DP15 – Security and Provenance

1. Purpose of This Draft

This draft articulates Desirable Property 15 (DP15) as the condition under which security is structural and provenance is inspectable, so participants and communities can verify what happened, who did it, under which policy, and whether artifact integrity holds.

DP15 connects DP1 (accountability), DP4 (data handling evidence), DP11–DP13 (AI disclosure and containment), DP14 (trust and transparency), DP7 (interoperability of signed exports), and DP16 (roadmap honesty about security posture).

If DP15 is weak, predictable failures follow: silent breaches, undetected tampering, unverifiable AI outputs, supply-chain compromise, and governance decisions without evidence chains.

DP15 does not require every participant to understand cryptography. It defines the minimum conditions under which strong claims are backed by strong, accessible evidence.

2. Problem Statement

In today's web, security is often reactive and provenance is optional.

Systems are patched after breaches, logs are controlled by operators, and downstream tools discard evidence needed to verify origin or integrity. Participants are asked to trust without meaningful ways to verify.

This produces recurring failures:

- integrity breaks without participant-visible signals
- opaque build and runtime supply chains
- AI outputs unanchored to sources or model versions
- moderation and policy actions that cannot be independently verified
- exports stripped of signatures or provenance context

These failures are structural. Trust without verification scales fraud.

DP15 reframes security as continuous assurance and provenance as default metadata carried across systems.

3. Threats and Failure Modes

3.1 Key and signing centralization

A single operator controls signing keys.

Example: A platform can forge logs or artifacts without independent verification.

Why this matters: High-stakes systems require distributed trust or independent witnessing.

3.2 Tampered content and deepfakes

Synthetic or altered media appears authentic without verifiable origin.

Example: A fake executive video triggers financial action.

Why this matters: Provenance must bind claims to verifiable evidence.

3.3 Opaque supply chains

Dependencies and build processes are not visible or verifiable.

Example: A compromised package leaks secrets undetected.

Why this matters: Supply-chain integrity is a systemic dependency.

3.4 Policy–execution gap

Security claims do not match actual system behavior.

Example: Promised encryption is disabled or bypassed in support workflows.

Why this matters: Security must be verifiable, not declarative.

3.5 Log laundering

Logs can be edited or selectively disclosed.

Example: Appeals fail because events cannot be independently verified.

Why this matters: Accountability requires tamper-evident records.

3.6 AI output ambiguity

AI-generated outputs lack traceability.

Example: A model produces authoritative-seeming content with no source or version context.

Why this matters: AI outputs must be bound to retrievable evidence where claims are made.

3.7 Coordinated multi-vector attacks

Attackers combine multiple weaknesses across identity, content, and infrastructure to create convincing but false realities.

Example: A coordinated campaign introduces a malicious dependency (supply chain), uses AI to generate plausible documentation and endorsements, and distributes signed-looking artifacts through compromised or misrepresented keys.

Why this matters: Individual protections (signatures, logs, provenance) can appear valid in isolation but fail when combined in adversarial scenarios.

Failure mode: **composed deception**, where multiple weak signals reinforce each other into a credible but false system state.

3.8 Insider + AI + supply-chain compromise

Trusted insiders or compromised accounts introduce changes that appear legitimate, amplified by AI-generated artifacts and opaque dependencies.

Example: An insider approves a malicious update, AI generates supporting documentation and changelogs, and downstream systems accept the update because provenance appears complete but is internally compromised.

Why this matters: Trust anchored only in roles or signatures is insufficient without independent verification, multi-party validation, and anomaly detection.

Failure mode: **trusted-path compromise**, where legitimate authority is used to introduce unverifiable or malicious changes.

3.9 Replay and rollback attacks

Old but valid artifacts, logs, or attestations are reused to mask current compromise or regress system state.

Example: A previously signed and valid build is redeployed after a security patch, bypassing protections while appearing authentic.

Why this matters: Provenance must include temporal context and revocation awareness.

Failure mode: **temporal deception**, where authenticity is preserved but relevance is not.

3.10 Cross-system degradation exploitation

Integrity and provenance are lost or weakened as artifacts move across systems, enabling tampering without detection.

