



UNORYN

UNORYN Whitepaper v0.1 - Protocol Design Draft

中文修订版 | 2026-09-12 | 技术与尽调审查修订 3

Genesis / Prototype Stage

Proposed Design / 设计提案

本文面向实现原型的工程人员、评估机制的研究人员及审阅项目的合作方。目标是说明一个受控任务网络如何接收授权输入、分发有限工作、验证结果并记录贡献，以及哪些证据才能支持继续开发。本文不是已部署协议规范，不是安全审计报告，也不承诺任何代币或经济权利。

协议运行状态以 2026 年 9 月 11 日交接材料为基线。2026 年 9 月 12 日已形成可本地预览的中英文静态网站修改包及文档；这证明展示材料已制作，不证明线上部署已更新。Node、多机演示、公开报名、真实事件 Dashboard、外部重复使用和收入仍缺少验收证据。下述技术栈、字段、接口、门槛和运行政策均为拟议设计。文中“必须”指本设计的验收要求，不表示实现已经满足。

阅读目录

1 摘要与阅读约定

2 问题与相关工作

3 首发任务与需求假设

4 架构与信任模型

5 Node 身份与注册

6 任务生命周期与接受语义

7 Runner 与可重复环境

8 验证政策

9 Contribution Graph 数据与血缘

10 Verified Delta 评测设计

11 贡献计量与声誉

12 经济协调与资金约束

13 威胁、安全与数据权利

14 治理、升级、争议与恢复

15 更开放网络的前置条件

16 实验、阶段验收与失败条件

17 开放问题与参考使用

附录 A 最小实体模型

附录 B 拟议接口草案

附录 C 幂等与事务伪代码

附录 D 状态转换与指标

附录 E 最小测试矩阵

附录 F 引用索引

可核查来源链接

1 摘要与阅读约定

UNORYN 拟服务需要在参与者之间复现检查结果的小型开发团队。Genesis 建议采用离线合约文本规则任务，验证从 Node 注册、领取兼容任务、本地执行、提交结果，到验证、贡献记录和展示的闭环。在此基础上，再接入一项获授权的小模型本地评测，判断这套证据流程能否用于真实 AI 工作负载。

系统起初依赖一个中心协调器及其数据库。节点可能离线、重复提交或报错，网络可能延迟和丢包；调度按至少一次分发设计。协议通过租约代次、提交幂等和接受槽位唯一约束限制重复记账，不能阻止所有重复计算，也不承诺整个网络只执行一次。

Contribution Graph（贡献图）记录参与者、尝试、产物和验证的关系。Verified Delta（经验证的变量）记录基于冻结评测协议的候选与基线比较。结果被接受不要求带来改进；记录存在也不证明原创或奖励权。这些机制使用常规血缘、实验追踪和事务技术，额外价值须由跨参与者复现效率与实际需求实验验证。

建议先做无原生代币的受控原型，不开放任意任务或资金托管。节点报酬、收费、许可证、治理主体和 Token 均需另外决定。原型是否继续，取决于第二次真实使用、可接受的验证开销、可恢复的服务及可承担的安全责任。

1.1 术语和状态

Contributor 是参与者记录，不等于经过现实身份认证的人。Node 是获准客户端实例身份，不等于独立电脑。Task 是一次逻辑工作请求；Attempt 是一个槽位的一次执行尝试，Assignment 是该尝试对应的租约分配。Artifact 是版本化产物；Result 是与一次 Attempt 绑定的结果信封；Verification 是某个验证政策对某份结果的判定；Contribution Record 是被接受工作对应的贡献记录。

Completed 表示节点报告运行已结束，服务端仅保留为遥测事件；Submitted 表示服务端收到有效结构的结果；Verified 表示规定技术检查通过；Accepted 表示通过任务政策、资格和事务约束后被正式接纳。Failed 表示逻辑任务在规则内耗尽尝试或到达不可恢复结局。节点进程报错只是 Attempt 失败，不能立刻等同 Task 失败。

本文用“已知”指附件支持的项目事实，用“建议”指本次设计选择，用“示意/合成数据”指教学或情景数字。具体项目成果需要独立运行记录，不能以外部项目文件替代。参考资料统一采用 R 编号，关键来源复核日期为 2026-09-12；官方机制描述不等于第三方性能测量。

1.2 文档导航

第 2-3 章回答需求与差异化，第 4-8 章定义受控执行与验证，第 9-11 章定义记录、评测和计量，第 12-15 章处理经济、安全和控制权，第 16 章提出实验门槛。附录 A-E 提供实体、接口、事务、状态与验收矩阵。建议先以附录编写测试，再实现服务，避免只照着成功路径编程。

2 问题与相关工作

2.1 需要协调的是证据而不只是算力

一个维护者把脚本交给同事运行后，往往还需要确认输入是否一致、依赖是否更新、输出是否被编辑，以及当前采用的结论是否来自旧版本。简单任务并不缺执行工具，跨边界协作缺的是一致的证据交接。

UNORYN 拟把请求、尝试、验证和修订关联起来，使使用者能解释为什么接受某份结果。

这种问题不是所有用户都有。单人一次性任务通常不需要网络；已有 CI 和实验管理平台的团队可能只需增加导出字段。UNORYN 不应把已有通用工具描述成不能记录贡献，也不能把拟议功能与竞品缺少同名功能混为差异。基于当前资料，没有证据支持“全球首个”或“技术领先”的主张。

2.2 比较方式

以下每项按协调对象、用户、验证、信任、归因和经济进行比较。对方能力只描述已核查文件中的机制，未评估其实际部署性能。UNORYN 的比较项全部是待验证方案；不得把我们的愿景当现有优势。

Bitcoin 的协调对象是交易历史与双重支付问题，用户是支付参与者。白皮书通过签名、工作量证明链及诚实算力假设组织验证，费用和发行提供激励；归因围绕交易授权与出块，而非 AI 产物作者。[R1] 与 UNORYN 的重叠是需要明确问题、可验证规则、网络失败和激励。差别在于任意 AI 结果没有同样简单的全局有效性谓词。本文借鉴论证结构，不将有用计算直接替换哈希难题并声称继承安全性。

GEOD 是 GEODNET 的 Token。其官方资料描述 RTK 数据服务、基站贡献，以及在线时间、多路径等质量相关指标与奖励。[R2-R3] 目标用户包括定位服务使用者和基站提供者；质量判断依赖其设备、测量与平台规则，贡献归因落到站点服务。与 UNORYN 的重叠是供给需连接可消费服务；未核查其收入或独立服务质量。UNORYN 尚无同等明确的付费需求，不应移植其分配比例来弥补需求证据。

Gensyn 官方资料覆盖机器智能执行、验证、通信及协调，协调层涉及参与激励与支付。[R4] 用户包括 AI 开发者及计算参与者。REE 文档描述支持环境中的可复现推理，区分普通、同硬件确定性及可复现模式；页面标为 v0.6.0，部分组件有专有许可。[R5] 验证依赖具体实现与覆盖范围，收据不自动证明方法作者权利。本文未测评其经济或性能。UNORYN 与其方向明显重叠，待证明差异只能收窄到任务证据的使用体验，不能宣称 Gensyn 只是算力租赁。

Bittensor 的子网机制协调服务供给与效用评价。官方 Yuma 文件描述验证者提交评分并按质押权重形成协调和激励；其中抗操纵论证有诚实多数质押等假设。[R6] 用户包括子网服务参与者与评价者，归因主要反映评分支持的服务效用，不是对所有产物原创性的证明。与 UNORYN 的重叠是评价贡献及分配激励。UNORYN 目前没有无许可验证者制度，不能用“多种贡献”宣称已超越子网模式。

io.net 的官方文档描述计算资源供给、Worker 的周期性 Proof of Work 挑战及显存核查，并列 IO、USDC 和信用卡等支付方式。[R13] 用户是计算需求方与设备供给方，归因侧重设备和任务服务；验证仍依赖平台挑战及其实现。这些资料不证明任意模型质量，也没有在本轮接受独立性能测量。UNORYN 的重叠是资源匹配、设备可信度与执行记录；我们尚未证明跨产物复核流程能带来足够不同的需求。

W3C PROV 给出实体、活动和参与者的血缘框架；OpenLineage 记录 Job、Run 和 Dataset 及事件；MLflow 提供运行参数、代码版本、指标与产物追踪。[R7-R9] 它们服务开发、数据与模型团队，依赖生成记录的系统和权限管理；通常不替用户证明实际执行或自动建立奖励权。UNORYN 的拟议工作应复用这些概念，增加可导出的任务委托、验证政策和接受决定，而不是声称图关系是全新技术。

Celery 等任务编排实践已经处理重试和幂等要求。[R10] UNORYN 的调度部分应视为普通工程，不以自建队列作为技术壁垒。若已有工具加一个薄的证据导出层就够用，应选择该组合。后续可以适配已有格式，而不要求用户完全迁移数据。

2.3 差异化假设

建议只验证一个窄假设：外部协作者使用同一证据包，能更快回答“该结果为何被接受、如何复现、什么已失效”。以本地脚本加结构化日志/MLflow 为对照，比较追踪时间、缺失字段与错误定位成功率。没有改善时，贡献图可缩为标准导出，而非继续投入图数据库或复杂界面。

当前无法证明网络效应。节点增加可能提高环境覆盖，也可能增加验证与支持成本；贡献种类增加可能丰富资料，也可能使归因和权限更混乱。只有不同参与者反复复用证据，且新增协调成本可控，才值得讨论更广的共享智能网络。

3 首发任务与需求假设

3.1 三个候选

任务	输入与消费方	主要验收	两人团队负担
离线合约文本规则检查	授权样本；工具维护者	已知答案与参考实现	最低，适合测协议
本地小模型分类评测	授权模型与标签；评测人员	逐样本输出和冻结指标	中等，需处理设备差异
实时 Crypto Intelligence 汇总	时变来源；信息使用者	来源、时效及主观量表	高，难稳定复现

推荐首个候选仅作为任务网络闭环测试。它不执行合约，不签名，不触及资产；输出只说明文本模式或解析情况，不出具审计结论。它也不能证明 AI 技术能力，必须在宣传和 Dashboard 的 task_family 字段中保持这个名称边界。

3.2 任务包规范

团队先自行编写少量可公开发布的示例文本，并给出明确的使用许可及预期模式。第三方样本需记录原始来源、版本、许可证、分发和衍生使用限制。许可未核实的样本不发给外部 Node；许可只允许内部使用时不得在公共产物链接中暴露正文。

输入 manifest 包含 sample_id、relative_path、byte_length、content_digest、license_ref 和 expected_encoding；相对路径不得跳出任务目录，禁止符号链接逃逸、压缩炸弹和重复文件名歧义。规则包由维护者控制，不从样本中的 URL 拉取依赖。样本可包含恶意文本，但解析器应将其始终作为数据，不执行脚本、宏、编译插件或动态导入。

输出采用 result_schema_version 固定的 JSON，包含规则版本、逐样本判定、命中区间、输入与产物摘要、无法处理原因。首版每个 (sample_id, rule_id) 恰有一行，状态为 MATCH、NO_MATCH 或 UNSUPPORTED；多个命中在同一行的 matches 数组中列出。MATCH 要求非空数组，NO_MATCH 要求空数组，UNSUPPORTED 要求原因且不能转换成二元 0。对于不支持的语法，状态为 UNSUPPORTED 并列明范围；不能作为阴性结果混入准确率。报错不得泄露环境变量、个人路径或凭据。

命中位置采用原始文件的 UTF-8 字节半开区间 [start_byte, end_byte)，要求 0 <= start_byte < end_byte <= byte_length，且两端落在 UTF-8 字符边界。读取时不得静默转换 CRLF、去除 BOM 或执行 Unicode 归一化；是否接受 BOM 在任务家族中冻结。多个区间按起点、终点排序并去重，重叠是否

