TOKEN FANSPORTMARKET (FSM) - WHITEPAPER

Why did we choose the BSC (Binance Smart Chain) network?

After launching the mainnet community in April 2019, Binance Chain showcased its high-speed, high-throughput design. Binance Chain's main focus, its native decentralized application ("dApp") Binance DEX, demonstrated its low latency matching with large headroom when handling millions of trade volumes in a short period of time.

Flexibility and usability are often in inverse relationship to performance. Focusing on providing a convenient digital asset issuance and trading platform also brings limitations. The most requested feature of Binance Chain is the programmable extensibility, or simply the Smart Contract and Virtual Machine functions. Issuers and owners of digital assets struggle to add new decentralized features to their assets or introduce any type of governance and community activities.

Another point to be analyzed in relation to competing blockchain networks is the fees (GAS) charged for each transaction within the network. At this point the Binance Smart Chain network has a performance 30 times better than the ETH network, making the movement of FMS Tokens faster and at a lower cost.

The security of a verified blockchain, with thousands of established validators, was another determining factor in choosing the TOKEN FANSPORTMARKET (FMS) network.

TOKEN FANSPORTMARKET (FSM)

Technical Details:

NAME: FANSPORTMARKET

TICKER: FSM

TOTAL SUPPLY: 100.000.000,00 (one hundred million units)

MAX SUPPLY: 100.000.000,00 (one hundred million units)

Blockchain Network: BSC (Binance Smart Chain)

Block Explorer: https://bscscan.com

Contract Address: (soon)

Block Time: 3-5 seconds

Wallets Supported: Metamask and TrustWallet by: MaC, Windows, Linux, iOS and Android

***In the next 15 months, October 2021 - December 2022, 4.000.0000 (Four million) Fansportmarket (FSM) TOKENS will be transferred to a virtual wallet from UJUART, SL, until the total value of 60.000.000 Fansportmarket (FSM) TOKENS is reached in order to carry out the capitalization 1st January 2023.*

Technical Details - BSC network

BC supports BEP2 tokens and upcoming BEP8 tokens, which are native assets transferrable and tradable (if listed) via fast transactions and sub-second finality. Meanwhile, as BSC is Ethereum compatible, it is natural to support ERC20 tokens on BSC, which here is called “BEP2E” (with the real name to be introduced by the future BEPs,it potentially covers BEP8 as well). BEP2E may be “Enhanced” by adding a few more methods to expose more information, such as token denomination, decimal precision definition and the owner address who can decide the Token Binding across the chains. BSC and BC work together to ensure that one token can circulate in both formats with confirmed total supply and be used in different use cases.

Token Binding

BEP2 tokens will be extended to host a new attribute to associate the token with a BSC BEP2E token contract, called “Binder”, and this process of association is called “Token Binding”.

Token Binding can happen at any time after BEP2 and BEP2E are ready. The token owners of either BEP2 or BEP2E don’t need to bother about the Binding, until before they really want to use the tokens on different scenarios. Issuers can either create BEP2 first or BEP2E first, and they can be bound at a later time. Of course, it is encouraged for all the issuers of BEP2 and BEP2E to set the Binding up early after the issuance.

A typical procedure to bind the BEP2 and BEP2E will be like the below:

Ensure both the BEP2 token and the BEP2E token both exist on each blockchain, with the same total supply. BEP2E should have 3 more methods than typical ERC20 token standard:

- symbol(): get token symbol
- decimals(): get the number of the token decimal digits
- owner(): get BEP2E contract owner’s address. This value should be initialized in the BEP2E contract constructor so that the further binding action can verify whether the action is from the BEP2E owner.

Decide the initial circulation on both blockchains. Suppose the total supply is S, and the expected initial circulating supply on BC is K, then the owner should lock S-K tokens to a system controlled address on BC.

Equivalently, K tokens is locked in the special contract on BSC, which handles major binding functions and is named as TokenHub. The issuer of the BEP2E token should lock the K amount of that token into TokenHub, resulting in S-K tokens to circulate on BSC. Thus the total circulation across 2 blockchains remains as S.

The issuer of BEP2 token sends the bind transaction on BC. Once the transaction is executed successfully after proper verification: It transfers S-K tokens to a system-controlled address on BC.

A cross-chain bind request package will be created, waiting for Relayers to relay.

BSC Relayers will relay the cross-chain bind request package into TokenHub on BSC, and the corresponding request and information will be stored into the contract.

