PHANTOM

The Phantom Technical WhitePaper

[Version 0.1]

The Phantom Team

Phantom Technical Whitepaper V2

Abstract: PHANTOM software introduces a new blockchain structure that aims to increase transaction rates.
Please note that Phantom's design allows it to double its transaction rate, adding hundreds of nodes to its network. As of this article, the Phantom network has multiple nodes.

No one may use, copy, or distribute any material in the White Paper without permission.

Disclaimer: This Phantom Technical White Paper v2 is for reference only. The accuracy or conclusions of this white paper are not guaranteed, but this white paper (1) does not make or expressly waive all express, implied, statutory or other forms of representations and warranties, including but not limited to: marketability, suitable for a particular purpose, suitability, purpose, title, or non-infringement; (2) the contents of this white paper are error-free; and (3) they do not infringe the rights of third party. The block and its affiliates shall not be liable for any kind of damages resulting from the use, reference or reliance on this white paper or any content. It is excluded even if you are informed of the possibility of such damage. In no case will it be blocked. Any person or its affiliates are responsible for any damages, losses, liabilities, expenses or any prior individuals or entities. Any kind of purse, whether direct, indirect, consequential, compensatory, incidental, actual, exemplary, punitive or special, for the use, citation or reliance of this white paper or any content contained in this document including but not limited to any business loss, income, profit, data, use, goodwill or other intangible losses.

Phantom is a redesign from scratch and has been researching and developing for more than 2 years. The cornerstone of Phantom's design is to divide the mining network into smaller parts. The agreed-upon group is called as fragment and each group can process transactions in parallel. If Phantom's network is 8,000 nodes, Phantom will automatically create 10 sub-networks, each with a size of 800, in a decentralized manner without a trusted coordinator. Now, if a subnetwork can agree on a set of (for example) 100 transactions at a time, then 10 subnetworks can agree on a total of 1 transaction and there are a total of 1,000 transactions. The key to secure aggregation is to ensure that sub-networks handle different transactions (without overlapping) without the need for double spending.

These assumptions are similar to existing blockchain-based solutions. We assume that the mining network will have a small number of malicious nodes/identifiers with a total computational power of

frac, which is(<1/4) of the entire network. It is based on a standard scheme, but it has a new two-level block chain structure. The algorithm has a highly optimized consensus algorithm.

Phantom also provides an innovative, special-purpose smart contract language and execution environment that leverages the underlying architecture to provide large-scale and highly-efficient complications to the orbital platform.

The smart contract language in Phantom follows the data flow programming style, in which smart contracts can be represented as directed graphs. The node is an operation or func. The arc between the two nodes represents the output of the first node and the input to the second node. Once all the inputs to the node are in effect, and therefore become a data stream, the nodes are activated (or operated). w Contracts are essentially parallel and suitable for decentralized systems, including simple calculations such as search, sorting, and linear algebraic calculations. Complex

calculations such as training neural networks, data mining, financial modeling, scientific calculations, and general MapReduce tasks.

The smart contract language in Phantom follows the data flow programming style, in which smart contracts can be represented as directed graphs. The node is an operation or func. The arc between the two nodes represents the output of the first node and the input to the second node. Once all the inputs to the node are in effect, and therefore become a data stream, the nodes are activated (or operated). w contracts are essentially parallel and suitable for decentralized systems, including simple calculations such as search, sorting, and linear algebraic calculations, and complex calculations, such as training neural networks, data mining, financial modeling, scientific calculations, and general MapReduce tasks.

Background

System settings and assumptions

Smart contract layer

Free Usage

Easy to upgrade and failback

Low latency

Sequential performance

Parallel performance

Transaction confirmation

Proof of transaction as a bet (Tapos)

Named permission levels

Permission mapping

Evaluation privilege

Default privilege group

Privilege calculation

Forced procrastination

Parallel execution of application certainty

Minimum communication delay

Completeness and integrity

Conclusion

Background

Blockchain technology was introduced in 2008 with the introduction of Bitcoin. Since then, entrepreneurs and developers have been trying to promote this

technology to support a wider range of applications. Applications on f single blockchain platform.

