

iQs GROUP INC.

EXECUTIVE WHITEPAPER

Why We Open-Sourced ProofGate ZP-11

A scholarly rationale for open-sourcing a release-governance standard the world can verify

By Naquib Hatami

Inventor & Author

github.com/iQs-Group/ZP-11

November 2026 · Released under the MIT License

iQs GROUP INC. · EXECUTIVE & TECHNICAL WHITEPAPER

Why We Open-Sourced ProofGate ZP-11 under the MIT License

A scholarly rationale for open-sourcing a release-governance standard the world can verify

Naquib Hatami
Inventor & Author · iQs Group Inc. · github.com/iQs-Group/ZP-11

Abstract

This paper sets out the reasoning behind a deliberate act: the release of ProofGate ZP-11, the Zero-Point Standard as open, MIT-licensed a standard by its author, Naquib Hatami of iQs Group Inc. We draw a distinction that the discussion of open source usually blurs. For an artifact whose purpose is to be trusted, one thing is a logical necessity: the specification and the procedure by which its verdicts are checked must be public, because a verdict cannot be independently verified while the method of checking it is secret (§3). That necessity does not, by itself, require the full implementation to be open; it requires the rules of the game to be legible. The further decision — to open the implementation and to do so under the most permissive mainstream licence — rests not on logic but on the economics of standards and network effects (§4) and on mechanism design that minimises the activation energy of adoption (§5). We separate two senses of trust throughout: verifiable trust in a verdict, which the standard makes evidence-based, and institutional trust in the steward, which that very legibility makes legitimate and durable. We then answer the strongest objections (§6) and describe the stewardship that preserves authorship while surrendering control (§7). The conclusion is that openness here is a strategic choice rather than a concession, and we argue it is the most reliable route to durable authority over the category.

Keywords: *software supply chain, release governance, trust, open standards, network effects, MIT License, provenance, mechanism design.*

1. Introduction: the crisis of trust in the software supply chain

Modern software is assembled, not written. A typical application is a thin shell of original code floating atop a vast, mostly unseen substrate of third-party libraries, build tools, container images, and continuous-integration machinery, each maintained by strangers and each capable, in principle, of betraying the whole. The discipline that studies this substrate — software supply-chain security — was for decades a specialist concern. Two public events ended that quiet. In December 2020 the compromise of a widely used network-management product (the SUNBURST incident) demonstrated that an attacker who subverts a trusted build pipeline can distribute a backdoor to thousands of organisations at once, bearing a valid signature the entire way [8]. A year later a single logging library produced CVE-2021-44228, “Log4Shell”, and the world learned that a flaw in a transitive dependency few had ever named could place a remote-code-execution primitive inside an enormous number of Java services worldwide [9].

The institutional response was swift and, revealingly, converged on a single word: evidence. The United States codified Executive Order 14028 [11]; NIST published the Secure Software Development Framework

[10]; the Open Source Security Foundation advanced SLSA, a graduated set of provenance guarantees [7]; and the research community had already given us in-toto, a scheme for attaching farm-to-table attestations to every step of a build [6]. What unites these efforts is a shift of the burden of proof. The old question — "do we trust this vendor?" — is replaced by a harder and better one: "what can this change prove about itself?"

ProofGate ZP-11, the Zero-Point Standard is an answer to that question, and this paper is not primarily about the answer. It is about a secondary decision that is, we argue, as consequential as the invention itself: the decision to publish openly, under the most permissive of the mainstream licences, rather than to retain the work as proprietary advantage. That decision looks, on a naive reading, like the surrender of an asset. We will show that it is better understood as engineering: for this class of artifact, concealment of the rules would have destroyed the very value the artifact exists to create, and openness of the implementation is the most reliable way to turn an invention into a category.

The thesis, stated once and then earned: trust in a verdict evaporates the moment the verdict can only be taken on faith — so the method of checking it must be public; whether the rest of the machinery is open is a question of strategy, and the strategy favours openness.