Example: A signed artifact is exported into a system that strips signatures, then modified and reintroduced without clear indication of lost provenance.

Why this matters: Cross-system integrity must be preserved or explicitly degraded with visible signaling.

Failure mode: **silent degradation**, where users assume guarantees that no longer hold.

4. Core Principle

Security and provenance in the meta-layer mean that critical actions, artifacts, and automated behaviors are integrity-protected, attributable, and explainable at useful depth.

Participants and auditors must be able to verify chains of custody for data, software, and AI outputs.

Example: A document includes content hash, signing keys, model version, and policy state at time of generation.

What this feels like: Serious claims carry verifiable receipts.

Without this: Trust collapses into presentation rather than evidence.

5. Primary Mechanisms and Structural Conditions

5.1 Authenticity and integrity by default

Critical artifacts are signed, hashed, and versioned where feasible, with **clear trust anchors** (DP1) and **verifier pathways**.

Systems MUST define:
- which artifacts require signatures (high-stakes vs low-stakes)
- accepted signature schemes and key provenance
- how verification is performed at read time and export time

Verification UX should surface: valid, invalid, unknown signer, or unverifiable.

Failure modes:
- **implicit trust** (unsigned artifacts treated as authoritative)
- **verification bypass** (signatures present but not checked)

5.2 Software supply-chain transparency

Systems publish dependency inventories and build provenance (e.g., SBOMs) with **linkage to build outputs and runtime artifacts**.

This includes:
- dependency versions, sources, and integrity hashes
- build pipelines with reproducibility or attestations
- mapping from source → build → deployable artifact

Verification MUST allow independent parties to confirm that shipped artifacts correspond to declared inputs.

Failure modes:
- **opaque dependencies**
- **non-reproducible builds** that cannot be audited

5.3 Runtime attestation

Execution environments provide attestations that policies are enforced as declared, including **configuration state, policy versions, and integrity measurements**.

Systems SHOULD support:
- remote attestation for critical services
- linkage between attestation and policy claims (DP8)

Verification MUST allow participants to detect divergence between declared and actual runtime state.

Failure modes:
- **policy–execution gap**
- **stale attestations** reused beyond validity

5.4 AI output provenance

AI outputs include model identifiers, relevant inputs or retrieval sources, and policy context, bounded by privacy constraints.

This includes:
- model/version identifiers and evaluation context
- prompt/retrieval lineage where claims are made
- safety/policy configuration at generation time (DP12 linkage)

Verification MUST distinguish:
- attributable outputs vs. non-attributable outputs
- grounded claims vs. ungrounded generation

Failure modes:
- **attribution ambiguity**
- **capability inflation** via unverifiable claims

5.5 Tamper-evident logs

Moderation, governance, and system events are recorded in append-only or verifiable logs with **ordering guarantees and witnessability**.

Systems SHOULD support:
- append-only structures (e.g., Merkle trees)
- external witnesses or checkpoints
- inclusion proofs for specific events

Verification MUST allow participants to confirm that events were not removed or altered.

Failure modes:
- **log laundering**
- **selective disclosure** of events

5.6 Key management and recovery

Systems implement key rotation, compromise handling, and transparent custody models with **documented ownership and recovery pathways**.

This includes:
- rotation schedules and revocation mechanisms
- multi-party custody for high-risk keys
- recovery procedures with audit trails

Verification MUST surface key status (active, rotated, revoked) at use time.

Failure modes:
- **stale keys** used after compromise
- **opaque custody** enabling unilateral control

5.7 Red-team and disclosure practices

Security issues are surfaced through coordinated disclosure, bug bounties, and public postmortems with **remediation evidence and timelines**.

Systems SHOULD maintain:
- disclosure policies with response SLAs
- postmortems linking vulnerabilities → fixes → verification

Verification MUST allow participants to see whether issues were resolved and how.

Failure modes:
- **silent remediation** without accountability
- **recurring vulnerabilities** without learning

5.8 Provenance envelopes

Artifacts carry structured metadata describing origin, transformations, and policy context across systems.

Envelopes SHOULD include:
- origin identity and signatures (DP1)
- transformation steps and handlers
- policy and consent context (DP4)

Verification MUST preserve envelopes across export/import and signal degradation when fields are lost.

