IPS

A scalable and regulatable distributed platform based on DHT

Nicolas Brock, Chevron Hon, Edison, MAK, EDC

hello@ipst.io

March 2018

ABSTRACT: We present a scalable and regulatable distributed computing and storage platform based on DHT. The distributed systems are characterized by the formalism. We illustrate how it is applied to some existing distributed systems, and how we build it with DHT. We present a detailed formal specification of the IPS system, along with an analysis of its systemic integrity, capacity for evolution, total system computational complexity and implications for use-cases.

I.INTRODUCTION

A . NEW LEVEL OF DISTRIBUTED COMPUTING PLATFORMS

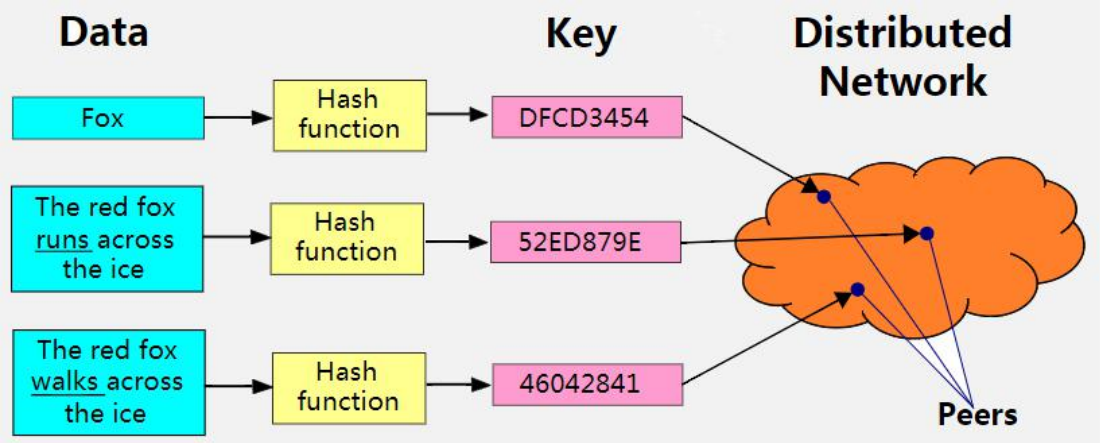
Distributed computing platforms have achieved a new viability level with the advent of two foundational cryptographic tools, namely the secure hashing algorithms(SHA) and public-key encryption(ECC). These have provided solutions for the to key problems in distributed computing: variable, tamper-proof data for sharing state across nodes in the distributed system and confirmation of data provenance via digital signature algorithms. The former is achieved by hash-chains, where monotonic data-stores are rendered intrinsically tamper-proof (and thus confidently sharable across nodes) by including hashes of previous entries in subsequent entries. The latter is achieved by combining cryptographic encryption of hashes of data and using the public keys themselves as the addresses of agents and hence to allow other agents in the system to mathematically verify the data's source.

B.BEYOND BLOCKCHAIN

In Bitcoin (and blockchain in general), the problem is assumed to be how to figure out and choose one block of transactions among the many variants being experienced by the mining nodes (as they collect transactions from clients indifferent orders), and committing that single variant to the single globally shared chain. It focuses on creating , its focus on creating a single shared data reality among all nodes. In Ethereum, although with new magic called sharding or channel state, the solution is almost the same. We claim that this fundamental original stance results directly in the most significant limitation of the blockchain: scalability. This limitation is quite common and therefore there have already been many solutions being offered. IPS offers a way forward by directly addressing the root data-centric assumptions of the blockchain approach on account of the distributed hash table. With sharding and state channel, Ethereum is just a transition for scalability. The IPS architecture based on the distributed hash table serves as the best solution for Dapp.

C. THE MAGIC OF DHT

A distributed hash table (DHT) is a class of a decentralized distributed system that provides a lookup service similar to a hash table with the (key, value) pairs stored in a DHT. Any participating node can efficiently retrieve the value associated with a given key. Responsibility for the maintenance of the mapping between the keys and values is distributed among the nodes in a way that once there is any change over the set of the participants, it would lead to a minimal amount of disruption. This allows a DHT to scale to extremely large numbers of nodes and to handle continual node arrivals, departures, and failures.