允许由规则说明。规则规范必须冻结大小写、注释/字符串是否参与、语法支持范围与多次命中语义；否则“独立参考实现”可能只是比较两套不同规则。

JSON 解析拒绝重复键、非有限数及无效 Unicode。按 RFC 8785 规范化适用的 JSON 对象；原始样本和其他二进制产物则哈希原始字节，不先转成文本。[R12] schema 必须说明数组顺序；记录行按 sample_id、rule_id 排序，比较按精确 ID 配对。不得把 JSON 属性排序误当数组无序。摘要以 sha256: 加 64 个小写十六进制字符表示；生成摘要的对象排除自身摘要字段。

3.3 运行边界

初版建议只支持一种已验证的 OS/架构组合，CPU 执行、只读输入、临时输出、禁止任务期间联网。示意工程初值、非实测：每次尝试上限 60 秒、内存 512 MiB、输入 10 MiB、输出 2 MiB；这些值只用于实现资源限制测试，正式任务必须在支持设备上测量后冻结。资源超限是明确失败码，不能无限重试挤占系统。

任务参数只允许白名单规则和数值选项，禁止 shell 命令字符串。即使是可信样本，也要测试超长行、非法 UTF-8、嵌套结构及解析器异常。真实公开发发要进一步落实第 13 章安全门槛，不能把内部实验限额误当完整沙箱。

3.4 谁用与谁付费

任务提交者可能是需要比较规则版本的开源维护者；使用者可能是决定是否合并该版本的同事或研究人员；潜在付费者可能是持续需要托管复核和支持的团队。三种角色可相同，但节点数量不构成需求。当前没有订单与付款证据。

低成本试验采用同一项用户真实任务的交叉对照：先保存现有脚本流程的总耗时和错误，再让使用者尝试 UNORYN 证据流程，调换试用顺序以减少学习效应。小样本不声称统计代表性，重点寻找明确的二次使用原因。未真实发生的报价反馈只能称意向，不能计作收入。

建议 G3 门槛为至少两名非团队用户无奖励承诺地再次提交有权使用的任务，并说明使用结果所作的具体决定；最初可访谈 3-5 人。数量只是最小观察门槛，不能代替任务质量或代表性。若所有需求都是团队代填、宣传互动或期待空投，否定当前需求假设。

3.5 本地 AI 的后续任务

闭环稳定后，增加一个许可证允许离线使用的紧凑模型，对固定文本样本作二分类或有限类别分类。模型与分词器摘要、推理后端、量化方式、线程及随机参数写入环境清单。不能在本稿中凭空指定未经许可核查的模型版本或性能。

验收包括断网运行、实际进程和资源日志、逐样本预测及参考环境复核。上述证据在受控设备上支持“运行了该程序”，不能在恶意远程设备上成为密码学执行证明。远程大模型 API 可作为替代成本对照，必须单独标记执行提供者，不能记作参与电脑本地推理。

4 架构与信任模型

4.1 三个执行域与六个逻辑组件

执行域分别是提交与管理端、协调服务端、贡献者机器。六个逻辑组件为 Task Registry（任务登记）、Coordinator（调度与权限）、Node Runner（执行）、Verifier（验证）、Contribution Store（记录存储）、Dashboard（只读展示）。组件可以在同一个服务进程或数据库中实现，不要求部署六套服务。

建议原型采用 FastAPI 接口、Python CLI Node 和 SQLite 数据库；产物存放受控目录，以摘要寻址，并通过权限检查访问。技术栈未部署且未做负载验证。初版可以用数据库工作表轮询验证任务，不必同时引入消息队列、图数据库、P2P 和区块链。

数据流依次为：提交者登记冻结包 → 任务登记审核 → Node 拉取租约 → 仅下载被授权的产物 → 本地执行 → 结果信封提交 → 验证工作排队 → 接受事务 → 只读查询与证据导出。内容存储负责字节，数据库负责关系和状态，验证器负责政策下的检查；三者不能互相替代。

4.2 谁能控制什么

维护者可批准参与者、发布 Runner、设定任务家族、暂停分发和撤销凭据。协调器控制排队顺序和接受政策。验证器可拒绝不合格结果，但不得自行修改任务输入或奖励规则。Dashboard 不能写接受状态；提交者不能直接给自己的结果盖章。

两人团队可能无法做到机构独立。应以不同权限和复核步骤降低误操作，但必须披露共同控制。若两人使用同一管理账号，日志不能可靠区分操作人；应使用个人凭据而非共享密码。多签或双人审批在此只能增加操作约束，不创造外部独立性。

4.3 初期假设

假设协调器诚实执行已公布政策、操作系统基本可信、受控参与者遵守测试规则；网络可能不可靠，节点可能崩溃。服务端时钟是租约与截止的权威，客户端时钟仅作诊断。存储可能损坏或丢失，因此需要恢复测试；不能将单机事务等同持久可用或抗管理员篡改。

Genesis 不提供对恶意协调器的共识保证。协调器能审查接入、延迟任务或篡改未被外部保留的历史；公开可导出记录和审计摘要可降低隐蔽修改风险，但参与者仍然承担信任。这一限制在进入无许可或有偿环境前必须重新评估。

4.4 分开三个演进维度

增加执行设备解决容量或环境覆盖；增加独立验证者解决复核信任；分散治理控制权解决单方决策。三者不是同一件事。即使使用多个数据中心，管理者相同仍是中心控制；即使链上记录摘要，实际数据和 Runner 仍可能由一个人控制。

5 Node 身份与注册

Contributor 使用服务端生成的不含个人含义的 ID，关联参与条款版本和权限状态。Node 注册时绑定 contributor_id、客户端版本、设备声明与一把可撤销凭据。device_ref 是受控设备登记引用，不是以 MAC 地址或指纹永久追踪个人的理由。公开展示只给化名及粗粒度兼容信息。

能力声明包括 OS、架构、CPU/RAM、可用加速器、Runner 支持版本、资源上限和当前可用槽位。调度器只将声明视为筛选提示，先通过小型能力测试及兼容检查；测试通过也不证明设备永久真实、独占或没有转发任务。虚报时长与硬件不得直接转换为高积分。

5.1 注册与凭据

受控测试由维护者发放单次邀请，完成注册后邀请失效。服务端保存凭据的安全验证材料，不记录可直接使用的秘密。Node 凭据只能请求自身租约、提交自身结果与读取自身状态，不能创建任意任务或读取他人产物。公网通信使用经过正确配置的 TLS；本机回环测试可以另设开发模式，但禁止把开发配置带到公开环境。

凭据轮换生成新 key_id，并保留旧凭据的撤销时间。撤销立即阻止新的领取与提交；已接受记录仍保留，但若泄漏可能影响既往证据，标记受影响时间窗并重新验证。不能简单删除历史来假装没有事件。签名若后续加入，仍需处理密钥失窃、轮换和可信绑定。

5.2 心跳与退出

心跳上报可用状态、当前 attempt_id 和本地资源，不承诺节点一定有能力完成任务。心跳与租约续期分开：续期必须引用活动尝试和代次，不能超过任务的总截止或最大执行窗口。建议初版采用固定有限租约而非无限续期，先降低状态复杂度。

正常退出停止领取，可选择完成已有合法租约或主动放弃。异常离线由服务端过期扫描处理。撤销、暂停与维护状态不能直接当作恶意；必须结合明确违规证据，避免对不稳定网络用户作错误惩罚。

5.3 独立性登记

多机验收记录物理设备数、运营者数、网络条件和软件实现来源四个维度。受控登记可由参与者说明并经过视频或现场检查支持，避免收集无关敏感材料；报告必须说明该方式依赖信任。两台电脑属于同一人时可以支持硬件差异测试，不能称为两个独立运营者。不同 IP 也可能由代理产生，相同 IP 可能是共享办公室，均不能自动判定女巫攻击。

6 任务生命周期与接受语义

6.1 逻辑任务与槽位

Task 在创建时冻结 spec_digest、输入、Runner、验证政策、权限、截止和最大尝试次数。修改这些语义必须创建新任务版本，不允许运行中悄悄替换。每个 Task 冻结 task_role=primary 或 validation，并各有一个 primary 接受槽位；槽位名称不代表任务用途。task_role=validation 必须提供 verifies_task_id 并指向一个 primary Task，不允许复核任务递归生成复核贡献。额外复跑与原任务分别验收，但不计入新增客户主任务量。每个槽位至多一个历史 Acceptance，撤销只改变当前有效性。

Task 创建还须冻结独立的 acceptance_policy_id/digest 和 accounting_policy_id/digest；verification_policy 负责技术检查，acceptance_policy 负责接纳条件，accounting_policy 负责单位与计数。三个政策各有用途，不能从未定义的全局“当前版本”临时读取。当前撤销与权限是额外资格检查，不改变被冻结的验收标准。TaskSpec 摘要只覆盖固定语义，排除 state、槽位、生成时间和 spec_digest 自身；字段列表和规范化版本须写进 spec_version。

每个槽位有 generation 整数。分配新的 Attempt 时递增代次，并生成租约到期时间和不可猜测令牌。Result 必须绑定 task_id、slot_id、attempt_id、generation 和包摘要。新提交还须在独立认证字段携带秘密租约令牌，由服务端校验其与该 Attempt 的绑定；不可将令牌塞入 Result、幂等内容摘要或公开日志。主体使用稳定 Node ID，轮换 key_id 不产生新的幂等命名空间。旧代次即使带来正确答案，也不能覆盖新代次的活动权利。

6.2 正常路径

Task 从 CREATED 经权限和包检查进入 QUEUED。Node 请求兼容任务时，在短事务内选择可分配槽位、写 Attempt 并更新 Task 为 IN_PROGRESS。没有任务返回明确空结果，不应创建虚假的工作记录。Node 在本地检查包后报告开始，运行结束后提交结果信封。

提交服务在同一短事务中检查节点权限、绑定关系、活动代次、服务端截止、Task 未取消、结果结构与摘要，再写不可变 Result 和待验证工作项。成功提交使 Attempt 进入 SUBMITTED。此后不再受执行租约到期影响：有效及时提交已转换为服务端负责的验证工作，不能一边验证一边因旧执行租约过期另派任务。

验证器读取固定内容在事务外执行耗时检查，然后提交判定。PASS 表示技术验证通过；最后接受事务再次检查 Task 为 IN_PROGRESS、审阅未过期、接受开关与权限允许、相关版本未撤销、当前验证符合冻结政策、槽位仍未接受，再原子写 Acceptance 和 ContributionRecord，更新 Task 为 ACCEPTED。即使多个验证工作重复完成，唯一约束也只允许一次接受。

6.3 迟到、重试与取消

接收资格时间 received_at 在产物已经完整保存后、数据库写事务内完成资格检查时由服务端一次取值；必须严格小于租约到期与 execution_deadline。只有该事务最终提交成功，这个时间才生效；事务回滚则不存在有效提交。提交成功响应可以在截止后到达，不能用响应到达时间重判迟到。迟到结果可留下最小拒绝审计事件，但不进入有效 Result 队列、不记贡献；避免把“可见”误当“被接受”。网络入口到包时间和客户端签名时间都不是资格时间；数据库排队可能导致失去资格，须在客户端说明。网络导致的迟到不能自动视为作弊。

执行失败或过期后，在还有尝试预算且未到任务总截止时，槽位重新 QUEUED；新 Attempt 使用新代次和退避。超过预算进入 FAILED 并说明最后错误类别。失败原因分为输入不可处理、Runner 缺陷、设备不兼容、资源超限和临时基础设施错误；不同原因不应一律无限重试。

取消在事务中把尚未接受的 Task 置为 CANCELLED，并撤销活动槽位资格。提交或接受与取消竞争时以数据库提交顺序为准：取消先提交则后续拒绝；接受先提交则取消返回冲突，需采用独立撤销流程而不能抹掉已接受事实。Node 的紧急停止应尽快中止本地执行，但服务端取消无法保证远程进程立刻消失。

6.4 验证失败与协调器重启

验证结果不合格为 FAIL，应保存检查证据，并依据任务政策决定是否重新分配执行。验证服务崩溃、超时或依赖不可用为 ERROR，不等于 Node 提交错误，不扣除贡献者分数。验证工作自身使用可恢复租约；重启后重新领取过期工作，继续检查同一 Result，不重新要求 Node 执行。

