KAON DOCS - OUTLINE [DRAFT]

#### **QUICK LINKS**

* **Network Details & Parameters**: Overview of MainNet & TestNet parameters, Chain IDs, RPC Endpoints, contract size, gas policies, and block explorers.
* **Quick Setup**: Initial setup commands, configuration templates, and network connections.
* **Important Addresses**: List of deployed contracts, core system contracts, and treasury address.

### 

### **LEARN**

#### **Overview [Beginner ⭐️]**

* **Visual Introduction**: High-level architecture and simple flow of how Kaon works. [Link to visual diagram or video tutorial for added clarity].
* **Key Benefits**:
  + Leveraging an L1 chain with an open architecture: Enhanced transparency and security as every transaction is verifiable on the blockchain.
  + Platform Flexibility: Users can access any trading platform on an EVM-compatible BTC side chain.
  + Simplified Transaction Process: Direct purchase of BTC using wallets like MetaMask, and easy transfer to Bitcoin wallets.
  + Integration with Bitcoin: Publicly spread nodes and open validator participation maintain Kaon's connection to the Bitcoin ecosystem.
  + Full EVM Support: Enables the use of Solidity smart contracts within the BTC ecosystem.
  + UTXO Chain with Merge Compilation: Allows for cross-verification and enhanced security by enabling nodes to operate simultaneously in both Bitcoin and Kaon chains.
  + **mirrorBTC**: Ensures mirrored token interactions are genuinely connected to the original BTC, addressing trust and security issues associated with wrapped BTC.
  + **UTXO-EVM Combination**: Seamless interaction with inscriptions, ordinals, BRC20, and runes from smart contracts.
  + **RLP Transaction Support**: Enables seamless integration with Metamask and other Ethereum-based tools.
  + **1e-18 Denomination**: Offers 10 times less expensive gas fees and complete EVM support.
  + **RPC Ethereum Full Support**: Ensures compatibility with Ethereum tools like Blockscout.
  + **Cross-Contracts**: Facilitates smart contract interactions across different chains.
  + **Observers**: Enhances user experience and interaction complexity.
  + **Control over Pools by the Chain**: Enhances security and trust by ensuring the non-custodial nature of liquidity pools.
  + **Merge of Nodes**: Eliminates the need for oracles in cross-chain interactions.
  + **Same Private Keys as in BTC**: Leverages account abstraction with BTC, simplifying transaction signing.
* **Key Terms**: Definitions of important terms, such as $KAON, Mirrored BTC, Wrapped BTC, Cross-chain Validators, Voucher, and **Inventory**: A special type of transport container commonly used in classic blockchains to transfer data about blocks and transactions from peer to peer.

#### **Why Kaon? [Beginner ⭐️]**

* **Market Context**: Discusses bridge security crisis, such as the 2022 Ronin Bridge hack, systemic security risks, and highlights Kaon’s “Bridge-less” approach.
* **Comparison With Existing Solutions**: Analysis of bridge-based solutions (prone to hacks and exploits), multisig/rollup limitations (scalability and complexity), and Kaon’s innovations (bridge-less and oracle-less design for enhanced security and efficiency).
* **Kaon’s Innovations**: Details on renegotiation agreements, eliminating intermediaries, and leveraging advanced OP\_CODES.

#### **Core Concepts [Intermediate ⭐️⭐️]**

* **UTXO-EVM Integration**: Integration model, transaction processing, and state management.
* **Key Innovations**: Overview of mirrorBTC, Cascade Signature System, Oracle-less operations, and Cross-Consensus Messaging.
* **Trustless Design**:
  + **Ownership and Control**: Users retain full control over their assets, as Kaon eliminates reliance on third parties or intermediaries.
  + **Oracle-less Operations**: Kaon achieves trustless operations without depending on external oracles, enhancing security and transparency.
  + **Permissionless Interaction**: Users can interact with the Kaon network freely, without requiring permissions or approvals from centralized authorities.
* **Security**: Incident handling, potential threats, and mitigation strategies. Covers incident processing, arbitration, and validator slashing mechanisms.
* **Consensus Layer**: Basic explanation of dPoS and Galaxy consensus mechanisms, roles of Master Nodes, Validators, and Epoch structure.
* **EVM Integration**: How Kaon integrates the EVM within its UTXO model for smart contract functionality and compatibility with Ethereum tools.
* **Bitcoin and Alternative Consensus Interactions**:
  + **Peg Mechanisms**: Explanation of Peg-In and Peg-Out processes, including instant exits and controllable locks.
  + **Advanced Uses**: Discussing applications such as proof of ownership, BitVM2 integration, and Automated Market Makers (AMMs).
  + **Handling Different Amounts**: Explanation of Front and Reimburse strategies for managing varying transaction amounts.
  + **Financial Concepts**: Elaboration on concepts like Surplus and Shortage, and Slippage Management in the context of cross-chain interactions.

For more advanced details, see the [Technical Deep Dives](https://notebooklm.google.com/notebook/1063993f-b187-4b51-ace0-48810009dfc6#technical-deep-dives).

### 

### **TUTORIALS**

#### **Quickstart [Beginner ⭐️]**

* **5-Minute Setup**: Environment configuration, first network connection, and transaction. [More data needed].
* **Hello Kaon**: Simple contract deployment and interaction. [More data needed].
* **Hello Bitcoin**: Creating a basic Bitcoin vault.

