Fingerprinting All AI Cluster I/O Without Mutually Trusted Processors

Naci Cankaya [1 2] **Jakub Kryś** [3] **Jonathan Ng** [4] **Luke Marks** [5] **Felix Krueckel** [6]

Abstract

In preparation for potential international agreements on artificial intelligence, the development of verification infrastructure for AI data centres is vital. We propose a method for cryptographically committing all information entering and leaving a data centre: Hashes are computed by network taps placed on all the information-carrying wires between the cluster and the outside world, enabling an auditor to retroactively challenge the preimage data to be sent to a privacy-preserving verification facility performing compliance checks. Our goal is to make it infeasible to covertly exfiltrate the results of undisclosed workloads in the cluster through the tapped wires. To this end, we specify the architecture of a "Secure Gateway Device", which handles the erasure of covert channels that post-hoc verification on hashed data cannot address: analogue and timing side-channels, as well as steganography in network protocol headers. The architecture eliminates the need for any processors trusted by both the Prover and the Verifier, leveraging passive optical fibre splitters and coin-flip protocols for random number generation where needed. We expect development costs of a demonstration device to be roughly equivalent to the cost of a small team of engineers for a few months, with a comparatively small bill of materials.

1. Introduction

The rapid progress of Artificial Intelligence (AI) in recent years has opened the door to accelerating advancements in multiple fields including medicine, engineering and computer science. However, as a very powerful general-purpose tool, AI also exacerbates existing concerns around technology misuse, as well as introduces novel risk vectors such as the threat of autonomous misaligned agents (Bengio et al., 2026). For this reason, several jurisdictions have already introduced legally-binding regulations governing the acceptable uses, transparency obligations or testing requirements of generative AI systems (Cyberspace Administration of China, 2023; European Parliament and Council of the European Union, 2024; Wiener, 2025). Additionally, in the future the need and political will for treaties and agreements might also arise between different jurisdictions. For example, in recognition of the tremendous power afforded by AI, leading actors in this technology competition could agree on 'red lines' that neither of them should cross (Centre pour la Sécurité de l'IA (CeSIA) & The Future Society, 2025). These may emulate historical agreements in other safety-critical areas such as The Nuclear Non-Proliferation Treaty or START.

Regardless of the nature of such legislation or treaties, there is a clear need for the ability to verify compliance with agreements on AI by all affected parties. This problem is particularly important and challenging in the context of AI, since computing power as a resource has a much more dual-use nature than fissile materials or even DNA synthesis tools (Sastry et al., 2024). Advanced artificial intelligence is seen as a decisive strategic asset and economic engine (National Security Commission on Artificial Intelligence, 2021), and the question of whether unilateral restraint in AI development and deployment is compatible with national security is contested.[1]

Consequently, the field of AI Verification has recently emerged as a research agenda spanning both technical and governance aspects. The importance of robust verification mechanisms is reflected by the appearance in literature of three major works detailing open problems, policy goals and

[1]MATS [2]Oxford Hardware AI Governance Lab [3]SaferAI [4]Independent [5]Martian [6]RWTH Aachen University, Aachen, Germany. Correspondence to: Naci Cankaya <naci.c@protonmail.com>.

Second Workshop on Technical AI Governance Research (TAIGR) @ ICML 2026, Seoul, South Korea. 2026. Copyright 2026 by the author(s).

[1]US Vice President Vance, when asked about pausing dangerous AI development: "The honest answer to that is that I don't know, because part of this arms race component is if we take a pause, does the People's Republic of China not take a pause? And then we find ourselves all enslaved to P.R.C.-mediated A.I.?" (Douthat, 2025)

possible research directions in this area (Scher & Thiergart, 2024; Harack et al., 2025; Baker et al., 2025). These are presented concisely in Figure 1.

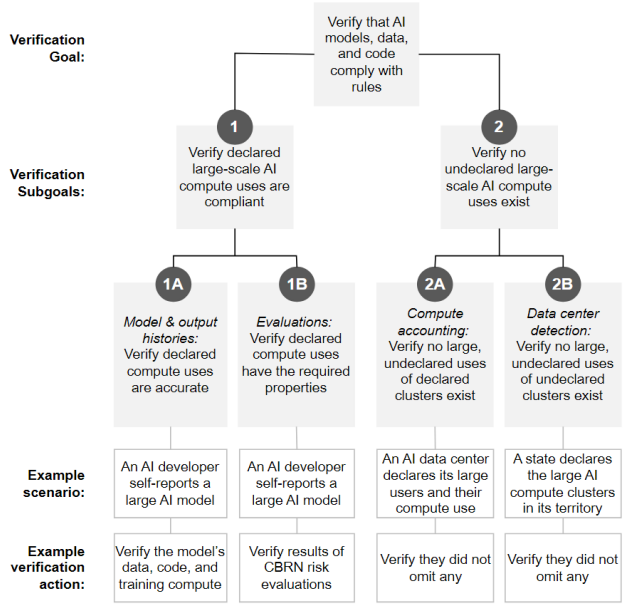

Figure 1. A breakdown of the AI Verification research agenda into its goals and subgoals. Original from Baker et al. (2025) with the authors' permission. In this work, we focus on verification Subgoals 1A and 2A.

In this work, our main focus is on Subgoals 1A and 2A of Baker et al. (2025), which can be summarised as follows: an entity operating a computing cluster — the Prover — aims to prove to the auditing party — the Verifier — that no undeclared workloads were executed on said cluster. The Verifier checks if the evidence was generated in a way the Prover can not tamper with, and if the claims are self-consistent/accurate.

We do not engage with the problem of determining which workloads should be allowed or not in the first place (regulation) or of verifying whether the declared workloads are compliant (Subgoal 1B). However, our proposed method offers raw evidence of compute use, which is needed for the downstream verification of correctness (Subgoal 1A) and compliance (Subgoal 1B). The issue of *preventing undeclared compute resources* (Subgoal 2B) is fully outside of the scope of this work.

Proposed solutions for verifying compliance and comprehensiveness include on-chip mechanisms such as Trusted Execution Environments, HBM-based chiplets that monitor memory traffic (Petrie, 2025; Aarne et al., 2024), and tamper-proof enclosures for accelerators (Happel, 2026).[2]

[2]For a thorough overview of possible directions, we refer the reader to the three-part series by the FlexHEG initiative (Petrie & Aarne, 2025).

Off-chip, network-level approaches have also been described: Baker et al. (2025) detail an explicit network-tap layer, Harack et al. (2025) introduce "digital perimeters" that commit to cluster traffic, and Scher & Thiergart (2024) explore how external networking equipment can enforce workload-classifying bandwidth limits. We build directly on these off-chip paradigms. Instead of targeting individual accelerators or servers, we operate at the level of the whole cluster, focusing on the communication chokepoint between the hardware and the Prover—the data centre north-south uplink (connecting the data centre's clusters with the internet/outside world). If the Verifier can capture, hash and timestamp all traffic in and out of the data centre, and the Prover can — upon later request — demonstrate that their declared workloads correspond to this hashed traffic, the Verifier can be satisfied that the Prover is not performing undeclared computations in the cluster. To achieve this goal, we propose a schematic design of a "Secure Gateway Device" (SeGaDev) that uses active and passive network taps to capture and process such traffic for later verification in a mutually trusted manner.

Main contributions.

- A detailed description of the problem statement and the design requirement for a verification mechanism meant to address Subgoals 1A and 2A of Baker et al. (2025) (Section 3).
- An actionable design specification of our proposed device that addresses these requirements. This includes a detailed description of how the device processes captured traffic in order to eliminate steganographic degrees of freedom in egress data (Section 4).
- A feasibility assessment of this solution, including estimating the storage requirements needed for the associated verification protocol (Section 5 and Appendix B).

2. Background and Related Work

The security problem of covert communication through an observed link is an active area of research. We differentiate two attack types: side-channels and steganography.

The first exploits physical channels other than the ones used regularly by a device. For example, while an optical fibre link may be specified to only use amplitude modulation for communication, additional information can be encoded in polarization or phase, if the outside observer is not considering them. One physical side-channel that is not trivially closed by active network taps is timing modulation, which is explored by Lee et al. (2014) on the offensive side and Uttarwar & Dakhane (2025) on the defensive side. We address this side channel in our proposed solution.

The second channel, steganography, does not hide any signal directly from the observer. Rather, it encodes hidden messages in plaintext. An intuitive example for this is hiding messages in the second character of every word in a text. Applied to Ethernet links (which are the practically universal standard for internet communication), steganography can operate on multiple levels. Jankowski et al. (2013) demonstrate this at the protocol layer, encoding secret messages in padding bytes (usually only used to ensure a minimum length of network packets).

