LOVE KEY LINK

Respectful Synchronised Protocol

Whitepaper & Event Token Minting Specification

Version	1.6
Date	June 2026
Author	Jack Oswald
Protocol	lovekeylink.com/rsp
Genesis NFT	0xA1755730C6F66dbe3de29e24F4Db9F448ef3FDD5 · Token ID 1 · Ethereum
npm	@rsp-protocol/core · @rsp-protocol/react
GitHub	github.com/rsp

Translate behaviour. Synchronise the signal. Burn the identifiable source.

RSP is a privacy-first coordination framework. It defines how systems observe behaviour, translate it into weighted signals, synchronise group state, and irreversibly destroy the identifiable source. This document covers the protocol specification, NFT tier structure, Event Token design, and the Solidity smart contract specification for on-chain minting of Event Tokens on Base.

Table of Contents

SECTION 1

Overview & Core Principles

RSP — Respectful Synchronised Protocol — is a privacy-first coordination framework built into the core of Love Key Link. It provides a formal model for how systems observe human or agent behaviour, translate it into low-resolution weighted signals, synchronise group state across nodes, and irreversibly destroy the identifiable source once it is no longer necessary.

RSP is not an analytics framework. It is not a profiling system. It is a protocol for coordination that treats privacy destruction as a first-class architectural primitive — not an afterthought.

Core Principles

Privacy by Destruction

Identifiable source data is removed, anonymised, or cryptographically erased as soon as it is no longer necessary. Retention is only permitted where required by law, explicit consent, safety, or legitimate accountability.

Weighted, Not Absolute

Behaviour is translated into weighted, low-resolution signals — never high-fidelity surveillance records. Signal weights are normalised and bounded. No single event produces a definitive judgement.

Synchronise, Never Coerce

Nodes synchronise toward shared states without forcing, ranking, or punishing individuals. The protocol describes group state, not individual verdicts.

Portable Across Systems

RSP applies equally to humans, AI agents, and hybrid systems. The same protocol governs wherever coordination meets privacy — education, healthcare, enterprise, civic governance, or multi-agent AI.

Consent Architecture

Signal collection requires valid consent. Consent state is tracked per node and per signal type. Signals collected without valid consent are rejected at the protocol level before translation.

SECTION 2

Signal Model

RSP reduces raw behaviour to 13 low-resolution visual states. These states describe the coordination health of a node — not the behaviour of any individual within it. Signal weights control how strongly each event type contributes to state transitions.

Visual States

`resonant`	`active`	`aware`
`dormant`	`friction`	`overload`
`drop_off`	`support_needed`	`cooling`
`converting`	`mastery`	`coordination_degraded`
`coordination_healthy`		

Signal Weights

Event Type	Weight	Relative
Completion / conversion	**25**	100%
Return visit	**20**	80%
Safety escalation	**20**	80%
Resource download	**15**	60%
Human correction	**12**	48%
Form interaction	**12**	48%
Active minute	**10**	40%
Agent handoff	**10**	40%

SECTION 3

The Burn Clause

> *When user behaviour is translated into a protocol signal, burn the identifiable source.*

When user behaviour is translated, synchronised, aggregated, or converted into a protocol signal, any identifiable source information shall be removed, destroyed, cryptographically erased, or irreversibly decoupled as soon as it is no longer necessary — unless retention is required by law, explicit consent, safety, or legitimate accountability.

Three-Step Protocol

1 Translate Behaviour

Raw events from people, agents, or systems are converted into weighted, low-resolution signals. The event payload is never stored in its raw form beyond the translation window.

2 Synchronise the Signal

Translated signals are aggregated to produce a node state — resonant, friction, cooling, and so on. The node state describes collective coordination health, not individual behaviour.

3 Burn the Identifiable Source

Once a signal has been translated and aggregated, the identifiable source information is deleted, anonymised, cryptographically erased, or irreversibly decoupled. A burn receipt is generated as proof of destruction.

SECTION 4

Integration Verticals

RSP is designed for any system where group coordination intersects with privacy. The protocol is agnostic to domain — the same five principles apply wherever behaviour is observed and group state must be maintained without surveilling individuals.

