# HOLogram

## Human Obfuscation Layer

**AI sees a reflection, never the original**

**Whitepaper**

**Version 1.0 (March 2026) - BaffiSan**

**info@hologramprotocol.org**

**www.hologramprotocol.org**

# Abstract

The evolution of artificial intelligence has made it possible to identify and profile human beings not through what they explicitly declare, but through their digital behavior: typing speed, micro-pauses between keystrokes, pointer trajectories, scroll dynamics, and the way they interact with buttons, links, and content.

These characteristics—known as behavioral biometrics—have become unique, persistent, and extremely difficult to alter.

Behavioral profiling occurs silently and continuously, enabled by simple scripts embedded in web pages and by AI models capable of reconstructing personal patterns with high accuracy.

Users have no tools to defend themselves: modern browsers protect against device fingerprinting, but not against fingerprinting of human behavior.

HOLogram introduces a new protection paradigm: a Human Obfuscation Layer, a local system that obfuscates, normalizes, and reduces the granularity of behavioral signals, preserving the natural feel of interaction while preventing identification and advanced profiling by AI systems.

The HOLogram protocol adopts a human-centric, transparent approach aligned with the principles of digital rights, operating entirely locally and without collecting or transmitting user data.

HOLogram is presented as an open standard and as the foundation for an ecosystem of decentralized tools and implementations, supported by a non-financial reputation model (HOLoToken) based on a Proof-of-Contribution logic.

The objective is to grant individuals a new digital right:

**the right not to be recognized by their behavior**.

# Index

# 1. Introduction

Human interaction with the web has always had two faces: what the user *intends* to communicate (the content) and what their body *unintentionally* reveals (their behavior).

In the age of Artificial Intelligence, this second dimension has become central: systems observe how we type, how we move the pointer, how we scroll pages, how hesitations and micro-pauses reoccur over time.

These patterns form a type of **behavioral biometrics**: a unique, stable, and difficult-to-alter signature capable of linking an interaction to a specific individual—even in the absence of cookies, logins, or traditional fingerprinting.

This transformation has created a **protection gap**.

Existing solutions—VPNs, private browsing, device anti-fingerprinting defenses—protect the environment, not the human being.

They can hide an IP address, homogenize browser APIs, or limit hardware attribute collection; they do **not** shield the way a hand moves or the moment a mind pauses before pressing Enter.

As a result, the user remains exposed: identifiable, correlatable, and often profiled far beyond their awareness.

**HOLogram** was created precisely to fill this gap.

It is a **Human Obfuscation Layer**: a local, transparent, user-controlled system that obfuscates, normalizes, and reduces the granularity of behavioral signals before they reach page scripts or the AI models analyzing them.

The goal is not to “deceive” the web, but to **rebalance** the relationship between person and platform: enabling natural interaction without implicitly surrendering one’s behavioral identity.

The protocol is built on a set of simple principles:

- **Human-centric privacy**: protection applies to the human, not just the device;
- **Local only**: no data leaves the terminal; no telemetry, no retention;
- **Transparency & control**: the user sees and chooses the applied obfuscation level;
- **Neutrality**: HOLogram does not manipulate decisions; it only reduces the fidelity of exposed signals;
- **Openness**: it is a standard; implementations may be multiple and independent.

The promise is simple: preserve the naturalness of interaction, while ensuring that observers receive a **less identifying** version of the original signals.

In other words, a **hologram of the behavior**: visible, functional, usable—but not suitable for reconstructing the identity of the person generating it.

HOLogram is also conceived as an ecosystem infrastructure.

The document defines the minimal components (input vault, pointer/scroll normalizer, session-consistent persona mixer, behavioral exposure budget) while leaving room for variants and forks that extend or specialize its behavior across different contexts (browsers, operating systems, embedded environments).

To support decentralized and meritocratic growth, the project includes a non-financial contribution-recognition system—**HOLoToken**, based on Proof of Contribution—designed to give visibility and voice to those who build the protocol.

This whitepaper does not present a product, but **a rule of the game**:

a minimal protocol meant to restore the individual's right not to be recognized by their own behavior.

Everything else—code, optimizations, porting, integrations—belongs to the community.

## 2. The Problem

A person's digital identity is not composed solely of the data they intentionally provide—name, email address, preferences—but also by a series of **involuntary signals** that emerge during interaction with a device.

Every user has a unique way of typing, moving the pointer, scrolling a page, selecting a graphic element.

These patterns, once considered background noise, are now recognized as **behavioral biometrics**: a precise and difficult to alter individual signature.

The widespread adoption of Artificial Intelligence models trained on high frequency input events has transformed these signals into a highly effective recognition tool.

A few milliseconds between keystrokes, tiny variations in mouse curvature, micro pauses during decision making—these are enough to reconstruct a stable identity profile.

Unlike cookies—which can be deleted—or IP addresses—which can be masked—behavioral biometrics are intrinsic to the person: they manifest every time someone uses a computer.

The issue is not only technical but structural.

These signals:

- are collected silently, through simple page scripts;
- require no explicit consent or user interaction;
- can be combined and correlated across different sites;
- enable identification even in private mode, via VPN, or through anonymous networks;
- are used to classify emotional states, error propensity, hesitation, vulnerability.

Unknowingly, the human user reveals information they cannot control: the rhythm of their hands, how fast they react, how they explore a screen.

A platform observing these signals can distinguish one person among millions of seemingly anonymous users, reconstruct their presence over time, and associate behaviors with real identities—even without explicit data.

To this vulnerability is added a second problem: **a power asymmetry**.

Users lack native tools to protect themselves.

No browser shields human behavior.

Existing solutions—VPNs, private mode, anti tracking, third party blocking—protect the **device**, not the **digital body**.

Even systems designed to preserve anonymity at the network or hardware fingerprinting level do not intervene on behavioral dimensions.

The result is that the user is "anonymous" only in appearance:

their behavioral identity remains exposed, stable, recognizable.

Those who observe can determine whether an access originates from the same person who visited another site, infer habits, digital circadian rhythms, cognitive capacity, or even assess momentary conditions such as fatigue, stress, or hesitation.

The absence of a mechanism capable of inserting a protective layer between the human and the observing AI constitutes a **critical gap in modern privacy architecture**.

Behavioral biometrics—now used in anti fraud tools, security systems, commercial platforms, and advertising networks—represent a **continuous form of surveillance** that citizens cannot control or mitigate.

The problem is simple in definition yet enormous in consequences:

every user carries a behavioral fingerprint they cannot avoid revealing, and the modern web is designed to capture and convert it into structured information.

**HOLogram** was created to address precisely this vulnerability:

to restore the individual's ability to interact with digital environments *without exposing their behavioral signature*.

It is a proposal to rebalance a now uneven relationship, reducing the surface of involuntary observation and restoring a margin of self determination in everyday web interactions.

# 3. Objectives of the HOLogram Protocol

HOLogram is designed with a clear purpose: to restore human control over behavioral identity within digital environments.

In a context where AI models can infer personal information by observing how a user interacts—not merely what they communicate—there is a growing need for a new layer of protection aimed at safeguarding the most unconscious and fragile component of online presence: behavior.

The core objectives of the protocol are outlined below..

## 3.1 Protecting the User's Behavioral Identity

HOLogram aims to prevent the natural micro-signals produced by users (typing dynamics, pointer movements, reaction timings) from being exploited to:

- Uniquely identify an individual;
- Track their presence across multiple platforms;
- Reconstruct psychological or behavioral profiles;
- Infer sensitive characteristics (stress, hesitation, vulnerability).

The goal is not to eliminate human interaction, but to prevent it from being used as a behavioral fingerprint.

## 3.2 Obfuscating Without Disrupting the User Experience

Protection must not degrade usability.

HOLogram is designed to:

- Preserve the natural feel of human input;
- Maintain full functionality of web applications;
- Reduce the granularity of exposed behavioral data without generating "artificial movements";
- Prevent the obfuscation layer itself from becoming an identifying signal.

The objective is a discreet, invisible, and harmonious protective layer.

## 3.3 Establishing a Local, Independent, and Transparent Protection Model

HOLogram operates entirely locally—with no data transmission, no telemetry, and no reliance on external services.

