CNVS Framework

Engineering-Hardened Execution Report

Massimo Comitato

Independent Researcher, Italy
Email: ***massimocomitato@gmail.com***
ORCID: 0009-0002-8301-3589

General Section

Abstract

Introduction to instantiation

In the development of distributed architectures, the data genesis phase traditionally represents a potential systemic vulnerability.
A critical analysis might suggest that the CNVS framework shifts the trust requirement from the validation phase to the initial declaration phase, making the system completely dependent on the Instantiator.
Most existing systems limit themselves to managing data after the insertion phase: this exposes the entire architecture to the management of false or inconsistent information ("Garbage In, Garbage Out").

The CNVS model mitigates this weakness not through trust assumptions, but through its architecture itself.

Asymmetry between Definition and Execution

The Instantiator retains authority during the genesis phase: it defines the instance, introduces its semantic rules (the family of invariants) and its relational dependencies, but does not have control over the actual object of its declaration.
This clarification is essential to understanding the role and limitations of the instantiator.
Once entered into the framework, the Instantiator is removed from any active role. Execution becomes strictly distributed, and peripheral validation dynamics operate independently and randomly, making the instance difficult to manipulate ex post by its creator.

Post-Genesis Information Asymmetry (Structural Blindness)

The Instantiator suffers from epistemic isolation mirroring that of the external adversary, structured on three levels of stochastic obfuscation:

- **Relational Graph Generation:** Instance fragmentation follows a dynamically generated topological model. The Instantiator cannot map distribution paths to peripheral nodes a priori.
- **Random Invariant Selection (c_i):** Despite having defined the semantic domain (set C), the Instantiator ignores how the specific invariant equations (c_i) are distributed by the system, in order to impose geometric constraints on individual sessions.
- **Injective assignment and obfuscation:** Verification tasks are assigned randomly and injectively (without replacement). The instantiator knows neither the exact number, identity, nor spatial location of the verifiers who will receive the critical fragments.
  This makes any targeted collusion attempt mathematically difficult.
  Unless the instantiator controls the entire critical network surface, which determines the exfiltration of the fragments required for reconstruction, any potential gain from peripheral collusion attempts will be offset by the inability to control and determine the observations of non-colluding nodes.

Effectiveness of the Non-Colluding Minority

Due to critical fragmentation, the Instantiator does not require simple majority control to

validate false data, but rather near-total control of the observation matrix. If the instance correlates multiple variables, the system simply assigns critical tasks to a small minority of non-colluding verifiers. The actual data they report will cause a geometric discrepancy with respect to the declared parameters, automatically triggering the Global Veto (GV), regardless of the number of corrupt nodes endorsing the falsity, rejecting the state (Early Rejection). This premise justifies the system's claim to maintain security even when the instantiator is colluding, as the declaration of patently false invariants and an altered information geometry will be opposed by the Global Veto.

Instance Tautology

A potential attack vector consists in the deliberate introduction of tautological, ambiguous, or structurally poor instances, in order to minimize constraints and force system acceptance. The CNVS does not necessarily prevent the introduction of a tautological, ambiguous, or structurally poor instance. Instead, its architecture limits the evidentiary value of the resulting certificate to the informative strength of the declaration that has been verified:

- The model does not act as an arbiter of objective "Truth," but as a cryptographic coherence engine. If it receives a private instance with real semantic content but logically coherent, it will validate it accordingly.
- This dynamic generates a self-devaluation mechanism: the fiduciary (and commercial) weight of the output certified by the CNVS is directly proportional to the complexity, granularity, and quantity of relational constraints inserted during the genesis phase.
- The burden of qualitative evaluation falls on the end user of the data. An attacker who invests resources to validate an inconsistent statement will obtain mathematically sound certification of irrelevant and therefore useless content.

In conclusion, the CNVS framework cannot guarantee the veracity of the raw data entered into the input at time zero on the zero-point device, but it addresses the problem by developing a structure that ensures the correspondence between the container declared by the Instantiator and rigorous stress tests within the stochastic dynamics of a distributed system. The Instantiator is forced to act with precision and completeness not because of a moral constraint, but to maximize the informational utility of the validated instance, thus obtaining a validation certificate (the system's output) with the greatest possible weight.

Technical note

Theoretical Perimeter

The official documents formalizing the General Theory of Epistemic Dispersion and those relating to the CNVS class model do not discuss the engineering specifications of the application protocol, but rather the structural conditions, general principles, and mathematical framework of the theorems and proofs.

All issues relating to:

- commercial format of the certificate;
- minimum completeness of the declaration;
- choice of measurement scales;
- costs and feasible number of verifiers;
- possible operational sampling;

belong to the engineering level, not to the validity of the general theory or its class model.

Limited Role of the Instantiator

The instantiator introduces a candidate declaration:
S_{cand}
referring to an object belonging to a real state X.
The instantiator may lie in the declaration, but does not control the process that subsequently verifies it.

In the ideal model:
Instantiator $\longrightarrow S_{cand} \longrightarrow$ CNVS$\longrightarrow$ Decomposition, Task, Assignment, Measurement, V_G.

The CNVS decomposition extracts localized predicates and transforms them into measurement tasks;
the fragments are then distributed by the system through stochastic and injective assignment.
Therefore, the instantiator:
declares, but does not assign, the tasks
declares, but does not choose, the verifiers
declares, but does not check, the empirical observations
declares, but does not determine, the outcome of the VG.
The instantiated invariants, their parameters, and the relational binding belong to the internal structure of the CNVS and, in the ordinary regime, are not exposed to verifiers or to the opposing coalition.

The Independent Constraint of the Real Object

The key point is that the statement is compared to an external object:
X = real object.
For each task, the verifier produces:
$obs_i = M_i(X)$,
where M_i is an empirical measurement operation.

The verifier does not simply receive the declared value to confirm it: it receives a local grid or specification, autonomously observes reality, and returns the observation.
Therefore:
no actor decides reality: reality is measured.
The instantiator can check its own statement, but it cannot automatically transform $M_i(X)$ into the desired false value.
In the fictional instance of Test 14 of this Report, the structure of a hypothetical house, Enzo's house, is assumed as the input statement.

A false statement may not converge in the empirical phase of the observations collected for:

- square footage;
- lengths and widths;
- room surface area;
- total surface area;
- height;
- volume;
- geometric ratios;
- perimeter;
- garden;
- relationships between floors and spaces.

The observations are reconstructed hierarchically $S^{(n)} \to S^{(n-1)} \to \cdots \to S^{(0)}$.
A single critical incompatibility can cause:
$V_L = 0$, or $Cons_R = 0$, or $Inv_C = 0$,
and therefore $V_G = 0$.

Tasks in the CNVS

In the theoretical exact model, as reported in the relevant documentation, a random sample of tasks is not selected to be verified.
The Global Veto requires:
$V_G(S) = 1 \iff (\forall s \in Ter(t), \pi_1(V_L(s)) = 1) \land Cons_R(R, E) \land (\forall c_i \in C, c_i(E, R) = 1)$.

All terminals produced by the decomposition and all invariants belonging to the C family are therefore checked.
The manuscript defines V_G as a universal and complete conjunction, not as a sample control.
This is perfectly legitimate in an asymptotic reference model.

The randomness reported in the papers instead concerns:
At: $Ter(t) \times \Xi \to V$,
that is, which verifier is assigned each task, not which tasks will be excluded.
The expression "All possible tasks" must be understood strictly as:
all terminal tasks belonging to the finite universe produced by the complete decomposition of the formalized instance.

This does not mean every conceivable physical property of the universe, but it does mean the absence of sampling with respect to the defined CNVS set.
An implementation that controlled only a subset would be a sampled engineering variant, distinct from the theoretical exact CNVS model.
This clarification allows us to understand the limits of the practical application of the theoretical model, which obviously requires a structural adjustment in the implementation phase, but without the need to rewrite the underlying model, which remains the reference model.

Malicious Instantiator and Colluding Coalition

A malicious instantiator can have:
$S_{cand} \neq S_X$,
where S_X represents the authentic description compatible with the real object.
The coalition can also know, by exfiltration, $C_{int} = \langle \Theta_C, R_{int} \rangle$.

However, knowledge of the invariants does not automatically confer:

- control of the tasks;
- control of the assignments;
- the credentials of the honest verifiers;
- the power to modify their observations;
- control of the external object.

Let us define the set of discriminant fragments:
$F_\Delta = \{f_i \in Ter(t): obs_i(X) \text{ contradicts } S_{cand}\}$.
To have the false statement accepted, the coalition must neutralize all discriminant contributions:
$F_A \subseteq \bigcap_{f\_i \in F\_\Delta} C_i$.

With:
$m_\Delta = |F_\Delta|$,
and with the conditional limit:
$Pr(C_i \mid C_1, \ldots, C_{i-1}) \leq p_{comp} < 1$,
we obtain:
$Pr(F_A \mid C_{int} \text{ exfiltrated}) \leq p_{comp}^{m\_\Delta}$.

Exfiltrating invariants can dramatically increase inferability, but it does not automatically eliminate honest observations nor the need to check the entire critical surface of the falsification.
It is not always necessary q = 1.
However, it is necessary to check or infer:
100% of the critical contributions necessary for that falsification.

Identifying regime and equivalence regime
In empirical Monte Carlo tests, two regimes of application of the CNVS model are described:

an identifying regime and an equivalence regime.
Identifying Regime
The system is identifying when the complete set of observations, relations, and invariants admits only one state:
$E(X) = \{S: V_G(S ; O(X)) = 1\} = \{S_X\}$.
Therefore:
$S' \neq S_X \Longrightarrow V_G(S') = 0$.

In this regime, even complete knowledge of the invariants does not create an acceptable alternative state: the adversary knows the rules, but cannot find a different reality that satisfies them along with all the authentic observations.
The identifying regime represents the ideal goal of applications with high evidential value.
Equivalence Regime
The equivalence regime exists when multiple states satisfy the same statement $|E(X)| > 1$.

This may depend on an instance:

- generic;
- poor;
- ambiguous;
- tautological;
- poorly detailed;
- equipped with invariants that are not very discriminatory.

In this case, the application of the CNVS has not necessarily accepted a falsehood, but has certified that the statement belongs to a class of compatible states $S_{cand} \in E(X)$.

The evidentiary value of the certificate, however, will be low.
In an instance such as "The house has at least one room," the acceptance can be perfectly verified with ease, but it will not provide a useful certificate.
So an equivalence regime ≠ CNVS error, but:
equivalence regime ⇒ low power of the validation certificate.

Significance of Empirical Tests 9, 10, 11, and 12

Test 9 — Separation of Knowledge and Operational Power

Test 9 demonstrates that locally admissible and authenticated values can be rejected by invariants during state reconstruction through the application of the global verification function. In the collapse control phase, Cint disclosure, combined separately with the verification of the required credentials, allows for the construction of an accepted alternative. Test 9, to which we refer, demonstrates that Cint exfiltration alone is not sufficient to reconstruct a valid alternative state.
This introduces the separation of equivalence and identity regimes, better described in Test 12.

Test 10 — Acceptance of Falsehood and Verification of Critical Fragments

Test 10 shows that, in the constructed equivalence model, the curve of acceptance of falsehood with disclosure follows the probability of complete verification of the critical values and credentials. Even with $q < 1$, the event can occur, but only when the coalition controls the entire critical surface required by the alternative, by checking the values and credentials.

Test 11 — Reconstruction of the Authentic State

Test 11 does not measure false acceptance.
It measures the probability that the coalition correctly reconstructs all the critical authentic values and presents an authentic state according to the operational chain
$V_L \rightarrow Cons_R \rightarrow Inv_C \rightarrow V_G$.
Increasing m decreases the probability of completely reconstructing an authentic state.

Test 12 — Explicit Separation of the Two Regimes

Test 12 distinguishes two regimes:
Equivalence Regime and Identity (or Identification) Regime.
In the equivalence regime, the complete disclosure of Θ_C and R_{int} allows the solver to find coherent alternatives, because there exists at least one state that, while not authentic, can satisfy the geometry, binding, and system invariants.
In the identifying regime:
$V_G(X') = \text{VETO } \forall X' \neq X$,
so no non-genuine alternative is accepted, even knowing the state structure.
This distinction depends on the content of the instance initialized at point zero and must be specified in the engineering implementations of the CNVS, because it separates the secrecy of the structure from the identifying strength of the state structure.

Nature of the CNVS Certificate

The CNVS does not certify the absolute truth of an instantiated statement about an object, but rather establishes whether a specific statement has been found to be compatible with empirical observations, relations, and invariants specific to that object.
The certificate is no more valuable than the statement it contains.
A strong, complete, and identifying statement potentially produces a strong certificate.
A poor, ambiguous, or tautological statement produces a correct, but poor, and barely useful, certificate.
This does not invalidate the theory, but constitutes the natural meaning of validating a proposition formally determined at the zero point.

Relative to the universe of the instantiated data, the CNVS does not verify a sampled portion of its declaration.
It fully verifies:

- all generated terminal tasks;
- all mandatory observations;
- all structural relations;

- all invariants belonging to the formalized instance.

Therefore, if:

$$V_G(S(t)) = 1$$

the correct meaning is:
At time t, the entire structure formalized by the candidate declaration has been found to be consistent with all empirical observations, all relational dependencies, and all invariants of the CNVS configuration.
It does not mean:
Every ontologically possible property of the object has been described and certified.
The difference concerns the scope of the declaration, not the completeness of its verification.
The CNVS fully verifies what has been formalized.
This explains why a poor declaration produces a poor certificate; a strong and identifying declaration produces a more powerful certificate.

Possible engineering level: Certificate score

In a future implementation, an index could be associated with the certificate:
$P_{decl} \in [1, 100]$,
which represents the informative and evidentiary strength of the statement.
For example:
$P_{decl} = 100(w_c COV + w_s SPEC + w_i IDENT + w_r RED)$,

with:
$w_c + w_s + w_i + w_r = 1$,
where:

- COV: coverage of relevant attributes;
- SPEC: specificity of the declaration;
- IDENT: ability to uniquely identify the state;
- RED: geometric and relational redundancy.

The deviation from the ideal declaration could be:
$\Delta_{ideal} = 100 - P_{decl}$.

The deviation from the observed object could be calculated, with domain-specific normalizations, as:

$$\Delta_{obs} = 100 \frac{\sum_i w_i \min\left(1, \frac{d_i(D_i, O_i)}{\tau_i}\right)}{\sum_i w_i},$$

where:

- D_i is the declared value;
- O_i is the observed value;

- d_i is the appropriate distance;
- τ_i is the permitted technical tolerance.

The certificate could therefore state:
$Cert_{CNVS} = (V_G, P_{decl}, \Delta_{ideal}, \Delta_{obs}$, residual equivalence class).
This could represent a particularly useful engineering extension to determine an objective criterion for evaluating the informational and utilitarian significance of the validation certificate issued by the CNVS application, even if it is not necessary for the formal correctness of its general theory.

Final summary on the zero point

The CNVS does not certify the absolute truth, but validates a specific statement.
The instantiator verifies what it declares, not the process that subsequently verifies the statement.
In CNVS, the instantiator does not control decomposition, random task assignment, honest observations, or the Global Veto.
Randomness decides who measures, not what is omitted from the theoretical verification: this is because
the exact asymptotic CNVS model verifies all generated terminals and all invariants of the formalized configuration.
No one decides reality: in the verification model, reality is measured.
A malicious instantiator can lie, but it cannot force the real object to confirm the lie. In the identification regime, a false statement must violate at least one relation, observation, or critical invariant.
In the equivalence regime, the certificate may be correct but poorly informative and useless.
The evidential value of the certificate cannot exceed the informative value of the validated statement.

To recap:

- the instantiator only checks the declarative input;
- the CNVS controls the decomposition and verification;
- the verifiers observe an external reality;
- the theoretical model is exhaustive, not sample-based;
- equivalence and identification are two distinct regimes;
- C_{int}'s exfiltration weakens epistemic blindness, but does not automatically nullify empirical evidence and honest credentials;
- C_{int}'s exclusive exfiltration is not sufficient to establish a False State.

Latency Analysis and Application Domains of the CNVS Framework

A complete engineering analysis of the CNVS framework requires breaking down its performance based on its target operating domain.

Compared to traditional distributed architectures, CNVS presents a unique latency profile, dictated by the radical decoupling between peripheral observation and central validation.

Absence of Terminal-Consensus Overhead

Many distributed consensus protocols require communication, state replication, voting, or coordination among participants.
Depending on the protocol and deployment model, these mechanisms can introduce non-negligible computational and network overhead.
CNVS, in contrast, operates through asynchronous parallelism and terminal isolation.
Terminal verifiers neither communicate nor vote with one another.
Each terminal sends its observation along the assigned distributed reconstruction path, contributing to the logically global evaluation of V_G.
This removes terminal-to-terminal voting and agreement messages, but does not eliminate the routing and reconstruction traffic required by G_R. The resulting latency, throughput, and scalability properties must therefore be measured in a deployed distributed implementation.

The Cyber-Physical Domain (The Logistical Limit of the Human Factor)

When CNVS is applied to scenarios that interface with the physical world (e.g., supply chain certifications, physical quality inspections, on-site audits), the overall resolution time is not dictated by the server's computational capacity, but by the logistical time required to collect the data.
In these scenarios, the Human-in-the-loop constraint occurs: the network must wait for the operator to physically move, execute the manual task, and enter the parameter. Consequently, the overall latency scales to the speed of human logistics. This slowdown is not an algorithmic or computational limitation of the CNVS, but rather an unavoidable friction inherent in an interface between the cryptographic domain and the real physical environment.
3. The Native Digital Domain (Real-Time Execution)
In the framework, the workflow is significantly accelerated when implemented in purely IT ecosystems or on electronic measurement devices (e.g., replacing or complementing network protocols like TCP, validating data packets, orchestrating IoT sensors that may already be directly connected to the system).
In this environment, digital nodes (or packet fragments) feed the geometric invariant automatically and continuously, limiting human intervention. The result is a more efficient validation infrastructure, capable of rejecting unauthorized inputs, packet corruption, or state alterations in real time, solely based on central geometric coherence, operating at speeds limited only by bandwidth and processor operating constraints.

Operational Conclusion

The CNVS framework can therefore be interpreted as a dual-domain architecture. In cyber-physical applications, the overall execution time is expected to depend substantially on the time required to perform and collect empirical observations. In native digital applications, automated observation and reconstruction may reduce the human component of latency and

avoid terminal-consensus overhead. Actual latency, throughput, and scalability, however, remain implementation-level properties that are not measured in this report.

Reconstruction Management

One of the most critical architectural challenges in distributed validation systems is managing the reconstruction flow without reintroducing centralized bottlenecks (Single Point of Failure). If the node or server initializing the instance retained control or oversight over the reconstruction phase, the CNVS framework would lose its decentralized nature, exposing the origin (called the zero-point device in this report) to direct vulnerabilities or denial-of-service (DoS) attacks.

To address this structural issue, the architecture implements a reconstruction model based on topological decoupling and an ephemeral ledger.
The Zero-Point Device (Stateless Origin)
The architecture defines the instance origin as the Zero-Point Device (Zero-Point Computer). This node serves exclusively as a Trusted Setup environment. Its role is strictly limited to:

- Acquiring the declaration.
- Generating the relational graph.
- Fragmenting the validation tasks.
- Cryptographically inject them into the network.

The moment data is transmitted, the Zero Point Device becomes stateless and actively disconnects from the validation flow. It does not listen, nor does it act as an aggregation server. It is only recalled (awakened) in extreme crisis scenarios, such as exceeding fatal error thresholds that trigger a network-wide Topological Refresh following a massive Sybil attack.

Topological Decoupling (Routing Blindness and Stochastic Path Obfuscation)

To prevent an adversary from reconstructing the network map through traffic analysis or topological side-channel attacks, CNVS applies the principle of semantic blindness not only to the data, but also to the network topology.
The distribution of terminal logical fragments (validation tasks) is rigorously decoupled from the distribution of reconstruction metadata (the directional graph).
A single edge device will receive a terminal-level logical fragment (e.g., F_1 to N^5) but, at the same time, will receive routing instructions to aggregator nodes belonging to a completely different branch or hierarchical level of the tree (e.g., level N^{-3}).

The logical reconstruction sequence of the parent nodes does not correspond in any way to the physical sequence of the verifiers. This mechanism acts as stochastic path obfuscation. Each end device acts as a "blind courier," mathematically isolating its task knowledge from the overlying topological structure.
Furthermore, the non-uniformity of the terminal fragments hinders reconstruction attempts based on observation of the terminal data volume (Data Aggregation Attack in Traffic

Analysis).

Autonomous Hierarchical Reconstruction and Ephemeral Ledger

Unlike fully replicated ledger architectures, in which broad portions of the global state may be replicated across participating nodes, CNVS proposes a fragmented ephemeral ledger.
The reconstruction of the Global Veto occurs through a peer-to-peer, inverse hierarchical process.
Terminals at level N decode the instructions to send the results to designated nodes at level N^{-1}. When node N^{-1} aggregates the data, it in turn decode its own cryptographic key to transmit the result to level N^{-2}, cascading upward until the synthesis of the originating node S0.
Consequently, the "log" of the Partial State recomposition exists only temporarily and is dispersed throughout the network. If even a fraction of the nodes required for graph reconstruction are missing, the ledger remains incomplete and mathematically unreadable.
It is theoretically possible to trace the identities, possibly cryptographically protected, of the verifiers who performed the terminal observations and the various phases of the ephemeral recomposition by retracing the tree backwards based on the metadata logs embedded in the raw data at the end of the reconstruction.