2. The object of the decision: ProofGate ZP-11 in brief

ProofGate ZP-11 is an open, fail-closed release-governance standard. Between every proposed change and a protected branch it interposes eleven independent checks — the Zero-Point zones — each with a defined input, a deterministic verdict, and machine-readable evidence. The gate is closed by default. It opens only when every zone returns PASS and the complete result is sealed into a Proof-Pack: a single, self-describing, cryptographically signed record that a third party can re-run to obtain the identical verdict.

The design inverts the prevailing convention. Most release checks are advisory — they warn, then permit the merge regardless. ZP-11 treats the absence of affirmative, reproducible evidence as failure rather than as consent. A change advances on proof, never on default. Integrity rests on two public primitives: each zone receipt is digested over a canonical serialisation (RFC 8785 [14]) and signed with Ed25519 (RFC 8032 [13]); the Zero-Point seal signs the ordered chain of receipt digests, so a pack cannot be assembled from receipts that never shared a gate — a tamper-evident, append-only construction in the lineage of signed attestation logs such as in-toto [6].

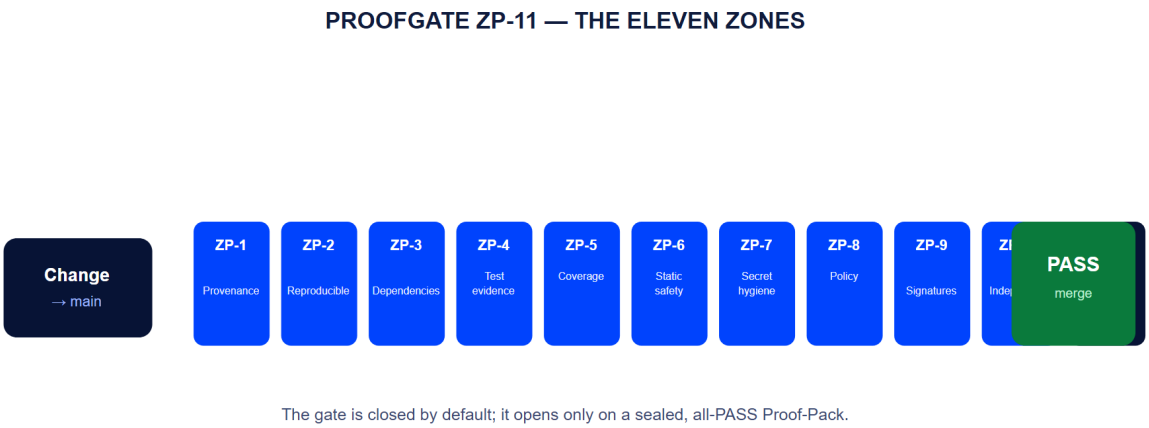


Figure 1. The eleven Zero-Point zones form a fail-closed gate between a change and the main branch; the gate opens only on a sealed, all-PASS Proof-Pack that any third party can re-run.

Crucially for the argument of this paper, the Proof-Pack is verifiable from nothing but the pack and a public key, so trust in the verdict is reconstructed from evidence rather than extended to the pipeline — subject to the honest caveat of §3.1 that the verifier itself must be an independently buildable root. This property — that the mechanism asks to be checked rather than believed — is what makes ZP-11's specification a thing that must be public (§3), and what makes opening its implementation the most reliable route to a category (§4).

3. What must be open, and why

3.1 The verification paradox

Begin with the deepest version of the problem, which predates the supply chain by forty years. In his 1983 Turing Award lecture, published in 1984, Ken Thompson described a compiler modified to insert a backdoor — and modified further so that the backdoor reproduced itself even after the incriminating source was removed. His conclusion was lapidary: "You can't trust code that you did not totally create yourself" [1]. The lecture is usually read as a counsel of despair. We read it as a specification. Thompson proved that trust cannot be manufactured by inspection of any single artifact in isolation; it must be reconstructed from an independently checkable chain of evidence, and it must ultimately bottom out in a trusted root one did not oneself create.

