OTCHive

Whitepaper

Trustless escrow infrastructure for Telegram-native Web3 work on TON.

v3.0 · Open Beta · June 2026

Field	Value
Document version	v3.0 — Open Beta edition (supersedes Whitepaper v2.9)
Product status	Open Beta — Mini App live, backend in production, escrow contract on TON mainnet
Website	https://otchive.com
Mini App	https://app.otchive.com
Main bot	@OTCHiveBot
Support bot	@OTCHiveSupport_Bot
Audience	TON ecosystem partners, grant evaluators, technical reviewers, prospective contributors

Contents

A. Abstract

OTCHive is a Telegram-native marketplace for crypto and Web3 services, secured by a TON smart-contract escrow. Clients post tasks, executors bid, and every funded deal is held on-chain until the client confirms delivery — through a single contract call named AcceptWork — or one of the documented escape paths (CancelDeal, RefundAfterDeadline, admin-resolved dispute) is triggered. The platform never custodies funds; the backend coordinates UX and notifications around an on-chain primitive.

This paper describes the product as it stands in Open Beta (June 2026): a working Mini App, a production backend on Railway, an escrow contract deployed on TON mainnet, a Trust Score reputation system in its second generation, an internal retention-points layer (Hive Points), a Premium subscription, an L1 referral layer (multi-level planned), and an Admin Command Center for operational supervision. It also outlines what is intentionally not promised — there is no public HIVE token live today, no committed conversion from Hive Points to any token, and no retroactive airdrop guarantee.

OTCHive's thesis is straightforward: the Web3 freelance market today loses billions of dollars per year to payment fraud, and the marketplace where it happens — Telegram — has no native escrow layer. We are building that layer on TON, the network that grew up inside Telegram.

B. The Problem

B.1 A Trust Gap You Can Measure

Most P2P crypto-services deals — smart-contract audits, DeFi development, NFT design, on-chain marketing — are negotiated and paid for inside Telegram. There is no escrow. There is no reputation that travels with the user. There is no recourse when one side reneges.

Two observable failure modes drive the problem:

- Pre-pay fraud — client pays upfront; executor ghosts.
- Post-delivery fraud — executor delivers; client refuses to pay and disappears.

Industry estimates place annual losses to crypto payment fraud at multiple billions of dollars. Beyond the direct losses, the trust gap suppresses transaction volume — work that should happen doesn't, because either side can't afford the counterparty risk.

B.2 Why Existing Solutions Fail

Approach	Why it fails for this market
Traditional freelance platforms (Upwork, Fiverr)	Fiat-only payouts; do not pay in TON / USDT; KYC requirements wall out anonymous Web3 contributors; not native to Telegram.
Manual escrow agents ("crypto OTC trusted middlemen")	Single-point-of-trust; opaque; doesn't scale; the agent is the systemic risk.
DAO bounty boards	Discovery problem (you have to know where to look); no chat layer; no Telegram presence; weak reputation portability.
Generic crypto wallets with multisig	UX is professional-grade; the average user — including most Web3 freelance clients — won't use it.

C. Market Context

Three macro trends create the window OTCHive targets.

C.1 Telegram Has Distribution

Telegram passed 1 billion monthly active users in 2025 and continues to grow. The Mini App platform — launched and stable — turns the messenger into an app store with zero install friction. A Web3 product distributed as a Telegram Mini App reaches a user inside an app they already open many times per day.

C.2 TON Has Wallet Penetration

TON is no longer experimental infrastructure. With tens of millions of monthly active wallets across @wallet, Tonkeeper, TON Space, and MyTonWallet, mainstream users transact on TON without thinking about it. Fees are sub-cent. Confirmations are seconds, not minutes. For an escrow product, those properties matter more than abstract throughput numbers.

C.3 Web3 Freelance Demand Is Real

The Web3 freelance market — smart-contract auditors, protocol engineers, DeFi marketers, NFT designers, on-chain analysts — is large and growing fast. The talent is on Telegram. The clients are on Telegram. The settlement layer they need was missing until now.

OTCHive is positioned at the intersection: a Telegram-native marketplace, TON-settled, with the trust layer built in.

D. The OTCHive Solution

D.1 Core Principle

> *Funds move only when the contract says so. No platform, no admin, no intermediary can route an escrow payment to a third address.*

The OTCHive escrow contract is single-purpose: it holds funds for one specific deal between exactly two participants. There are no upgrade hooks that allow the platform to drain a deal. There are no admin keys that allow the platform to redirect a payout. The platform's commission is a fixed fraction set at deploy time.

