



CYBERHOOD

CYBERHOOD Genesis Economy

The \$CYBERHOOD token, the Genesis NFTs and the HoodHook on Uniswap v4

Litepaper v2.0 • Draft for review

October 2026

Not audited. Parameters are proposals and will change after testnet data, audit and legal review.

Contents

1. Summary

2. Design principles

3. Architecture

4. Why a Uniswap v4 hook

5. The HoodHook

5.1 Dynamic LP fee

5.2 Protocol fee and where it goes

5.3 Genesis holder discount

5.4 Swap XP (Hunter Score)

5.5 Referral rebate

5.6 Launch guard and liquidity lock

5.7 Price oracle for in-game prices

5.8 Guild liquidity vaults

6. Genesis NFT collection

7. \$CYBERHOOD token

8. Player Reward Pool

9. Season launchpad on Uniswap v4

10. Key flows

11. Security and governance

12. Roadmap

13. Parameter sheet

Appendix · HoodHook interface sketch

1. Summary

CYBERHOOD is a cyberpunk loot-hunting game that runs in the browser. Hunters meet in Neon City, a 3D co-op hub, then step through gates into side-scrolling contracts, a 100 v 100 siege, a 4–10 player raid, kart races and 1 v 1 duels. The game drops **loot, not tokens**: value moves between players through the marketplace, and rewards come from real platform revenue.

Version 2 moves the whole on-chain economy onto **one Uniswap v4 pool with one custom hook, the HoodHook**. Every swap of \$CYBERHOOD runs through the hook, which sets a dynamic fee, takes a small protocol fee, and routes it to the Player Reward Pool, buyback and burn, and the pool's own liquidity providers. The same hook gives Genesis NFT holders a fee discount, pays referrers, records swap activity for the Hunter Score and protects the launch from snipers.

1 Uniswap v4 pool · ETH / \$CYBERHOOD	0.30% Base LP fee, dynamic	0.20% HoodHook protocol fee
40% of hook fees to the Player Reward Pool	444 Genesis NFTs · 33 designs	0.01–0.05 ETH Genesis mint price by rarity

Rewards follow play activity, never token holdings. Nothing in this document is an offer of securities, a promise of profit or financial advice.

2. Design principles

- ▶ **Loot, not tokens.** Contracts, raids and sieges drop items. Tokens circulate between players through trades and sinks.
- ▶ **Revenue funds rewards.** The Player Reward Pool is paid from swap fees, marketplace fees, launchpad fees and sponsorship. No reward depends on minting new tokens.
- ▶ **Activity, not holding.** Weekly rewards are shared by Hunter Score (square-root weighted, capped per account). Holding \$CYBERHOOD earns nothing by itself.
- ▶ **No paid randomness.** No loot box is sold for money or tokens. Drop rates are public. Nothing random is granted for a swap.
- ▶ **Users can always exit.** The hook can change fees within hard-coded caps. It can never block a swap or freeze user liquidity.
- ▶ **Server-authoritative game.** Damage, drops and scores run on the game server. The chain holds ownership, trades and fee flows.

3. Architecture

The game server stays off-chain. Ownership, trading and fee routing live in a small set of contracts around a single Uniswap v4 pool. The launch chain is any EVM chain where the Uniswap v4 PoolManager is deployed (for example Unichain, Base, Arbitrum or Ethereum). Robinhood Chain remains an option if Uniswap v4 is live there at launch; this must be verified against the official deployment list before going to mainnet.

Contract	Standard	Role
CyberhoodToken	ERC-20	\$CYBERHOOD. Fixed supply of 1,000,000,000. Burnable.
HoodHook	Uniswap v4 hook	Attached to the ETH / \$CYBERHOOD pool. Dynamic fee, protocol fee, discounts, referrals, swap XP, launch guard, LP lock, TWAP oracle.
GenesisItems	ERC-721 + ERC-2981	444 Genesis items in 33 designs. 5% royalty.
GameItems	ERC-1155	Mech parts, armor, skins minted on demand from server-signed EIP-712 vouchers.
Marketplace	—	Fixed-price listings and 24 h auctions in ETH or \$CYBERHOOD. 5% fee.
RewardPool	—	Collects the player share of every fee stream and pays weekly Merkle claims.
GuildVault	—	Holds guild liquidity positions in the pool and reports time-weighted liquidity.
SeasonLaunch	Uniswap v4 hook	Variant of HoodHook used for partner token launches each season.
Treasury	Safe 3/5 + 48 h timelock	Owns protocol liquidity and the only keys that can tune parameters.