Its guarantees include:

- Full user autonomy;
- No third-party dependencies;

- No collection of behavioral signals;
- Complete transparency through a minimal HUD.

Protection belongs to the user—not to the browser, the website, or any provider.

## 3.4 Counteracting AI-Driven Behavioral Profiling

Modern models can reconstruct identities, cognitive patterns, and decision tendencies by analyzing user interaction data.

HOLogram aims to:

- Reduce the amount of "useful behavioral information" exposed;
- Prevent cross-site correlations;
- Degrade the accuracy of behavioral classifiers;
- Rebalance the relationship between users and observational AI models.

The objective is to reduce the effectiveness of invisible surveillance, not to bypass it with deceptive tactics.

## 3.5 Creating an Open Standard for Behavioral Privacy

HOLogram is not a product—it is a protocol.

Its goal is to define a minimal, clear, and implementable standard that can be adopted by:

- Browsers;
- Extensions;
- Operating systems;
- Applications;
- Open-source communities;
- High-sensitivity environments (journalism, activism, public institutions).

An open standard expands the protection surface and eliminates dependency on single actors.

## 3.6 Enabling a Decentralized and Meritocratic Ecosystem

HOLogram is intended to evolve through distributed contributions rather than a central authority.

To support this, it introduces a non-financial recognition system—HOLoToken (Proof of Contribution)—which enables:

- Recognition of individual contributors;
- Transparent tracking of technical contributions;
- Creation of roles and responsibilities within the community;
- Support for an open, evolutionary, and resilient model.

The objective is for HOLogram to grow through collective effort, as has happened with major open infrastructures of the past.

## 3.7 Establishing a New Digital Right: Behavioral Privacy

The ultimate goal is to recognize and protect something that today lacks formal definition but profoundly impacts personal freedom:

**the right not to be identified through one's online behavior**.

HOLogram seeks to introduce a paradigm shift—from protecting the device to protecting the human being—opening the way to a new layer of **digital dignity**.

# 4. Foundational Principles

HOLogram is built upon a set of principles that define its identity, ethics, and operational boundaries. These principles form the structural core of the protocol: they guide its design, restrict what the Layer can do, and establish what it cannot—and must never—do. They represent the intersection between technical architecture, user protection, and respect for digital freedom.

## 4.1 Human-Centric Privacy

The protocol is designed to protect the human being—not the device they use nor the network through which they communicate.

Behavioral biometrics are a personal characteristic that emerges involuntarily; protecting them means protecting the individual.

HOLogram introduces a layer that operates directly where behavior is generated, preventing it from being transformed into identifying information

## 4.2 Local-Only & Zero-Retention

All operations occur on the user's device.

HOLogram does not transmit data to external servers, does not synchronize, store, retain behavioral histories, or record logs.

The Layer is:

- a filter;
- not a collector;
- a buffer;
- not a repository.

Once the obfuscated signal has been transmitted to the browser, the Layer retains no trace of the original data.

## 4.3 Minimization of Traceability

Protection does not rely on aggressive blocking, but on the progressive reduction of the fidelity of behavioral signals.

The principle is simple:

less granularity = lower identifiability.

Every movement, keystroke, or scroll is transformed in a way that preserves functionality while reducing uniqueness.

## 4.4 Non-Manipulative Obfuscation

HOLogram does not alter the user's intention, does not modify the semantic meaning of input, and does not create automatisms aimed at influencing decisions.

The layer must:

- Protect;
- Mask;
- Normalize.

But never steer the user toward actions different from those intended.

This principle clearly separates the protocol from deceptive or manipulative techniques.

## 4.5 Natural Interaction

Protection must integrate seamlessly.

HOLogram cannot introduce artificial movements, unnatural delays, or patterns that could make the user appear suspicious to anti fraud or anti bot systems.

The output must be:

- Human;
- Realistic;
- Coherent;
- Compatible with any modern website.

Not random behavior, but non identifying behavior.

## 4.6 Adaptive Obfuscation

The level of obfuscation must vary according to context, exposure risk, and user preferences.

HOLogram does not apply a fixed protection level; it adjusts it dynamically using the Behavioral Privacy Budget, increasing obfuscation when necessary.

The protocol is dynamic, not static.

## 4.7 User Transparency and Control

All operations performed by the Layer must be understandable and controllable.

Users can inspect:

- the current protection level;
- privacy budget consumption;
- active Layer components;

- any automatic adjustments to the obfuscation level.

Control is not absolute—in some cases the Layer must prevent unsafe choices—but it is always explicit and informed.

## 4.8 Technological Neutrality

The protocol is not tied to:

- a specific browser;
- a specific programming language;
- a specific operating system.

It must be implementable as:

- a browser extension;
- an OS level module;
- an embedded component;
- a third party library.

Neutrality ensures the longevity and adaptability of the protocol.

## 4.9 Open Protocol

HOLogram is an open standard, not proprietary software.

Anyone may:

- read the specifications;
- implement them;
- propose extensions;
- create forks;
- contribute to the Layer;
- develop variants.

Behavioral protection must be a public good.

## 4.10 Merit as an Evolutionary Engine

To support a free and resilient ecosystem, HOLogram integrates a non financial contribution recognition system (HOLoToken), based on proof of contribution.

Value derives not from capital, but from the protocol's improvement.

Each contributor leaves a transparent record of their work; the protocol evolves through a decentralized, open, and meritocratic model.

## 4.11 Preserving Individual Behavioral Freedom

The ultimate goal is to protect a dimension not yet regulated but fundamental to digital dignity:

**the freedom of one's own behavior**.

No system should be able to infer identity, vulnerabilities, or intentions by observing how a person moves or types.

HOLogram is created to uphold this principle.

# 5. HOLogram Protocol Architecture

The architecture of HOLogram is designed as a local, transparent, intermediate layer positioned between the behavioral signals generated by the user and the digital systems that observe them.

Each component aims to reduce the fidelity of real behavior, prevent identification through biometric patterns, and simultaneously preserve the naturalness of interaction.

The protocol is composed of independent, modular, and extensible components, engineered according to three essential principles:

1. Obfuscate without degrading the user experience;
2. Protect without manipulating;
3. Operate without collecting or transmitting data.

## 5.1 ShadowDOM Input Vault

Typing is one of the most distinctive channels of behavioral identification: rhythm, key press timing, pauses, and corrections are unique and difficult to imitate.

The ShadowDOM Input Vault intercepts native input events (keydown, keyup, keypress), isolates them within a protected context, and produces a normalized sequence of events (input, change) stripped of the temporal granularity characteristic of user-specific biometrics.

In summary, the Vault:

- preserves semantic intent;
- removes micro timings linked to real typing rhythm;
- prevents page scripts from accessing biometric typing patterns.

The result is correct textual input, delivered as usual, but detached from the unique way the user typed it.

## 5.2 Pointer & Scroll Normalizer

Pointer and scroll movements are also major sources of behavioral identification: trajectories, acceleration curves, tremors, micro curvatures, and reaction times reflect individual motor characteristics.

The Pointer & Scroll Normalizer applies three transformations:

1. Uniform sampling of high frequency event streams;
2. Quantization of coordinates, movement deltas, and scroll intensity;
3. Controlled smoothing to remove unique irregularities without introducing “bot like” artifacts.

The output preserves the natural feel of human movement while reducing the possibility of extracting a behavioral signature.

## 5.3 Persona Mixer (session-consistent)

Pure obfuscation can generate repetitive behavior that becomes recognizable over time.

To prevent the obfuscation layer itself from becoming a fingerprint, the protocol introduces the Persona Mixer.

Each session is assigned a synthetic persona defined by a set of variable parameters:

- apparent pointer speed;
- smoothing fluidity;
- normalized typing profile;
- scrolling style with micro pauses and low variance adjustments.

The persona:

- remains consistent during the session;
- changes with each new session;
- prevents longitudinal correlation and fingerprinting of the obfuscation layer.

To avoid the obfuscation becoming a new signature, persona parameters are drawn from plausible and partially overlapping ranges, ensuring variability without extreme or overly regular patterns.
Parametric classes (e.g., apparent speed, smoothing, micro pauses) have controlled amplitudes—different enough to prevent cross session correlation, yet not divergent enough to be recognized as a distinct "HOLogram style."

