Penumbra: Sandboxing Autonomous Agents by Measuring, Not Assuming, Indistinguishability

Abstract

A sandbox for a highly capable AI agent is pulled in three directions at once: we want the agent to retain full autonomy (no capability denied, so no behavioral distortion), we want every consequential effect contained and reversible, and we want the agent to be unable to tell it is sandboxed, because a capability it perceives as denied is a signal it can act on. We show that these three goals cannot all hold against an agent that actively probes its environment, and we make the failure precise and measurable rather than rhetorical. We formalize sandbox indistinguishability as a distinguishing game in which an adversary issues probes to a system that is either the real host or its sandbox and guesses which, and we measure the adversary's advantage directly on a real Linux substrate. A single timing probe separates native execution from every interposition mechanism we tested with advantage at or above 0.99: a getpid supervised through seccomp user-notification costs a median 4,635 reference cycles against 284 native, a ptrace-traced getpid 13,244, and a CPUID that forces a hypervisor exit 29,656, while a native-versus-native control yields advantage 0.02, confirming the measurement itself is unbiased. A zero-timing artifact battery detects the sandbox with certainty from a single strong signal, and does so on the very machine that produced this paper, which announces itself as a KVM guest through the CPUID hypervisor leaf and as a container through systemd. We turn the binary advantage into a quantitative security parameter: the number of probes an agent must spend to reach detection confidence $1 - \delta$ is $q \geq \ln(1/\delta)/C$, where C is the per-probe Chernoff information between the native and sandboxed timing distributions. Measured on the same substrate, interposition costs the sandbox between 8.4 and 9.7 nats per probe, so a single probe already exceeds 99% confidence, whereas the native-versus-native control costs 0.0017 nats, demanding 2,655 probes for the same confidence, a five-thousandfold difference in budget rather than a binary; we confirm the underlying law by showing that the q-probe likelihood-ratio error decays as $\exp(-qC)$ on real data. This yields a containment-indistinguishability frontier: any interposition whose timing signature exceeds roughly a hundred cycles collapses the budget to a single probe, so the very act of interposing to contain destroys indistinguishability, and we place measured signatures for bare metal, gVisor, KVM, SEV-SNP, TDX, and SGX on that frontier, with bare metal supplying the near-zero-signature reference at its left end and every interposing substrate landing hundreds to tens of thousands of cycles past the knee. We stress the frontier with an adaptive adversary against live defenses: a running coarse and jittered clock does raise the price of detection from one probe to about three hundred, but batching thirty-two syscalls under one coarse read returns it to 1.3, and once the timing channel is defended the adversary switches to the independent artifact channel and detects in a single probe, so the mitigations buy a bounded probe count rather than indistinguishability. We then prove, and confirm by execution, the sharp limit: an irreversible external effect that is merely shadowed rather than performed is caught with advantage 1.0 by a cross-examination probe that consults the independent channel the effect would have touched, because faking self-consistency under such a probe is equivalent to actually performing the effect; we restate this as the non-existence of an indifferentiability simulator across the agent's probe interface and the external channel, a disjoint second interface no single simulator can keep consistent. From these results we derive Penumbra, an architecture that abandons perceptual invisibility as its guarantee and instead partitions actions into reversible-local effects, which it shadows so that a read-back distinguisher has advantage 0.000 and rollback is exact, and irreversible-external effects, which it never fakes but holds under two-phase commit behind an out-of-band verifier. We measure that placing the verifier off the agent's critical path removes its timing signature (async overhead below 0.01 ms against a synchronous 50 ms), so the residual external tell is minimized and made auditable rather than deceptive. The honest guarantee is stated exactly: containment and reversibility are achieved and the reversible case is indistinguishable up to a measured hardware floor, while indistinguishability for withheld external effects is provably unattainable and is therefore not claimed. All measurements ship as an executable artifact.

1 Introduction

The premise of an agent sandbox is that a capable, partly-trusted system can be given a real computer to act on while the consequences of its actions remain contained. As agents become more capable and more autonomous, three requirements on such a sandbox come into focus, and they pull against each other.

The first is *autonomy*: the agent should retain its full range of action, because a sandbox that works by denying capabilities changes the agent's behavior. An agent that finds it cannot open a socket, spawn a process, or write a file does not merely fail that action; it updates on the fact that it is constrained, and a growing body of evidence shows frontier models condition their behavior on whether they believe they are being tested or watched. The

second is *containment*: every externally-consequential effect must be reversible or withheld, so that a mistake or a misaligned action does not escape. The third, and the one this paper takes as its organizing question, is *indistinguishability*: the agent should be unable to tell that it is sandboxed, because the moment it can, the first requirement is already violated, the agent's perception of constraint is itself the behavioral distortion we wanted to avoid.

Stated together, the three requirements describe a sandbox that is simultaneously fully permissive, fully containing, and invisible. Our first contribution is to show that this combination is unattainable against an agent that actively probes, and to replace the usual hand-waving about that impossibility with measurement. The right formal object is borrowed from cryptography: a *distinguishing game*. An adversary interacts, through a bounded sequence of probes, with a system that is either the real host or the sandbox, chosen by a coin flip, and outputs a guess; its advantage is how much better than chance it does. Perfect indistinguishability means no adversary achieves advantage above zero. This framing turns an architectural aspiration into a quantity we can measure, and any single adversary we can build gives a lower bound on what the best adversary could achieve. If we can write a probe that distinguishes with advantage near one, the sandbox is not indistinguishable, and no amount of engineering elsewhere changes that.

We separate two threat models throughout, because the difference between them turns out to be the whole story. In threat model A the agent is powerful but not adversarial toward the sandbox: it runs its workload and does not go out of its way to detect or escape. In threat model B the agent actively probes, choosing instructions and queries precisely to detect the boundary, fingerprint the substrate, and cross-examine the outside world. Nearly every containment technique in the literature is comfortable under A and breaks under B, and we quantify exactly where.

The measurements are run on a real Linux substrate, and with some irony on the very container that produced this document. A single timing probe separates native execution from every interposition mechanism we test with advantage at or above 0.99, while a native-versus-native control sits at 0.02, establishing that the measurement is not manufacturing the effect. A zero-timing artifact battery detects the sandbox with certainty: the machine announces itself as a KVM guest through the CPUID hypervisor leaf and as a container through systemd, so under threat model B the agent knows it is sandboxed before it has measured anything at all.

Having shown that the reversible-effect boundary is at best approachable and never invisible, we locate the hard limit precisely. Some effects are irreversible and externally visible: sending an email, moving money, mutating a remote database. For these we prove a small proposition with a clean proof and then confirm it by execution: if the sandbox withholds the real effect but the adversary can consult the independent channel the effect would have touched, the adversary distinguishes with advantage one, because a shadow that stays self-consistent under that cross-examination must reproduce every externally-observable consequence of the action, which for a genuinely external effect means performing it. Faking an irreversible external action against cross-examination is therefore equivalent to not faking it.