To defend against both channels, so-called "Active Wardens" were first conceptualized (Simmons, 1984; Anderson & Petitcolas, 1998), prototyped in software (Fisk et al., 2003) and, more recently, built with re-purposed network switching hardware (Xing et al., 2020). Active Wardens are devices that intercept and re-emit traffic (closing channel 1), while re-writing free fields (closing channel 2 for those free fields). We draw from this research on Active Wardens for erasing steganography channels in Ethernet protocol headers, as well as work from Uttarwar & Dakhane (2025) for erasing timing side-channels.

While ensuring that computations can be faithfully replayed and verified is not part of our project scope, we refer to Karvonen et al. (2025) and Cankaya (2025) for highlighting the importance of this downstream challenge for closing off exploitable degrees of freedom (Rinberg et al., 2025) in the non-deterministic outputs of ML computations.

We discuss our estimates for covert bandwidths in Section 5.

3. Problem Statement

For our proposed device, we set the following requirements:

- The device needs to capture all of the ingress/egress traffic of a data centre. This means it is to be placed at the north-south part of the data centre network, intercepting every link with the outside world.[3]
- It must be mutually trusted and/or monitorable by both Prover and the Verifier. This means that:
  - The Prover must not have the ability to hide significant amounts of information in the traffic being monitored. Otherwise, they could transfer undeclared workloads out of the data centre using steganographic techniques, side-channels, timing attacks, and similar.
  - The Verifier must not have the ability to extract Prover's proprietary information out of the device. Concretely, the Verifier receives hashes, and the Prover needs confirmation that these are computed and transmitted correctly.
- The solution should be retrofittable to realistic, full-scale deployments at the scale of $O(100\text{k})$ accelerators and beyond. To the extent possible given the other requirements, it should rely on existing and affordable technologies.
- The device should introduce as little additional latency as possible in order to minimise the slowdown of usual data centre operations. It should be able to process millions of communication packets streaming in/out every second.[4]

We are working under the following assumptions:

- We focus on traffic serving LLM inference API endpoints to external customers. We also assume API traffic egress is at least on the same order of magnitude as ingress, if not larger. This is because customer traffic served via the 'streaming API' format, commonly used for chatbot applications, is optimised for low latency in delivering responses to customers at the cost of high data inefficiency. Every single output token is wrapped in a JSON packaging, adding $\sim$200 bytes of overhead. (OpenAI, 2026)[5]
- Steganography via input payload data (i.e., the plaintext requests to the LLM) is unconstrained.[6] User and developer inputs are free-form and inherently high-entropy.
- The Prover needs to store all information that forms a part of the hash for as long as the agreement requires, or as long as the Prover wants to be able to demonstrate compliance with rules. In practice, this includes everything the SeGaDev is configured to hash. Our working assumption is a full capture of the payload data (Ethernet protocol layer 7), which includes everything but the connection metadata (which is actively sanitised by the Active Warden part of the SeGaDev).

[3] Strictly speaking, one can parallelize this task across many SeGaDevs. This is an engineering decision with potential tradeoffs regarding the number of tapped links per device.

[4] This is the minimum required to handle a single 100G link, which is a typical bandwidth used at a data centre edge. Though as noted in the previous footnote, one SeGaDev can handle multiple links if adequately equipped.

[5] Strictly speaking, streaming responses may contain batches of a few tokens, although in practice the number is often one. We take this simplifying assumption because we are also interested in a worst-case feasibility assessment of the SeGaDev solution. With only 1 token per streaming response, the amount of data produced represents an upper bound for the storage and bandwidth requirements of our method.

[6] The Warden device could in principle rewrite prompts and input media files while preserving semantics. However, this carries complications related to a potential degradation of the inference service quality. We leave this open for future research.

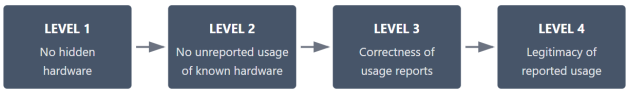

Figure 2. We are specifically aiming for Level 2: capturing and safely transmitting raw evidence of computation (only hashes, not plaintext). The hashes, together with the plaintext data they point to, later feed into a secure cluster trusted by the Verifier, without the plaintext information being shown to them directly.

Out of scope. In order to convey the nuance of the problems we are — and are not — tackling, we briefly explain the "verification pipeline" we are assuming. This pipeline consists of four separate stages outlined in Figure 2.

The verification of original data in a dedicated, separate cluster is twofold and out of scope for us: (1) replaying observed inputs and outputs to confirm they match precisely (Level 3), and (2) evaluating the computations for legal compliance (Level 4, corresponding to Subgoal 1B in Baker et al. (2025)).[7] While the specific replay protocol is out of scope here, we refer to related work on confidential-computing facilities where the Prover replays challenged computations under attestation and governed egress (Tlaie Boria, 2026). The Verifier can challenge stored hashes at arbitrary cadence and need not challenge every record.[8]

We do not attempt to solve the physical monitoring of either the Prover's cluster or the verification facility needed to prevent secret communication devices. Also, we do not consider steganographic degrees of freedom in unverifiable information contained in packet payloads (e.g. a limit on GPU temperature diagnostics leaving the facility). These degrees of freedom are not for our SeGaDev to erase, but for rules to constrain.[9]

We only present a network tap at the north-south/frontend network. This does not (directly) prevent covert workloads, but rather the exfiltration of their results. Additional network taps in the east-west/backend network can offer defense-in-depth by directly capturing tensors used in computation. We detail their feasibility and security properties in forthcoming work.

4. Architecture

In this section, we describe the placement and design of the SeGaDev — our evidence capture device.

[7]There are challenges with precise replay. See Section 5.

[8]k randomly sampled hashes detect a fraction p_{false} of misreported records with probability $1-(1-p_{\text{false}})^k$, independent of total throughput. For instance, $k = 3{,}200$ challenges detect 0.1% misreporting at 96% confidence.

[9]Since this information is hashed, compliance checking here is trivial: checking if the egress data contains illicit amounts of unverifiable data such as precise timestamps, nonces, GPU temperature and fan speed, power supply metrics etc.

4.1. The Macroarchitecture (Placement of the Secure Gateway Device)

Lifecycle of a packet going through the macroarchitecture. Traffic between the isolated AI data centre and the outside world (which includes company-internal peering metros) routes through a Wide Area Network (WAN) backbone router.[10] This router marks the physical north-south boundary of the data centre network. Because modern distributed architectures encrypt all internal transit, packets passing through this router arrive as ciphertext — typically encrypted via mutual TLS (mTLS) or proprietary internal protocols. Once inside the facility, this transport encryption is terminated at a frontend gateway. From there, the decrypted internal traffic is routed through the internal data centre fabric down to the individual compute units. This stage is where an SeGaDev could be placed to intercept plaintext traffic. We illustrate this in Figure 3, placing the device between "TLS Termination" and "Internal Routing Fabric". The decrypted packet traverses the SeGaDev. Protocol headers are scrubbed of degrees of freedom (port choices, initial sequence numbers (ISNs), timing) while application payloads are hashed for later verification (we detail the sanitization and hashing process happening inside this device in the next section). From there, the sanitised packet travels through the data centre-internal fabric to the individual compute unit (server, rack etc.). A compute unit transmitting results back is practically the same process in reverse, taking the same overall path.

This macroarchitecture relies on the following three assumptions:

- The facility is under physical monitoring, ensuring that any traffic in and out of the cluster needs to pass through the SeGaDev chokepoint.
- Every bit of traffic sent to the Verifier (hashes only) can be passively observed by the Prover. This places the root of trust in physics: beam splitting via fused biconical taper (FBT) or planar lightwave circuit (PLC) splitters and optical isolators ensure the Prover can observe and double-check that hashes have been computed exactly as they expect, without being able to interfere with the Verifier's monitor leg. Timestamps

[10]While early network security models relied on single, centralized hardware chokepoints, modern hyperscale architectures like Google's Espresso demonstrate that line-rate packet processing at the terabit scale must be decentralized (Yap et al., 2017). Following this precedent, the SeGaDev would likely not be deployed as a single monolithic device. Instead, stateful operations (such as TCP sequence number translation and port randomisation) would be implemented as distributed software packet processors at a lower level, perhaps one per pod (still thousands of TPUs). This shards the state-table memory requirements across more devices, avoiding the extreme state-table memory needed in centralized hardware appliances at the cluster level.