需要为验证设置最大重试及审阅期限。持续 ERROR 时将 Task 置为 BLOCKED 并公开说明“验证基础设施不可用”，停止自动重派，避免制造无限计算。经人工处理可恢复为验证待办，或明确失败关闭。execution_deadline 约束接收；接受事务的服务端检查时间还必须严格小于 review_deadline。审阅超

时或 ERROR 达限时转 BLOCKED，阻止自动接受和重派。仅 ERROR 阻塞且仍在原 review_deadline 内时，维护者可记录原因后恢复为 IN_PROGRESS 并重启有效验证工作；超出 review_deadline 后不得延长原任务争取接受，可保留诊断结果并另建 replacement Task。暂停接受期间时间仍流逝，过期同样阻塞。

调度器重启以数据库为准恢复 QUEUED、活动租约、过期待办与验证工作。进程内队列只作优化，不作唯一事实来源。重启扫描不能把 SUBMITTED 当成仍在运行而再次分配。若采用外部消息队列，须通过事务 outbox 记录待发送事件，处理重复投递；初版数据库工作表更易审阅。

6.5 幂等和原子性边界

客户端重试必须复用请求幂等键。服务端键范围至少是 actor_id、route 和 key，并保存规范化请求摘要；相同键同载荷返回原响应，不同载荷返回冲突。键不能只存成功布尔值，否则客户端无法知道之前拿到哪个 Attempt。对失效或撤销凭据仍须先做授权，幂等不能绕过权限。

唯一约束包括每个尝试至多一份有效提交、每个主槽位至多一个 Acceptance、每个 Acceptance 至多一个初始贡献记录。同一内容摘要不应成为贡献去重唯一键，因为两个独立受委托复核可能输出相同内容。也不能只按 contributor_id 去重，防止同一人的不同任务被吞掉。

SQLite 支持多个读取但同一时刻只有一个写事务。[R11] 建议短暂 BEGIN IMMEDIATE、对每条数据库连接在事务外启用并验证 PRAGMA foreign_keys=ON、设置有限 busy 重试，在事务外做下载和验证。繁忙、磁盘满或提交失败必须返回可重试错误，不能先向 Node 确认成功再赌数据库写入。外键启用须按连接验证，不能假定默认生效。[R14] 事务只保证当前数据库的接受一致性；磁盘、备份、跨系统付款及恶意管理员仍有另外风险。

7 Runner 与可重复环境

Runner 是任务家族的受审程序，不是任意用户命令代理。发布 manifest 记录 runner_id、版本、源代码引用、依赖锁定、输入输出 schema、支持环境、资源上限及包摘要。版本号方便人读，内容摘要绑定具体字节；两者同时存在，避免同版本被重新打包而无法识别。

初版依赖尽量少，安装与任务执行分离。运行期间不安装包、不读取任意配置、不调用外部模型 API。环境清单记录 Python、关键库、OS、架构和区域编码；本地 AI 再增加模型、分词器、量化、推理后端和驱动。许可证和包真实性审阅属于发布门槛，不因文件能下载就默认许可。

可重复性分三档：相同输入的规范化离散输出完全一致；在冻结误差范围内数值一致；统计分布或任务指标在约定范围内相容。Task 必须选择适用档位。不能把不同浮点后端的合法差异一律判作弊，也不能用宽阈值遮掩实质退化。环境不在验证矩阵内时返回不支持，先扩测试再扩兼容声明。

资源限制包括执行时间、内存、磁盘、输出大小、CPU/加速器配额和网络访问。操作系统层面真正生效的限制与应用自报限制需分开验证；如果某平台尚无法落实，则不把它列入公开支持范围。用户主动停止任务始终优先于积分或租约。

Runner 应在隔离工作目录生成产物，结束后按保留政策清理自己创建的临时文件，不递归删除用户任意路径。错误日志脱敏，调试模式默认不外传环境。自动更新初期延后，用明确版本的手动升级；未来自动更新也需独立验证包签名、撤销旧版本和失败回滚。

8 验证政策

8.1 分层检查

第一层检查权限、结构、长度、摘要和对象绑定，排除无效输入；第二层检查任务特定语义，如逐样本规则结果；第三层按预定策略复现或抽查；第四层决定任务是否满足接受政策。前面通过不代表后面必然通过，Verification 必须记录各层结果与 reason_code。

验证政策在任务创建时冻结 policy_id、version、参考实现、测试数据、复核要求、容差及适用环境。修改参考实现后不能悄悄覆盖历史 PASS，应创建新政策和新验证，必要时撤销相关贡献。验证者不能拿执行结果当成自己的标准答案重新输出。

8.2 确定性任务

首版用已知答案样本验证 Runner，再由独立编写的参考规则实现或人工审过的黄金结果检查任务输出。完全相同 Runner 在另一台机器复跑，只能检验环境与传输的一致性；独立参考实现更有机会发现共同软件错误，但也可能共享错误规范。因此还要使用刻意构造的反例、边界值和人工核查。

结果可预测的任务天然容易复制。任务绑定、隐藏抽查标签和延迟公开结果是拟议控制；其检测率、误报率与成本尚未测量，不能断言已提高作弊成本。发给 Node 的输入对它不再隐藏，即使标签不公开，Node 也能将任务转发给代跑方。Genesis 的目标是闭环正确与成本观察。无现金积分仍可能引发刷量、声誉竞争或空投预期，不能把低奖励视为诚实保证。测试需主动尝试正确答案复制、任务转发与共同错误，分别报告哪些被阻止、哪些仅被观察、哪些仍无法识别；在获得结果前不得宣称解决“有用工作证明”。

8.3 模型与主观任务

模型评测保存逐样本预测、标签版本及逐次运行结果，区分可解析率、质量和资源指标。无输出或超时计入覆盖率及预算，不能只保留成功样本提高分数。重复次数与抽样单位先确定，硬件差异分别报告；无法保证位级一致时按冻结容差和统计计划复核。

主观任务使用明确量表、至少能披露的审阅角色、盲审安排及分歧处理。多个审阅者可能共享偏差，投票一致不等于事实正确。Genesis 不将主观意见自动兑换经济奖励；如果一致性和争议成本不适合自动化，保留人工服务或不支持该任务家族。

8.4 抽查的边界

只有完整验证成本过高且风险可接受时才研究抽查。抽样概率必须由政策控制，不能由提交者挑选容易通过的样本；抽查不到错误不证明全部正确。若假设样本错误相互独立，可推导发现概率，但串谋、成簇错误和自适应攻击会破坏该假设。本文不提供未经实验的通用“安全抽查率”。

验证成本要单独计量，包括隐藏集维护、人工争议、重复运行与验证服务闲置。不能只计算 Node 执行费用后宣称平台更便宜。第 12 章给出无实测条件下的成本框架，第 16 章要求用实际任务重新估计。

9 Contribution Graph 数据与血缘

9.1 实体不等于价值证明

图是关系与证据索引，可以先用关系数据库实现。Contributor 代表谁被登记，Node 代表哪个客户端凭据，Attempt 代表哪次获授权执行，Artifact 代表哪份字节内容，Verification 代表谁按哪版规则判定，ContributionRecord 代表哪次接受所记录的工作。每个对象回答不同的问题，不能把一个带钱包地址的哈希对象当作全部模型。

Task 使用不透明 ID，并以 spec_version 与 spec_digest 冻结工作内容。Result 包含 attempt_id、input_digest、output_artifact_id 和 environment_digest；Verification 引用 result_id、policy_digest、verifier_version 与 evidence_artifact_id。所有外部引用都要验证存在与权限，不能接受悬空对象后在 Dashboard 显示“已验证”。

服务端生成 created_at、received_at、accepted_at 等 UTC 时间；客户端 start/end 另以 claimed_ 前缀记录，明确不是可信计费时钟。schema_version 描述字段结构，record_version 描述对象修订，两者不能混用。内容摘要明确算法和规范化版本，不使用无说明的 digest 字符串。

9.2 边类型与 DAG

derived_from 只连接具体版本的产物，表示某产物基于另一产物产生；插入时检测环，拒绝自指及间接回环。uses 表示任务使用某产物，produced_by 表示系统将产物登记到某 Attempt，边本身不证明生产过程真实发生；verified_by 表示结果由某 Verification 检查，credited_to 表示贡献记录对应参与者。disputes、revokes、supersedes 是审阅关系，不强行施加与派生相同的 DAG 语义。

如果一个方法使用多份数据和代码，保留所有输入版本；不因为图更漂亮就把真正依赖合并成单个来源。图的可视化是查询投影，底层稳定 ID 与关系记录才是证据。节点和边必须可导出，使用者不应依赖某张截图才能解释贡献。

9.3 重复、归属与复用

内容相同可用一个 blob 存储，Artifact 实体仍可因不同许可、声明或版本语义而不同。同一任务同槽位重复提交不再计贡献；独立受委托复核即使产物相同，可以形成另一项 Validation 工作，但必须明确复核目的。未受委托重复运行不得自动变成有偿需求。

结果提交者不自动成为方法作者。每项作者、派生和运行声明必须附声明人、证据引用、审阅人及当前状态；未审阅时显示“声明未核实”。代码或数据许可支持使用范围，不单独证明作者声明真实。方法贡献由 source_ref、author_claims 和许可支持，作者声明发生争议时进入审阅；链上地址或最早上传时间都不能单独确立原创。复用记录说明某产物被采用，不自动产生分成，也不保证此前贡献者授予了商业使用权。

经济奖励权必须来自另行同意的任务条款、权限和结算政策。Genesis 不定义永久追溯分红。多个贡献者共同形成方法时可记录共同作者及争议状态；不能准确分摊就暂不分配，而不以看似精确的“智能贡献百分比”代替证据。

9.4 修订、撤销与审计

已接受对象不原地修改关键字段。修正形成新版本，记录理由、操作者、旧引用和新验证。撤销不删除原接受：新增 Revocation 事件和必要反向计量记录，当前有效查询排除它，但历史查询保留发生过的接受。一个历史事实与当前是否有效是两个维度。

政策重评可能使一组结果失效。必须按 task_family、Runner 版本、数据版本或 verifier_version 找出影响范围，暂停相关工作并重新验证。没有重新验证之前不得把原 PASS 自动迁移到新政策。积分统计应随撤销重算，保留更正前后差异。

追加审计日志记录 actor、action、object_ref、前后版本摘要、UTC 时间与 reason。哈希链只能在外部保留旧链头等条件下帮助发现改写；管理员能重写整条本地链时不能称其不可篡改。早期可定期导出摘要到另一受控位置，后续再评估签名或公开锚定成本。

9.5 能证明与不能证明

数据库记录首先证明系统曾按某个政策登记一项事件，可信程度依赖权限、日志与备份。内容哈希只支持字节一致性；签名只支持密钥与消息认证。它们不自动证明执行、设备独立、作者身份、数据权利或语义正确。链上锚定同样不能修复没有许可的数据和错误的测试标准。

最小可导出证据包应包括冻结 Task、Result、Artifact manifest、Verification、Acceptance、当前撤销状态和环境清单。导出包还须声明 exported_at、截至该时刻的撤销快照和访问范围。离线读取只能确认快照当时的状态；不能声称在导出后持续有效。重新获取有权访问的当前状态才能检查后续撤销。包内不含节点秘密和私人路径。审阅者应能离线验证摘要和引用完整性；若无权限读取输入，则报告只能说明有受限证据，不能宣称任何人都能公开复现。

10 Verified Delta 评测设计

10.1 问题定义

Verified Delta 是在约定范围内判断候选相对基线变化的一套记录与复核流程，底层仍是对照评测。它不定义通用智能单位，不保证所有贡献可相加，也不取代需求验证。普通任务只需满足自身验收，不必优于任何模型；对比任务必须显式声明 baseline 与 candidate。

设评测协议 P 包含：基线 B、候选 C、数据 D、环境 E、预算 U、随机计划 S、指标集合 M、最小实际改善阈值和资源约束。P 还包含选择候选的历史、留出集访问规则与复核计划。B/C/D/E/U/S/M 都由版本和摘要绑定，任何改变都需要新 protocol_id 或新 revision，不能混成同一实验。