D.2 What OTCHive Provides on Top

- Discovery — a marketplace where clients post tasks and executors bid
- Identity — Telegram-native sign-in; portable Trust Score reputation
- Coordination — chat, notifications, milestone tracking
- Conflict resolution — a transparent dispute process with admin arbitration today and a documented decentralization path
- Retention — Hive Points, Premium, Boost, and an L1 referral economy (multi-level planned)
- Safety — Anti-Scam pre-checks, Trust-Score-aware ranking, wallet-change rate limits

D.3 What OTCHive Explicitly Does NOT Do

- Does not custody user funds at any stage
- Does not require KYC for ordinary marketplace use (subject to future regulatory developments)
- Does not promise a public token, an airdrop, or a guaranteed conversion from Hive Points
- Does not auto-resolve disputes by AI alone — AI assists; humans decide

E. Why Telegram-Native

OTCHive is built as a Telegram Mini App, not a standalone web app with a Telegram bot bolted on. This is a structural choice with real consequences.

E.1 Zero-Install Distribution

Users open OTCHive from any chat that mentions @OTCHiveBot. No App Store, no Google Play, no domain to remember, no password to recover. For a market where users discover services from someone they already trust in a private group, this matches their behaviour exactly.

E.2 Identity Without a Login Screen

Telegram WebApp's HMAC-signed initData is a free, cryptographically verifiable identity. We don't run a password database. We don't run an account-recovery flow. We don't run "forgot your email" support tickets. The Telegram account is the account.

E.3 Notifications That Land

Every Mini App user already accepts Telegram notifications. OTCHive piggybacks on this — our notification layer routes through @OTCHiveBot, with per-user toggles (orders / messages / system). A new bid lands in the user's main Telegram notification stream. There is no separate channel to opt into.

E.4 Trade-offs We Accept

- Discovery is constrained to Telegram surfaces (channels, groups, direct shares); SEO is harder
- We are downstream of Telegram's policies, including ad-policy and Mini App approval standards
- Some advanced UI (e.g. complex multi-window flows) is harder than on web — we design around this

F. Why TON / GRAM

F.1 Settlement on TON

TON's properties suit escrow specifically:

- Sub-second to a few-second confirmation; users don't wait
- Sub-cent transaction fees; OTCHive doesn't pass meaningful gas costs on to users
- Native wallet ecosystem inside Telegram (@wallet, TON Space) — no separate wallet install
- TonConnect 2.0 — a clean wallet-bridge protocol that works inside Mini Apps
- Smart contract languages (FunC, Tact) with reasonable tooling and verification stories

F.2 GRAM as the User-Facing Label

Inside the OTCHive UI, monetary amounts are labeled GRAM. "GRAM" is a display convention — the underlying asset is TON, the network is TON, and on-chain settlement is in TON. GRAM is what the user sees on a bid card, in an escrow status, on a payout receipt.

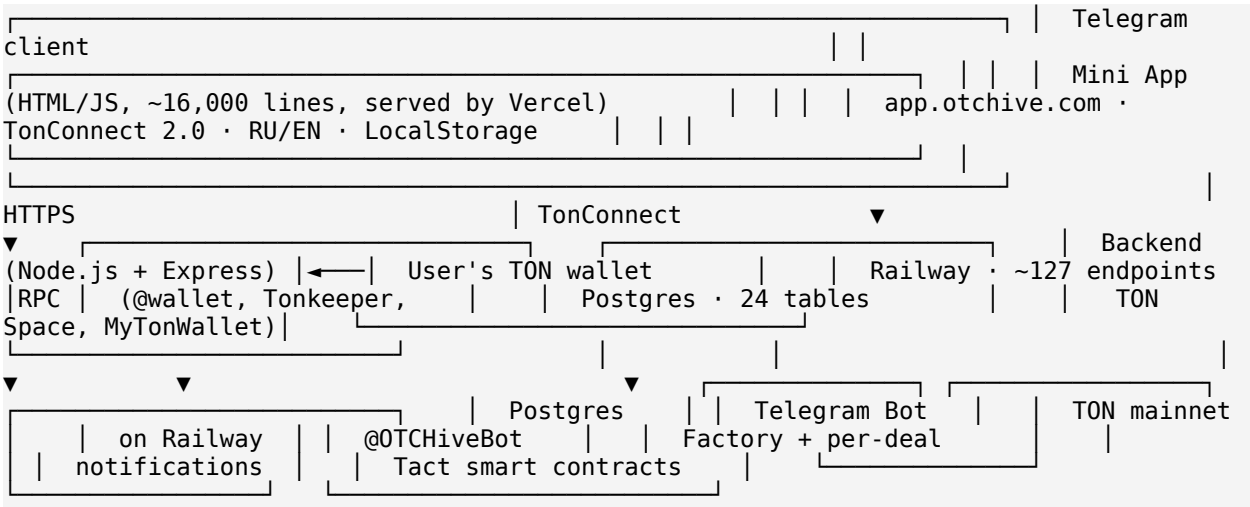
The motivation is product, not technical: "5 TON" reads as a crypto-trader unit; "GRAM 5" reads as platform money. When users think of OTCHive money the same way they think of in-game currency or platform credit, the cognitive friction of using an on-chain product drops materially.