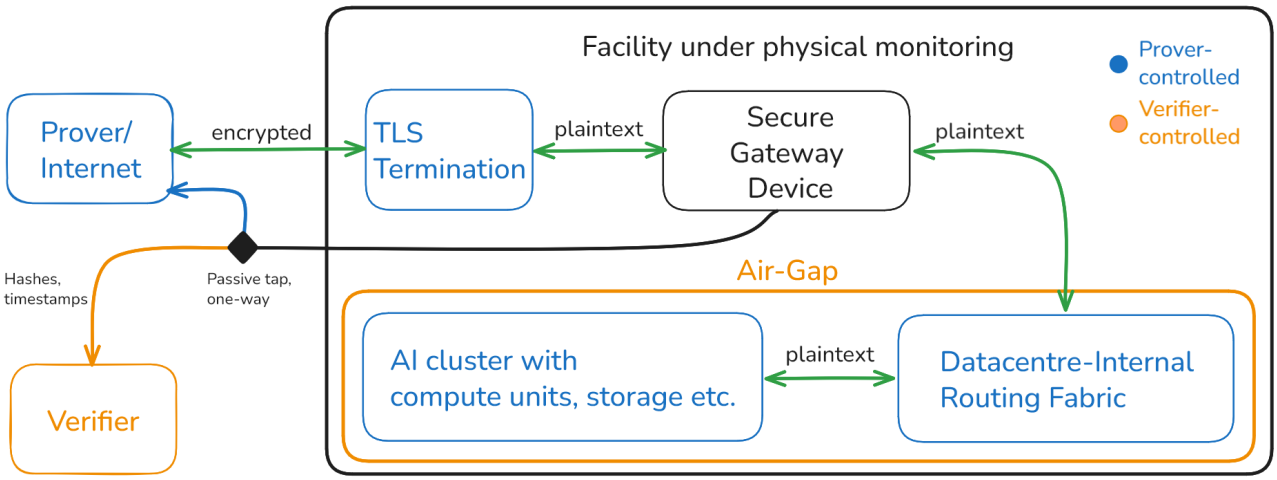

Figure 3. The placement and main task of the SeGaDev. Here, it is shown as an interceptor between TLS termination point (encryption boundary) and the data centre fabric routing traffic to individual compute units. This placement is not the only option: one could also place network taps between the edge router(s) and the compute units, at any level of the north-south network hierarchy. The general principle is that all accelerators are air-gapped, apart from links eventually passing through the SeGaDev. Note the passive splitter at the link carrying hashes to the Verifier: the Prover can capture and double-check the hashes, ensuring no other information reaches the Verifier. Components drawn in orange represent components that the Verifier relies on to behave correctly, and components drawn in blue represent the same for the Prover.

can be computed by both parties upon receiving the hashes at practically the same time.

Figure 3 represents a common case where transport encryption terminates before compute units. However, if the data centre additionally uses end-to-end encryption on internal links, the SeGaDev captures and hashes ciphertext while still scrubbing unencrypted protocol headers. The Prover stores session keys alongside their plaintext records, enabling a verification cluster to decrypt captured traffic after the sessions have concluded and the keys are not in active use.[11,12]

4.2. The Microarchitecture (the Secure Gateway Device Internals)

When a packet reaches the SeGaDev, it passes through two stages: sanitization and hashing. The hashes likewise pass through a warden for re-timing, eliminating potential covert communication to the Verifier. A key security feature used within the device is enabled by the passive (optical) splitters providing observability of untrusted components. In this architecture, no individual processor requires mutual trust. This property is valuable because establishing mutual trust in a single hardware component is difficult: Ilhan et al. (2026) survey design- and fabrication-stage verification mechanisms and conclude that no single technique provides sufficient assurance against a high-capability adversary, with attack vectors including malicious third-party IP, dopant-level Trojans, and package-level die substitution, none of which are patchable post-deployment. By removing the need for such jointly trusted hardware, our design sidesteps this entire class of supply-chain integrity problems. In Figure 4, we colour-code the minimum device trust assumptions. Transparent design and supply chains could also establish mutual trust in components, which would further add defense in depth, but our architecture does not rely on this.

Lifecycle of a packet going through the device.

Coin flip protocol. This stage prepares pseudorandom numbers for the warden to use for sanitizing packets, before these packets arrive. Standard network protocols contain unconstrained header fields such as ephemeral source ports, TCP Initial Sequence Numbers (ISNs), and session IDs. While replacing these values with a deterministic, post-hoc verifiable heuristic would eliminate the covert channel, doing so would fundamentally compromise the protocol's security — predictability in fields like TCP ISNs or encryption keys exposes connections to sequence prediction attacks and session hijacking.

[11] For unfamiliar readers, TLS is the encryption between the outside world and the data centre. Unencrypted communication within data centres is not unusual (Microsoft, 2024).

[12] If the Prover can freely choose encryption keys, this could be a steganography channel. One could require a deterministic key generation heuristic, but there may be security issues with such predictability. A solution may be to handle this via a coin flip protocol between the Prover and Verifier. A buffer of keys generated in the same way the SeGaDev generates ISNs.

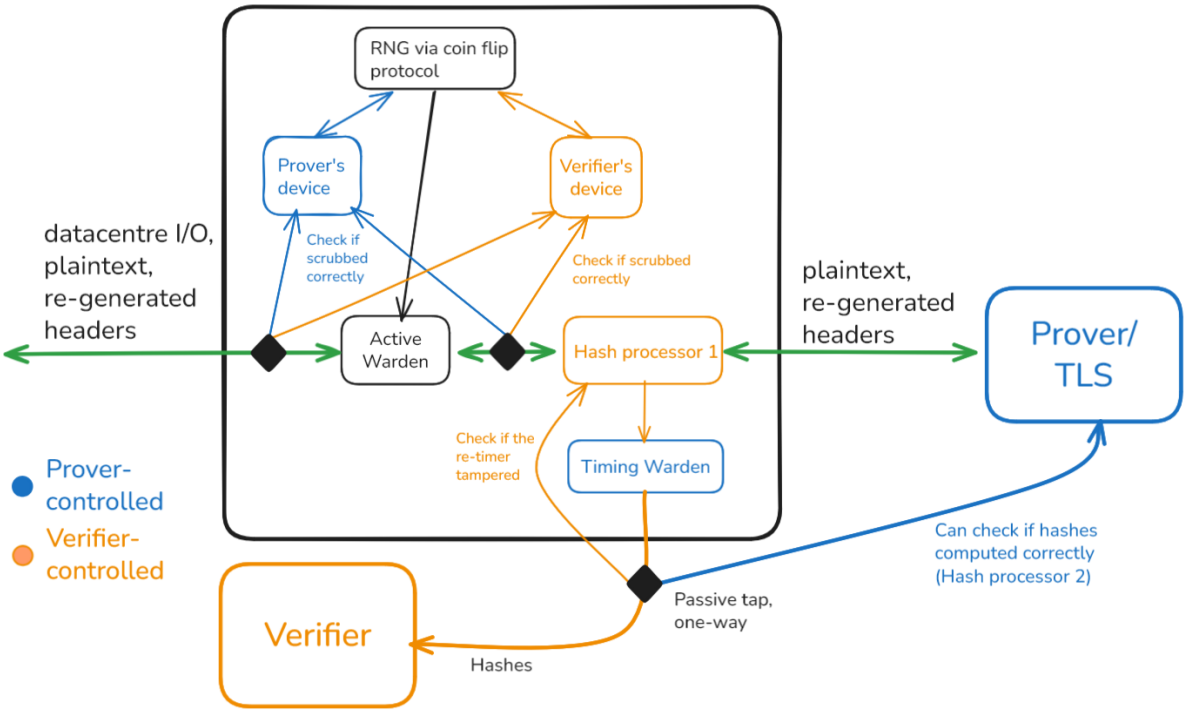

Figure 4. When north-south egress traffic from the data centre hits the SeGaDev, we first scrub non-reproducible fields with an Active Warden. Where necessary, we use a coin flip protocol as a source of randomness. Traffic leaving the Active Warden is hashed and forwarded. Components drawn in orange represent components that the Verifier relies on to behave correctly, and components drawn in blue represent the same for the Prover.

The Active Warden overwrites all such fields with pseudo-random values. If either party controlled this randomness, they could use it as a covert channel. This motivates introducing the coin-flip protocol, which can be summarised as follows. Both the Prover and the Verifier contribute their own number-generating devices for participating in the protocol. Each device first commits to a value (or a list of values), both reveal it after commitment, and the shared output is computed from both (e.g. using XOR on every bit). These values are pre-computed in bulk and consumed in strict FIFO order ('first in, first out'), so that no per-packet latency is introduced. Both devices can verify that the warden uses the correct value for each field by observing the sanitised output through their passive taps.