Although many blockchain platforms have difficulty in supporting decentralized applications, application-specific blockchains such as Bitshare Distributed Exchange (2014) and Steem's social media platform (2016) have become the heavily used blockchains for thousands of active users every day. They do this by increasing performance to thousands of transactions per second. The delay was reduced to 1.5 seconds, eliminating the cost per transaction and providing a user experience similar to that provided by existing centralized services.

The existing blockchain platform has a huge burden and limited computing power, which hinders widespread adoption of blockchain.

System settings and assumptions

Entities in Phantom. There are two main entities

Phantom: Users. Users are external entities that use Phantom's infrastructure to transfer funds or run smart contracts. Nodes of the Phantom consensus project running in the network are rewarded for their services. In the rest of this white paper, we used the terms miner and nodes interchangeably.

Phantom's mining network is further divided into several smaller networks called fragments. A group called a DS node is assigned to a fragment, and this group of ds nodes is also referenced. As a DS committee, each fragment and DS committee has a leader. Leaders play an important role in Phantom's consensus agreement and the overall operation of the network.
Each user has a public private key pair for a digital signature scheme. Each node in the network has a related IP address and a public key that is used as an identifier.

Phantom has an intrinsic token called payings, or pays for short. Payings grants platform usage rights to users when it is used to pay for transactions or smart

running. Regarding the contract in this white paper, any reference to amount, value, balance, or payment should be assumed to be pays antagonistic model. We assume that the mining network has a small fraction_of Byzantine nodes/identities at any point in time. Its total computational power is at most a part of the entire network, where $0 \leqslant f < 1$ and n is the total size of the network. The factor is an arbitrary constant that is bounded by the selected parameter to obtain a reasonable constant parameter. We further assume that the honest node S is reliable during the operation of the protocol, and failed or disconnected nodes are calculated in the Byzantine fraction.

The Byzantine node can deviate from the protocol, delete or modify the message, and send different messages to the honest node.

In addition, all Byzantine nodes can be collocated. We assume that the total computational power of the Byzantine opponent is still limited to the standard cryptographic hypothesis of the probability polygon,

nominally the opponent. However, we believe that information of honest node (in the case of network partitions) can be delivered to honest destinations after a certain constraint δ, but δ may change. The bound δ is used to ensure activity but not safe. If this timing and connection assumption is not satisfied, the Byzantine node may significantly delay the message (simulation c gain), or worse, "eclipse" the network. In the case of network partitions, according to the requirements of the CAP theorem, one can only choose between consistency and availability. In Phantom, we OOSE must be consistent and sacrifice usability.

Smart contract layer

Phantom provides an innovative, special-purpose smart contract language and execution environment that leverages the underlying architecture to provide large-scale and efficient computing. Platform, in this section, we will introduce the smart contract layer using data flow programming architecture.

A. Using Data Flow Paradigm to Calculate Fragments

Phantom's smart contract language and its execution platform are designed to take advantage of the underlying network and transactional segmentation architecture. The segmentation architecture is ideal for running calculations. Efficient work, the key idea is as follows: Only a subset of the network (such as fragmentation) can perform calculations. We call this method computational sand. Compared with existing smart contract architectures (such as Islien), ding requires a very different approach to handle contracts when Phantom calculates the fragments. In Phantom, each complete N requires ODE to perform the same calculation to verify the calculation results and update the global state. Although safe, this completely redundant programming model is a daunting e for running large-scale computations that can be easily parallelized. Examples include simple calculations such as search, sort, and linear algebra calculations, as well as more complex calculations. UCH is used to train neural networks, data mining, and financial

modeling.

Phantom's computational segmentation method relies on a new smart contract language. It is not Turing-complete, but for many applications, scalability is much better. The smart contract language Liqa follows the data flow programming style. In the data flow execution model, the contract is represented by a directed graph. The nodes in the figure are primitive instructions or operations. Tedarcs between two nodes of Derek represent data dependencies between operations, which is the output to the first operation and the input to the second operation. A node is activated when all its i are activated (or operated). Nput is available. This is in stark contrast to the classic von Neumann execution model (as used in Phantom), where instruction, regar is executed only when the program counter reaches it, regardless of whether it can be executed earlier.