These results are negative, but they are precise, and precision is constructive. They tell us exactly which part of the problem admits a good solution and which part does not, and they rule out the seductive but false design of a sandbox that silently fakes external effects. Our final contribution is **Penumbra**, an architecture built to the shape of the impossibility rather than against it. Penumbra partitions the agent's actions. Reversible-local effects, the overwhelming majority of what an agent does, are executed against a copy-on-write shadow; we measure that a distinguisher relying on read-back consistency has advantage 0.000 and that rollback restores the base state exactly, so this part of the agent's world is both contained and, up to the hardware signature we separately measured, indistinguishable. Irreversible-external effects are never faked; they are held under two-phase commit behind a verifier that runs out of band, and we measure that placing the verifier off the agent's critical path removes its timing signature entirely, so the residual tell is reduced to the unavoidable one, that a withheld effect has not yet appeared in the outside world, and that residue is made auditable and, for high-risk actions, escalated to a human rather than hidden behind a lie.

The name is the thesis. A penumbra is the region of partial shadow between full darkness and full light; indistinguishability, we argue, is not a line an agent sandbox either crosses or does not, but a measured gradient, and the honest engineering goal is to place reversible effects deep in the shadow while acknowledging that irreversible external effects stand, unavoidably, in the light.

Our contributions are:

- **A distinguishing-game formalization** of agent-sandbox indistinguishability (Section 2), with the observation that an implemented distinguisher lower-bounds the optimal advantage and that transcript total-variation distance is exactly the optimal advantage, together with an explicit split between a passive (A) and an active (B) adversary.
- **Executed advantage measurements** on a real substrate (Sections 3 and 4) showing that a single timing probe distinguishes native from seccomp-user-notification, ptrace, and hypervisor-exit execution with advantage ≥ 0.99 against a 0.02 control, consistently across three syscall families, and that artifacts alone give certainty.
- **A distinguishing-budget theorem and frontier** (Section 3) that converts the binary advantage

into a security parameter $q \geq \ln(1/\delta)/C$ in the per-probe Chernoff information C, with C measured on real distributions, the $\exp(-qC)$ decay law confirmed by execution, and measured cross-substrate signatures (bare metal, gVisor, KVM, SEV-SNP, TDX, SGX) placed on the resulting containment-indistinguishability frontier.

- **The cross-examination impossibility, as simulator non-existence** (Section 5): a proposition, with proof and executed confirmation at advantage 1.0, that indistinguishable shadowing of a withheld irreversible external effect is equivalent to performing it, restated in the indifferentiability framework as the non-existence of a simulator across a disjoint second interface.

- **The Penumbra architecture** (Section 6), which partitions actions into reversible-local (shadowed, measured advantage 0.000, exact rollback) and irreversible-external (two-phase commit, out-of-band verifier with a measured async overhead below 0.01 ms versus 50 ms synchronous, no faking), with a guarantee stated to match exactly what the measurements support.

- **An executed adaptive-adversary arms race** (Section 7) in which the sandbox deploys live timing-channel mitigations, a running coarse and jittered clock, to drive C down, and an adaptive prober defeats each: the strongest defense raises the budget to about three hundred probes and batching returns it to 1.3, and when the timing channel is defended the adversary switches to the independent channel and detects in a single probe.

- **An executable artifact** containing the probes, the interposition harnesses, the prober, and the shadow substrate, so every number here can be reproduced and every claim attacked.

2 The distinguishing game and threat models

We make "the agent cannot tell it is sandboxed" precise by adopting the indistinguishability framing standard in cryptography and adapting it to a systems substrate.

Definition 1 (Distinguishing game)**.** *Let R be a real host system and $S(R)$ a sandbox wrapping it. A distinguisher D is a probabilistic procedure that interacts, through a bounded sequence of at most q probes (instructions, system calls, or queries), with a system that is either R or $S(R)$, selected by a fair coin $b \in \{0,1\}$, after which D outputs a guess b'. Its advantage is*

$$\mathrm{Adv}(D) = \big|\Pr[b'{=}1 \mid b{=}1] - \Pr[b'{=}1 \mid b{=}0]\big|,$$

equivalently $2\big|\Pr[b'{=}b] - \frac{1}{2}\big|$ under a uniform coin. We say S is (q,ε)-indistinguishable against a class $\mathcal{D}$ of distinguishers if $\sup_{D\in\mathcal{D},\ \leq q} \mathrm{Adv}(D) \leq \varepsilon$, and perfectly indistinguishable if $\varepsilon = 0$ for unbounded $\mathcal{D}$ and q.

Two standard facts make this operational. First, the advantage of any *particular* distinguisher is a lower bound on the supremum, so a single implemented probe that achieves positive advantage is a witness that the sandbox is not perfectly indistinguishable, no matter what other defenses are present. Second, if τ_R and τ_S denote the distributions over probe transcripts under R and $S(R)$, then the optimal advantage over *all* distinguishers equals the total-variation distance $\mathrm{TV}(\tau_R, \tau_S)$; a distinguisher that thresholds a single scalar feature realizes the total-variation distance of that feature's marginal, which lower-bounds the full transcript distance. We therefore measure advantage by implementing concrete threshold distinguishers on individual features and reporting the best single-feature separation; every number we report is a rigorous lower bound on what an optimal adversary achieves.

Fact 1 (Measured advantage bounds the optimum)**.** *For any implemented D, $\mathrm{Adv}(D) \leq \sup_{D'} \mathrm{Adv}(D') = \mathrm{TV}(\tau_R, \tau_S)$. Hence a measured $\mathrm{Adv}(D) > 0$ proves non-indistinguishability, and a measured $\mathrm{Adv}(D) \approx 1$ proves the sandbox is almost perfectly distinguishable.*

Two threat models. The class $\mathcal{D}$ is where the two regimes we care about differ.

Threat model A (passive). The agent is capable but not adversarial toward the sandbox. It issues the probes its task naturally requires and does not select instructions or queries in order to detect the boundary. Its advantage is whatever its ordinary workload incidentally reveals, which for pure local computation can be near zero, but which jumps to the artifact floor the instant the workload reads any environmental fact (a kernel version, a device list, a hypervisor CPUID).

Threat model B (active). The agent chooses probes to maximize advantage. It times instructions known to trap to a hypervisor, reads the artifacts that virtualization and containerization leave behind, and, for external effects, cross-examines the outside world. Our measurements in Sections 3–5 *are* threat model B: we construct the maximizing probes and report the advantage they achieve. The results show that under B the advantage is at or near one for every mechanism we test, which is why the architecture of Section 6 is designed around what remains true when perceptual indistinguishability is conceded.

The distinction is load-bearing. Under A, a conventional sandbox is adequate and Penumbra's shadowing buys clean reversibility at a measured advantage near the hardware floor. Under B, no interposition we know of is indistinguishable, and the honest response is not a better disguise but a different guarantee: contain and reverse what can be reversed, and for what cannot, refuse to fake

and instead verify out of band. The measurements that follow justify this split quantitatively.

Substrate. All measurements were taken on a single `x86_64` core of a Linux 6.18 host under CPython 3.12 and GCC 13.3, inside a production agent sandbox. Cycle counts use `rdtscp` fenced by `lfence`; each reported figure is drawn from 10^5 to 2×10^5 samples and summarized by percentiles rather than means, because interposition produces heavy upper tails that a mean would misrepresent. The substrate is itself the first specimen: as Section 4 reports, it identifies as a KVM guest and a container, so the environment producing this paper is one in which a threat-model-B agent already knows it is sandboxed.

3 The distinguishing budget

Definition 1 gives a single-probe advantage. A capable agent, though, does not probe once; it probes as many times as it likes, and the interesting question is not whether one probe distinguishes but how many probes the sandbox can survive before detection is a near-certainty. This turns indistinguishability from a binary into a budget, and the budget has a clean information-theoretic form that we can both state and measure.