Sanitization (Active Warden). Using the pre-generated random values (and heuristics where applicable, e.g. setting all padding bits to zero), the warden replaces all exploitable header fields and maintains bidirectional connection-tracking tables so that normal communication is preserved (see Appendix E). This is known in literature as an Active Warden Fisk et al. (2003) and Xing et al. (2020): a device that scrubs steganographic degrees of freedom from transit traffic. Prior implementations assume single-party trust in the device; our coin-flip protocol and passive observability eliminate this requirement.

An important part of sanitization is re-timing: even after header sanitization, the timing of packet transmission can carry information. An adversary could modulate inter-packet gaps or exploit analogue properties of the optical signal. The device buffers all packets and releases them on a fixed schedule at maximum link rate (PHY-Covert Channels (Lee et al., 2014)). After this sanitization, traffic can be forwarded and leave the SeGaDev. Since hashes are computed only on the payloads, rather than also on the headers scrubbed by the Warden, the order of hashing and sanitization is not important and can differ between egress and ingress (in turn simplifying the design).

Hashing. The packet is fed to a hash processor that computes a cryptographic commitment over a configurable group of packets referred to as the 'hashing boundary' (see Appendix A). These hashes — and only these hashes — are transmitted to the Verifier over a dedicated fibre link. A passive optical splitter on this link gives the Prover a copy, allowing them to independently verify that no information beyond hashes reaches the Verifier. The hash output itself passes through a separate, Prover-controlled re-timing stage ("Timing Warden" in Figure 4) to prevent the hash processor from covertly communicating with the Verifier through timing or optical modulation. The Verifier's hash processor, in turn, can check if the Prover's re-timer tampered with

the hashes, by also using a passive splitter on the re-timer's egress.

5. Covert Channel Estimates

We consider only covert channels through the SeGaDev in scope. Physical air-gapping of the monitored cluster —apart from the monitored links— is a requirement, but a separate challenge.

Making exfiltration of even single bits outright impossible is likely futile, as very low-bandwidth covert channels are inherently offense-dominant. The residual risk is not bulk data exfiltration (practically all meaningful computation output far exceeds what very low bandwidth channels can carry), but rather the combination of covert internal computation with minimal-bandwidth signalling of results. Defending against this requires complementary verification of compute utilisation (e.g. backend/east-west network taps, throughput accounting, numerical fingerprinting, etc.) (Cankaya, 2025), which is outside the scope of the SeGaDev.

After the mitigations achieved via the Active Warden within the SeGaDev, we expect the following residual channels to remain:

- Remaining timing/order modulation
- Steganography in ML outputs
- Failure channel: not responding to select inputs and covering it up as "hardware failures" or "software errors"

5.1. Timing Modulation

Re-timing in the SeGaDev can upper-bound the frequency of available timing modulation channels, depending on the buffer windows. With a 10 ms buffer window at 100 Gbps and ≈200-byte streaming packets, each window contains up to ≈625k packets. The attacker can modulate packet count per window, yielding up to $\log_2(625\text{k}) \approx 19$ bits/window. However, the dominant channel is packet ordering: unlike the single-flow threat model assumed in most covert channel literature (Uttarwar & Dakhane, 2025; Xing et al., 2020; Park et al., 2023), in our setting the Prover controls all egress flows, making cross-connection interleaving a covert channel of potentially very high capacity, motivating the fixed-frame-count and canonical-ordering mitigations below.[13]

[13] A covert communication method where secret information is encoded by strategically manipulating the sequence in which packets from multiple independent data streams are merged onto a single outgoing network link.

Mitigations: By fixing a packet count per connection per window and giving a canonical ordering to the connections assigned by the Warden, we can make the packet stream for a given connection constant-rate and deterministic. The remaining covert bandwidth from timing modulation reduces to connection lifecycle events (opening/closing a given connection), reducing the covert bandwidth substantially.

5.2. Steganography in ML Outputs

Non-determinism of ML outputs offers steganographic degrees of freedom for the Prover, which have been studied in related literature. With replay verification using seed synchronization (Rinberg et al., 2025), the residual covert bandwidth through ML inference output tokens is approximately 0.1 bits per token.[14,15] Using this estimate, for a large inference cluster (200k GPUs at 2000 tokens/GPU/second) at maximum capacity, we calculate a covert egress bandwidth of approximately 40 Mbit/s, sufficient to exfiltrate a 1 TB model in under 3 days. Though this is the worst-case assumption of full inference utilization and the attacker having full control over all the outputs, it highlights the significance of the payload-steganography channel.

Mitigations:

- **Eliminate the non-determinism.** The 0.1 bits/token residual exists because replay verification tolerates some deviations originating from floating-point non-associativity. Recent work shows this floor can be pushed toward zero. He & Thinking Machines Lab (2025) and Cankaya (2025) show that within fixed conditions (same hardware, same kernels, same batch size) inference is already bitwise reproducible even for many performance-maximizing inference engines and settings. DeepSeek-AI (2026) go further, open-sourcing end-to-end batch-invariant and deterministic kernels with supposedly minimal performance overhead in a frontier production MoE model, which demonstrates that determinism is not confined to research demonstrations and can be maintained at scale. The practical implication is that if the Prover is required to log their full metadata (batch composition, kernel versions, parallelism configuration, sampling seeds), the Verifier can either:
  - replay the computation on identical hardware

[14] Making "temperature" sampling, as often used in generative AI, verifiable.

[15] (Rinberg et al., 2025) report $< 0.5\%$ exfiltrateable information for Qwen3-30B-A3B at 10^{-4} FPR using three-tier FSSL + logit-rank classification. Their metric normalises against the theoretical maximum of $\log_2 |V| \approx 17$ bits/token ($|V| = 151{,}936$ for Qwen3-30B-A3B). This gives $0.005 \times 17 \approx 0.1$ bits/token, i.e. a $\geq 200\times$ slowdown over the unverified channel.

without needing to trust it, by controlling information flow using network taps similar to the SeGaDev (Cankaya, 2026); or
  - emulate the Prover's hardware behaviour and the forward passes using deterministic software. This option would build on existing simulation work for individual tensor cores (Khattak & Mikaitis, 2025) and expand the scope to whole inference pipelines.

- **Active Warden scrubbing of hardware-related entropy.** Since only the trailing bits of token likelihood predictions are affected by hardware non-associativity, in principle it should be possible to add just enough noise to scrub any fluctuations at this order of magnitude, while still maintaining inference quality.[16]

5.3. Failures and Errors

The Prover controls when connections interrupt and hardware "fails". From the perspective of an external observer like the Verifier, a failure is the absence of a complete response. Public data suggests ≈50,000 GPU-hours per unexpected hardware failure (Dubey et al., 2024), predicting ≈96 failures/day for a 200k GPU cluster.[17] Assuming the Prover generates responses of up to 8000 tokens, each failure can encode at most $\log_2(8000) \approx 13$ bits in the choice of the interruption point. At baseline rates, this yields ≈1 kbit/day. Note that this is the worst-case assumption of full inference throughput across the entire data centre.

Mitigations: Screen the data already hashed by the SeGaDev for suspicious failure rates. If the covert bandwidth is still considered intolerable, the Warden could inject "pseudo-failures" indistinguishable from real ones, as seen from the outside.

6. Limitations and Future Work

Empirical validation of the setup is still required.[18]

We expect the bill of materials of a somewhat representative device (FPGAs or NICs, transceivers, memory, custom mainboards and optical components) to cost less than the researcher's and engineer's time needed to build it. A small-scale demonstration can be as basic as a single inference server, FPGA-based SeGaDev and a verification server, while still being representative of larger scales (where one would add more of the same hardware).

The design of the SeGaDev, as well as the broader monitoring scheme, should undergo advanced red-teaming carried out by specialists in fields such as cybersecurity and physical security.

The whole architecture depends on the assumption that the SeGaDev is the only communication channel. Other covert channels exist beyond physical monitoring scope: RF emanation, power line signaling or even thermal channels from a facility that the Prover physically controls. The security problem is structurally similar to an insider threat, since the aim is to prevent secret communication between the Prover and their own hardware. We refer the reader to related works by Guri (2024) and Park et al. (2023).

Until solutions for imperfect reproducibility of ML outputs are found, steganography in output payloads dominates all other residual channels by orders of magnitude, making it the primary target for follow-on work.

We do not constrain bandwidth for covert communication *into* the data centre. While the input data has many uncontrollable variables (media input, user prompts, etc.), input Wardens may re-write such information while preserving semantics. We leave this to future work. Threat modeling for very low outbound bandwidth — but still impactful — covert workloads in the data centre would further provide (or negate) motivation for such additional traffic sanitization.

Development and testing of network taps for backend traffic within the data centre can add defense in depth on top of air-gapping and combating covert channels through the SeGaDev.