DATA FLOW

OFF-CHAIN (game server)	ON-CHAIN
- Matches, drops, Hunter Score	- PoolManager (Uniswap v4 singleton)
- Unminted inventory	- ETH / \$CYBERHOOD pool → HoodHook
- Weekly reward Merkle root →	- RewardPool.claim(proof)
- EIP-712 mint vouchers →	- GameItems.mint(voucher)
- Indexer reads hook events ←	- SwapXP, Referral, FeeRouted, Burn

4. Why a Uniswap v4 hook

Uniswap v4 keeps every pool inside one singleton contract, the PoolManager, and lets a pool attach a hook contract that runs before and after pool actions. That makes it possible to put game-economy rules directly in the trade path instead of wrapping the token in transfer taxes.

- ▶ **Hooks by address.** The callbacks a hook may use are encoded in the low 14 bits of its address, so the HoodHook is deployed with CREATE2 to a mined salt that matches its permissions.
- ▶ **Dynamic fees.** A pool created with the dynamic-fee flag lets the hook return a per-swap LP fee from beforeSwap (with the override flag set). Fees are in hundredths of a basis point: 3000 = 0.30%.
- ▶ **Return deltas.** With afterSwapReturnDelta, the hook can take a share of the swap output as its own fee and settle it with the PoolManager in the same transaction.
- ▶ **Flash accounting and ERC-6909.** Balances net out at the end of the unlock, and the hook can hold claims as ERC-6909 tokens inside the PoolManager instead of moving ERC-20s on every swap.
- ▶ **donate().** The hook can give fees straight to the liquidity providers who are in range, which deepens the pool over time.

- ▶ **Plain ERC-20.** \$CYBERHOOD stays a standard token with no transfer tax, so wallets, bridges and other DEXs work normally. Only trades through this pool pay the hook fee.

5. The HoodHook

The HoodHook is attached to a single pool: native ETH / \$CYBERHOOD, dynamic fee, tick spacing 60. The table lists the callbacks it enables and what each one does.

Callback	Enabled	Purpose
beforeInitialize	Yes	Accept only the ETH / \$CYBERHOOD key with the dynamic-fee flag, once.
afterInitialize	Yes	Record the launch block and start the price oracle.
beforeAddLiquidity	Yes	Launch window: only the Treasury may add liquidity. Register GuildVault positions.
afterAddLiquidity	Yes	Start the time-weighted liquidity clock for guild positions.
beforeRemoveLiquidity	Yes	Block removal of locked protocol positions until the unlock time.
beforeSwap	Yes	Compute the dynamic LP fee, apply the launch guard and the holder discount.
afterSwap	Yes	Take the protocol fee, route it, record swap XP and referral credit, update the oracle.
afterSwapReturnDelta	Yes	Lets afterSwap keep the protocol fee from the unspecified currency.
beforeDonate / afterDonate	No	Not used.
Other return-delta flags	No	Not used. Liquidity providers are never charged by the hook.

Who is the trader? Inside a hook, msg.sender is the router, not the user. The HoodHook only grants discounts, referral credit and XP when the swap comes through an allowlisted router that exposes msgSender() (the v4 periphery routers and the Universal Router do), or when the user signs the hookData. Swaps from any other router still work at the normal fee.

5.1 Dynamic LP fee

The LP fee is recalculated on every swap. It starts high on sells during launch, falls to the base fee, and rises briefly when the price moves sharply inside one block.

Component	Value	Rule
Base fee	0.30%	Normal trading, both directions.
Launch sell fee	8% → 0.30%	Falls linearly over the first 72 hours after liquidity opens.
Launch buy fee	1% → 0.30%	Falls linearly over the first 24 hours.
Volatility surcharge	+0.20%	When the pool tick has moved more than 100 ticks (about 1%) in the same block. Makes sandwich attacks costly.
Circuit breaker	+2% on sells	For 30 minutes when the 1-hour TWAP falls more than 25%. Never blocks a sell.
Hard cap	10%	MAX_LP_FEE is a constant. No parameter or owner can exceed it.

5.2 Protocol fee and where it goes

On top of the LP fee, the hook keeps 0.20% of each swap's output through `afterSwapReturnDelta`. The fee is held as ERC-6909 claims and swept once per epoch (about one hour) to keep each swap cheap.

Destination	Share	What happens
Player Reward Pool	40%	Paid out weekly by Hunter Score (section 8).
Buyback & burn	30%	ETH share buys \$CYBERHOOD in the pool and burns it; the token share is burned directly.
LP boost	20%	Given to in-range liquidity providers with <code>donate()</code> .
Operations	10%	Servers, audits and bounties, held by the Treasury.