#### **Basic Development [Beginner ⭐️]**

* **Environment Setup**: Development tools and testing environment setup.
* **First Smart Contract**: Deployment process and testing.

#### **Cross-Chain Operations [Intermediate ⭐️⭐️]**

* **BTC Integration**: Locking, mirroring, withdrawal, and cross-chain transactions. Covers locking BTC on the Bitcoin network, mirroring those tokens on Kaon, and withdrawing them back to Bitcoin.

#### **Advanced Development [Advanced ⭐️⭐️⭐️]**

* **Complex Implementations**: Advanced security integration, creating Bitcoin Vaults, and Zero-Knowledge Proofs.
* **BTC Locking and Mirroring Process**: Steps involved in locking BTC and creating mirrored tokens on the Kaon network.
* **Gasless Operating Process**: How users can conduct transactions without holding the native gas token for fees.
* **Operator Set Changes Process**: Guidelines on changing operator sets and validator participation.

#### **Detailed Tutorials [All Levels]**

* **How to Lend BTC**: Lending smart contract setup, managing collateral.
* **Setting Up a Node**: Hardware and installation guide.

### **NETWORK & TOOLS**

#### **Technical Architecture**

* **System Design**: Architecture overview, component interaction, data flow.
* **Diagrams - Key Processes**: Visual representations of Kaon's data flow for various processes, such as Preparation for Launch, Oracle-less Cross-chain Message Processing, Bridge-less ERC Cross-chain Transfers, BTC Locking and Mirroring, BTC Withdrawal, Mirrored BTC and EVM Interactions, Metamask and other Offchain EVM Wallets Support, Mirrored BTC Transfer to EVM Chain, Restore Mirror BTC from Wrap, Gasless Operating, Operator Set Changes, and Slashing Incidence Process.
* **Consensus Mechanisms**: Delegated Proof of Stake, Byzantine Fault Tolerance, Galaxy Consensus.
* **Network Infrastructure**: Node architecture, security framework, validator system, EVM integration.

#### **Development Tools**

* **Kaon CLI & SDKs**: Kaon-cli guide, Rust libraries, Metamask integration.
* **Testing Framework**: Test environments, quality assurance strategies, and security testing.

#### **Deployment & Monitoring**

* **Deployment Process**: Checklist, security verification, monitoring solutions, maintenance procedures.

### **API REFERENCE**

#### **Core APIs**

* **RPC & WebSocket APIs**: Endpoints, methods, real-time events, state updates.
* **Bitcoin & Ethereum Support**: Bitcoin REST, RPC support, and Ethereum RPC.

#### **Advanced Features**

* **Ethereum Gate API**: Web3.js integration, performance optimization, rate limiting, caching strategies.

### **NETWORK & TOOLS**

#### **Network Parameters**

* **Chain Configuration**: Opcodes, gas model, network limits.
* **Security Parameters**: Validator requirements, slashing conditions, attack prevention.

### **TECHNICAL RESOURCES**

#### **Technical Deep Dives**

* **Key Innovations**: Detailed insights into mirrorBTC, Cascade Signature System, Oracle-less operations, and Cross-Consensus Messaging.
* **Renegotiation and Prolonged Agreements**: Explanation of renegotiation in Bitcoin, its importance, and how Kaon facilitates it. Techniques like OP\_CAT, OP\_CSFS, and OP\_TXHASH are used for advanced functionality without compromising Bitcoin's security.
* **BTC-Based Lending**: Mechanics, benefits, and potential use cases of lending on Kaon. Includes concepts like Expiration, Repayment, Collateral Forfeiture, Mutual-Agreement Contract Closure, Surplus & Shortage, Direct Access to BTC, and Advanced Features like Non-Interactive Protocols.
* **Covenants**: Introduction to covenants, their role in enhancing smart contract functionality, and how they are implemented in Kaon.
* **Oracle-less Operation**: Detailed explanation of how Kaon achieves oracle-less operations, including the use of threshold signatures and managing cross-chain interactions.
* **Mirroring**: Explanation of the mirroring process that allows Kaon to reflect transactions and states between chains for seamless interoperability.
* **Trustless Design**: In-depth discussion of ownership, control, oracle-less operations, and permissionless interaction to illustrate Kaon's trustless nature.
* **Security**: Incident handling, potential threats, and mitigation strategies. Covers incident processing, arbitration, and validator slashing mechanisms.
* **Consensus Layer**: Detailed explanation of dPoS and Galaxy consensus mechanisms, roles of Master Nodes, Validators, and Epoch structure.
* **Bitcoin and Alternative Consensus Interactions**: Kaon's interaction with Bitcoin and other consensus mechanisms, including peg mechanisms and advanced financial concepts.

#### **Network Parameters**

* **Chain Configuration**: Opcodes, gas model, network limits.
* **Security Parameters**: Validator requirements, slashing conditions, attack prevention.

### **COMMUNITY & RESOURCES**

#### **Support & Community**

* **Developer Support**: Documentation updates, technical support, migration guides.
* **Community Resources**: Forums, contribution guidelines, governance, and voting mechanisms.

#### **Roadmap & Future Developments**

* **Planned Milestones**: Development roadmap, future enhancements, community engagement.
* **Resources**: GitHub repos, research papers, FAQs, and troubleshooting guides.