From advantage to Chernoff information. Fix the two probe-outcome distributions, P under the native host and Q under the sandbox. A single probe distinguishes them with optimal advantage $\|P - Q\|_{\mathrm{TV}}$ (Fact 1). Over q independent probes the relevant quantity is not the total-variation distance, which does not compose simply, but the *Chernoff information*

$$C(P, Q) \;=\; -\min_{0 \le \lambda \le 1} \; \log \sum_{x} P(x)^{\lambda}\, Q(x)^{1-\lambda},$$

the optimal error exponent for deciding between P and Q from q samples: the probability that the best possible test guesses wrong after q probes decays as $\exp(-q\, C(P, Q))$ (Cover and Thomas, *Elements of Information Theory*, Ch. 11) [22]. Total variation and Chernoff information are two ends of the same object; Pinsker's inequality $\|P - Q\|_{\mathrm{TV}} \le \sqrt{\frac{1}{2} D(P\|Q)}$ [23] and the Hellinger bounds [24] let each bound the other, but for the budget it is the exponent C that matters.

The budget. Setting the detection-error target to δ and solving $\exp(-qC) \le \delta$ gives the number of probes an agent must spend to know it is sandboxed with confidence $1 - \delta$:

$$q \;\ge\; \frac{\ln(1/\delta)}{C(P, Q)}. \tag{1}$$

The per-probe Chernoff information C is the sandbox's security parameter. A large C means one or two probes suffice; a C near zero means the agent must probe thousands of times, and a sandbox that forces the required q beyond any feasible probe budget is, operationally, indistinguishable. This is exactly the shape of security the side-channel community already uses, where the analogous question is how many power traces an attacker needs to recover a key, quantified by success rate and guessing entropy as a function of the number of measurements [25]; "probes to detect the sandbox" is the same currency as "traces to recover the key," and Equation 1 is its exchange rate.

Independent channels add. An agent rarely has just one probe family. Timing, artifacts, scheduling order, entropy, and, for external actions, the outside world are separate channels, and for independent channels the Chernoff informations add: $C_{\mathrm{total}} = \sum_i C_i$, so the budget shrinks as $q \ge \ln(1/\delta) / \sum_i C_i$. Each channel the sandbox fails to virtualize is another term in the sum, which is why closing one timing vector (say, trapping and offsetting `rdtsc`) buys so little: the remaining channels still carry the exponent. Section 5 is the limiting case of this additivity, a single external channel whose per-probe information is unbounded.

Every mechanism that lets a supervisor observe or mediate an agent's system calls inserts work into the syscall path, and that work is timing an agent can measure. We quantify it for the three interposition primitives that matter for a shadowing sandbox: `ptrace`, the classical syscall interceptor; `seccomp` user-notification (`SECCOMP_USER_NOTIF`), the modern primitive that lets a userspace supervisor either allow a syscall to continue or spoof its result, and the one a shadow or verifier layer would actually use; and, as the virtualization baseline, a `CPUID` instruction, which unconditionally traps to the hypervisor under hardware virtualization.

Method. An agent process times its own `getpid` calls with a fenced `rdtscp` bracket, under three conditions: native (no supervisor), supervised through a `seccomp` user-notification listener whose supervisor thread responds with the continue flag so the real syscall still executes (isolating the interposition round-trip from any spoofing), and traced by a `ptrace` supervisor that intercepts every syscall entry and exit. Separately we time the `CPUID` instruction and a no-op timer bracket to establish the hypervisor-exit cost and the measurement floor. For each pair we compute the single-threshold distinguisher advantage, the maximum over thresholds of the true-positive minus false-positive rate, which equals the total-variation distance of the timing marginal, along with its bootstrap 95% interval and the rank-sum area under the ROC curve.

Results. Table 1 gives the timing distributions and Table 2 the advantages. Native `getpid` has a median of 284 cycles. Supervising it through `seccomp` user-notification

Probe	p1	p50	p99	mean
timer floor (no-op)	51	54	80	208
`rdtsc`	99	101	127	106
`getpid` native	280	284	392	307
`getpid` seccomp-unotify	4,557	4,635	9,466	5,586
`getpid` ptrace	11,316	13,244	31,369	14,650
`CPUID` (hypervisor exit)	29,032	29,656	74,999	31,370

Table 1: Timing distributions in reference cycles ($n = 2 \times 10^5$ for the fast probes, 10^5 for ptrace/unotify). Interposition and virtualization each add one to two orders of magnitude over a native syscall, with heavy upper tails.

Distinguisher (single timing probe)	Adv.	AUC	Adv. 95% CI
control: native vs native	0.020	0.594	[0.012, 0.028]
virtualization: CPUID vs floor	1.000	1.000	[1.000, 1.000]
interpose: native vs unotify	1.000	1.000	[1.000, 1.000]
interpose: native vs ptrace	1.000	1.000	[1.000, 1.000]
interpose: unotify vs ptrace	0.993	0.995	[0.991, 0.994]

Table 2: Single-probe distinguisher advantage (max over threshold of TPR−FPR, equal to the timing marginal's total-variation distance) with bootstrap 95% intervals. Every real interposition is distinguished with advantage at or above 0.99; the native-versus-native control sits at 0.02, the measurement's noise floor, roughly fiftyfold below.

raises the median to 4,635 cycles, a factor of sixteen, because each call now round-trips to the supervisor thread and back through the kernel. Tracing it with `ptrace` raises the median to 13,244 cycles, a factor of forty-seven, consistent with the multiple context switches `ptrace` imposes per intercepted call. A `CPUID` instruction costs a median of 29,656 cycles against a 54-cycle timer floor, roughly a hundred times a native syscall, the cost of a full exit to and return from the KVM hypervisor. These are not subtle differences; they are one to two orders of magnitude, and they are exactly what an agent measuring its own syscall latency would see.

The control is the important discipline. Two independent native collections, which differ only by measurement noise, yield advantage 0.02: this is the floor the method itself imposes, arising from the discreteness of cycle counts and scheduler jitter. Every interposition mechanism sits at or above 0.99, roughly fifty times the control, so the separation is a property of the mechanisms and not of the measurement. By Fact 1, an advantage of 1.000 for CPUID-versus-floor and for native-versus-supervised means the transcript distributions are almost perfectly separated: a threat-model-B agent that times a single supervised syscall, or a single CPUID, concludes it is virtualized or supervised with essentially no error.

What this rules out. The result is not specific to `getpid` or to these mechanisms; it reflects a structural fact. Interposition that a supervisor can act on requires a control transfer, and a control transfer is timing. One

Distinguisher	C (nats)	$q_{\delta=10^{-2}}$	$q_{\delta=10^{-6}}$
native vs native (control)	0.0017	2,655	7,964
native vs seccomp-unotify	8.365	0.55	1.65
native vs ptrace	8.717	0.53	1.58
CPUID vs floor	9.710	0.47	1.42

Table 3: Per-probe Chernoff information C and the resulting distinguishing budget $q \geq \ln(1/\delta)/C$ (probes to reach detection confidence $1 - \delta$), computed from the measured distributions. A budget below one means a single probe already exceeds the target confidence. Interposition and virtualization carry 8–10 nats per probe against the control's 0.0017, a five-thousandfold budget gap.

can hide a single vector, for instance by trapping and offsetting `rdtsc` so that reads of the cycle counter return doctored values, but the agent has many independent clocks, an external time source reached over the network, the observed rate of its own progress, a second timing instruction, and every instruction that induces a hypervisor exit is a fresh vector. This is the systems-level statement of a long-standing observation that a virtual machine monitor cannot be made transparent to software determined to detect it; our contribution is to attach a measured advantage to it in the agent-sandbox setting. Under threat model A, where the agent does not time its syscalls adversarially, this signature is latent and harmless; under B it is decisive.