F.3 The Phrasing Convention

When you need…	Use…
A balance, a price, a payout amount in UI	GRAM
A wallet address, a network confirmation, the TonConnect flow	TON
Both layers together ("escrow payments use GRAM on TON")	GRAM on TON
A future ecosystem token (planned, not live)	HIVE — and only in the planned/internal sections

G. Product Architecture

G.1 The Five Moving Parts



G.2 Stack at a Glance

Layer	Choice	Notes
Frontend	Single-file HTML/JS Mini App	No SPA framework — startup speed matters in a messenger context.
Backend	Node.js, Express, pg	Direct SQL, no ORM. ~6,600 lines. 127 endpoints.
Database	PostgreSQL	Railway-managed. 24 application tables.
Smart contract	Tact (compiles to FunC)	Factory + per-deal pattern. Source code committed.
Hosting	Railway (backend), Vercel (frontend, admin, landing)	Standard managed PaaS.
Wallet connect	TonConnect 2.0	Standard TON wallet bridge.

G.3 Engineering Posture

- Parameterised SQL, never string concatenation
- HMAC-verified Telegram WebApp authentication on every user endpoint
- Idempotency keys on critical writes (chat send, dispute resolve, escrow steps)
- Cross-table integrity invariants exposed via /api/admin/integrity-check
- Kill switches for new orders / hires / escrow / premium / boost, controlled via env

H. User Roles

H.1 Client

Posts orders, reviews bids, hires an executor, funds the escrow contract, accepts the delivered work (AcceptWork) or initiates a documented escape path. Optional Premium subscription unlocks a lower commission tier, unlimited orders, AI translation in chat, and priority feed placement.

H.2 Executor

Browses the exchange feed, submits bids on open orders, delivers the work, calls SubmitWork on the per-deal contract to mark delivery, and receives the funds (minus platform commission) when the client calls AcceptWork. Free executors are bid-limited per day; Premium executors are not.

H.3 Admin

Operates the platform via the Admin Command Center. Reviews disputes, resolves them through documented on-chain calls (AcceptWork, RefundAfterDeadline, split-payout), monitors system health, and audits its own actions through the admin_actions log. Admin cannot custody funds, cannot send escrow to a third address, and cannot bypass the on-chain truth — the contract is the constraint.

H.4 Bots

Two Telegram bots serve operational roles.

- @OTCHiveBot — the user-facing bot. Sends notifications to clients and executors, validates Mini App initData, hosts the main entry point.
- @OTCHiveSupport_Bot — the staff-facing bot. Posts dispute alerts and other operational signals into the internal Staff Group.

I. Escrow Flow

The happy-path lifecycle of an OTCHive deal, end to end. Every step has a backend endpoint behind it and an on-chain reconciliation guard.

1. Client posts an order. Anti-Scam pre-check screens the description (planned via backend; see §T).
2. Executors submit bids. Free executors are limited to 5 bids per day; Premium executors are not.
3. Client picks an executor. The backend prepares escrow-deploy parameters.
4. Client funds the escrow. The Mini App constructs the unsigned transaction; the user signs in their TON wallet via TonConnect; the wallet broadcasts.
5. Backend observes the transaction on-chain (or accepts a frontend confirmation, then verifies). Escrow status becomes Funded.
6. Both parties receive a Telegram notification: "Funds locked in escrow."
7. Executor delivers the work. Calls SubmitWork on the per-deal contract. Backend verifies and marks the escrow Submitted.
8. Client reviews the delivery. If satisfied, signs AcceptWork. The per-deal contract releases funds: (deal_amount × (1 − commission_rate)) to the executor wallet, and the commission to the platform treasury.
9. Backend reconciles the on-chain truth: order status moves to Done, escrow status to Completed, and +20 Hive Points are awarded to both sides (once per deal per role, enforced by the points_transactions audit table).
10. Optional: each side leaves a review. A 5-star review awards the recipient +30 Hive Points (once per review, ON CONFLICT-guarded).

Escape paths exist for every state, documented in §R.

