Autheo Node Sale Platform - Queries

Contract Design & Implementation

- **What was the rationale behind separating functionality between AutheoCentralStore and tier-specific contracts?**
  The CentralStore smart contract manages user details that are shared across all tier contracts. This approach helps eliminate redundant logic and reduces the number of transactions required from users.

- **How did you determine the lock period of 365 days for token transfers?**
  We've had discussions before the start of the development where we explained how other Node Sale Platforms are also giving a 12 months of lock period. After multiple discussions, we came to the conclusion that we'll also use a 12 months of lock period.

Access Control & Roles

- **Who are the initial superAdmin and admin addresses at deployment?**
  The smart contract defines two distinct roles: superAdmin and admin. Only one superAdmin can exist at a time, and this role holds the highest level of authority. The superAdmin is responsible for adding or removing admins, transferring their role to another account, and upgrading the contract (as the AutheoCentralStore contract is upgradeable).

  On the other hand, multiple admins can be assigned within the contract. Admins are granted permissions to perform various operational tasks, including:
  a) Updating referral status (inactive referrals cannot be used during purchases)
  b) Blacklisting users
  c) Managing user KYC updates
  d) Adding internal accounts (accounts allowed to purchase more nodes than the public cap)
  e) Removing internal accounts
  f) Adjusting the node cap for internal accounts
  g) Can update the referrals and discount percentage

  Initially, the smart contracts will be deployed from our wallet, which is an Externally Owned Account (EOA). Later, the ownership will be transferred to the Autheo team's wallets. The Autheo team can assign ownership to any type of wallet they choose to use

in the future, including contract-based wallets or multi-sig wallets.. Later, Super admin can transfer its ownership to any other wallet address and can create and remove admin users. Please note that the private key of the super admin should not be lost.

- **Can the superAdmin or admin be externally owned accounts (EOAs), multisigs, or smart contracts?**
  Initially, the smart contracts will be deployed from our wallet, which is an Externally Owned Account (EOA). Later, the ownership will be transferred to the Autheo team's wallets. The Autheo team can assign ownership to any type of wallet they choose to use in the future, including contract-based wallets or multi-sig wallets.

- **What operational scenarios are expected for access role transfers (e.g., transferSuperAdmin)?**
  As mentioned above, the superAdmin role holds the highest level of authority within the smart contract. If there is ever a need to change the superAdmin in the future, the transferSuperAdmin function can be used to securely transfer this role to another account. Please note that the private key of super admin should never be lost.

- **Is the admin expected to be rotated periodically or locked down?**
  The smart contract supports multiple admins because several operations are restricted to admin-only access. This design ensures that critical tasks are not delayed—if one admin is occupied with a time-consuming transaction, other admins can continue performing necessary operations without interruption. This redundancy helps maintain the platform's efficiency and responsiveness. There is no need for rotation or lock down.

- **How are the admin keys managed? Is there a multisig wallet or timelock mechanism for sensitive operations?**
  We use AWS Key Management Service (KMS) for secure management of admin access on the platform, enabling them to safely perform transactions.

- **Why must users complete KYC before generating referral codes? Is this enforced off-chain as well?**
  As per the Autheo team's requirement, every user who generates referral codes or purchases nodes must complete KYC verification. This process is handled off-chain using Blockpass.

- **Can referral codes be reused across tiers, or are they tier-specific?**
  Yes, referral codes can be used across different tiers, which is why the referral functionality is implemented within the AutheoCentralStore smart contract.

- **How is referral fraud prevented (e.g., self-referrals, bots)?**

Self-referrals are allowed within the system; however, to prevent fraud, we have implemented checks that ensure only KYC-verified users can generate referral codes.

- **Are commissions stored and claimable, or only computed and settled instantly?**
  No commissions are calculated and settled instantly within the same transaction during node purchases.