D.THE IPS WAY TO BREAK THE LIMITATION

We have adopted a creative solution to solve the performance and scalability problems of the decentralized system. A technology architecture, namely DHT, rather than the blockchain is adopted. Although DHT is not a new technology and even has been adopted in the traditional P2P network for many years, its adoption in blockchain is definitely revolutionary and radical. We use a shared DHT network to store block data instead of storing a complete block data every full node as bitcoin. At the same time, 100 security nodes are established around the world to store complete block data so that data recovery can be automatically carried out if there is any problem occurs in DHT network.

All the previous transactions can be wrapped up in the latest block when packaged. Afterwards, the block data is stored in the DHT network, and the Hash value of the block is broadcast. All the other nodes can trace the latest block data by Hash value and verify the transactions. Once the transactions are verified, the latest block hash can be acknowledged as valid.

That is to say, each node only needs to retain the Hash (if it is a term in your field, it should be capitalized; otherwise, it would be unnecessary) value of the latest 5400 longest chain blocks to ensure the security of the system. The corresponding tertiary block also needs to be retained. This ensures that the data generated by very node, which needs to be saved, is always a fixed size. Meanwhile, it will not grow

with the growth of the system, because the growth data will only exist in the DHT network, which can be easily extended horizontally with the increase in newly added nodes.

In this way, we can achieve unlimited expansion of the block while the delay of any transaction will not exceed 2 seconds, because 1 second is exactly the interval between our blocks. At the same time, it also can release the problem of storage resources waste caused by the unlimited growth of block data and full redundancy.

II. DISTRIBUTED SYSTEMS

A. FORMALISM

We define a simple generalized model of a distributed system Ω using hash-chains as follows:

1. Let N be the set of elements $\{n_1, n_2, \ldots n_n\}$ participating in the system. Name the elements of N ***nodes*** or ***agents***.
2. Let each node n consist of a set S_n with elements $\{\sigma_1,\sigma_2,\ldots\}$. Name the elements of S_n the ***state*** of node n. For the purposes of this paper we assume $\forall \sigma_i \in S_n : \sigma_i = \{\chi_i, D_i\}$ with χ_i being a ***hash-chain*** and D a set of non-hash chain ***dataelements***.
3. Let H be a cryptographically secure hash function.
4. Let there be a ***state transition function***:

$$\tau(\sigma_i, t) = \left(\tau_\chi(\chi_i, t), \tau_D(D_i, t)\right) \qquad (2.1)$$

where:

(a) $\tau_\chi(\chi_i, t) = \chi_{i+1}$ where

$$\begin{aligned} \chi_{i+1} &= \chi_i \cup \{x_{i+1}\} \\ &= \{x_1, \cdots, x_i, x_{i+1}\} \end{aligned} \qquad (2.2)$$

with

$$x_{i+1} = \{h, t\}$$
$$h = \{H(t), y\} \qquad (2.3)$$
$$y = \{H(x_j) | j < i\}$$

Call *h* a ***header*** and notice how the sequence of headers creates a chain (tree, in the general case) by linking each header to the previous header(s) and the transaction.

(b) $D_{i+1} = \tau_D(\sigma_i, t)$

5. Let $V(t, v)$ be a function that takes *t*, along with extra validation data *v*; Verifies the validity of *t* and only if valid calls a transition function for *t*. Call *V* a ***validation*** function.

6. Let $I(t)$ be a function that takes a transaction *t* and then evaluates it via the function *V*; Use τ to transform *S* if it s valid. Call *I* the ***input*** or ***stimulus*** function.

7. Let $P(x)$ be a function that can create transactions t and trigger functions *V* and τ, while *P* itself is triggered by state changes or the passage of time. Call *P* the ***processing*** function.

8. Let *C* be a channel that allows all nodes in *N* to communicate, over which every node has a unique address A_n. Call *C* and the nodes that communicate on it the ***network***.

9. Let $E(i)$ be a function that changes functions *V*, *I*, *P*. Call *E* the ***evolution*** function.

Explanation: this formalism allows us to model key aspects of agents separately. First, it needs to separate the state of the agent into a cryptographically secured hash-chain part χ and another part that holds arbitrary data *D*. Then we split the process of up-dating the state into two steps, including 1) the validation of new transactions *t* through the validation function $V(t, v)$,and 2) the actual change of internal state *S* (as either χ or *D*) through the state transition functions $\tau_\chi$and τD. Finally, we distinguish between 1) state transitions triggered by external events, stimuli, received through $I(t)$,and 2) a node's internal processing P(x) that also results in calling *V* and τ with an internally created transaction.