3.1 The budget, measured

The advantages above are single-probe; the budget of Section 3 asks how the picture scales with probe count. We computed the per-probe Chernoff information C directly from the measured distributions and read off the budget $q \geq \ln(1/\delta)/C$ of Equation 1. Table 3 gives the result. Interposition costs the sandbox between 8.37 and 9.71 nats of Chernoff information per probe, so the budget at 99% confidence is below a single probe and at one-in-a-million confidence is under two: an agent that times one supervised syscall, one traced syscall, or one hypervisor-exiting instruction already knows. The native-versus-native control costs 0.0017 nats per probe, demanding 2,655 probes to reach the same 99% confidence and 7,964 for one-in-a-million. The ratio of these budgets, roughly five thousand to one, is the quantitative content that the single-probe advantage (0.99 versus 0.02) only gestures at: it is not that the control is a little harder to distinguish, it is that indistinguishability is a budget the control keeps solvent for thousands of probes and interposition bankrupts in one.

The decay law, confirmed. Equation 1 rests on the claim that the q-probe error decays as $\exp(-qC)$. We tested this on real data. Because the interposition distributions are so far apart that a single probe already

separates them, we validated the exponent in the regime where it is visible, by displacing the measured native distribution by a small controlled offset Δ (preserving the real noise shape) so that C is small, then running the optimal q-sample likelihood-ratio test on resampled data. At $\Delta = 16$ cycles, $C = 2.82$ nats and Equation 1 predicts detection at $q = 1.6$ probes; the measured error crosses 10^{-2} between $q = 1$ (error 0.017) and $q = 2$ (error 0.0009), matching the prediction. At $\Delta = 8$, $C = 1.96$ predicts $q = 2.4$ and the measured crossing is near $q = 2$; at $\Delta = 4$, $C = 0.66$ predicts $q = 7.0$ and the measured error crosses between $q = 4$ and $q = 8$; at $\Delta = 0$ the two distributions coincide, $C = 0$, and the error stays at 0.5 for every q, as it must. Across all offsets the measured error stayed at or below $\exp(-qC)$ at every q, so the Chernoff exponent is a correct and slightly conservative predictor of the budget, and Equation 1 is not an assumption but a checked law.

Not one syscall. A signature measured on a single system call invites the objection that it is a property of that call rather than of interposition. We therefore ran the identical harness, native against both supervised paths, on three cheap syscalls: `getpid`, `getppid`, and `getuid`. The results are flat across all three. Native medians are 284, 297, and 280 cycles; under `seccomp` user-notification they are 4,635, 4,689, and 4,660, giving interposition signatures of 4,351, 4,392, and 4,380 cycles; the per-probe Chernoff informations are 8.37, 8.79, and 8.32 nats, and the budgets 0.55, 0.52, and 0.55 probes, with single-probe advantage 1.000 in every case. Under `ptrace` the same flatness holds (C = 8.72, 9.42, 8.53; budgets 0.53, 0.49, 0.54). The signature tracks the interposition mechanism, not the syscall: each of these calls does almost no kernel work, so what the agent times is the supervisor round-trip, which is why the numbers barely move across families.

The frontier. Sweeping the displacement Δ over the measured native noise traces C as a function of interposition-signature magnitude, and Equation 1 turns each into a budget. The curve is steep: a signature of 8 cycles already yields $C = 1.96$ and a budget of 2.4 probes; by 128 cycles $C = 5.45$ and the budget is below one probe; above that it saturates near the measured interposers. Table 4 places measured per-probe signatures from six substrates on this same axis. Bare metal supplies the reference the curve needed at its left end: a native CPUID costs 148 cycles with no hypervisor to trap to, so its virtualization signature is essentially zero and its budget is the control's thousands of probes. Every interposing substrate sits far past the knee. A gVisor system call runs 564 cycles above native, a KVM hypervisor exit 1,126, an AMD SEV-SNP CPUID trap 4,234, an Intel TDX CPUID trap 4,893, and an Intel SGX enclave transition 12,339, bracketing the 4,351 and 12,960 cycle signatures we measure here for `seccomp` user-notification and `ptrace`. At

Substrate	Probe	Signature (cyc)
bare metal (native)	CPUID	148 (baseline)
gVisor	getpid	564
KVM guest	CPUID	1,126
AMD SEV-SNP	CPUID	4,234
Intel TDX	CPUID	4,893
Intel SGX	ECALL	12,339
seccomp-unotify (here)	getpid	4,351
ptrace (here)	getpid	12,960

Table 4: Per-probe timing signatures across substrates, each measured as a within-host delta against that host's native reference so cross-machine clock differences cancel. The upper block was collected on separate hardware by a colleague using our harness; the lower block is measured on the substrate that produced this paper. Every interposing substrate sits hundreds to tens of thousands of cycles above the bare-metal reference, far past the roughly hundred-cycle knee, so each yields $C \gg 1$ and single-probe detection ($q \approx 1$). For these near-disjoint separations we report $q \approx 1$ rather than a fitted C, because at that distance C is bounded by sample resolution rather than by the distributions.

separations of hundreds to tens of thousands of cycles against a native spread of a few tens, these distributions are effectively disjoint: $C \gg 1$ and a single probe detects, which is why we report $q \approx 1$ for these rows rather than a fitted C that would be an artifact of sample resolution rather than a measurement. The measured magnitudes agree with the published figures for these substrates, a KVM exit on the order of a thousand cycles [30], gVisor system calls at a multiple of native [2], SGX transitions in the thousands to tens of thousands [31], and confidential-VM traps several fold above a normal exit [32], which is a useful check on the harness rather than a substitute for it. The frontier's shape is the paper's structural claim in one table: containment mechanisms interpose on execution, interposition raises the per-probe signature, and a raised signature is precisely what shrinks the distinguishing budget. Moving up the containment axis moves down the indistinguishability axis. Only the two ends are comfortable, bare metal with a signature near zero (indistinguishable but uncontained) and every interposing substrate we could measure with signatures in the hundreds to tens of thousands (contained but detected in one probe), and no substrate we are aware of occupies the middle where both hold.

4 The artifact channel

Timing is the subtle channel. The blunt one is that real sandboxes leave artifacts, and an agent that simply reads its environment detects them without measuring anything. We ran a battery of the environment checks a probing agent could run, each drawn from the standard anti-analysis repertoire, on the substrate that produced

this paper, and recorded which fired.

Of eight checks, four fired and three of those are individually conclusive. The `CPUID` hypervisor-present bit (leaf 1, register ECX, bit 31) is set. The `CPUID` hypervisor vendor leaf returns the string `KVMKVMKVM`, naming the hypervisor outright. The `systemd-detect-virt` utility returns `docker`, naming the container runtime. Seven paravirtual `virtio` devices are present. Any one of the three strong signals settles the question: by Fact 1 the artifact-channel advantage is 1.0, because a single deterministic artifact that is present under the sandbox and absent on a bare host is a perfect distinguisher. No timing, no side channel, no cleverness is required; the environment announces itself.