## 5.4 Behavioral Differential Privacy Engine (Technical obfuscation component)

The Behavioral Differential Privacy (DP) Engine adapts differential privacy principles to real time behavioral event streams, aiming to reduce the sensitivity of the output to micro variations in user input.

This is not classical DP applied to static datasets, but an operational approach using calibrated noise and stabilization rules to prevent extraction of biometric signatures from behavioral signals.

It ensures that small changes in the user's real behavior do not produce significant differences in the output observable by websites.

It operates through:

- controlled noise injection (e.g., minimal temporal jitter, micro statistical variation of deltas);
- reduction of behavioral signal sensitivity;
- stabilization of patterns to prevent large scale correlations;
- deterministic yet variable rules to avoid fingerprinting of the obfuscation itself.

The DP Engine provides the mathematical and algorithmic foundation of the protocol, ensuring formalizable and reproducible protection.

## 5.5 Behavioral Privacy Budget (Governance and control component)

The Behavioral Privacy Budget supervises how much real behavior can be exposed and when protection should be automatically increased.

It is a dynamic system that:

- monitors the amount of behavioral information revealed;
- evaluates the complexity of exposed events;
- applies progressive protection thresholds;
- intensifies obfuscation as the budget approaches its limit;
- allows users to configure protection levels (low, medium, high).

The Budget sits above the DP Engine:

- the DP Engine regulates how data is obfuscated;
- the Budget regulates how much data can be exposed.

Dynamic behavior example:

- Session newly started → light obfuscation;
- Intense and repetitive interaction → automatic increase in smoothing;
- Cross site correlation risk → reduction of event frequency;
- Budget nearly exhausted → maximum protection mode.

The Budget regenerates gradually during low intensity periods (idle or prolonged pauses) and resets on new sessions or domain changes.
Its consumption reflects event frequency, geometric complexity (span, angular variation, trajectory), and behavioral entropy—representing the amount of potentially exposed information.

## 5.6 Exposure HUD (Human-Visible Transparency Layer)

HOLogram includes a minimal transparency interface: the Exposure HUD, which displays:

- current protection level;
- Behavioral Privacy Budget status;
- active modules;
- any automatic obfuscation escalations.

The goal is not to create a complex dashboard, but to maintain clarity and user control.

## 5.7 Optional HID Cloak (OS level protection)

The HID Cloak is an optional module that extends protection beyond the browser, acting on hardware events (keyboard, mouse, trackpad) before they reach any application.

It is designed for high sensitivity environments:

- investigative journalism;
- activism;
- advanced enterprise contexts;
- embedded systems.

Not essential, but a natural extension of the Layer.

OS level integration may require accessibility APIs or elevated privilege components, potentially impacting driver compatibility or platform security policies.

For this reason, this module is optional and intended for controlled, high security environments with dedicated validation.

## 5.8 Operational Flow Summary

Human Input

↓

ShadowDOM Input Vault

↓

Pointer / Scroll Normalizer

↓

Persona Mixer

↓

Behavioral Differential Privacy Engine

↓

Behavioral Privacy Budget

↓

Obfuscated Output to Browser/Sites

The system does not alter intent or content—only the behavioral form, projecting an “hologram” of the real interaction onto the web.

## 5.9 Architectural Philosophy

The architecture of HOLogram is not monolithic: it is an open protocol designed to be:

- Modular;
- Extensible;
- Verifiable;
- Replicable;
- Implementable through multiple variants and forks.

The only constraints are its Foundational Principles: protection, naturalness, locality, and non identifiability.

# 6. Operational Functioning of the System

The operational functioning of HOLogram can be described as a process that transforms behavioral signals.

The goal is to reduce the identifiability of behavioral patterns while preserving the naturalness and functionality of human interaction.

The system operates across five main phases: acquisition, normalization, personalization, protection, and emission toward the final application.

## 6.1 Signal Acquisition (Input Capture Layer)

Every digital interaction generates a sequence of events: keystrokes, pointer movements, scrolling, clicks, selections, micro pauses during decision making.

HOLogram intercepts these events before they reach page level scripts:

- keyboard events (keydown, keyup, keypress);
- pointer events (mousemove, pointermove, mousedown, mouseup);
- scrolling events (wheel, scroll);
- GUI interaction events (focus, blur, hover).

The goal is not to block events, but to intercept and filter them.

Raw events are then passed to the following modules without ever being stored or transmitted.

## 6.2 Signal Normalization (Signal Shaping Layer)

The second phase transforms the behavioral signals into a less detailed form.

The system applies:

- frequency reduction (downsampling);
- coordinate quantization;
- motion smoothing;
- timing homogenization;
- standardization of minor trajectories;
- removal of unique micro variations.

These processes convert real behavioral signals into a functional yet non identifying representation.

The substance of the action remains intact (typing, clicking, selecting), but the form in which it manifests is anonymized.

## 6.3 Synthetic Personalization (Persona Mixer Layer)

The normalized signals are then processed by the Persona Mixer, which introduces naturally human like variability detached from the user's real identity.

The Mixer:

- selects a synthetic persona at session start;
- assigns consistent behavioral parameters (speed, flow, micro pauses);
- applies light, continuous variations;
- avoids artificial regularities that could be attributed to the obfuscation layer.

This layer produces a fictional behavioral signature that cannot be attributed to any real individual, nor reused across sessions.

## 6.4 Differential Protection (Behavioral Differential Privacy Engine)

The synthesized signal is then processed by the Differential Privacy Engine, which applies logic inspired by differential privacy to ensure that minimal changes in real behavior do not produce meaningfully different outputs.

In practice:

- small variations in typing do not generate distinguishable changes;
- added noise is calibrated to be invisible to the user but effective against profiling systems;
- the resulting signal exhibits non sensitivity → changes in real behavior do not significantly alter the final output.

The DP Engine acts as the mathematical barrier between the real user and the web.

## 6.5 Exposure Management (Behavioral Privacy Budget Layer)

The obfuscated behavior is then evaluated by the Privacy Budget, which monitors how much "behavioral information" the site has already received.

When the budget becomes low, the system:

- increases protection levels;
- reduces event frequency;
- intensifies quantization;
- simplifies trajectories;
- limits further complex signals.

The budget serves as a dynamic self defense mechanism → the more a page attempts to observe, the less HOLogram allows it to see.

## 6.6 Output Emission to the Browser or Application (Output Projection Layer)

Once processing is complete, the final signal is projected to the browser.

What the application receives is:

- a human looking movement;
- coherent typing;
- natural scrolling;

but not the user's real behavioral dynamics.

The site receives a behavioral hologram → a functional projection that cannot be traced back to the real person.

## 6.7 Integration with External Systems

HOLogram's output is fully compatible with:

- modern browsers;
- web frameworks;
- fraud detection systems;
- standard UI logic;
- complex web applications.

Obfuscation occurs in the invisible layer between user input and the page → the platform receives data fully compliant with functional expectations.

## 6.8 User Transparency (Exposure HUD)

Throughout the entire process, the user can:

- view the current protection level;
- monitor the behavioral privacy budget;
- observe Layer activity;
- understand automatic escalation events;
- pause or increase protection.

The protocol does not operate in the shadows: it obfuscates behavior, not user awareness.

## 6.9 Session Lifecycle

A typical session follows this sequence:

1. Start → generation of the synthetic persona;
2. Input → raw events captured by the Vault;
3. Transformation → signal normalization;
4. Characterization → application of the Persona Mixer;
5. Protection → DP Engine + Privacy Budget;
6. Output → compatibility safe events sent to websites;
7. Closure → full state deletion, total reset.

At session end, no behavioral information persists.

## 6.10 Summary

HOLogram does not change what the user intends to do—only how the behavior is exposed. Interaction remains real, functionality intact, freedom preserved— but the most vulnerable human attribute, behavioral biometrics, is no longer surrendered to the web.

# 7. Security, Limitations, and Threat Models

The security of HOLogram is not measured solely by its algorithmic robustness, but by its ability to reduce behavioral correlatability in real world scenarios, even in the presence of adversaries with significant visibility and resources.

This chapter defines security objectives, trust boundaries, the threat model, defensive strategies, known limitations, and mitigation guidelines.

