in the executive summary. In seconds, that is **1,080 against 528**, a saving of **552 seconds (51.1%)** and a ratio of **2.05**. A press report on the same product

describes a different configuration, 6 minutes 18 seconds on one RTX 5090 (www.digitalcitizen.life) against 3 minutes 48 seconds on two (www.digitalcitizen.life), which is 378 against 228 seconds, a saving of **39.7%** and a ratio of **1.66**. The two ratios come from different hardware, models and workloads, and the press figures were not traced to an NVIDIA primary page, so they are shown only to illustrate that the outcome varies by configuration. NVIDIA's own caveat, that the demo is configuration-specific and not a general benchmark, applies to every ratio here. No ratio here should be used as a forecast.

Energy arithmetic (Hypothetical Example). Assume each node draws 200 W on average while serving, a figure chosen for illustration and to be replaced by metered power. The single-node run uses 200 W × 0.30 h = 0.060 kWh. The three-node run uses 3 × 200 W × (528/3,600 h) = 0.088 kWh, ignoring idle draw. EIA's Table 5.6.A lists the U.S. total commercial average at 14.47 cents per kWh for August 2026, a preliminary estimate (www.eia.gov). The costs are **$0.0087** and **$0.0127**, a difference of about **$0.004 per run**, or about $4 per thousand runs. For hardware limits, the documented upper bounds are 600 W for an RTX PRO 6000 board (www.nvidia.com) and a 240 W supply for DGX Spark, which are ceilings, not measured draw.

Break-even formula (Hypothetical Example). For owned nodes with zero incremental capital cost, the pilot pays back when N × (S × V minus ΔE × P) exceeds A, where N is the number of runs, S the seconds saved per run in hours, V the hourly value of the waiting time, ΔE the extra energy, P the energy price and A the one-time administration and security cost. With S = 552 s = 0.153 h, V = $50 per hour, ΔE × P = $0.004 and A = 40 hours × $100 = $4,000, the break-even N is about **522 runs**. Every input is assumed. The point of the formula is that energy is negligible, so the result depends on S, V and A, all of which the pilot must measure.

PIN arithmetic. A six-digit PIN space is 1,000,000. A cap of 100 consecutive failures corresponds to 0.01% blind-guess success, and one guess per invitation attempt corresponds to 0.0001%, assuming uniformly random PINs and no attacker control over invitations (the documentation says a mistyped PIN ends the attempt, but also calls retrying cheap, so this figure is an assumption). The vendor says only that the PIN is low entropy.

Eviction window (Hypothetical Example for the request rate). Twelve consecutive misses in about a minute implies a detection cadence near 5 seconds per miss, so an unreachable node can remain a routing candidate for roughly a minute, and a stale-telemetry node for at least the 10 second threshold. At 5 requests per second, that window could misdirect on the order of 300 requests, before retry or failover to the next advertised owner. The rate is an illustrative assumption.

Go/No-Go Gate, Rollback and Unpair Plan

Go criteria for a pilot with non-sensitive data (all required):

- **Release pinning.** Every node on the same signed release, with the installer checksum recorded (the release page publishes SHA-256 digests per asset (github.com)).
- **Segmentation.** Isolated VLAN, host firewalls, a port scan showing only documented ports, and no plaintext response to non-loopback callers.
- **Model governance.** An approved model list with digests, staging-host import, and egress blocked except to the mirror.

- **Asset inventory.** All paired nodes mapped to owners and patch status, consistent with NIST AI RMF inventory expectations (nvlpubs.nist.gov).
- **Test evidence.** A completed failure-injection grid and per-node request, latency and correctness record.

No-go triggers for sensitive prompts (any one blocks): undocumented certificate rotation or revocation, no confirmed PIN retry limit, no client-side audit layer, or a vendor support path that the business cannot accept. Support is community best-effort (github.com), and the roadmap is explicitly not a commitment to delivery (github.com).

Rollback, unpair and uninstall. NVIDIA's rollback guidance is to remove paired systems, uninstall PAIR, and remove engines or models (build.nvidia.com). Operationally:

- **Unpair.** Leaving first is not strictly required, since removal works after the fact (github.com).
- **Uninstall versus data.** Removing PAIR and removing data are separate steps (github.com). Data includes settings, logs, cluster identity, certificates and engine data (github.com), so a purge is needed to destroy certificates.
- **macOS.** Dragging the app to the Trash leaves a privileged helper in place, so the uninstaller must be used (github.com).
- **Residuals.** Delete engine model stores separately and record deletion evidence per node.