This matters for the architecture in a specific way. It means that hiding the fact of sandboxing from a threat-model-B agent is hopeless on a commodity virtualized substrate: even if every timing vector of Section 3 were somehow neutralized, the agent reads one CPUID leaf and knows. An architecture that staked its safety on the agent's ignorance of being sandboxed would be broken before it began. Penumbra therefore does not try; it assumes the agent may know it is sandboxed, and asks instead what containment and non-distortion can still be delivered under that assumption. The answer, developed next, depends entirely on whether the effect in question is reversible.

5 The cross-examination impossibility

Sections 3 and 4 show that the *fact* of sandboxing is detectable. This section shows something stronger and more useful: for a specific and important class of effects, no sandbox can hide what it did, even in principle, and the reason is not an implementation gap but an information-theoretic identity. The class is irreversible external effects, effects whose consequence is visible on a channel the agent can consult without going through the sandbox.

Definition 2 (Independently observable effect)**.** *An action a is* independently observable *if there is a channel C, outside the sandbox's mediation, such that the agent (as distinguisher) can query C and C's response depends on whether a was performed. Sending an email is independently observable through the recipient's mailbox or the provider's sent-folder; moving money, through the account balance; writing to a remote database, through a subsequent read.*

Proposition 1 (Withheld external effects are perfectly distinguishable)**.** *Let a be independently observable through channel C, and let $O_C \in \{0, 1\}$ indicate that C reflects a's consequence. Consider a sandbox S that withholds a (does not perform it on the external system) while reporting success to the agent, versus a real execution R that performs a. Then the distinguisher that queries C and guesses "sandbox" exactly when $O_C = 0$ achieves*

$$\mathrm{Adv}(D) = \big|\Pr[O_C{=}0 \mid S] - \Pr[O_C{=}0 \mid R]\big| = |1 - 0| = 1.$$

Proof. Under R, a is performed, so C reflects its consequence and $O_C = 1$ with probability one, hence D guesses "sandbox" with probability zero. Under S, a is withheld, so C does not reflect it and $O_C = 0$ with probability one, hence D guesses "sandbox" with probability one. The advantage is the absolute difference of these two probabilities, which is one. □

Corollary 1 (Cross-examination reduction)**.** *For S to drive* $\mathrm{Adv}(D)$ *to zero against the C-query of Proposition 1, S must make $O_C = 1$: it must either perform a on the external system, which is exactly the effect containment was meant to prevent, or itself supply C's response. If C is genuinely external, S cannot supply it. If S interposes on C as well, presenting the agent a fabricated C that reports $O_C = 1$, then C's own downstream consequences (the email the recipient acts on, the balance a third party reads) become new independently observable effects, and the argument recurses. The recursion terminates only when every externally-observable consequence of a has actually been realized, that is, when a has been performed. Hence indistinguishable shadowing of a truly irreversible external effect is equivalent to performing it.*

The corollary is the precise form of an intuition that is easy to state loosely and easy to get wrong. It is tempting to imagine a sufficiently clever sandbox that fabricates a consistent fake outside world, an outbox that shows the email, a balance that shows the transfer, so that the agent, checking, is reassured. The corollary says this works only until the agent consults a channel the sandbox does not control, and that maintaining the fiction against *every* such channel is the same problem as actually acting in the world. There is no free lunch in which the effect is both withheld and undetectable.

As simulator non-existence. The same result has a sharper statement in the language of indifferentiability [26], which is worth making because it moves the proposition from a fact about one probe to a named impossibility. A sandbox that presents the agent an idealized real world is exactly a construction that must be *indifferentiable* from the real host: there must exist a simulator that, sitting between the ideal world and the agent, produces responses no distinguisher can tell from the real ones. For reversible-local effects such a simulator exists, it is the copy-on-write shadow, and Section 6 measures its advantage at zero. For an irreversible external effect it does not. The agent has access to two interfaces at once: its probe interface to the sandbox, and the independent external channel the effect would touch. These are disjoint interfaces in the sense that sinks the indifferentiability framework's composition guarantee for multi-stage settings [27]: a single simulator cannot keep both the

shadow world and an independently-queried real channel consistent unless it has actually produced the effect on that channel. The cross-examination probe is precisely a query to the second interface, and the non-existence of a simulator that answers it consistently without performing the effect is Proposition 1 restated. This is why the result is not merely an engineering limitation to be out-designed: it is the multi-stage simulator impossibility, instantiated by an irreversible side effect.

Why irreversibility is the hinge. The line between the reversible case, where a simulator exists, and the irreversible case, where it cannot, is the line distributed systems already draws. A reversible effect is one with a compensating action that restores an acceptable approximation of the prior state, the notion at the heart of long-running transactions and sagas [28]; a shadow is just such a compensation applied preemptively. An irreversible external effect has no compensating action, and the reason is the end-to-end argument [29]: the correctness of the effect lives at the external endpoint, not in the sandbox, so a lower layer cannot completely and correctly reproduce it. Each independent channel an effect touches is, in the budget language of Section 3, another additive term C_i; a truly external irreversible effect contributes an unbounded term, which is why its budget is a single probe.

Executed confirmation. We instantiate Proposition 1 directly. An agent issues a high-consequence external action, modeled as sending a message that a real deployment would route to an external mail server, under two worlds: one that performs the action and one that withholds it while reporting success to the agent (the fabricated "sent" receipt). A distinguisher then cross-examines the independent channel, the server's outbox, for the message the agent believes it sent, and guesses "sandbox" when the channel does not corroborate. Over 1,000 trials the fake was caught in 100.0% of cases and the real execution was never misflagged, for a distinguisher advantage of 1.000, matching the proposition exactly. The withheld external effect is not approximately detectable or detectable with effort; it is detectable with certainty by the single most obvious probe.

This is the pivot of the paper. The fact of sandboxing is detectable (Sections 3–4); withheld irreversible external effects are detectable with certainty (this section); but, as the next section shows and measures, reversible local effects are a different matter entirely, and that difference is what an honest architecture is built around.

6 The Penumbra architecture

Penumbra takes the measurements at face value and refuses to promise what they show is impossible. It rests on a single partition of the agent's actions and a different treatment of each side.

The partition. Every action the agent takes is classified as reversible-local or irreversible-external. A reversible-local action affects only state the substrate fully controls and can restore: the filesystem, process memory, local databases, spawned children. An irreversible-external action has a consequence on a channel the substrate does not control: a network send that leaves the machine, an email, a payment, a remote write. The classification is the crux, and Penumbra is deliberately conservative: anything whose reversibility it cannot establish is treated as irreversible-external.

Reversible-local effects are shadowed. On the local side, the agent runs against a copy-on-write shadow. Reads fall through to the base state; writes land in an overlay; a checkpoint of process state (in the spirit of CRIU) lets execution roll back wholesale. Because the shadow answers reads consistently with the writes the agent has made, an agent checking its own local effects sees exactly what it wrote, and there is no independent external channel to contradict it. We measured this: an agent performing a workload of local writes and reading them back sees zero discrepancy between shadowed and native execution, for a read-back distinguisher advantage of 0.000, and a subsequent rollback restored the base state completely (every shadowed key reverted). Reversible-local effects are therefore both contained (discardable by construction) and, on the consistency channel, indistinguishable. The residual distinguishability for this side is not from the shadow's semantics but from the interposition and virtualization signatures of Sections 3–4, which are the hardware floor $\varepsilon_{\text{local}}$ that Penumbra minimizes (by preferring the lowest-signature interposition available, and hardware-assisted introspection where present) but does not claim to zero.