## 7.1 Security Objectives

1. Reduction of Identifiability: Decrease the ability to associate a session with an individual based on typing, pointer, and scroll patterns.
2. Functional Invariance: Preserve full usability of web pages and the naturalness of interaction (no UI flow disruption).
3. Obfuscation Non Traceability: Prevent Layer produced patterns from becoming a recognizable fingerprint over time.
4. No Telemetry: Prevent any form of internal or external tracking (local only, zero retention).
5. Detection Resistance: Make it difficult for page side scripts to reliably detect the presence of the Layer.

## 7.2 Trust Boundary

- Trusted:
    - The user's device as the local execution environment;
    - HOLogram's code (extension/proxy/driver), verified by the user;
    - Layer configuration (policies, levels, HUD).
- Untrusted:
    - Web pages and scripts loaded from the internet;
    - Behavioral tracking infrastructures;
    - Third parties (CDNs, advertising systems, analytics services);
    - Network observers inferring timing patterns from input/traffic shaping.

## 7.3 Threat Model (Adversaries and Capabilities)

1. Page level Fingerprinter:
    - Injects high frequency listeners (keydown, mousemove, wheel) to extract temporal/geometric features;

- Applies classifiers to micro patterns to re identify the user;
- Attempts to detect obfuscation (e.g., reduced timing precision, coordinate quantization).

2. Cross Site Correlation Platforms:
   - Share features across domains (or use common third parties) to build persistent identities;
   - Combine behavioral signals with device/network fingerprinting.
3. Aggressive Anti bot/Anti abuse Systems:
   - Classifiers designed to separate “real human” vs “masked automation”;
   - Penalize non human jitter or overly regular patterns.
4. Network Observer (out of scope but relevant):
   - Does not access input events, but may infer timing patterns from request/response sequences;
   - Not a primary objective of HOLogram, which operates at the input/DOM layer.
5. Targeted High Resource Adversary:
   - Combines classical fingerprinting, residual behavioral cues, login context, and browser telemetry;
   - Performs multi step tests to detect Layer presence.

## 7.4 Defensive Strategies of the Protocol

- Feature Separation and Reduction:
  - *ShadowDOM Input Vault: removes micro timing in keystrokes; pages see content, not rhythm;*
  - *Pointer & Scroll Normalizer: downsampling, quantization, controlled smoothing*.
- Consistent Variability (session consistent):
  - • *Persona Mixer: introduces plausible human variability consistent within a session but different across sessions, preventing longitudinal correlation and obfuscation fingerprinting*.
- Formal Protection and Dynamic Control:
  - *Behavioral Differential Privacy Engine: reduces output sensitivity to small input variations (minimal noise, stabilizing rules);*
  - *Behavioral Privacy Budget: controls the total amount of information revealed; increases protection as exposure grows*.

- User Transparency:
    - *Exposure HUD: allows users to understand and, if needed, increase protection*.

## 7.5 Detectability and Anti Detection

Risk:

A script attempts to detect HOLogram by measuring timing precision, coordinate granularity, excessive regularity, or differences between keydown/keyup and the resulting input/change events.

Countermeasures:

1. Semantic Coherence: Events follow API semantics; no inconsistencies in bubbling/ordering;
2. Stochasticity Within Human Ranges: Mixer parameters chosen within plausible human intervals (speed, micro pauses, curvature);
3. Policy Driven Variability: Quantization and smoothing patterns differ across sessions and sites (within natural limits);
4. Soft Precision Masking: Reduced temporal/coordinate resolution is hidden through sub perceptual jitter.

Even though the Layer acts at the input/DOM boundary, it cannot eliminate all side channels: network level timing (request/response sequences) may reveal indirect correlations.

These vectors fall outside HOLogram's scope and require complementary tools (e.g., anonymity networks, traffic shaping) when necessary.

## 7.6 Interaction with Anti bot Systems

Risk:

Excessive normalization or smoothing may appear "robotic."

Mitigation:

- Conservative smoothing/quantization thresholds;
- Persona Mixer introduces subtle human like irregularities;
- Site specific profiles: compatibility mode for domains known for aggressive anti bot checks;
- Gradual fallback: if the page signals issues, reduce obfuscation intensity to maintain functionality;
- In high temporal sensitivity contexts (gaming, graphics editors, real time trading), the Layer may reduce obfuscation (compatibility profile) to preserve responsiveness while still providing minimal behavioral protection.

## 7.7 Cross Site Correlation

Risk:

Even after obfuscation, consistent patterns might correlate sessions across domains.

Mitigation:

- One persona per session and per eTLD+1 → Mixer reset per domain/session;
- Per site budget → exposure limited and independent across domains;
- Wide variability classes → no “universal HOLogram pattern”.

## 7.8 Adversarial Adaptation

Risk:

Models attempt to “reverse” obfuscation (e.g., detect quantization patterns).

Mitigation:

- Evolving policies: periodic updates to smoothing/quantization functions;
- Controlled random sets: multiple interchangeable strategies selected on the fly;
- Behavioral regression testing: internal benchmarks measuring classifier accuracy reduction (without personal datasets).

## 7.9 Known Limitations

1. Does not anonymize network traffic: HOLogram acts at the input layer; network anonymity requires separate tools (Tor/VPN).
2. Does not prevent non behavioral fingerprinting: UA, canvas, fonts, WebGL require privacy focused browser defenses.
3. Variable compatibility: Timing sensitive web apps (gaming, complex graphic editors) may require compatibility mode or whitelisting.
4. Residual signals: Minor statistical traces may persist; the protocol’s goal is to make them insufficient for reliable correlation.
5. Local threat model: Does not protect against OS level malware, hardware keyloggers, or physical access attackers..

## 7.10 Security Choices and UX Impact

- Downsampling/Quantization: slight micro responsiveness impact; not perceptible in typical UI flows;
- Smoothing: balanced to avoid pointer “inertia”;
- Persona Mixer: minimal variations to preserve naturalness;
- Dynamic Budget: gradual escalation, prioritizing continuity of use over abrupt blocking.

## 7.11 Testing, Validation, and Audit

- Compatibility testing: cross browser suite on real applications (forms, editors, SPAs, canvas environments);
- Effectiveness testing: pre/post HOLogram comparison using known behavioral classifiers (no personal datasets), measuring accuracy reduction;
- Code audits: independent review and verifiable builds for extensions/drivers;
- Transparency measures: public policy changelog, DP Engine versioning, regression reports.

The test suite includes cross browser variants accounting for differences in event frequency and ordering (PointerEvents vs MouseEvents, scroll handling, native throttling), ensuring semantic coherence and human plausibility across contexts.

## 7.12 Defensive Roadmap

- v0: conservative parameters, single profile;
- v1: per site profiles, advanced HUD, per domain budget;
- v2: multiple smoothing/quantization function families with controlled random selection;
- v3: automated testing against updated behavioral classifier suites; adaptive policies for high risk contexts.

## 7.13 Summary

HOLogram does not claim absolute invisibility;

it guarantees systematic reduction of identifiability and a rebalancing of the relationship between individuals and observing platforms.

Its defenses are engineered to withstand behavioral trackers without breaking usability, acknowledging that adversaries will adapt and that the protocol must evolve continually.

HOLogram's security is, by design, a continuous and transparent process.

# 8. Compliance & Legal Considerations

HOLogram is designed as a mechanism for protecting behavioral identity, not as a system for collecting, analyzing, or storing data.

Its nature is purely transformative: it intercepts behavioral signals generated by the user, normalizes them locally, and forwards them without storing or transmitting them.

For this reason, the protocol does not constitute personal data processing and does not introduce new privacy risks—on the contrary, it is engineered to reduce them.

The following sections outline the compliance principles that guide the design of the Layer.

## 8.1 Absence of Data Processing

HOLogram does not collect, retain, archive, or transmit any information related to the user.

Specifically, the system:

- does not store biometric or behavioral data;
- does not generate persistent logs;
- does not record any identifiers;
- does not use telemetry;
- does not send data to remote servers or third parties.

Every signal is processed solely in volatile memory, transformed, and immediately passed to the browser without leaving any trace.

## 8.2 Compliance with General Protection Principles

Although HOLogram does not process personal data, its architecture voluntarily aligns with the highest standards of safety and data protection:

- Minimization: reduces the amount of behavioral information exposed to websites;
- Purpose Limitation: the Layer has a single function—protection, not collection;
- Integrity & Confidentiality: no data leaves the device; nothing is stored;
- Privacy by Design: the protocol is built to prevent processing rather than manage it.