The main advantage of using the data flow method is that multiple instructions can be executed at the

same time. Therefore, if several nodes in the graph are activated at the same time, they can be executed, d parallel. This simple principle offers possibilities for massively parallel execution. To see this, we show a simple sequential program in Figure 1a, which has three instructions, 1b, w and e, rendering the data stream variants. Under the Von Neumann execution model, the program will run in three time units: first calculate A, then B and finally C. The model does not capture the fact that the DB can be calculated independently. On the other hand, the data flow program can calculate these two values in parallel. Once A and B are available, the node that performed the addition is activated.

When running on Phantom's slicing network, each node in the data flow program can ultimately be attributed to a fragment, or even a subset of nodes in a fragment. Therefore, the architecture is id. The EAL is used for any MapReduce-style computational task, where some nodes perform mapping tasks and another node can act as a reducer to aggregate the work done by each

mapper.

To facilitate the execution of data flow programs, Phantom's SMART contract language has the following features:

Operate globally shared virtual memory space across the entire blockchain.

The middle unit is locked in the virtual shared memory space during execution.

In the process of submitting to the block chain, check the results of the middle point.

B. Smart security budgeting

In addition to providing the benefits of parallelism from the data flow computing model, Phantom also provides a flexible security budgeting mechanism for computing sharding. This feature is Ena. Computational resources in a blockchain network are segmented by coverage over a consensus process. Calculation fragment allows Phantom users and application running on Phantom specifies the size of the negotiation group to be calculated for each subtask.

Then, each consensus team will be assigned to calculate the same subtask and produce results. User specified conditions. After accepting the results, for example, all members of the consensus group must produce the same result, or 3/4 of them must produce the same result, and so on.

Users of applications running on Phantom can budget for how much she wants to spend on computing and security. In particular, users running a particular deep learning application may choose sp. When running more different neural network tasks, more gas costs will be ended instead of repeating the same calculations for too many nodes. In this case, she can specify a smaller consensus group to run each neural network calculation. On the other hand, complex financial modeling algorithms that require higher accuracy may require a consensus group consisting of more nodes to compute the critical values. Some algorithms are better protected against potential tampering and manipulation.

C. Extensible application

Phantom aims to provide a platform to run highly scalable calculations in multiple areas such as data mining, machine learning, and financial modeling. Self-supporting effective s intensive Turing-complete programs are very challenging, and there are public blockchains that support Turing-complete smart contracts (eg, Etal um), and Phantom focuses on specific applications, but today it did not meet the requirements.

Calculations with parallel computational load: Scientific calculations of big data are a typical example where a large amount of distributed computing power is required. Moreover, these calculations have a high degree of parallelism, such as linear algebra operations on large matrices, searches for massive data, and simulations on large datasets. Phantom PRO calculation tasks are a cheap and short-term turnaround option. In addition, Phantom can be used as an off-the-shelf, highly reliable resource with a large

amount of computation if appropriate incentive mechanism, computational segmentation and security budgets are available.

Training of neural networks: With the increasing popularity and use of machine learning (especially deep learning), an infrastructure must be established that allows for deep learning models of trai. n is about large data sets. As we all know, the training of large data sets is crucial to the accuracy of the model. For this reason, Phantom's computational segmentation and streaming languages will be particularly useful for building machine learning applications. It will serve as a basic structure. You can run tools like TensorFlow by performing different calculations independently on the task group of the Phantom node, to calculate gradients, apply activation functions, calculate training losses, etc.

The application of high complexity and high accuracy algorithms: Unlike the above applications, some applications, such as calculations on financial models, may require high-precision calculations. Any slight

deviation in a part of the calculation may result in a significant loss of investment. Such an application can be in the consensus group with more task nodes in Phantom to allow them to cross. Giving each other's calculation results, the key challenge of offloading the computational tasks of such financial modeling algorithms to a common platform, such as Phantom is the concern of dat. About the privacy and intellectual property of the algorithm, first of all, we envisage that some of the well-known parts of this calculation can be placed in Phantom for efficient and secure calculations first, and future research and development of wh ILE will further strengthen the protection of data privacy and intellectual property rights for such applications..