Some key properties of the distributed systems are defined as follows:

1. Call a set of nodes in N for which any of the functions T,V,P and E have the properties of being both reliably known and also known to be identical for that set of nodes: ***trusted*** nodes with respect to the functions so known.
2. Call a channel C with the property that messages in transit can be trusted to arrive exactly as sent: ***secure***.
3. Call a channel C, on which the address A_n of a node n is $A_n = H(pk_n)$, where pk_n is the public key of the node n while all the messages include a digital signature of the message signed by sender: ***authenticated***.
4. Call a data element that is accessible by its hash ***content addressable***.

For the purposes of this paper, we assumed there are some untrusted nodes, i.e., independently acting agents solely under their own control, and an insecure channel. We do this because the very *raisond'être* of the cryptographic tools mentioned above can allow the individual nodes to trust the whole system under this assumption. The cryptography immediately is made visible in the state data when any other node in the system uses a version of the functions different from itself. This property is often referred to as a ***trustless*** system. However, as it simply means that the locus of the trust has been shifted to the state data instead of other nodes, we refer to it as systemic reliance on ***intrinsic dataintegrity***. See IVC for a detailed discussion on trust in distributed systems.

B. ETHEREUM

Ethereum provides the current premier example of generalized distributed computing using the Blockchain model. The Ethereum approach comes from an ontology of replicating the data certainty of single physical computer, on top of the stratum of a bunch of distributed nodes using the blockchain strategy of creating a single data reality in a cryptographic chain, but commiting computations, instead of just monetary transactions as in bitcoin, into the blocks.

This approach does live up to the constraints listed above as described by Wood

[EIP-150] where the bulk of that paper can be understood as a specification of a validation function $V_n()$ while the described state transition function $\sigma_{t+1} \equiv \Upsilon(\sigma, T)$ can be understood as a specification of how constraints above are met.

Unfortunately the data-centric legacy inherited by Ethereum from the blockchain model, is immediately observable in its high compute cost and difficulty in scaling.

C. IPS

An analysis about the agent-centric distributed generalized computing system is made subsequently. In this system, the nodes can still confidently participate in the system as whole even though they are not constrained to maintain the same chain state as all other nodes.

In broad strokes: a IPS application consists of a network of agents maintaining a unique source chain of their transactions, paired with a shared space implemented as a validating, monotonic, sharded, distributed hash table (DHT) where every node enforces validation rules on that data in the DHT as well as providing provenance of data from the source chains where it originated.

Based on our formalism, a IPS based application Ω_{hc} is defined as:

1.Call χ_n the ***source chain*** of n.

2. Let M be a virtual machine used to execute code.

3. Let the initial entry of all χ_n in N be identical and consistent in the set DNA$\{e_1, e_2, \ldots, f_1, f_2, \ldots, p_1, p_2, \ldots\}$ where e_x are definitions of entry types that can be added to the chain while f_x is the function defined as executable on M (which we also refer to as the set $F_{app} = \{app_1, app_2, \ldots\}$). Meanwhile, p_x refers to the system properties, which among other coefficients can declare the expected operating parameters of the application being specified. For example, the resilience factor defined as below is defined as such a property.

4. Let τ_n be the second entry of all χ_n and be a set of the form $\{p, i\}$ where p is the public key and i is identifying information appropriate to the use of this particularΩ_{hc}. Note that though this entry is of the same format for all χ_n it's content

is not the same. This entry is named as the ***agent identity*** entry.

5. $\forall e_x \in DNA$ let there be an $app_x \in F_{\text{app}}$, which can be used to validate transactions that involve entries of type e_x. Call this set F_{v} or the ***application validation functions***. (not even a complete sentence)

6. Let there be a function $V_{\text{sys}}(ex,e,v)$, where e refers to the form specified by the entry definition for $e_x \in DNA$. Name this function as the ***system entry validation function***.

7. Let the overall validation function $V(e,\ v) \equiv \vee_x F_v(e_x)(v) \wedge V_{sys}(e_x, e, v)$.

8. Let F_{I} be a subset of F_{app} distinct from F_{v} such that $\forall f_x(t) \in F_{\text{I}}$, there is a t to $I(t)$ that will trigger $f_x(t)$. Name the functions in F_{I} the ***exposed functions***.

9. Call any functions in F_{app} not in F_{v} or F_{I}***internal functions*** and allow them to be called by other functions.