The contract owner and only the owner can run a special method of TokenHub contract, ApproveBind, to verify the binding request to mark it as a success. It will confirm:

- the token has not been bound;
- the binding is for the proper symbol, with proper total supply and decimal information;
- the proper lock are done on both networks.

Once the ApproveBind method has succeeded, TokenHub will mark the two tokens are bounded and share the same circulation on BSC, and the status will be propagated back to BC. After this final confirmation, the BEP2E contract address and decimals will be written onto the BEP2 token as a new attribute on BC, and the tokens can be transferred across the two blockchains bidirectionally. If the ApproveBind fails, the failure event will also be propagated back to BC to release the locked tokens, and the above steps can be re-tried later.

Cross-Chain Transfer and Communication

Cross-chain communication is the key foundation to allow the community to take advantage of the dual chain structure:

- users are free to create any tokenization, financial products, and digital assets on BSC or BC as they wish;
- the items on BSC can be manually and programmingly traded and circulated in a stable, high throughput, lighting fast and friendly environment of BC.

Users can operate these in one UI and tooling ecosystem.

Cross-Chain Transfer

The cross-chain transfer is the key communication between the two blockchains. Essentially the logic is:

- the transfer-out blockchain will lock the amount from source owner addresses into a system-controlled address/contracts;
- the transfer-in blockchain will unlock the amount from the system-controlled address/contracts and send it to target addresses.

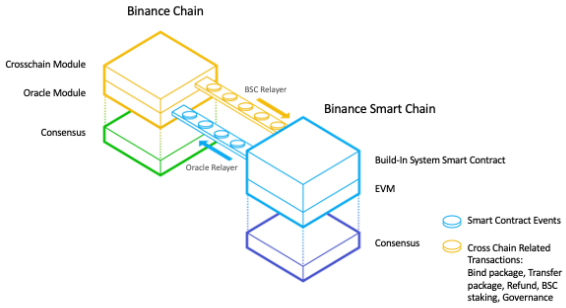
The cross-chain transfer package message should allow the BSC Relayers and BC Oracle Relayers to verify:

Enough amount of token assets is removed from the source address and locked into a system-controlled addresses/contracts on the source blockchain. And this can be confirmed on the target blockchain.

Proper amounts of token assets are released from a system-controlled addresses/contracts and allocated into target addresses on the target blockchain. If this fails, it can be confirmed on source blockchain, so that the locked token can be released back (may deduct fees).

The sum of the total circulation of the token assets across the 2 blockchains are not changed after this transfer action completes, no matter if the transfer succeeds or not.

Cross-Chain:



The architecture of cross-chain communication is as in the above diagram. To accommodate the 2 heteroid systems, communication handling is different in each direction.

BC to BSC Architecture

BC is a Tendermint-based, instant finality blockchain. Validators with at least ⅔*N+1 of the total voting power will co-sign each block on the chain. So that it is practical to verify the block transactions and even the state value via Block Header and Merkle Proof verification. This has been researched and implemented as “Light-Client Protocol”, which are intensively discussed in the Ethereum community, studied and implemented for Cosmos inter-chain communication.

BC-to-BSC communication will be verified in an “on-chain light client” implemented via BSC Smart Contracts (some of them may be “pre-compiled”). After some transactions and state change happen on BC, if a transaction is defined to trigger cross-chain communication,the Cross-chain “package” message will be created and BSC Relayers will pass and submit them onto BSC as data into the "build-in system contracts". The build-in system contracts will verify the package and execute the transactions if it passes the verification. The verification will be guaranteed with the below design.

BC blocking status will be synced to the light client contracts on BSC from time to time, via block header and pre-commits, for the below information:

- block and app hash of BC that are signed by validators;
- current validatorset, and validator set update;
- the key-value from the blockchain state will be verified based on the Merkle Proof and information from above #1.

After confirming the key-value is accurate and trustful, the build-in system contracts will execute the actions corresponding to the cross-chain packages. Some examples of such packages that can be created for BC-to-BSC are:

- Bind: bind the BEP2 tokens and BEP2E;

- Transfer: transfer tokens after binding, this means the circulation will decrease (be locked) from BC and appear in the target address balance on BSC;

- Error Handling: to handle any timeout/failure event for BSC-to-BC communication;

- Validatorset update of BSC.

To ensure no duplication, proper message sequence and timely timeout, there is a "Channel" concept introduced on BC to manage any types of the communication.

For relayers, please also refer to the below "Relayers" section.

BSC to BC Architecture

BSC uses Proof of Staked Authority consensus protocol, which has a chance to fork and requires confirmation of more blocks. One block only has the signature of one validator, so that it is not easy to rely on one block to verify data from BSC.