10.2 数据与随机性

开发集用于选择方案；留出集仅用于按冻结计划评价。公开合约示例可以是闭环测试数据，但若候选作者看过答案，它不再适合证明泛化改善。原型可先用公开集测试评测流程，再用权利清楚的新留出集作正式比较，并清楚标记两者。

独立性通常发生在样本来源、设备、种子或执行批次中的某一层，不能把同一份样本重复跑一百次当作一百倍独立数据。若样本来自相同项目，应按项目分组估计不确定性，防止训练和测试包含近重复文本。版本、采样条件和排除规则必须记录。

随机任务预先设定重复次数或顺序停止规则，保存所有运行，包括失败和超时。为对比使用共同随机种子可以减少部分差异，但种子相同不确保跨后端相同输出。置信区间方法必须适配指标和依赖结构；小样本或没有逐样本结果时标记估计不可靠，不提供装饰性的显著性星号。

10.3 分别定义质量、延迟和成本

设 q_B 与 q_C 为在相同协议下的质量指标，单位为指标自身分值。绝对改善 $d_q = q_C - q_B$ ，越大越好。若 q_B 非零且量纲适合，相对变化 $r_q = d_q / q_B$ ；百分比为 $100 \times r_q$ 。对于 F1 从 0 到 1 的分值，百分点变化为 $100 \times d_q$ ，不可与相对百分比混用。

设 l_B 与 l_C 为同一硬件、相同冷/热启动口径下的延迟，单位毫秒/样本；延迟节省 $d_l = l_B - l_C$ 。设 c_B 与 c_C 为同一成本边界下的单位成本，单位货币/样本；成本节省 $d_c = c_B - c_C$ 。不同设备不同价格时，延迟和成本可有不同方向，必须分别展示。

示意政策、非已批准参数：一个质量优先任务可以要求 d_q 至少 0.02，且候选延迟不超过基线 1.10 倍，单位成本不超过 1.05 倍。实际任务应由使用者根据错误代价和预算确定这些阈值。不得为候选恰好通过而在看过结果后调整。

若指标没有自然比例尺度、基线为零或负值，相对百分比可能没有合理含义，应报告绝对值和单位。多个质量指标冲突时，在运行前指定主指标和不可退化护栏；不临时挑选最漂亮的数字。资源约束超限不因质量分数更高而自动被抵消。

10.4 四种结论

Verified Delta 是待实现的评测记录流程名称，不表示本稿或当前项目已经通过验证。首版自动判定仅支持第 10.6 节的固定有限集合、确定性二元输出比较。“验证出改善”只表示该集合上的质量变化达到冻结门槛，且所记录运行满足资源约束；不推断未来样本、实际执行者或商业价值。

判定先检查证据资格；不合格一律为 INSUFFICIENT（证据不足），并保留已有失败观察。资格完整后，质量下降超过预设容忍或任一资源护栏失败，为 DEGRADED（按该协议退化），逐项标明原因。其余情况下，质量改善大于等于 $\min_gain$ 为 IMPROVED（验证出改善），否则为 NOT_CONFIRMED（没有确认改善）。所有护栏都必须通过，不能以提高 F1 抵消超时或超支。

缺少输入、许可、完整输出、可信测量或复核时，不得输出前三类结论。若已有某轮超限，但质量证据不全，主结论仍是证据不足，同时明确记录“观察到资源约束失败”；不得隐去坏结果。未实现随机任务的抽样和不确定性方法前，随机任务只能输出观察报告与“证据不足”，不能由默认统计函数自动升级为改善。

10.5 数字演示

以下全部为示意/合成数据，非 UNORYN 实测。在 200 个二分类样本中，正类与负类各 100 个。基线 $TP=80$ 、 $FP=20$ 、 $FN=20$ 、 $TN=80$ ；候选 $TP=86$ 、 $FP=18$ 、 $FN=14$ 、 $TN=82$ 。 $F1 = 2TP / (2TP + FP + FN)$ ，因此基线为 $160/200=0.8000$ ，候选为 $172/204 \approx 0.8431$ 。

仍为示意/合成数据：绝对变化约 0.043137，相当于 4.3137 个百分点；相对变化约 5.3922%。不能写为“准确率提升 5.39%”，因为这里计算的是 F1，且百分点与相对值不同。混淆矩阵总数一致只能支持算术检查，不能支持真实运行、样本代表性或统计独立性。

仍为示意/合成数据：若平均延迟由 100 ms/样本变成 112 ms/样本， $d_l = -12$ ms；质量点估计较高，但超过示意的 1.10 倍延迟护栏。如果只有本表，没有逐样本配对输出、数据冻结及用途范围记录和复核，最终状态是“证据不足”，附注“观察到延迟护栏未通过”。只有相应测量证据满足协议后，才可确认资源维度退化。

这组示例刻意不提供置信区间。仅有边际混淆矩阵不能推断两个模型在每个样本上的成对差异，不能伪造配对检验。若要评价不同实现的质量变化，需要保存每个 `sample_id` 的两份预测，按预注册方法估计差异，并报告失败样本和不确定性。

10.6 首版可实现的有限集合判定

首版沿用离线规则任务。评测清单列出固定的 (`sample_id`, `rule_id`) 行，每行只有一个经审阅的二元期望标签；正类定义为命中指定文本模式，不代表存在安全漏洞。清单不得为空，至少含一个正类和一个负类。基线和候选都须对全部行返回唯一的 0/1 结果；缺项、重复 ID、额外行、UNSUPPORTED、解析失败或超时均使本次比较证据不足，保留失败数与已花预算，不删行后重新算成完整评测。

运行前冻结 `protocol_id`、`revision`、`protocol_digest`，以及 B/C/D/E/U/M 各版本摘要。M 固定主指标 $F1 = 2TP / (2TP + FP + FN)$ 。阈值必填 `min_gain`（大于 0 的 F1 分值）、`max_quality_loss`（大于等于 0 的 F1 分值）、`latency_ratio_cap`、`cost_ratio_cap`（均大于 0）及总运行预算。阈值必须来自该任务的验收要求，未填不得执行比较；10.3 节的数字仅为示意，不能作隐含默认值。阈值以十进制字符串存储并按精确有理数比较；F1 由整数计数构成分子，不先转成舍入浮点。延迟和费用同样拒绝 NaN、Infinity、负数与缺少单位的值；比值的基线须严格为正。用于比较的计价和分摊版本在运行前冻结。

S 在本口径中固定为非随机；同一受控参考环境中按 B、C、C、B 顺序各完整运行两次，保存全部四次记录。两次重复用于检查确定性，不构成独立样本或置信区间。每个版本两次逐行输出必须一致；不一致则超出支持范围并记证据不足。不得在四次中挑最快、最好的一次，运行失败后补做的尝试也保留并计成本；补做须另立完整评测 `revision`。

每轮延迟定义为可信观察端从装入冻结输入到输出完整落盘的耗时除以行数，单位 ms/行；冻结是否包含模型加载、预热及缓存状态。按第一次 B/C 和第二次 B/C 配成两对，每一对都要求 $|C/I|_B$ 小于等于 `latency_ratio_cap`。只报告这些观测运行的护栏结果，不把两次测量称为稳定性能保证。任一基线延迟为零、负数或测量无效则证据不足。

同样逐对比较 c_C/c_B 小于等于 `cost_ratio_cap`。c 是同一货币单位/行的全流程评测成本，边界包含本轮执行、传输、产物保存、验证和按冻结分摊规则计入的支持成本；现金或经济成本口径须在协议中二选一并标明，不能混算。观察端记录用量，按冻结计价表或可核查账单计算；自报 `cost` 不满足条件。零基线成本、缺少验证成本或只测执行耗时均不能用于此比例判定。协议级复核、失败重试和共享支持成本须列成唯一费用行，按运行前冻结的分摊规则归入对应运行；同一费用不能同时算入运行与额外总费。无法合理分摊时，保留总额并判该比例证据不足，不默认零。另报告整次实验包括失败的总支出是否超过冻结预算。

受控复核者必须核对版本摘要、逐行标签及输出、四次运行记录、计量来源和成本分摊；独立重算 F1 和护栏。复核者的权限、控制关系与证据引用写入 `Verification`。此复核不要求再免费运行整套程序；需要额外复跑时，先登记次数、费用及预算。参考标签需事先人工检查代表性反例；复核仍依赖标签和观察端可信，不能证明未知错误不存在。

冻结协议 P 只保存运行前已确定的版本、规则、预算、候选历史、访问政策与复核计划；protocol_digest 不包含其自身，也不包含未来运行或复核结果。运行后另存证据包，引用被冻结的 protocol_id、revision 和 protocol_digest，并保存完整运行、测量及复核引用；证据包使用自己的内容摘要，不回填改写 P。资格审阅同时核对 P 的冻结时间、许可与候选历史，以及该证据包。固定公开样本允许作有限集合流程比较，但报告 scope 必须注明 public_finite_set，不能冒充未见数据或泛化验证。若声称留出集，则额外检查访问与泄漏记录；访问规则违反即证据不足。不能仅靠提交者勾选“无泄漏”“已复核”通过资格检查。

数值判定在这些资格检查之后执行：令 $d_q = q_C - q_B$ ；若 $d_q < -\max_quality_loss$ ，或任一配对资源比例大于上限，或总预算超限，输出 DEGRADED；否则若 $d_q \geq \min_gain$ ，输出 IMPROVED；否则输出 NOT_CONFIRMED。等于质量损失容忍边界或资源上限时视为通过护栏；等于改善门槛时视为达到改善。资格不全优先返回 INSUFFICIENT。所有输出保留 reason_codes、指标值、单位、scope 和协议摘要。

这使首版判定条件可以被编码和逐项验收，但尚不是端到端运行实现。随机模型、连续值误差、未来样本推断与主观评分仍需各自冻结统计计划；首版不自动支持。没有资格证据时，纯数值判定器即使返回 IMPROVED，也只是一项计算示例，不得成为对外的 Verified Delta 报告。

10.7 候选筛选与联合归因

完整候选登记应包括失败和被放弃的候选。反复查看留出集、挑选最佳种子、改指标或只上报赢过的对照都会制造偏差。若要从多个候选比较推断未来样本表现，须事先明确比较家族，并确定适用的误差控制方法或独立最终确认集。固定公开集合上的比较只报告该集合结果，并保留所有候选记录。发现泄漏时使该评测协议失效，保留记录，换新数据重新评估。

若方法 A 与 B 共同应用，A 单独相对基线的变化与 B 单独变化之和一般不等于联合变化。消融可以提供局部证据，但结果依赖基线、组合顺序和数据。Genesis 不估算通用单人边际智能价值；共同成果记录为联合贡献，经济分配另行约定，归因争议未解决时暂停分配。

11 贡献计量与声誉

计量必须服务诊断和接受事实，不能先以积分造热度。基本单位是一个被接受的、定义清楚的工作项。Compute 在首版仅计量被接受的主任务结果提交，不推断 CPU/GPU 工作量或真实执行者；Validation 记录受委托复核接受，依赖相应复核证据；Knowledge 和 Intelligence 记录获许可产物及其被采用关系。不同单位分别列示，不直接合成跨任务价值排行榜。

Genesis 可采用简单计数或任务家族内的预先公布权重，所有规则标版本。权重不表达市场价值；不能以自报 GPU 型号、连续在线小时或任意上传文件数量获得无限积分。未受委托重复提交、迟到结果、结构无效提交和同槽位重放不增加有效计量。

声誉应分维度报告：在某任务家族中被接受比例、按期提交比例、被复核覆盖率和撤销率，并显示样本量及时间窗。新参与者不能因样本少被判定不可靠，也不能因少数成功被认证为通用专家。任务难度和调度选择偏差必须说明，不能直接把不同参与者原始通过率排序后决定所有权利。

积分无现金价值，不承诺兑换 Token、空投或收益权。更正需要公开理由和证据引用；基础设施故障不扣分。若认定违规，先限制新任务或标记争议，允许参与者补充证据；无法确认责任时不自动施加经济处罚。本稿没有批准任何罚没、收费或资产托管机制。