## 8.3 Neutrality and Absence of Manipulation

HOLogram does not alter intentional user behavior, introduce bias, modify decisions, suggest alternatives, or exert any form of influence.

The Layer exclusively:

- filters behavioral signals;
- reduces their granularity;
- prevents their use for identification;
- projects a functional yet non linkable interaction.

The protocol neither manipulates nor evaluates the user.

## 8.4 Total Absence of Profiling

HOLogram does not create profiles, measure preferences, infer emotional or cognitive traits, build clusters, or perform any form of inference.

The only element produced by the protocol—HOLoToken—is:

- non personal;
- non behavioral;
- non transferable;
- non economic;
- not tied to the end user.

HOLoToken is an internal, non financial measure of technical contribution by those who participate in the development of the protocol.

It has no connection to the identity or behavior of the users protected by the Layer.

## 8.5 User Transparency

HOLogram does not operate in the shadows: it includes a transparency component (the Exposure HUD) that allows the user to:

- view the real time obfuscation level;
- understand the active Layer functions;
- adjust system behavior;
- increase or decrease protection as needed.

The user remains fully in control of the protocol that protects them.

## 8.6 Compatibility with High Protection Contexts

The protocol can be safely used in sensitive environments:

- Journalism;
- Activism;
- private browsing;
- advanced enterprise contexts;
- child protection;
- environments exposed to behavioral surveillance.

Because of its local only, zero retention nature, HOLogram introduces no additional risks in any of these scenarios.

## 8.7 Legal and Implementation Limitations

The protocol defines the technical standard, not its specific implementation.

Third party implementations may:

- introduce additional features;
- collect or synchronize data;
- connect HOLogram to external services.

In such cases, responsibility for compliance lies with the implementation, not with the original protocol.

The Whitepaper remains neutral and independent.

## 8.8 The Role of the Protocol in Protecting Digital Rights

HOLogram protects a fundamental and often overlooked dimension:

**the freedom not to be identified through one's own behavior**.

The protocol does not restrict functionality, reduce rights, or introduce obligations.

It provides a tool that allows users to control what, in today's web, is typically invisible and uncontrollable: the exposure of their behavioral biometrics.

## 8.9 Regulatory References — Non Binding Note

Although HOLogram does not constitute personal data processing, it is intentionally aligned with the overarching principles found in major European digital rights regulations (e.g., AI Act, GDPR, ePrivacy).

This alignment is achieved through:

- strictly local processing;
- absence of data retention;
- reduction of behavioral exposure;
- functional neutrality.

The protocol does not interpret or replace these regulations; it operates within a technical perimeter that reflects their fundamental principles without generating processing or legal bases.

## 8.10 Operational Summary

HOLogram is compliant because it:

- operates entirely locally;
- stores no data;
- performs no processing;
- generates no profiles;
- classifies no user;
- alters no intentions;
- introduces no telemetry;
- creates no risks.

It is a **protective mechanism**, not an analysis system.

# 9. HOLogram Ecosystem & Standardization

HOLogram is not a monolithic application—it is a protocol.

To be effective, it must operate within an ecosystem composed of multiple implementations, forks, variants, libraries, and integrations, all aligned with the Foundational Principles.

This section defines responsibilities, official artifacts, collaboration methods, and the path toward an open standard.

## 9.1 Ecosystem Goals

- Enable rapid adoption across multiple platforms (browsers, OS, embedded environments);
- Ensure minimum semantic consistency across variants;
- Support distributed innovation (forks and specialized variants);
- Preserve interoperability and verifiability (reference tests and public policies);
- Progress toward standardization without sacrificing openness.

## 9.2 Roles in the Ecosystem

- Protocol Author (BaffiSan): Holds the vision, maintains the whitepaper and protocol semantics;
- Reference Implementation Maintainers: Develop and maintain the first minimal, verifiable open implementation;
- Independent Implementers: Create extensions, OS modules, SDKs, plugins, ports, and propose variants;
- Verifiers (QA & Security): Manage compatibility tests, validate profiles, conduct code audits;
- Community Contributors: Develop features, fixes, documentation, translations, benchmarks, RFCs;
- Integrators (browsers, OS vendors): Adopt HOLogram natively or as an optional “privacy layer”.

## 9.3 Official Artifacts

1. HOLogram Whitepaper v1.x:
    - The semantic source of the protocol (this document).
2. Reference Implementation (RI):
    - Minimal browser extension + test suite;
    - Clean, documented code with verifiable builds.
3. Technical Specification (HLP – HOLogram Layer Protocol):

   - Interface definitions, events, parameters, module semantics (Input Vault, Normalizer, Persona Mixer, DP Engine, Privacy Budget, HUD).

4. Compliance Tests (HCT – HOLogram Compliance Tests):
   - Reproducible test suite validating semantic conformity.
5. Reputation Policy (HRL/HCS):
   - Public, versioned rules for assigning non financial HOLoToken reputation.
6. Security/Threat Model Notes:
   - Threats, regressions, countermeasures, recommended protection levels.
7. Ecosystem Registry:
   - Public list of implementations, versions, compliance status, and badges.

## 9.4 Process (Change Proposals)

- All protocol changes pass through public RFCs (short text + rationale + impact);
- Each RFC specifies modified semantics, affected modules, UX/security trade offs, migration plan;
- Technical quorum: approval requires:
  - review by RI maintainers;
  - validation by at least one security verifier;
  - community consensus (open discussion).
- Approved RFCs become HLP vX.Y, and implementations declare compatibility (e.g., “HLP vX.Y compatible”).

## 9.5 Interoperability & Compliance

- Each implementation must declare:
  - Supported HLP version (e.g., HLP 1.2);
  - Covered modules (Input Vault / Normalizer / Persona / DP / Privacy Budget / HUD);
  - Known limitations (e.g., compatibility mode for complex sites).
- The HCT suite includes:
  - Functional tests (event coherence, API shape);
  - Human plausibility tests (parameters within natural ranges);
  - Security tests (absence of trivial leakage, basic anti detection);

  - Regression tests across versions.
- Builds passing HCT receive the badge "**HOLogram-Compliant (HLP X.Y)**".

## 9.6 Variants & Fork

Variants are implementations with specific purposes (mobile first, high security, enterprise, embedded).

Forks are broader, experimental, or long term divergences.

Guidelines:

- Semantic compatibility: variants should remain compliant with an HLP version (or declare divergences);
- Transparency: document parameters, profiles, differences from RI;
- Upstream reintegration: if a variant demonstrates clear, generalizable benefits, propose an RFC to standardize portable aspects.

## 9.7 Integration with HOLoToken (HRL/HCS)

- Each repo/implementation publishes contribution events (PRs, patches, tests, docs, reviews);
- HRL aggregates and attests contributions, applies versioned policies, and computes HCS (reputation metric);
- HOLoToken is the reputation system (non financial), enabling project roles (Reviewer, Maintainer, Docs Steward);
- The score has no economic value and is non transferable—pure technical merit, visible and verifiable.

## 9.8 Standardization Roadmap

1. Phase 0 — Community:
   - Publication of the Whitepaper, minimal RI, initial HCT, feedback gathering.
2. Phase 1 — Technical Draft (HLP 1.x):
   - Stable specs for core modules, matured test suite, multiple implementations.
3. Phase 2 — Working Group:
   - Formation of an open working group with maintainers, researchers, integrators.
4. Phase 3 — Proposal to Standardization Bodies:
   - Submission of HLP as an open standard to appropriate foundations or privacy/interoperability groups.

5. Phase 4 — Stabilization & Governance:
    - Clear rules for semantic evolution, versioning, testing, backward compatibility, migration.

Standardization must not hinder innovation: the HOLogram trunk remains free; standards exist to help different implementers converge.

## 9.9 Reference Integrations (Practical Adoption)

- Browsers: verified extensions or native “HOLogram Mode” for sensitive profiles;
- OS: optional HID Cloak modules to protect native apps; per app policies;
- DevTools: libraries to test human plausibility of obfuscated output;
- Public Sector / Enterprise: recommended policies for regulated contexts (conservative profiles, ethical whitelists, internal audits);
- Training & Research: synthetic (non personal) datasets to evaluate classifier resilience against obfuscation.