01	**AI Model Congregations & Multi-Agent Systems**	Coordinate agent behaviour and collective state without individual attribution.
02	**LMS & Online Education**	Track cohort engagement and learning health without profiling students.
03	**Product Analytics & UX**	Understand product health and friction without user-level surveillance.
04	**Customer Support & AI Service Operations**	Monitor service coordination quality without exposing individual ticket data.
05	**Workplace Collaboration**	Surface team coordination signals without creating employee surveillance infrastructure.
06	**Healthcare & Care Coordination**	Coordinate care team state while destroying patient-identifiable source data.
07	**Governance, DAOs & Civic Coordination**	Aggregate participation and deliberation signals with privacy-preserving destruction.
08	**Cybersecurity & Incident Response**	Coordinate incident state and response signals without retaining raw log identity.
09	**Creator Platforms & Communities**	Track community health and creator coordination without audience profiling.
10	**E-commerce & Marketplaces**	Measure market coordination signals without building persistent buyer profiles.

SECTION 5

NFT Tier Structure

RSP NFTs are utility tokens — provenance, access, participation, and certification. They are not investment products. Each tier serves a distinct function within the protocol ecosystem.

Tier	Name	Type	Supply
0	**Genesis NFT**	Provenance anchor	Supply: 1 · Ethereum mainnet
1	**Founder Pass**	Early supporter access	Supply: 25–100 · 100 credits
2	**Builder Pass**	SDK access & priority review	Supply: 100–500 · 250 credits
3	**Certification Badge**	Verifiable credential	Issued after review
4	**Partner Licence**	Commercial partner marker	Approval-based
5	**Audit Token**	Proof of completed review	Service-based issuance
6	**Event Token**	Signal proof of coordination	Unbounded · auto-minted · Base

Genesis NFT

Token ID 1. Contract: 0xA1755730C6F66dbe3de29e24F4Db9F448ef3FDD5. The singular origin and provenance anchor for the RSP protocol. Held by Jack Oswald. Not for sale.

SECTION 6

Event Token — Design Specification

> *The token proves the event. It cannot prove who caused it.*

The Event Token (NFT Tier 6) is a cryptographic record that a validated coordination event occurred across an RSP node. It is the protocol's answer to a specific problem: how do you prove that coordination happened — verifiably, permanently — without retaining any recoverable trace of who was involved?

The answer is the burn receipt. Source identity is cryptographically destroyed at or before mint. The burn receipt hash — embedded in the token — proves that destruction happened. The coordination is verifiable. The person is not recoverable.

Token Payload

EXAMPLE TOKEN PAYLOAD	
event_type	coordination.resonant
signal_weight	20
node_state	resonant
timestamp	2026-06-16T14:00Z [hour-level blur]
burn_receipt	0xd4e8f3a1...c9f1a2
source_id	null // destroyed before mint

What the Token Carries vs Never Carries

WHAT THE TOKEN CARRIES	
+ **Event type**	e.g. coordination.resonant, safety.escalation
+ **Signal weight**	Normalised 0–25 contribution value
+ **Node state at event**	resonant, friction, cooling, etc.
+ **Blurred timestamp**	Hour-level resolution only — not exact time
+ **Burn receipt hash**	Proof that source identity was destroyed

WHAT IT NEVER CARRIES	
- **Identity**	No user ID, account reference, or profile link
- **Raw location**	No coordinates, IP address, or device signal
- **Message content**	No text, media, or payload from the event
- **Exact timestamp**	Sub-hour precision is discarded before mint

Mint Lifecycle

Event validated	A coordination event clears the validation delay and commits to node state.
Source identity burned	Identifiable data is cryptographically destroyed. A burn receipt hash is generated.
Token auto-minted	The RSP backend relayer calls mint() on the Event Token contract. Supply unbounded.
Verifiable, not traceable	Anyone can verify the coordination happened on-chain. No one can recover who caused it.

Supply is unbounded. One Event Token is minted per validated coordination event, across every RSP node, forever. There is no cap. The token is not scarce by design — its value is evidentiary, not speculative.

SECTION 7

Event Token — Minting Contract Specification

This section defines the smart contract specification for the RSP Event Token. The contract is deployed on Base (Ethereum L2). It implements ERC-721, with mint authority restricted to a trusted RSP backend relayer address. This is a v1 specification — permissionless minting via ZK burn proof verification is a planned v2 upgrade.