Honesty about that root is essential, because it is where a careless argument fails. ProofGate ZP-11 does not eliminate trust; it relocates and minimises it. The trusted base is reduced to two things: the custody of a signing key, and the integrity of the verifier a third party runs. Neither is magic, and Thompson's regress applies to both. The standard countermeasure is well understood — a reproducible, independently buildable verifier, bootstrapped by diverse double-compilation [15] — so the verifier itself

can be checked rather than believed. The honest claim is therefore not that trust is abolished but that the trusted root is made small, explicit, and independently re-derivable.

Now observe the recursion that gives the section its name. If trust in a build must rest on evidence rather than on faith in the builder, then the procedure that evaluates the evidence inherits the same demand. If that procedure is secret, a third party is asked to accept a verdict it cannot check — the precise move the entire field exists to abolish. This is the verification paradox: a verdict offered without a public, re-runnable method of checking it is no better than "because we say so." It follows that the specification and the verification procedure of a trust mechanism must be public. This much is a necessity, not a preference.

"A trust mechanism whose specification is secret is a contradiction in terms."

3.2 Two things that are easy to conflate

What the paradox establishes is narrow and must not be overstated. It requires that the rules — the specification, the evidence format, and the method of verification — be public and independently executable. It does not, by itself, require that the entire implementation be released under an open-source licence. The distinction is real and a sophisticated reader will insist on it: verifiability and secrecy are not globally incompatible. Zero-knowledge proofs verify a claim precisely while keeping the witness secret; a signature is checked against a public key while the private key stays hidden. What cannot be secret is the procedure by which the verdict is checked. A closed implementation that emitted a fully re-runnable, publicly specified proof could, in principle, satisfy the paradox.

Why, then, open the implementation at all? For two further reasons, neither of which is logical necessity and both of which are decisive in practice. The first is that not everything that matters about a governance mechanism is captured by its cryptographic evidence: the policy it encodes, the completeness of its checks, the cases it silently ignores — these are audited by reading the implementation, not by verifying a signature. The second is strategic, and occupies the remainder of this paper: a standard's value is a function of its adoption, and openness is the most reliable instrument of adoption (§4–§5). The analogy to cryptography is exact on the narrow point. Kerckhoffs's principle holds that a system must remain secure even if everything about it except the key is public; modern schemes such as Ed25519 [12][13] and canonicalisation standards such as JCS [14] are trusted precisely because their design is open to universal scrutiny. The specification's strength is a public fact. Openness of the specification is a precondition of trust; openness of the implementation is a precondition of a category.

This also dissolves an apparent contradiction the reader may already sense. We will shortly build a business on reputation and "managed assurance" (§4.2, §7) — that is, on a kind of trust extended to the steward partly on the strength of its record. There is no inconsistency, because two different things are called trust. Verifiable trust is trust in a verdict, and the standard makes it evidence-based, owed to no one. Institutional trust is trust in the organisation that stewards and operates the standard, and it is

reputational. Removing the need for faith at the verdict layer is exactly what makes reputational trust at the service layer legitimate rather than lazy.