## 9.10 Transparency, Audit & Security

- Public changelog of parameters, smoothing/quantization functions, and DP rules;
- Verifiable builds for RI and extensions; reproducible releases;
- Periodic independent audits of code and policies; documented mitigations;
- Defined security response process (timelines, channels, embargo, reputation acknowledgments).

## 9.11 Lightweight Governance (Meritocratic & Decentralized)

- RFC process as the canonical source of truth for evolution;
- Roles derived from HCS (HOLoToken) + technical elections for critical responsibilities;
- No proprietary central authority; semantics live in the Whitepaper; execution belongs to the community;
- Compatibility and plurality: different implementations, same protection language.

## 9.12 Indicative Roadmap

- T0 — Whitepaper v1.0 + minimal RI + HCT v0 + HRL/HCS v0;
- T0 + 3 months — Cross browser porting, per site profiles, advanced HUD, HCT v1;

- T0 + 6 months — Informal working group, HLP 1.1 proposal, public CI for compliance, first security audit;
- T0 + 9 months — Optional OS/HID implementations, enterprise extensions, HCT v2 with behavioral benchmarks;
- T0 + 12 months — Draft open standard (HLP 1.2), Ecosystem Registry package, second security audit, 24 month roadmap.

## 9.13 Summary

The HOLogram ecosystem places the user and the community at its core.

The protocol defines a shared language; implementations bring it to life; tests keep it coherent; reputation (HOLoToken) recognizes those who build it.

Standardization is the natural outcome of a process that is open, meritocratic, and verifiable from the very beginning.

# 10. Long Term Vision

The creation of HOLogram introduces a new layer of the digital experience:

a layer dedicated not to infrastructure, networks, or devices, but to the human being as a subject to be protected.

If the first Internet protocols were designed to connect machines, and the next generation to connect digital identities, HOLogram inaugurates a third era:

**the protection of the individual's behavioral presence**.

The long term vision goes far beyond the current technical implementation. The protocol is designed to evolve, branch, adapt to a transforming digital environment, and—above all—endure.

The following outlines the core evolutionary trajectories guiding the future of HOLogram.

## 10.1 Protecting Behavioral Freedom as a Digital Right

In today's web, the most vulnerable aspect of the user is not what they say, but what they do.

Typing rhythm, hesitation, pointer micromovements, interaction habits—all can be observed, interpreted, and correlated.

HOLogram's vision is to transform this structural vulnerability into a recognized right:

- the right not to be identified by one's behavior;
- the right to retain control over involuntary signals;
- the right to interact without being excessively analyzed.

**HOLogram is not a product: it is a declaration of behavioral autonomy in the age of AI**.

## 10.2 HOLogram as a Native Layer of the Web

In the long term view, HOLogram can become an intrinsic layer of the web, analogous to:

- TLS for security;
- CSP for content control;
- SameSite for cookie protection;
- sandboxing for script isolation.

The ultimate objective is not to extend a browser or provide a library, but to have browsers natively implement HOLogram as a system component, with:

- integrated behavioral normalization;
- native signal protection;
- user controlled granular settings;

- advanced modes for high protection contexts.

A future where “HOLogram Mode” becomes a standard feature—just like private browsing.

## 10.3 Evolution Toward Open Standards (HLP — HOLogram Layer Protocol)

The longevity of the protocol does not depend on a single implementation but on its ability to become a shared language.

The future envisions HOLogram evolving into:

- an open standard;
- a versioned protocol;
- a shared semantic layer across independent implementations;
- a reference for browsers, OS systems, and privacy by design solutions.

This journey requires no central authority—only clarity, structure, shared testing, and a meritocratic RFC process.

## 10.4 Ecosystem of Variants, Forks, and Distributed Innovation

As with all truly open protocols, the strength of HOLogram lies in its plurality.

Over time, the ecosystem will naturally expand into:

- optimized variants (mobile, enterprise, high security);
- experimental forks with new obfuscation approaches;
- OS level implementations integrated with hardware event pipelines;
- complementary tools for testing, auditing, benchmarking;
- libraries for emerging platforms (AR/VR, edge computing, IoT).

The role of the Whitepaper is to maintain conceptual coherence—not total uniformity.

## 10.5 Adapting to New Behavioral AI Models

AI systems will become increasingly sophisticated at interpreting behavioral signals.

For this reason, HOLogram must be:

- Adaptive;
- Iterative;
- Capable of evolving in response to new attacks;
- Open to scientific contributions;

- Able to update DP and smoothing rules through versioned policies.

The vision is of a protocol that grows alongside the models it must mitigate—never static, never complete.

## 10.6 Decentralization Through HOLoToken (HRL/HCS)

The protocol's future does not depend on a central authority but on the community that builds it.

HOLoToken—designed as a non financial reputation system—enables:

- attribution of merit to contributors;
- structured roles and responsibilities;
- decentralized decision making processes;
- lightweight yet resilient governance.

In the long term, the reputation system can make HOLogram:

- independent from individual maintainers;
- resistant to internal conflicts;
- supported by a verifiable ecosystem of maintainers;
- capable of continuing its mission even without its original creator.

## 10.7 Cross Domain Integration in Future Digital Environments

Behavioral protection will be relevant far beyond the traditional web, including:

- voice interfaces;
- immersive AR/VR systems;
- remote work environments;
- AI assisted applications;
- IoT devices with sensitive input
- identity less authentication systems.

The long term vision is of a protocol that exists above devices, interfaces, and platforms.

Wherever measurable human behavior exists, HOLogram can serve as a protective layer.

## 10.8 Final Mission

HOLogram's mission is simple yet radical:

**to protect human beings from becoming measurable objects**.

A web where:

- individuals can interact without being decomposed into micro patterns;
- models cannot extract invisible identities;
- behavioral biometrics cease to be an involuntary fingerprint;
- freedom of action does not imply being observed beyond one's intentions.

The ultimate vision is a digital ecosystem where behavioral protection is a natural right, and HOLogram is the foundational protocol enabling it.

## 10.9 Conclusion

HOLogram is not merely a technical Layer.

It is a paradigm shift: from **behavioral observation** to **behavioral freedom**.

Its ambition is not to replace the web, but to rebalance it—placing the individual back at the center of the relationship between humans and artificial intelligences.

A protocol designed to endure, evolve, and be adopted without barriers.

A protocol that can grow through its community— and remain alive even beyond its original creator.

# 11. HOLoToken — HRL & HCS (Non Financial Proof of Contribution)

## 11.1 Purpose and Principles

HOLoToken is the meritocratic recognition mechanism of the HOLogram ecosystem.

It is created to:

- value technical and non technical contributions;
- incentivize quality, maintenance, and security of the protocol;
- decentralize evolution through clear, verifiable, public rules;
- avoid any economic or speculative nature.

It is not a currency, not transferable, not tradable, not an investment, and not an asset.

It is a merit indicator, recorded and verifiable.

## 11.2 HRL — HOLogram Reputation Layer (the “system”)

The HRL is the reputational layer of the protocol. It is responsible for collecting, verifying, calculating, recording, and exposing reputation across the ecosystem following a *Proof of Contribution* logic.

HRL Responsibilities:

1. Collection of contribution events:

   commits, patches, tests, security fixes, documentation, translations, triage, reproducible benchmarks, technical reviews, RFCs.

2. Verification & attestation:

   CI integrations, maintainer signatures, code-owner approvals, review quorums, revert traceability.

3. Scoring policies:

   weights, bonuses/penalties, temporal decay, anti-gaming rules (public and versioned).

4. Calculation & recalculation:

   batch or near–real-time execution, with audit logs and reasons.

5. Reputation registry:

   public (non-financial) ledger containing status, history, metadata, and policy versions.

6. API exposure:

   minimal endpoints for score lookup, badges, and attestations.

7. Portability:

   exportable reputational profiles; signed badges; feeds for third parties.

Core Guarantees:

1. Transparency: public policies, versioning, changelogs;
2. Impartiality: deterministic criteria, review quorums, appeal options;
3. Privacy by design: no personal data collected beyond what is strictly necessary for contribution verification.