Dashboard 的建议核心口径为：submitted_tasks 按已登记的 distinct task_id；execution_attempts 按 attempt_id；submitted_results 按有效 result_id；verified_results 按满足冻结政策且未失效的 distinct result_id。上述指标按 Task.task_role 分列。accepted_tasks 只计 task_role=primary 且存在当前有效 Acceptance 的 distinct task_id；accepted_validation_tasks 只计 task_role=validation 的有效接受。revoked_acceptances 单列。主任务统计仍须区分团队演示与外部真实请求，不能把 accepted_tasks 自动当作付费订单。相同 Result 多次验证不增 verified_results，复核的 primary 槽位不使其进入主任务分母。

Completed 仅是节点上报事件，不进入有效贡献分母。失败率必须写分母与时间窗，例如“在该批已达到终态的逻辑主任务中，FAILED 占比”；仍在处理的任务单列，避免将它们先计失败或排除到看不见。接受后撤销在当前有效指标中扣除，但历史接受流量保持可查。

12 经济协调与资金约束

12.1 无原生 Token 的当前结论

无原生 Token 的产品方案最适合 Genesis：用已有电脑做受控实验，未来可提供按任务或支持范围计费的服务。它解决客户采购和明确责任的问题，但需要处理合同、费用、支持与退款。志愿贡献可以降低现金支出，不代表没有经济成本或可以无限持续。

既有支付或结算资产方案可以在真正有跨参与者支付需求时考虑。普通支付渠道、账单或适用的既有链上资产均可能满足要求；选择取决于地区、对手方、争议、手续费和托管负担。其优点是不必创建新资产，缺点包括价格或信用风险、链上费用和操作复杂性。本稿未批准具体渠道或收费。

当前结论：本文全部已定义功能都可以不使用原生 Token 实现，尚无一必要功能因无币而无法工作。身份用凭据，领取用队列与配额，接受和归因用数据库，复核用任务政策；若未来有付费需求，可另定账单、服务报酬或既有结算渠道。上述替代仍各有成本和适用条件，但不要求新资产。原生 Token 不是 Genesis 交付项、发布门槛或默认后续阶段。未来 Token 方案仅在已观察到的需求无法由这些替代满足时重新论证。需要说明为什么既有支付、配额、保证金或成员治理不足，以及新增资产能否在没有价格上涨预期的条件下支持真实服务。若主要吸引力只是奖励发行，则属于补贴获取供给，不证明客户需求。

12.2 同功能替代与尚未成立的必要性

任务结算的需求是支付可验收服务。前置条件是明确交付、价格和争议处理；替代方案是普通账单或既有资产。原生币会增加计价波动和使用摩擦，失败模式是客户不愿为买服务先承担资产风险，因此不得强迫所有角色持币。

Node/Validator Bond 的需求是约束可客观证明的违约。前置条件是可验证证据、合理申诉和错误更正；替代是邀请制、信誉额度、普通保证金或保险。对随机模型分歧或网络故障直接罚没可能伤害诚实者；质押数额不能证明答案正确或人员独立。

资源访问的需求是分配真正稀缺的容量。前置条件是有稳定需求和可观察拥堵；替代是队列、配额、预约和货币定价。没有拥堵时制造锁仓门槛只是增加阻力，不能把锁仓价值作为使用价值。

贡献激励的需求是补偿被使用的有效工作。前置条件是有需求预算、验收和合法使用权；按结果采购可按约定交付付款，按实际执行量补偿还需要相应执行与归属证据。替代是服务收入、资助、明确报酬和非经济署名。若发行奖励超过消费需求而长期无补足来源，系统依赖外部补贴，不能称自循环。

有限治理的需求是让承担后果的参与者影响明确参数。前置条件是权限边界、升级与紧急响应；替代是维护者制度、用户代表或合同治理。持币投票可能集中，不能授权使用他人数据或保证安全决策。这些功能需求不构成发行原生 Token 的理由。若未来提出方案，必须用同一工作负载比较采用率、总成本、争议负担及安全假设；没有发行补贴和价格上涨预期时仍有净改善，才有继续研究依据。本稿没有这项证据。

12.3 单位经济变量

以同一 task_family、输入规模、政策及观察周期的主任务批次核算，团队演示、外部免费试用、付费客户分别列示。A 为该批当前有效接受的 primary 任务数，撤销和复核任务不计入 A。A 不是计费数；另报可计费任务数 A_billable。X 是这批主任务的所有执行尝试（包括失败、重试），V 是其全部验证运行（包括独立 validation Task、内部验证、验证重试和人工复核对应工作），不得同时把同一次运行放进 X 和 V。

R 为对应同一批已交付服务、按约定条款计算并扣除退款和折让的收入，不用该月所有现金流代替。尚未交付的预付款、融资和补贴不计入 R；另列该批净收款 R_cash、未收款及资助金额，避免将应收当现金。无确认价格或付款条款时 R 标未知，不能把免费试用乘假设价格写成收入。尚未结案的服务义务、争议、退款敞口与待完成任务须单列，最终核算随结案更新。

c_x 与 c_v 分别为每次普通执行和验证的平均变动成本；实际应优先逐条汇总金额，避免不同规模用简单均值。W 为带宽，S 为存储，O 为协调、客服、争议和人工返工中未计入 X/V 的成本；F 为预留节点容量固定成本，J 为支付处理费及未在前项覆盖的履约补救成本。各项使用相同货币、时间窗及分摊规则，列明税费计入口径；无法估计的成本标未知，不默认零。退款已在 R 扣除，不能再在 J 重复扣一次。

单位全成本 $C = (X \times c_x + V \times c_v + W + S + O + F + J) / A$ ；平均实现收入 $P = R/A$ ；单位全成本差额 $m = P - C$ 。它不是会计净利润。若 $A=0$ ，单位值不定义，只报告总费用与完成状态；若 R 或任一重要成本未知，则不输出确定的 m。主任务接受不等于可计费， $A_{\text{billable}} < A$ 时按全体 A 分摊成本仍有必要，同时披露免费接受占比。

现金表只列实际支付和对应收款；经济成本表另计创始人、志愿者工时与自有设备机会成本，并说明估价依据。同一节点按可用小时收费时计入 F，不再把相同费用加到 c_x；按任务付费则计入 c_x。验证用量、工资和协调人工也按同一规则防止重复归集。当前没有实测成本、价格和付款证据，不能确认验证成本合理或商业成立。

容量先作上限约束，不能直接当销售预测。设 H_x/H_v 分别为执行/验证可用容量时间， t_x/t_v 为对应平均忙碌时间， u_x/u_v 为各自利用率， $a=A/X$ ， $v=V/A$ ；在规模稳定、没有重复计入容量时，A 不高于 $\min(H_x \times u_x \times a / t_x, H_v \times u_v / (t_v \times v))$ ，并受实际请求量和人工支持容量限制。执行和验证若共用机器，须划分 H_x/H_v ，不能把同一小时用两次；a、v 不稳定时只报告实测吞吐，不外推。

每批至少记录 V/X 、 V/A 、验证总费/执行总费、验证总费/A、p95 等待时间及失败/撤销/返工率。接受量降低或验证排队增长时，固定成本会进一步摊薄效率。验证费用大于执行费用本身不是逻辑错误，但没有用户愿意为复核证据多付钱时，不能建立低价网络叙事。

12.4 合成情景与敏感性

示意/合成数据，非 UNORYN 实测或获准预算：某一已结案批次 $A=A_{\text{billable}}=100$ ，假设对应净收入与净收款均为 $R=120$ 个货币单位， $X=120$ ， $V=110$ ； $c_x=0.20$ ， $c_v=0.30$ ； $W+S=5$ ， $O=20$ ， $F=20$ 。仅为保持算例简洁，假设本批 $J=0$ 且没有退款、未结义务与漏计人工；实际核算必须核实这些项。总成本为 $24+33+5+20+20=102$ ；单位成本 1.02、单位收入 1.20、全成本差额 0.18。验证/执行总费比为 $33/24=1.375$ ，验证费用占本例总成本约 32.35%；已有明显验证负担。不能据此声称项目盈利或验证成本可接受。

仍为示意/合成数据：若在相同固定支出下利用率下降，接受量降至 50，按相同比例有 $X=60$ 、 $V=55$ ，并假设 $W+S$ 随量减半为 2.5，则成本为 $12+16.5+2.5+20+20=71$ ；单位成本 1.42。若单位支付仍为 1.20，则每单位亏损 0.22。假设人工支持未随任务减少，必须在真实数据中检查。

仍为示意/合成数据：在首个情景中，若单次验证成本升至 0.60，总成本为 135，单位成本 1.35，超过单位支付。验证贵于执行未必绝对无用，但必须有愿意支付复核价值的用户；否则应改成批处理、托管参考运行、降低任务种类或停止该服务，不能依赖不断发行奖励掩盖亏损。

12.5 现金、融资与退出

代币市值是某价格乘相应数量的估值口径，不是公司银行余额；交易量是累计成交额，不是协议收入；出售所得只有在实际成交并扣除费用和义务后才可能成为资金。流动性池中的受约束资产、客户预付款与退款准备不能同时当作研发自由现金。

Token 名称、Ticker、链、发行量、分配、解锁、主体、司法辖区、TGE、交易所和做市均未确定。本稿没有设定 FDV 或融资目标，也不暗示任何未来收益。法律判断应围绕实际活动、用户地区、资金流和权利安排获得专业意见，不能靠“海外主体加境内研发”自动成立。

未融资时，建议限于现有设备、手工批准任务、小数据包和可停止的测试；一次只扩大一个变量。创始人须先确认可支配现金、每周工时及停止支出线。若关键贡献依赖尚未到位的全职工程能力，先交付本地脚本与证据格式，不承诺公开网络。资金不足时保留开源文档和可迁移数据，停止新增长期托管与补贴责任。

13 威胁、安全与数据权利

13.1 攻击模型

攻击者可能是恶意 Node、提交者、验证者、被入侵的依赖或拥有管理权限的人。目标包括获取不应有的贡献、耗尽资源、窃取资料、污染评测和改写历史。受控原型降低暴露面，不意味着不存在攻击。以下防护均为要求，尚无实测抗攻击结果。

威胁	目标与方式	Genesis 控制	剩余风险
伪造身份	多账号多 Node 刷贡献	邀请、设备关系登记、槽位计量	登记可撒谎，不能无许可防女巫
复制结果	不执行而领取工作信用	限制计量与委托、控制答案公开	输入可转发；检测效果未知，合格输出不证明执行

威胁	目标与方式	Genesis 控制	剩余风险
重放	旧提交重复接受	绑定尝试/代次、幂等和唯一键	实现缺陷或管理员绕过
串谋	节点与验证者共同作弊	披露控制关系、参考实现、抽检	小团队缺乏机构独立
虚报硬件时长	抬高补偿或声誉	自报仅诊断、能力挑战、不按自报付款	挑战可被转发或租用代跑
低质量数据	污染判断或刷上传量	许可与质量检查、反例、版本隔离	标签和规范可共同错误
基准泄漏	只优化已知答案	访问控制、候选登记、新留出复核	反复调试和相似数据泄漏
恶意任务	读文件、联网、耗资源	只读数据、白名单 Runner、上限	解析器和隔离缺陷
供应链污染	替换 Runner 或依赖	冻结摘要、审阅依赖、发布撤销	可信发布者失陷
密钥泄漏	冒充节点或管理员	最小权限、轮换、撤销、审计	既往影响范围可能不清
协调器作恶	审查、改写或选择性接受	导出、备份、操作记录	中心控制仍存在
资源耗尽	大包、长任务或请求洪泛	配额、尺寸限制、退避、暂停	公开环境需专门压力验证

13.2 受控与公开门槛

内部单机可以先验证状态语义，但不能因此发布面向陌生人的通用 Runner。受控外部测试要让参与者看到执行内容、资源限制、会访问的路径、是否联网、保存何种日志以及如何停止。默认拒绝个人文件和凭据访问，不要求导入钱包密钥来“绑定贡献”。