Implications and Future Directions

Several release changes show where the product is moving. The 1.0.0 release added llama.cpp support and RTX Spark support (github.com), and it changed local endpoints to follow each engine's cross-origin policy instead of allowing every browser origin (docs.nvidia.com). The launch coverage described a beta (tbreak.com); the repository pages reviewed do not say whether 1.0.0 retains that label, so the maturity status should be confirmed with NVIDIA before any procurement decision.

For governance teams, the larger implication is that "private" has several layers. Prompt confidentiality on the wire is one layer, metadata exposure another, and supply chain a third. The vendor's metadata and discovery exposure is documented, so a reviewer can accept or reject it knowingly. For platform teams, the economic implication is that reusing idle owned systems may help parallel agent workloads, while single-request latency, large models and strict SLOs call for a different design. For procurement, buying more PAIR nodes does not substitute for buying a larger single-node memory footprint, because nothing is pooled.

This report's view is that the right unit of decision is evidence, not product positioning. A pilot that produces a ports capture, a failure grid and a measured per-node record gives a security reviewer something to sign. Teams that need enterprise features such as identity-aware access, quotas and audited retention are likely to pair PAIR-style routing with a gateway rather than rely on it alone, although this report found no documentation of that combination and it would need its own test.

Items to raise with the vendor, and to track across releases, before sensitive prompts are routed:

- **Certificate lifecycle.** Documented expiry, rotation and revocation behavior for the pinned self-signed certificates.
- **PIN handling.** A stated retry limit, lockout and invitation expiry for the six-digit PIN.

- **Scheduler inputs.** Whether GPU model, memory, latency or model load state become routing signals, as the roadmap hints.
- **Client authentication.** Whether the proxy can distinguish callers, since loopback ingress identifies only the local machine.
- **Support path.** An enterprise support and security-advisory channel beyond best-effort community support.

Frequently Asked Questions (FAQs)

Is NVIDIA PAIR an inference router or a GPU cluster? It is a router. Each request is sent to one system, and GPU memory is not combined (build.nvidia.com).

Can PAIR run a model too large for one node? No. PAIR does not combine GPU memory or split a model (build.nvidia.com), and a node can serve only models it holds.

Is it air-gapped? Not by default. As noted above, the playbook requires internet access for downloads, and a press report states that internet access is not required to operate PAIR once models are downloaded (tbreak.com). The two statements are consistent only if the whole artifact and update path is staged offline, which an operator must verify.

Which network segment may host it? An isolated VLAN with host firewalls, because telemetry is plaintext and readable on the subnet and the project says segmentation and firewall rules remain the operator's responsibility (github.com).

What does the private AI controls checklist need? Segmentation, pinned releases, approved model digests, an audit-layer decision, per-node SLO records and a rollback plan, as laid out above.

Does the scheduler favor the fastest GPU? No. Table 2 shows that GPU model, available memory and measured latency are not inputs.

How does it compare with an enterprise gateway for local LLM routing? Gateways document identity, spend tracking, health-based removal and latency-aware strategies (docs.litellm.ai), which PAIR does not.

Conclusion

PAIR is best understood as a heterogeneous request router with a documented, deliberately modest trust model. Its strengths are simple deployment, loopback ingress, mTLS between paired nodes and a clear statement of what it does not do. Its limits are equally documented: capacity-blind scheduling, plaintext node telemetry and discovery, a PIN-bootstrapped trust, undocumented certificate lifecycle and best-effort support. A controlled pilot on an isolated segment with approved models is reasonable, and routing sensitive prompts through it is premature until the evidence listed in the go/no-go gate exists. Where measured parallel gains are small or unattributable, a single inference server or a gateway is the safer design.

Source links

Links cited in this article, in order of first appearance. Full addresses are provided for readers using extracted text.