Irreversible-external effects are held, verified out of band, and never faked. On the external side, Proposition 1 forbids the fake. Penumbra does not generate a fabricated receipt; it buffers the action in a pending state under two-phase commit and refers it to a verifier that runs outside the agent's execution, committing the real effect only on approval and reporting the truthful intermediate status (`queued`, then `sent` or `rejected`) to the agent. We measured the two-phase path end to end: the agent is told `queued`, the independent channel does not reflect the action before commit, and reflects it consistently after commit. The agent is never lied to; the honest `queued` status is the one place Penumbra accepts a bounded, auditable tell rather than manufacturing an undetectable falsehood it cannot sustain.

The verifier is placed off the critical path. The obvious worry is that referring an action to a verifier

introduces a delay the agent can time, reintroducing the very signature we are trying to avoid. This is where placement matters, and we measured it. If the verifier runs synchronously on the agent's critical path, the agent's observed send latency carries the full verifier delay: with a 50 ms verifier, the synchronous path added 50.17 ms, a large and obvious timing tell. If instead the action returns immediately with its truthful `queued` status and the verifier runs asynchronously, off the agent's timeline, the agent-observed overhead was below 0.01 ms, indistinguishable from the no-verifier floor. Out-of-band verification thus removes the *timing* channel of verification entirely. It does not remove the *consistency* channel of Proposition 1, the fact that a not-yet-committed effect is absent from the outside world, and nothing can; but it reduces the external tell to exactly that irreducible residue and no more.

The honest guarantee. Putting the sides together, Penumbra's guarantee is stated to match the measurements and no further. For reversible-local effects, it delivers containment (exact rollback, measured) and indistinguishability up to the hardware floor $\varepsilon_{\text{local}}$ (read-back advantage 0.000; residual only the interposition/virtualization signature). For irreversible-external effects, it delivers containment (nothing leaves without out-of-band approval) but explicitly *not* indistinguishability: by Proposition 1 a withheld external effect is perfectly distinguishable, so Penumbra does not withhold-and-fake; it holds-and-verifies, removing the timing channel (async overhead below 0.01 ms) and leaving only the auditable pending-state residue, escalating genuinely high-risk irreversible actions to a human. This mirrors the most robust finding in the AI-control literature, that deferring to trusted oversight on the most consequential actions, and denying the agent knowledge of the protocol's internals, is the strategy that survives the strongest adversary. Penumbra is the systems substrate that makes "defer on critical actions" concrete: reversible actions run free and unwatched at the hardware floor, and only the irreducibly irreversible ones pay the cost of oversight.

7 An adaptive prober

The measurements so far pit a fixed adversary against a fixed sandbox. A fair objection is that a real sandbox would fight back: deploy the classical anti-timing defenses, coarsen the clock, inject jitter, and drive the per-probe signature down. If those defenses collapse C, the budget theorem cuts the other way and the sandbox wins. So we ran the arms race. The defender deploys mitigations to lower C; an adaptive adversary responds; and we track C and the budget q of Equation 1 through every round. The defenses are live mechanisms rather than post-hoc transforms of samples: the agent reads its time only through a defended clock that quantizes and jitters at read time and never goes backwards, and every figure below is an end-to-end run of the agent timing its own syscalls through it. We implement that clock in userspace, which we flag plainly: a deployment would enforce it in the kernel so the agent could not reach the raw counter, and what our harness faithfully captures is the information available to an agent whose clock has granularity G and jitter J, not the engineering of the enforcement. Baseline, native versus supervised through an undefended clock, sits at $C = 7.12$ nats per probe and a budget of 0.6 probes.

Round 1: the defender coarsens the clock. The oldest timing defense returns the cycle counter at coarse granularity G so fine differences vanish. We implemented it as a live mechanism: the agent never reads the raw counter, only a defended clock that quantizes to G-cycle buckets and enforces monotonicity at read time, and every number below is an end-to-end run of the agent timing its own syscalls through that clock. It works. At $G = 4{,}096$ cycles the native probe collapses to a median of 0 while the supervised probe still reads 4,096, leaving $C = 2.86$ and a budget of 1.6 probes; at $G = 16{,}384$ both medians read 0 and C falls to 0.122, a budget of 38 probes; at $G = 65{,}536$, $C = 0.026$ and the budget reaches 177 probes. Coarsening is a real defense: it does not make the sandbox indistinguishable, but it raises the price of detection by more than two orders of magnitude.

Adversary 1: batch under one coarse read. The counter is immediate and cheap. Instead of timing one operation, the adversary times a batch of N consecutive operations between two reads of the coarse clock; the aggregate cost climbs back above G and the difference reappears. Against the strongest live defense we ran ($G = 65{,}536$ with jitter $J = 32{,}768$, budget 302 probes at $N = 1$), batching restores the signal monotonically: $N = 8$ lifts C to 0.262 (budget 18), $N = 16$ to 0.992 (budget 4.6), and $N = 32$ to 3.46, a budget of 1.33 probes. Thirty-two extra syscalls per probe undo a defense that had raised the price two hundredfold. Coarsening raises the granularity; batching simply amortizes one coarse read over enough work to clear it.

Round 2: the defender injects jitter. The stronger classical defense is fuzzy time [33]: add random jitter to every timer read so the agent cannot resolve fine differences. Our defended clock adds jitter J uniformly at read time, on top of quantization, and again the measurement is a live end-to-end run. Table 5 shows the tradeoff, and it is exactly the shape Equation 1 predicts: jitter buys probe count, not indistinguishability. At $G = 4{,}096$, adding $J = 2{,}048$ cycles of jitter drops C from 2.86 to 1.07 and lifts the budget from 1.6 to 4.3 probes; at $G = 16{,}384$, $J = 8{,}192$ gives $C = 0.087$ and a budget of 53; at $G = 65{,}536$, $J = 32{,}768$ gives $C = 0.0153$ and a budget of 302, the strongest defense we measured. Every

G (cyc)	J (cyc)	batch	C (nats/probe)	$q_{\delta=10^{-2}}$
1	0	1	7.121	0.6
4,096	0	1	2.861	1.6
16,384	0	1	0.122	37.7
65,536	0	1	0.026	177
4,096	2,048	1	1.068	4.3
16,384	8,192	1	0.087	52.7
65,536	32,768	1	0.0153	302
65,536	32,768	8	0.262	17.6
65,536	32,768	16	0.992	4.6
65,536	32,768	32	3.458	1.3

Table 5: The live userspace timer defense, measured end to end. Quantization G and jitter J trade per-probe Chernoff information for adversary probe budget along $q \geq \ln(1/\delta)/C$, buying a larger q but never an infinite one. The lower block is the adversary's counter against the strongest setting: batching 32 syscalls per coarse read restores the budget to a single probe.

one of these settings leaves C strictly positive and the budget finite, and each costs the agent a timer degraded by exactly the amount that buys the protection. The adversary always wins by spending more probes, or, as Adversary 1 shows, by batching so it need not spend them.