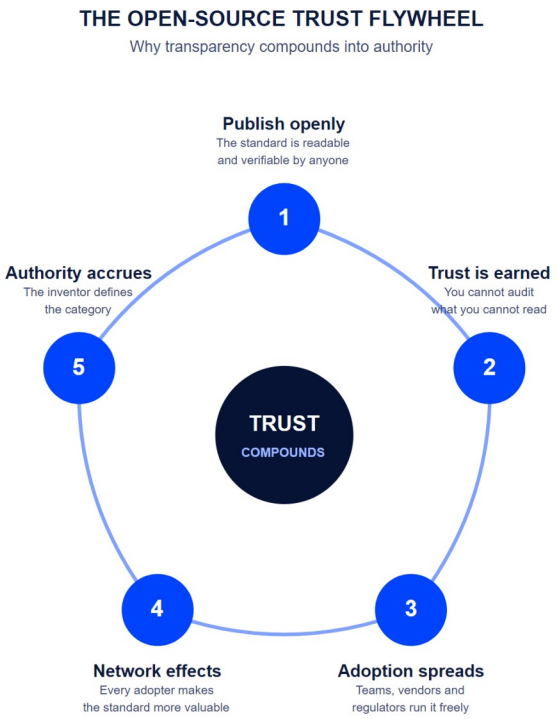


Figure 2. Openness of the implementation is the first mover in a self-reinforcing cycle: legible rules invite trust in verdicts; that trust is adopted; adoption compounds the value of the standard and, with it, the standing of its author.

4. The economics of standards: adoption as the decisive variable

Granting that the rules of ProofGate ZP-11 must be legible to be credible, a separate question remains: why open the implementation and license it so permissively that even a competitor may build on it without payment? The answer lies in the peculiar economics of standards, which do not behave like ordinary goods. The value of a standard is not intrinsic; it is a function of how many others have adopted it. Katz and Shapiro's analysis of network externalities [2] formalises the intuition: for goods whose utility rises with the size of the adopting population, the dominant strategic variable is not price or secrecy but installed base. A measuring system that only one laboratory uses measures nothing comparable; a protocol that only its author speaks is not a protocol but a monologue.

ProofGate ZP-11 is a network good of a strong kind. Its purpose is to let mutually distrustful parties — a vendor and its customer, an auditor and an insurer, a regulator and the regulated — agree on what counts as evidence. That agreement is worthless unless it is shared. Every additional adopter does not merely add one user; it adds an edge to a graph of parties who can exchange proofs without renegotiating what proof means. The utility rises with adoption; precisely how fast is debated — the quadratic "Metcalfe" intuition is contested, with careful estimates closer to n·log n [18] — but the direction is not. Any barrier to adoption — a fee, a licence restriction, a dependence on the author's goodwill, even the friction of

legal review — is therefore not a toll that captures value; it is sand in the gears of the process that creates it [17].



Figure 3. Six independent lines of reasoning converge on a single decision.

4.1 The standard as a commons

If adoption is the engine, the natural fear is the tragedy of the commons: a shared resource that everyone uses and no one maintains. Elinor Ostrom's work dissolved the inevitability of that tragedy by showing, with decades of empirical cases, that communities sustain shared resources when the rules of use, contribution, and stewardship are clear and legitimate [3]. The lesson transfers directly. An open standard is not abandoned to the commons; it is governed as one, with an explicit steward, versioned rules, and a published process. The permissive licence supplies the frictionless substrate; the governance model (§7) supplies the institutional structure Ostrom showed to be decisive. Raymond's account of open-source production [4] and Benkler's analysis of commons-based peer production [5] describe the same phenomenon from the inside: openness, properly stewarded, is not the absence of order but a different and often more durable kind of it.

4.2 Give the standard, sell the assurance

The apparent paradox — that a commercial enterprise would open-source its own invention — dissolves once one distinguishes the standard from the service. Arrow observed that information is costly to produce and nearly free to copy, so that the private return on invention is chronically below its social value [16]. The conventional remedy is secrecy or patents, which recapture value by restricting use. For a

standard that remedy is self-defeating, because restriction destroys the adoption on which value depends. The alternative, pursued deliberately here, is to let the standard circulate at zero friction and to capture value one layer up: in the scarce, non-copyable goods that adoption makes valuable — reputation, expertise, tooling, integration, and the managed assurance that enterprises will often prefer to buy rather than build. iQs Group gives away the ruler and sells the surveying.