公开发布前必须有可验证的软件包来源、签名或同等真实性机制、最小权限执行、有效资源约束、任务网络白名单、文件访问隔离、版本撤销、紧急停止及漏洞响应。要测试恶意包、退出失败与撤销不可达，不只测试正常启动。未经独立安全验证不宣称可以安全运行任意第三方代码。

Docker 是环境封装和部分隔离工具，不自动解决宿主机权限、挂载、网络 and 内核漏洞。数字签名能帮助验证发布来源，不能保证发布内容无恶意。质押和多数投票不能保护用户文件系统。安全措施必须围绕实际权限和故障场景组合评估。

13.3 权利、保留和删除

输入、模型、输出和日志分别记录许可及可见性。任务提交者声明有权使用不等于平台已核实，应按风险审阅许可证与来源；不明确的内容隔离，不分发、不作为公开基准。个人数据不进入 Genesis 样本；后续引入时需要新的权限、保留和删除设计。

建议设置任务级 retention_policy，区分短期调试日志、产物证据和必要审计记录，具体期限待创始人确认。删除申请不能只删 UI 链接而保留可公开访问的对象；还应处理缓存、备份保留边界及引用失效。审计需求与删除要求冲突时保留最小必要元数据，明确范围，不把原始私人数据永久锚定上链。

公开 Dashboard 默认仅显示获准公开的摘要信息和化名，不显示私人 IP、绝对文件路径、设备序列号、邀请令牌或密钥。对输入做哈希也不必然匿名，低熵数据可能被猜测；秘密和个人资料不应以“只有哈希”为由公开。

14 治理、升级、争议与恢复

Genesis 采用维护者治理，公开说明任务审批、Runner 发布、验证政策和暂停权限。建议两名团队成员分配发布与复核责任，实际能力和工时仍待确认；只有一人可处理的环节应披露单点，不能在文稿中虚构独立委员会。

协议升级先发布新 schema 与兼容范围，测试旧客户端拒绝或正确升级。已经分发的 Task 固定旧版本，不能在一半节点运行后改变规则。数据库迁移先备份并在副本验证，再安排维护窗口；迁移失败可回滚到经验证副本，同时处理窗口内新增记录，避免简单恢复旧备份导致接受记录丢失。

暂停分为停止新领取、停止某任务家族、停止接受和撤销客户端版本。每次记录原因、影响对象和恢复条件。暂停后继续提交的处理由任务状态决定，不可临时凭好恶接受部分结果。重大安全事件先限制风险，再确认证据与范围，最后发布脱敏说明；不因担心数据难看而删除失败记录。

争议记录包含原决定、参与者主张、证据引用、受理时间、审阅人及结果。审阅者尽量不与原决定同一人；无法独立时明确说明。复核可能维持、撤销或要求重做，不能修改旧产物来使原决定看似正确。初期没有有偿结算，因此没有自动罚没；未来引入付款必须另定争议期间资金状态。

恢复演练至少包括数据库进程崩溃、服务重启、备份还原、产物缺失和凭据泄漏。备份需将数据库与产物清单对应，恢复后检查引用完整性、唯一约束及接受/贡献对账。单独复制正在写的数据库文件并假定完整不可接受；选用数据库支持的备份流程。[R15]恢复成功要有可复现日志，不以“文件能打开”代替。

15 更开放网络的前置条件

去中心化不是按日期打开的开关。第一步可以让外部用户自行导出和验证记录；第二步引入独立运营的复核者；再考虑对协调器选择性行为的公开审计。只有出现中心控制造成的实际限制，才需要比较更复杂的通信、存储或结算方案。

开放接入需要处理身份成本、任务垃圾、恶意包、复制答案、验证者串谋和争议资源。当前邀请制不能直接推广为无许可安全。候选方案可能包括挑战任务、可信硬件、可验证计算或多方复核，但各有覆盖、成本与信任依赖，不能在未做工作负载实验前宣称已采用或有效。

链上锚定只应在参与者确有跨主体审计需求、数据最小化和成本可承担时研究。P2P 应在任务传输和可达性需要支持时研究，不先承担任意 NAT、更新与端点安全。原生 Token 需满足第 12 章必要性，不能作为去中心化完成度指标。

训练研究另行立项，先回答同步带宽、检查点权限、恶意梯度、容错和评测。分布式推理或分类评测通过不代表这些问题已解决。若早期产品已通过托管评测满足用户，不应为保持宏大叙事强行进入训练协议。

16 实验、阶段验收与失败条件

16.1 G0 单机 workflow

假设是一个受限任务可以从注册到接受完整闭环。最小产物为一个 Runner、参考验证器、数据库、CLI 和 JSON 导出。前置条件是授权样本、字段冻结和基本资源限制。通过条件为正常路径、重复提交、错误结果、租约过期、取消竞争和重启恢复均符合附录测试；接受与贡献数量逐项相符。

失败时不扩节点，优先收窄任务种类、关闭非必要状态或修正幂等。没有必要为单机调试引入区块链、复杂图数据库或公网服务。G0 只证明实现基本语义，不证明市场与抗恶意能力。

16.2 G1 独立多节点

假设是不同真实设备能执行同一兼容任务并正确处理失联。最小产物是 2-5 台真实电脑的环境清单、运营者关系、运行记录和差异报告。前置条件为 G0 通过、受控授权及支持环境明确。通过条件为每台被列入支持范围的设备有成功及故障记录，结果差异可解释，不能只数 Node ID。

失败时只支持已验证设备，不为展示“多节点”而在同机启动多个进程充数。不同网络可补充可达性测试，但不是硬件独立的代替。两台机器跑通不证明规模、共识或商业成立。

16.3 G2 真实 Dashboard 与证据

假设是使用者能从展示追溯到真实记录。最小产物为只读页面及可导出证据包。前置条件为真实数据库事件；禁止用假数据冒充运行成绩。通过条件为任取任务可追到输入、尝试、验证和接受，重试不增加成功数，撤销改变当前有效指标，敏感信息不出现在公开字段。

失败时回到命令行和导出报告，先解决数据语义再做界面。模拟数据演示必须明显标“合成”，并与实测页面物理或逻辑隔离。可视化美观不是通过条件。

16.4 G3 外部使用与需求

假设是非团队用户有反复使用价值。最小产物为经同意的试用记录、授权任务、前后流程对照和再次使用证据。前置条件为受控测试安全门槛。建议先访谈 3-5 人，并以至少两名不同用户的无奖励二次使用作为初步继续依据，仍需检查任务是否真实。

失败后缩为本地证据工具或改变客户类别；不能通过空投预期、刷任务或付费买成员来修饰结果。付款意向、按条款形成的收入与实际收款分别记录；没有收款不报现金收入。

16.5 G4 成本与可靠性

假设是服务在完整验证后仍值得采用。最小产物为分任务家族的现金及经济成本表、失败和支持日志，以及用户可接受的时间/价格边界。前置条件为真实任务和完整计量。通过条件是按 12.3 节完成同批任务对账，现金和经济成本分别可解释，主要故障可恢复且不重复记账。商业扩张要求对应净服务收入覆盖完整经济成本，并用实际失败、验证排队与支持工时做敏感性检查；依赖资助时只可判为“预算内受补贴实验”，须记录资助期限与停止线，不能算商业验证通过。没有这些数据时 G4 保持未通过。

失败后比较批处理、可信托管验证、减少任务家族和直接本地运行，不降低未披露的验证强度来制造低价。若保留验证后长期没有可持续需求和资金来源，停止有偿扩张并公布实验限制。

16.6 G5 开放与经济层决策

假设是更开放参与解决已观察到的供需或信任限制。最小产物为对抗测试、独立安全检查、权利和争议规则、单位经济与资金方案。开放接入与支付方式分别评审；只有提出新的原生 Token 方案时，才额外要求提供现有结算不能满足的具体证据。前置条件是前述证据足够，且创始人批准费用和权利安排。

通过条件不是“10 个节点、200 个成员”，而是安全、需求、成本和适用活动条件均可承担。未通过则保持受控无币；这仍然是有效的研究结果。经济层参数必须单独立案，不由文稿作者代为承诺。

16.7 时间与成熟度

交接目标为 2026 年 9 月底 Genesis Ready、10 月 1-3 日小范围发布。截至本文编写日属于未来目标，未承诺。若没有通过外部测试门槛，窗口应改为设计审阅或内部演示，而不是强行上线。后续引用本稿时不得倒填这些阶段已经完成。

两人团队建议先分配任务协议/Runner 与样本/验证/需求访谈两组责任，但具体成员能力与工时待本人确认。延后独立研究任务，不要求同一周完成本地 AI、公开客户端、P2P、Token 和融资。文稿完整仅表示设计可审阅，不等于工程计划已经具备执行资源。

17 开放问题与参考使用

尚未解决的问题包括真实需求强度、验证低于可支付价值的任务区间、开放环境执行证明、独立运营者识别、方法贡献与许可争议，以及两人团队的维护容量。这些问题不阻碍内部受控实验，但分别限制外部测试、公开客户端和经济层。

本稿与 Litepaper 共用术语、状态及合成示例。引用清单提供官方来源和支持范围，不引入虚构融资、价格、市场统计或背书。W3C PROV/OpenLineage/MLflow 的存在提示优先复用，Bitcoin/GEODNET/Gensyn/Bittensor/io.net 的机制提示必须把服务、验证与经济分别证明。下一步工程应以本稿附录建立可失败的验收测试，而非将文稿措辞直接复制为产品成绩。

附录 A 最小实体模型

以下是拟议 schema，不是现有 API 的导出。所有实体含 schema_version、created_at、record_version；不可变载荷与可变状态分表或保留事件版本。ID 建议使用服务端生成的 UUID，示例中的 demo 前缀仅供阅读。摘要使用明确算法；对于 JSON 载荷，建议采用 RFC 8785 JCS 或同等被正式固定的规范化方案，不能用语言默认序列化的偶然键顺序。[R12]

A1 参与者与设备

Contributor 必填 contributor_id、display_name、status、terms_version；可选 operator_group_ref 为非公开的控制关系说明。身份验证等级须单独标记，默认 SELF_DECLARED，不自动称实名。Node 必填 node_id、contributor_id、key_id、client_version、capabilities、status；可选 device_ref、last_heartbeat_at。Node 与设备是一对多或多对一可能存在的关系，不作一一对应假设。

DeviceDeclaration 包含 device_ref、operator_ref、os_arch、hardware_claims、evidence_level、verified_at、verification_method。Genesis 只保存必要信息；公开报告可写“人工核对的两台设备”，

不公开序列号。更换设备后保留声明版本，不允许同一 Node 身份静默变成不同设备而继续沿用能力认证。

A2 任务与尝试

Task 必填 task_id、requester_id、task_family、task_role、spec_version、spec_digest、input_manifest_id、input_manifest_digest、runner_digest、verification_policy_id、verification_policy_digest、acceptance_policy_id、acceptance_policy_digest、accounting_policy_id、accounting_policy_digest、resource_limits、execution_deadline、review_deadline、max_attempts、state、visibility、license_refs。task_role 仅为 primary/validation；validation 必填 verifies_task_id，primary 不填。Slot 包含 slot_id、task_id、role、generation、active_attempt_id、accepted_ref；初版每个 Task 的 slot.role 都为 primary，统计必须使用 Task.task_role。

Attempt 必填 attempt_id、task_id、slot_id、node_id、contributor_id、key_id、generation、lease_expires_at、state、issued_at；contributor_id 与 key_id 在领取时从已授权 Node 绑定中固化，不从结果载荷采信，不允许事后改绑历史归属；可选 claimed_started_at、claimed_completed_at、failure_code。秘密租约令牌只提供给获分配 Node，数据库使用安全验证形式，不公开导出。服务端判断接收资格时还需检查该 Node 当前权限。