To take full advantage of validator quorum of BC, an idea similar to many Bridge or Oracle blockchains is adopted.

The cross-chain communication requests from BSC will be submitted and executed onto BSC as transactions. The execution of the transanction wil emit Events, and such events can be observed and packaged in certain "Oracle" onto BC. Instead of Block Headers, Hash and Merkle Proof, this type of "Oracle" package directly contains the cross-chain information for actions, such as sender, receiver and amount for transfer.

To ensure the security of the Oracle, the validators of BC will form anothe quorum of "Oracle Relayers". Each validator of the BC should run a dedicated process as the Oracle Relayer. These Oracle Relayers will submit and vote for the cross-chain communication package, like Oracle, onto BC, using the same validator keys. Any package signed by more than ⅔*N+1 Oracle Relayers' voting power is as secure as any block signed by ⅔*N+1 of the same quorum of validators' voting power.

By using the same validator quorum, it saves the light client code on BC and continuous block updates onto BC. Such Oracles also have Oracle IDs and types, to ensure sequencing and proper error handling.

Timeout and Error Handling

There are scenarios that the cross-chain communication fails. For example, the relayed package cannot be executed on BSC due to some coding bug in the contracts. Timeout and error handling logics are used in such scenarios.

For the recognizable user and system errors or any expected exceptions, the two networks should heal themselves. For example, when BC to BSC transfer fails, BSC will issue a failure event and Oracle Relayers will execute a refund on BC; when BSC to BC transfer fails, BC will issue a refund package for Relayer to relay in order to unlock the fund.

However, unexpected error or exception may still happen on any step of the cross-chain communication. In such a case, the Relayers and Oracle Relayers will discover that the corresponding cross-chain channel is stuck in a particular sequence. After a Timeout period, the Relayers and Oracle Relayers can request a "SkipSequence" transaction, the stuck sequence will be marked as "Unexecutable". A corresponding alerts

will be raised, and the community has to discuss how to handle this scenario, e.g. payback via the sponsor of the validators, or event clear the fund during next network upgrade.

Cross-Chain User Experience

Ideally, users expect to use two parallel chains in the same way as they use one single chain. It requires more aggregated transaction types to be added onto the cross-chain communication to enable this, which will add great complexity, tight coupling, and maintenance burden. Here BC and BSC only implement the basic operations to enable the value flow in the initial launch and leave most of the user experience work to client side UI, such as wallets. E.g. a great wallet may allow users to sell a token directly from BSC onto BC's DEX order book, in a secure way.

Cross-Chain Contract Event

Cross-Chain Contract Event (CCCE) is designed to allow a smart contract to trigger cross-chain transactions, directly through the contract code. This becomes possible based on:

- Standard system contracts can be provided to serve operations callable by general smart contracts;
- Standard events can be emitted by the standard contracts;
- Oracle Relayers can capture the standard events, and trigger the corresponding cross-chain operations;
- Dedicated, code-managed address (account) can be created on BC and accessed by the contracts on the BSC, here it is named as "Contract Address on BC" (CAoB).

Several standard operations are implemented.

BSC to BC transfer: this is implemented in the same way as normal BSC to BC transfer, by only triggered via standard contract. The fund can be transferred to any addresses on BC, including the corresponding CAoB of the transfer originating contract.

Transfer on BC: this is implemented as a special cross-chain transfer, while the real transfer is from CAoB to any other address (even another CAoB).

BC to BSC transfer: this is implemented as two-pass cross-chain communication. The first is triggered by the BSC contract and propagated onto BC, and then in the second pass, BC will start a normal BC to BSC cross-chain transfer, from CAoB to contract address on BSC. A special note should be paid on that the BSC contract only increases balance upon any transfer coming in on the second pass, and the error handling in the second pass is the same as the normal BC to BSC transfer.

IOC (Immediate-Or-Cancel) Trade Out: the primary goal of transferring assets to BC is to trade. This event will instruct to trade a certain amount of an asset in CAoB into another asset as much as possible and transfer out all the results, i.e. the left the source and the traded target tokens of the trade, back to BSC. BC will handle such relayed events by sending an "Immediate-Or-Cancel", i.e. IOC order onto the trading pairs, once the next matching finishes, the result will be relayed back to BSC, which can be in either one or two assets.

Auction Trade Out: Such event will instruct BC to send an auction order to trade a certain amount of an asset in CAoB into another asset as much as possible and transfer out all the results back to BSC at the end of the auction. Auction function is upcoming on BC.