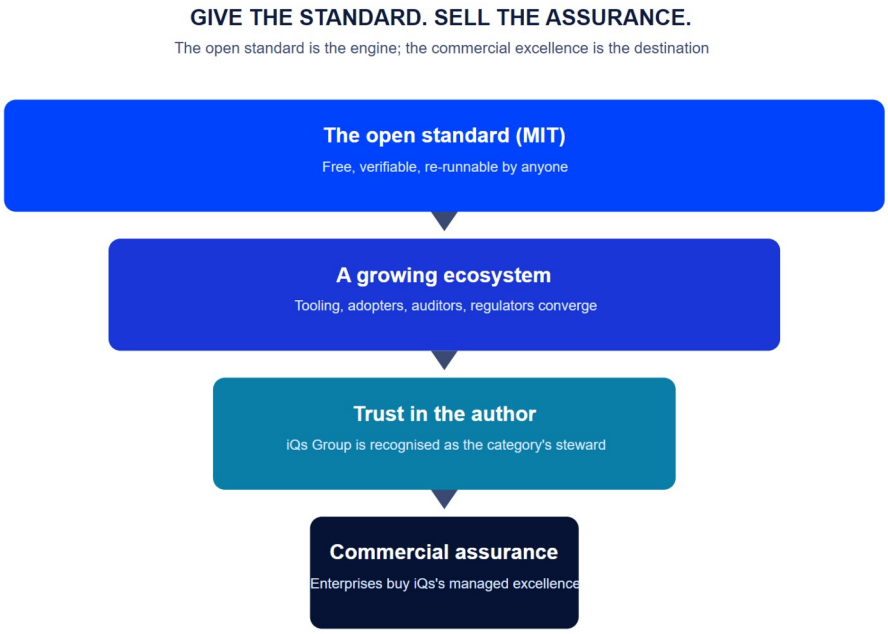


Figure 4. The open standard is the broad, frictionless top of the funnel; commercial value is realised in the narrow, high-trust layer the standard makes legible and legitimate.

5. The licence as mechanism design: why MIT, honestly

A licence is not a legal footnote; it is the mechanism that sets the activation energy of adoption. The question is therefore an engineering one: which licence minimises the barrier between a potential adopter and the act of running the standard, across the widest range of adopters — individuals, start-ups, Fortune 100 legal departments, regulators, and the commercial competitors whose participation, counter-intuitively, strengthens rather than threatens the author's position?

The permissive licences (MIT, BSD, Apache-2.0) permit commercial use, modification, and redistribution; the copyleft licences (the GPL family) permit the same but require that derivative works also be opened. For a standard the choice is strategic and has a determinate answer on the dimension of friction. Copyleft introduces a contingent obligation — "if you build on this, you must open your own work" — that every enterprise legal review treats as a risk to be contained, and that therefore suppresses precisely the enterprise adoption a governance standard most needs. Among the permissive options, MIT is the minimal permissive licence that still asserts authorship: a single paragraph granting use, modification, and

distribution with no obligation beyond preserving the notice. (Lighter dedications exist — 0BSD, CC0 — but they forfeit the attribution that anchors authorship, which we are unwilling to give up.)