A3 产物与结果

Artifact 必填 artifact_id、content_digest、media_type、byte_length、storage_ref、license_ref、visibility；版本化声明可含 derived_from 和 author_claims。Result 必填 result_id、task_id、slot_id、attempt_id、generation、task_spec_digest、input_digest、output_artifact_id、environment_digest、payload_digest、received_at。冗余绑定字段必须与冻结 Task/Attempt 完全一致；Result.task_spec_digest 对应 Task.spec_digest，Verification.policy_id/policy_digest 对应 Task.verification_policy_id/verification_policy_digest。input_digest 对应 Task.input_manifest_digest，指完整输入 manifest 的摘要，而非把所有样本简单拼接所得摘要。result_id 与 output_artifact_id 不相同：前者记录谁提交，后者记录提交了什么。

环境清单包含 runner/package 摘要、依赖版本、OS/架构、硬件声明及适用的模型参数；不得含秘密。费用或时间自报 claimed，服务端测量标 observed，实际账单标 billed，避免同一 cost 字段混入不同可信度。

A4 验证、接受与贡献

Verification 必填 verification_id、result_id、policy_id、policy_digest、verifier_id、verifier_version、review_round、job_generation、outcome、reason_code、evidence_artifact_id、started_at、finished_at。outcome 为 PASS/FAIL/ERROR，不直接代表 Verified Delta 的四种比较结论。重复运行同一验证可有多个记录，但 policy 对“满足验证”的聚合规则必须固定。

Acceptance 必填 acceptance_id、task_id、slot_id、result_id、verification_set_ref、policy_version、policy_digest、accepted_at；policy_version/digest 对应 Task 冻结的接受政策。ContributionRecord 必填 contribution_id、acceptance_id、contributor_id、contribution_type、quantity、unit、accounting_policy_version、accounting_policy_digest；对应 Task 冻结的计量政

策；初版 quantity=1；主任务的 unit=accepted_primary_result_submission，验证任务的 unit=accepted_validation_result_submission，禁止暗含经济价格或算力时长。实际执行声明另附证据与审阅状态；没有可信观察时显示“执行未独立核实”。Knowledge/Intelligence 仍只登记许可、作者声明与采用关系，不由一次 Acceptance 自动生成四类经济贡献。Revocation 包含 revocation_id、target_ref、reason、evidence_ref、actor_id、created_at；当前有效计量通过关联撤销计算。

验证工作表 VerificationJob 必填 job_id、result_id、policy_id/digest、review_round、job_generation、assigned_verifier_id、lease_expires_at、state、retry_count 和 max_retries；当前轮次及代次只能由协调器在事务中更新。Verification 还须包含 job_id，不能只凭 result_id 加一个 PASS 写入。

Artifact 存储记录应区分 STAGED、READY 和 QUARANTINED，并记录引用或保留锁。只有完整字节已持久化且摘要通过的 READY 对象可以关联 Result；已关联待审结果、有效接受或受保留政策约束的对象，垃圾回收不得删除。删除和引用建立须经同一元数据事务仲裁；文件删除在获准后执行，恢复扫描负责修复未完成的删除工作。

A5 连贯示例

示意/合成数据，非 UNORYN 实测。以下用 symbolic 字段明确表示摘要占位符，不是有效 SHA-256；不能直接提交给未来接口。随附 JSON 文件将生成可校验的完整示例摘要，并保留 synthetic=true 标签。

```
{
  "synthetic": true,
  "schema_version": "0.1-proposed",
  "task_id": "demo-task-001",
  "task_family": "offline-rule-loop",
  "task_role": "primary",
  "slot_id": "primary",
  "attempt_id": "demo-attempt-002",
  "node_id": "demo-node-a",
  "generation": 2,
  "result_id": "demo-result-002",
  "output_artifact_id": "demo-artifact-a",
  "digest_symbolic": "SHA256_OF_CANONICAL_OUTPUT",
  "verification_id": "demo-verification-002",
  "verification_outcome": "PASS",
  "acceptance_id": "demo-acceptance-001",
  "contribution_id": "demo-contribution-001",
  "contribution_type": "Compute",
  "quantity": 1,
  "unit": "accepted_primary_result_submission",
  "economic_entitlement": null
}
```

该示例表示第一次尝试可能失败，第二次在同一主槽位被接受；两次尝试不等于两项有效主任务。真实 schema 应检查同一 contributor 关系、外键、时序和枚举值；读取 JSON 成功并不表示通过协议验证。

附录 B 拟议接口草案

路径前缀为 /v0。所有写请求要求身份验证、权限检查、请求大小限制与 Idempotency-Key；GET 不具有写副作用。示例没有可调用域名，本轮没有部署接口。Task 创建和 Runner 发布仅向维护者开放，Node 不能把字符串命令提交为任务。

方法与路径	角色与请求	成功响应	关键错误
POST /v0/nodes	邀请持有者；声明和客户端版本	201 Node 与单次凭据	邀请失效、版本不支持
POST /v0/nodes/{id} /heartbeat	自身 Node；状态和 Attempt	200 服务端时间	401 撤销、403 越权
POST /v0/claims	Node；支持家族与空闲槽位	200 租约；204 无任务	409 幂等冲突、429 限流
POST /v0/tasks	维护者；冻结 TaskSpec	201 task_id	422 包/许可/政策不完整
GET /v0/artifacts/{id }	获该产物权限者	200 字节及摘要	403 无权、404 不存在
POST /v0/attempts/{id}/results	分配的 Node；绑定结果	202 result_id、SUBMITTED	409 过期/代次冲突；422 无效
POST /v0/tasks/{id}/cancel	提交者或维护者	200 CANCELLED	409 已接受；403 越权
GET /v0/tasks/{id}	有读取权限者	200 状态与可见证据	403 或 404
POST /v0/verifications	验证服务；固定 Result 判定	201 verification_id	403 非验证者；409 版本冲突
POST /v0/revocations	维护者；原因和证据	201 撤销记录	422 缺依据；403 越权
GET /v0/contributions	参与者或公共视图	200 分页有效记录	无权字段不返回

内部验证服务的接受动作可以是服务函数而不是公开 HTTP 路由；禁止 Node 通过直接调用接受接口给自己记账。Claim 的幂等响应返回原租约，即使其后来已过期也不能隐式生成新租约；响应同时返回当前资格状态。客户端明确要新任务时使用新请求键，防止重试歧义。

标准错误信封包含 error_code、message、request_id、retryable 与可选 retry_after_seconds；不回传堆栈秘密。区分 LEASE_EXPIRED、STALE_GENERATION、TASK_CANCELLED、PAYLOAD_CONFLICT、SCHEMA_INVALID、POLICY_MISMATCH、VERIFIER_UNAVAILABLE 与 STORAGE_UNAVAILABLE。HTTP 状态只作传输分类，程序用稳定 error_code 决定后续行为。

提交成功响应 202 只说明结果已持久化等待验证，不表示 Accepted。服务端应允许查询 result_id 的验证与接受状态，客户端在超时后先按原幂等键重试或查询，不自行换键复制结果。列表采用稳定游标并注明查询时间，避免统计正在变化时不同页面重复或漏计。

认证信息不写入 URL、公共 artifact 或日志。注册响应中的秘密凭据不能作为普通幂等响应明文长期存储；建议以客户端生成密钥的注册握手替代，或将单次秘密响应加密短期保存并明确恢复/重发策略。幂等键仅保证同一授权请求的语义，不是身份秘密。对于被撤销 Node 的旧提交重试，先返回权限失败；原已接受记录仍可按参与者读取权限查询，不因为禁止重试而抹除历史。

附录 C 幂等与事务伪代码

C1 建议约束

```
-- Proposed constraints, not a deployed migration.
UNIQUE(actor_id, route, idempotency_key)      -- request_dedup
UNIQUE(task_id, slot_id, generation)          -- attempts
UNIQUE(attempt_id)                            -- valid_results
UNIQUE(task_id, slot_id)                      -- acceptances
UNIQUE(acceptance_id)                        -- contribution_records
UNIQUE(result_id, policy_id, review_round)    -- verification_jobs
```

Acceptance 唯一约束保留历史：撤销后不能重用同一槽位插入第二个历史 Acceptance。若需重新完成，创建 replacement Task 并引用旧任务；这样避免撤销使重复接受约束失效。计量更正用 Revocation/Adjustment 记录，不再创建第二条初始贡献。字段和状态更新还需条件检查，唯一索引不是完整状态机。

C2 接收结果

以下为事务流程草案，服务辅助函数尚未实现。资格、权限、暂停与撤销状态均在同一个数据库写事务中重新读取；管理员修改这些状态也必须经该数据库提交，不能只改进程缓存。密钥正常轮换可以接受新的有效 key，但必须属于同一获分配 Node；历史 Contributor 归属保持 Attempt 领取快照。

```
def submit_result(actor, attempt_id, key, payload, lease_token):
    authorize_request(actor, attempt_id)
    digest = canonical_digest({"attempt_id": attempt_id, "payload": payload})
    # For a new result, bytes are already durable and immutable.
    with db.immediate_transaction():
        require_current_node_permission(actor, attempt_id)
        old = request_dedup(actor.id, "submit", key)
        if old:
            require(old.payload_digest == digest, "PAYLOAD_CONFLICT")
            response = old.response
        else:
            task, slot, attempt = load_bound_objects(attempt_id)
            prior = valid_result_for_attempt(attempt.id)
            if prior:
                require(prior.payload_digest == digest, "PAYLOAD_CONFLICT")
                response = original_submission_receipt(prior)
            else:
                require(task.state == "IN_PROGRESS", "TASK_NOT_ACTIVE")
                require_submission_controls_open(task, attempt)
                require_valid_lease_token(lease_token, attempt)
                require(slot.active_attempt_id == attempt.id, "STALE_GENERATION")
                require(payload.generation == attempt.generation == slot.generation,
                        "STALE_GENERATION")
                require(attempt.state in ["LEASED", "RUNNING"], "ATTEMPT_CLOSED")
                require_schema_bindings_and_durable_artifact(payload, task, attempt)
                received_at = server_now()
                require(received_at < attempt.lease_expires_at, "LEASE_EXPIRED")
                require(received_at < task.execution_deadline, "DEADLINE")
```

```

        result = insert_valid_result_unique(attempt.id, payload, digest,
                                           received_at=received_at)
        set_attempt_state(attempt, "SUBMITTED")
        enqueue_verification_in_database(result, task.verification_policy_id)
        response = submission_receipt(result)
        save_dedup_response(actor, "submit", key, digest, response)
    return response # every branch returns only after a successful commit

```

幂等摘要覆盖 attempt_id 路由参数和规范化载荷，不含认证秘密；同一主体在提交路由复用同一键却更换 Attempt 时，必须报 PAYLOAD_CONFLICT，不能返回另一 Attempt 的收据。同 Attempt 同载荷换键重发时，必须先找既有 Result，再判断新提交资格；否则 SUBMITTED 状态检查会错误拒绝重试。原响应是提交收据，不代表当前接受状态；重试不得重启验证或恢复已关闭资格，当前状态另查。换键换载荷始终冲突。复用响应前仍检查当前权限，被撤销凭据不能靠幂等绕过授权。唯一约束冲突不得盲目视为成功，须回滚并按上述顺序重查载荷。

大产物在事务外按大小和摘要验证后写入不可变、可持久读取的对象；事务内只核对已准备 manifest、摘要和绑定，不做下载、文件扫描或运行验证。准备完成到引用之间不能允许替换同一 storage_ref 的字节或清理其保留锁；存储真实性和写入持久性仍是受信前提。未关联对象按保留规则清理，恢复扫描要发现关联缺失或内容损坏并阻止接受。文件存储与数据库没有共同事务，需要实际崩溃测试。