J. Smart Contract Security Model

J.1 Non-Custodial by Design

The OTCHive backend never touches deal funds. The flow is:

- Client funds the per-deal contract directly from their wallet
- Per-deal contract holds funds until AcceptWork / RefundAfterDeadline / CancelDeal
- Executor receives funds directly to their bound wallet; commission goes to platform treasury

There is no platform-controlled hot wallet that holds escrow principal.

J.2 What the Per-Deal Contract Guarantees

- Two beneficiary addresses (client and executor) are baked into the contract's initial state at deploy. They cannot be changed afterwards.
- Commission rate is set at deploy time and cannot be raised on an active deal.
- Funds can only flow to one of three destinations: executor (on AcceptWork), client (on CancelDeal or RefundAfterDeadline), platform commission address (split out of AcceptWork). No fourth destination exists.

J.3 What Admin Can and Cannot Do

Admin capability	How it works	What admin cannot do
Resolve a dispute in favour of executor	Calls AcceptWork on the per-deal contract using DEPLOYER_MNEMONIC	Cannot send funds to a third wallet — the contract has only two beneficiary addresses
Resolve a dispute in favour of client	Calls RefundAfterDeadline on the per-deal contract	Cannot mint funds — the contract only holds what the client originally locked
Cancel an unfunded order	Updates DB only; no on-chain action needed	Cannot cancel a Funded deal without a documented dispute or admin reason
Force a status change in the DB	Possible via PATCH endpoints	Cannot make the DB lie about on-chain truth — integrity checks will flag drift

J.4 Audit Posture

The smart-contract source code is committed in the contract repository, including the Tact source, the compiled FunC, the deploy artifacts (.boc, .pkg), and the ABI. The deployed code hash is published in the technical specification.

A third-party security audit is planned ahead of any major token-related launch (see §U). For Open Beta, the contract is in production with conservative deal sizes and a transparent dispute process; expanding to larger deals follows the audit.

K. Trust Score v2

K.1 Why a Portable Reputation Layer

In a market where the buyer and the seller have never met, reputation is the only continuous signal. Trust Score v2 is OTCHive's attempt to build that signal cleanly: a single number, 0.0 to 10.0, computed entirely from observable on-platform activity, with documented penalties for misbehaviour and natural decay over time.

K.2 Composition

Positive contributions (each with diminishing returns past a threshold):

- Average review rating, weighted by review count (sub-linear)
- Number of closed deals (cap kicks in around 50)
- Deal volume in TON (sub-$5 deals excluded to discourage volume farming)
- Account age — staggered milestones at 7, 30, 90 days
- Active Premium subscription (+0.3 flat while active)
- Verification level (L1 through L5)

Negative contributions:

- Open dispute — −1.0 per open dispute (temporary; restored on resolution)
- Lost dispute — −2.5 for the first, −1.0 each thereafter (permanent, slowly decays)
- Cancellation after hire — −0.3 each (an additional penalty applies past 3 cancellations)

K.3 Tiers and Benefits

Tier	Range	Effect
JUNIOR	0.0 – 4.99	Standard 10% commission, neutral feed placement
MID	5.0 – 6.99	Standard 10% commission
SENIOR	7.0 – 8.99	8% commission tier activates at Trust ≥ 8.0
ELITE	9.0 – 10.0	Top-of-feed bias; "Verified Pro" badge with closed_deals ≥ 1

K.4 Time Decay

Older reviews and older deals contribute less than recent ones. This keeps Trust Score current — a long-dormant veteran account naturally drifts back toward neutral, while consistent recent activity sustains a high score.

K.5 Anti-Abuse

- Self-deal detection (same wallet on both sides) contributes 0
- Wash-trading detection (pairs of accounts transacting only with each other) caps contribution
- Reviews from cancelled or refunded deals do not count

- Trust Score recompute is rate-limited to once per minute per user

L. Hive Points

L.1 What Hive Points Are

Hive Points are an internal retention-points system. They are an accounting number in the OTCHive database, used to gamify engagement (daily check-ins, deal completion, review streaks) and to provide an alternative path to Premium and Boost.

L.2 What Hive Points Are NOT

- Not a cryptocurrency
- Not GRAM
- Not HIVE
- Not withdrawable, not transferable, not tradeable
- Not guaranteed to convert into any future token — including HIVE (see §U)

This separation is deliberate, and it is repeated in user-facing copy. Promising convertibility before regulatory and technical certainty would be irresponsible to users.