1. https://developer.nvidia.com/blog/nvidia-pair-virtual-inference-router-expands-available-compute-on-your-local-network
2. https://github.com/NVIDIA/Personal-AI-Router/releases
3. https://build.nvidia.com/rtx/pair/overview
4. https://docs.nvidia.com/local-ai/nvpair
5. https://docs.nvidia.com/local-ai/nvpair/known-issues
6. https://docs.nvidia.com/local-ai/nvpair/architecture
7. https://github.com/NVIDIA/Personal-AI-Router/blob/main/SECURITY.md
8. https://github.com/NVIDIA/Personal-AI-Router/blob/main/SUPPORT.md
9. https://www.eia.gov/electricity/monthly/epm_table_grapher.php?t=epmt_5_6_a
10. https://build.nvidia.com/spark/pair
11. https://tbreak.com/nvidia-personal-ai-router-local-ai-cluster
12. https://github.com/NVIDIA/Personal-AI-Router
13. https://www.nvidia.com/en-us/products/workstations/professional-desktop-gpus/rtx-pro-6000
14. https://docs.nvidia.com/dgx/dgx-spark/hardware.html
15. https://www.nvidia.com/en-us/products/workstations/dgx-spark
16. https://docs.ollama.com/api/tags
17. https://www.rfc-editor.org/rfc/rfc6762.txt
18. https://docs.ollama.com/faq
19. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf
20. https://doi.org/10.6028/NIST.SP.800-53r5
21. https://media.defense.gov/2024/Apr/15/2003439257/-1/-1/0/CSI-DEPLOYING-AI-SYSTEMS-SECURELY.PDF
22. https://www.rfc-editor.org/rfc/rfc6763.txt
23. https://pages.nist.gov/800-63-4/sp800-63b.html
24. https://docs.nvidia.com/local-ai/nvpair/troubleshooting
25. https://www.rfc-editor.org/rfc/rfc8446.txt
26. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-52r2.pdf
27. https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.100-1.pdf
28. https://lmstudio.ai/docs/app/offline
29. https://docs.nvidia.com/local-ai/nvpair/engine-lifecycle
30. https://github.com/ggml-org/llama.cpp/blob/master/tools/server/README.md
31. https://genai.owasp.org/llmrisk/llm032025-supply-chain
32. https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.600-1.pdf
33. https://docs.nvidia.com/local-ai/nvpair/inference-dispatcher
34. https://genai.owasp.org/llmrisk/llm022025-sensitive-information-disclosure
35. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-92.pdf
36. https://docs.nvidia.com/local-ai/nvpair/getting-started
37. https://lmstudio.ai/docs/developer/core/ttl-and-auto-evict

38. https://sre.google/sre-book/service-level-objectives
39. https://sre.google/workbook/implementing-slos
40. https://raw.githubusercontent.com/open-telemetry/semantic-conventions-genai/main/docs/gen-ai/gen-ai-metrics.md
41. https://mlcommons.org/benchmarks/inference-datacenter
42. https://raw.githubusercontent.com/kubernetes-sigs/gateway-api-inference-extension/main/docs/proposals/003-model-server-protocol/README.md
43. https://lmstudio.ai/docs/developer/core/authentication
44. https://docs.litellm.ai/docs/proxy/virtual_keys
45. https://docs.litellm.ai/docs/enterprise
46. https://docs.litellm.ai/docs/routing
47. https://docs.litellm.ai/docs/proxy/health_check_routing
48. https://docs.vllm.ai/projects/production-stack/en/latest
49. https://gateway-api-inference-extension.sigs.k8s.io
50. https://www.digitalcitizen.life/nvidia-pair-can-turn-multiple-home-pcs-into-a-shared-local-ai-cluster
51. https://docs.nvidia.com/local-ai/nvpair/release-notes

THE PUBLISHER

About GPU Smith

GPU Smith provides independent engineering and research for private AI infrastructure. We help technical buyers and operators reason about GPU workload sizing, hardware selection, procurement, deployment and inference operations, from the compute system to the networking, power and cooling requirements around it.

Start with the workload

Useful infrastructure decisions begin with the models, data, throughput, latency and operating constraints a team actually has. GPU Smith helps connect those requirements with system design and deployment choices. The goal is a privately controlled AI environment whose capacity and operational demands are understood before a purchasing decision.

Hardware and supplier research

Our public reference library covers GPUs, complete systems and networking components. The server-vendor directory and manufacturing research help teams investigate suppliers, compare options and follow sources behind technical claims. We distinguish manufacturer specifications from measured benchmarks and advertised capabilities from tested configurations. Reference pages are research resources, not a live inventory listing or a binding equipment quote.

Deployment and operations

GPU Smith's engineering scope includes infrastructure integration, networking, power, cooling and the practical operation of inference workloads. Our published articles explain the tradeoffs behind deployment, procurement and ongoing operations so teams can ask better questions and document decisions.

Work with GPU Smith

Explore the hardware library, GPU references, networking references and vendor research. Contact GPU Smith with your workloads and deployment constraints to discuss a project and confirm current scope, pricing and availability.

Inclusion of a manufacturer or product in our research does not imply a partnership, certification or endorsement.

Website: https://gpusmith.com

This article is educational. Verify consequential medical, legal, financial, regulatory and technical decisions with appropriate professionals and the cited primary sources. Company services are described separately from the article's research findings.