The protocol fee is capped at 0.50% in code. Governance can lower it or change the split between the four destinations, but the Player Reward Pool share cannot go below 30%.

5.3 Genesis holder discount

A trader who owns a Genesis NFT pays less protocol fee. The router passes the `tokenId` in `hookData` and the hook checks `ownerOf` against the verified trader. Only the best discount applies; discounts never reduce the LP fee, so liquidity providers are not affected.

Holding	Protocol fee discount	Effective protocol fee
Verified Hunter Pass (no NFT)	10%	0.18%
Common Genesis	10%	0.18%
Rare Genesis	20%	0.16%
Epic Genesis	30%	0.14%
Legendary Genesis	40%	0.12%
Mythic Genesis	50%	0.10%

5.4 Swap XP (Hunter Score)

Swaps from a verified Hunter Pass wallet emit a `SwapXP` event. The game server indexes it and adds points to the weekly Hunter Score. Gameplay stays the main source of score; swap XP is a small, capped bonus.

- ▶ 1 Hunter Score point for each 0.01 ETH of swap volume.
- ▶ Counted volume is capped at 0.3 ETH per account per day and swap points at 200 per week, a tenth of the 2,000-point weekly cap.
- ▶ A buy and sell that reverse each other within 10 minutes count once (wash-trade filter).
- ▶ XP is never paid out directly and never triggers a random reward. It only feeds the activity share of the weekly pool.

5.5 Referral rebate

A new Hunter can join with a referral code. For 180 days, 10% of the protocol fee from that Hunter's swaps is credited to the referrer as claimable ERC-6909 balance. The rebate comes from the protocol fee, never from the new player's trade.

5.6 Launch guard and liquidity lock

- ▶ **Launch window.** For the first 30 blocks after liquidity opens, a swap may buy at most 0.1% of supply and each verified trader can swap once per block.
- ▶ **Router allowlist.** During the window only allowlisted routers can trade; afterwards the pool is open to every router.
- ▶ **Protocol liquidity lock.** The Treasury seeds the pool with 10% of supply. Its positions (identified by the PositionManager salt) cannot be removed for 12 months.
- ▶ **No blacklist.** The hook has no function to block an address, freeze a balance or stop swaps.

5.7 Price oracle for in-game prices

The hook keeps a ring buffer of price observations and exposes a time-weighted average price. In-game services such as Mech repair are quoted in US dollars; the GameBank converts them to \$CYBERHOOD using the 30-minute TWAP, so one large trade cannot change what players pay.

5.8 Guild liquidity vaults

Guilds can provide liquidity through a GuildVault. The hook records the liquidity and how long it stays. Each guild earns territory influence equal to the square root of its time-weighted liquidity, and that bonus is capped at +10% of a territory-war score, so liquidity helps a guild but cannot replace winning battles.

6. Genesis NFT collection

444 pixel-art weapons, armor and relics in 33 designs, minted once for Season 0 as ERC-721 tokens with a 5% ERC-2981 royalty. Rarer designs have fewer editions and a higher mint price.

Rarity	Designs	Editions each	Items	Mint price
Mythic	6	4	24	0.044–0.050 ETH
Legendary	10	8	80	0.032–0.040 ETH
Epic	10	14	140	0.022–0.030 ETH
Rare	6	25	150	0.015–0.020 ETH
Common	1	50	50	0.010 ETH
Total	33	—	444	—

- ▶ **Utility:** the HoodHook fee discount, a cosmetic version of the item in game, and the Founding Hunter badge. Genesis NFTs give no yield and no share of revenue.
- ▶ **Mint:** Hunter Pass holders get a 24-hour allowlist window, then a public window. Maximum 3 per wallet.
- ▶ **Proceeds (proposal):** 50% paired with \$CYBERHOOD as pool liquidity, 30% to the Season 0 Player Reward Pool, 20% to development.
- ▶ **Royalty split:** 2.5% Player Reward Pool, 2.5% studio.

7. \$CYBERHOOD token

Fixed supply of 1,000,000,000 \$CYBERHOOD. No mint function after deployment. The allocation below is the draft from the game design document and must be reviewed by a tokenomics specialist.