One of the concepts we need is the cumulative weight of a transaction: it is defined as the sum of its own weight for a particular transaction plus the own weight for all transactions directly or in Indir. Approval of transaction: the algorithm of cumulative weight calculation. These boxes represent transactions, and

the small numbers in the SE corner of each box represent their own transactions. Weights: bold numbers indicate cumulative weights. For example, transaction F is approved directly or indirectly by transactions A, B, C, and E. The cumulative weight of F is 9 = 3 1 31 1, w HICH is the sum of the self-weight of F and the sum of the weights of A, B, C, and E.

D. More

We define the weights and related concepts of the transaction. The proportion of the weight of the transaction and the node to which the issuing node invests: in current IM for Phantom, the weight may only assume a value of 3n, where n is a positive integer and belongs to a non-empty interval of acceptable values. In fact, I do not know how Wei has achieved good results in practice. The important thing is that each transaction has a positive integer whose weight is attached to it. In general, the idea is that the transactions with greater weight are more ″i″. It's more important than a small weight deal. To avoid spam and other attack patterns, it is assumed that no entity

can generate a large number of transactions with “acceptable” weights in the sho.

One of the concepts we need is the cumulative weight of a transaction: it is defined as the sum of its own weight for a particular transaction plus its own weight for all transactions directly or in Indir. Approve this transaction. The algorithm for calculating cumulative weights is shown below. These boxes represent transactions, and the small number in the SE corner of each box represents its own transaction. Weights, bold numbers indicate cumulative weights. For example, transaction F is approved directly or indirectly by transactions A, B, C, and E. The cumulative weight of F is 9 = 3 1 31 1, w HICH is the sum of the self-weight of F and the sum of the weights of A, B, C, and E.

Let’s define "hint" as an unapproved transaction in a confusion graph. In the top chaotic snapshot, the only hints are A and C, when new transaction X arrives and approves A and C in b. Tottom snapshot, X becomes the only hint. The cumulative weight of all other

transactions has increased by 3, and the weight of X itself has increased its own weight.

We need to introduce two additional variables to discuss the approval algorithm. First of all, for the entangled business site, we will introduce its:

Height: The length of the longest directional path leading to creation;

Depth: The length of the longest reverse path pointing to some tip.

There is a height of 1 and a height of 4 because of the reverse path F, D, B, A, and D has a height of 3 and a depth of 2. In addition, let us introduce the concept of scores. By definition, a transaction is the sum of its own weights for all transactions approved by the transaction, plus the transaction's own weight. The only tips are A and C. Transaction A approves transaction B, D, F, and G directly or indirectly, so A's score is 1+3+1+3+1=9. Similarly, the score for c is 1+1+1+3+1=7.

In order to understand the arguments presented in this article, one can safely assume that all transactions have their own weight equal to one. From now on, we insist on this assumption. Under this assumption, Transaction X's cumulative weight is 1 plus the number of transactions that directly or indirectly approve X, with a score of 1 plus direct or the transactions of I, being recognized indirectly by X-rays.

We note that in the definitions of this section, cumulative weight is the most important measure, although height, depth, and score will be briefly entered in some discussions.

E. Splitting attack

The following attack scheme for the Phantom algorithm. Under high load conditions, an attacker can try to split kelp into two branches and balance them. This will allow these two branches to continue to grow.

The attacker must place at least two conflicting transactions at the beginning of the split to prevent the honest node from effectively connecting, while referring to the branches. Then, the attackers hope that about half of the networks can contribute to each branch so that they can "compensate" for randomization. Volatility means a relatively small personal computing power. If this technique works, the attacker will be able to spend the same money on both branches.