There are some details for the Trade Out:

- both can have a limit price (absolute or relative) for the trade;
- the end result will be written as cross-chain packages to relay back to BSC;
- cross-chain communication fees may be charged from the asset transferred back to BSC.

BSC contract maintains a mirror of the balance and outstanding orders on CAoB. No matter what error happens during the Trade Out, the final status will be propagated back to the originating contract and clear its internal state.

With the above features, it simply adds the cross-chain transfer and exchange functions with high liquidity onto all the smart contracts on BSC. It will greatly add the application scenarios on Smart Contract and dApps, and make 1 chain +1 chain > 2 chains.

Staking and Governance

Proof of Staked Authority brings in decentralization and community involvement. Its core logic can be summarized as the below. You may see similar ideas from other networks, especially Cosmos and EOS.

Token holders, including the validators, can put their tokens “bonded” into the stake. Token holders can delegate their tokens onto any validator or validator candidate, to expect it can become an actual validator, and later they can choose a different validator or candidate to re-delegate their tokens1.

All validator candidates will be ranked by the number of bonded tokens on them, and the top ones will become the real validators.

Validators can share (part of) their blocking reward with their delegators.

Validators can suffer from “Slashing”, a punishment for their bad behaviors, such as double sign and/or instability.

There is an “unbonding period” for validators and delegators so that the system makes sure the tokens remain bonded when bad behaviors are caught, the responsible will get slashed during this period.

Staking on BC

Ideally, such staking and reward logic should be built into the blockchain, and automatically executed as the blocking happens. Cosmos Hub, who shares the same Tendermint consensus and libraries with Binance Chain, works in this way.

BC has been preparing to enable staking logic since the design days. On the other side, as BSC wants to remain compatible with Ethereum as much as possible, it is a great challenge and efforts to implement such logic on it. This is especially true when Ethereum itself may move into a different Proof of Stake consensus protocol in a short (or longer) time. In order to keep the compatibility and reuse the good foundation of BC, the staking logic of BSC is implemented on BC.

The staking token is BNB, as it is a native token on both blockchains anyway

The staking, i.e. token bond and delegation actions and records for BSC, happens on BC.

The BSC validator set is determined by its staking and delegation logic, via a staking module built on BC for BSC, and propagated every day UTC 00:00 from BC to BSC via Cross-Chain communication.

The reward distribution happens on BC around every day UTC 00:00.

Rewarding

Both the validator update and reward distribution happen every day around UTC 00:00. This is to save the cost of frequent staking updates and block reward distribution. This cost can be significant, as the blocking reward is collected on BSC and distributed on BC to BSC validators and delegators. (Please note BC blocking fees will remain rewarding to BC validators only.)

A deliberate delay is introduced here to make sure the distribution is fair.

The blocking reward will not be sent to validator right away, instead, they will be distributed and accumulated on a contract.

Upon receiving the validator set update into BSC, it will trigger a few cross-chain transfers to transfer the reward to custody addresses on the corresponding validators. The custody addresses are owned by the system so that the reward cannot be spent until the promised distribution to delegators happens.

In order to make the synchronization simpler and allocate time to accommodate slashing, the reward for N day will be only distributed in N+2 days. After the delegators get the reward, the left will be transferred to validators' own reward addresses.

Slashing

Slashing is part of the on-chain governance, to ensure the malicious or negative behaviors are punished. BSC slash can be submitted by anyone. The transaction submission requires slash evidence and cost fees but also brings a larger reward when it is successful.

So far there are two slashable cases.

Double Sign

It is quite a serious error and very likely deliberate offense when a validator signs more than one block with the same height and parent block. The reference protocol implementation should already have logic to prevent this, so only the malicious code can trigger this. When Double Sign happens, the validator should be removed from the Validator Set right away.

Anyone can submit a slash request on BC with the evidence of Double Sign of BSC, which should contain the 2 block headers with the same height and parent block, sealed by the offending validator. Upon receiving the evidence, if the BC verifies it to be valid.

The validator will be removed from validator set by an instance BSC validator set update Cross-Chain update;

A predefined amount of BNB would be slashed from the self-delegated BNB of the validator; Both validator and its delegators will not receive the staking rewards.

Part of the slashed BNB will allocate to the submitter's address, which is a reward and larger than the cost of submitting slash request transaction

The rest of the slashed BNB will allocate to the other validators' custody addresses, and distributed to all delegators in the same way as blocking reward.