Failure modes:
- **provenance stripping** across systems
- **semantic drift** of metadata

5.9 Verification UX

Verification tools are accessible without requiring expert knowledge, with **progressive disclosure** for advanced users.

Systems MUST provide:
- simple indicators for common cases (valid/invalid/unknown)
- drill-down views for signatures, chains, and evidence
- clear explanations of limits of verification

Failure modes:
- **expert-only verification** (users cannot act on signals)
- **misleading indicators** that overstate certainty

5.10 Cross-system verification

Provenance and signatures remain valid and interpretable across platforms and tools (DP7 alignment), with **explicit degradation signaling**.

This requires:
- standard formats for signatures and provenance
- mapping rules when systems differ
- signaling when verification cannot be preserved

Verification MUST not silently drop integrity guarantees during transfer.

Failure modes:
- **silent degradation** of verification across boundaries
- **incompatible formats** preventing validation

5.11 Security & Provenance System Layer: Trust Anchoring, Verification Flow, and Adversarial Resilience

DP15 requires a coherent system layer that binds identity (DP1), data (DP4), governance (DP8), and AI (DP12) into a **continuous verification lifecycle**.

Core properties:
- **Trust anchoring:** identities and keys are attributable, with clear custody and revocation states
- **End-to-end verification flow:** artifacts can be verified at creation, storage, transmission, and use
- **Propagation:** provenance travels with artifacts; loss is explicitly signaled
- **Auditability:** logs, receipts, and attestations allow reconstruction of events
- **Adversarial resilience:** the system detects and contains tampering, forgery, and replay

Verification lifecycle:
1. **Creation:** artifact is signed/hashed with provenance envelope
2. **Distribution:** integrity preserved; intermediaries cannot alter without detection
3. **Consumption:** client verifies signatures, keys, and policy context
4. **Audit:** logs and receipts enable retrospective verification and dispute

Failure modes:
- **broken chain of custody** between stages
- **replay/rollback attacks** using old but valid artifacts
- **cross-system loss** of provenance without signaling

This layer ensures security claims are not merely present but **continuously checkable under real conditions**.

6. Governance, Accountability, and Agency Surfaces

DP15 requires that security and provenance are not only visible but **actionable within governance and participant agency systems (DP8, DP2)**.

Participants must be able to:

- verify high-stakes claims and distinguish verified vs. unverifiable states
- access evidence supporting decisions that affect them
- detect integrity failures (invalid signatures, missing provenance, log inconsistencies)
- submit challenges or disputes with attached evidence

When verification fails, systems MUST:

- clearly signal failure states (invalid, missing, degraded, unverifiable)
- prevent high-risk actions from proceeding without override acknowledgment
- preserve evidence for audit and dispute resolution

Communities and governance bodies must be able to:

- define assurance thresholds for zones (e.g., require signed artifacts, attested environments)
- reject or downgrade artifacts lacking required provenance
- trigger audits when inconsistencies or anomalies are detected
- require remediation (re-signing, re-verification, disclosure, rollback)

Systems SHOULD support:

- escalation pathways tied to specific artifacts or events
- annotation and commentary layers on provenance and logs
- integrity-based gating (e.g., cannot promote, fund, or amplify unverifiable outputs)
- confidence or assurance scoring for artifacts and systems

Failure modes include:

- **non-actionable verification** (signals exist but do not affect outcomes)
- **accountability gaps** (no mechanism to enforce correction)
- **verification fatigue** (users overwhelmed and stop checking signals)

7. Incentives and Power Analysis

Security and provenance exist within strong incentive gradients that often discourage their full implementation.