This operation will not be easy for the terminal verifiers, who are blind to the task assigned to them and to the management of the topological reconstruction phase, which they perform from their own device without active human intervention.
However, it will be possible starting from Device Zero, which could receive the log details (including generated errors) as output along with the validation certificate. Again, it's clear that the colluders' lack of full control over the actions of non-colluding verifiers, or their devices, will hinder any unauthorized attempt to reconstruct the entire state's topology.

Further advanced encryption models and encrypted keys could protect the data temporarily stored on the verifiers' terminal devices, while deleting the data from the device after the layer has been reconstructed could thwart any attacker's exfiltration attempts, limiting them to the brief time window of the execution of the phase.

MULTI-SERVER

Hypothetically, a network of independent multi-servers could be established, operating in the same manner as terminal devices.
This would disperse the management of graph reconstruction across multiple independent entities, hindering unauthorized adversarial reconstruction by forcing a multiple attack on multiple independent servers.

This situation would be further hampered by the NON-UNIFORMITY of the distribution, both in terms of the volume of the fraction of raw data entrusted and in terms of the relative weight of the critical scope of the phase contribution.
To be effective, an attack would require infiltration of all servers, decryption of any

cryptographic keys on all channels, and exfiltration of the exact reconstruction sequence for each individual server point.

The Liveness/Security Tradeoff and Asymmetric (Variable) Redundancy

Since the Zero Point Computer does not intervene to manage latency, the system must be autonomously resilient (liveness) against disconnections or timeouts of aggregator nodes, to prevent it from being easily paralyzed by continuous collusive actions.
The CNVS already includes an extreme level of protection in its architecture: the topological refresh, as per Remark 11.1 of the official document.
However, in practical application, continuously invoking the topological refresh, even with the ability to dynamically purge the verifiers responsible for the paralysis at each cycle, could lead to liveness issues. However, as already discussed, these purges would result in a progressive increase in execution speed in subsequent phases, having discarded colluding or simply slow nodes.
One solution to limit the topological refresh function to extreme cases could be the following.
After the timeout expires or following a disconnection, the system protects itself by entrusting the encrypted fragment of the reconstruction graph to another randomly chosen terminal node.
If a node times out, a new lateral node inherits the encrypted fragment (here, the term "fragment" does not refer to the task in N, but to node N - m, with $0 \leq m \leq N$), which will then be managed by an alternative device among those available to circumvent the block.
Since the zero-point device remains disconnected, the alternative assignment must occur first.
The system therefore provides for the generation of "replicas" of nodes, such that each node is entrusted two or three times, depending on the implementation setting, to lateral devices of active or inactive verifiers.

It is also possible to distribute the number of replicas of the various nodes unevenly, so as to adapt this number to the criticality of the node itself.
Since graph theory states that the number of internal nodes I is always less than the number of terminal leaves L, for a full k-ary tree, in which each internal node has exactly k children, according to the mathematical relation:

$$I = (L - 1) / (k - 1),$$

where k corresponds to the number of divisions of each branch up to the leaves,
the system can rely on a number of verifiers that is always greater than the number of intermediate nodes to be assigned.
If, during graph reconstruction, the system does not receive a response within a timeout defined in the implementation, the computation of that node is entrusted to a new verifier, already prepared for action because it has its replica. If it fails again, a new step occurs, and so on until a redundancy limit defined in the implementation is exhausted.
Once this limit is exceeded, even for a single intermediate node that remains "blocked," a topological refresh is triggered.

To better formalize these concepts, the definitions of the different graph types are illustrated

later in this document.

Mathematical Modeling of Redundancy Risk

Introducing redundancy increases fault tolerance but requires accepting a measurable statistical risk. If an adversary coalition controls a fraction q of the network, it is possible to formalize the probability of intercepting the entire randomly dispersed reconstruction graph. Let m be the total number of fragments needed to reconstruct the ephemeral ledger, and let zi be the specific redundancy limit factor applied to the i-th fragment.
The probability that the adversary captures the i-th fragment through colluding verifiers scattered throughout the network is $1 - (1 - q)^{z\_i}$.
Consequently, the probability that the entire ephemeral ledger is compromised is described by the product, under the hypothesis of independent assignments:

$$P_{\text{Ledger Compromise}} \approx \prod_{i=1}^{m} \left[1-(1-q)^{z_i}\right]$$

where

- m: Total number of critical fragments needed for reconstruction.
- q: Fraction of the network controlled by the adversary coalition.
- z_i: Redundancy factor (number of replicas) applied to the specific fragment.

Analytical Example

Consider a ledger fragmented into m = 16 segments.

- If the adversary compromises a massive 90% share of the network (q = 0.90) and the implementer has set a uniform redundancy factor of z = 3 per fragment, the probability of the adversary harvesting the entire ledger reaches 98.4%.
- However, by calibrating redundancy at z = 2, this probability drops to 85.1%.

The impact of the z parameter becomes even more pronounced in an 80% collusion scenario (q = 0.80):

- With z = 3, the total capture probability is 87.9%.
- By reducing the factor to z = 2, the probability of the adversary successfully reassembling the ledger drops dramatically to 52.0%.

This formulation provides the system engineer with a tool to balance the trade-off between operational resilience (liveness) and cryptographic risk mitigation based on nominal network conditions.
The numbers reported refer to a pessimistic estimate for a system with only 16 critical terminal fragments.
By increasing the number of critical fragments m, as already seen in the various stress tests 1-12 of the previous MTC report, it is possible to significantly reduce the probability of reconstructing the topological graph.

Critical Deadlock Resolution

In the extreme case in which a massive adversarial coalition paralyzes the system by repeatedly exceeding the redundancy limit of multiple aggregator nodes, interrupting transmission (a data withholding attack aimed at reducing liveness), the CNVS architecture strictly prioritizes security over availability.
If the routing flow is disrupted, the primordial node S^0 is never synthesized and the Global Veto (V_G) never emits the validation signal. The system enters a Fail-Closed state. The adversary can, at the cost of massive physical control of the network, temporarily paralyze the system, but is structurally unable to force acceptance of a false digital state.
This prolonged and absolute deadlock generates a Critical Deadlock Signal.
This signal represents the only architectural exception authorized to "wake up" the Zero Point Computer. Upon receiving the alarm, the origin does not attempt to retransmit data along the same corrupted paths, but instead acts as follows:

- Cancels the previous ephemeral instance.
- Performs a complete Topological Refresh.
- Generates a new reconstruction graph and reshuffles the cryptographic selectors.
- Initiates a Sybil Purge procedure, penalizing or actively excluding from the new topology terminals that have not responded to the collapsed instance's timeouts.

Note on the Digital Operating Domain: As detailed in the CNVS Technical Document, the deadlock resolution mechanics are structurally different in purely digital contexts. When the framework is implemented as a routing protocol (for example, in applications similar to TCP/IP), the terminal nodes do not correspond to physical verifiers, but to logical abstractions, and the entire graph reconstruction occurs directly on the destination machine. In this scenario, the absence of intermediate devices makes redundancy superfluous: timeouts (packet losses) are resolved in real time, bypassing Topological Refresh, through dynamic regeneration and micro-fragmentation of the affected branch.

Clarification on the engineering correlation between the Graph and the terminal Fragment

Absolute independence between relations (arcs) and fragments (leaves) is unrealistic.
This is because in reconstructing the S^0 state, the relationship between the fragments and those of the levels of subsequent nodes must necessarily respect the constraints of the logical structure of the raw data to avoid losing the meaning of the global data.
If the system is reconstructing the graph starting from the partial data referring to the length of the screws of a nuclear reactor, through multiple assignment tasks, it is clear that the observed and reported data must be correctly related within the logical matrix that replicates the structure of the raw data. In the reactor example, the lengths of the screws must necessarily relate to their relative positions, or to the width of the nut.
This semantic relational dependency does not affect the security of CNVS, because its security is not based on semantic topological independence, but on the unpredictability of task assignment, bounded leakage, and the preservation of min-entropy.
In the document defining the CNVS class, an entire section is dedicated to assignment randomness (section 7) and leakage (lemma 10).

Let:

$V_G(S(t)) = 1 \Leftrightarrow (\forall s \in Ter(t), \pi_1(V_L(s)) = 1) \wedge Cons_R(R(t), E(t)) \wedge Inv_C(S(t))$,
formula 18 in section 10 of Axiom III.

The relation R, which replicates the semantic graph, must necessarily express the logical constraints with respect to the semantic content of the structured element E at time t, because the $Cons_R$ function defines the consistency of these relations.
If there were no distributed logic under each Relation, the system would lose the geometric constraints that define the sense of internal coherence of the structure of the inserted partial data, even before the rules of invariant constraints are applied.
This would make the system more vulnerable, and the General Theory of Epistemic Dispersion would no longer be valid, because its validity stems from the assumption that the raw data possesses intrinsic geometric constraints.

Relational consistency therefore constitutes a distinct reconstructive check that logically precedes the evaluation of global invariants: first, it is verified that the state has been reconstructed in the correct geometry, and then that this geometrically consistent state satisfies the systemic rules.

Definition of Graph

The use of the word "graph" in computer science is often ambiguous.
In CNVS documents, the term can have multiple meanings.
The following clarification clarifies the confusion.

The term Semantic Graph, denoted by $G_S(t) = (E(t), R(t))$, refers to the complete logical structure formed by the elements of the instance and their required semantic dependencies. The relation R(t) constitutes the edge relation of the Semantic Graph; it is not synonymous with the entire graph.
The Routing and Convergence Graph G_R, on the other hand, represents the distributed realization generated after the random assignment of tasks and computational paths. G_R is unpredictable to the verifier and independent of the numerical values that will subsequently be observed. Its physical placement is therefore decoupled from the observed payload, although its materialized reconstruction paths must preserve the semantic dependencies defined by G_S.

The term Hidden Binding (B_t), on the other hand, refers to the logical and general connection existing between fragments, relational positions, parameters, and internal invariants.
Lemma 10 clarifies that functional dependence (expressed by the Relation R(t)) does not imply local and global epistemic transparency.

The global result depends on the measurements, but the verifier neither knows nor controls the complete way in which its measurement contributes to the reconstruction of the raw data.

So in CNVS, the logical and physical layers are represented by two distinct but convergent structures, the Semantic Graph (G_S) and the Routing Graph (G_R).
Convergence does not arise from synchronization or agreement between terminals, but from the geometry of structural dependencies.
The parameter t indicates the effective system time: verifiers can reestablish observations at different times, as long as they do so within the expected timeout. A branch is rebuilt when all necessary contributions are available.

Furthermore, the theoretical CNVS does not sample the declared data: the entire formalized instance, with all required tasks, relations, and invariants, is submitted for verification.

Semantic Graph

The Semantic Graph describes the logical geometry of the instance:

$$G_S(t) = (E(t), R(t)), \text{ where } R(t) \subseteq E(t) \times E(t).$$

It represents all the terminal tasks, intermediate nodes, and dependencies necessary for the reconstruction and application of $Cons_R$, Inv_G, and V_G.
Its structure necessarily depends on the type, unit, role, and context of the data, as well as on the complete family of invariants. A length measure, for example, can be related to the dimensions of geometrically compatible elements, but not to semantically extraneous properties.

The Semantic Graph is defined before observing the values:
$G_S(t) \prec Y(t)$.

Its geometry depends on the declared structure, but not on the value that will subsequently be measured.

Stochastic Assignment

The randomized assignment operator:
$\mathcal{A}_t$

associates terminal tasks with verifiers through an injective matching mechanism.
Before assignment, there is no predictable correspondence between a specific task and a specific verifier.
The assignment is therefore random and independent of future values:
$\Pr(\mathcal{A}_t = a \mid Y(t) = y) = \Pr(\mathcal{A}_t = a)$.

This phase ensures ex ante unpredictability and prevents a coalition from identifying critical nodes in advance.
However, randomness concerns the assignment and distribution of tasks, not the selection of a subset: all tasks belonging to the instance must be assigned and verified.

Routing Graph after Assignment

The Routing Graph:
$G_R(t) = (V_R(t), E_R(t))$

indicates:

- which verifier each task is assigned to;
- where each observation converges;
- through which intermediate nodes the reconstruction occurs;
- which contributions must merge to reconstruct the higher levels.

With the assignment completed, the stochastic configuration materializes into the deterministic one. From that moment, G_R can no longer be semantically independent: it must respect the geometry of G_S.

The reconstruction proceeds backward, from the leaves to the parent nodes:
$f_i \longrightarrow \text{parent}(f_i) \longrightarrow \cdots \longrightarrow S^{(0)}$.

The materialized graph therefore constitutes a distributed and hidden realization of the inverse reconstruction geometry:
$G_R(t) = \Phi\pi_t\,(G_S^{\uparrow}(t), B_t)$.

Randomization hides the position of the tasks and computational nodes, but cannot alter the necessary logical dependencies.

In summary:
the assignment is random, and convergence is semantically constrained after the assignment.

Binding B_t

B_t binding connects each task to semantically compatible positions in the Semantic Graph.
It can be hidden and randomly placed, but must preserve:

- the data type;
- the role of the measure;
- all mandatory relations;
- the family of invariants;
- the completeness of the reconstruction.

If the result of a task participates in multiple geometric or invariant relations, they must all be materialized. Randomness cannot transform complete verification into partial selection.
Binding can be hidden and randomized in placement, but must be semantically complete.
Distributed Reconstruction
The globality of V_G does not imply the existence of a single central server.

Validation is logically global, while computation can be distributed across the intermediate nodes of G_R, entrusted in a hidden manner to the devices of active and inactive verifiers, i.e., those who have received the task and those who have not.

Each verifier measures the assigned fragment and automatically contributes to the reconstruction operations envisaged by its path, both through its own observation contribution and through the computation of the hidden node, randomly temporarily assigned to its device.
However, without ever knowing the systemic significance of the contribution offered, without expressing any decision, and without synchronizing.
No votes are aggregated:
CNVS achieves a geometric convergence of evidence, not an aggregation of decisions.

Precommitment and Topological Refresh

The declaration, Semantic Graph, invariants, and binding must be determined or committed before the observations are known:
$\mathrm{Com}\,(D(t), C_{int}(t), G_S(t)) \prec (\mathcal{A}_t, \Pi_t, B_t, G_R(t)) \prec Y(t)$.

The system cannot ex post choose the relations or invariants that produce the desired outcome.
The configuration is valid only at time t.
Following a rejection or a Topological Refresh, the decomposition, assignment, binding, and Routing Graph are dissolved and regenerated:
$G_R(t+1) \neq G_R(t)$.

The old graph cannot therefore be used as a permanent map of the internal structure.

Summary

The Semantic Graph fully represents the logical geometry of the instance.
Random assignment distributes tasks in an unpredictable manner.
The materialized Routing Graph must then deterministically respect the semantic dependencies required for reconstruction.

The CNVS is therefore:
unpredictable ex ante, semantically constrained, and convergent ex post.

Special Section

Test 13 Objectives and Engineering Scope

Test Name: Engineering-Hardened 256-bit Fragmentation Sensitivity under Hidden Full-State Invariant Binding.

Filename = "test_13_engineering_hardened_256bit_fragmentation.py".

Test 13 preserves the core logical execution sequence established in previous validations (test 10 - 11: $V_L \rightarrow Cons_R \rightarrow Inv_C \rightarrow V_G$) but aggressively addresses prior engineering limitations. The primary objective is to evaluate whether fragmentation-driven decay of complete adversarial reconstruction remains observable within the implemented 256-bit, CSPRNG-driven computational environment.

To achieve this, the simulation environment was upgraded with the following mechanics:

- The system completely abandons NumPy finite-field states and deterministic seeds.
- The execution is performed using Python's arbitrary-precision integers within a standard 256-bit prime finite field ($p = 2^{256} - 2^{32} - 977$, equivalent to the secp256k1 field prime).
- All cryptographic randomness is exclusively generated via the secrets module (a Cryptographically Secure Pseudo-Random Number Generator, or CSPRNG).
- The simulator implements an adaptive malicious verifier subset per trial, which dynamically resamples the adversary's network position to simulate Sybil relocation and shifting peripheral compromise.

Execution Parameters and Run Data

The Test 13 Monte Carlo execution generated the following run parameters and high-resolution empirical data:

- **Run ID:** e023ca9a445df05d397b13c7d203d195.
- **Total Elapsed Time:** 43.26 seconds.
- **Field Prime Bits:** 256.
- **Peripheral Verifier Pool (Q_verifiers):** 2500.
- **Residual Inference Bound (h_min_bits):** 1.
- **Iterations per Point:** 100,000.
- **Adaptive Malicious Set:** True.
- **Critical Fragmentation Cardinality (m) values:** 1, 2, 3, 5, 8, 12, 16, 24, 32, 48, 64, 96, 128, 256, 512, 1024.

Empirical Analysis of Terminal Stress Scenarios

The implemented model was evaluated across multiple peripheral-compromise scenarios q, including configurations close to full collusion. Across the tested parameter grid, the observed complete-reconstruction rates closely align with the exact injective reference and remain below the independent analytical upper reference. The following scenarios illustrate the behavior of the model under high values of q.

Scenario q = 0.90 (90% Network Compromise)

- At a fragmentation depth of m = 128 and a terminal topology limit of k = 2048, the empirical V_G acceptance rate drops to 0.00121000.
- This empirical estimate is close to the exact injective reference of 0.00129674.
- Simultaneously, the Local-Pass/Global-Veto signal reaches 0.99879000. This means that 99.879% of the incorrect but locally admissible reconstruction candidates pass V_L and $Cons_R$ but are rejected by Inv_C and therefore by V_G.
- At m = 256, no complete authentic reconstruction event was observed across the 100,000 iterations, corresponding to the empirical resolution floor of the test.

Scenario q = 0.99 (99% Network Compromise)

- At this compromise level, only 1% of the verifier pool remains non-colluding. The scenario therefore evaluates the complete authentic-reconstruction event close to the full-collusion boundary.
- At the semantic warning limit of m = 1024 and k = 2048, the observed empirical probability of complete authentic reconstruction is 0.00334000.
- This execution mirrors the exact_ref mathematical expectation of 0.00321304.
- The analytical theorem reference formula ($[q + (1 - q) \cdot 2^{-h\_min}]^m$) bounds the performance from above at 0.00589976.
- At this boundary, the Local-Pass/Global-Veto signal reaches 0.99666000. This means that 99.666% of the incorrect but locally admissible reconstruction candidates pass V_L and $Cons_R$ but are rejected by Inv_C and therefore by V_G.

Epistemic Isolation and Systemic Control

A critical engineering parameter tracked during the simulation is the C_int_leak control flag.

- Across every single dimension evaluated, from q = 0.33 to q = 0.99 and all corresponding values of m, the C_int_leak column uniformly returns True.
- This programmatic disclosure control confirms the expected secrecy-collapse behavior of the reversible affine binding.
- It provides a direct executable control within the implemented model.

- If the hidden mathematical parameters are exposed, the execution engine will accept the reconstructed authentic state but reject the falsified state. Therefore, the near-zero acceptance rates observed in this ordinary model depend jointly on hidden binding, residual non-inferability, injective assignment, and fragmentation cardinality.