To defend against such attacks, we need to use a "sharp threshold" rule, which makes it too difficult to maintain the balance between the two branches. An example of such a rule is to select the longest chain on the bitcoin network. Let us turn this concept into an entanglement that is suffering from a splitting attack. Assume that the total weight of the first branch is 537 and the total weight of the second branch is 8528. If an honest node selection probability is very close to the first branch of 1/2, then the attacker is likely to be able to maintain the balance between these branches. However, if the probability of the first

branch of an honest n ODE selection is much greater than 1/2, the attacker may not be able to maintain balance. It is not possible to maintain balance e between two branches. The latter case is due to the fact that after inevitable random fluctuations, the network will quickly select one of the branches and give up the other. In order for the Phantom algorithm, it is necessary to choose a very fast decay function f and initiate random walks at deeper nodes, making it very likely to start walking before branching. In this case, the random walk will select the "heavy" branch with a higher probability, even if the cumulative weights between the competing branches are very small.

It's worth noting that due to network synchronization issues, the attacker's mission is very difficult: they may not know much of the recently released transactions. Another effective MET defense against splitting attacks is to allow a powerful entity to publish a large number of transactions on one branch at a time, thus rapidly changing power balance. Make it difficult for attackers to handle this change. If

an attacker manages to keep the split, then the nearest transaction will have only about 50% of the confirmed trust (SectSect). The branches will not grow. In this case, the "honest" node may decide to start approving transactions that occurred before the fork, bypassing t. It has the opportunity to approve conflicting transactions on separate lines.

One can consider other versions of the hint selection algorithm. For example, if two large subtle points are seen in a node, it will select a node with a larger sum of its own weights before performing the Phantom prompt according to the above selection algorithm.

For the future implementation, the following ideas may be worth considering. We can make the transition probability defined in (13) depend on both hx-hy and hx, so that the next step of the third stage can be changed. Markov chains are almost deterministic when the walkers are in deep consensus, and become more random when the walkers approach the end. This will help

to avoid entering weaker branches, while ensuring that when these two hints are selected, there is an uncertainty of randomness.
End (plural of conclusion):

We considered the attack strategy when the attacker tried to double the expenses by "over" the system speed.

A “massive” attack means that, in order to double the expenses, the attacker tries to give the dual spending transaction a very large weight so that it will exceed the legal subtleties. When the allowed weight is unbounded, s policy is a threat to the network. As a solution, we can limit the transaction's own weight from above, or set it to a constant value.

When the maximum self-priority of a transaction is m, the best attack strategy is to generate a transaction with its own weight, m, that refers to a dual-overhead transaction. When the INP is compared to the attacker's computing power, the ut flow of an "honest" transaction is large enough so that it is possible to estimate the

likelihood of a double-spending transaction having a greater cumulative weight. formula).

The attack method of constructing the "parasite chain" makes the height or score-based approval strategy obsolete because the attacker's site has higher value on these indexes than LE. The ultimate tangled. On the other hand, the PhantomTip selection algorithm described in Section 4.1 seems to provide protection against such attacks.

The Phantom hint selection algorithm also provides extra incentives to prevent delay nodes.

Resistance of F. to quantum computing

It is well known that a sufficiently large quantum computer 35 can very effectively deal with problems that rely on trial and error to find a solution. Finding a "now" process, eating a bitcoin block is a good example. Since today, we must check an average of 268 non-CES to find a suitable hash that allows new blocks to be generated. (Everybody knows) TOM computer words

need to handle INSTRON(N) operations to solve problems similar to the Bitcoin puzzle mentioned above. The same problem also requires INSTEM(N) operations on a classic Song computer. Therefore, the quantum computation r value is about 268 = 234 ≈ 17 billion times, which is more efficient than a classical computer on mining a bitcoin blockchain. In addition, it is worth noting that if a blockchain does not increase its difficulty, as a response to increased hash capabilities, the rate of isolated blocks will be increased.

For the same reason, "heavy weight" attacks will also be more effective on quantum computers. However, as suggested in Section 4, limiting the weight from above will effectively prevent qua, as will computer attacks. This is obvious because the number of that need to be checked to find the right hash to issue a non-Ces transaction is not unreasonable. Age on Aver is about 38 years old. Therefore, the "ideal" quantum computer's efficiency gain is 34 = 81, which is quite acceptable. What's more, the algorithm used to implement the structure is such a time to find a random number. In

order to obtain a wide range of applications, the applications on the chain need a platform that is flexible enough to meet the following requirements:

Supporting millions of users to compete with companies such as ebay, uber, airbnb, and facebook requires high-performance technologies capable of handling tens of millions of active users. In some cases, the application may not reach a critical number of users, so a platform that can handle a large number of users is crucial.