Key structural tensions include:

- **speed vs assurance:** teams are rewarded for shipping quickly, not for making systems verifiable
- **opacity vs accountability:** operators benefit from controlling logs and narratives
- **growth vs integrity:** unverifiable claims can accelerate adoption
- **cost externalization:** security failures impose costs on users and ecosystems, not just builders

Adversarial actors exploit these tensions:

- attackers rely on weak provenance to inject malicious artifacts
- coordinated misinformation exploits unverifiable content
- insiders may bypass controls for convenience or advantage
- AI systems can generate plausible but unverifiable outputs at scale

DP15 requires realignment of incentives such that:

- unverifiable outputs carry reduced trust, reach, or eligibility
- verified artifacts gain preferential treatment in governance, ranking, or funding (DP9 linkage)
- repeated integrity failures trigger governance review or restrictions

Systems SHOULD incorporate:

- cost imposition for unverifiable or tampered artifacts (rate limits, gating)
- rewards for maintaining strong provenance (priority processing, trust weighting)
- visibility into integrity posture (public assurance signals)

Failure modes include:

- **security underinvestment** due to misaligned incentives
- **credibility arbitrage** where actors benefit from unverifiable claims
- **attacker asymmetry** where defense costs exceed attack costs

8. Community Signals Informing DP15

- fatigue with unverifiable claims
- demand for signed exports and build transparency
- concern about AI hallucination presented as fact

- calls for meaningful postmortems after failures

9. Non-Goals and Explicit Boundaries

DP15 does not:

- require all content to be signed or verified
- eliminate all risk
- replace legal or forensic systems
- mandate a specific cryptographic approach

10. Minimum Alignment (Non-Normative)

Minimum alignment defines the threshold at which security and provenance become **reliable verification infrastructure rather than optional features**.

A system claiming DP15 alignment MUST satisfy the following conditions:

10.1 Artifact Integrity

- Critical artifacts MUST be signed or otherwise integrity-protected
- Systems MUST verify integrity at use time, not only at creation

Failure mode: **implicit trust of unverified artifacts**

10.2 Provenance Availability

- High-stakes artifacts MUST carry provenance metadata (origin, transformation, context)
- Systems MUST signal when provenance is missing or degraded

Failure mode: **provenance absence or stripping**

10.3 Verification Pathways

- Participants MUST be able to verify claims without specialized expertise
- Systems MUST provide both simple indicators and detailed inspection paths

Failure mode: **unusable verification systems**

10.4 Tamper-Evident Logging

- Critical events MUST be recorded in tamper-evident logs
- Logs MUST support independent verification or witnessing

Failure mode: **log manipulation or selective disclosure**

10.5 Key and Identity Integrity

- Signing keys MUST have visible ownership, rotation, and revocation status
- Systems MUST detect and signal compromised or invalid keys

Failure mode: **undetected key compromise**

10.6 Cross-System Preservation

- Integrity and provenance MUST persist across exports and integrations
- Systems MUST signal degradation when preservation is not possible

Failure mode: **silent integrity loss across systems**

10.7 AI Output Traceability

- AI-generated outputs MUST be labeled and include relevant provenance when claims are made
- Systems MUST distinguish attributable vs. non-attributable outputs

Failure mode: **AI attribution ambiguity**

10.8 Auditability and Contestability

- Participants MUST be able to inspect, compare, and challenge evidence
- Systems MUST preserve records necessary for dispute resolution

Failure mode: **evidence opacity or loss**

Systems that omit verification, provenance, or auditability MUST NOT be considered aligned with DP15.

11. Open Questions and Future Work

- balancing privacy with auditability
- scaling verification without overwhelming users
- cross-border evidence standards
- decentralized witnessing vs cost tradeoffs
- quantum-resilient cryptography planning

12. Relationship to Other Desirable Properties

- DP1: attribution and appeals
- DP4: integrity of data handling and exports

- DP7: interoperability of signed artifacts
- DP11–DP13: AI traceability and containment
- DP14: transparency practices
- DP16–DP17: roadmap and funding alignment for security

13. Foresight and Failure Design

DP15 assumes breach, key compromise, and manipulation attempts will occur.

Systems must predefine response pathways, including key rotation, participant notification, and audit procedures.

14. Path Toward ML-RFC

- standardize provenance formats for content and AI outputs
- align with emerging supply-chain and identity standards
- develop reference implementations for high-assurance zones

15. Closing Orientation

DP15 is where the meta-layer moves from claims to evidence.

Security without provenance is performative. Provenance without security is unreliable.

Together, they create a system where shared reality can be verified, contested, and trusted.