[16] This may need to be interactive, though: autoregressive generation means that a sampled token becomes a prior for the next generation steps. This adds a round trip between a Warden and the compute cluster, and therefore some latency: the Warden round trip (send ∼50 filtered logits, receive selected token ID) adds microseconds per token (typical network latency for short-mid-range Ethernet). Against a ∼30 ms decode step, this is $< 0.02\%$ overhead and falls entirely within the 10–100 ms re-timing jitter already applied for timing channel suppression. Bandwidth is ∼400 kB/s at 2000 tokens/s/GPU, trivial for wardens on 100 GbE links.

[17] From Meta's Llama 3 training (Dubey et al., 2024) on 16,384 H100 GPUs (419 interruptions over 54 days). This reflects hardware-level reliability and is used as an order-of-magnitude proxy; we found no inference-specific GPU MTBF (Mean Time Between Failures) data.

[18] Still, we presented strong precedents for the individual components, and Active Wardens have been built to handle 100G line rates (Xing et al., 2020), which would in fact be over-provisioned for most long-range internet fibre connections. The per-connection state for TCP/UDP scrubbing (port translation tables, ISN offsets, connection phase tracking) requires approximately 200 bytes per connection (Appendix E). Xing et al. demonstrated that programmable switch SRAM can sustain 200,000 concurrent connection states with comparable per-connection footprint (Xing et al., 2020), well exceeding the concurrent connection count expected on a data centre's north-south uplink. However, dedicated software packet processors have been proven to operate at edge line rates. For instance, Google's "Espresso" architecture relies on host-based software processing to handle global peering traffic (Yap et al., 2017), demonstrating that a single commodity server can process 37 Gbps with negligible CPU overhead.

7. Conclusion

In this work, we presented a design for cryptographically fingerprinting all traffic entering and leaving an AI data centre, with the goal of verifying that no undeclared workloads have been covertly exfiltrated from the facility. The core of our proposal is the SeGaDev: an Active Warden combined with a hashing pipeline, placed at the north-south network boundary of the cluster. By scrubbing steganographic degrees of freedom from protocol headers, re-timing packets to suppress analogue and timing side-channels, and hashing the sanitised traffic for later auditing, the device enables a cryptographic commitment to all egress data that the Prover can be later challenged to justify.

A key property of our design is that it removes the need for processors trusted by both parties. Instead, we achieve mutual verifiability via passive optical fibre splitters and coin-flip protocols for shared randomness. This grounds trust in the physics of beam splitting and optical isolation rather than in the integrity of any single device. Both the Prover and the Verifier can independently observe and check the device's behaviour without being able to interfere with each other's monitoring.

Our feasibility analysis suggests that the approach scales to clusters of 100k+ accelerators using commercially available hardware: FPGA-based network taps and hash processors, off-the-shelf optical components, and storage volumes that represent a negligible fraction of modern data centre costs. The individual building blocks — traffic scrubbing, passive optical tapping, hardware hashing — are already deployed in adversarial settings such as financial regulation and military networks.

For now, the core remaining challenge is that steganography in ML output payloads represents a significant residual covert channel, with a potential bandwidth of tens of Mbit/s for a fully utilised cluster. Closing this channel depends on progress in deterministic inference replay or active scrubbing of hardware-induced entropy in token likelihoods, both of which are active areas of research. Other residual channels — timing modulation through connection lifecycle events and information encoded in faked hardware failures — offer an adversarial Prover a much lower bandwidth. Additionally, the entire architecture rests on the assumption that the SeGaDev is the *only* communication path out of the facility, which requires robust physical monitoring that we do not address.

We encourage the community to pursue a hardware demonstration of the SeGaDev, which we expect to be inexpensive relative to the human engineering effort involved. Given the pace of AI capability development and the growing interest in international agreements, verification infrastructure of this kind may be needed sooner than the policy frameworks it is meant to support.

LLM Usage Statement

Large language models were used for web search, coding assistance, ideation and editorial advice, but not for writing. The authors are responsible for all factual claims.

References

A-Team Insight. A ticking clock: The tricky issue of timestamping for MiFID II. https://a-teaminsight.com/blog/a-ticking-clock-the-tricky-issue-of-timestamping-for-mifid-ii/, May 2018. Accessed: 2026-02-26.

Aarne, O., Fist, T., and Withers, C. Secure, governable chips: Using on-chip mechanisms to manage national security risks from AI and advanced computing. Technical report, Center for a New American Security (CNAS), January 2024. URL https://www.cnas.org/publications/reports/secure-governable-chips.

Anderson, R. J. and Petitcolas, F. A. P. On the limits of steganography. *IEEE Journal on Selected Areas in Communications*, 16(4):474–481, May 1998. doi: 10.1109/49.668971. Special Issue on Copyright and Privacy Protection.

Arista Networks. An overview of Arista Ethernet capture timestamps. Solution brief, Arista Networks, Inc., Santa Clara, CA, March 2019. URL https://www.arista.com/assets/data/pdf/Whitepapers/Overview_Arista_Timestamps.pdf. Accessed: 2026-02-26.

Arista Networks. Deutsche Börse group monitors every trade with Arista. Case study, Arista Networks, Inc., Santa Clara, CA, October 2020. URL https://www.arista.com/assets/data/pdf/CaseStudies/CaseStudy-DeutscheBoerse.pdf. Accessed: 2026-02-26.

Baker, M., Kulp, G., Marks, O., Brundage, M., and Heim, L. Verifying international agreements on AI: Six layers of verification for rules on large-scale AI development and deployment. *arXiv preprint arXiv:2507.15916*, 2025. URL https://arxiv.org/abs/2507.15916.

Bengio, Y. et al. International AI safety report 2026. https://internationalaisafetyreport.org/, February 2026.

Cankaya, N. Catching misreporting about ML hardware use by turning noise into signal. https://nacicankaya.substack.com/p/catching-misrepor

ting-about-ml-hardware, 2025. Technical blog post.

Cankaya, N. Catching misreporting about ML hardware use by turning noise into signal — part II, January 2026. Technical blog post.

Centre pour la Sécurité de l'IA (CeSIA) and The Future Society. Global call for AI red lines. https://red-lines.ai/, 2025.

Cyberspace Administration of China. Interim measures for the management of generative artificial intelligence services. https://www.chinalawtranslate.com/en/generative-ai-interim/, July 2023.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. Technical report, DeepSeek, April 2026. URL https://huggingface.co/deepseek-ai/DeepSeek-V4-Pro/blob/main/DeepSeek_V4.pdf.

DigiCert. RFC 3161 compliant time stamp authority server. https://knowledge.digicert.com/general-information/rfc3161-compliant-time-stamp-authority-server, 2023. DigiCert knowledge base article.

Douthat, R. JD Vance on his faith and Trump's most controversial policies. Interesting Times podcast, The New York Times, May 2025. URL https://www.nytimes.com/2025/05/21/opinion/jd-vance-pope-trump-immigration.html. Interview with Vice President JD Vance.

Dubey, A. et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, July 2024. URL https://arxiv.org/abs/2407.21783.

European Parliament and Council of the European Union. Regulation (EU) 2024/1689 of the european parliament and of the council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). https://eur-lex.europa.eu/eli/reg/2024/1689/oj, July 2024.

Fisk, G., Fisk, M., Papadopoulos, C., and Neil, J. Eliminating steganography in Internet traffic with active wardens. In Petitcolas, F. A. P. (ed.), *Information Hiding (IH 2002)*, volume 2578 of *Lecture Notes in Computer Science*, pp. 18–35, Berlin, Heidelberg, 2003. Springer. doi: 10.1007/3-540-36415-3_2.

Guri, M. Air-gap electromagnetic covert channel. *IEEE Transactions on Dependable and Secure Computing*, 21 (4):1–18, July 2024. doi: 10.1109/TDSC.2023.3300035.

Happel, J. TamperSec: Securing AI hardware against physical tampering. https://tampersec.com/, 2026.

Harack, B. et al. Verification for international AI governance. Technical report, Oxford Martin AI Governance Initiative, July 2025. URL https://aigi.ox.ac.uk/publications/verification-for-international-ai-governance/.

He, H. and Thinking Machines Lab. Defeating nondeterminism in LLM inference. Thinking Machines Lab: Connectionism, September 2025. URL https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/.

Ilhan, A., Withers, C., Gietz, H., and Harack, B. Verifiable semiconductor manufacturing. Research paper, Oxford Martin AI Governance Initiative, April 2026. URL https://aigi.ox.ac.uk/publications/verifiable-semiconductor-manufacturing/.