Cross-chain transaction of Free Usage

Application developers of cross-chain transaction need to provide users with the flexibility to trade cross-chain transactions for free; users should not pay to use the platform or benefit from their services. A blockchain platform, which is freely available to users, may be used more widely. Developers and companies can develop effective cross-chain monetization strategies.

Easy to upgrade and failback

Businesses building blockchain-based applications need the flexibility to enhance the new features of their applications. The platform must support software and smart contract upgrades.

All non-trivial software is affected by errors, even the most rigorous formal verification. The platform must be robust enough to fix bugs when bugs inevitably occur.

Low latency

A good user experience requires reliable feedback with a delay of no more than a few seconds. Longer delays can frustrate users and make applications built on the blockchain uncompetitive with existing NOs. N-block chain alternatives: the platform should support low latency for transaction latency.

Due to the order-dependent steps of DPOS's large super nodes, some applications cannot implement parallel algorithms. Applications such as swaps need enough sequential performance to handle hi. Gh volume. Therefore, the platform should support fast sequential performance.

Parallel performance

Large applications need to divide workloads across multiple CPUs and computers.

Consensus algorithm (BFT)

The PHANTOM.IO software uses the only known decentralized consensus DPOS algorithm. This algorithm can satisfy the performance requirements of the application program on the blockchain, that is, the trust proof of delegation (DPO). Lander algorithm, using PHANTOM.IO software to hold tokens on the blockchain, can select block producers through a continuous approval voting system. Anyone can choose to join the block. Produce, and will have the

opportunity to produce blocks, as long as they can persuade token holders to vote for them.

PHANTOM.IO software allows blocks to be generated precisely every 0.5 seconds, and exactly one producer is authorized to generate blocks at any given point in time. If there is no editing time in the SCH generation block, then skip the block of that time slot. When one or more blocks are skipped, there is an interval of 0.5 or more seconds in the blockchain.

Used in software, blocks are produced in n rounds (x blocks per unit, n producers per time). At the beginning of each round, n unique producers were decided by Toke's vote. n holder, the selected manufacturer arranges the order according to the agreement of n or more manufacturers.

If a producer misses a block and has not produced any blocks in the past 24 hours, they will be considered to be removed from therein until they inform the blockchain that they intend to start producing blocks.

Gain: it ensures that the network runs smoothly, by minimizing the number of blocks missed by the non-scheduled producers, which have proven to be unreliable.

Under normal circumstances, the DPOS blockchain does not experience any bifurcation because block producers cooperate in producing blocks rather than compete. If there is a fork, the consensus will switch to the longest chain in automatic way. This method works because the speed at which blocks are added to a block fork is directly related to the percentage of block producers that share the same block. DPOS consensus: in other words, the length of the blockchain forks of more manufacturers grows faster than the lesser forks of manufacturers, because the more producers, the less the forks, and they are ignored.

In addition, any block manufacturer should not produce two fork blocks at the same time. A block maker that prevents doing so is likely to be eliminated. This type of cryptographic evidence for dual production of

m can also be used to automatically clear abusers.

Byzantine fault tolerance was added to the traditional handicapped organization, allowing all producers to sign all blocks as long as no producer marks two blocks with the same time stamp or the same block height. A 15-minute painter has signed a block that is considered irreversible. Any Byzantine writer must sign two blocks with the same time stamp or block, resulting in treasonable cryptographic evidence. Under this model, an irreversible consensus should be reached within 1 second.

Transaction confirmation

A typical association of organizations of persons with disabilities has 100% of producers involved. After an average transaction time of 0.25 seconds, the transaction can be considered as confirming 99.9% certainty.

In addition to disabled people's organizations, PHANTOM.IO also added asynchronous Byzantine Fault

Tolerance (ABFT) to accelerate irreversibility. The aBFT algorithm provides 100% irreversibility confirmation within 1Seco.