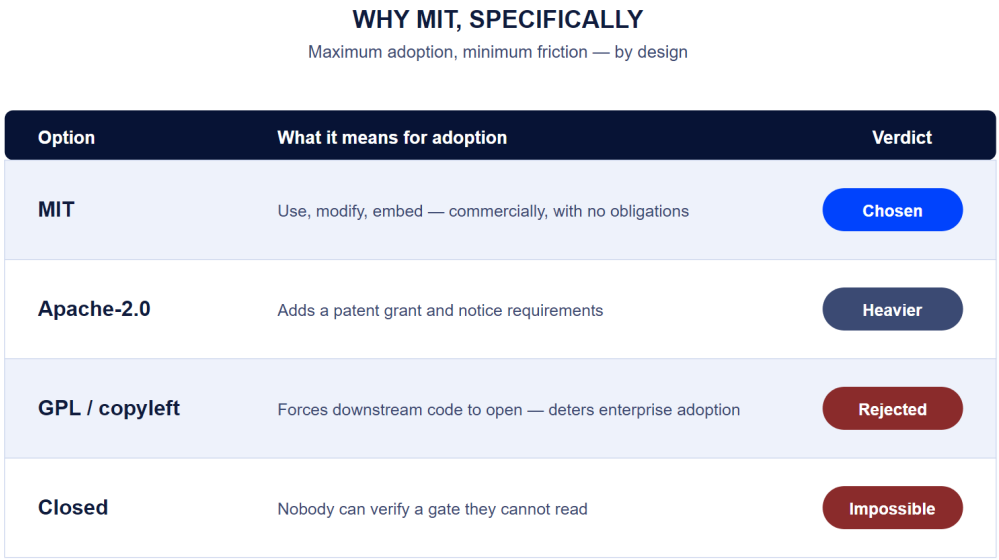


Figure 5. The licence decision framed as friction minimisation — with one honest asterisk, addressed below: friction is necessary but not sufficient.

Here we must confront the one place where a pure friction argument would be naive, because the most sophisticated reader — an enterprise IP counsel — will raise it immediately. MIT contains no explicit patent grant; Apache-2.0 does. For a security standard that is not a rounding error: a missing patent grant leaves implementers exposed to third-party patent claims and offers them no patent peace from the author. On that single dimension Apache-2.0 is the safer instrument, and it would be dishonest to wave its patent clause away as mere "weight." We choose MIT anyway, and we close the gap explicitly rather than leaving it for a reviewer to find: the author makes a public, irrevocable commitment not to assert any patent he holds or may obtain against any implementation or use of this standard, and is aware of no third-party patent that reads on it. MIT supplies the minimal-friction grant; the non-assertion pledge supplies the patent peace the licence omits. The combination, not MIT alone, is the mechanism.

There is a second, quieter reason to prefer the lightest licence for a trust mechanism. Copyleft operates by compulsion — the threat of licence violation forces a behaviour. A standard whose entire claim is "we ask you to take nothing on faith; verify everything" is better served by the licence that compels least. The medium should match the message.

6. Objections, and why they fail

The decision invites sharp objections from intelligent readers. We state the strongest versions and answer them, because an argument that only survives weak objections has not been tested.

Objection	Response
“You are giving away the crown jewels.”	The replicable part — the prose of the specification — is not the jewel. The non-obvious design is: the specific decomposition of release governance into independent, deterministic zones; the fail-closed evidence semantics in which silence is failure; and, for ZP-11, the seal computed over an ordered chain of canonicalised receipt digests so a pack cannot be assembled from receipts that never shared a gate. Publication secures priority over that design as public record, and converts it into the author’s category rather than a secret a competitor can independently rediscover and claim.
“Competitors will copy it.”	Adoption by competitors is the mechanism of victory, not its defeat. Every party that implements the standard ratifies the author’s definition of the category and enlarges the network whose centre is the author. A standard with many implementers is a standard that won; a standard with one is a product that lost.
“MIT has no patent grant — that is a real exposure.”	It is, and we do not minimise it (§5). We pair MIT with an explicit, public patent non-assertion commitment from the author, which supplies implementers the patent peace Apache-2.0’s clause would, while keeping MIT’s minimal friction. The gap is closed deliberately, not ignored.
“Open means unaccountable — anyone can ship a broken version.”	Openness is paired with stewardship (§7): a versioned specification, a normative reference implementation, and conformance criteria. Forks are permitted, but a non-conformant fork cannot claim conformance — the model used by POSIX, OpenGL, and other standards backed by a conformance suite.
“There is no revenue in a free standard.”	Correct, and intended. Revenue is realised one layer up, in assurance, tooling, and expertise (§4.2). The standard is a trust-building and customer-acquisition instrument whose return is measured in category authority, not licence fees.