Jankowski, B., Mazurczyk, W., and Szczypiorski, K. PadSteg: Introducing inter-protocol steganography. *Telecommunication Systems*, 52(2):1101–1111, 2013. doi: 10.1007/s11235-011-9616-z.

Karvonen, A., Reuter, D., Rinberg, R., Marks, L., Garriga-Alonso, A., and Warr, K. DiFR: Inference verification despite nondeterminism. *arXiv preprint arXiv:2511.20621*, 2025. URL https://arxiv.org/abs/2511.20621.

Khattak, F. A. and Mikaitis, M. Accurate models of NVIDIA tensor cores. *arXiv preprint arXiv:2512.07004*, December 2025. URL https://arxiv.org/abs/2512.07004.

Lee, K. S., Wang, H., and Weatherspoon, H. PHY covert channels: Can you see the idles? In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pp. 173–185, Seattle, WA, 2014. USENIX Association. URL https://www.usenix.org/conference/nsdi14/technical-sessions/presentation/lee.

Microsoft. SSL/TLS termination overview with Application Gateway. https://learn.microsoft.com/en-us/azure/application-gateway/ssl-overview, 2024. Microsoft Learn documentation.

National Security Commission on Artificial Intelligence. Final report of the national security commission on artificial intelligence. Technical report, National Security Commission on Artificial Intelligence (NSCAI), March 2021. URL https://reports.nscai.gov/final-report/. Archived at https://cybercemetery.unt.edu/nscai/20211005220507/https://www.nscai.gov/2021-final-report/.

OpenAI. Chat completions streaming events. OpenAI API Reference, 2026. URL https://developers.openai.com/api/reference/resources/chat/subresources/completions/streaming-events. Accessed: 2026-04-27.

Park, J., Yoo, J., Yu, J., Lee, J., and Song, J. A survey on air-gap attacks: Fundamentals, transport means, attack scenarios and challenges. *Sensors*, 23(6):3215, March 2023. doi: 10.3390/s23063215.

Petrie, J. Guaranteeable memory: An HBM-based chiplet for verifiable AI workloads. In *ICML 2025 Workshop on Technical AI Governance (TAIG)*, June 2025. URL https://openreview.net/forum?id=uc79kOv0MV.

Petrie, J. and Aarne, O. Flexible hardware-enabled guarantees (FlexHEG): A three-part series. https://flexheg.com/, 2025.

Rinberg, R., Karvonen, A., Hoover, A., Reuter, D., and Warr, K. Verifying LLM inference to detect model weight exfiltration. *arXiv preprint arXiv:2511.02620*, 2025. URL https://arxiv.org/abs/2511.02620.

Sastry, G., Heim, L., Belfield, H., Anderljung, M., Brundage, M., Hazell, J., O'Keefe, C., Hadfield, G. K., Ngo, R., Pilz, K., Gor, G., Bluemke, E., Shoker, S., Egan, J., Trager, R. F., Avin, S., Weller, A., Bengio, Y., and Coyle, D. Computing power and the governance of artificial intelligence. *arXiv preprint arXiv:2402.08797*, February 2024. URL https://arxiv.org/abs/2402.08797.

Scher, A. and Thiergart, L. Mechanisms to verify international agreements about AI development. https://techgov.intelligence.org/research/mechanisms-to-verify-international-agreements-about-ai-development, 2024.

Simmons, G. J. The prisoners' problem and the subliminal channel. In Chaum, D. (ed.), *Advances in Cryptology: Proceedings of CRYPTO 83*, pp. 51–67, Boston, MA, 1984. Springer. doi: 10.1007/978-1-4684-4730-9_5.

Tlaie Boria, A. Securing deep frontier AI evaluations: A confidential-computing architecture for external model access. Technical report, Pour Demain, Brussels, Belgium, May 2026. URL https://www.pourdemain.ngo/en/post/securing-deep-frontier-ai-evaluations.

U.S. Commodity Futures Trading Commission. CFTC orders Panther Energy Trading LLC and its principal Michael J. Coscia to pay $2.8 million and bans them from trading for one year, for spoofing in numerous commodity futures contracts. https://www.cftc.gov/PressRoom/PressReleases/6649-13, July 2013. Accessed: 2026-02-26.

U.S. Securities and Exchange Commission. Rule 613 (Consolidated Audit Trail). https://www.sec.gov/about/divisions-offices/division-trading-markets/rule-613-consolidated-audit-trail, October 2017. Accessed: 2026-02-26.

Uttarwar, V. U. and Dakhane, D. M. Delay normalization technique to disrupt covert timing channels using active warden. *EPJ Web of Conferences*, 315:01049, 2025. doi: 10.1051/epjconf/202531501049.

Wiener, S. SB-53: Artificial intelligence models: Large developers (Transparency in Frontier Artificial Intelligence Act). https://leginfo.legislature.ca.gov/faces/billNavClient.xhtml?bill_id=202520260SB53, September 2025.

Xing, J., Kang, Q., and Chen, A. NetWarden: Mitigating network covert channels while preserving performance. In *29th USENIX Security Symposium (USENIX Security 20)*, pp. 2039–2056. USENIX Association, 2020. URL https://www.usenix.org/conference/usenixsecurity20/presentation/xing.

Yap, K.-K., Motiwala, M., Rahe, J., Padgett, S., Holliman, M., Baldus, G., Hines, M., Kim, T., Narayanan, A., Jain, A., Lin, V., Rice, C., Rogan, B., Singh, A., Tanaka, B., Verma, M., Sood, P., Tariq, M., Tierney, M., Trumic, D., Valancius, V., Ying, C., Kallahalla, M., Koley, B., and Vahdat, A. Taking the edge off with Espresso: Scale, reliability and programmability for global Internet peering. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '17)*. ACM, 2017. doi: 10.1145/3098822.3098854.

A. Prover's Verification Mechanism

While the exact detail of the verification cluster, as well as the verification pipeline itself, are out of scope for this paper, we now briefly describe one possible workflow. The fundamental challenge is as follows: given a hash value chosen by the Verifier out of the stored pool of hashes, how can the Prover reliably and efficiently demonstrate that they possess a record which hashes to this value?[19]

A simple idea is to maintain a traditional 'hash table' — a dictionary of all plaintext LLM inputs, the corresponding plaintext outputs and their hash values. Since the Prover already controls the egress from the data centre, they can easily register how many data packets (e.g., Ethernet frames) each output is broken into. Furthermore, since the SeGaDev specification would be fully open-source, they also know the 'hashing boundary' — how many frames are included in one hash. This allows the Prover to have a causal association between inputs, outputs and the corresponding sequence of their hashes. The hash values flow back to the Prover through a passive network tap placed on the connection which carries them from the SeGaDev to the verification cluster (see Figure 3).

The number of Ethernet frames within each hashing boundary corresponds to the fundamental 'verification unit' and is arbitrary. It should strike a balance between storage requirements and recomputation time. For example, an extreme choice would be to hash every single Ethernet frame, leading to the highest possible number of hashes in storage. Since in this scenario one hash corresponds to a very small number of output tokens, this minimises the number of outputs that need to be recomputed in the verification facility. On the other hand, we could hash together all Ethernet frames that pass the SeGaDev in a month, which would lead to only one hash per month in storage, but would require one month of recomputation during verification.

The Prover needs to maintain knowledge of which inputs and parts of the corresponding outputs were included in each hash. For example, assume that the hashing window is 150 Ethernet frames, two input prompts generate 120 and 70 tokens, respectively, and one output token is streamed out via a single Ethernet frame. The Prover must then associate the following information with the hash values h_1 and h_2:

```
{
    h1: [(prompt_1, output_1, 1, 120),
         (prompt_2, output_2, 1, 30)],
    h2: [(prompt_2, output_2, 31, 70), ...],
    ...
}
```

In this way, when the Verifier demands that hash h_1 be reproduced, the Prover can immediately look up the fact that prompts 1 and 2 must be sent to the verification facility (alongside the corresponding outputs and their indices).[20] This information is necessary for the facility to faithfully reproduce the exact stream of 150 Ethernet frames that should be hashed and compared to h_1. If this new hash matches h_1, the Verifier is satisfied that the Prover's cluster must have originally carried out the same workload. If this verification pipeline is employed enough times, the Verifier can gain confidence that the Prover did not execute hidden workloads — because they would not have had the time and free computational resources in their cluster to execute such workloads in addition to the ones that were verified post hoc.