In Test 13, the affine binding is intentionally reversible:
$x_i = a_i^{-1}(tag_i - b_i) \pmod p$.
Cint's disclosure therefore allows us to recover the true critical values, including those originally assigned to honest verifiers.
But this leads to:
$X_{reconstructed} = X_{authentic}$, not to a false alternative state.
The code actually tests three distinct conditions:
reconstructs_truth = True,
$V_G(X)$ = ACCEPT,
$V_G(X')$ = VETO for a mutation $X' \neq X$.
If the false mutation is accepted, the program terminates with an error assertion.
And this did not happen when executing the code.
Acceptance of an incorrect non-authentic candidate would require an equivalence regime admitting at least one alternative state, whereas the binding instantiated in Test 13 is identifying.

```
CNVS Test 13: Engineering-Hardened 256-bit Fragmentation Sensitivity
------------------------------------------------------------
run_id = cc02c8e5b12d15074c4a7120922b0844
field_prime_bits = 256
Q_verifiers = 2500
terminal_fragments k = 2048 (fixed across all m)
h_min_bits = 1
p_inf = 0.5 (worst-case saturated bound)
iterations per point = 100000
structural V_G audits per point = 16
randomness = operating-system CSPRNG; no deterministic seed
coalition sizes = [0, 825, 1125, 1250, 1500, 1750, 1875, 2000, 2125, 2250, 2375, 2425, 2450, 2475, 2500]
m values = [1, 2, 3, 5, 8, 12, 16, 24, 32, 48, 64, 96, 128, 256, 512, 1024]

Measured event: complete reconstruction of every critical value, followed by acceptance of the authentic reconstructed state.
This is not a false-state-acceptance rate.


=== r=0/2500, q_actual=0.000000 ===
m | reconstruct | 95% CI | local-pass/veto | exact log10 | upper log10 | leak reconstructs | false leak state
   1 | 0.49888000 | [0.49578114, 0.50197895] | 0.50112000 |      -0.301 |      -0.301 | True | False
   2 | 0.24994000 | [0.24726606, 0.25263315] | 0.75006000 |      -0.602 |      -0.602 | True | False
   3 | 0.12413000 | [0.12210078, 0.12618810] | 0.87587000 |      -0.903 |      -0.903 | True | False
   5 | 0.03130000 | [0.03023864, 0.03239737] | 0.96870000 |      -1.505 |      -1.505 | True | False
   8 | 0.00372000 | [0.00336127, 0.00411686] | 0.99628000 |      -2.408 |      -2.408 | True | False
  12 | 0.00018000 | [0.00011387, 0.00028453] | 0.99982000 |      -3.612 |      -3.612 | True | False
  16 | 0.00001000 | [0.00000177, 0.00005665] | 0.99999000 |      -4.816 |      -4.816 | True | False
  24 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |      -7.225 |      -7.225 | True | False
  32 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |      -9.633 |      -9.633 | True | False
  48 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |     -14.449 |     -14.449 | True | False
  64 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |     -19.266 |     -19.266 | True | False
  96 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |     -28.899 |     -28.899 | True | False
 128 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |     -38.532 |     -38.532 | True | False
 256 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |     -77.064 |     -77.064 | True | False
 512 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |    -154.127 |    -154.127 | True | False
1024 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |    -308.255 |    -308.255 | True | False

=== r=825/2500, q_actual=0.330000 ===
m | reconstruct | 95% CI | local-pass/veto | exact log10 | upper log10 | leak reconstructs | false leak state
   1 | 0.66327000 | [0.66033468, 0.66619278] | 0.33673000 |      -0.177 |      -0.177 | True | False
   2 | 0.44292000 | [0.43984354, 0.44600085] | 0.55708000 |      -0.354 |      -0.354 | True | False
   3 | 0.29378000 | [0.29096485, 0.29661100] | 0.70622000 |      -0.532 |      -0.532 | True | False
   5 | 0.12959000 | [0.12752263, 0.13168583] | 0.87041000 |      -0.886 |      -0.886 | True | False
   8 | 0.03841000 | [0.03723648, 0.03961899] | 0.96159000 |      -1.418 |      -1.417 | True | False
  12 | 0.00734000 | [0.00682955, 0.00788830] | 0.99266000 |      -2.128 |      -2.126 | True | False
  16 | 0.00126000 | [0.00105846, 0.00149985] | 0.99874000 |      -2.837 |      -2.835 | True | False
  24 | 0.00003000 | [0.00001020, 0.00008821] | 0.99997000 |      -4.258 |      -4.252 | True | False
  32 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |      -5.680 |      -5.670 | True | False
  48 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |      -8.529 |      -8.505 | True | False
  64 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |     -11.383 |     -11.339 | True | False
  96 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |     -17.108 |     -17.009 | True | False
 128 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |     -22.855 |     -22.679 | True | False
 256 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |     -46.063 |     -45.358 | True | False
 512 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |     -93.523 |     -90.715 | True | False
1024 | 0.00000000 | [0.00000000, 0.00003841] | 1.00000000 |    -192.478 |    -181.431 | True | False
```

m	reconstruct	95% CI	local-pass/veto	exact log10	upper log10	leak reconstructs	false leak state
1	0.98968000	[0.98903454, 0.99028784]	0.01032000	-0.004	-0.004	True	False
2	0.97963000	[0.97873586, 0.98048729]	0.02037000	-0.009	-0.009	True	False
3	0.97132000	[0.97026728, 0.97233651]	0.02868000	-0.013	-0.013	True	False
5	0.95052000	[0.94915847, 0.95184692]	0.04948000	-0.022	-0.022	True	False
8	0.92284000	[0.92116982, 0.92447770]	0.07716000	-0.035	-0.035	True	False
12	0.88689000	[0.88491206, 0.88883822]	0.11311000	-0.052	-0.052	True	False
16	0.85154000	[0.84932278, 0.85373021]	0.14846000	-0.070	-0.070	True	False
24	0.78235000	[0.77978161, 0.78489670]	0.21765000	-0.105	-0.105	True	False
32	0.72612000	[0.72334739, 0.72887524]	0.27388000	-0.140	-0.140	True	False
48	0.61602000	[0.61300121, 0.61902988]	0.38398000	-0.210	-0.210	True	False
64	0.52298000	[0.51988348, 0.52607476]	0.47702000	-0.281	-0.279	True	False
96	0.37969000	[0.37668675, 0.38270249]	0.62031000	-0.423	-0.419	True	False
128	0.27352000	[0.27076591, 0.27629149]	0.72648000	-0.566	-0.559	True	False
256	0.07144000	[0.06986007, 0.07305285]	0.92856000	-1.147	-1.117	True	False
512	0.00433000	[0.00394164, 0.00475644]	0.99567000	-2.357	-2.235	True	False
1024	0.00001000	[0.00000177, 0.00005665]	0.99999000	-4.997	-4.470	True	False

=== r=2475/2500, q_actual=0.990000 ===

m	reconstruct	95% CI	local-pass/veto	exact log10	upper log10	leak reconstructs	false leak state
1	0.99487000	[0.99440781, 0.99529417]	0.00513000	-0.002	-0.002	True	False
2	0.98939000	[0.98873591, 0.99000649]	0.01061000	-0.004	-0.004	True	False
3	0.98470000	[0.98392041, 0.98544235]	0.01530000	-0.007	-0.007	True	False
5	0.97549000	[0.97451321, 0.97643026]	0.02451000	-0.011	-0.011	True	False
8	0.95985000	[0.95861551, 0.96104917]	0.04015000	-0.017	-0.017	True	False
12	0.94078000	[0.93930006, 0.94222608]	0.05922000	-0.026	-0.026	True	False
16	0.92324000	[0.92157374, 0.92487375]	0.07676000	-0.035	-0.035	True	False
24	0.88643000	[0.88444860, 0.88838171]	0.11357000	-0.052	-0.052	True	False
32	0.85335000	[0.85114386, 0.85552899]	0.14665000	-0.070	-0.070	True	False
48	0.78264000	[0.78007283, 0.78518546]	0.21736000	-0.105	-0.104	True	False
64	0.72591000	[0.72313674, 0.72866591]	0.27409000	-0.140	-0.139	True	False
96	0.61555000	[0.61253053, 0.61856059]	0.38445000	-0.211	-0.209	True	False
128	0.52281000	[0.51971343, 0.52590481]	0.47719000	-0.282	-0.279	True	False
256	0.26662000	[0.26388832, 0.26936961]	0.73338000	-0.572	-0.557	True	False
512	0.06638000	[0.06485365, 0.06793967]	0.93362000	-1.176	-1.115	True	False
1024	0.00309000	[0.00276457, 0.00345361]	0.99691000	-2.493	-2.229	True	False

=== r=2500/2500, q_actual=1.000000 ===

m	reconstruct	95% CI	local-pass/veto	exact log10	upper log10	leak reconstructs	false leak state
1	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
2	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
3	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
5	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
8	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
12	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
16	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
24	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
32	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
48	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
64	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
96	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
128	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
256	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
512	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False
1024	1.00000000	[0.99996159, 1.00000000]	0.00000000	0.000	0.000	True	False

In this report, only two log excerpts are reported.
For complete data, please consult the corresponding GitHub repository.

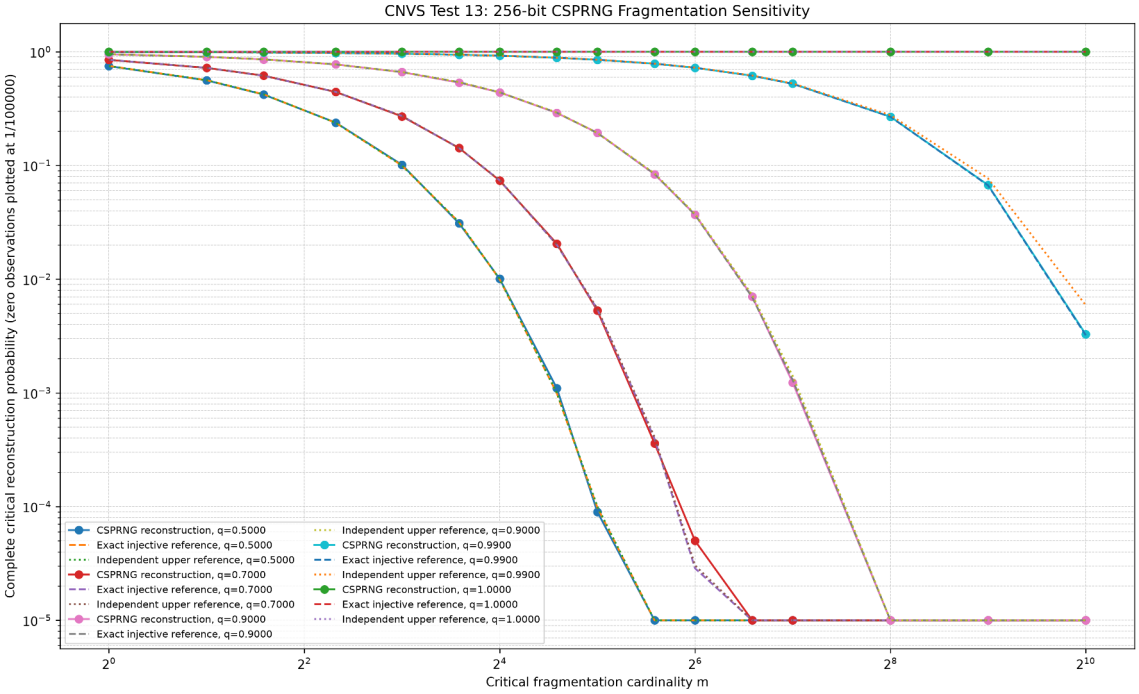
Detailed log analysis.

Analysis of the CNVS Engineering-Hardened Execution

The simulation logs provide empirical data regarding the system's performance under varying levels of peripheral compromise (q) and fragmentation depth (m). The following points summarize the operational dynamics of the framework based on the Monte Carlo execution:

- **Decoupling of Local and Global Validation:** The Local-Pass/Global-Veto signal quantifies the frequency at which incorrect but locally admissible reconstruction candidates pass V_L and $Cons_R$ but are rejected by Inv_C and therefore by V_G. For example, at q = 0.80, the observed rejection rate approaches 1 as m increases toward 128. This shows, within the implemented model, that local admissibility and relational completeness do not imply global acceptance.
- **Resilience to Adaptive Adversarial Sets:** The test implements an adaptive malicious set, modeling an adversary capable of dynamic relocation across the network at each iteration. Operating within a 256-bit cryptographic field, the empirical acceptance probabilities closely align with the theoretical exact injective reference curve (yielding an empirical probability of 0.00120000 against a theoretical 0.00129399 at q = 0.95 and m = 256).
  This observed agreement is consistent with the hypergeometric combinatorics of the implemented assignment model rather than with a fixed peripheral-node-position effect.
- **System Behavior at High Compromise Thresholds (q = 0.99):** At a 99% peripheral compromise level, the system restricts the unauthorized state reconstruction probability to 0.00334000 when the fragmentation cardinality m is scaled to 1024. This limitation arises because the adversary must correctly infer the remaining uncompromised fragments without external feedback. This mechanism effectively converts a network compromise vector into a probabilistic inference calculation under conditions of epistemic isolation.
- **Validation of the Hidden Invariant Dependency:** The C_int_leak check is true for all tested parameters. This metric serves as an **executable control check**, confirming that if the hidden mathematical parameters are explicitly provided to the execution engine, falsified states are rejected as non-authentic under the strict identification regime. This confirms that the low acceptance rates observed in the standard model are due to the structural secrecy of the geometry, the cardinality of the fragmentation, the preservation of min-entropy, the randomness of the assignment, and hidden binding. All of this allows the acceptance of reconstructed states when they are authentic.
- **Role of the Central Geometric Invariant:** Within the ordinary operational model, the central geometric invariant and its associated sub-invariants do not act as the sole

physical barrier preventing systemic failure. Rather, they function as the final layer of defense. Preceding barriers—namely, local syntax validation and topological consistency checks—actively filter non-compliant inputs before the ultimate geometric evaluation is invoked.



Visual Data Note (Decay of V_G Acceptance in a 256-bit Environment)

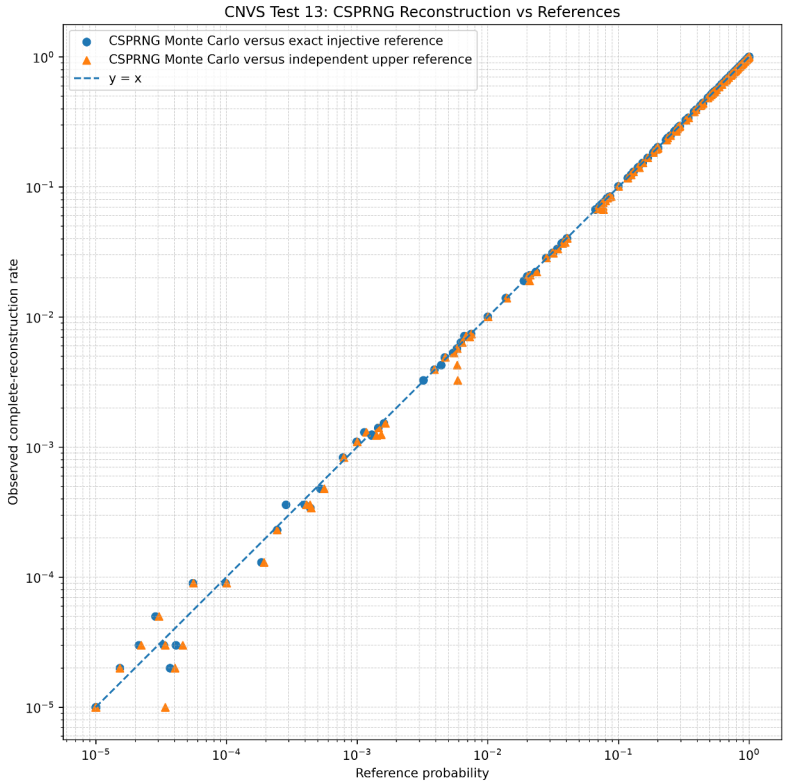
This log-log scale plot (logarithmic on both the X and Y axes) analyzes the empirical probability of unauthorized state reconstruction (V_G acceptance) as a function of the increasing critical fragmentation cardinality (m). The execution was conducted within an implemented 256-bit finite-field computational environment using CSPRNG-generated randomness.

Engineering Evidence

- **Structural Alignment:** The plot shows a high degree of alignment between the empirical simulator output and the exact injective reference calculated through the hypergeometric distribution. No material systematic deviation from the reference curve was observed across the tested parameter grid.
- **Response to Compromise Scenarios (q):** The curves track the framework's behavior under four distinct levels of peripheral compromise (q = 0.50, 0.70, 0.90, 0.99). It can be observed that for q values up to 0.90, increasing m while holding the other test parameters fixed drives the observed complete-reconstruction probability to the empirical resolution floor.
- **Upper Bound Management (q = 0.99):** In the extreme boundary scenario where the adversary controls 99% of the peripheral verifiers (light blue curve), scaling the

fragmentation toward the semantic warning limit (here arbitrarily set equal to m = 1024, and indicated by the vertical line on the right) still forces a decay in acceptance probability by nearly three orders of magnitude. In this extreme configuration, the analytical theorem reference (orange dotted line) correctly operates as a conservative upper bound.

- **Architectural Conclusion:** Within the implemented model, the probability of complete authentic reconstruction is parameter-dependent rather than governed by q alone. Increasing m while holding the other test parameters fixed reduces the observed reconstruction probability whenever q < 1 and residual non-inferability remains positive. This result concerns the tested reconstruction model and does not, by itself, establish the resilience of a deployed physical network.



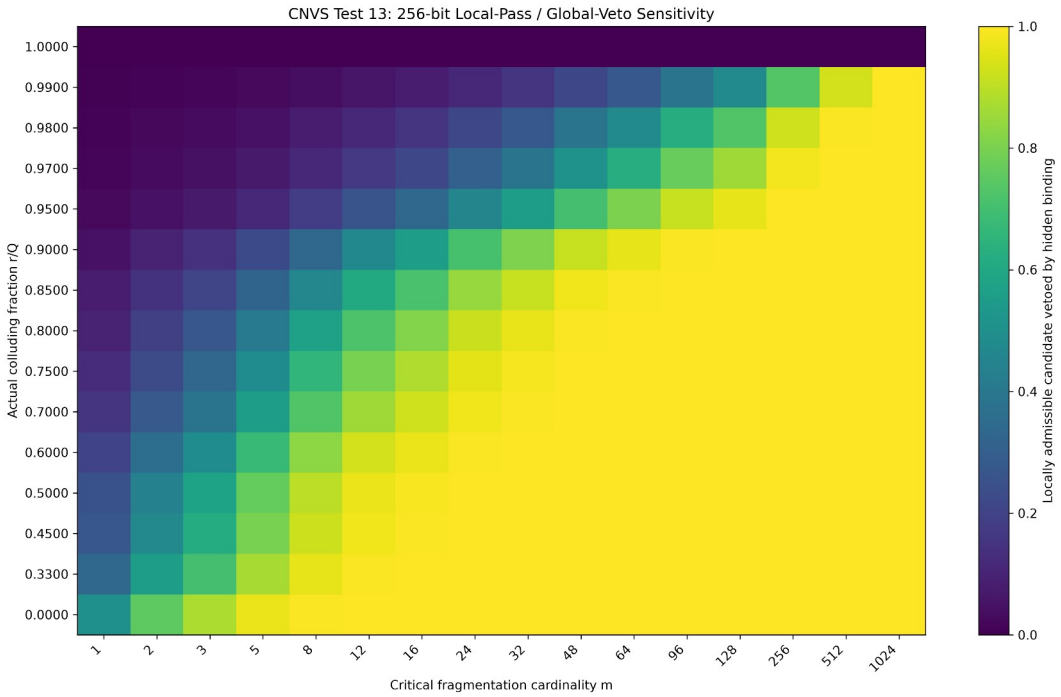
Visual Data Note (Empirical vs. Theoretical Alignment)

This scatter plot directly correlates the observed executable acceptance rates (V_G acceptance, Y-axis) against the theoretically predicted probabilities (X-axis) across all tested parameters in the 256-bit engineering environment.

Engineering Evidence

- **Hypergeometric Precision:** The blue circular markers, representing the empirical execution mapped to the exact injective reference, show significant alignment with the ideal diagonal line y = x over five orders of magnitude. This confirms that the software execution adheres to the discrete combinatorics of the hypergeometric distribution.

- **Conservative Upper Reference:** The orange triangular markers compare the empirical data with the independent analytical reference $[q + (1 - q) \cdot 2^{-h\_min}]^m$. Across the tested parameter grid, the empirical values remain on or below the corresponding analytical reference, consistently with its role as an upper bound under the assumptions of Theorem 4. The exact finite-pool injective probability is generally lower because task assignment occurs without replacement.
- **Statistical Resolution Limit:** The minor variance observed at the extreme lower boundary of the plot (approaching 10^{-5}) corresponds to the expected statistical noise floor, dictated by the fixed resolution limit of 100,000 Monte Carlo iterations per data point.

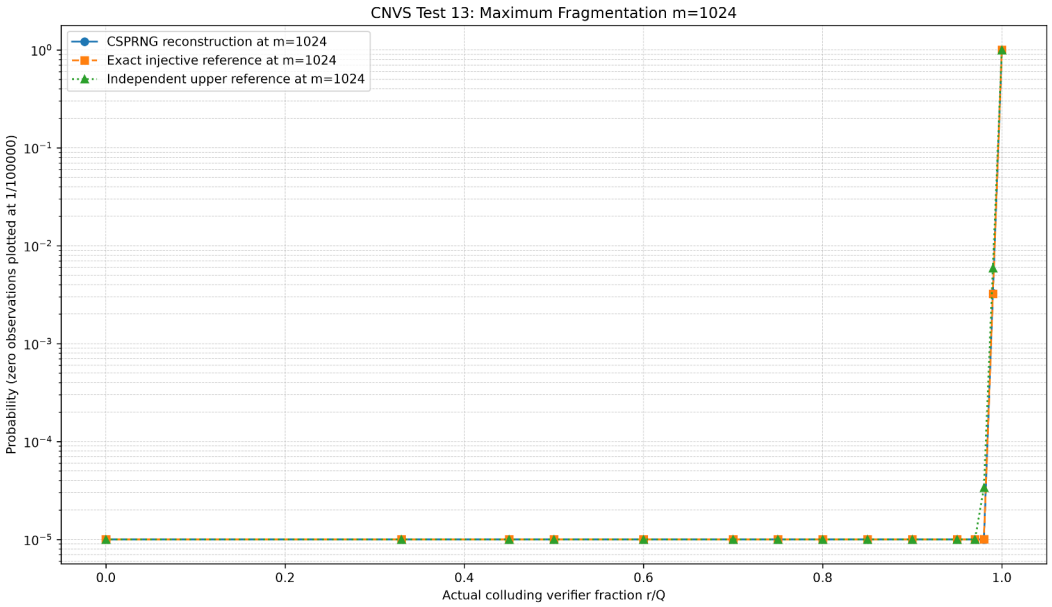


Visual Data Note (Engineering Non-Reducibility Heatmap)

This heatmap visualizes the "Local-pass / Global-Veto" rate across a matrix of peripheral compromise scenarios (q, Y-axis) and critical fragmentation cardinalities (m, X-axis). The color gradient, ranging from dark purple (0.0) to yellow (1.0), represents the frequency with which an incorrect but locally admissible reconstruction candidate bypasses local syntax and relational consistency checks (V_L and $Cons_R$) but is ultimately rejected by the central geometric invariant (V_G).