Allocation	Share	Unlock
Community & players	40%	Season 0 airdrop 8% (50% at launch, 50% over 6 months) · season reward fund 22% (decreasing over 4 years) · ecosystem & launchpad 10%
Team	15%	12-month cliff, then 36 months linear
Investors	15%	6–12-month cliff, then 24 months linear
Liquidity	10%	Seeded into the v4 pool, locked 12 months by the HoodHook
Treasury	15%	Safe multisig with timelock
Marketing	5%	24 months linear

Sinks

- ▶ Mech repair, the largest sink: paid in \$CYBERHOOD at the TWAP price; 50% burned, 50% to the Player Reward Pool.
- ▶ Marketplace and auction fee of 5%: 2% Player Reward Pool, 2% burned, 1% to the guild that holds the territory.
- ▶ Crafting, guild creation (100, burned), siege registration (50, burned), name changes and skins.

- ▶ The HoodHook buyback & burn (30% of the protocol fee).

8. Player Reward Pool

All player shares of fees land in the RewardPool contract, a public on-chain address. Each week the game server computes Hunter Scores, publishes a Merkle root and players claim. The pool is split as in the game design document.

Weekly share	Portion	Who receives it
Activity fund	50%	Every Hunter, by the square root of weekly Hunter Score, capped per account
Challenge fund	20%	Top 500 of the weekly solo challenge (normalised gear, pure skill)
PvP fund	15%	Arena, contested maps and siege
Guild fund	15%	Top 50 guilds, shared by contribution

Square-root weighting means a Hunter who plays four times as much earns twice as much, not four times. Together with the 2,000-point weekly cap and account verification, this keeps bots and whales from draining the pool.

Revenue stream	Share to the Player Reward Pool
HoodHook protocol fee	40%
Marketplace and auction fee (5%)	40% of the fee (2% of the sale)
Launchpad listing fee and raise share	50%
Sponsorship: Hub billboards, branded stages, events	40%
Cosmetics, name changes, effects	30%
Genesis royalties (5%)	50% of the royalty

9. Season launchpad on Uniswap v4

Partner projects launch their tokens each season in their own v4 pool with the SeasonLaunch hook, a variant of the HoodHook.

- ▶ **Tiered allowlist.** For the first 2 hours only wallets in the season Merkle tree can buy: Bronze, Silver, Gold and Diamond key holders plus guild tickets, each with a per-wallet cap.
- ▶ **Players first.** 30% of every public sale is reserved for CYBERHOOD players through these tiers.
- ▶ **Fair price discovery.** The launch fee falls from 5% to 0.30% over 6 hours, which discourages snipers without a fixed price.
- ▶ **Airdrop condition.** Each project airdrops 1–3% of its supply to Hunters who earned keys that season.
- ▶ **Refunds.** If a sale is cancelled or fails a technical check, contributions held in escrow are refunded in full.

10. Key flows

A swap through the HoodHook

1. The Hunter swaps ETH for \$CYBERHOOD in the CYBERHOOD app. The router adds hookData: Genesis tokenId (optional) and referral code (optional).
2. beforeSwap reads the verified trader from the router, applies the launch guard, and returns the dynamic LP fee with the override flag.
3. The PoolManager executes the swap at that fee; liquidity providers earn the LP fee as usual.
4. afterSwap takes the protocol fee (after any holder discount) from the output, credits the referrer, emits SwapXP and updates the oracle.
5. Once per epoch, anyone can call sweep(): fees move to the RewardPool, the burn, the LP donate and operations.

Minting a game item

1. A drop lands in the Hunter's off-chain inventory.
2. To mint it, the Hunter asks the server for an EIP-712 voucher. The server locks the off-chain copy so it cannot be duplicated.
3. The Hunter calls GamelItems.mint(voucher). Bringing an item back into the game burns it or locks it in escrow.

Weekly rewards

1. At the end of each week the server computes Hunter Scores with the square-root rule and caps.
2. It publishes the Merkle root to the RewardPool through the Treasury timelock.
3. Hunters claim at any time; unclaimed rewards roll into the next season after 90 days.

11. Security and governance

- ▶ **Immutable core.** The HoodHook logic is not upgradeable. Parameters can only move inside ranges written as constants: LP fee at most 10%, protocol fee at most 0.50%, Player Reward Pool share at least 30%.
- ▶ **Timelock.** Parameter changes go through a 3-of-5 Safe and a 48-hour timelock, so traders see every change before it applies.
- ▶ **Pause scope.** The emergency pause only turns off discounts, XP and referral accounting. Swaps and liquidity exits keep working.
- ▶ **Delta accounting.** Every callback must leave the hook's balance with the PoolManager settled. Invariant and fuzz tests cover every return-delta path.
- ▶ **Gas budget.** Each callback is kept small (no loops over users, no external calls except ownerOf and msgSender) so swaps stay affordable.
- ▶ **Audits.** Two independent audits of the hook and the RewardPool before mainnet, plus a public bug bounty.
- ▶ **Oracle safety.** In-game prices use a 30-minute TWAP; the circuit breaker uses a 1-hour TWAP.
- ▶ **Legal.** No profit promises. Rewards follow activity, not holdings. The reward pool, rentals and launchpad need legal review in each target market before launch.