- **What is the economic rationale behind the referral system's commission percentages?**
  The commission % is configurable by the admin**.** The idea behind implementing referral module is to motivate people to get more people on the platform. The best way to do it is to incentivize it.

- **Have you performed any legal reviews of the referral system to ensure compliance?**
  No, there was no notification or requirement of referral compliance or legal review. It is just a commission based referral module. Where, whose referral code has been used will receive a certain percentage of the total amount. If there is any need for compliance, you can share it with us.

User Management & KYC

- **How is KYC information verified and who performs this verification?**
  KYC information is verified through Blockpass.

- **Where and how is KYC data stored? Is any sensitive data kept on-chain?**
  Since we are using a third party KYC platform. We are not storing any personal information in our database. These KYC platforms follow global compliances. Here is the link if you need more information. https://www.blockpass.org/node-sale-kyc/

- **Are KYC and blacklist mappings synchronized with any off-chain systems?**
  Yes, once Blockpass approves the KYC, an admin will update the user's KYC status on the blockchain via the smart contract.

- **What happens if a user is blacklisted after minting a node license?**
  This is an exceptional case, and once a node is purchased, it cannot be reversed or taken back. Therefore, blacklisting is only available before purchasing the Node.

- **How does the blacklisting mechanism work, and what criteria are used for blacklisting?**

This is an optional feature that allows the platform to blacklist a user if any suspicious or fraudulent activity is detected. Admin have access to blacklist the user.

- **What regulatory frameworks have you considered in designing these contracts?**
  There are no regulatory frameworks considered.

Token Support & Price Feeds

- **What specific Chainlink price feeds are you using, and have you implemented fallback mechanisms?**
  We are using Chainlink's decentralized price feeds for ETH/USD, which provide reliable and tamper-resistant data. No fallback mechanism implemented.

- **How does the calculation for converting ETH to USD price work in practice?**
  The conversion from ETH to USD is based on the real-time price data provided by the Chainlink price feed. This price feed aggregates data from multiple trusted sources to ensure accuracy and reliability. When a transaction or query requires the ETH/USD price, the smart contract fetches the latest price from the Chainlink oracle, which is updated at regular intervals. This value is then used to perform any necessary calculations, ensuring that the conversion is based on the most up-to-date market information.

- **How often is the Chainlink ETH/USD price feed expected to update? Any fallback mechanism?**
  The Chainlink ETH/USD price feed is updated at regular intervals, typically every few minutes, depending on market volatility.

- **Are there any protections if the Chainlink price feed returns stale or manipulated values?**
  No not yet. The idea of using chainlink came in because it is a decentralized oracle, where manipulation is difficult.

- **Can tokens other than ETH, USDC, and USDT be supported in the future?**
  No plans as of now.

Financial & Economic Aspects

- **What is the expected volume of transactions and total value that will flow through these contracts?**

Since we will be selling only 400 nodes as NFTs, the platform is expected to experience a low volume of transactions.

- **How was the pricing model (100 USD per node license) determined?**
  Discussed before the start of the development with the Autheo team.

Payments & Minting Logic

- **What's the rationale behind a 12-month lock period on token transfers? Is this enforced at the smart contract level?**
  Yes, we have implemented a 12-month lock on direct NFT transfers to prevent users from reselling their NFTs immediately after purchase. This approach has already been discussed and agreed upon with the Autheo team.

- **Are users refunded automatically for overpayments (e.g., excess ETH)?**
  Yes

- **Is the minting cap global or per wallet? Is there a tiered total supply?**
  The minting cap is enforced per wallet across all four sale tiers, and each tier will have its own predefined total supply (i.e., number of nodes available for sale). All of these configurations were provided by the Autheo team during the smart contract's mainnet deployment. Once deployed, it cannot be changed.

- **Are refunds handled in case of partial minting failures or price slippage?**
  So there is no partial minting available. Public users are going to buy only 1 Node Token. However, there will be an option for admin to configure wallet addresses who can purchase more than 1 Node Token

Node Licenses (ERC721)

- **Why was ERC721 chosen over ERC1155 for node licenses?**
  In Phase 2, we will verify the ownership of each node and also associate additional metadata with each NFT's URI.