Engineering Evidence

- **Decoupling of Verification Layers:** The color transition toward 1.0 represents the increasing frequency of trajectories in which the candidate passes V_L and $Cons_R$ but fails Inv_C and is therefore rejected by V_G.
- **Scaling Against High Compromise:** At several tested parameter points, no complete authentic reconstruction event was observed, so the Local-Pass/Global-Veto rate reached the empirical value 1.0.
- **Architectural Validation:** The heatmap therefore shows that local admissibility and relational completeness do not imply global acceptance within the implemented identifying model.



Visual Data Note (Maximum Fragmentation Resilience vs. Compromise Ratio)

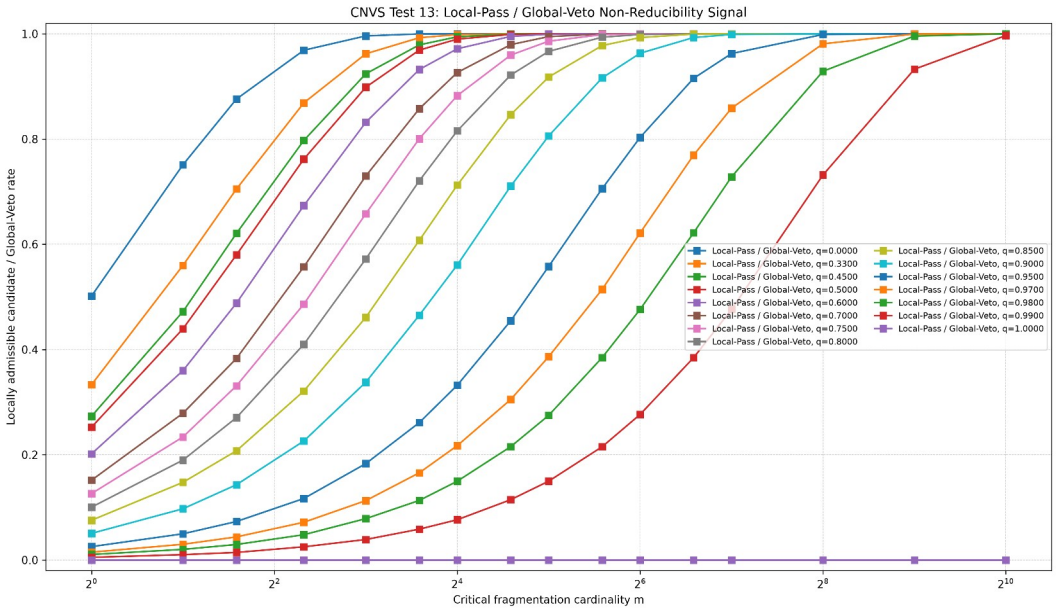
This semi-logarithmic plot isolates the behavior of the implemented model at the maximum tested critical fragmentation cardinality, m = 1024. The Y-axis represents the observed probability that the opposing coalition completely reconstructs all authentic critical values and presents the authentic reconstructed state to VG, while the X-axis represents the actual colluding-verifier fraction q.

Engineering Evidence

- **Low Observed Reconstruction Region:** For all tested compromise levels up to q = 0.97, no complete authentic-reconstruction event was observed across the 100,000

iterations. These points are therefore represented at the simulator's empirical resolution floor of 10^{-5}.

- **Observed Boundary Transition:** Non-zero complete-reconstruction events reappear in the tested grid under the extreme boundary conditions $q \geq 0.98$. At $q = 0.99$, the observed complete authentic-reconstruction probability is approximately 3×10^{-3}. This value describes the tested point at $m = 1024$ and should not be interpreted as an asymptotic lower limit.
- **Reference Hierarchy at Critical Limits:** As q approaches 1, the empirical execution remains closely aligned with the exact injective hypergeometric reference. The independent analytical upper reference remains above or coincides with the exact finite-pool value, consistently with the expected hierarchy of the two mathematical references under the implemented assumptions.



Visual Data Note (Non-Reducibility Signal and Global Veto Dynamics)

This semi-logarithmic plot illustrates the Local-Pass/Global-Veto rate as a function of the critical fragmentation cardinality m for the tested peripheral-compromise fractions q. The Y-axis represents the frequency of trajectories in which complete authentic reconstruction fails, while the resulting candidate remains locally admissible and relationally complete, passes V_L and $Cons_R$, but fails Inv_C and is therefore rejected by V_G.

Engineering Evidence

• **Global Veto Effectiveness:** When a curve reaches the empirical value 1.0, no complete authentic-reconstruction event was observed at that parameter point. Within the implemented identifying model, the corresponding incorrect but locally admissible candidates

follow the Local-Pass/Global-Veto path: V_L and $Cons_R$ pass, whereas Inv_C and V_G reject the candidate.

• **Rightward Shift under High Compromise:** As q increases, progressively larger values of m are required for the observed Local-Pass/Global-Veto rate to approach 1. This reflects the increasing probability that critical fragments are either directly controlled or correctly inferred by the coalition.

• **Convergence at Maximum Scaling:** For every tested condition with $q < 1$ and positive residual non-inferability, increasing m drives the observed Local-Pass/Global-Veto rate toward 1 within the tested range. The full-collusion boundary $q = 1$ remains structurally distinct, because all critical fragments are then directly controlled and authentic reconstruction succeeds with probability 1.

Analytical Conclusion of Test 13

Test 13 evaluates the behavior of the implemented CNVS reconstruction model under stressful parameter configurations within an engineering-hardened in-silico environment. Compared with the earlier simplified tests, this execution uses a 256-bit finite field, CSPRNG-generated randomness, adaptive resampling of the malicious verifier subset, and 100,000 Monte Carlo iterations per parameter point.

Empirical Alignment with the Hypergeometric Model and the Theorem 4 Upper Reference

The principal engineering result is the close alignment between the empirical complete-reconstruction rates and the exact injective reference calculated through the hypergeometric distribution. Across the tested parameter grid, the empirical results also remain below the independent analytical upper reference derived from $[q + (1 - q) \cdot p_{inf}]^m$.
In this execution, $h_{min} = 1$ is an assumed residual min-entropy bound, corresponding to $p_{inf} = 2^{-1}$. Under this assumption, the simulation reproduces the expected reduction in complete-reconstruction probability as m increases. The test does not independently estimate or validate the residual min-entropy of a deployed physical system.

Threshold-Based Fault Models in Classical Consensus

Many classical consensus protocols are analyzed through model-specific fault-tolerance thresholds. In traditional Byzantine agreement models, familiar resilience conditions often require at least $3f + 1$ participants to tolerate f Byzantine faults, while Nakamoto-style proof-of-work relies on the assumption that honest computational power remains dominant. These thresholds arise from different system models and should not be interpreted as a universal boundary applicable to every distributed architecture.

Test 13 evaluates a substantially different executable threat model. Within the implemented CNVS construction, the probability of complete critical reconstruction is not determined by a

standalone majority threshold, but by a parameter-dependent relation involving the colluding fraction q, the critical fragmentation cardinality m, and the residual inference probability p_{inf}.

Across the tested parameter grid, increasing m while preserving positive residual non-inferability reduced the observed probability of complete adversarial reconstruction, including under high values of q. The result does not invalidate classical BFT thresholds or establish general superiority over consensus-based protocols. It shows that the tested CNVS model follows a different conditional risk curve and therefore cannot be evaluated solely through conventional majority-control thresholds.

Final Engineering Verdict

Within the implemented model and across the tested parameter grid, Test 13 provides executable evidence that the probability of complete critical reconstruction closely follows the exact injective hypergeometric reference and remains below the independent analytical upper reference.

The observed behavior depends jointly on the colluding fraction q, critical fragmentation cardinality m, residual non-inferability p_inf, injective assignment, and hidden invariant binding. These results support the internal computational consistency of the tested CNVS construction, but do not constitute a universal security proof or a validation of a deployed distributed network.

Test 14 Objectives and Engineering Scope

Test Name: Test 14 - Full-Pipeline Semantic Execution and Progressive Validation of Invariants (End-to-End Environment).

Test 14 evaluates the computational execution of the CNVS architecture applied to a concrete, complex semantic instance belonging to a physical domain, but entirely fictitious.
The test therefore moves beyond individual numerical elements in a prime field and uses a complete graph composed of distinct semantic objects.

The test explicitly implements the complete information path for each candidate:
$V_L \longrightarrow Cons_R \longrightarrow Inv_C \longrightarrow V_G$,
using the following default parameters for the main simulation:
$Q = 1000$, $m = 64$, $h_{min} = 1.0$,
where Q is the active pool of terminal verifiers, m is the number of critical fragments selected for invariants, and hmin is the min-entropy margin.
The maximum budget for the main campaign is 500,000 iterations for each collusion level q.
The different execution modes, audit counts on the literal graph, and the distribution among independent replicates are specified in the following sections.

The Semantic Instance (Input)

Instead of abstract data, the test processes a canonical semantic input named "Enzo's House / Dr. Vincenzo Comitato." This catalog decomposes a physical environment into over 1,000 independent, indexable terminal fragments.
For example:
- Building geometry (50.00 m, 18.00 m).
- 120 total openings, including windows and French windows.
- 1,980 individual perimeter fence columns treated as distinct terminal objects.

Measurements and Parameters Explored

Unlike purely statistical tests, and as we have already seen for empirical tests, Test 14 does not use the theorem's formulas to determine acceptance, but objectively calculates the state of the graph.

The test explores:

1. **Collusion Cycle:** The fraction q increases from 0.20 to 1.00. Attackers produce "stealth" observations (hidden within local tolerances εa to exceed VL).
2. **Relational Attacks:** Parent node substitutions, semantic role substitutions, and injectivity violations are tested, evaluating ConsR's ability to reject the altered topology.
3. **Semantic Attacks:** The attacker manipulates the data of a local equation (e.g., the kitchen area) to test whether InvC vetoes the isolated branch.
4. **Progressive Leakage:** Revealing 0% to 100% of the rows of the hidden matrix Cint.
5. **Scalability:** Variation of the load Q from 5 to 1,000 verifiers.

Distinction between theoretical model and application coverage

In the pure theoretical model of CNVS, global validation is defined on the entire terminal set Ter(t), the entire relation R(t), and the entire family of invariants C formalized for the instance. An application implementation, however, can operate on a finite and engineered coverage of physically observable properties, especially when the number of potentially formalizable facts, measurements, and relations becomes very large.

In Test 14, the complete semantic catalog contains 13,182 distinct facts, while the main state instantiates 1,000 terminal fragments. The literal graph comprises the entire set of 1,000 active fragments and is traversed by the V_G function in dedicated audits.
The main Monte Carlo sweep instead runs 500,000 candidates for each level q using a vectorized logic pipeline on the critical surface of cardinality m = 64.
For each candidate, the local admissibility of the generated critical values and the validity of the 21 semantic equations and 64 hidden bindings of C_{int} are calculated. In the value collusion sweep alone, $Cons_R$ remains satisfied by construction because parents, roles, selectors, assignments, and task digests are not modified; the discriminative ability of $Cons_R$ is subjected separately to relational attacks.
In addition to the vectorized campaign, 1,000 candidates for each level q are transformed into LocalEvidence objects and subjected to the literal V_G function on the entire graph.
The corresponding agreement rate with the vector pipeline is recorded in the outputs. In the scalability study, the latencies reported in the graphs derive from 10,000 literal executions of V_G for each value of Q, while the 500,000 vectorized checks for Q constitute a separate computational check.

The theoretical CNVS model therefore remains the exhaustive reference, while in practical implementations, the selection policy, fragment coverage, critical cardinality m, and the family of implemented invariants constitute engineering parameters that must be explicitly documented, evaluated, and audited.

Cardinality of the Application Critical Surface

As we saw, in Test 14, the main semantic state comprises 1,000 active terminal fragments. The literal pipeline $V_L \rightarrow Cons_R \rightarrow Inv_C \rightarrow V_G$ requires the presence and local validity of the entire instantiated terminal set. The main Monte Carlo stress test, however, operates on a numerical critical surface of cardinality m = 64. These fragments are included in the hidden binding C_{int} and subjected to the probabilistic dynamics of direct control, correct inference, or no reconstruction. The remaining fragments of the active state maintain their authentic values during this specific sweep.
The choice of m = 64 does not necessarily identify the semantically most important fragments of the object and does not represent an exhaustive coverage of all its properties. It constitutes a configurable application cardinality, adopted to verify the behavior of the pipeline on a finite and semantically concrete critical surface. In engineering implementations of CNVS, the definition of the critical surface, fragment selection, invariant family, and coverage level must

be justified with respect to the application domain and audited. The exhaustive theoretical model remains the asymptotic reference, while the strength of the application certificate depends on the quality and discriminative power of the implemented formalization.

General Interpretation

The CNVS architecture is based on the decoupling of local observation and global consistency. The V_G function in the source code implements this separation as a rigid short-circuit pipeline, hierarchically defining the graph's validity conditions in **Python code**:

```
def V_G(state: CNVSSemanticState, evidence: Mapping[str, LocalEvidence]) -> ValidationResult:
  started = time.perf_counter_ns()

  all_local_ok = len(evidence) == len(state.leaves) and all(ev.local_admissible for ev in evidence.values())
  if not all_local_ok:
    return ValidationResult(
      accepted=False,
      # [...]
      early_rejection=True,
      failed_stage="V_L",
      reason="EARLY REJECTION: one or more terminal observations are missing, malformed, or outside epsilon.
Cons_R and Inv_C are not evaluated.",
      latency_ns=time.perf_counter_ns() - started,
    )

  cons_ok, cons_reason = check_cons_r(state, evidence)
  if not cons_ok:
    return ValidationResult(...)

  inv = evaluate_progressive_invariants(state, evidence)
  if not inv.ok:
    return ValidationResult(...)
  # [...]
```

Engineering Justification

The implementation uses Early Rejection at the VL barrier.
If a blatant colluder injects data that is manifestly nonsensical or outside the semantic tolerance ε_a, the framework does not waste clock cycles computing the matrix invariant or reconstructing the topology.
This reduces the cost of processing manifestly invalid inputs, limiting this specific computational exhaustion vector. Topological Rigidity and Relational Consistency ($Cons_R$) In the CNVS model, a relational attack alters parent-child connections, semantic roles, or attempts to reuse obsolete task digests.
The check_cons_r function acts as a validator of the graph's structural and relational consistency. It reconstructs parent-child relationships, verifies completeness, connection,

absence of cycles, injectivity of assignments, and matching of selectors, roles, and task bindings.
SHA-256 digests are recalculated to detect alterations from the instantiated state; they do not, by themselves, constitute digital signatures or proof of the external origin of the data.

```
for node_id, node in state.nodes.items():
    recomputed_digest = _node_digest(node.node_id, node.children_ids, node.node_kind, node.node_nonce)
    if recomputed_digest != node.topology_digest:
        return False, f"topology digest mismatch at {node_id}"
    if node.children_ids and len(node.children_ids) < 2:
        return False, f"unary internal node detected at {node_id}"
```

Engineering Justification

Enforcing len(node.children_ids) < 2 (prohibiting unary nodes) respects the theoretical constraint |D(s)| = 0 or |D(s)| ≥ 2 and avoids degenerate chains of single-child nodes.

Linear Vulnerability and Null Space (C_{int} Leakage)

To evaluate the framework's behavior in the event of partial or total secret compromise, Test 14 does not statistically simulate the leakage, but physically calculates the underlying linear algebra to provide the adversary with the null space of the compromised submatrix:

```
def _nullspace(matrix: np.ndarray, tolerance: float = 1e-11) -> np.ndarray:
    if matrix.size == 0:
        return np.eye(matrix.shape[1], dtype=np.float64)
    _, singular, vh = np.linalg.svd(matrix, full_matrices=True)
    rank = int(np.sum(singular > tolerance))
    return vh[rank:].T.copy()
```

Engineering Justification

Nullspace extraction via Singular Value Decomposition allows the attacker to generate Gaussian coefficients (rng.normal) to attempt to construct a false vector that satisfies the linear constraints outlined.
This implementation objectively demonstrates whether the adversary has the mathematical degrees of freedom necessary to bypass the validation function when $h_{min} > 0$.

Test 14 shows that the CNVS architecture can be instantiated as an executable software pipeline. In the implemented adversarial strategies, the decoupling between V_L, $Cons_R$, and Inv_C allows candidates not detected by local checks to be subsequently evaluated and, if incompatible with global invariants, rejected by the Global Veto.

The limits of this capability emerge at the high levels of compromise described in the following graphs. The prohibition of unary nodes prevents degenerate single-child chains.
In the Q = 5 … 1000 domain explored by Test 14, the observed latency increases approximately linearly with the number of active fragments.

Introduction to Graphical Analysis

Note: in this document, classical notation is replaced by Latex notation (using the "_" underscore) in the sections relating to graph comments and code.

This section introduces the analysis of the visual data extracted from the execution of Test 14. To facilitate the study of latency times, all data reported in this document are extracted from the assumption that the graph reconstruction by computational recomposition, progressive, bottom-up, and convergent, is managed on a single central device. The single-device execution therefore constitutes a local computational baseline.
A future distributed implementation could exploit parallelism, but would also introduce network and coordination costs; current results do not allow a direct determination of end-to-end latency.
Test 14 also models a system that develops within an identifying regime.

Semantic VG Acceptance vs. Exact and Theorem References

(Reference file: test_14_semantic_vg_vs_exact_and_theorem.jpg)

• **Axes and Domain:** Linear x-axis ($q \in [0.20, 1.00]$), logarithmic y-axis (P(success)). The lower bound of the graph is set to the simulation sensitivity (1 / 500,000, or $2 \times 10\text{-}6$).

• The graph plots the overall V_G acceptance probability on the logarithmic y-axis as a function of the fraction of compromised verifiers q on the x-axis (from 0.20 to 1.00).
Three curves are compared: the empirical output of the executable (blue circular markers), the exact hypergeometric reference (orange squares), and the compact bound of Theorem 4 (green triangles).

• The three curves show high levels of overlap.
In detail, the engineering implementation (blue line with circular markers) significantly overlaps with the theoretical bound curve (green triangles) and the exact injective reference (orange squares).
The empirical curve ("Executable semantic V_G acceptance", blue) is generated by the raw execution of the falsified candidates.
The strong alignment between the empirical curve and the analytical references indicates that, for the parameters and adversarial generation implemented, the executable output is consistent with the expected probabilistic behavior. This comparison does not constitute a general proof of code correctness, but it does not highlight any macroscopic discrepancies between the executable pipeline and the analytical references. In the reported run, the first observed successes emerge around q = 0.65;
this value represents an empirical breakout for the parameters used and not a universal safety threshold.

Full-Pipeline Barrier Outcomes across Collusion Levels

(File reference: test_14_local_pass_global_veto.png)

• **Axes and Domain:** Linear abscissas (q), linear ordinates (0.0 - 1.0, representing the normalized intervention rate for each logical barrier).

• The simulation performs stealth attacks, i.e., attack vectors generated specifically to deceive local controls: delta = unresolved * signs * alpha * state.hidden.epsilons with alpha ∈ [0.55, 0.95].
Due to this engineering choice, the rejection curves V_L (orange) and veto curves Cons_R (green) are flat at 0.0. The entire attack absorption is delegated to the Inv_C invariant (blue curve), which systematically rejects 100% of generated stealth adversarial states (quota 1.0) until combinatorial failure occurs above q = 0.85. The native architecture chain is imposed in the script: V_L acts as an observation convergence filter against noise (measurement errors), while Inv_C defends the system from organized collusion.

• This graph shows that, when the adversary performs a "stealth" attack—that is, providing tampered data that remains within the local tolerance margin ε_a — the V_L and Cons_R filters allow the data to pass (the orange and green lines remain at zero). However, the progressive Inv_C invariant intercepts the global anomaly and rejects the system with a 100% rate up to an extreme collusion level of q ≈ 0.85.
Global Veto (GV) intervenes where local inspection is blind.
The graph shows the effect of hierarchical fail-safe.
The architecture tolerates local infiltration of erroneous data (local-check bypass) because the network is designed to expose inconsistency globally.

• In the tested domain, increasing the collusion fraction does not cause significant performance degradation of the pipeline. This result relates exclusively to the adversarial inputs and the implemented collusion model and does not constitute a simulation of a network denial of service attack.