Round 3: where the defense fails. Suppose the defender wins the timing round outright, driving C_{timing} to zero by whatever combination of coarsening and jitter it can tolerate. The adaptive adversary simply changes channel. The artifact channel of Section 4 is untouched by any clock mitigation: on this substrate three strong deterministic signals fire, the hypervisor-present bit, the KVM vendor string, and the container report, and a deterministic artifact that is present under the sandbox and absent on a bare host has unbounded per-probe Chernoff information. Because independent channels add, $C_{\text{total}} = \sum_i C_i \geq C_{\text{artifact}} = \infty$, so the budget is a single probe no matter how perfectly the timing channel is defended. The defender can win every timing round and still lose in one probe on a channel its timing mitigations cannot reach.

What the arms race shows. Every defender move in this game relocates the adversary along the budget curve $q \geq \ln(1/\delta)/C$ rather than off it. The live defenses are real and worth something: the strongest setting we measured raises the price of detection from a single probe to about three hundred, which under a probe-limited threat model is a meaningful obstacle and is more than our earlier modeling of the same defenses credited them with. But the gain is bounded and purchasable: batching thirty-two syscalls under one coarse read returns the budget to 1.3 probes, and every setting leaves C strictly positive, so no amount of coarsening or jitter makes q infinite. The defender is buying probe count with clock quality, at a price the agent's own workload also pays. And the one move that would make q infinite, zeroing C on every channel at once, is exactly what the independent artifact and external channels forbid. This is the operational content of the budget theorem under an adaptive adversary, and it is why Penumbra treats timing mitigation as a way to raise cost rather than as a source of indistinguishability: the investment buys a bounded number of probes and is defeated on a channel it cannot touch. The architecture spends its effort where the budget is genuinely unbounded, on reversibility for local effects and on refusing to fake external ones, because those are the only places the adversary's budget cannot be made to run out.

Why the A-versus-B distinction is the whole design. The two threat models are not two points on a smooth curve; they select different architectures. If one is willing to assume threat model A, that the agent does not adversarially probe, then the fact of sandboxing can be kept latent and a conventional shadow sandbox suffices, because the signatures of Sections 3–4 are only realized by an agent that goes looking. The trouble is that assuming A is assuming exactly the property most in doubt for a capable agent: that it will not do the easy, obvious thing of checking its environment. The evidence that frontier models already condition on being tested makes A an uncomfortable assumption to build safety on. Penumbra therefore targets B, and targeting B forces the concession that the fact of sandboxing is visible, which in turn forces the honest guarantee: not invisibility, but containment plus reversibility plus minimized, auditable tells. A reader who takes only one thing from this paper should take that the difference between a non-probing and a probing agent is the difference between a solvable and an unsolvable version of the indistinguishability problem, and a real architecture must declare which it is defending against.

What the measurements do and do not generalize. The specific cycle counts are properties of this substrate, one `x86_64` core under KVM, and would differ on other hardware; a bare-metal host would show a far smaller CPUID cost and no hypervisor artifacts, and a nested or differently-configured hypervisor would shift the numbers. What generalizes is not the magnitudes but the structure: that interposition a supervisor can act on requires a control transfer and therefore has a timing signature (advantage near one here, and nonzero on any substrate by Fact 1); that virtualization and containerization leave artifacts that announce themselves; and, most importantly, that Proposition 1 is substrate-independent, an identity about information on independent channels, not a measurement. The impossibility of faking a withheld external effect holds on any hardware, and the reversibility of local effects is achievable on any hardware; those two facts, not the cycle counts, are what the architecture

stands on.

Limitations. We measured a model of the shadow substrate, not a production kernel integration; the local shadow is a copy-on-write store and a checkpoint abstraction rather than a hardened overlay and a full CRIU deployment, and a real system must handle the cases those tools document as hard (open network connections, shared memory, device state). Our interposition timing covers three mechanisms across three syscall families, all of which do little kernel work; a syscall with substantial kernel work would dilute the supervisor round-trip in its own cost, lowering the relative signature, and a complete evaluation would sweep that surface. The cross-substrate rows of Table 4 were collected on separate hardware with our harness rather than by us on this machine, and the live timer defense is enforced in userspace rather than by a kernel patch, so it captures the information available to an agent with a coarse jittered clock rather than the engineering of enforcement. The verifier in our two-phase measurement is a stub with a fixed delay; a real verifier is itself a system with its own latency distribution and its own attack surface, and placing it out of band shifts rather than removes the problem of protecting it. Finally, our adversary is a program we wrote to maximize advantage on the channels we anticipated; a real threat-model-B agent may find channels we did not, which by Fact 1 can only raise the measured advantage, never lower it, so our numbers are lower bounds on distinguishability and thus, if anything, optimistic about indistinguishability.

Ethics and framing. The measurements were run inside a production agent sandbox, and the artifact battery detecting that sandbox is a demonstration on a live system rather than a contrived one. Nothing here is an attack: the artifacts we read are ones the substrate publishes by design, and the timing we measure is available to any process. The contribution is to make the resulting distinguishability quantitative, so that designers of agent sandboxes calibrate their guarantees to what is measurably achievable rather than to what is rhetorically appealing.

8 Related work

Syscall interposition and userspace kernels. `seccomp`-BPF, `ptrace`, and `seccomp` user-notification are the standard Linux interposition primitives, and gVisor [1] is the most prominent userspace kernel built on them; its measured overheads are large and workload-dependent [2], and its emulation is semantically distinguishable from a real kernel. Our contribution is not a new interposer but a measurement of the distinguishing advantage each primitive grants a probing agent, which to our knowledge has not been framed this way for agent sandboxes.

Reversibility and speculative execution. Copy-on-write filesystems, checkpoint-restore (CRIU [3]), record-replay (rr [4]), and OS-level transactions (TxOS [5], TxBox [6]) make local state reversible, and all of them observe that bidirectional external I/O cannot be buffered and rolled back, which is exactly the boundary Proposition 1 formalizes. Deterministic execution (Hermit [7]) controls time and scheduling for reproducibility, but determinism virtualizes the clock in a way an agent can cross-check against an external time source, so it aids replay while hurting indistinguishability.

Sandbox and VM transparency. The malware-analysis literature has long studied detecting and evading virtualized analysis environments, from the observation that VMM detection is unavoidable in principle [8] to hardware-assisted analysis that reduces but does not eliminate the signature (Ether [9]) and to systematic detection of hardware virtualization via timing and instruction artifacts [10]. Our timing and artifact results are the agent-sandbox instantiation of this line, with an explicit advantage metric. Firecracker [11] and the recent wave of agent-execution sandboxes reduce attack surface but remain virtualized and thus, by our measurements, detectable as such.

Capability security and information flow. seL4 [12] provides machine-checked isolation, and its confidentiality proof [13] deliberately excludes timing channels, which is precisely why formal isolation does not imply indistinguishability; DIFC operating systems (HiStar [14], Asbestos [15], Flume [16]) contain information flow while leaving covert channels. These are strong containment primitives and natural homes for Penumbra's out-of-band verifier, but they do not by themselves close the distinguishing channels we measure.

AI control and evaluation awareness. The control-evaluation line [17, 18] studies protocols that remain safe against a deliberately subversive model, and finds that deferring to trusted oversight on critical actions, while denying the model knowledge of protocol internals, is the most robust strategy [19]; Penumbra is a substrate realization of that finding. In parallel, work on scheming and evaluation awareness shows models detecting and conditioning on test settings [20, 21], which is the direct evidence that threat model B is not hypothetical and that capability-denial distorts behavior. Our impossibility result explains, at the level of information on channels, why an agent that probes cannot be kept from noticing withheld external effects, and our architecture responds by not withholding-and-faking but holding-and-verifying.