- **Is metadata mutable? Can the base URI be changed post-deployment?**
  Yes, the smart contract includes functionality to update the base URI in the future.

- **Can nodes be revoked or burned by admins for any reason?**

No, the smart contract does not include any functionality to burn the nodes.

- **Is there a plan for secondary sales after the 12-month lock?**
  No

- **Can you clarify how the tier system works? Are there multiple licenses with different properties?**
  The primary reason for selling nodes in tiers is to implement a tier-based pricing strategy—nodes in the initial tiers are priced lower than those in subsequent tiers. This approach is commonly used across most node sale platforms.

Upgradeable Design

- **What specific upgrade logic do you anticipate using in the future?**
  Since the core functionality is centralized within the AutheoCentralStore smart contract, any future changes or additions to the logic can be implemented more efficiently.

- **Are there any protections in place to ensure a secure upgrade process (e.g., timelocks, community oversight)?**
  No superAdmin of the platform can upgrade the smart contract any time.

Architecture and Off-Chain Dependencies

- **Which parts of the system rely on off-chain data (e.g., user KYC statuses, referral analytics)?**
  Only the KYC data relies on the off-chain response from Blockpass.

- **How is synchronization between the off-chain and on-chain components maintained?**
  We have set up a Node.js application integrated with AWS KMS. Once the Blockpass response is received and triggers our APIs, the admin account executes a smart contract function on the blockchain to update the user's KYC status.

- **Are there any APIs or external systems that interact with the contracts?**
  NO

- **Are there any dependencies beyond Chainlink price feeds and OpenZeppelin?**
  NO

- **What measures are in place if Chainlink oracles become unavailable?**

Chainlink price feed contracts are highly reliable, and currently, we do not have a backup solution in place.

- **Have you analyzed the security history of your external dependencies?**
  Yes, we have reviewed the security track record of OpenZeppelin contracts and found them to be reliable and secure.

Testing and Edge Cases

- **What testing has been performed (unit tests, integration tests, formal verification)?**
  We have conducted both unit testing and live testing on the testnet, and have also ensured comprehensive test coverage using Solidity coverage tools.

- **What unit/integration tests have been conducted for these contracts?**
  Will share the test cases file.

- **Have the contracts been tested against known smart contract vulnerabilities (e.g., reentrancy, overflow, frontrunning)?**
  Yes

- **How are upgrade simulations tested before being pushed live?**
  The smart contract has not been deployed yet. However, we have thoroughly tested the upgradeability functionality through both unit tests and live testing on the testnet.

Security Practices

- **Have these contracts undergone any previous security reviews or audits?**
  Zeeve has done an internal audit.

- **If audited, by whom and when?**
  By Zeeve Team

- **Are there bug bounty programs or community audits planned?**
  No

- **Are time-based actions (e.g., locking) reliant on block timestamps? If so, is there any mitigation for timestamp manipulation?**

Yes, it depends on the timestamp. While miners can manipulate the timestamp by 2–3 seconds, this minor variation does not affect our logic.

- **What security measures have you implemented beyond standard OpenZeppelin libraries?**

Operational Concerns

- **What is your process for key management and admin role transitions?**

- **Do you have a documented incident response plan if vulnerabilities are found?**
  The platform is in development. We used to prepare this documentation once we finish the development

- **How will you handle user support for failed transactions or other issues?**

- **What is your plan for communicating contract upgrades to users?**

Analytics & Monitoring

- **Are mints and referral commissions tracked off-chain as well?**
  Yes, we are using a blockchain indexer to index on-chain data and generate reports and analytics for our users.
- **Are there emergency shutdown functions in place?**
  No, but we can incorporate pausable functionality into the smart contract if needed.