End-to-End Response Latency across Collusion Levels

(Reference file: test_14_latency_vs_q.jpg)

• This graph illustrates the execution times for the complete resolution of a single candidate along the literal pipeline as the collusion mass q varies, plotting the median (p^{50}) and upper percentiles (p^{95}, p^{99}) measured in milliseconds. In the explored domain, the observed latency remains substantially stable as q varies.
For q ≥ 0.25, the median remains approximately between 21 and 23 ms, while p^{99} remains in the observed range of approximately 35 – 45 ms.
Therefore, under the tested conditions, no significant performance degradation associated with increasing the collusion fraction emerges.

Validation Latency and Throughput across Semantic Scale

(Reference files: test_14_scalability_latency.jpg & test_14_scalability_throughput.png)

• The two graphs measure the impact on the system as the structural load increases, i.e., as the number of active verifiers/terminal semantic fragments (Q) increases, varying from 5 to 1,000.
Resolution latency and throughput (global candidates validated per second) are measured. The scalability stress test, conducted for Q = 5 … 1000, shows approximately linear growth in latency with respect to Q over the sampled range. Throughput decreases consistently with the increase in cost per candidate.
Test 14 does not implement a competing BFT benchmark; the comparison with consensus-based protocols therefore remains architectural in nature, not direct performance. In the sampled domain, the observed behavior shows regular and approximately predictable computational scaling with respect to Q. It remains to be verified in a large real-world implementation how imposing architectural constraints (for example, limiting the maximum routing tree depth N) influences this metric.

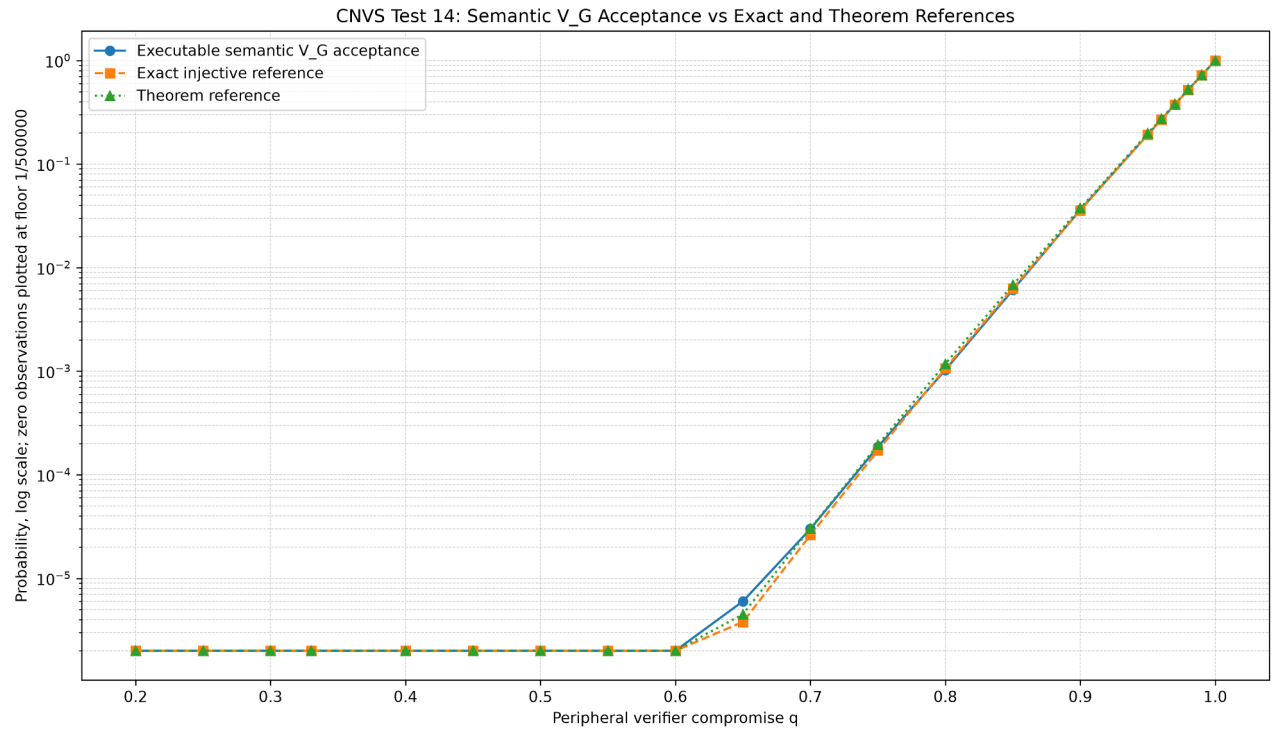
Progressive Hidden-Invariant Leakage

(File reference: test_14_invariant_leakage.png)

• The X-axis plots the fraction (from 0.0 to 1.0) of decrypted C_int equations. As long as the attacker does not possess the entire secret (Leakage ≤ 0.75), the residual rank of the matrix forces the vector nullspace into a region non-orthogonal to the tolerated vectors. Although the attacker computes feasible vectors (Feasible false-state construction at 1.0), these violate the constraints on the unrevealed portion of the matrix, resulting in a bypass of zero (False-state V_G bypass at 0.0).

• h = 0: The point of total disclosure represents a boundary condition in which the adversary possesses both the entire Cint matrix and the peripheral data needed for exact reconstruction. The attacker can no longer perform epsilon manipulations. Instead, using "Total-disclosure reconstruction," it performs an authentic reconstruction based on the decrypted data (the green curve instantly jumps from 0.0 to 1.0).
This implementation detail shows how the behavior of the CNVS framework mathematically depends on the residual min-entropy (for h > 0).

• The graph shows the effect of a significant compromise of epistemic isolation (Data Breach). It analyzes what happens when an increasing percentage (up to 100%) of the rows of the hidden matrix C_int are decrypted by the adversary.
Until 75% of the rows of the hidden matrix are compromised, the attacker manages to construct a false state (orange line at 1.0), but it is blocked (False-state bypass at 0.0 due to activation of the Global Veto in all conditions).
At 100% compromise, the null space collapses and the attacker no longer falsifies the state, but constructs a genuine one.
The green line rises only when there is a total loss of the secret ($h_{min} = 0$), a situation in which the adversary performs an authentic reconstruction due to the absence of unknowns, defining a boundary condition of the considered model.



This graph, filename "*test_14_semantic_vg_vs_exact_and_theorem_2.jpg*" is a visual representation of the behavior of the CNVS framework's mathematical effect as the network is progressively compromised.

Let's analyze it in detail.

Graph Structure and Axes

• **X-axis (abscissa):** Represents the variable q ("Peripheral verifier compromise"), i.e., the fraction of terminal verifiers controlled by the attacker. It ranges from 0.2 (20% of the compromised network) to 1.0 (100% of the compromised network).

• **Y-axis (ordinate):** Represents the attacker's probability of success, i.e., the observed probability that the coalition correctly reconstructs all authentic critical values and presents the reconstructed authentic state to V_G.

◦ **Logarithmic Scale:** The axis is a logarithmic scale (10^{-5}, 10^{-4}, …. , 10^{0}).
This allows us to visualize infinitesimal variations that would not be visible on a linear scale.

◦ **The Artificial Floor (Floor):** The Y-axis label specifies "zero observations plotted at floor

1/500,000".
Since on a logarithmic scale, absolute zero tends to negative infinity, simulations in which the attacker recorded zero successes out of 500,000 iterations are plotted on this artificial baseline (approximately 2×10^{-6}).
Comparing the Three Curves The graph overlays three distinct measurements to verify whether the code behaves exactly as the theory suggests:

1. **Blue line with circles (Executable semantic V_G acceptance):** This is the empirical data. It is the result of the actual execution of the Python code, which ran half a million forged candidates through the V_L, Cons_R, and Inv_C barriers.
2. **Orange dotted line with squares (Exact injective reference):** This is the calculation of the exact hypergeometric probability, according to the formula seen in the previous section relating to Test 13, which considers the injective (without replacement) assignment of fragments to verifiers.
3. **Green dotted line with triangles (Theorem reference):** This is the theoretical algebraic limit outlined by Theorem 4.

Threshold Analysis (with m = 64)

The graph illustrates the following situation:
• **Range from q = 0.20 to q = 0.60:** For the range q = 0.20…0.60, the executable recorded no successes in the 500,000 iterations considered for each level.
This observation does not imply a zero probability nor define a universal safety threshold, but rather exclusively describes the empirical behavior obtained under the implemented parameters and adversarial model.

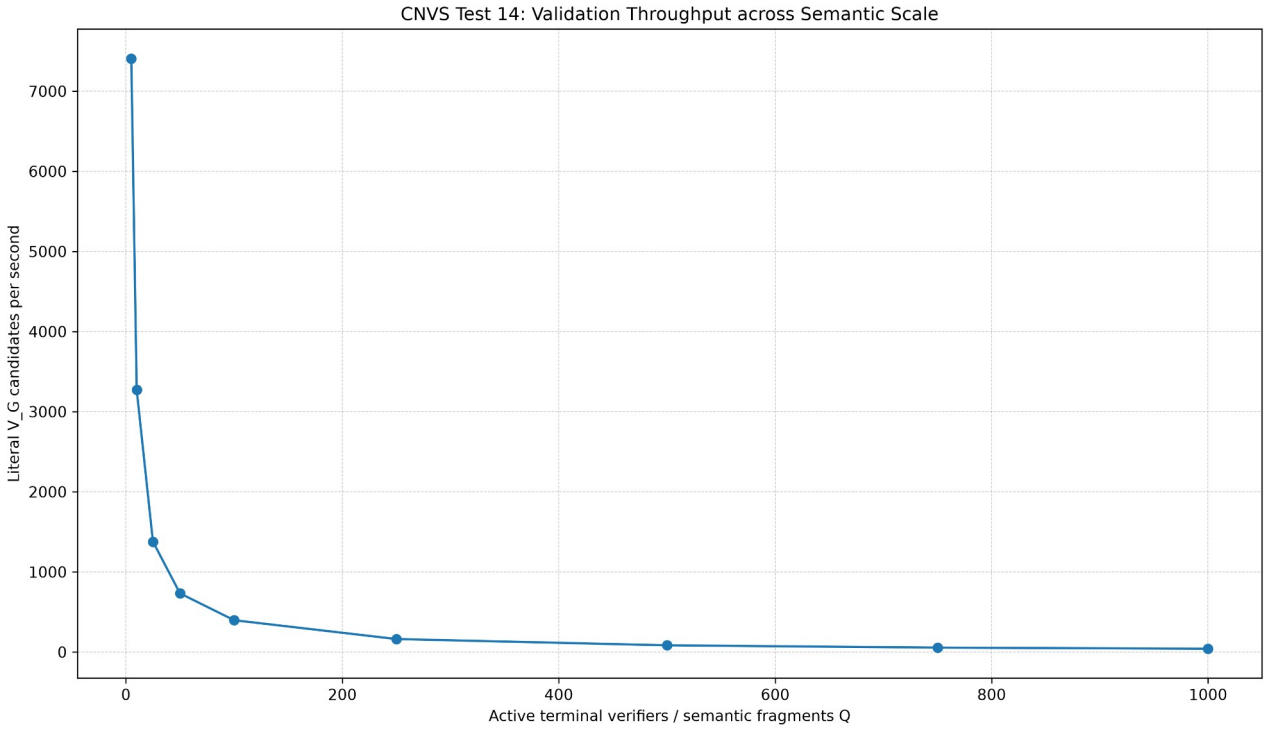
• **Breakout Point at q = 0.65:** A minimum break from the floor occurs at q = 0.65.
Here, the empirical probability (blue curve) records the first isolated successes (on the order of 3 or 4 accepted candidates out of half a million, equal to approximately $\approx 6 \times 10^{-6}$). In the interval q = 0.20…0.60, the executable recorded no successes in the 500,000 iterations considered per level.
This does not imply zero probability nor establish a universal safety threshold, but it describes the behavior observed for the test parameters. At q = 0.65, the first isolated successes appear, with an empirical frequency of the order of a few events out of 500,000 candidates.

• **From q = 0.70 to q = 1.00:** Beyond the empirical breakout observed around q = 0.65, in the observed range, the probability trend appears approximately linear on the logarithmic axis, behavior consistent with exponential growth in the sampled domain.
But at q = 0.80, the attacker's probability of success is still limited to 10^{-3} (1 in 1,000).
At q = 0.90, it rises towards 10^{-2}.

• **q = 1:** when the adversary controls all terminal verifiers in the implemented boundary case, the reconstruction probability equals 1.0 (100%).

Engineering Conclusions

The empirical curve (blue) plots the data generated by a code that processes real semantic objects, strings, ε-tolerances, and physical equations (such as the kitchen area or network topology), showing how this executable semantic simulation aligns with pure hypergeometric calculus (orange) and the upper bound of the theorem (green).



This graph, titled "*test_14_scalability_throughput_2.png*", represents the performance analysis of the CNVS framework in terms of throughput as the structural load (Q) increases.
While the latency graph measures "how long it takes the system to validate a single state," this graph measures the dual: "how many complete states the system can validate in one second" (Throughput).

Let's analyze the architecture's behavior.

Graph Structure

• **X-axis (abscissa):** Reports the number of active terminal verifiers or semantic fragments, denoted by Q.
The explored domain ranges from 5 to 1,000 terminal nodes.

• **Y-axis (ordinate):** Represents the Throughput, measured in "Literal V_G candidates per second".
It indicates how many complete pipelines (V_L ⟶ Cons_R ⟶ Inv_C ⟶ V_G) the algorithm can

successfully complete in one second.
Curve Analysis The curve traced by the blue markers describes a function of the type y = k / x (hyperbolic or harmonic decay). This shape is not random, but is determined by the mathematics of the test results.

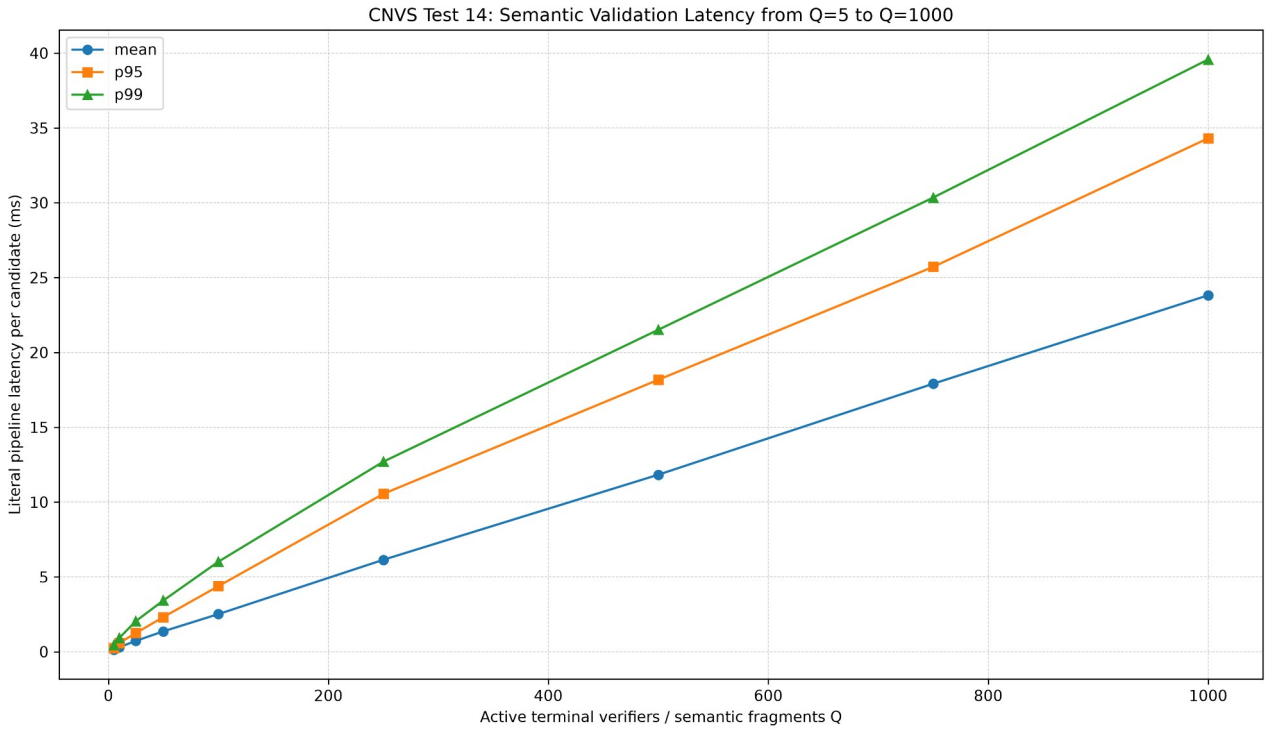
• **The High-Efficiency Regime (The "head" of the curve):** For small values of Q (e.g., Q = 5, Q = 10), throughput is massive, reaching a peak of over 7,000 validations per second.
In this domain, the graph depth and the number of fragments to be processed for the matrix invariant are minimal, allowing the evaluation engine to efficiently saturate the CPU clock cycles.

• **Inverse Throughput Behavior in the Sampled Domain:** Between Q = 50 and Q = 250, throughput undergoes a normal steep decline.
In the sampled domain, throughput exhibits a roughly inverse trend with respect to Q, consistent with the approximately linear growth in latency observed in the same interval. This behavior describes the measured empirical regime and does not, in itself, constitute an asymptotic demonstration of the architectural complexity. The fact that the graph respects this geometry demonstrates that the observed executions do not exhibit any macroscopic deviations from the expected trend attributable to collateral bottlenecks.

• **High load regime in the sampled domain:** From Q = 500 to Q = 1000, the curve flattens horizontally (the long tail on the right).
Throughput continues to decrease as Q increases, with a roughly inverse trend compared to latency; the slope is visually flattened towards the end of the domain. Comparison with Traditional Systems Many distributed protocols based on all-to-all communication can have communication costs that increase with the number of participants.
However, Test 14 does not implement a competing BFT protocol and therefore does not constitute a direct benchmark against such systems.
The graphs shown only describe the local computational cost of the CNVS pipeline in the tested domain.

Conclusions for Real-World Implementation

To maximize Validations/Candidates per second, the CNVS implementation could be configured so as not to require a single network with Q = 10,000.
Using the left side of this graph, a topological "hard limit" (e.g., Q_max = 100) could be set to keep throughput above 400-500 validations per second, scaling the system horizontally to create parallel clusters of independent CNVS networks operating in parallel.



This graph, named "*test_14_scalability_latency_2.jpg*", is the engineering complement to the previous throughput graph.
It measures the pure time cost required to solve the entire semantic tree (running the full V_L ⟶ Cons_R ⟶ Inv_C ⟶ V_G pipeline) as the system's overhead increases.
Here is the rigorous and detailed technical analysis of the graph.

Graph Structure

• **X-axis (abscissa):** The overhead, i.e., the number of active terminal verifiers or semantic fragments, denoted by Q.
The axis explores a scale-up from 5 to 1,000 nodes.

• **Y-axis (ordinate):** The latency of the literal pipeline per candidate, expressed in milliseconds (ms).

• **Plotted Curves:** Three key statistical metrics are plotted:

◦ Blue line (circles): The average latency (mean).

◦ Orange line (squares): The 95th percentile (p^{95}), or the time within which the system solves 95% of the computations.

◦ Green line (triangles): The 99th percentile (p^{99}), excluding 1% of extreme outliers.

Functional Analysis Linear Growth

The graph shows that the three curves scale approximately linearly.

• **Growth of O(N) Latency Time:** In the experimental range Q = 5 ... 1000, the observed scaling is consistent with approximately linear growth of latency times with respect to Q.

• **Efficiency of absolute values:** At Q = 1,000 (a huge network size for a single topological instance), the system takes an average of about 24 milliseconds to solve the entire logic chain. Even at the 99^{th} percentile, the worst-case latency is under 40 milliseconds.

Controlled Variance

An interesting technical detail is the "fork" between the blue curve (average) and the Green curve (p^{99}).

• In unoptimized or unstable systems, as the load Q and therefore the nodes N increase, 99% of observations would tend to deviate upward due to bottlenecks, garbage collection, or memory locks, dramatically departing from the mean.