B. Storage Requirements

To demonstrate the feasibility of our proposed verification pipeline, we now estimate the upper bound of the volume of hashes and plaintext data that would need to be stored by the Verifier and the Prover respectively. First, let us assume that the data centre under monitoring contains 100k GPUs of the NVIDIA Hopper generation. Each such GPU is assumed to be capable of a 2000 token/s inference throughput, meaning that the total throughput of the cluster is 2×10^8 tokens/s.[21] Next, we need to calculate how many bytes of data this token volume corresponds to. For this, we assume that the tokens are

[19] As a reminder, in this work we do not consider the problem of verifying that declared workloads are compliant with regulations, only the problem of verifying that no other undeclared workloads have been run.

[20] Note that in general, these could be batched prompts or some other type of input. We used single prompts for simplicity in this illustrative example.

[21] Deepseek's inference setup for their v3/R1 models is reported to achieve a token throughput of $\sim$14.8k tokens per H800 node (8 GPUs, decode unit) per second.

wrapped in OpenAI's streaming API template of the following format:

```
{
  "id": "chatcmpl-123",
  "object": "chat.completion.chunk",
  "created": 1694268190,
  "model": "gpt-4o-mini",
  "system_fingerprint": "fp_44709d6fcb",
  "choices": [
    {
      "index": 0,
      "delta": {
        "content": "Hello"
      },
      "logprobs": null,
      "finish_reason": null
    }
  ]
}
```

Importantly, note that this template includes a single response token only. This is because the streaming API aims to deliver generated tokens to the user as quickly as possible, rather than waiting for the full generation to finish before sending the payload. The 'one-token-per-json' scenario represents the most challenging configuration for our methodology, as this will lead to the highest possible number of hash computations.

Furthermore, we operate under an (unrealistic) assumption that each such template is allocated to one Ethernet frame. Therefore, the total number of Ethernet frames streamed out of the cluster is simply 2×10^8 per second, the same as number tokens. The template carries around 200 ASCII characters, corresponding to 200 bytes (+ headers) — well below the Maximum Transmission Unit (MTU) of the Ethernet protocol (1500 bytes). We then calculate the SHA-256 hash of each such frame, leading to $2 \times 10^8 \times 24 \times 3600 \times 365 = 6.3 \times 10^{15}$ hash computations performed in a year. The corresponding hash volume is $6.3 \times 10^{15} \times 256/8 = 2.0 \times 10^{17}$ bytes $= 200$ PB.

Additionally, the Prover needs to maintain a year-long record of all plaintext data transmitted from their cluster to the outside world. Each generated token is assumed to be wrapped in the template above. Since one ASCII character is stored by exactly 1 byte, 200 bytes are required to stream one token out of the cluster.[22] On an annual scale, this corresponds to $2 \times 10^8 \times 200 \times 24 \times 3600 \times 365 = 1.26 \times 10^{18}$ bytes $\sim 1{,}260$ PB. Note that we do not separately estimate the storage requirements for input data (prompts), as they are negligible in comparison.[23]

The cost of storing such a large amount of data is small ($< 1\%$) compared to the costs of AI data centres at the scale considered here: a typical price for a 28 TB HDD hard drive is around 500€, thus the full cost amounts to $1{,}260$ PB$/28$ TB $\times$ 500€ $\sim$ 22,500,000 €.[24]

Note that all these estimates represent a pessimistic upper bound that arises from *extremely* inefficient data management. When using non-streaming APIs, multiple tokens are included as part of one chat completion template. Further, multiple such templates could be transmitted as part of a single Ethernet frame, as long as the total payload size does not exceed the standard Ethernet MTU of 1500 bytes. Storing API JSON files for each token is also likely unnecessary when they can be reconstructed from tokens and heuristics. We also did not include data compression at all. Finally, we do not need to compute the hash of each Ethernet frame. Instead, the 'fundamental unit' of verification could be an arbitrarily long chain of frames. All these modifications could reduce storage by multiple orders of magnitude.

[22]The Prover must store the complete streaming API response templates rather than the bare output tokens. This is because the hashes computed by the SeGaDev are taken over the Ethernet frames as transmitted, which include the full JSON payload with all metadata fields.

[23]While input prompts are typically longer than individual output tokens (especially in multi-turn conversations), they are stored as single objects rather than in per-token JSON templates. Even under a pessimistic assumption that input prompts are 20x longer than outputs in terms of the raw token count, the per-token storage overhead from the streaming format dominates by a factor of around $200/20 = 10$.

[24]As of early 2026.

C. Preventing Dictionary Attacks

The Prover might be worried that the Verifier can deduce the plaintext content of the messages passing through the SeGaDev by guessing which input hashes to the observed hash values. Deducing this content by inverting the hash is not possible — hash functions are by design irreversible. However, an adversarial Verifier could be tempted to use so-called dictionary attacks. In other words, they could pre-compute the hashes of the most common plausible inputs and simply compare them to the observed ones. The feasibility of this approach is determined by the expected size of the input space of the hash function. For example, if there can only ever be 10 possible inputs, then it is trivial to pre-compute their 10 corresponding hashes and later infer the input that has produced the hash coming out of the SeGaDev.[25]

Unfortunately for the Prover, the input space of their API responses is indeed somewhat limited: these can only be tokens generated by the LLM. These in turn are constrained by the size of the LLM vocabulary, typically around 200,000 tokens. Thus, in the scenario presented above — where only one output token is inserted into the streaming API template, this template corresponds to one Ethernet frame and later one hash — the attacker could pre-compute the hashes of all tokens inserted into these templates. They would then simply perform the lookup in a hash table with 200,000 values and immediately infer the right token. Thus, it would be trivial for the Verifier to guess all the output tokens from the streaming API responses.

To calculate how many tokens need to be inserted into the output template to prevent such attacks via combinatorial complexity, let us consider how many hash computations a motivated attacker could perform in a month. Specialised chips can achieve throughputs of 200 TH/s (terahashes per second). We use this Bitcoin-mining figure as a worst-case upper bound. Against the SeGaDev's longer inputs, an attacker would need a custom ASIC and would likely achieve 1–2 orders of magnitude lower throughput in practice, but we conservatively assume the Bitcoin rate. Within a month, this corresponds to 5.2×10^{20} hashes computed. On the other hand, a sequence of n tokens out of a 200k vocabulary can lead to $(2 \times 10^5)^n$ distinct combinations. Thus, the attacker is guaranteed to guess the sequence corresponding to a given hash if the sequence is shorter than $\log_{2\times10^5}(5.2 \times 10^{20}) \sim 3.9$ tokens long. Running multiple specialised chips in parallel will not help the attacker much — with 1000 chips, the token number increases to 4.5.

To address this, we recommend that the streaming API template should enclose at least 5 tokens in order to prevent brute-force hash guessing by the Verifier. However, this estimate assumes that the attacker needs to try all possible n-gram combinations of tokens from the vocabulary. This is an unrealistic assumption, as the tokens streamed out of the cluster will almost always be semantically meaningful — sequences of tokens that form sentences, code snippets, equations, or similar. Thus, the search space is greatly reduced by noting that only a small fraction of the vocabulary is likely to follow the previous token. Assuming that only 1000 most likely tokens need to be checked, our 'effective vocabulary' is reduced to 1000. Then, the calculation becomes $\log_{1000}(5.2 \times 10^{20}) \sim 7$.

In other words, any semantically meaningful sequence of fewer than 7 tokens could be brute-forced by the Verifier in less than a month with a single hash-crunching chip.[26] Thus, to prevent this, the streaming API template could wrap at least 10 tokens. This should have a negligible effect on user experience — instead of seeing tokens appear on their screen one at a time, the user will receive them 10 at a time. Alternatively (or additionally), the hashes could simply be computed on more than one JSON template or more than one Ethernet frame. For example, if the hashing boundary is set every 100 Ethernet frames, the whole problem disappears even with 1 streaming token per JSON template.

D. Precedents, Available Hardware and Feasibility

While our proposal is conceptual for now, we emphasize that all of the individual components for the SeGaDev are either established and field-tested (passive optical fibre splitters, coin flip protocols, hash functions with timestamps (DigiCert, 2023)) or built from quickly reconfigurable commercial hardware.

[25] Note that this might not actually be an issue at all, since in our method, the hashed content also includes metadata such as timestamps of the API response. If the adversarial Verifier does not know these, then dictionary attacks are not possible in the first place. However, we do see this threat vector as important, as in the future AI outputs might be wrapped in standardized templates that do not include unpredictable metadata entries. This could be, for example, to reduce storage requirements on Prover's side (see Appendix B). In such cases, dictionary attacks become relevant.

[26] Once again, we note that this discussion considers worst-case assumptions. In reality, an adversarial Verifier would probably need to somehow know in advance which hashes to attack — presumably, only specific AI outputs would be of interest to them (for example, those carrying sensitive intellectual property), not any randomly selected sequences of outputs. Additionally, a month-long attack period is something that only the most motivated of attackers are likely to attempt.