The software requires that each transaction contain a portion of the hash of the most recent block header. This hash provides two purposes:

To prevent replay of transactions on forks that do not include reference blocks; and

Signals when a particular user and their listed shares are placed on a particular fork.

Over time, all users will eventually confirm the blockchain directly, which makes it difficult to fake chains because fakes cannot migrate transactions from legitimate chai. Haploid generation (code)

Named permission levels

For use in software, accounts can define named

permission levels, and each permission level can be derived from higher level named permissions. Each named privilege level defines one privilege; one privilege is a threshold multi-signature check that consists of the keys and/or naming privilege levels of other accounts. For example, the “Friends” permission level on the account can be set to be controlled equally by any friend of the account.

Another example is the Steem blockchain, which has three hard-coded named permission levels: owner, activity and release. Delivery licenses can only perform social actions such as voting and post-processing, while Active permissions can do everything except change the owner. The owner’s permission is for refrigerated and can do all things. The PHANTOM.IO software allows each account holder to define their own hierarchy and grouping of actions.

Permission mapping

The PHANTOM.IO software allows each account to

define the mapping between any other account's contracts/operations or contracts and their own named permission levels. For example, account holders can map ac. The counting holder's social media applies for the account holder's "friend" permission group. With this map, any friend can post the account holder' s identity on the account holder' s social media. Even numbers, they will put their account holders, they still use their own keys to sign the action. This means that it is always possible to determine which friends use the account and how to use it.

Evaluation authority

When an action of "Action" is passed, the PHANTOM.IO software will first check whether Alicepc's permission map. If nothing is then mapping. If no more matches are found, the map is assumed to point to the named permission group

Once the mapping is identified, the threshold multi-signature process and the rights associated with

the naming authority verify the signature authority. If it fails, it will move upwards. Go to parent permissions and eventually point to owner permissions.

Default permission group

The technology also allows all accounts to have an "Owner" group that can complete all tasks, and can do any "Activity" group other than changing the owner group. All other permission groups S are derived from "activity".

Parallel Computing of Permission

The permission evaluation process is "read-only" and changes to the permissions of the transaction do not take effect until the end of the block. This means that all keys and permission calculation transactions can be executed in parallel. In addition, it means that you can quickly verify permissions without starting expensive application logic, and these logic must be rolled back. Finally, it means that transaction privileges can be evaluated when pending transactions

are received without having to re-evaluate them when applied.

For all things considered, privilege validation represents a significant percentage of the calculations needed to validate transactions, making this a read-only and trivial parallel process can greatly improve performance.

When replaying to regenerate a deterministic state from the operation log, there is no need to re-evaluate the rights. The block of facts contained in the known product is sufficient to skip this step, which greatly reduces the computational load associated with replaying a growing blockchain.

Actions of mandatory delays

Time is a key component of security. In most cases, it is not possible to know if the private key has been stolen until it is used. Time-based security is used when people have applications that require a key to be

stored on a computer connected to the Internet for daily use. PHANTOM.IO software enables application developers to indicate certain operations that must wait for the minimum value of each operation, and they can be applied after being included in a block. During this time, they can be cancelled.

When these operations are broadcasted, users can receive notifications via email or text messages. If they are not authorized, then they can use the account recovery process to restore their account d withdrawal actions.

The delay required depends on how sensitive the operation is. Buying coffee may not be delayed and is irreversible within a few seconds, while buying a house may require a 72-hour clean-up time. It may take up to 30 days for the entire account to be handed over to the new control. The actual delay is chosen by the application developer and the user.

Application deterministic parallel execution

Blockchain consensus depends on deterministic (repeatable) behavior. It means that all parallel executions must be freed from the use of mutexes or other locking primitives. Without locks, there must be some way to ensure that transactions that can be executed in parallel do not produce ambiguous results.

Software designed in 2018 will run a single thread, but it contains the data structures necessary for future multithreaded, parallel execution.

Based on PHANTOM.IO software, once the parallel operation is enabled, the task of the block manufacturer is to transfer the actions to independent fragments so that they can be evaluated. The schedule is the output of the block producer and will be executed, but the process of generating the schedule is not deterministic. This means that block producer S can use parallel algorithms to schedule transactions.