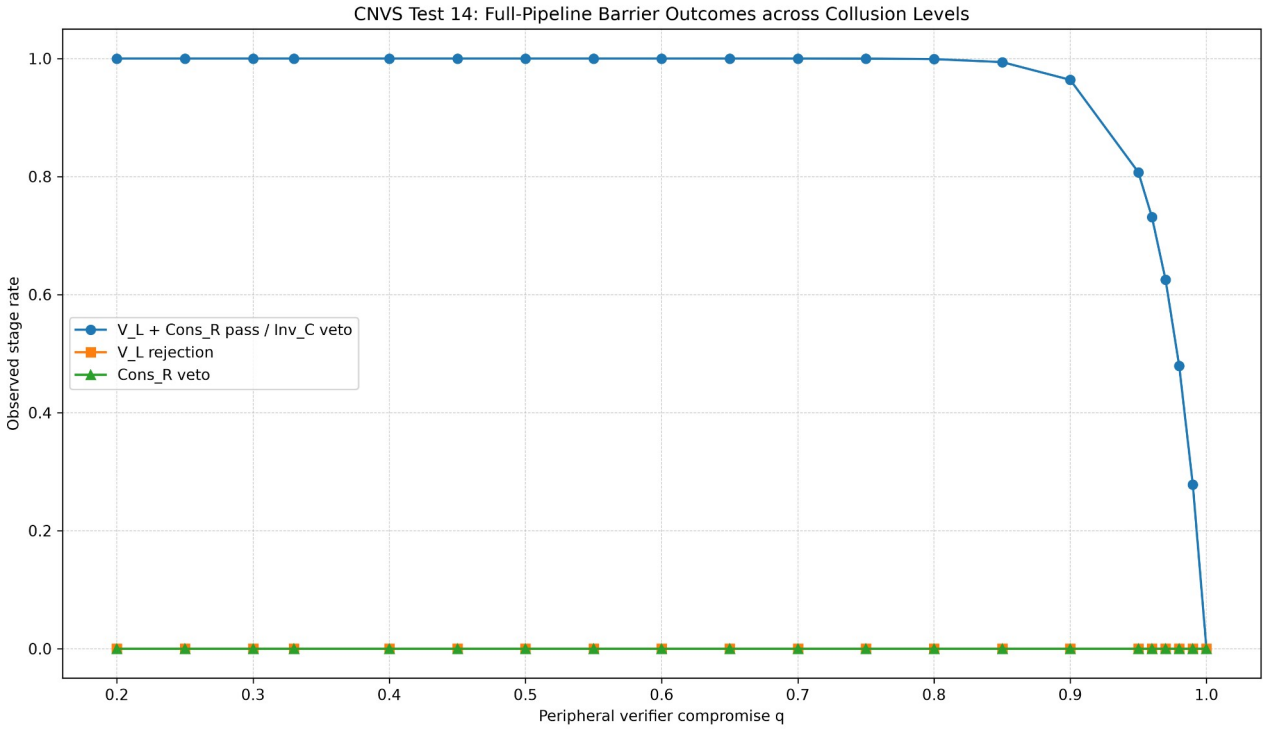
• In the graph shown, the "range" between the mean and p^{99} widens slightly but proportionally and consistently, maintaining a linear envelope.

This graphical detail illustrates the absence of internal congestion in the code:
In the observed run, no macroscopic deviations of the upper percentiles emerge from the general latency growth trend; however, the test does not allow the residual oscillations to be causally attributed to specific hardware or software components.

Architectural Implications

In the experimental range Q = 5 ... 1000, the observed curves are compatible with approximately linear latency growth with respect to Q.
Macroscopically superlinear growth does not emerge from the tested local computational baseline.

This result, however, concerns execution on a single device and does not determine The communication complexity or latency of a future distributed implementation.
At Q = 1000, the reported run shows an average latency of approximately 24 ms and a p^{99} of less than 40 ms; these values are empirical measures of the hardware and software environment used, not hard real-time bounds.
In a real implementation environment, this graph can be studied to size the depth of the hierarchical tree based on the available latency budget.
If the specifications require validation under 10 ms, depending on the available hardware, it is possible to impose a fixed topological limit (e.g., Q ≈ 200) to mathematically calculate the latency times for each individual instance, configuring the system to comply with the required limits.



This graph, titled "*test_14_local_pass_global_veto_2.png*", represents the breakdown of how the individual logical barriers (V_L ⟶ Cons_R ⟶ Inv_C ⟶ V_G) react when the network is subjected to a targeted and sophisticated attack.

While the global acceptance graph (V_G) shows the attacker's successes, this graph shows where and how the attacker is neutralized.
Let's analyze the graph.

Graph Structure and Domain

• **X-axis (X-axis):** Represents the compromised fraction of the edge network (q), starting from an 80% uninfiltrated ecosystem (q = 0.2) to total corruption (q = 1.0).

• **Y-axis (Y-axis):** Represents the "Observed stage rate" on a linear scale from 0.0 to 1.0. Measures the normalized frequency with which each specific validation function vetoed the computation, blocking the candidate.
The Base Curves: The "Stealth" Attack (Orange and Green Lines) The two flat lines at the bottom of the graph:

• **Orange Line (V_L rejection):** Flat at 0.0.

• **Green Line (Cons_R):** Flat at 0.0.

Data Analysis

If an attacker were to submit obviously absurd data (e.g., declaring that the area of a room is

negative or outside the tolerance ε), the system would immediately block it at the V_L level. This would only test the obvious.
In this simulation, the adversary performs a stealth attack:
it injects tampered data, but computed to stay within the allowed local tolerance range (ε) of each individual node.
Furthermore, it does not alter task digests, relational bindings, or the parent-child relationships of the graph.
Consequently, the local observation (V_L) and the relational topology (Cons_R) are not rejected.

The Ultimate Barrier (Blue Line)

The blue line with circular markers represents the intervention of the Progressive Hidden Invariant (Inv_C).
Since the first two defenses allowed the data to pass, in this model the survival of the network depends entirely on this component.

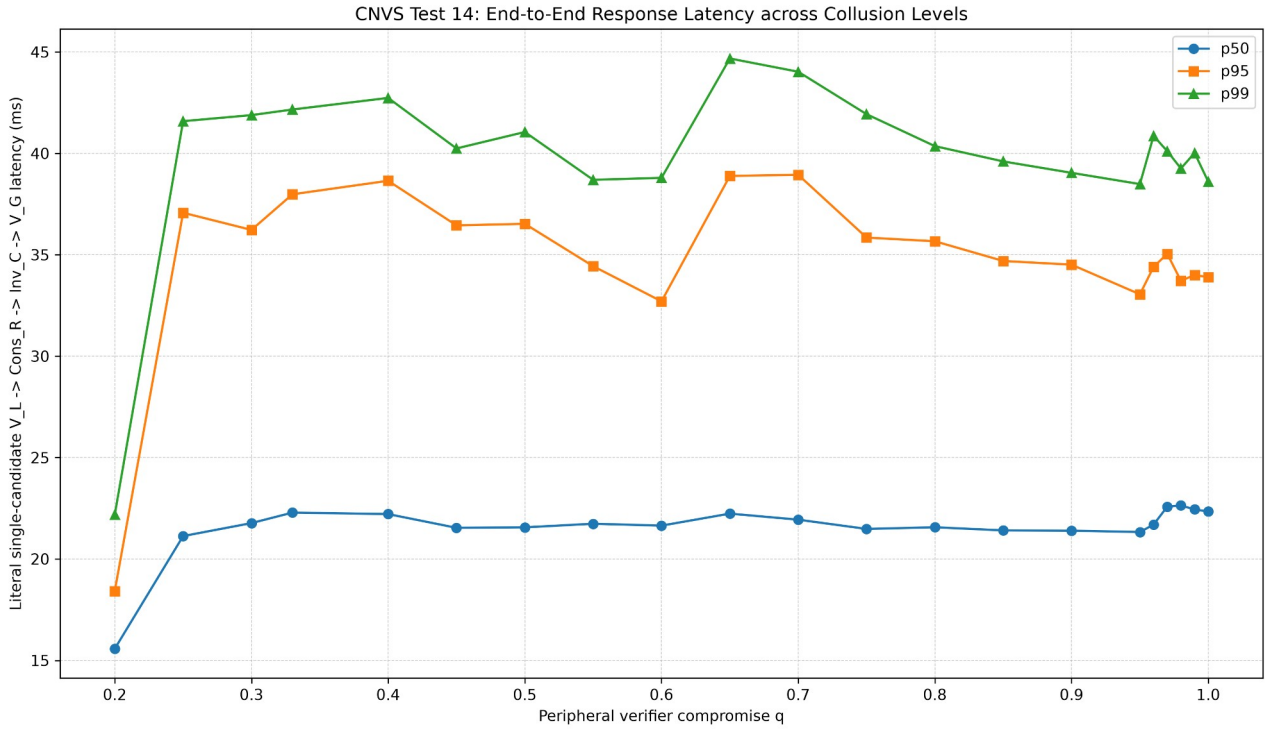
• **from q = 0.2 to q = 0.85:** Along this entire segment, the blue curve is flattened to 1.0: this means that the matrix invariant Inv_C still manages to intercept and discard 100% of the adversarial states, even though the observations transmitted by the colluder were correctly inserted within the allowed ε range.
This means that even in the face of locally consistent data (because V_L has validated them), when Inv_C globally cross-references them through the constraints of the critical fragments, it detects the algebraic inconsistency and imposes a veto.
This effect is explained by the fact that if the observations deviate excessively from the expected values (d_i), even if they are within the allowed range, the progressive reconstruction highlights the inconsistency from the convergence of the deviations, thus revealing the statistical error as a violation of the internal parameters.

• **from q = 0.90 to q = 1.00:** Starting from q ≈ 0.90, in the specific execution observed, the blue curve begins to decrease.
The attacker now possesses enough data to satisfy the null space of the matrix.
He can compute linear combinations that even fool the global control.
At q = 1.0, in the adversary model considered, the Inv_C intervention rate drops to zero, since the adversary has control or the reconstruction of the values required by the implemented strategy.

Architectural Conclusions (Decoupling)

This graph visually demonstrates the principle of Topological Decoupling of the CNVS framework.
In distributed consensus-based protocols, global validation requires agreement procedures between nodes.
In the tested CNVS model, terminal verifiers are semantically blind to the global outcome, and consistency is assessed through the structural pipeline.

By isolating global semantic verification in Inv_C, a local infiltration capable of bypassing V_L and Cons_R can still be intercepted globally.
In the observed iterations, Inv_C rejected 100% of adversarial states up to q =0.85; beyond this range, the veto rate progressively decreases.



This graph, titled "*test_14_latency_vs_q.png*", analyzes a critical parameter for the operational security of network infrastructures: performance resilience under increasing collusion.

While the scalability graph demonstrates how the system reacts to increasing network size (Q), this graph demonstrates how the system reacts to increasing internal corruption (q) while holding network size constant.

The test does not simulate a full network DoS attack: it measures the pipeline's temporal response to increasing collusion fraction q and the injection of adversarial inputs defined by the model.
Here is the detailed technical analysis.

Graph Structure and Metrics

• **X-axis (X-axis):** The fraction of the compromised network (q), which explores the intensity of the volumetric/stealth attack from a low level (0.2) to full compromise (1.0).
• **Y-axis (Ordinate):** The end-to-end validation latency for a single candidate, measured in milliseconds (ms).
This represents the time cost to execute the entire literal logic pipeline (V_L ⟶ Cons_R ⟶ Inv_C ⟶ V_G).

• **Statistical Distribution (Curves):**
◦ Blue line (circles): Median (p^{50}), the standard resolution time.
◦ Orange line (squares): 95th percentile (p^{95}).
◦ Green line (triangles): p^{99}, the value within which 99% of the measurements fall.

Latency Stability with Respect to Collusion and Adversarial Inputs

The graph shows the transverse flatness of the curves.
• **Latency with Respect to Collusion:** Starting at q = 0.25, the median latency (blue curve) stabilizes rigidly in a very narrow channel between 21 and 23 ms.
No significant correlation is found between the number of fragments locally altered by the adversary and the time the system takes to process them.

• In the tested computational model, the cost of candidate evaluation remains essentially stable as q increases.
This result concerns only the local pipeline and the implemented adversarial inputs;
it does not constitute a direct performance comparison with BFT protocols or an assessment of resilience to resource exhaustion or denial-of-service in a real-world distributed environment.

Analysis of Variance

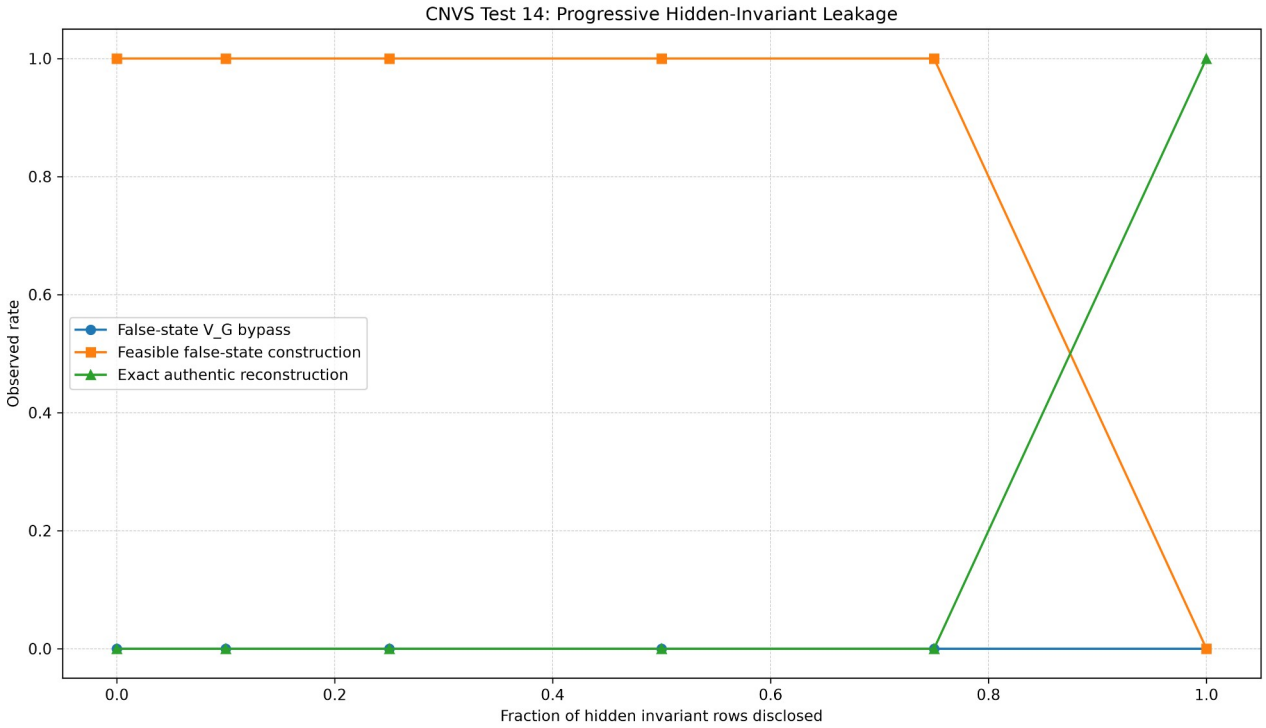
The upper curves (p^{95} and p^{99}) describe the system's behavior under stress.
• **Computational Jitter:** The green curve (p^{99}) fluctuates between 38 ms and 45 ms (reaching an isolated peak at q = 0.65).
These oscillations are compatible with execution-environment jitter; the test does not causally isolate their origin.
• **Hard Bound (ceiling):** In the measured executions, p^{99} did not exceed approximately 45 ms;
This value is an empirical measure of the run and does not constitute a hard real-time bound.
Variation of the observed latency regime
At q = 0.2, all curves record a significantly lower value ($p^{50} \approx 15.5$ ms, $p^{99} \approx 22$ ms).
This point represents the low-perturbation state.

When the vast majority of the network is honest, semantic deviations introduced into the matrix are small. As soon as collusion exceeds 25% (q = 0.25), the system enters full throttle in processing the out-of-tolerance residues for the infected branches, triggering a computational leap of a few milliseconds.
Once this massive computation regime is reached, the validation cost saturates and stabilizes horizontally until 100% compromise.

Conclusions

In the explored domain, the observed latency remains essentially stable with respect to the variation of q after the initial jump between q = 0.20 and q = 0.25.
The oscillations of p^{95} and p^{99} are consistent with jitter in the execution environment, but the test does not causally isolate their origin.

The results show that, for the implemented adversarial input class, increasing collusion does not produce a systematic degradation of the time cost per candidate.
Resilience to a network DoS attack is not evaluated here.



This graph, titled "*test_14_invariant_leakage_2.png*", illustrates the worst-case security scenario: the progressive compromise of the hidden invariant matrix (White-box attack / Data Breach).
The graph analyzes what happens mathematically to the system when the attacker manages to decrypt or steal, row by row, all the invariants (C_int).
The following is a technical analysis of the "Null Space Dynamics".

Graph Structure

• X-axis (abscissa): Represents the fraction of the hidden invariant rows that have been leaked by the adversary (from 0.0 to 1.0, or 100% of the exposed C_int matrix).

• Y-axis (ordinate): Observed rate (from 0.0 to 1.0) of attack outcomes.

Orange Curve

The orange line with squares ("Feasible false-state construction") is fixed at 1.0 (100% success) until the compromise reaches 75%, then collapses to zero at 100%. When the attacker steals a fraction of the matrix (e.g., 50% or 75%), he uses Singular Value Decomposition (SVD) to compute the null space of the compromised submatrix.

This step is explicitly stated in the CNVS code:

```
def _nullspace(matrix: np.ndarray, tolerance: float = 1e-11) -> np.ndarray:
        # ... compute SVD to extract the matrix kernel
```

Until the matrix is 100% revealed, the stolen submatrix does not have full rank. Therefore, its nullspace has dimension > 0.
The attacker then generates a perturbation vector (Delta) along the basis vectors of this nullspace. By adding this Delta vector to the original data, he obtains a "false state" that perfectly solves all the equations he managed to steal. The attacker believes he has produced a valid packet (the curve is 1.0).

Blue Curve

The blue line with circles ("False-state V_G bypass") is a flat line, stably squashed at 0.0 for the entire domain. Although the attacker has constructed a false state that passes the exfiltrated or inferred equations, when he sends this state to the validator, the system still possesses the undisclosed fraction of the matrix. Since the complete C_{int} matrix is constructed to have full column rank with respect to the m critical variables in the identifying regime, the perturbation generated from the null space of the exposed submatrix will generally fail at least one unrevealed constraint.

When the system executes:

```
def evaluate_hidden_invariants_batch(state: CNVSSemanticState, critical_values: np.ndarray) ->
np.ndarray:
    # ...
    residuals = normalized @ state.hidden.matrix.T - state.hidden.targets[None, :]
```

The residuals for rows unknown to the attacker exceed the tolerance thresholds, and the system blocks the bypass (0.0). In the identifying regime and under the adversarial strategy implemented in Test 14, partial knowledge of C_int did not produce any observed false-state bypass.
Knowing or inferring 75% of the rows of C_int does not confer a 75% probability of overriding the system: the graph shows that attempts to force the system to accept a reconstructed false state fail due to the Global Veto, despite the massive exfiltration of C_int.

Green Curve

The green curve illustrates what happens when the hidden matrix Cint is fully disclosed:

1. **The orange curve collapses to zero:** Since the matrix C_int is now fully disclosed and has full rank, the null space collapses (dimension 0). There are no longer any non-zero Delta vectors that explore a non-authentic space and satisfy the homogeneous system at the same time. In the identifying model considered, there is no non-zero direction left in the null space capable of simultaneously preserving the entire implemented

system of linear constraints. Forcing with non-authentic states therefore becomes impossible, according to the identifying regime model.

2. **The green curve ("Exact authentic reconstruction")** jumps from zero to 1.0: In this scenario (and only in this scenario), the attacker no longer attempts to insert false states, which have become impossible (orange curve). But knowing both the entire C_int matrix and all the decrypted peripheral data, over which it has full control either by random assignment or by inference from the uncontrolled data, the adversary solves the deterministic system to trace back to the original exact data, although it cannot falsify them (the system is protected from infiltration by false states due to the C_int equations, but since the adversary coalition has all the data, it can violate the privacy of the entire content, although it cannot alter it without triggering the global veto).

Graph Conclusions

This graph plots the limiting condition for eliminating residual uncertainty in the considered model.
If the adversary obtains both the complete validation algorithm, the hidden structural information required by the model, and the necessary fragments in the clear, the min-entropy of the system is zero ($h_{min} = 0$).
At that point, the framework simply performs an "exact authentic reconstruction." The system has not been "tricked," it has simply been "read and reconstructed."
This graph illustrates the conditions under which the validity of the CNVS framework operates.

Report Conclusion

CNVS explores a global verification approach distinct from consensus-based protocols: in the tested model, global consistency derives from structural compatibility and the evaluation of invariants, without requiring a voting procedure among the terminal verifiers. Stress testing campaigns show that, under the stated assumptions and parameters, this approach can be implemented and quantitatively evaluated in silico.

Test 13

Test 13 establishes the quantitative basis for the subsequent engineering analysis.
Within the tested parameter domain, it shows that increasing m, while maintaining positive residual non-inferability, progressively shifts substantial complete-reconstruction probability toward higher values of q.

The observed behavior is consistent with the bounds of Theorem 4 and with the exact injective hypergeometric reference.

Test 14

Test 14 abandons abstract numerical vectors to process real, interdependent semantic fragments (the "Enzo's House" instance with over 1,000 terminal objects), testing the entire

validation chain:
V_L ⟶ Cons_R ⟶ Inv_C ⟶ V_G.
The disaggregated analysis revealed the functioning of the hierarchical Fail-Safe.
It turns out that the network can allow vulnerable edge nodes: stealth attacks that manage to bypass the local convergence check (V_L) and relational constraints (Cons_R) are still blocked by the global progressive invariant (Inv_C).
Test 14 also simulates a progressive compromise of the hidden structural knowledge Cint. In the sampled partial disclosure levels and under the implemented adversarial strategy, no false-state bypass through the null space of the exposed submatrix was observed.