For information processing, Field-Programmable Gate Arrays (FPGAs) are a suitable choice for two reasons. First, they can be adapted to any specialised application after manufacturing. Second, they are inherently more secure against supply chain attacks, since an attacker cannot anticipate the circuit design in advance.[27] For these reasons, FPGAs find use in a range of adversarial, high-stakes environments, most notably military networks (Infodas SDoT Secure Network Card, Infodas SDoT Security Gateway), high-frequency trading, and financial law enforcement. FPGA network cards and network taps are commercially available from multiple vendors (e.g. BittWare IA-780i, Arista 7130 Series). These systems are deployed at scale, operate at line rate (10–400 Gb/s), introduce negligible latency, and are relied upon for regulatory compliance where evidence integrity is legally contested (e.g., CAT NMS / Reg NMS (U.S. Securities and Exchange Commission, 2017), MiFID II (A-Team Insight, 2018)).[28]

Traffic scrubbing (Xing et al., 2020), traffic normalization, and re-timing devices (Uttarwar & Dakhane, 2025) against timing and analogue side channels have all been built and tested at line rates exceeding 100 Gb/s.

We now give a brief feasibility assessment from the bandwidth and storage perspective. In Appendix B, we estimate that a 100k GPU AI cluster streaming 200M tokens per second corresponds to an egress API bandwidth of ∼40 GB/s. The SeGaDev, even under such extreme assumptions, can use off-the-shelf hardware. Modern hashing algorithms are efficiently parallelizable on mid-range FPGAs.[29] The storage requirements for both the Prover and the Verifier are insignificant for any realistic data management solution (see Appendix B).

Inference clusters, network taps and verification clusters can be quickly demonstrated on small scales, while still being almost fully representative of large-scale deployments: a single server or rack to simulate the Prover's and Verifier's clusters respectively. Scaling the system becomes a matter of adding more of the same hardware. At the scale of real-world deployments, the main technical challenge then shifts to physical security and red-teaming.

For the sub-scale demonstration, we expect the bill of materials to be insignificant compared to the cost of the engineering and security red-teaming staff required for a pilot demonstration.

E. Eliminating Steganography Needs Connection Tables

Preventing steganographic communication between the data centre and the outside world introduces the need to eliminate certain degrees of freedom from the protocol metadata. To illustrate this more concretely with an example, in this Appendix we justify why source and destination ports need to be explicitly handled by the Active Warden.

Network protocols grant both parties control over certain fields. In UDP, the initiating party (client) selects its source port, whilst the responding party (server) must use the destination port specified in the request as its source port for the response. Without intervention, either party could encode information by selecting these ports strategically.

Consider a simple example: an external client sending API requests to a data centre. By choosing source port 5000 versus 5001, the client transmits one bit of information per connection. At 1000 requests per second, this yields 125 Bps or ∼10 MB/day of covert channel capacity. Similarly, when the data centre initiates outbound connections (e.g., fetching training data), it could encode information in its choice of the ephemeral source port (typically ranging from 32768 to 60999, providing $\log_2(28232) \sim 15$ bits per connection). The Active Warden must eliminate both channels.

Two separate connection-tracking tables are required, because connections can be initiated from either direction and the security logic differs in each case:

- **External-initiated connections** (e.g. API requests to the data centre) allow an external client to encode information in their choice of the client source port. The Warden must replace this with a random value, then maintain a state table to translate responses back to the original port.

[27] With some limited exceptions. For example, there are fixed I/O locations.

[28] A specific example is Metamako/Arista Networks, whose 7130 FPGA taps are used for precision timestamping (Arista Networks, 2019). The German stock exchange (Deutsche Börse) publicly disclosed deployment of dozens of such devices in a case study (Arista Networks, 2020). Their threat model in trading networks is explicit: traders attempt to hide behaviour (U.S. Commodity Futures Trading Commission, 2013); regulators require complete, lossless, timestamped records (Arista Networks, 2019).

[29] The SHA-3 256 hashing function is a natural choice for hardware hashing at high bandwidth, achieving ∼34 Gbps per core in ∼1,375 slices (∼5,500 lookup tables, aka. LUTs) on a Virtex-7 FPGA. Twelve parallel cores sustain ∼410 Gbps hashing in ∼66,000 LUTs, roughly 13% of a commodity Kintex UltraScale+ KU15P (522,720 LUTs, ∼$4,000 or lower at volume).

- **data centre-initiated connections** (e.g. fetching external datasets) allow the data centre operator to encode information in their choice of ephemeral source port. Again, the Warden must replace this with a random value and maintain a reverse translation state.

Maintaining separate tables for each direction simplifies the translation logic, as the scrubbed field and the party being defended differ between the two cases. Both tables have keys characterised by the full connection 4-tuple: (source IP, source port, destination IP, destination port).

Each state-tracking table has two distinct security functions during the connection lifecycle. First, during the request phase, the table performs port randomisation: the initiating party's source port (client port) is replaced with a value from the RNG, eliminating their ability to encode information. Simultaneously, the table records the target server port from the request's destination field. During the response phase, the table performs enforcement and back-translation: it verifies the responding party uses the correct server source port (which prevents the server from encoding information through port choice), and it translates the response destination port from the RNG value back to the original client port. Overall, this ensures that the response is delivered to the correct client port and does not carry information steganographically.

Similar translation tables need to be maintained wherever parts of the relevant network protocol offer degrees of freedom that could be used for covert communication. This could introduce significant programmatic complexity for more complicated protocols like TCP, which are stateful. In TCP, the Active Warden must additionally perform sequence number translation on every single packet, and in both directions. More precisely, during the so-called 'TCP handshake', the client's Initial Sequence Number (ISN) is replaced with a random number. Thus, the Warden must record the offset between the original and scrubbed ISN, so that all subsequent packets in this connection have their outbound sequence numbers incremented by this consistent offset. Similarly, the corresponding inbound ACK numbers in this connection need to be decremented by the same offset. This is necessary to maintain consistency in both the original and scrubbed sequences.

Overall, the Active Warden must track connection state to handle the handshake phase, data transfer phase, and connection termination. Each TCP connection might require around 200 bytes of state-tracking data. For a data centre with $O(1000)$ concurrent connections processing millions of packets per second, the Active Warden must implement efficient table lookups to avoid slowdowns in serving this traffic to customers.

F. Glossary

For convenience, this appendix collects acronyms and terms used in the paper.

Acronyms

FBT Fused Biconical Taper. An optical splitter formed by fusing two fibres together and tapering them so that light couples between them.

FPGA Field-Programmable Gate Array. A reconfigurable digital logic chip whose internal connections are programmed after manufacture.

FSSL Fixed-Seed Sampling Likelihood (Rinberg et al., 2025). Given a sampling seed and a claimed token, the probability the token could have been produced honestly under benign numerical noise.

HBM High Bandwidth Memory. 3D-stacked DRAM placed adjacent to a processor die for very high memory bandwidth.

ISN Initial Sequence Number. The 32-bit starting sequence number negotiated during a TCP handshake.

LUT Look-Up Table. The basic logic primitive of an FPGA; implements arbitrary Boolean functions of a few inputs.

MTU Maximum Transmission Unit. The largest payload size that fits in a single frame of a given link layer (1500 B for standard Ethernet).

mTLS Mutual TLS. A TLS handshake in which both client and server authenticate to each other with certificates.

NIC Network Interface Card. The hardware connecting a host to a network.

OSI Open Systems Interconnection. The seven-layer reference model used to describe networking functionality, from the physical layer up to the application layer.

PHY Physical Layer. Layer 1 of the OSI model; the electrical, optical, or RF signaling on the wire.

PLC Planar Lightwave Circuit. A silica-on-silicon optical waveguide chip used as a splitter or combiner.

RNG Random Number Generator.

SRAM Static Random-Access Memory. Fast on-chip memory that holds state without periodic refresh.

Terms

Active Warden A network element that actively rewrites traffic (rather than only inspecting it) to remove covert channels (Anderson & Petitcolas, 1998).

Coin flip protocol A commit-reveal protocol in which both parties commit to a value, both reveal, and the joint output (e.g., bitwise XOR) is used as shared randomness; neither party can bias the result unilaterally.

Hashing boundary The unit of network traffic over which a single cryptographic commitment is computed by the SeGaDev (e.g. one packet, a packet group, or a time window).

Prover / Verifier The two roles in the verification protocol. The Prover is the operator of the AI compute facility being inspected; the Verifier is the external party checking compliance.

Re-timing Buffering packets and releasing them on a fixed schedule to remove timing-based covert channels.

SeGaDev Secure Gateway Device. The verification device proposed in this paper.