Design Decisions & Rationale

Chain	Base — Ethereum L2. Low gas (~$0.001–0.01/tx), strong EVM compatibility. Unbounded supply requires high-volume minting; mainnet gas is economically unviable. Genesis NFT (Tier 0) stays on mainnet as provenance anchor.
Token Standard	ERC-721 — Each Event Token represents a distinct coordination event with a unique payload. ERC-1155 implies editions of the same token (fungibility between events), which is semantically incorrect. At Base gas prices the cost difference is negligible.
Mint Authority	Trusted relayer — The RSP backend validates burn proofs off-chain before calling mint(). Clean for v1. Relayer address is upgradeable by contract owner without affecting token holders. Permissionless ZK verification is v2.
Metadata	On-chain via tokenURI override — Event payload stored on-chain in the token struct. No IPFS dependency for core signal data. Optional extended metadata URI may point off-chain.
Supply	Unbounded — No cap. One token per validated coordination event. The token is evidentiary, not speculative.
Burn receipt	Stored on-chain per token — The burn receipt hash is immutable once minted. It is the cryptographic proof that source identity was destroyed.

Contract Interface

The contract inherits from OpenZeppelin ERC721, Ownable, and ReentrancyGuard. All state-changing functions are protected against reentrancy. The relayer address is set by the owner and can be rotated without redeployment.

```
// SPDX-License-Identifier: MIT
// RSP Event Token — ERC-721 on Base
// v1.0 Specification

pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
import "@openzeppelin/contracts/access/Ownable.sol";
import "@openzeppelin/contracts/utils/ReentrancyGuard.sol";

contract RSPEventToken is ERC721, Ownable, ReentrancyGuard {

// ■■ Token payload stored on-chain ■■■■■■■■■■■■■■■■■■■■■■■■
struct EventPayload {
string eventType; // e.g. "coordination.resonant"
uint8 signalWeight; // 0–25
string nodeState; // e.g. "resonant", "friction"
uint64 timestampHour; // Unix timestamp, truncated to hour
bytes32 burnReceiptHash; // keccak256 of burn proof
}

// ■■ State ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
uint256 private _nextTokenId;
address public relayer;
mapping(uint256 => EventPayload) private _payloads;
mapping(bytes32 => bool) private _usedReceipts;

// ■■ Events ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
event EventTokenMinted(
uint256 indexed tokenId,
bytes32 indexed burnReceiptHash,
string eventType,
uint8 signalWeight,
string nodeState,
uint64 timestampHour
);
event RelayerUpdated(address indexed oldRelayer,
address indexed newRelayer);

// ■■ Constructor ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
constructor(address _relayer)
ERC721("RSP Event Token", "RSPEV")
Ownable(msg.sender)
{
require(_relayer != address(0), "RSP: zero relayer");
relayer = _relayer;
}
```

```
// ■■ Mint ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
/// @notice Mint one Event Token.
/// @dev Only callable by the trusted RSP relayer.
/// Each burn receipt hash may only be used once.
function mint(
```

```
address recipient,
string calldata eventType,
uint8 signalWeight,
string calldata nodeState,
uint64 timestampHour,
bytes32 burnReceiptHash
) external nonReentrant returns (uint256 tokenId) {
require(msg.sender == relayer, "RSP: not relayer");
require(signalWeight <= 25, "RSP: weight > 25");
require(burnReceiptHash != bytes32(0), "RSP: empty receipt");
require(!_usedReceipts[burnReceiptHash],
"RSP: receipt already used");

_usedReceipts[burnReceiptHash] = true;
tokenId = _nextTokenId++;

_payloads[tokenId] = EventPayload({
eventType: eventType,
signalWeight: signalWeight,
nodeState: nodeState,
timestampHour: timestampHour,
burnReceiptHash: burnReceiptHash
});

_safeMint(recipient, tokenId);

emit EventTokenMinted(
tokenId, burnReceiptHash, eventType,
signalWeight, nodeState, timestampHour
);
}
```

```
// ■■ Read ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
/// @notice Return the full event payload for a token.
function getPayload(uint256 tokenId)
external view
returns (EventPayload memory)
{
require(_ownerOf(tokenId) != address(0), "RSP: nonexistent");
return _payloads[tokenId];
}

/// @notice Verify a burn receipt hash has been used.
function isReceiptUsed(bytes32 burnReceiptHash)
external view returns (bool)
{
return _usedReceipts[burnReceiptHash];
}

/// @notice Total tokens minted.
function totalMinted() external view returns (uint256) {
return _nextTokenId;
}
```

```
// ■■ Admin ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
/// @notice Rotate the relayer address.
function setRelayer(address newRelayer) external onlyOwner {
require(newRelayer != address(0), "RSP: zero relayer");
emit RelayerUpdated(relayer, newRelayer);
relayer = newRelayer;
}
}
```