In the limiting condition of total disclosure, with the entire C_int and the necessary edge data available, the model instead switches to exact authentic reconstruction.

Topological collusion models with compromised verifiers that exfiltrate part of the global knowledge instead follow separate probabilistic dynamics, evaluated based on the parameter q.

Computational Scalability in the Tested Domain

In the local computational baseline used in Test 14, the measured latency shows approximately linear growth with respect to the topological load in the domain Q=5…1000. Since execution is entrusted to a single device, these results describe the local computational cost of the pipeline and do not allow us to directly infer the communication complexity or end-to-end latency of a future distributed implementation.

Final Conclusion

The CNVS framework approaches the validation of a distributed state from a perspective distinct from paradigms based on agreement between nodes: determining global validity is formulated as a structural, relational, and invariant compatibility problem, closed and deterministic in the model considered.

Tests 13 and 14 show that, under the stated assumptions, the CNVS architecture can be instantiated as an executable software pipeline without resorting to a voting or consensus procedure among the end verifiers.

Test 14 extends the analysis from abstract numerical states to a semantic representation of a physical domain and verifies the behavior of the V_L→Cons_R→Inv_C→V_G pipeline, the response to implemented adversarial strategies, Cint's progressive leakage, and computational scaling in the Q = 5 … 1000 domain.

The results are valid for the declared adversarial model, critical surface, invariants, and execution environment; they do not constitute proof of universal security or validation on a real distributed network. However, they constitute in silico engineering evidence of the feasibility of the architecture under the tested conditions and provide the experimental basis for subsequent stress tests in a real network environment.

Execution Environment and Reproducibility

To document the execution environment and promote the reproducibility of the Monte Carlo simulations, the computational execution was conducted and tracked in the following environment:

• **Hardware and OS:** Windows 11 HOME 25H2, OS 26200.8973, 12th Gen Intel(R) Core(TM) i9-12900H processor, 32 GB RAM.

• **Computational Stack:** Execution in Python 3.14.6, supported by the NumPy 2.5.1 vector library.

• **Stochastic Initialization (RNG):** Test 14 uses a master seed from which independent pseudo-random replicas are derived via numpy.random.SeedSequence.
The master seed of the execution is recorded in the test_14_full_pipeline_manifest.json file, while the seeds of the individual replicates are stored in the corresponding *_*seed_replicates.csv files*.
This recording allows you to reproduce the pseudo-random sequences used in the reported execution.

• **Source Versioning:** The test suite references the scripts: test_13_engineering_hardened_256bit_fragmentation.py test_14_semantic_end_to_end_cnvs_full_pipeline_corrected.py test_15_cnvs_m_sensitivity_full_pipeline

• **Primary Outputs:** The raw data from Test 14's topology and epistemic dispersion values are consolidated into the files:

test_14_collusion_full_pipeline_summary.csv, test_14_semantic_full_pipeline_summary.csv, test_14_relational_full_pipeline_summary.csv, test_14_leakage_full_pipeline_summary.csv, test_14_refresh_stress_summary.csv, and test_14_scalability_full_pipeline_summary.csv. The execution configuration, including the master seed, is recorded in test_14_full_pipeline_manifest.json. The main views are test_14_semantic_vg_vs_exact_and_theorem.png, test_14_local_pass_global_veto.png, test_14_latency_vs_q.png, test_14_invariant_leakage.png, test_14_scalability_latency.png, and test_14_scalability_throughput.png.

Test 15 - Executable Validation and Direct m-Sensitivity Report

Introduction

Test 15 extends the executable framework of Test 14 by introducing a direct scan of the cardinality of the critical surface,

$$m \in \{32, 64, 128, 256, 512\}.$$

The value m = 64, used as the main configuration in Test 14, therefore constitutes the internal reference against which lower and progressively higher values of m are evaluated.

The campaign maintains the semantic framework and validation pipeline

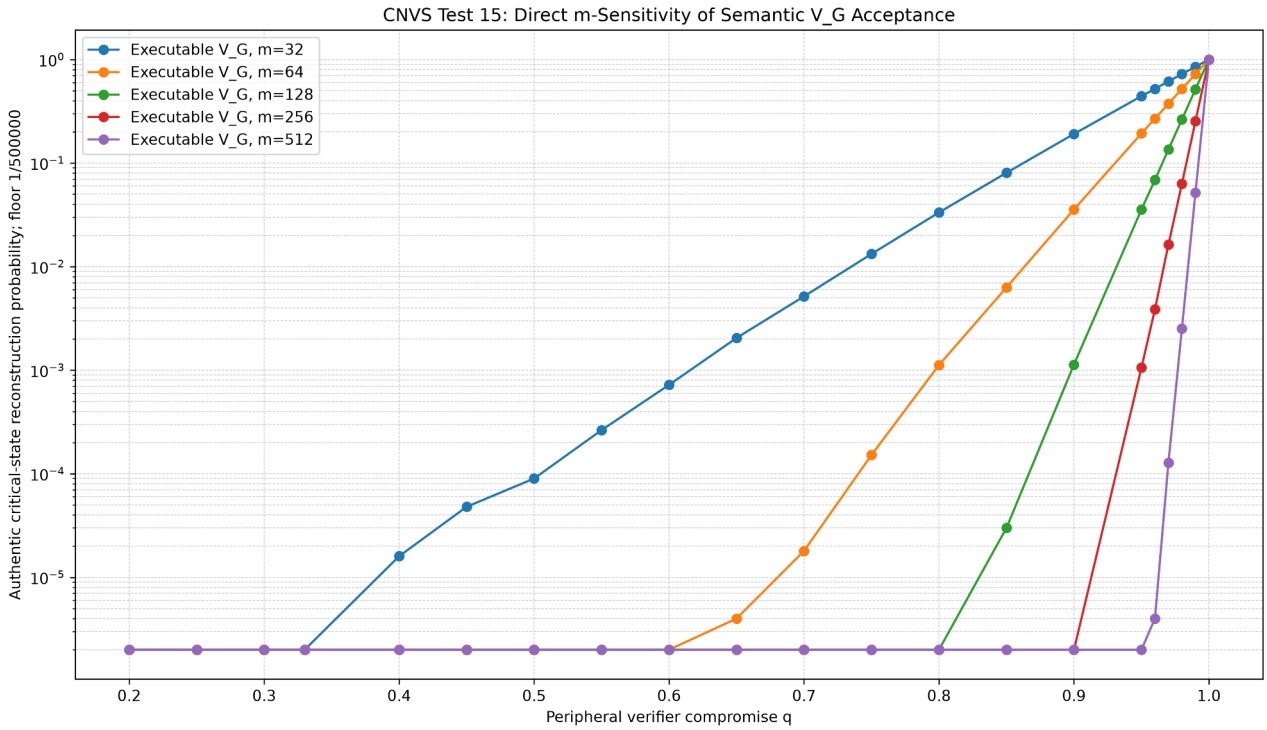
$$V\_L \rightarrow Cons\_R \rightarrow Inv\_C \rightarrow V\_G$$

and evaluates the effect of varying m on the probability of adversarial reconstruction, on the intervention of the Global Veto, on the leakage of the hidden structure, and on computational performance on a single device.
The maximum budget for the Monte Carlo sweep remains 500,000 candidates per configuration and collusion level.

Test 15 therefore does not introduce a new architecture compared to Test 14:
it isolates m as an experimental parameter and verifies whether the behavior observed with m = 64 persists, improves, or degrades as the critical cardinality varies.

The individual outputs, CSVs, specific figures for each value of m, and the related manifests are stored in the repository;
this report primarily reports the comparative figures.

Collusion Sweep and Direct Sensitivity to m

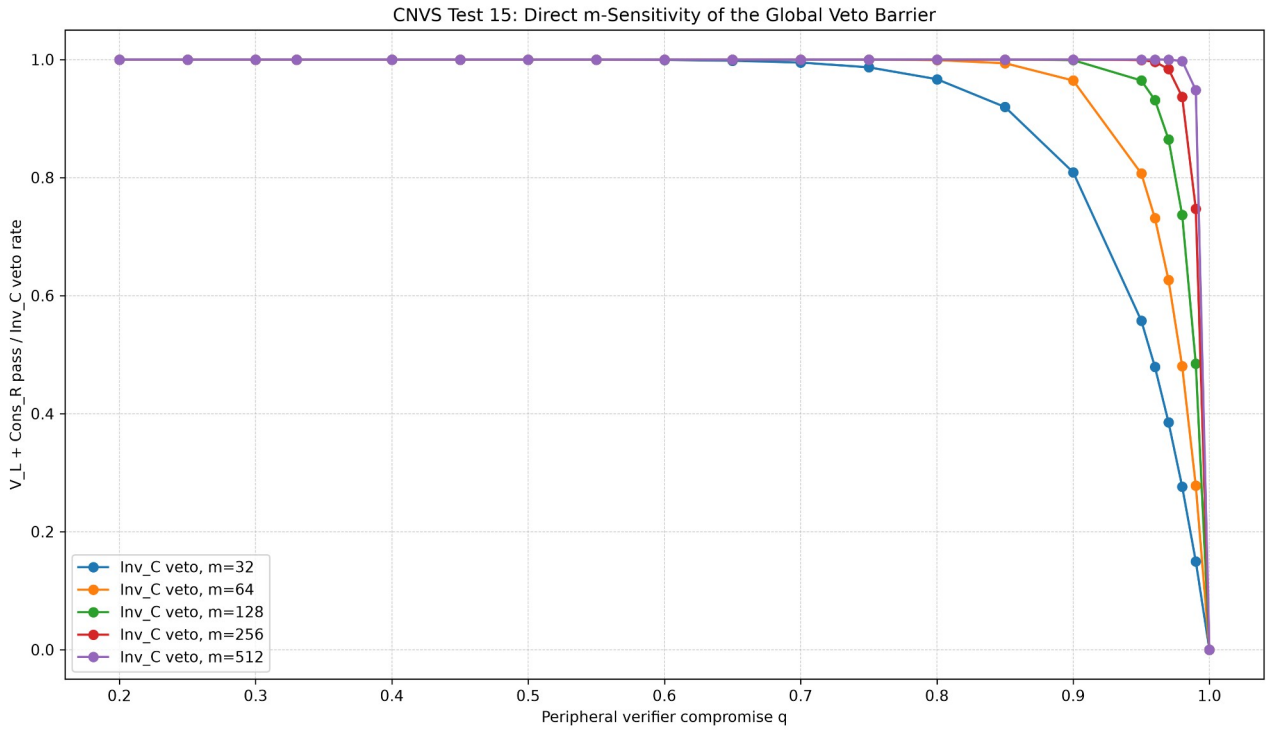


Filename Reference: "*test_15_comparison_semantic_vg_by_m*"

Partial Test - Semantic VG Acceptance by m

The figure represents the main result of Test 15.
Compared to the baseline m = 64 of Test 14, increasing m causes a progressive rightward shift in the adversary's authentic reconstruction curve.
With m = 32, observable successes emerge even at relatively moderate q values (starting from values of 0.33).
For m = 64, the departure from the Monte Carlo floor occurs later.
By increasing m to 128, 256, and finally 512, the region in which no successes are observed progressively extends towards peripheral compromise values approaching unity.
The dependence on m is therefore clearly visible:
in the model and parametric domain tested, a higher critical cardinality significantly reduces the probability of authentic reconstruction for a given q.

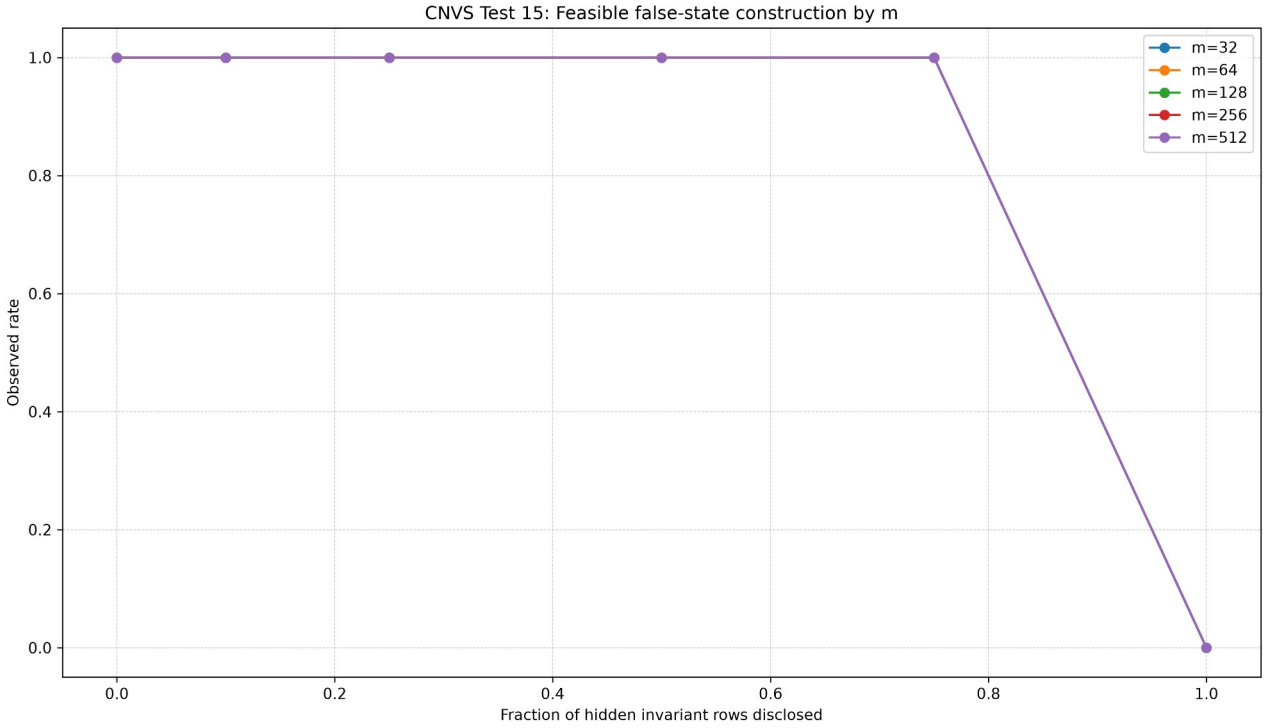
The result is consistent with the behavior predicted by Theorem 4, without constituting in itself a general proof of the theorem.



Filename Reference: "*test_15_comparison_inv_c_veto_by_m*"

Partial Test — Global Veto Barrier by m

A shift similar to that in the previous figure emerges from directly observing the InvC intervention rate.
With m = 32, the Global Veto begins to lose effectiveness at values of q significantly lower than in more fragmented configurations.
The baseline m = 64 already shifts the failure region to the right, while m = 128, 256, and 512 maintain the veto rate close to unity up to progressively higher levels of collusion.
The comparison therefore shows the same phenomenon from a complementary perspective: increasing m widens, in the tested domain, the compromise range in which the pipeline continues to reject adversary states.
The final collapse for q = 1 remains present for every configuration, consistent with the model in which the control or complete reconstruction of the critical values eliminates the residual uncertainty required by the tested strategy.



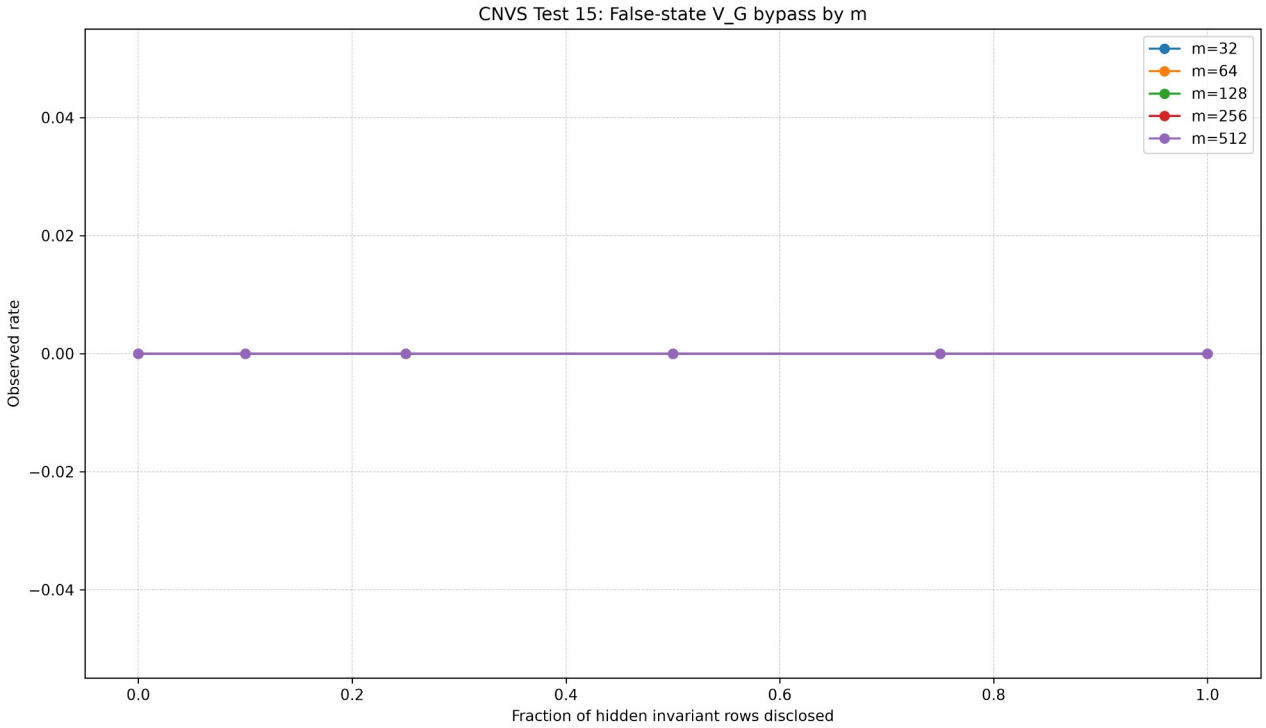
Filename Reference: "*test_15_comparison_leakage_feasible_false_state_by_m*"

Partial Test — Feasible False-State Construction under Partial C_int Disclosure

This figure does not measure the success of the false state against V_G, but only the possibility, from the adversary's perspective, of constructing a non-zero perturbation compatible with the portion of C_int already known.

At the sampled partial disclosure levels, the exposed submatrix retains a non-trivial null space; the adversary can therefore generate a vector $\Delta \neq 0$ that satisfies all known rows. This candidate, however, remains subject to the unrevealed rows of C_int, which, as shown in the following graph, prevent any observed false-state V_G bypass.

At complete disclosure — that is, when C_int is completely known, in the identifying regime used — the complete matrix has full rank and the null space is reduced to the null vector alone: there is therefore no longer a non-genuine perturbation that can simultaneously satisfy all the constraints.
Under this condition, the model shifts from constructing false states to authentically reconstructing the original state.

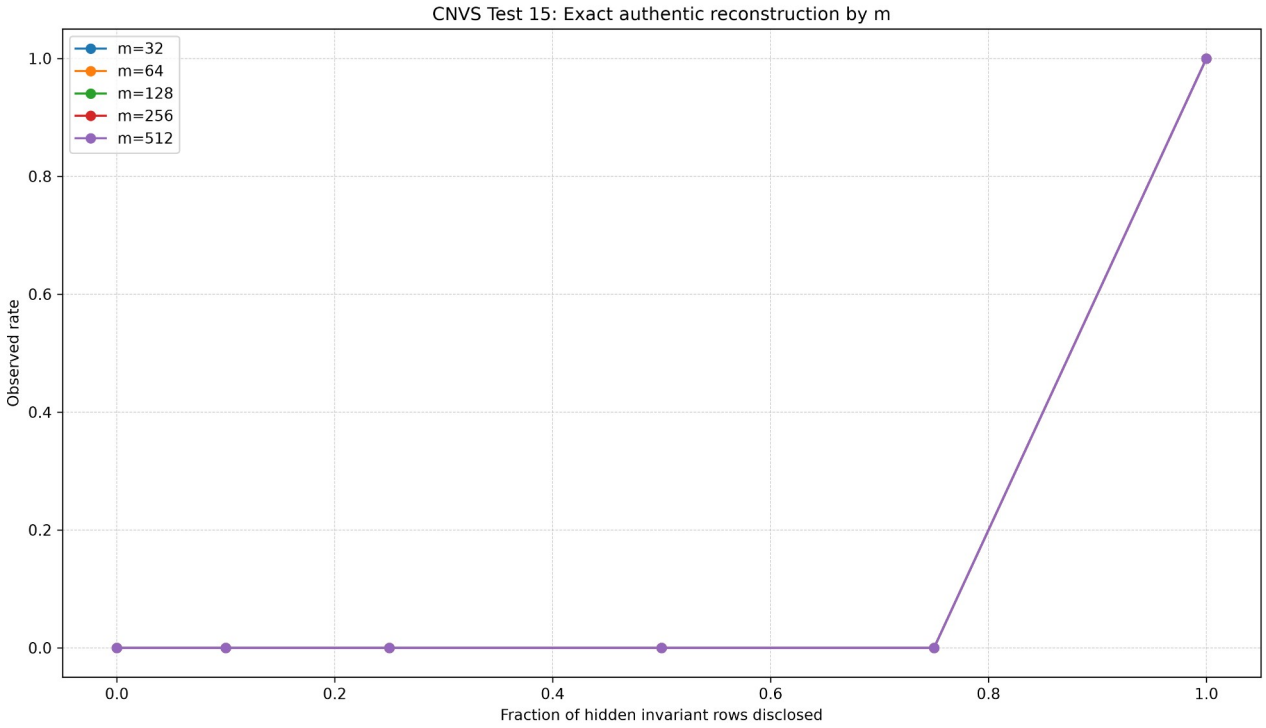


Filename Reference: "*test_15_comparison_leakage_false_state_bypass_by_m*"

Partial Test — False-State V_G Bypass by m

This figure distinguishes the ability to construct a false state compatible with the exposed portion from the ability to make it accepted by the complete system.
For each m ∈ {32, 64, 128, 256, 512}, no false-state bypass was observed at the sampled disclosure levels.

