Why ZKS Needs a Standard

The Case for Formalizing Zero-Knowledge Sovereignty

Executive Summary

The Zero-Knowledge Sovereignty (ZKS) Standard is an open technical specification that defines digital sovereignty as a falsifiable architectural property, rather than a matter of policy, intent, or organizational trust. Published by the ZKS Working Group as Candidate Release ZKS-1.0-CR1[1], ZKS moves beyond "end-to-end encryption" marketing claims by establishing rigorous, testable constraints on who can assemble the complete set of components required to decrypt user data.

In a ZKS-compliant system, **data privacy is defined not by assurances or controls, but by the absence of technical capability, and trust is minimized to what can be proven impossible by architecture.** ZKS provides system architects, auditors, and procurement teams with a normative framework for evaluating whether any service provider - or intermediary across the digital supply chain - retains the technical ability to access, derive, compel, or reconstruct decryptability, regardless of organizational policy or legal process.

A common objection arises: *"Isn't this just topology? Why do we need a formal standard for something apparently so simple?"*

This paper argues that the apparent simplicity of ZKS is precisely why rigorous standardization is necessary. It is easy to claim "the provider cannot decrypt." However, proving impossibility requires exhaustive closure of every path to decryptability: collusion, coercion, hidden recovery mechanisms, metadata correlation enabling key targeting, malicious update paths, and aggregation of ciphertext with decryption-relevant components over time. The cost of missing even one path is a permanent loss of sovereignty.

ZKS differs from frameworks such as Zero Trust in the nature of the problem it standardizes:

- **Zero Trust:** operational complexity (multi-component enforcement and continuous policy evaluation) → standardization *improves* interoperability and coverage.
- **ZKS:** constraint complexity (complete closure of decryptability paths) → standardization *ensures* falsifiability and completeness.

This matters not only for individuals, but especially for enterprises and public administrations. ZKS does not prevent organizations from controlling their own data; it ensures that operational access to infrastructure - whether externalized to a vendor or delegated internally - does not automatically confer access to content. In other words, ZKS formalizes separation of duties enforced by architecture, not by policy.

The ZKS Standard exists to make "zero-knowledge" auditable rather than merely claimable, enabling procurement teams to specify requirements, auditors to verify compliance, and organizations to demonstrate that sovereignty claims are grounded in enforceable technical constraints.

[1] https://www.zks-standard.org/

1. Who Is the Provider?

Before examining in detail why ZKS requires a standard, we must clarify a foundational concept that shapes the entire discussion: the definition of “provider.”

1.1 The Provider Is Not Always External

In ZKS terminology, the “provider” is any party that operates infrastructure on which the system runs. This includes:

Provider Type	Examples	ZKS Implication
External cloud vendor	AWS, Azure, Google Cloud, SaaS providers	Most commonly discussed scenario
Internal IT department	Corporate IT, data centre operations	Same architectural constraints apply
Shared services function	Group IT serving multiple business units	Provider to business units, even within same legal entity
Managed service provider	Outsourced IT operations	Third party operating on organisation’s infrastructure
Government shared services	GovCloud, shared agency platforms	Provider to constituent agencies

The critical insight: An organization’s internal IT department is still “the provider” in ZKS terms. The same architectural constraints that protect an organization from an external cloud vendor should protect business data from unnecessary exposure to infrastructure operators - regardless of employment relationship.

1.2 Why Internal Providers Matter

The assumption that internal IT is inherently trustworthy conflates organizational alignment with architectural security. This is not about questioning the integrity of IT staff; it is about removing the burden of toxic liability. If an admin credential is compromised by a hacker, ZKS ensures that administrative access does not become a data exfiltration vector.

Assumption	Reality	ZKS Response
“Our IT staff are employees - they’re trustworthy”	Trust is policy; employees can be compromised, coerced, or make errors	Architecture should not require trust
“We control our own infrastructure”	Operational control ≠ need for content access	Separation of duties should be architectural
“Internal breaches are less likely”	Insider threats account for significant portion of breaches	Architecture limits blast radius regardless of threat source
“We can audit our own staff”	Audit detects violations; architecture prevents them	Prevention superior to detection for confidentiality

ZKS formalizes a principle that security professionals already understand: separation of duties. Infrastructure operators should be able to maintain systems without accessing data content. Database administrators should manage storage without reading records. Network engineers should route traffic without inspecting payloads.