L.3 Earn Rules

Action	Reward
Deal completion (each side, once per deal per role)	+20 Hive Points
5-star review received (once per review)	+30 Hive Points
Daily check-in (7-day rotating sequence)	5, 5, 5, 15, 15, 15, 30 Hive Points
Quests (verify_profile, first_bid, first_deal, first_referral, streak_7)	varies; award-once gate

L.4 Spend Rules

Spend target	Cost
Premium for 30 days (Hive Points path)	50,000 pts
Boost an order — Free tier	500 pts or $2
Boost an order — Premium tier	250 pts or $1.50
Cosmetic perks (profile feature, seasonal badges)	Varies — no economic value

Every Hive Points movement is logged in points_transactions. The sum of a user's points_transactions deltas equals their current balance; this is one of the cross-table integrity invariants in §Q.

M. Premium

M.1 Plans

Plan	Price	Notes
Monthly	$30 / month	
Annual	$300 / year	Two months free vs. monthly
Hive Points path	50,000 pts → 30 days	No external payment required

M.2 Benefits

- Commission drops to 7% (from 10% Free / 8% Trust-discounted)
- Unlimited orders, bids, templates, AI translations
- AI Translator in chat (Anthropic Haiku via backend endpoint)
- Rejection Insights — feedback on why a bid was not chosen
- Extended portfolio (up to 10 cases vs 3 Free)
- Feed boost priority
- VIP support response time
- Boost discount: 250 pts / $1.50 instead of 500 pts / $2
- +0.3 Trust Score bonus while active

M.3 Implementation

Two columns on the users table: is_premium and premium_until. Trust Score recompute, commission calculation, and feature gates all check is_premium && premium_until > NOW(). A daily cron job demotes expired Premium accounts.

N. Boost

Boost lifts a single order to the top of the exchange feed for 24 hours.

Tier	GRAM price	Hive Points price
Free	$2	500 pts
Premium	$1.50	250 pts

Implementation: orders.boosted_until timestamp. Feed sort favours boosted orders within their category. A cron task clears expired boosts. Cancelled, completed, and disputed orders cannot be boosted; an active boost cannot be re-boosted before expiry.

O. Referrals

O.1 Rates

Level	Share of platform commission
L1 — direct invitee (live)	15%
L2 — invitee of invitee (planned)	10%
L3 — invitee of invitee of invitee (planned)	7%

O.2 Economics

In Open Beta only L1 is active: the direct inviter receives 15% of the platform commission on a referred user's completed deals. L2 and L3 are designed but not yet enabled. If the full structure is later activated, total referral payout would be at most 32% of platform commission, never of the user's deal amount. Unit economics remain positive on every commission tier (10% / 8% / 7%) — referrals draw from the platform's share, not the user's payout.

O.3 Accrual Rules

- Accrued only on real completed escrow deals
- Self-referrals (same Telegram ID, same wallet, or same IP burst) excluded from accrual
- Stored in referrals.accrued_amount; withdrawable to the user's wallet once a threshold is reached
- Attribution is set on first sign-in via Telegram start_param; immutable thereafter; no retroactive attribution

P. Admin Command Center

P.1 Purpose

The Admin Command Center is the operations surface for the platform. It is hosted on the same domain as the Mini App, but accessed through a separate authenticated page. Two authentication paths exist: Telegram identity (admin's Telegram ID is on the ADMIN_IDS allowlist) or a shared admin token (a fallback path; logs the action as token-admin).

P.2 Tabs

Tab	Purpose
Overview	Live KPIs (users, deal volume, revenue) and 7-day funnel analytics
Deals	Orders, escrow, disputes, and per-order lifecycle trace
Users	User detail: ban / unban, set premium, adjust points (each action logged)
Treasury	Platform commission totals; pending withdrawals
Monitoring	Open alerts (stuck escrows, failed notifications) + system health (RPC, DB, uptime, kill switches)
Action Log	Admin audit history with full attribution (admin Telegram ID, action, target, JSON details)

P.3 Audit Discipline

Every privileged write — ban, unban, set premium, adjust points, resolve dispute, escrow cancel / refund — produces an admin_actions row in the same database transaction as the operation. The Action Log tab resolves admin IDs back to usernames; admins who sign in via the legacy token path appear as "token-admin" until they re-authenticate with their Telegram ID.

P.4 Kill Switches

Six environment-variable kill switches can disable classes of writes without redeploying: MAINTENANCE_MODE, DISABLE_NEW_ORDERS, DISABLE_HIRE, DISABLE_ESCROW, DISABLE_PREMIUM_PURCHASE, DISABLE_BOOST.

Q. Monitoring & Integrity Checks

Q.1 Two Layers

OTCHive runs cross-table integrity invariants both on-demand (via /api/admin/integrity-check, called from the Monitoring tab) and continuously (a cron task runs the lightweight subset hourly and writes failures into admin_alerts).