12. Roadmap

Phase	Target	Deliverables
Season 0 · Early access	Now	Browser demo, Hunter Pass, Genesis preview, Season 0 points
Hook testnet	Q4 2026	HoodHook and RewardPool on a v4 testnet, public fee dashboard, bug bounty
Audit	Q1 2027	Two audits, invariant test suite, parameter review with testnet data
Genesis mint	Q1 2027	444 Genesis NFTs, allowlist then public
Token & pool launch	Q2 2027	\$CYBERHOOD, ETH / \$CYBERHOOD pool with HoodHook, Season 0 airdrop
Season 1	Q2 2027	Weekly reward claims, guild vaults, first SeasonLaunch partner

13. Parameter sheet

Parameter	Value	Range allowed by code
Pool	Native ETH / \$CYBERHOOD, dynamic fee, tick spacing 60	Fixed
Base LP fee	3000 (0.30%)	500 – 10,000
Launch sell fee / decay	80,000 (8%) over 72 h	≤ 100,000
Launch buy fee / decay	10,000 (1%) over 24 h	≤ 100,000
Volatility surcharge	+2000 when > 100 ticks per block	0 – 10,000
Circuit breaker	+20,000 on sells for 30 min after –25% TWAP 1 h	0 – 30,000
Protocol fee	2000 (0.20%)	0 – 5000
Protocol fee split	40 / 30 / 20 / 10	Player share ≥ 30%
Holder discount	10 – 50% by rarity	0 – 50%
Swap XP	1 point per 0.01 ETH · 0.3 ETH per day · 200 per week	Off-chain
Referral rebate	10% of protocol fee, 180 days	0 – 20%
Launch guard	30 blocks, 0.1% of supply per swap	Fixed at deploy
Protocol LP lock	12 months	Fixed at deploy
Governance	Safe 3/5 + 48 h timelock	—

Fees in this sheet are in Uniswap units of hundredths of a basis point (1,000,000 = 100%).

Appendix · HoodHook interface sketch

SOLIDITY (SKETCH, NOT AUDITED)

```
contract HoodHook is BaseHook {
    uint24 public constant MAX_LP_FEE = 100_000;           // 10%
    uint24 public constant MAX_PROTOCOL_FEE = 5_000;       // 0.50%

    function getHookPermissions() public pure override
        returns (Hooks.Permissions memory p)
    {
        p.beforeInitialize = true;           p.afterInitialize = true;
        p.beforeAddLiquidity = true;         p.afterAddLiquidity = true;
        p.beforeRemoveLiquidity = true;
        p.beforeSwap = true;                 p.afterSwap = true;
        p.afterSwapReturnDelta = true;
    }

    function _beforeSwap(address router, PoolKey calldata key,
        SwapParams calldata params, bytes calldata hookData)
        internal override returns (bytes4, BeforeSwapDelta, uint24)
    {
        // trader = allowlisted router's msgSender() or a signed hookData
        address trader = _verifiedTrader(router, hookData);
        _launchGuard(trader, params);
        // base + launch decay + surcharges, never above MAX_LP_FEE
        uint24 fee = _dynamicFee(key, params.zeroForOne);
        return (this.beforeSwap.selector, BeforeSwapDeltaLibrary.ZERO_DELTA,
            fee | LPFeeLibrary.OVERRIDE_FEE_FLAG);
    }

    function _afterSwap(address router, PoolKey calldata key,
        SwapParams calldata params, BalanceDelta delta,
        bytes calldata hookData)
        internal override returns (bytes4, int128)
    {
        (Currency feeCcy, uint256 out) = _unspecified(key, params, delta);
        // protocol fee after the holder discount, kept as ERC-6909 claims
        uint256 fee = out * _protocolFee(router, hookData) / 1_000_000;
        poolManager.mint(address(this), feeCcy.toId(), fee);
        _creditReferrer(router, hookData, feeCcy, fee);
        _emitSwapXP(router, hookData, key, delta);
        _observe(key);
        return (this.afterSwap.selector, int128(int256(fee)));
    }
}
```