Inavailability

The liveness of BSC relies on everyone in the Proof of Staked Authority validator set can produce blocks timely when it is their turn. Validators can miss their turn due to any reason, especially problems in their hardware, software, configuration or network. This instability of the operation will hurt the performance and introduce more indeterministic into the system.

There can be an internal smart contract responsible for recording the missed blocking metrics of each validator. Once the metrics are above the predefined threshold, the blocking reward for validator will not be relayed to BC for distribution but shared with other better validators. In such a way, the poorly-operating validator should be gradually voted out of the validator set as their delegators will receive less or none reward. If the metrics remain above another higher level of threshold, the validator will be dropped from the rotation, and this will be propagated back to BC, then a predefined amount of BNB would be slashed from the self-delegated BNB of the validator. Both validators and delegators will not receive their staking rewards.

Governance Parameters

There are many system parameters to control the behavior of the BSC, e.g. slash amount, cross-chain transfer fees. All these parameters will be determined by BSC Validator Set together through a proposal-vote process based on their staking. Such the process will be carried on BC, and the new parameter values will be picked up by corresponding system contracts via a cross-chain communication.

Relayers

Relayers are responsible to submit Cross-Chain Communication Packages between the two blockchains. Due to the heterogeneous parallel chain structure, two different types of Relayers are created.

BSC Relayers

Relayers for BC to BSC communication referred to as "BSC Relayers", or just simply "Relayers". Relayer is a standalone process that can be run by anyone, and anywhere, except that Relayers must register themselves onto BSC and deposit a certain refundable amount of BNB. Only relaying requests from the registered Relayers will be accepted by BSC.

The package they relay will be verified by the on-chain light client on BSC. The successful relay needs to pass enough verification and costs gas fees on BSC, and thus there should be incentive reward to encourage the community to run Relayers.

Incentives

There are two major communication types:

- Users triggered Operations, such as token bind or cross chain transfer. Users must pay additional fee to as relayer reward. The reward will be shared with the relayers who sync the referenced blockchain headers. Besides, the reward won't be paid the relayers' accounts directly. A reward distribution mechanism will be brought in to avoid monopolization.

- System Synchronization, such as delivering refund package(caused by failures of most oracle relayers), special blockchain header synchronization(header contains BC validatorset update), BSC staking package. System reward contract will pay reward to relayers' accounts directly.

If some Relayers have faster networks and better hardware, they can monopolize all the package relaying and leave no reward to others. Thus fewer participants will join for relaying, which encourages centralization and harms the efficiency and security of the network. Ideally, due to the decentralization and dynamic re-election of BSC validators, one Relayer can hardly be always the first to relay every message. But in order to avoid the monopolization further, the rewarding economy is also specially designed to minimize such chance.

The reward for Relayers will be only distributed in batches, and one batch will cover a number of successful relayed packages.

The reward a Relayer can get from a batch distribution is not linearly in proportion to their number of successful relayed packages. Instead, except the first a few relays, the more a Relayer relays during a batch period, the less reward it will collect.

Oracle Relayers

Relayers for BSC to BC communication are using the “Oracle” model, and so-called “Oracle Relayers”. Each of the validators must, and only the ones of the validator set, run Oracle Relayers. Each Oracle Relayer watches the blockchain state change. Once it catches Cross-Chain Communication Packages, it will submit to vote for the requests. After Oracle Relayers from ⅔ of the voting power of BC validators vote for the changes, the cross-chain actions will be performed.

Oracle Replayers should wait for enough blocks to confirm the finality on BSC before submitting and voting for the cross-chain communication packages onto BC.

The cross-chain fees will be distributed to BC validators together with the normal BC blocking rewards.

Such oracle type relaying depends on all the validators to support. As all the votes for the cross-chain communication packages are recorded on the blockchain, it is not hard to have a metric system to assess the performance of the Oracle Relayers. The poorest performer may have their rewards clawed back via another Slashing logic introduced in the future.

Outlook

It is hard to conclude for Binance Chain, as it has never stopped evolving. The dual-chain strategy is to open the gate for users to take advantage of the fast transferring and trading on one side, and flexible and

extendable programming on the other side, but it will be one stop along the development of Binance Chain. Here below are the topics to look into so as to facilitate the community better for more usability and extensibility.

Add different digital asset model for different business use cases.

Enable more data feed, especially DEX market data, to be communicated from Binance DEX to BSC.

Provide interface and compatibility to integrate with Ethereum, including its further upgrade, and other blockchain.

Improve client side experience to manage wallets and use blockchain more conveniently.