"ZKS ensures that operational access to infrastructure does not automatically confer access to data. This is separation of duties enforced by architecture, not policy."

1.3 Implications for This Document

Throughout this document, "provider" should be read broadly. The arguments for standardization apply equally whether the provider is:

- An external SaaS vendor storing your data in their cloud
- Your own IT department operating on-premises infrastructure
- A shared services function serving multiple business units
- A government agency operating shared platforms for other agencies

The standard exists to define architectural properties that should hold regardless of who operates the infrastructure.

2. Why Existing Standards Are Insufficient

A common response to ZKS is: "We already have standards for this." This section explains why existing frameworks - while valuable - do not address the specific problem ZKS solves.

2.1 Existing Standards Comparison

Standard	What It Addresses	What It Does NOT Address
NIST SP 800-207 (Zero Trust)	Network access control, identity verification, micro-segmentation	Whether the provider can decrypt data; assumes provider infrastructure is part of trust boundary
ISO 27001	Information security management system (ISMS) processes	Architectural constraints; certifies *process*, not *impossibility*
SOC 2 Type II	Operational controls around security, availability, confidentiality	Whether controls can be bypassed; audits *what you do*, not *what you cannot do*
FIPS 140-3	Cryptographic module integrity and operation	Key custody topology; a FIPS-validated system can still have provider-held keys
GDPR Article 32	"Appropriate technical measures" including encryption	Provider non-custodianship; encryption at rest satisfies GDPR even when provider holds keys

2.2 The Critical Gap

All existing security and privacy standards permit architectures in which:

- Data is encrypted at rest using keys controlled or mediated by the service provider
- The provider retains technical pathways to decrypt data for operational, compliance, or legal purposes
- A provider-side compromise may result in exposure of plaintext or decryption capability

These standards regulate safeguards and controls, but do not impose architectural constraints that make provider decryptability technically impossible**. They allow architectures that are compliant with their stated objectives but fundamentally fail the sovereignty test.**

ZKS closes this gap by defining compliance as an *architectural property* - not a policy commitment.

2.3 The "Encryption at Rest" Misconception

Many organizations believe "encryption at rest" provides meaningful protection against provider-side threats. It does not.

Scenario	"Encryption at Rest"	ZKS Protection
Provider-side breach	**None** - provider holds decryption keys	**Full** - attacker obtains only ciphertext
Legal compulsion to provider	**None** - provider can comply by decrypting	**Full** - provider cannot comply; lacks capability
Insider threat at provider	**Limited** - access controls only	**Full** - architecture excludes possibility
Provider policy change	**None** - policy can change unilaterally	**Full** - architecture cannot change without detection

ZKS makes the distinction auditable: "Does the provider hold decryption-enabling material?" is a topology question, not a policy question.

3. The "Wide vs. Deep" Distinction

3.1 Two Types of Complexity

Security frameworks can be complex in different ways:

Dimension	Zero Trust	ZKS
Complexity type	Wide - many assets, policies, identities	Deep - one property, extreme rigor
Analogy	Air traffic control - thousands of planes, thousands of rules	BSL-4 containment - one pathogen, catastrophic if it escapes
Failure mode	Graceful - fix the policy, revoke access	Catastrophic - sovereignty lost permanently
Verification	Continuous operational monitoring	Architectural verification against defined criteria

Both require standards. Zero Trust needs standards for interoperability across thousands of components. ZKS needs standards for **completeness** - ensuring every path to decryptability is closed.

3.2 The Objection Refuted

When someone says *"ZKS is just topology, why standardize?"*, the response is:

> *"It is 'just topology' until you get a subpoena. Zero Trust policies fail gracefully - if a policy is wrong, you fix it. ZKS failures are catastrophic - once the provider decrypts, sovereignty is gone forever. We need the standard not because the architecture is confusing, but because the stakes of getting it wrong are absolute."*

4. The Negative Proof Problem

4.1 Positive vs. Negative Proofs

Zero Trust and ZKS require fundamentally different types of proof:

Type	Framework	Question	Evidence Required
Positive	Zero Trust	"Is Alice authorised?"	Check policy → Yes/No
Negative	ZKS	"Is it impossible for the provider to decrypt?"	Prove no path exists through topology, collusion, coercion, updates, metadata correlation, support tools, recovery mechanisms…

Proving a negative is significantly harder than proving a positive.