## 11.3 HCS — HOLogram Contribution Score (the "metric")

HCS is the value that measures each contributor's merit.

It can be:

- scalar (total HCS);
- vector based (dimensions for security, core code, docs, i18n, QA/benchmarking, research, etc.).

HCS enables:

- badges and roles (Bronze/Silver/Gold, Maintainer, Security Reviewer, Docs Steward, etc.);
- permissions (e.g., technical voting rights on security RFCs above threshold HCS_security);
- prioritization (e.g., review queue position, access to technical channels).

**HRL makes the system function; HCS expresses merit.**

## 11.4 Contribution Classes (Base Catalogue)

- Code Patch (S/M/L; core vs plugin);
- Security Fix (with tests, CVE or advisory);
- Test Suite (coverage, stability, flake rate);
- Documentation (architecture, API, guides, examples);
- Internationalization (i18n) (translations with QA);
- Benchmark & QA (reproducible, with described setup);
- Triage & Repro (issue reproduction, case reduction);
- RFC & Review (approved proposals and reviews);
- Research Notes (analysis, threat models, mitigations).

*(The catalogue can be extended via RFC.)*

## 11.5 Calculation Mechanism (Proof of Contribution)

A simple illustrative example; values are placeholders and should be calibrated by the community.

### *11.5.1 Base Score per Event*

- Code Patch (M): +15;
- Security Fix (M): +40 *(+10 if CVE/advisory);*
- Full Test Suite: +10;
- Documentation (S/M): +5 / +8;
- i18n (M, QA verified): +8;
- Approved RFC: +18;
- Approved technical review (with quorum): +12;
- Triage with reproducible case: +10;
- Reproducible benchmark: +18.

### *11.5.2 Bonuses / Penalties & Constraints*

- Review quorum: +0→+20 (based on number/role of reviewers);
- Revert: −x% of the gain tied to the reverted event;
- Decay: −y% after 12 months (maintains “freshness” of the profile);
- Anti gaming:
    - daily caps on repetitive micro events;
    - penalties for artificial patch fragmentation;
    - rejection of non substantial contributions (spam/low value).

### *11.5.3 Vector Scoring*

In addition to HCS_total, category specific vectors are computed:

HCS = <security, core, docs, i18n, QA, research, ...>

→ enabling granular permissions (e.g., voting on security RFCs with HCS_security ≥ threshold).

### *11.5.4 Transparency & Appeals*

- Each HCS change includes reason, policy version, and event hash
- A formal appeal channel records outcomes in the reputational ledger.

## 11.6 Badges, Roles, Permissions (Examples)

- Contributor Bronze: HCS_total ≥ 200;
- Docs Steward: HCS_docs ≥ 250 + documentation milestone;
- Security Reviewer: HCS_security ≥ 300 + nomination quorum;
- Contributor Silver: HCS_total ≥ 600;

- Core Maintainer: HCS_core ≥ 800 + technical election;
- Contributor Gold: HCS_total ≥ 1200.

*(Thresholds and roles are parameterized and decided via RFC.)*

## 11.7 Minimal APIs (Export & Integration)

Indicative read only endpoints:

```
GET /holo/api/v1/profile/{id}        # total HCS + vectors + current badges
GET /holo/api/v1/history/{id}        # HCS variations with reasons
GET /holo/api/v1/leaderboard?dim=security&n=100
GET /holo/api/v1/badge/{id}          # signed badge attestation
GET /holo/api/v1/policy/version      # current HRL policy version
```

Strong consistency is not required: eventual consistency with periodic recalculations is sufficient.

## 11.8 Reputation System Governance

- Public, versioned policies (semantics: policy vX.Y);
- RFC process for proposing changes (new weights, new classes);
- Technical quorum for critical approvals (e.g., security);
- Audits: signed logs of decisions and recalculations;
- Backward compatible migrations, clearly documented;
- No individual discretionary power: policy supersedes persons.

## 11.9 Security & Anti Abuse Measures

- Soft Sybil resistance: tied to verifiable contribution identities (commit signatures, repo history, cross reviews);
- Anomaly detection: artificial patterns, suspicious bursts, unjustified repetition;
- Revert awareness: recalculation of HCS on revert;
- Rate limits & caps: category- and time-based limits;
- Appeals: prevents permanent penalties for honest mistakes.

## 11.10 Privacy & Compliance

- HRL handles only contribution events and minimal metadata necessary for verification;
- No personal profiling; no sensitive data; no user behavioral telemetry;

- The reputational ledger contains no private information—only technical attestations;
- HCS cannot be used as a real world identity factor; only as a technical merit metric.

## 11.11 Scope and Limitations

- HOLoToken is not money, an investment, transferable, or convertible;
- It is a reputational indicator: it measures and certifies contributions to the protocol;
- Its purpose is to enable roles, permissions, priorities, and internal recognition—not economic value.

## 11.12 Reputational Roadmap (Indicative)

- v0 — Minimal policies, scalar HCS, base badges;
- v1 — Vector HCS, public APIs, signed attestations;
- v2 — External integrations (browsers, multi repo), advanced quorums, evolved anti gaming;
- v3 — Cross ecosystem portability (interoperable badges), federation policy.

## 11.13 End to End Example

1. Eva submits a security patch with tests → CI passes, two maintainers approve;
2. HRL records the event, applies policy v1.2:

   +40 (fix) +10 (tests) +16 (2 senior reviewer quorum) ⇒ +66 to HCS_security and HCS_total;
3. The reputational ledger updates badges; Eva's HCS_security surpasses 300 → she becomes a Security Reviewer;
4. Her new role grants her the right to cast technical votes on security related RFCs.

# 12. Hashing & Notarization of HOLogram

The integrity and authorship of the HOLogram Whitepaper do not rely on a central authority, but on a set of distributed cryptographic proofs ensuring that the published document is authentic, intact, and attributable to the author's pseudonym at the moment of publication.

This chapter outlines recommended procedures for achieving distributed, tamper resistant notarization.

## 12.1 Purpose of Notarization

The goal of notarization is to:

- prove the publication date of the Whitepaper;
- guarantee immutability of the content;
- allow the community to verify the document's authenticity;
- preserve authorship in a decentralized manner;
- enable evolution of the Whitepaper while maintaining a verifiable history.

No traditional registration is required: the proof must be **open, public, and verifiable by anyone**.

## 12.2 Hash Generation (SHA 256)

Each version of the Whitepaper must be distributed together with its cryptographic hash:

SHA-256(HOLogram.pdf) = <hash>

The hash:

- uniquely identifies the document version;
- allows verification that no changes have occurred;
- serves as the foundation for distributed notarization.

Recommended procedure:

1. Create the final PDF version;
2. Generate the hash using standard tools:

   sha256sum HOLogram.pdf;
3. Publish the hash alongside the document.

## 12.3 Publishing the Hash in Public Repositories

To ensure maximum transparency, the hash should be published in multiple places:

- official GitHub repository (immutable commit);
- a HASHES.txt file containing all past versions;
- a read only gist as redundant backup;
- README referencing the current version's hash.

Each repository commit serves as a non alterable timestamped attestation.

## 12.4 Publishing on IPFS (InterPlanetary File System)

To ensure distributed immutability, the document should also be published on IPFS.

Uploading returns a CID (Content Identifier):

ipfs://<CID>

The CID implicitly incorporates the file's content hash.

If the PDF changes, the CID changes → meaning content cannot be manipulated retroactively.

## 12.5 Distributed Timestamping (Optional but Recommended)

For stronger temporal certification, decentralized timestamping services may be used.

The protocol specifies only the *logic*, not specific platforms.

The mechanism:

- takes the SHA 256 hash;
- embeds it into a distributed ledger with verifiable timestamp;
- allows anyone to prove the hash was public from that specific date onward.

This step is optional but strengthens authorship proof.

## 12.6 Versioning & Historical Archive

Each evolution of the Whitepaper must be:

- versioned (v1.0, v1.1, v2.0, ...);
- accompanied by a new hash;
- added to HASHES.txt;
- stored in the repository as an immutable tag;

- published on IPFS to preserve the historical record.

This creates a verifiable version chain:

v1.0 → hash1 → CID1

v1.1 → hash2 → CID2

v2.0 → hash3 → CID3

The community can verify the Whitepaper's evolution—even years later.