Q.2 Key Invariants

- Status drift — orders.status='done' must imply escrow_deals.status='completed', and vice versa
- Points balance — sum of points_transactions deltas per user equals users.hive_points (rebuildable from log)
- Reward idempotency — at most one +20 award per (order_id, role); guards against double-award regressions
- Escrow on-chain balance — the per-deal contract holds at least the recorded amount_ton until status moves to completed / refunded / cancelled
- Admin attribution — every PATCH /api/admin/user/* has a corresponding admin_actions row
- Wallet uniqueness — partial unique index on users.wallet_address (one wallet, one account)

Q.3 Operational Logging

Backend logs to stdout; Railway captures and displays them. Conventional prefixes — [API error], [StaffAlert], [Chat], [AdminLog], [Cron] — let operators grep for the relevant subsystem during incident response.

External monitoring (Sentry, UptimeRobot, log shipping) is a recommended next step ahead of any major scale push; it is not a blocker for Open Beta.

R. Dispute Resolution

R.1 Opening a Dispute

Either party can open a dispute from a Funded or Submitted deal via the Mini App. The user submits a reason and optional evidence (links, screenshots, message references). The backend:

- Creates a row in disputes with status=open
- Places the related order in a dispute hold (no AcceptWork allowed until resolved)
- Notifies the counterparty via the user-facing bot @OTCHiveBot
- Posts a structured alert in the internal Staff Group via @OTCHiveSupport_Bot (fire-and-forget; the alert never blocks the dispute creation if Telegram is unreachable)

Trust Score effect: −1.0 to every dispute participant while the dispute is open. Restored on resolution, except for the loser (see §K.5).

R.2 Admin Review

Admin opens the Dispute card in the Admin Command Center. It includes both parties' positions, the chat transcript for that order, the live on-chain escrow balance, and (when wired) AI Dispute Assist pre-analysis. Admin uses the per-order trace view to verify state consistency before resolving.

R.3 Resolving

PATCH /api/admin/disputes/:id/resolve with winner = client / executor / split.

- Atomic lock — status moves to resolving via an UPDATE WHERE status != 'resolving' RETURNING; prevents double-resolve
- On-chain action — AcceptWork (executor wins), RefundAfterDeadline (client wins), or a split-payout (two transactions)
- On confirmation — dispute status moves to resolved, order and escrow statuses finalize, both parties are notified
- Audit — the admin's action is logged in admin_actions with full attribution

R.4 What Cannot Happen by Design

The dispute resolution flow is constrained by the contract, not by trust in the admin:

- Admin cannot send funds to any wallet other than the two participants — only two beneficiary addresses exist on the per-deal contract
- Admin cannot resolve without on-chain confirmation — the backend never finalizes status until the transaction is observed
- Admin cannot create or destroy funds — the contract holds only what the client originally locked

R.5 The Decentralization Path

Centralized admin arbitration is appropriate for Open Beta because it is fast, transparent (audited via admin_actions), and constrained by the contract. A future model — community-elected arbiters, staked dispute committees, or DAO-style governance — is in the design backlog and is sketched in §T (Roadmap). It is not part of the Open Beta promise.

S. GRAM / TON Terminology

OTCHive uses three financial concepts that must not be confused — in code, in product copy, or in this document.

S.1 The Three Concepts

Concept	Role
GRAM	User-facing display label for monetary amounts in the OTCHive UI. Every "balance", "price", "bid amount", "payout" the user sees is labelled GRAM.
TON	The underlying network and native asset. Smart contract calls, on-chain confirmations, wallet bridge — all use TON in the technical sense.
Hive Points	Internal retention points. Not money. Not crypto. Not withdrawable. See §L.

S.2 The Phrasing Rule

In product copy, use "GRAM" by itself when the audience is the end user and only the amount matters. Use "TON" when the audience is a developer or a wallet-flow context and the network matters. Use the phrase "GRAM on TON" when both layers need to be named together (e.g., "escrow payments use GRAM on TON").

S.3 Why This Separation

"5 TON" reads as a crypto-trader unit; "GRAM 5" reads as platform money. Most OTCHive users are Telegram users first, crypto users second. Naming the user-facing unit GRAM lowers the cognitive cost of using the product without misrepresenting what is actually happening on-chain.

HIVE is a separate concept — a future ecosystem token (see §U). HIVE is not GRAM. HIVE is not Hive Points. The naming similarity between HIVE and Hive Points is unfortunate; the documents and the product copy take care to distinguish them.

T. Roadmap

The roadmap is organized around the current Open Beta phase and what each subsequent quarter unlocks. Dates are directional, not contractual.