4.2 Why This Matters

To prove ZKS compliance, you must demonstrate that **no path to decryptability exists** - not just that the obvious paths are closed, but that *every* path is closed:

- Direct decryption (provider holds keys)
- Collusion (multiple parties combine knowledge)
- Coercion (legal compulsion to attempt decryption)
- Support backdoors (password reset, account recovery)
- Metadata correlation (reconstructing decryption workflow from logs)
- Malicious updates (pushing compromised client code)
- Key recovery (provider-controlled recovery mechanisms)
- Wrapped key coexistence (holding encrypted keys alongside ciphertext)

The standard provides the **exhaustive enumeration** of these paths through its assertions (A1–A12), invariants (INV-1–INV-6), and conformance tests (T-STR, T-OBS, T-CAP, T-REV). Without this enumeration, vendors can claim "zero-knowledge" while leaving critical paths open.

5. The Falsifiability Requirement

5.1 Marketing Claims vs. Testable Assertions

Status	"Zero-Knowledge" Meaning	Verification
Without standard	Marketing claim	Unfalsifiable - "trust us"
With standard	Testable assertion	Falsifiable - audit against defined criteria

5.2 What the Standard Enables

The ZKS Standard makes "zero-knowledge" **scientific** - it can be disproven:

Test	What It Proves	Reference
T-CAP-1	Provider cannot decrypt with full infrastructure access	Section 7.5.2 E2
T-CAP-2	Collusion of all external parties cannot reconstruct decryptability	Section 5.3.6
T-REV-1	Service denial does not revoke decryptability	Section 2.2.6
T-REV-2	UKRS denial does not permanently revoke decryptability	Section 4.10

Without the standard, "zero-knowledge" is just a slogan. With the standard, "ZKS-compliant" is an auditable claim that can be verified or refuted.

6. The Definition of "Possession"

6.1 The Nuance Problem

In Zero Trust, "access" is binary: you either have authorization or you don't.
In ZKS, "possession" is nuanced:

Question	Naive Answer	Standard Answer
Does holding a wrapped key count as possession?	"No, it's encrypted"	**Yes** - coexistence with ciphertext is prohibited regardless of wrapping (Tenet 3)
Does holding a password reset token count?	"No, it's just for recovery"	**Yes** - if it enables key reconstruction, it's a decryption component
Does holding the recovery seed count?	"No, the user controls recovery"	**Depends** - A9 requires UKRS to be cryptographically blind
Does temporary possession during processing count?	"No, it's ephemeral"	**Yes** - even transient possession creates a path

6.2 Why This Matters

Without normative definitions, vendors claim compliance while holding "just the recovery key" - which is effectively everything. The standard provides rigid definitions:

- Complete Set of Components Required to Decrypt (Section 2.2.2)
- Client Sovereignty Domain boundary (Section 3.1)
- Plane separation requirements (Section 3.3)
- Coexistence prohibition (Tenet 3)

These definitions close the loopholes that vague “zero-knowledge” claims leave open.

7. ZKS in Enterprise and Public Administration Contexts

A critical misconception about ZKS is that it prevents organisations from managing their own data. This section clarifies how ZKS supports - rather than impedes - legitimate organisational requirements.

7.1 The Sovereignty Domain Owner Distinction

Context	Sovereignty Owner	Implication
Consumer	Individual user	User holds all keys; provider has no access
Enterprise	Organisation (not individual employees)	Organisation holds master keys; provider has no access
Public Administration	Government entity	Entity holds master keys; cloud provider has no access

ZKS protects the Sovereignty Domain Owner from the *service provider* - not employees from the organisation.