C3 接受判定

```

def accept_verified_result(result_id):
    authorize_internal_verifier()
    with db.immediate_transaction():
        task, slot, attempt, result = load_for_acceptance(result_id)
        old = existing_acceptance(task.id, slot.id)
        if old:
            decision = existing_decision_with_revocation_state(old)
        else:
            require(task.state == "IN_PROGRESS", "TASK_NOT_ACTIVE")
            require_acceptance_controls_open(task, attempt, result)
            require(server_now() < task.review_deadline, "REVIEW_EXPIRED")
            require(not revoked(result), "RESULT_REVOKED")
            require(result_is_valid_timely_submission(result), "INELIGIBLE")
            require(attempt.id == slot.active_attempt_id, "STALE_RESULT")
            require(result.generation == attempt.generation == slot.generation,
                   "STALE_GENERATION")
            require(attempt.state == "SUBMITTED", "INELIGIBLE")
            checks = load_current_policy_verifications(task, result)
            require_frozen_policy_satisfied(task, result, checks)
            require_ready_referenced_artifacts(result, checks)
            acceptance = insert_acceptance(task, slot, result, checks,
                                           policy=task.frozen_acceptance_policy)
            insert_contribution_unique(acceptance, attempt.contributor_id,
                                     task_role=task.task_role,
                                     policy=task.frozen_accounting_policy)
            set_slot_accepted_ref(slot, acceptance.id)
            append_audit_event("ACCEPTED", acceptance.id)
            set_task_state(task, "ACCEPTED")
            decision = acceptance
    return decision

```

controls_open 必须逐项检查：接受总开关、该 task_family 的接受开关、Task 未被暂停、Runner/政策/输入及输出版本未撤销，Node 与领取时 Contributor 当前具有接受权限。仅停止新领取不会自动禁止

已提交结果的接受；“停止接受”开关则必须禁止。凭据泄漏如可能污染旧提交，要在同一事件中暂停相应未接受结果并审阅，不能靠换 key 消除影响。

checks 由服务端从数据库加载，不采信调用者给出的 verification_set 或 PASS。每条检查必须绑定同一 result_id、被冻结的 policy_id/digest、允许的 verifier_version、当前 review_round 与 job_generation，且未撤销；验证者权限仍有效，证据对象已保存，验证完成于 review_deadline 之前。首版政策只允许受控参考验证器的一份有效 PASS；旧代次、FAIL、ERROR 或有未决冲突均不满足。额外受委托复核单独计工作，不以票数覆盖必需检查，也不宣称机构独立。

截止检查时间由事务内服务端确定，事务最终成功才生效。超时扫描将任务阻塞，接受入口自身也必须拒绝超时，不能依赖扫描恰好先跑。Result、Artifact 准备状态和撤销资格须在事务检查；内容损坏检测后立即冻结相关引用。

若已有 Acceptance，返回该既有决定及当前撤销状态，不能暗示当前另一个 Result 获得接受。贡献归属从不可变 Attempt.contributor_id 读取，不访问未定义的 result.contributor_id，也不接受客户端自填。接受、槽位、Task、贡献与审计记录在同一事务成功或回滚；这只限制数据库重复记账，未证明执行、付款完成或管理员诚实。

C4 验证工作恢复

当前 review_round 与 job_generation 由持久化验证工作表原子维护，worker 不得自选。验证 worker 从数据库领取 job lease，在事务外检查固定 Result。完成提交也须在短事务中检查

assigned_verifier_id、当前验证者权限、policy_id/digest、review_round、job_generation、工作仍为 LEASED/RUNNING，且服务端资格时间严格早于 job.lease_expires_at 与 Task.review_deadline。仅代次未变化不够：工作即使尚未重派，到期后也不能产生活跃 PASS。同一有效工作重复完成只返回原收据，同键换 outcome 或 evidence 摘要应拒绝；旧工作最多留诊断审计。然后用 job generation 条件写 Verification；旧工作超时后才完成时，旧代次不得覆盖新判定。检查结果可保留为历史，但聚合只读取符合政策的有效 review_round。Worker 崩溃由扫描器重新排队；反复 ERROR 达限后转 BLOCKED，不扣 Node 分。

服务端重启还需扫描临时产物、未关联上传和待更正投影。扫描本身必须幂等，同一对象处理两次不产生两个任务。设计不依赖消息“绝不丢失”，而依赖可从持久事实重建工作。

附录 D 状态转换与指标

对象与当前状态	事件	下一状态	计量影响
Task CREATED	输入与政策审核通过	QUEUED	无贡献
Task QUEUED	合法 Claim	IN_PROGRESS	Attempt +1
Attempt LEASED	收到开始事件	RUNNING	仅遥测
Attempt LEASED/RUNNING	及时有效提交	SUBMITTED	Result +1
Attempt LEASED/RUNNING	超时	EXPIRED	不计有效贡献
Task IN_PROGRESS	执行失败且可重试	QUEUED	新代次待派
Result 待验证	政策通过	验证 PASS	Verified 去重计数
Result 待验证	内容不合格	验证 FAIL	保留失败证据
Result 待验证	基础设施异常	验证 ERROR	验证重试，不罚 Node
Task IN_PROGRESS	验证 ERROR 达限或 review_deadline 到期	BLOCKED	停止自动接受与重派
Task BLOCKED	未过审阅期且维护者批准恢复	IN_PROGRESS	仅恢复验证，不增贡献
Task 未终态	取消先提交	CANCELLED	不再接受
Task IN_PROGRESS	有效验证且接受事务通过	ACCEPTED	按 task_role 计接受和贡献，不混主任务/复核
Task 未接受	执行预算耗尽	FAILED	逻辑失败 +1
Acceptance 有效	撤销获批准	历史保留、当前失效	当前有效计量扣除

Task BLOCKED 恢复验证须有维护者决定和原因，且尚未超过原 review_deadline；审阅过期只能保留诊断、取消或另建 replacement Task，不能延长原接受资格。取消仍可适用。Accepted Task 不直接改回 QUEUED；需要 replacement Task 重新运行。对于验证 FAIL 重新派发执行，应先使原 Result 的资格关闭、记录原因，再分配新代次，避免旧验证后续翻转导致两个结果争抢。

指标导出同时包含时间范围、task_family、role、政策版本和有效性口径。若对比不同时间段，应控制任务规模和工作种类。累计接受流量与当前有效存量不可混用；基准点估计与工作完成数量也不能相乘得出“总智能增长”。

附录 E 最小测试矩阵

以下为待实现验收，不是本轮已经运行的协议测试。本轮新增 T37-T44 覆盖修订发现的边界；附带的文稿数值检查不等于这些系统测试通过。每条应保留输入、版本、故障注入、数据库快照和断言结果；仅写“通过”而没有证据不足以推进阶段。

编号	故障或操作	必须观察到的结果
T01	正常任务一次完成	一个 Result、一个 Acceptance、一个贡献
T02	相同键相同载荷重发	返回原响应，不增记录
T03	相同键修改载荷或 attempt_id	PAYLOAD_CONFLICT，不覆盖、不返回其他尝试收据
T04	SUBMITTED/已过期后同 Attempt 换键重发	有权限者同载荷取得原收据，不重排验证；异载荷冲突
T05	资格检查时恰到租约截止	严格小于规则拒绝；检查在前且最终提交成功的响应可晚到
T06	过期后新代次已派	旧代次正确答案也不接受
T07	提交已持久化后执行租约过期	继续验证，不另派主任务
T08	Node 离线无提交	过期后有限重试，保留历史
T09	提交响应丢失	原键重试可找回 result_id
T10	取消与提交并发	事务顺序决定，无取消后新资格
T11	取消与接受并发	先取消则拒绝；先接受则取消冲突
T12	验证服务崩溃	重试同 Result，不扣 Node 分
T13	验证 FAIL	保存证据；按预算重派或失败
T14	两个验证 worker 重复完成	仅有效代次判定；一次接受
T15	调度器重启	从 DB 恢复，不重复主贡献
T16	DB 提交前后注入崩溃	无半条接受/贡献；响应可重查
T17	DB 忙或磁盘满	明确失败/重试，不虚报成功
T18	产物上传中断或未关联	不接受；孤立对象可清理
T19	包摘要、schema 或权限错误	分发/提交被拒，证据不外泄
T20	路径逃逸、大文件、超时	被限制；不读写用户目录
T21	未知依赖或联网请求	白名单之外拒绝

编号	故障或操作	必须观察到的结果
T22	凭据撤销后重试与 Claim	拒绝新行为，历史仍可审阅
T23	两 Node ID 在同机	报告一台设备，不虚增独立性
T24	注入一个已知共同实现错误	经独立人工核对的黄金反例发现该错误；不推断可发现所有共同错误
T25	基准泄漏与不完整预测	Delta 为证据不足，不报改善
T26	合成 F1 和成本案例	算术与本文一致，标签保持合成
T27	图派生回环和悬空引用	拒绝；普通争议引用不误禁
T28	撤销后再提交旧槽位	不生成第二个 Acceptance
T29	备份恢复含产物缺失	检测不完整，不显示可复现
T30	Dashboard 对账和字段脱敏	复核 Task 即使有 primary 槽位也不计客户主任务；无秘密或私址
T31	BLOCKED、暂停接受与 review_deadline 竞争	全部入口拒绝新接受；恢复不延长截止
T32	伪造/旧代次/撤销的 PASS	只加载数据库当前授权判定，拒绝不匹配检查
T33	领取后试图改绑 Contributor	历史贡献归属不变，客户端字段不能覆盖
T34	复制正确答案、转发任务与积分刷量	记录可绕过的控制及误报；输出合格不升级为执行证明
T35	Delta 恰等于门槛、缺行、重复行和零基线资源	符合 10.6 边界；证据不足优先，不删失败样本
T36	已准备产物被替换或恢复缺失	读取/接受检测失败，隔离受影响结果并记录

测试完成后还需用真实授权任务观察性能，不从小型故障矩阵推断全球扩展性。安全测试中未出现漏洞也不是不存在漏洞的证明；公开分发需另作独立检查。矩阵应随实际实现补充，而不是用文档规定掩盖未覆盖场景。

E1 本轮补充的边界验收

编号	注入情境	必须观察到的结果
T37	中文、CRLF、多个及重叠命中	UTF-8 字节位置一致，排序和支持范围符合冻结规则
T38	重复 JSON 键、非有限数、乱序记录	拒绝非法对象；合法记录规范化后得到相同摘要
T39	同 Node 正常换 key 后重复提交；错误租约令牌新提交	已有收据不重复计数；错误令牌拒绝；撤销凭据仍拒绝
T40	验证工作到期但尚未重派时返回 PASS	不能改变有效验证与接受状态；只留诊断记录
T41	运行中替换接受或计量政策	旧 Task 继续绑定冻结政策，不能偷换单位或接受条件
T42	垃圾回收与 Result 引用竞争，随后崩溃恢复	元数据事务防止删除保留对象；缺失对象冻结接受
T43	等于阈值、无效数值、共享复核成本	精确分数判定边界一致；无效值或缺少分摊证据为不足
T44	导出后发生撤销	离线包明确快照时间，不能宣称持续有效；当前查询显示撤销

附录 F 引用索引

R1 Bitcoin 白皮书；R2 GEOD Token Introduction；R3 GEODNET 站点质量与奖励；R4 Gensyn Protocol；R5 Gensyn REE v0.6.0 页面；R6 Bittensor Yuma Consensus；R7 W3C PROV Overview；R8 OpenLineage 1.53.0 Object Model；R9 MLflow Tracking；R10 Celery 5.6.3 Tasks；R11 SQLite Transaction；R12 RFC 8785 JCS；R13 io.net 官方文档及索引。完整 URL、访问日期、所支持主张与未核验边界见随附参考资料。引用只支持外部机制先例，不证明 UNORYN 已实现任何对应能力。

可核查来源链接

来源复核日期：2026-09-12。官方文件仅支持文中机制说明，不等于独立性能测量或 UNORYN 实现证据。版本号指所读取页面；主张与边界见随附参考说明。

R1 [Bitcoin whitepaper \(2008\)](#)
R2 [GEOD Token Introduction](#)
R3 [GEODNET station performance](#)
R4 [The Gensyn Protocol](#)
R5 [Gensyn REE v0.6.0](#)
R6 [Bittensor Yuma Consensus](#)
R7 [W3C PROV Overview \(2013\)](#)
R8 [OpenLineage Object Model 1.53.0](#)
R9 [MLflow Tracking](#)
R10 [Celery Tasks 5.6.3](#)
R11 [SQLite Transaction](#)

R12 [RFC 8785 \(2020\)](#)

R13 [io.net Proof of Work](#)

R13b [io.net Cloud Payments](#)

R14 [SQLite Foreign Key Support](#)

R15 [SQLite Backup API](#)