T.1 Open Beta — Now

- Mini App, backend, and escrow contract live and used in production
- Trust Score v2, Hive Points, Premium, Boost, L1 referrals operational (L2/L3 planned)
- Admin Command Center with audit log, integrity checks, kill switches
- Telegram notifications with per-user toggles and Staff Group dispute alerts

T.2 Next 3 Months

- Migrate AI features from frontend to backend endpoints — Anti-Scam, Deal Coach, Dispute Assist
- Wire third-party monitoring (Sentry, UptimeRobot)
- Self-service withdrawal UI for referral earnings (currently admin-triggered)
- Localization beyond RU / EN — start with the largest TON ecosystem languages
- Skeleton loading states in Mini App for high-latency networks
- Sticky table headers and additional mobile polish

T.3 Next 6 Months

- Third-party smart-contract audit ahead of larger deal sizes
- Reputation staking experiments — executors stake (Hive Points or other) on their bids; sketched in §U
- Expanded category coverage and reverse-marketplace (gigs) UI
- Telegram Stars / external payment improvements

T.4 Next 12 Months and Beyond

- Decentralized dispute layer — community-elected arbiters or staked dispute committees, replacing centralized admin arbitration over time
- HIVE token consideration — only after audit, regulatory clarity, and sufficient marketplace scale (see §U for the constraints)
- Cross-chain bridges from ETH / SOL user bases into the TON-settled OTCHive ecosystem

U. Future HIVE Ecosystem Model (Planned)

> *This section describes a planned model. HIVE is not a live token. There is no public distribution today. There is no committed conversion from Hive Points or any other current asset. Everything below is subject to future legal, technical, market, and governance review.*

U.1 Why Consider a Token at All

A native token is justified only if it does work the existing system cannot do. The candidates under consideration:

- Discount-and-burn flywheel — paying for Premium / Boost in HIVE at a discount creates predictable demand; HIVE used for these is partially burned, reducing supply with platform activity
- Staking-based reputation — executors stake HIVE on a bid; bid winners earn back the stake; misbehaving executors lose stake
- Governance — token holders vote on dispute-arbiter selection, parameter changes, treasury allocation
- Buyback-from-revenue — a portion of platform commission converts to HIVE on the open market, tying platform success to token value

If we eventually launch HIVE, this is the kind of utility it would carry. None of this is a promise of timing, mechanism, or distribution.

U.2 What HIVE Is Not

- Not live — there is no contract, no listing, no market
- Not publicly distributed — no presale, no airdrop campaign, no "connect wallet to claim"
- Not a 1:1 conversion from Hive Points — that promise has never been made and will not be made
- Not a security under any current claim of OTCHive; classification will be determined by future legal review in the relevant jurisdictions

U.3 Constraints Before Any Launch

- Third-party smart-contract audit of the escrow and any token contracts
- Legal review in the launch jurisdictions
- Sufficient marketplace scale to make the token economically meaningful — vanity token launches against a small user base destroy more value than they create
- Robust anti-Sybil and anti-farming measures for any retroactive distribution component

U.4 If Retroactive Recognition Happens

In the event a retroactive distribution program is approved, the eligibility model under consideration would weigh real marketplace contribution — completed deals, deal volume, account age, Trust Score, reviews received, Premium purchases, Hive Points earned and spent, quality of referrals, and the activity of invited users — rather than a flat "connect and claim" mechanic.

- A snapshot date would not be announced in advance, to deter the obvious gaming patterns (mass-account creation, fake activity, wash trades, referral farms)
- Sybil-like patterns would be excluded — self-referrals, multi-account clusters, fake deals, sham reviews, automated activity
- Hive Points alone would not equal HIVE; Hive Points would be one signal among several

All of the above remains conditional on §U.3. Users should not make economic decisions on this section.

U.5 Internal Numbers Are Not Public

Internal drafts of supply, distribution percentages, vesting schedules, and buyback mechanics exist for planning purposes. They are intentionally not reproduced on the landing page or in this Whitepaper's public-facing posture. Internal numbers belong in internal documents; making them public before they are real creates an obligation we cannot yet honour responsibly.

V. Risks & Mitigations

V.1 Smart Contract Risk

Risk: a bug in the escrow contract could freeze funds or allow unauthorized withdrawal.

Mitigation: simple single-purpose per-deal contract with no upgrade path; conservative deal sizes during Open Beta; third-party audit ahead of large-deal expansion; emergency kill switches at the backend level prevent new deals while existing in-flight deals continue to settle on-chain.

V.2 Telegram Platform Risk

Risk: Telegram changes Mini App policies, restricts access, or applies category-level bans.