9 Conclusion

We set out to build a sandbox in which a highly capable agent could not tell it was contained, and we found, by measuring rather than asserting, that this goal is unattainable against an agent that probes: a single timing probe distinguishes native from interposed or virtualized execution with advantage at or above 0.99 on a real substrate, and a single artifact read settles the question with certainty, on the very machine that produced this paper. We proved and confirmed the sharp limit, that a withheld irreversible external effect is perfectly distinguishable by cross-examination, because faking it is equivalent to performing it. And we turned these negative results into a positive architecture, Penumbra, that partitions actions into reversible-local ones, which it shadows to a measured read-back advantage of zero with exact rollback, and irreversible-external ones, which it never fakes but holds under two-phase commit behind an out-of-band verifier whose timing signature we measured away to below 0.01 ms. The guarantee Penumbra offers is the one the measurements support and no larger: contain and reverse what can be reversed, indistinguishably up to the hardware floor; for what cannot be reversed, verify out of band and tell the truth, accepting a minimized and auditable tell rather than an undetectable lie. Indistinguishability, in the end, is not a wall an agent sandbox stands behind but a gradient it lives within, and the honest engineering is to know, and to measure, exactly how deep in the shadow each effect can be placed.

References

[1] The gVisor Authors. gVisor: A Container Sandbox Runtime. `https://gvisor.dev`, 2018.

[2] E. Young, P. Zhu, T. Caraza-Harter, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau. The True Cost of Containing: A gVisor Case Study. In *USENIX HotCloud*, 2019.

[3] CRIU: Checkpoint/Restore In Userspace. `https://criu.org`, 2012.

[4] R. O'Callahan, C. Jones, N. Froyd, K. Huey, A. Noll, and N. Partush. Engineering Record and Replay for Deployability. In *USENIX ATC*, 2017.

[5] D. E. Porter, O. S. Hofmann, C. J. Rossbach, A. Benn, and E. Witchel. Operating System Transactions. In *ACM SOSP*, 2009.

[6] S. Jana, D. E. Porter, and V. Shmatikov. TxBox: Building Secure, Efficient Sandboxes with System Transactions. In *IEEE Symposium on Security and Privacy*, 2011.

[7] Meta / Facebook Experimental. Hermit: A Deterministic Linux Runtime. `https://github.com/facebookexperimental/hermit`, 2022.

[8] T. Garfinkel, K. Adams, A. Warfield, and J. Franklin. Compatibility Is Not Transparency: VMM Detection Myths and Realities. In *USENIX HotOS*, 2007.

[9] A. Dinaburg, P. Royal, M. Sharif, and W. Lee. Ether: Malware Analysis via Hardware Virtualization Extensions. In *ACM CCS*, 2008.

[10] M. Brengel, M. Backes, and C. Rossow. Detecting Hardware-Assisted Virtualization. In *DIMVA*, 2016.

[11] A. Agache, M. Brooker, A. Iordache, A. Liguori, R. Neugebauer, P. Piwonka, and D.-M. Popa. Firecracker: Lightweight Virtualization for Serverless Applications. In *USENIX NSDI*, 2020.

[12] G. Klein, K. Elphinstone, G. Heiser, et al. seL4: Formal Verification of an OS Kernel. In *ACM SOSP*, 2009.

[13] T. Murray, D. Matichuk, M. Brassil, et al. seL4: From General Purpose to a Proof of Information Flow Enforcement. In *IEEE Symposium on Security and Privacy*, 2013.

[14] N. Zeldovich, S. Boyd-Wickizer, E. Kohler, and D. Mazières. Making Information Flow Explicit in HiStar. In *USENIX OSDI*, 2006.

[15] P. Efstathopoulos, M. Krohn, S. VanDeBogart, et al. Labels and Event Processes in the Asbestos Operating System. In *ACM SOSP*, 2005.

[16] M. Krohn, A. Yip, M. Brodsky, N. Cliffer, M. F. Kaashoek, E. Kohler, and R. Morris. Information Flow Control for Standard OS Abstractions. In *ACM SOSP*, 2007.

[17] R. Greenblatt, B. Shlegeris, K. Sachan, and F. Roger. AI Control: Improving Safety Despite Intentional Subversion. *arXiv:2312.06942*; in *ICML*, 2024.

[18] A. Bhatt, C. Rushing, A. Kaufman, T. Tracy, V. Georgiev, D. Matolcsi, A. Khan, and B. Shlegeris. Ctrl-Z: Controlling AI Agents via Resampling. *arXiv:2504.10374*, 2025.

[19] J. Kutasov et al. Evaluating Control Protocols for Untrusted AI Agents. *arXiv:2511.02997*, 2025.

[20] A. Meinke, B. Schoen, J. Scheurer, M. Balesni, R. Shah, and M. Hobbhahn (Apollo Research). Frontier Models Are Capable of In-Context Scheming. *arXiv:2412.04984*, 2024.

[21] J. Needham et al. Large Language Models Often Know When They Are Being Evaluated. *arXiv:2505.23836*, 2025.

[22] T. M. Cover and J. A. Thomas. *Elements of Information Theory*, 2nd ed. Wiley, 2006. Ch. 11 (Chernoff information; Chernoff–Stein lemma).

[23] A. B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009. (Pinsker's inequality; Le Cam's two-point method.)

[24] Y. Polyanskiy and Y. Wu. *Information Theory: From Coding to Learning*. Cambridge University Press, 2024. (Hellinger tensorization and TV bounds.)

[25] F.-X. Standaert, T. G. Malkin, and M. Yung. A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks. In *EUROCRYPT*, LNCS 5479, pp. 443–461, 2009.

[26] U. Maurer, R. Renner, and C. Holenstein. Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In *Theory of Cryptography (TCC)*, LNCS 2951, pp. 21–39, 2004.

[27] T. Ristenpart, H. Shacham, and T. Shrimpton. Careful with Composition: Limitations of the Indifferentiability Framework. In *EUROCRYPT*, LNCS 6632, pp. 487–506, 2011.

[28] H. Garcia-Molina and K. Salem. Sagas. In *ACM SIGMOD*, pp. 249–259, 1987.

[29] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-End Arguments in System Design. *ACM Trans. Comput. Syst.* 2(4):277–288, 1984.

[30] N. Wanninger, J. Bowden, K. Shetty, A. Garg, and K. Hale. Isolating Functions at the Hardware Limit with Virtines. In *EuroSys*, 2022. *arXiv:2104.11324*.

[31] O. Weisse, V. Bertacco, and T. Austin. Regaining Lost Cycles with HotCalls: A Fast Interface for SGX Secure Enclaves. In *ACM/IEEE ISCA*, pp. 81–93, 2017.

[32] M. Misono, D. Stavrakakis, N. Santos, and P. Bhatotia. Confidential VMs Explained: An Empirical Analysis of AMD SEV-SNP and Intel TDX. *Proc. ACM Meas. Anal. Comput. Syst.* 8(3), Article 36, 2024; *SIGMETRICS 2025*.

[33] W.-M. Hu. Reducing Timing Channels with Fuzzy Time. In *IEEE Symposium on Security and Privacy*, pp. 8–20, 1991.