7.2 What ZKS Requires vs. What ZKS Permits

Requirement	ZKS Position
Provider cannot decrypt organisation’s data	**Required** - this defines ZKS
Provider cannot be compelled to provide decryption	**Required** - architectural impossibility
Organisation can decrypt its own data	**Permitted** - internal key management is the organisation’s choice
Organisation can implement internal access controls	**Permitted** - orthogonal to ZKS compliance
Organisation can audit employee activity	**Permitted** - internal capability, not provider capability

7.3 The Administrative Authority Model

ZKS-compliant systems can support legitimate enterprise requirements:

Legal Hold and eDiscovery

- Organization's key custody arrangements allow access to employee data when legally required
- The organization produces records, not the provider
- Provider subpoena yields nothing because provider has nothing

Employee Offboarding

- Organization revokes employee's access credentials
- Organization retains data under organizational keys
- Employees cannot access data after departure; organization can

Internal Audit and Compliance

- Organization's security team can access data for investigations
- Audit trails remain under organizational control
- No dependency on provider cooperation

Incident Response

- Security teams operate within organizational sovereignty domain
- Response does not require provider assistance
- Forensic capability is internal, not outsourced

7.4 Public Administration Requirements

Public sector organizations face unique requirements that ZKS explicitly supports:

Digital Sovereignty and Data Localization

Concern	Traditional Cloud Risk	ZKS Mitigation
Foreign provider access	Provider's jurisdiction may compel access	Provider lacks capability regardless of jurisdiction
Data residency	Data may be processed in foreign jurisdiction	Processing occurs only in Client Sovereignty Domain (local)
Cross-border subpoena	Foreign court orders may extract data	Subpoena to provider yields encrypted material only

ZKS ensures that data sovereignty is *architectural*, not contractual.

Freedom of Information and Records Management

- Organization (the public administration) is the Sovereignty Domain Owner
- Internal key custody ensures the administration can access its own records
- Provider cannot access records - but the administration can
- FOIA requests are satisfied by the administration, not the provider

Continuity of Government

- Organizational keys remain under institutional control
- Individual employee departure does not affect data access
- Key rotation and succession planning are internal matters
- No vendor lock-in on key custody

8. Hidden Paths: Why "Simple Topology" Fails

8.1 The Implementation Gap

A "simple" topological separation is easy to draw on a whiteboard. It is hard to implement without creating accidental backdoors.

The standard identifies and closes paths that naive implementations miss:

Hidden Path	Naive Assumption	Standard Closure
Support backdoor	"We separated keys from data"	Section 5.3.8: Sovereignty Failure
Metadata leak	"Databases are separate"	A7: Metadata minimisation
Update trap	"Server topology is secure"	Section 5.3.1: Binary Transparency
Recovery backdoor	"User controls recovery"	A9: UKRS cryptographic blindness
Wrapped key coexistence	"Keys are encrypted"	Tenet 3: Coexistence prohibition
Orchestration leak	"OP just routes metadata"	A5: OP Key-Plane blindness
Cryptographic decay	"AES-256 is sufficient"	Section 5.3.2: Shelf-life
Supply chain compromise	"We built it securely"	Binary Transparency; reproducible builds for ZKS-Enterprise
Jurisdictional compulsion	"We're outside that jurisdiction"	Provider cannot comply even if compelled - capability, not policy
Insider threat at provider	"We have access controls"	Architectural exclusion; insider has no keys to steal
Acquisition/merger	"Current management is trustworthy"	New owner inherits architectural constraints, not policy commitments
Discontinued service	"We'll migrate before shutdown"	User retains keys; ciphertext is portable
Rogue support engineer	"Support cannot access customer data"	Architecture enforces this; no policy override possible

"Simple" topology doesn't close these paths. The standard does.

9. The Catastrophic Failure Mode

9.1 Recoverable vs. Permanent Failures

Aspect	Zero Trust Failure	ZKS Failure
Example	Unauthorised access to resource	Provider decrypts user data
Detection	Audit logs reveal anomaly	May never be detected
Recovery	Revoke access, update policy	**None** - plaintext is exposed
Reversibility	Fully recoverable	**Permanent** - cannot un-expose
Blast radius	Single session/resource	All historical data

9.2 Why This Demands Rigor

> *"Zero Trust failures are recoverable - revoke access, update policy, move on. ZKS failures are permanent - once plaintext is exposed, you cannot un-expose it. The standard exists because the cost of getting it wrong is irreversible."*