When part of the parallel execution means that when

a script generates a new Action, it will not be delivered immediately, but it plans to deliver it in the next cycle. The reason is that it cannot be Ivered immediately because the receiver may actively modify its state in another fragment.

Minimum communication delay

Delay is the time it takes for receiving a response after an account sending an Action to another account. The goal is to enable two accounts to exchange operations back and forth in a single block. There is no need to wait 0.5 seconds between each action. To achieve this, PHANTOM.IO software divides each block into loops. Each loop is divided into fragments and each fragment contains a trans-list of actions. Each transaction contains a set of operations to be delivered. This structure can be visualized as a tree in which alternating layers are processed sequentially and in parallel.

Region

Cycle (order)

Fragments (parallel)

Transaction (order)

Action (order)

Receiver and notification account (parallel)

Transactions generated in one cycle can be delivered in any subsequent cycle or block. The block generator will always add a loop to the block until the maximum wall-clock time passes or there is no n. EW generates the transaction to be delivered.

You can use static analysis of blocks to verify that in a given period, no two fragments contain transactions that modify the same account. As long as the invariants remain unchanged, the block ca will remain unchanged. n Processes by running all fragments

in parallel.

Read-only operation handler

Some accounts may be able to handle a pass/fail based operation without modifying their internal state. If this is the case, these handlers can be executed in parallel as long as they are read only. The Ly operation handler for a particular account is contained in one or more shards within a specific period.

Multi-account atomic transactions

Sometimes it is necessary to ensure that actions are delivered and accepted by multiple accounts. In this case, both operations are placed in one transaction and two accounts will be allocated. The same pieces and actions apply in sequence.

Non-equilibrium partial evaluation

Scale block chain technology requires that the

components be modular. Everyone doesn't have to run everything, especially if they only need to use a small part of the application.

Exchange application developers run full nodes to display Exchange status to users. This exchange application does not require the status associated with social media.

The software allows any full node to select any subset of applications to run. If the application never depends on stat, operations passed to other applications will be safely ignored.

Subjective best effort scheduling

The software cannot force the vendor to pass any action to any other account. Each block producer makes its own subjective measurement of computational complexity and time. This applies to whether a transaction is generated by a user or automatically generated by a smart contract.

In the boot blockchain using PHANTOM.IO software, at the network level, all transactions count toward the computational bandwidth cost based on the number of WASM instructions executed. However, each block producer using the software can use its own algorithms and metrics to calculate resource usage. When a block producer concludes that a deal or account has consumed a prop. When generating their own blocks, they simply reject the transaction; however, if other block producers consider the transaction, they will still process the transaction.

In general, as long as a block producer considers a transaction to be valid, and all other block producers accept it under resource use restrictions, it may take 1 minute to find the producer's transaction.

In some cases, the producer may create a block that contains a transaction that is out of the acceptable range. In this case, the next block producer may choose to reject block D. The tie will be broken by the third producer. This is no different from what would happen

if a large block caused network propagation delays. The society has noticed a pattern of abuse. Remove ballots from hooligan producers.

This subjective evaluation of computational costs has forced people to accurately measure the time to run. With this design, there is no need for PRESI. Calculating directives greatly increases the opportunity for optimization without breaking the consensus.

Proof of integrity

When using the proof of Merkle from an external blockchain, there is a big difference between knowing that all transactions are valid and knowing that no transactions have been skipped or OMI. Although it is impossible to prove that all the recent transactions are known, it can be proved that there are no loopholes in the history of the transaction. PHANTOM.IO Software Tool is achieved by assigning a serial number to each operation passed to each account. Users can use these

serial numbers to prove that all operations for a specific account have been processed. And they are processed in order.

Conclusions

PHANTOM.IO software is designed based on proven concepts and best practices and represents the basic progress of blockchain technology. The software is part of the overall blueprint. For a blockchain society with global scalability, decentralized applications can be easily deployed and managed.

PHANTOM

The Phantom Technical WhitePaper

[Version 0.1]

The Phantom Team

Phantom Technical Whitepaper V2