Each objection, pressed to its end, reduces to a single premise that is true for ordinary software and false for the rules of a trust mechanism: that value lives in concealment. Concealment of the rules is the one thing guaranteed to destroy the value, because the value is verifiable trust, and verifiable trust is the refusal to conceal the method of verification.

7. Stewardship, provenance, and the paradox of authority

Surrendering control is not the same as surrendering authorship, and the distinction is the heart of the stewardship model. Control — the legal power to forbid — is given away by the MIT grant. Authorship — the recognised status of having defined the thing — is, paradoxically, amplified by that surrender. The history of standards is consistent on this point: authority over a category accrues not to whoever guards a specification most jealously but to whoever the community recognises as its origin and steward. The authors of the foundational internet and cryptographic standards hold no royalties on them; their authority has proved enduring precisely because they gave the work away and then tended it well.

Publication also performs a quieter legal function, with one honest limit. A dated, public, openly licensed release establishes prior art and provenance, which protects the author from a later party enclosing the idea and asserting it back against its inventor. Prior art, however, protects the author, not adopters; what protects adopters from patent claims is the author's non-assertion commitment (§5). The two instruments are complementary, and we name both rather than let the first stand in for the second.

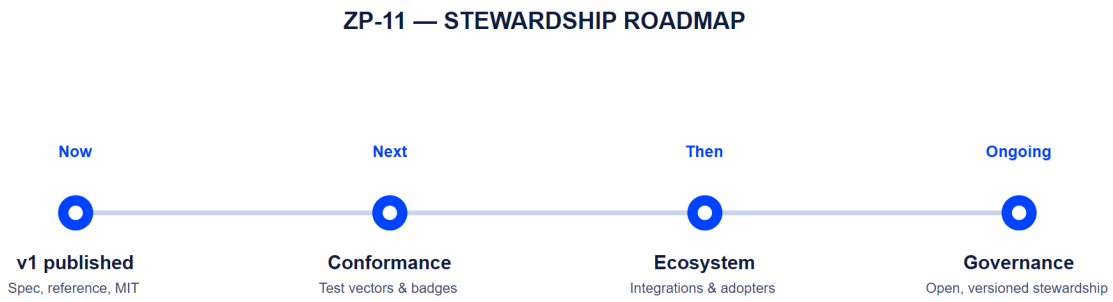


Figure 6. Stewardship is an institution, not a moment: a versioned specification and a normative reference, maintained in the open on a published cadence.

The steward's obligations are correspondingly concrete: to version the specification so that implementations can rely on a stable contract; to maintain a reference implementation as the normative oracle for conformance; to adjudicate conformance claims against a published suite; and to shepherd change through a transparent process. These are the design principles Ostrom identified in durable commons [3], translated into software. They are also, not incidentally, the credentials on which the commercial assurance business is built: the organisation trusted to steward the standard is the organisation enterprises will trust to operate it — the institutional trust of §3.2, earned rather than asserted.

8. Conclusion

The decision to release ProofGate ZP-11, the Zero-Point Standard under the MIT License is often read, by those who have not followed the argument, as idealism or as the abandonment of an asset. We have argued something more precise. One thing is a logical necessity: the specification and the verification procedure of a trust mechanism must be public, because a verdict cannot be checked while the method of checking it is secret (§3). The rest — opening the implementation and doing so under a minimal permissive licence — is not necessity but strategy: for an artifact whose value is a network, zero-friction adoption is the surest path to that value (§4–§5), and authority, far from being spent by the gift, is created by it, provided the author becomes the steward (§7). Where MIT is weaker than the alternatives — its silence on patents — we have closed the gap in the open rather than hidden it.

Openness here is a strategic choice rather than a concession. The most reliable way to hold a category is to define it in public, release it under the most generous terms that still assert authorship, and be the one the field trusts to keep it.

ProofGate ZP-11, the Zero-Point Standard is published, in full, at github.com/iQs-Group/ZP-11. It is offered to the field not as a product to be purchased but as an idea to be tested — which is, after all, the only kind of trust worth having.