9.3 Quantifying the Sovereignty Difference

The difference between ZKS-compliant and non-compliant architectures can be quantified in breach scenarios:

Factor	Non-ZKS Breach	ZKS Breach
Data exposure scope	**All historical data** - attacker obtains plaintext	**None** - attacker obtains ciphertext only
Notification requirement	**Mandatory** - personal data breach	**May not apply** - encrypted data may not constitute "breach"
Regulatory penalty exposure	**Full** - failure to implement adequate measures	**Mitigated** - encryption measures were adequate
Class action exposure	**High** - affected individuals have standing	**Low** - no demonstrated harm
Remediation cost	**High** - credit monitoring, identity protection	**Minimal** - no exposed personal data
Reputational damage	**Severe** - "your data was stolen"	**Limited** - "encrypted data was accessed"

The financial case for ZKS: Architectural sovereignty transforms breach economics.

10. The Topological Verification Advantage

10.1 Two Verification Models

Aspect	Zero Trust	ZKS
Verification type	Continuous operational	Architectural
What you audit	Logs, policies, behaviour	Topology diagram
Frequency	Ongoing	At design time + changes
Expertise required	Security operations	Security architecture

10.2 The Standard's Role

> *"Zero Trust requires continuous operational verification - you audit logs, review policies, monitor behavior. ZKS requires architectural verification - you examine a topology diagram. The standard defines what that diagram must show. Without it, you're trusting the diagram is complete. With it, you can verify."*

The standard defines:

- What components must be mapped (T-STR-1)
- How planes must be aligned (T-STR-2)
- What evidence proves impossibility (Section 7.5)
- What tests verify compliance (Appendix D)

This makes ZKS easier to audit than Zero Trust in one sense, but only if you have a standard that defines what the topology must demonstrate.

11. Comparison Summary

Feature	Zero Trust Complexity	ZKS Complexity
Nature	Operational - managing sprawl across thousands of assets	Constraint - rigorously closing every path to decryption
Goal	"Filter the noise"	"Contain the risk"
Failure mode	Graceful degradation	Catastrophic and permanent
Why standardise?	Ensure interoperability and coverage	Ensure falsifiability and completeness
Analogy	Air Traffic Control - complex because many moving parts	BSL-4 Lab - complex because a single leak is catastrophic

12. The Standard Exists Because…

Reason	Explanation
Proving negatives is hard	The standard defines what constitutes evidence of impossibility
Hidden paths are subtle	The standard enumerates and closes them systematically
"Possession" is nuanced	The standard defines what counts as controlling decryption components
Failures are catastrophic	The standard ensures rigor commensurate with irreversible risk
"Zero-Knowledge" must be falsifiable	The standard makes it auditable, not just claimable
Existing standards don't address this	Current frameworks permit provider decryption capability
Enterprises need clear requirements	The standard enables procurement specification and vendor evaluation
Regulators need audit criteria	The standard provides testable compliance verification

13. Conclusion

The objection "ZKS is just topology - why do we need a standard?" misunderstands the nature of the challenge.

Simplicity of concept does not mean simplicity of verification.
It is simple to say "the provider cannot decrypt." It is hard to prove it - you must close every path:

- Collusion between external parties
- Coercion through legal process
- Support backdoors and password resets
- Metadata correlation attacks
- Malicious client updates
- Provider-controlled key recovery
- Wrapped key coexistence
- Orchestration plane leakage
- Cryptographic shelf-life decay
- Supply chain compromise
- Jurisdictional compulsion
- Insider threats at the provider

The standard exists not because the architecture is confusing, but because **proving impossibility requires exhaustive rigor**. Every assertion, every invariant, every conformance test closes a path that a naive "simple topology" implementation would leave open.

Policy and operational summary

For enterprises and public administrations, ZKS provides:

- Architectural separation of duties between infrastructure operations and data access
- Clear procurement criteria for vendor evaluation
- Reduced regulatory exposure in breach scenarios
- Data sovereignty that is architectural, not merely contractual
- Compatibility with legitimate organisational control requirements

The cost of missing one path is not a recoverable policy violation, it is permanent loss of sovereignty.

That is why ZKS needs a standard.

Appendix A: Regulatory Mapping

ZKS compliance supports, but does not guarantee, regulatory compliance. This mapping clarifies the relationship.