Public Functions

`mint(address, string, uint8, string, uint64, bytes32)`	Mint one Event Token. Relayer only. Each burnReceiptHash usable once.
`getPayload(uint256 tokenId)`	Return full EventPayload struct for a given token ID.
`isReceiptUsed(bytes32)`	Returns true if a burn receipt hash has already been minted against.
`totalMinted()`	Returns the total number of Event Tokens minted to date.
`setRelayer(address)`	Rotate the relayer address. Owner only.
`tokenURI(uint256)`	Returns on-chain metadata URI. Override in deployment for JSON encoding.

Security Properties

- **Replay protection** — each burn receipt hash can only be used once. Attempting to mint a second token with the same hash reverts.
- **Relayer isolation** — mint() is restricted to a single relayer address. Owner can rotate without redeployment.
- **Reentrancy guard** — mint() uses OpenZeppelin ReentrancyGuard.
- **Weight bounds** — signalWeight is validated <= 25 on-chain, matching the RSP signal model maximum.
- **Zero-address guards** — relayer address cannot be set to zero.
- **Immutable payloads** — once minted, the EventPayload struct for a token cannot be modified. The burn receipt hash is permanent.

v2 Upgrade Path

v1 uses a trusted relayer for simplicity and auditability. The planned v2 upgrade replaces the relayer model with permissionless minting, where anyone can mint an Event Token by submitting a valid zero-knowledge proof of burn. The ZK circuit will verify that a valid RSP burn event occurred without revealing the source identity. This requires a ZK proof system (e.g. Groth16 or PLONK) and an on-chain verifier contract. The ERC-721 token structure and EventPayload schema remain unchanged between v1 and v2.

SECTION 8

RSP Credits & Services

RSP Coordination Credits are prepaid access to RSP services. 1 RSP Credit = £1 of redeemable RSP service value. Credits are not cash, not fiat-redeemable, not investments. They do not expire. All credit transactions are logged against the purchasing account.

Service Pricing

Service	Credits	Description
RSP Certification (Standard)	**100 credits**	Structured review against the five RSP principles. Written assessment and recommendations. No badge.
RSP Certification (Full + Badge)	**250 credits**	Standard review plus issuance of RSP Certification Badge (NFT Tier 3). Public verifiable credential.
RSP Partner Certification	**1,000 credits**	Enterprise certification covering multi-agent or commercial integrations. Includes Partner Licence (NFT Tier 4), registry co-authorship, and ongoing alignment support.
RSP Certifier Licence	**1,000 credits**	Upfront licence fee. Licensed certifiers conduct reviews independently and submit evidence. RSP countersigns and issues badges at 125 credits per issuance (50% discount). Licence recovers after 8 certifications.

Credits are fulfilled automatically after payment via Stripe. A confirmation is sent once credits are active on the account. Credits are redeemed against services at the time of service delivery.

SECTION 9

Legal & Disclaimers

RSP NFTs — Not Investment Products

RSP NFTs (all tiers, including Event Tokens) are utility tokens. They represent access, provenance, participation, certification, or evidentiary records within the RSP protocol ecosystem. They are not investment products, financial instruments, securities, or stores of value. No representations are made regarding their future value or transferability.

RSP Credits — Not Currency

RSP Coordination Credits are prepaid service credits. They are not cash, not fiat-redeemable, and not transferable between accounts. They have no value outside of the RSP services ecosystem.

Protocol Specification

This whitepaper describes RSP v1.6 as of June 2026. The protocol may be updated. The minting contract specification in Section 7 is a design document. Deployment addresses will be published separately upon mainnet launch on Base.

Copyright

lovekeylink.com/rsp · github.com/rsp · © 2026 Jack Oswald

RSP — Respectful Synchronised Protocol v1.6