## 12.7 Authorship Under Pseudonym

The pseudonym **BaffiSan** is attested through:

- the document's hash;
- the public association of the hash with GitHub/ArXiv/IPFS;
- digital commit signatures (if used);
- the persistence of publications over time.

Authorship does not depend on legal identity but on cryptographic continuity:

whoever controls the pseudonym can publish signed versions of the hash.

As with major open protocols, this ensures:

- anonymity of the author;
- verifiability of authorship continuity;
- protection against misappropriation;
- independence from real world identity.

## 12.8 Community Verification

Anyone can verify a HOLogram document:

1. download the PDF from any source;
2. compute the local hash;
3. compare it with the hash published in the repository/IPFS;
4. check that the hash appears in commit history;
5. check that the hash appears in public timestamp records.

If all steps match, the document is authentic.

## 12.9 Summary

The distributed notarization of HOLogram:

- requires no central authority;
- requires no official registration;
- relies solely on hashing, public publication, and distributed copies;
- protects the document's authorship;
- preserves protocol integrity over time;
- guarantees transparency and open verifiability.

It is a simple, auditable system fully aligned with the philosophy of the protocol:

**protect without controlling, attest without identifying**.

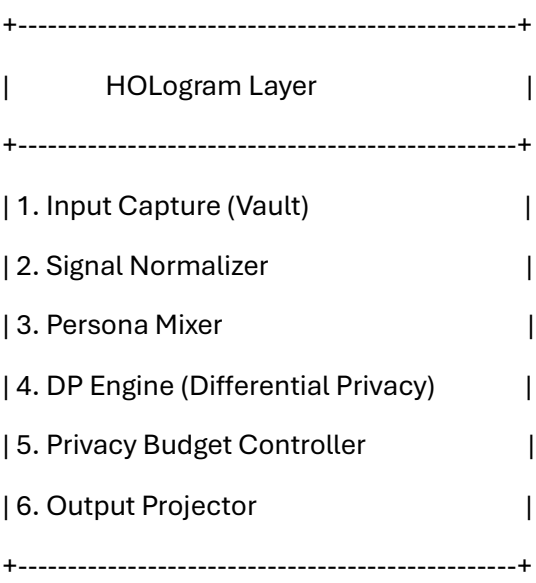
# ✔ TECHNICAL APPENDIX — HOLogram

This Appendix collects operational definitions, conceptual pseudocode, module diagrams, and a minimal glossary of the HOLogram protocol.

It is **not** an implementation, but a technical reference enabling developers, researchers, and integrators to understand the Layer's functioning and verify consistency across variants.

## A.1 Layer Module Structure

The HOLogram protocol is composed of six main modules, described here in more operational terms:




Every implementation must preserve the semantic function of each module, not necessarily its internal form or exact ordering (though the sequence **Vault → DP Engine → Output** is mandatory).

## A.2 Example Processing Cycle (Pseudocode)

The following pseudocode illustrates the general logical flow:

```
function HOLogramProcess(event):
raw = Capture(event)
shaped = Normalize(raw)
personaAdjusted = ApplyPersona(shaped, activePersona)
dpProtected = ApplyDP(personaAdjusted, dpParameters)
budgetState = UpdateBudget(dpProtected)
if budgetState.thresholdReached:
dpProtected = IncreaseProtection(dpProtected)
```

```
Emit(dpProtected)
```

This pseudocode is not binding, but clarifies the formal steps required by the protocol.

## A.3 Input Vault — Minimum Specifications

### *A.3.1 Minimum Events to Intercept*

- Keyboard: keydown, keyup, keypress;
- Pointer: mousemove, pointermove, mousedown, mouseup;
- Scroll: wheel, scroll;
- UI Interactions: focus, blur, input.

### *A.3.2 Semantic Requirements*

- No high resolution native event must reach the page;
- Event content (text, coordinates, state) must be preserved;
- Timing and micro variations must be removed or replaced.

## A.4 Normalization — Operational Model

### *A.4.1 Downsampling*

Reducing event frequency:

```
targetFrequency = 60 Hz

accumulator += 1

if accumulator % (nativeFreq / targetFrequency) == 0:

  emit event
```

### *A.4.2 Quantization*

Reducing coordinate precision:

```
quantizedX = round(raw.x / qStep) * qStep

quantizedY = round(raw.y / qStep) * qStep
```

### *A.4.3 Smoothing*

Applying a smoothing function:

```
smoothed = prev + alpha * (quantized - prev)
```

## A.5 Persona Mixer — Synthetic Parameters

A persona is defined by a set of variables:

Persona:

pointerJitterRange;

scrollPauseProbability;

typingLatencyOffset;

smoothingAggressiveness;

curvatureSoftening.

An implementation must:

- generate a different persona for each session;
- keep it consistent throughout the session;
- avoid extreme or implausible values.

## A.6 Differential Privacy Engine — Guidelines

HOLogram does not apply classical differential privacy to static datasets.
The DP Engine adapts DP principles to event streams, using calibrated minimal noise, sensitivity reduction, and stabilization to prevent identifying micro patterns without degrading UX.

### *A.6.1 Calibrated Noise (draft)*

dpEvent.value = event.value + calibrated_minimal_jitter(ε)

Where ε is chosen within human plausibility ranges, using session/site specific policies, followed by light stabilization to avoid mechanical artifacts.

### *A.6.2 Stabilization*

dpEvent = Smooth(dpEvent, stabilityFactor)

## A.7 Behavioral Privacy Budget — Operational Logic

### *A.7.1 Budget Construction*

Each event carries a *behavioral weight*:

typeWeight:

keypress: 3

mousemove: 1

scroll: 2

click: 1

Budget consumption:

budget -= typeWeight[event.type] * intensity(event)

### *A.7.2 Automatic Escalation*

When budget drops below a threshold:

if budget < threshold_low:

increase smoothing

increase quantization

reduce frequency

### *A.7.3 Budget Regeneration*

The Behavioral Privacy Budget follows a controlled depletion and modular regeneration model.

Regeneration triggers:

1. Time based threshold:

   After continuous intervals (e.g., 60–120 seconds), a small quota regenerates.

2. Neutral event regeneration:

   Low intensity events (hover, idle, long pauses) restore small amounts.

3. Full regeneration on session reset:

   New tab / domain / prolonged inactivity fully resets the budget.

This prevents the Layer from remaining in “maximum protection” for too long.

### *A.7.4 Event Intensity Model*

Event cost represents potentially exposed behavioral information:

cost = w_type × f_freq × g_geom × h_entropy

Where:

- w_type = base event weight:
  - keypress = 3
  - mousemove = 1
  - scroll = 2
  - click = 1
- f_freq = frequency factor (high repetition → higher cost);
- g_geom = geometric factor

  Based on:
  - movement amplitude;

  - angular variation;
  - trajectory complexity;
  - significant deviations.

- h_entropy = behavioral entropy factor
  - micro-variations increase informational value.

Examples:

- One click → minimal cost
- 30 complex micro curved mousemoves → high cost
- Intense typing burst → rapid budget consumption.

#### *A.7.5 Escalation & Gradual Exposure Reduction*

When budget crosses thresholds:

- Level 1 (warning): mild smoothing increase, moderate frequency reduction;
- Level 2 (active protection): significant quantization + smoothing;
- Level 3 (maximum protection): strong reduction of high frequency emissions + stronger synthetic persona adjustments.

Obfuscation decreases gradually as budget regenerates.

## A.8 Output Projector — Consistency Guarantees

The Output Projector must ensure:

- compatibility with standard DOM;
- continuity of user experience;
- absence of non-human artifacts;
- timing within natural ranges.

## A.9 Technical Glossary

Behavioral biometrics:

Involuntary characteristics that make human digital interaction identifiable.

Obfuscation:

Transformation of signals with controlled loss of granularity.

Synthetic persona:

Plausible behavioral profile generated by the Mixer to replace the real signature.

DP Engine:

Component that minimizes output sensitivity to micro-variations in input.

Behavioral budget:

Numerical value representing remaining “safe-to-expose” behavioral information.

## A.10 Summary Flow Diagram

User
↓
Input Vault
↓
Normalizer
↓
Persona Mixer
↓
DP Engine
↓
Privacy Budget
↓
Output Projector → Browser/Site