Regulation	Relevant Requirement	ZKS Contribution
GDPR Article 32	"Appropriate technical measures"	ZKS exceeds minimum requirements; provides architectural data protection
GDPR Article 34	Breach notification (with encryption exception)	ZKS may qualify for encryption safe harbour
HIPAA Security Rule	Access controls, encryption	ZKS enforces access controls architecturally
HIPAA Breach Notification	"Unsecured PHI" triggers	ZKS-protected PHI may not constitute "unsecured"
NIS2 Directive	Supply chain security, incident response	ZKS reduces supply chain exposure; limits incident scope
DORA	ICT risk management, third-party risk	ZKS reduces third-party risk to availability only
Schrems II	Data transfer safeguards	ZKS provides technical measure against foreign surveillance

Important caveat: ZKS is a technical standard, not a compliance certification. Regulatory compliance requires organizational measures beyond architecture.

Appendix B: Procurement Language

Organizations can include the following language in procurement documents:

Mandatory Requirements

Architectural Non-Custodianship: The vendor shall demonstrate that no component of their infrastructure - including storage systems, orchestration services, key management systems, and support tools - possesses the technical capability to decrypt customer data. This shall be an architectural property, not a policy commitment.

Falsifiable Verification: The vendor shall provide evidence that their non-custodianship claims can be independently verified through architectural inspection, not merely through policy documentation or attestation.

Binary Transparency: The vendor shall log all client software distributions to a public, append-only transparency log, enabling customers to verify that the software they receive matches publicly committed versions.

Evaluation Criteria

Criterion	ZKS-Aligned Response	Non-Compliant Response
"Can your employees access our data?"	"No - we lack the architectural capability"	"No - our policies prohibit this"
"Can you comply with a court order to produce our data in plaintext?"	"No - we cannot produce what we cannot decrypt"	"We would challenge the order but could technically comply"
"What happens to our keys if you go out of business?"	"You already have them; we never did"	"We would provide escrow recovery"
"How do you prevent rogue employee access?"	"Architecture makes it impossible"	"Access controls and monitoring"

DOCUMENT INFORMATION

Version History

Version	*Date*	*Changes*
1.0	November 2025	Initial release
2.0	January 2026	Integrates new material addressing enterprise, public administration, and policy concerns.

Status: Position Paper - Prepared for Public Comment

Audience: CISOs, Security Architects, Standards Bodies, Procurement Teams, Policy Makers

References

1. NIST SP 800-207: Zero Trust Architecture
https://nvlpubs.nist.gov/nistpubs/specialpublications/NIST.SP.800-207.pdf
2. ISO/IEC 27001: Information Security Management Systems
https://www.iso.org/standard/27001
3. SOC 2: Trust Services Criteria
https://en.wikipedia.org/wiki/System_and_Organization_Controls
4. FIPS 140-3: Security Requirements for Cryptographic Modules
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-3.pdf
5. GDPR: General Data Protection Regulation
https://gdpr-info.eu/
6. HIPAA: Health Insurance Portability and Accountability Act
https://www.hhs.gov/hipaa/index.html
7. NIS2: Network and Information Security Directive
https://digital-strategy.ec.europa.eu/en/policies/nis2-directive
8. DORA: Digital Operational Resilience Act
https://eur-lex.europa.eu/eli/reg/2022/2554/oj/eng

Acknowledgments

This analysis benefited from public documentation and frameworks published by the National Institute of Standards and Technology (NIST), the International Organization for Standardization (ISO), and the Internet Engineering Task Force (IETF). We acknowledge the ongoing work of the cryptographic research community in defining the boundaries of verifiable sovereignty.

Contact

For technical inquiries regarding this document:

- ZKS Working Group - zks.working.group@zks-standard.org

Legal Notice

Disclaimer: This document is provided for informational purposes only and does not constitute legal, financial, or technical advice. The ZKS Working Group makes no warranties, express or implied, regarding the accuracy, completeness, or fitness for a particular purpose of the information contained herein.

No Liability: In no event shall the authors or publishers be liable for any damages, including but not limited to direct, indirect, special, or consequential damages, arising out of or in connection with the use of this document.

Document Classification: Public