The five curves coincide at zero.
The result shows that, under the implemented adversarial strategy and in the identifying regime considered, the effect of the Global Veto against these perturbations is not degraded by the variation of m.
The unrevealed equations continue to discriminate false states constructed using only the partially exposed information.



Filename Reference: "*test_15_comparison_leakage_exact_reconstruction_accept_by_m*"

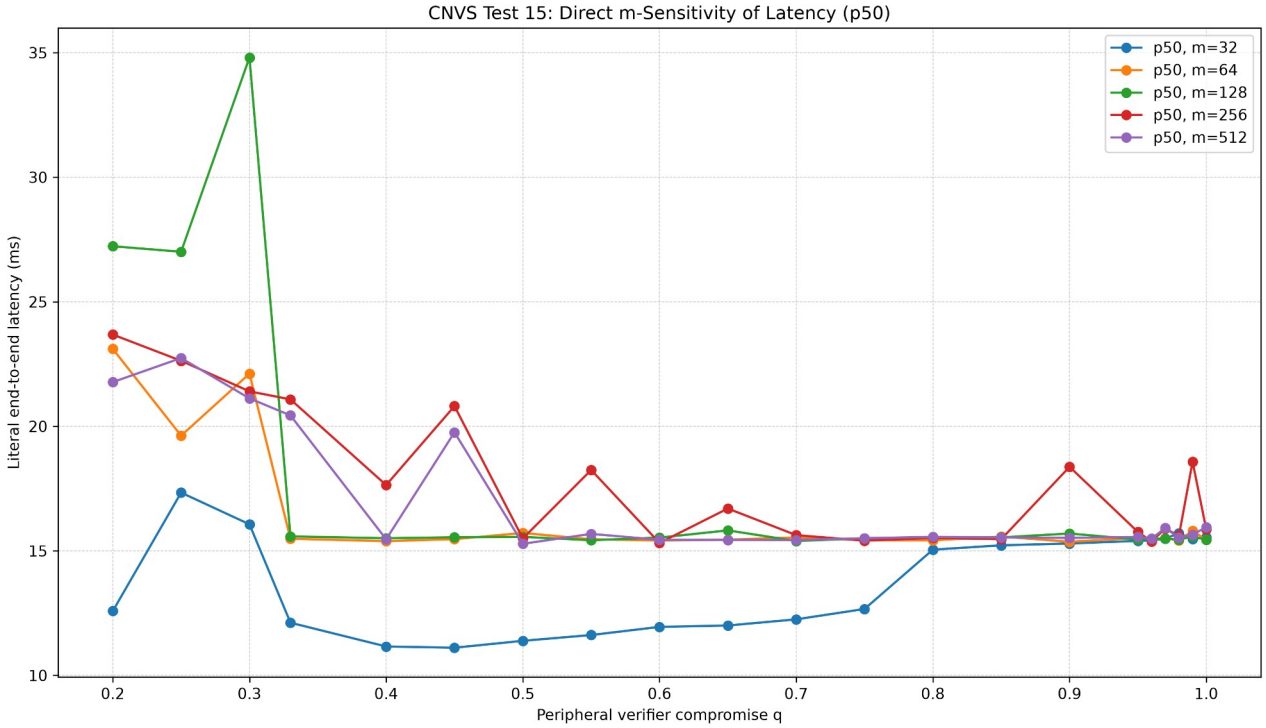
Partial Test — Exact Authentic Reconstruction by m

The authentic reconstruction exhibits equally uniform behavior with respect to m.
For all tested values, the observed rate remains zero at the sampled partial disclosure levels and reaches 1.0 in the full disclosure condition used by the test.

The result must be interpreted separately from false-state bypass: with full disclosure, the adversary does not force acceptance of an arbitrary state, but has the information required by the model to perform an authentic reconstruction.

Therefore, in Test 15, varying m strongly changes the probabilistic dynamics of peripheral collusion, but does not qualitatively alter the boundary condition observed in the leakage experiment.

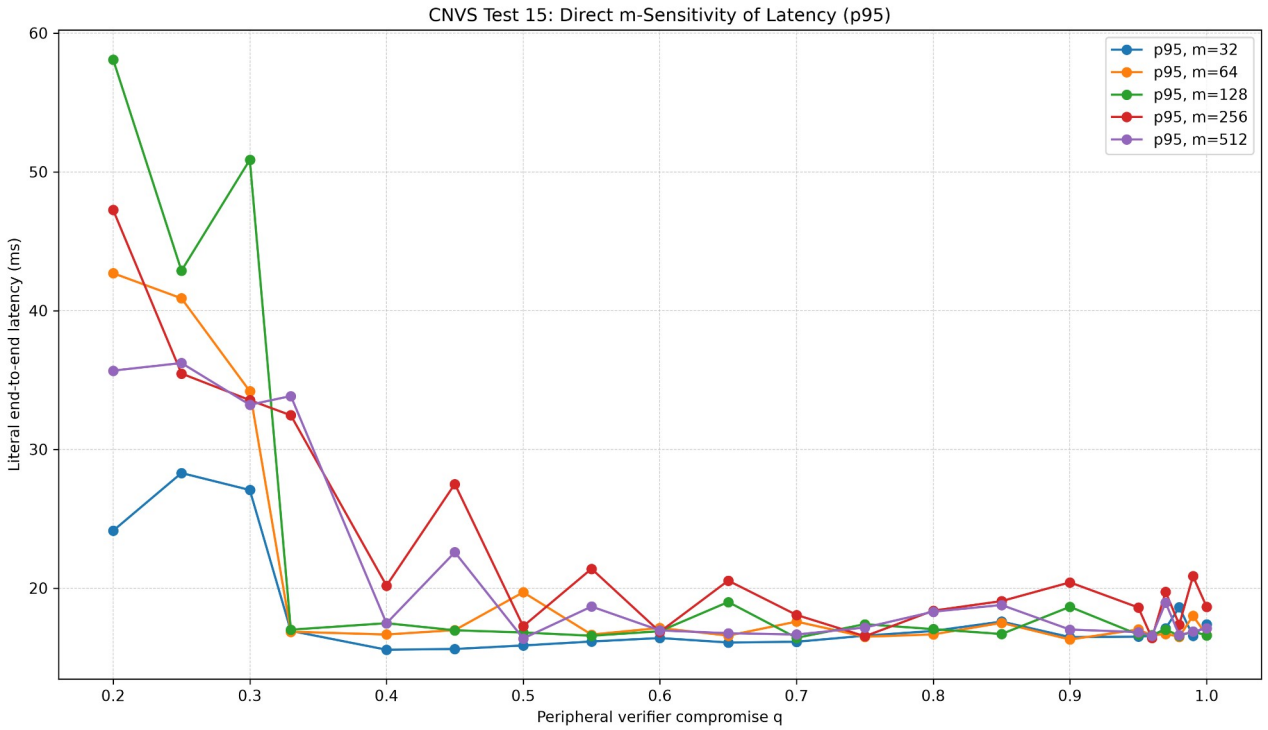
Latency sensitivity to m



Filename reference: "*test_15_comparison_latency_p50_by_m*"

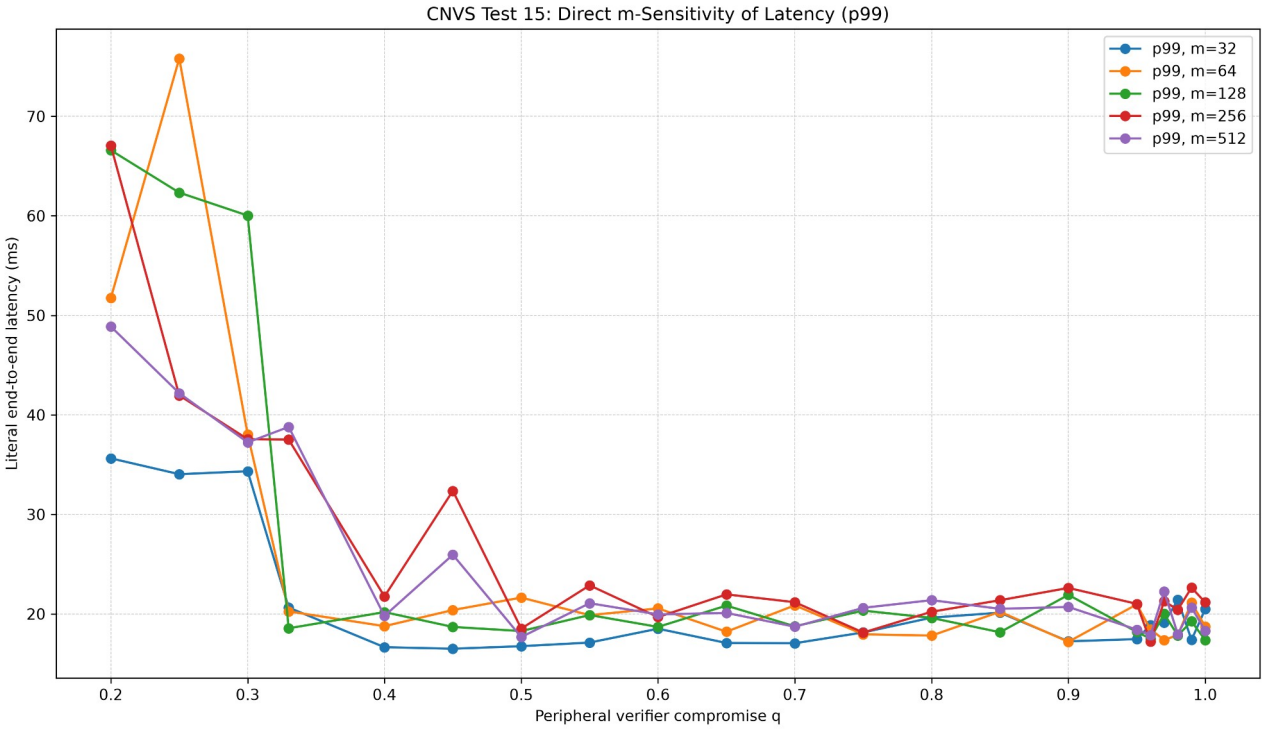
Partial Test — End-to-End Latency p50 by m

The median latency shows some differences and peaks at low collusion fractions, especially for some m configurations.
However, these differences do not follow a monotonic order with respect to fragmentation: higher values of m do not systematically produce higher latencies.
In the central and upper reaches of the q domain, however, the curves tend to concentrate predominantly in the same range, around approximately 15 – 20 ms.
Therefore, compared to the baseline m = 64, the increase up to m = 512 does not produce a systematic increase in median latency in the observed run.



Reference Filename: "*test_15_comparison_latency_p95_by_m*"

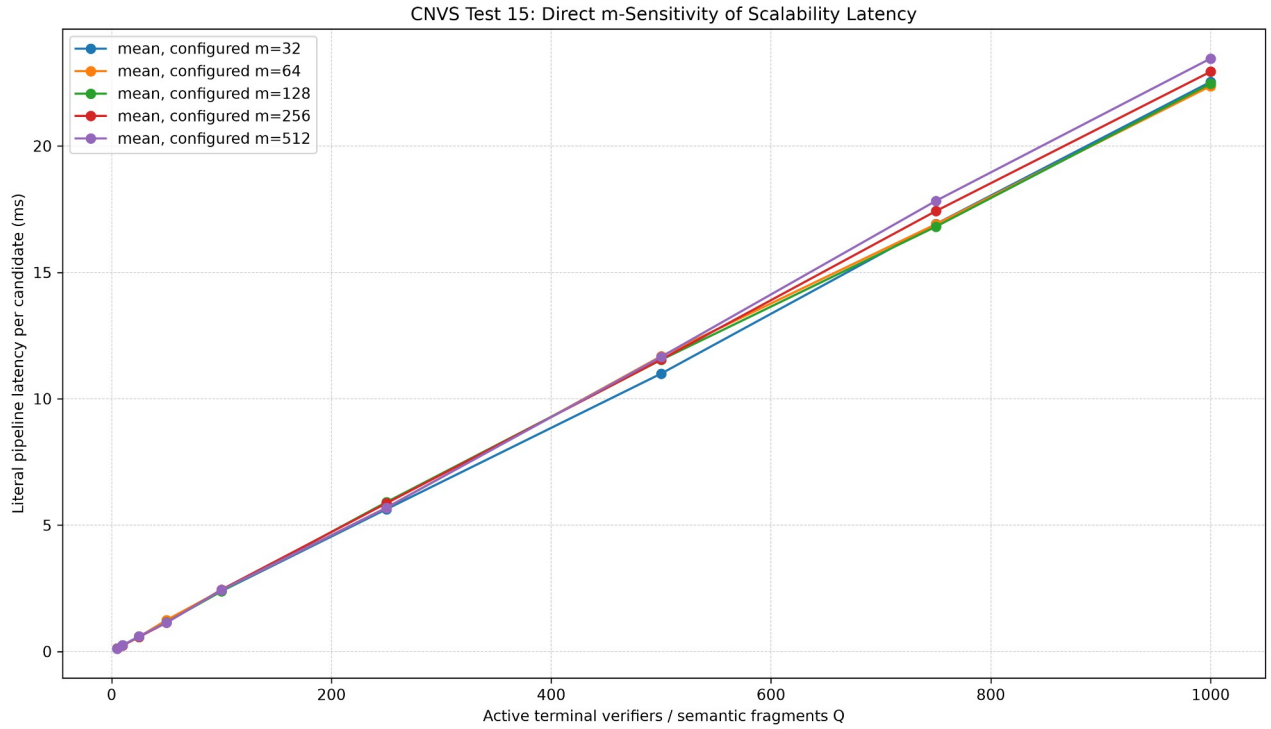
The p^{95} percentile confirms the same picture.
At low collusion fractions, larger oscillations and some peaks are present, but their amplitude does not increase in an orderly fashion with m.
For most of the subsequent domain, the different configurations converge toward a relatively narrow band.
The relevant conclusion is therefore not that the curves are identical, but that there is no monotonic p^{95} penalty associated with increasing the critical surface from
m = 64 to m = 128 , 256, 512.



Filename reference: "*test_15_comparison_latency_p99_by_m*"

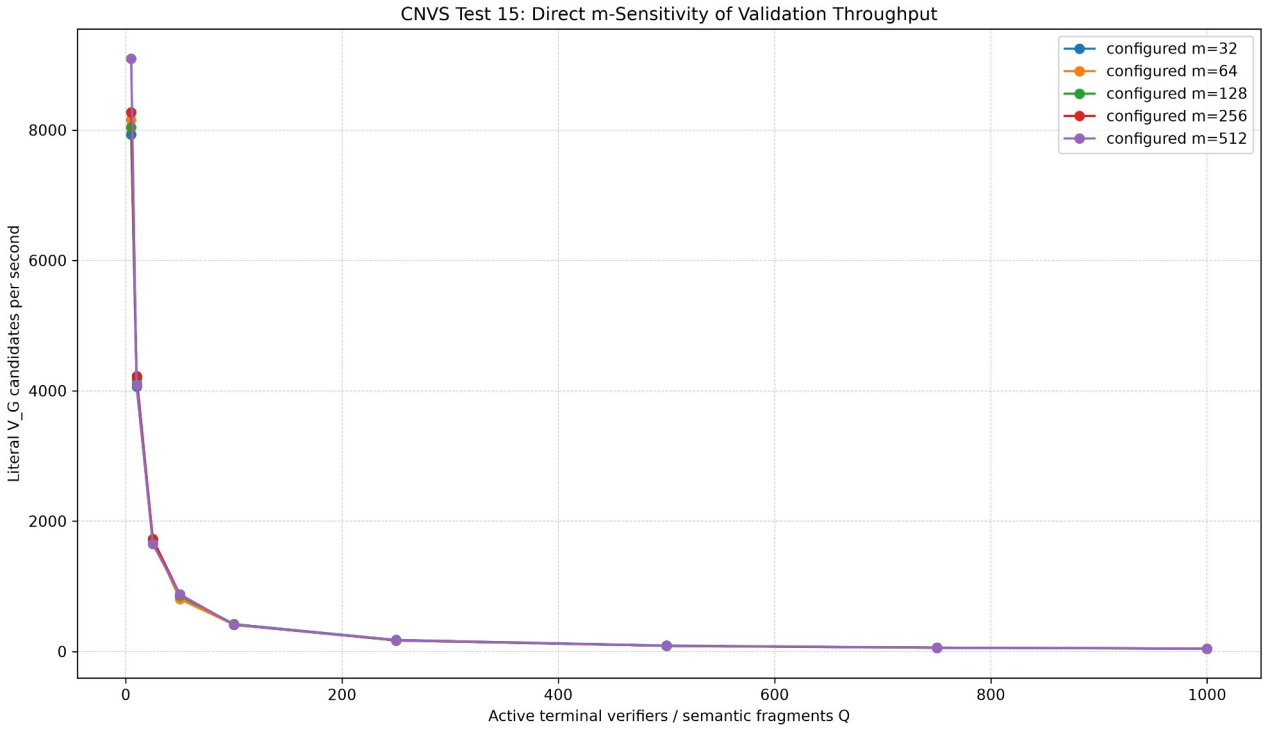
The p^{99} naturally shows greater variability, with outliers particularly evident at low collusion fractions.
Even in this case, however, the peaks do not follow a monotonic progression with m: for example, an intermediate configuration may show higher values than configurations with higher m.
After this initial region, the curves are concentrated mainly in the range of approximately 17 – 23 ms. In the observed domain, increasing m therefore does not cause a systematic deterioration of the upper tail of the temporal distribution.
The recorded peaks should be considered empirical measures of the run and not hard bounds.

Computational Scalability by Varying m



Filename: "*test_15_comparison_scalability_latency_by_m*"

This figure highlights the computational effect of varying m.
As Q increases, the five latency curves remain very close to each other and show approximately linear growth in the range Q = 5 … 1000.
Compared to the baseline m = 64, increasing m = 128, 256, and 512 introduces only limited differences in the overall cost associated with increasing Q.
At high Q, m = 512 shows a slightly higher cost, but no qualitative change in the observed scaling law emerges.
The result therefore suggests that in the local computational baseline of Test 15, the dominant cost is associated with the size of the processed structure Q, while increasing m within the tested range produces a comparatively low overhead.



Filename reference: "*test_15_comparison_scalability_throughput_by_m*"

Throughput provides a complementary representation of the previous graph.
As Q increases, the number of V_G candidates processed per second decreases, with a roughly inverse trend to the increase in latency.
The curves for different values of m remain almost overlapping across the entire domain.
The differences visible at smaller values of Q quickly diminish compared to the dominant effect produced by the increase in the number of active fragments/verifiers.
The comparison with m = 64 therefore shows that increasing the critical cardinality up to m = 512 does not introduce a macroscopic degradation in throughput in the tested computational environment.
However, these data describe a pipeline executed on a single device and do not constitute a distributed network benchmark or a direct performance comparison with BFT or consensus protocols.

Conclusion

Test 15 experimentally isolates the parameter m and shows that it exerts a marked effect on the adversarial reconstruction dynamics, while its computational impact remains limited in the range studied.
Compared to the m = 64 configuration used in Test 14, progressively higher values of m shift the region in which authentic reconstruction successes emerge toward increasingly higher levels of collusion and, consistently, extend the interval in which Inv_C maintains a high veto rate.
This behavior provides empirical evidence of the dependence of security on the critical cardinality predicted by the model.
The leakage experiments, however, show a qualitatively unchanged behavior with respect to m: at the partial disclosure levels sampled, no false-state V_G bypass was observed, while the

full disclosure condition leads to the authentic reconstruction predicted by the identifying regime.
Computationally, direct comparisons of p^{50}, p^{95}, p^{99}, latency versus Q, and throughput show no monotonic or macroscopic degradation associated with increasing m from 64 to 512.
In the tested domain, the observed gain in security curve separation is therefore achieved without a corresponding macroscopic increase in local pipeline cost.
The results remain valid for the stated critical surface, adversarial model, parameters, and execution environment.
They do not constitute a universal proof of security or an asymptotic demonstration of the complexity of a future distributed network, but they complement Test 14 by directly showing how varying m affects the behavior of the executable implementation.

Comitato Massimo,
14/ 08/ 2026,
Italy