References

[1] Thompson, K. (1984). Reflections on Trusting Trust. Communications of the ACM, 27(8), 761–763. (1983 Turing Award lecture, published 1984.)

[2] Katz, M. L., & Shapiro, C. (1985). Network Externalities, Competition, and Compatibility. American Economic Review, 75(3), 424–440.

[3] Ostrom, E. (1990). Governing the Commons: The Evolution of Institutions for Collective Action. Cambridge University Press.

[4] Raymond, E. S. (1999). The Cathedral and the Bazaar. O'Reilly Media.

[5] Benkler, Y. (2006). The Wealth of Networks: How Social Production Transforms Markets and Freedom. Yale University Press.

[6] Torres-Arias, S., Afzali, H., Kuppusamy, T. K., Curtmola, R., & Cappos, J. (2019). in-toto: Providing Farm-to-Table Guarantees for Bits and Bytes. USENIX Security Symposium.

[7] Open Source Security Foundation (OpenSSF). SLSA: Supply-chain Levels for Software Artifacts. https://slsa.dev

[8] FireEye/Mandiant (2020). Highly Evasive Attacker Leverages SolarWinds Supply Chain to Compromise Multiple Global Victims with SUNBURST Backdoor. Threat Research, 13 December 2020; see also CISA Emergency Directive 21-01 (December 2020).

[9] MITRE. (2021). CVE-2021-44228 ("Log4Shell"): Apache Log4j2 remote code execution.

[10] U.S. National Institute of Standards and Technology (2022). Secure Software Development Framework (SSDF), NIST SP 800-218.

[11] The White House (2021). Executive Order 14028: Improving the Nation's Cybersecurity.

[12] Bernstein, D. J., Duif, N., Lange, T., Schwabe, P., & Yang, B.-Y. (2012). High-speed high-security signatures (Ed25519). Journal of Cryptographic Engineering, 2(2), 77–89.

[13] Josefsson, S., & Liusvaara, I. (2017). Edwards-Curve Digital Signature Algorithm (EdDSA). IETF RFC 8032.

[14] Rundgren, A., Jordan, B., & Erdtman, S. (2020). JSON Canonicalization Scheme (JCS). IETF RFC 8785.

[15] Wheeler, D. A. (2009). Fully Countering Trusting Trust through Diverse Double-Compiling. Annual Computer Security Applications Conference (ACSAC); PhD dissertation, George Mason University, 2009.

[16] Arrow, K. J. (1962). Economic Welfare and the Allocation of Resources for Invention. In The Rate and Direction of Inventive Activity. Princeton University Press.

[17] Shapiro, C., & Varian, H. R. (1999). Information Rules: A Strategic Guide to the Network Economy. Harvard Business School Press.

[18] Odlyzko, A., & Tilly, B. (2005). A refutation of Metcalfe's Law and a better estimate for the value of networks and network interconnections. University of Minnesota.

Appendix A. ProofGate ZP-11 at a glance

ProofGate ZP-11 is an open, fail-closed release-governance standard. Between every proposed change and a protected branch it interposes eleven independent checks — the Zero-Point zones — each with a defined input, a deterministic verdict, and machine-readable evidence. The gate is closed by default. It opens only when every zone returns PASS and the complete result is sealed into a Proof-Pack: a single, self-describing, cryptographically signed record that a third party can re-run to obtain the identical verdict.

Crucially for the argument of this paper, the Proof-Pack is verifiable from nothing but the pack and a public key, so trust in the verdict is reconstructed from evidence rather than extended to the pipeline — subject to the honest caveat of §3.1 that the verifier itself must be an independently buildable root. This property — that the mechanism asks to be checked rather than believed — is what makes ZP-11's specification a thing that must be public (§3), and what makes opening its implementation the most reliable route to a category (§4).

Repository (full specification and reference implementation): **https://github.com/iQs-Group/ZP-11**
License: **MIT** · Copyright (c) 2026 Naquib Hatami.