Mitigation: the underlying escrow is fully usable directly with wallets; the Mini App is a delivery surface, not the system of record. A web version of the marketplace can be brought up; the on-chain layer is unaffected by Telegram policy.

V.3 TON Network Risk

Risk: TON congestion, fork, or network-level incident affects user transactions.

Mitigation: the backend monitors RPC reachability and surfaces it in the Monitoring tab; idempotent retry logic on the frontend handles temporary failures; deals already on-chain are unaffected by upstream issues. A multi-chain expansion path exists if TON-specific risk becomes structural.

V.4 Operational / Custody Risk

Risk: deployer wallet key compromise; staff bot token leak; database compromise.

Mitigation: deployer wallet holds only platform commission, never deal principal; the per-deal contract beneficiaries are fixed at deploy and cannot be redirected; bot tokens are rotated under a documented runbook (one was rotated during this release); secrets are not in git, not in logs, not in client code; integrity checks surface drift between DB and on-chain truth.

V.5 Regulatory Risk

Risk: future jurisdictional rules on crypto marketplaces, escrow services, KYC, or token distribution affect OTCHive's operating model.

Mitigation: no public token live today, no securities-like promises, no investment language; the escrow contract is non-custodial by construction, which is a favourable starting posture in most jurisdictions; future token consideration is explicitly gated on legal review (see §U.3).

V.6 Dispute-Volume Risk

Risk: a spike in disputes — frivolous or coordinated — overwhelms the admin review queue.

Mitigation: open-dispute penalty in Trust Score (−1.0) discourages frivolous filing; Anti-Scam pre-checks (planned) reduce upstream entry of bad-faith deals; AI Dispute Assist (planned)

accelerates admin throughput; the staked-arbiter decentralization path is the long-term scaling answer.

W. Open Beta Readiness

W.1 What Is Live

- Telegram Mini App fully usable on iOS, Android, and Telegram desktop
- Backend with 127 endpoints, parameterized SQL, HMAC-verified authentication, idempotency keys on critical writes
- Escrow contract deployed on TON mainnet; happy path, cancel, refund, and dispute resolution all verified end-to-end
- Admin Command Center with full lifecycle visibility, audit log, integrity checks, and kill switches
- Telegram notifications — including chat-message bridge (privacy-by-default: fact only, no body) and Staff Group dispute alerts
- Trust Score v2, Hive Points, Premium, Boost, and L1 referrals (L2/L3 planned)
- Localization in RU and EN with unified phrasing and no overpromising claims

W.2 What Is Honestly Limited

- AI runs through a shared backend proxy today (Translation plus several assisted UI flows); dedicated productized endpoints for Anti-Scam, Deal Coach, and Dispute Assist — with separate quotas and monitoring — are planned
- Third-party smart-contract audit not yet conducted (planned before any token-related launch)
- Referral payouts are admin-triggered today (self-service UI on the near roadmap)
- External monitoring (Sentry, UptimeRobot) not yet wired
- The deployed mainnet escrow code hash allows the client to call AcceptWork before SubmitWork — a client self-harm path, not a theft vector, since funds still flow only to the executor and treasury per contract rules. A strict AcceptWork build that enforces the Submitted state is prepared for testnet validation and mainnet redeploy
- Localization is currently RU + EN only

W.3 Hard Stops Before Each Release

Internal release-readiness checklist:

- All six kill switches verified working in production
- /api/admin/integrity-check returns no failures
- A successful $1 mainnet end-to-end golden flow
- Latest backups taken; restore tested

X. Conclusion

OTCHive is built around a single thesis: the Web3 freelance market exists, the talent and the demand are on Telegram, and the missing piece is a trustless settlement layer. We built that layer on TON, wrapped it in a Telegram Mini App, added the reputation and retention systems that make a marketplace function, and put it into Open Beta with the operational disciplines (audit logs, integrity checks, kill switches, runbooks) that production-grade financial software requires.

We have intentionally not promised a public token, a guaranteed airdrop, or any conversion of Hive Points to any future asset. Those decisions are conditional on audit, regulatory clarity, and meaningful scale — and they will be communicated honestly when they happen, not used as bait beforehand.

The work ahead is concrete: finish migrating the AI features to the backend, complete a third-party audit, scale dispute resolution, expand localization, and grow the marketplace responsibly. We invite contributors, partners, and grant evaluators to look at the working product, the source code, and the operational posture, and to engage with us on the parts we are still building.

OTCHive — Trustless escrow infrastructure for Telegram-native Web3 work on TON.

https://otchive.com · @OTCHiveBot · @OTCHiveSupport_Bot

— end of document —