10. Let the channel C be ***authenticated***.

11. Let *DHT* define a distributed hash table on an authenticated channel as follows:

(a) Let Δ be a set $\{\delta_1, \delta_2, \cdots\}$ where δ_x is a set {*key*,*value*} where *key* is always the hash $H(value)$ of *value*. Name Δ as the ***DHTstate***.

(b) Let F_{DHT} be the set of functions $\{dht_{\text{put}}, dht_{\text{get}}\}$ where:

i. $dht_{\text{put}}(\delta_{\text{key,value}})$ adds $\delta_{\text{key,value}}$ to Δ

ii. $dht_{\text{get}}(key) = value$ of $\delta_{\text{key,value}}$ in Δ

(c) Assume $x, y \in N$ and $\delta_i \in \Delta_x$ but $\delta_i \notin \Delta_y$.

When y calls $dht_{\text{get}}(key)$, δ_i will be allowed to retrieve from x over channel X and added to Δ_y.

DHT are sufficiently mature and hence there are a number of ways to ensure property 11c. For our current alpha version, we use a modified version of [Kademlia] as implemented in [LibP2P].

12. DHT_{hc} augment *DHT* is regulated as follows:

(a) $\forall \delta_{\text{key,value}} \in \Delta$ constrain *value* to be of an entry type as defined in DNA.

Furthermore, enforce that any function call $dht_x(y)$ which modifies Δ also uses $F_v(y)$ to validate y and records whether it is valid. It is necessary to notice that this validation phase may include contacting the source nodes involved in generating y to gather more information about the context of the transaction (for the details, see IVC2).

(b) Ensure that all elements of Δ can only be changed monotonically, that is, elements δ can only be added to Δ not removed.

(c) Include F_{DHT} into the functions defined in A.

(d) The sets $\delta \in \Delta$ are allowed to include more elements as defined in A.

(e) Let $d(x, y)$ be a *symmetric* and *unidirectional* distance metric within the hash space defined by H, as for example the XOR metric defined in [Kademlia]. It is of great importance to notice that this metric can be applied between entries and nodes alike since the addresses of both are values of the same hash function H (i.e. $\delta_{key} = \mathrm{H}(\delta_{value})$ and $A_n = H(pk_n)$).

(f) Let r be a parameter of DHT_{hc}, which is a set dependent on the characteristics deemed beneficial for maintaining multiple copies of entries in the *DHT* for the given application. Call r the ***resilience factor***.

(g) Each node is allowed to maintain a set $M = \{m_n, \ldots\}$ of metrics m_n about other nodes, where each m_n contains both a node's direct experience of n with respect to that metric, as well as the experience of other nodes of n. Meanwhile, *n*.Enforce*e*nforce that one such metric kept is uptime which keeps track of the percentage of time a node is experienced to be available. Call the process of nodes sharing these metrics ***gossip*** and refer to IVC3 for details.

(h) Enforce that $\forall \delta \in \Delta n$ each node n maintains a set $V_\delta = \{n_1, \ldots, n_q\}$ of q closest nodes to δ as seen from n, which are *expected by n* to also hold δ. Resiliency is maintained by taking the node uptimes into account and choosing the value of q so that:

$$\sum_{i=0}^{q} uptime\left(n_i\right) \geq r \qquad (2.4)$$

whith $uptime(n) \in [0,1]$.

Call the union of such sets V_δ, from a given node's perspective, the ***overlap list*** and also note that $q \geq r$.

(i) Allow every node n to discard every $\delta_x \in \Delta n$ if the number of closer (with regards to $d(x,y)$) nodes is greater than q (i.e. if other nodes are able to construct their V_δ sets without including n, which in turn imply that there are enough other nodes responsible for holding δ in their Δ_m and hence to have the system satisfied the resilience set by r even without n participating in storing δ). It should be noticed that his these results in the network adapting to changes in topology and DHT state migrations by regulating the number of network-wide redundant copies of all $\delta_i \in \Delta$ to match r according to node uptime.

Name DHT_{hc} as a ***validating,monotonic, sharded*** DHT.

13. According to $\forall n \in N$, it is assumed that n implements DHT_{hc}, which is: Δ is a subset of D (the non hash-chain state data) and F_{DHT} are available to n, even though these functions are not directly available to the functions F_{app} defined in DNA.

14. Let F_{sys} be the set of functions $\{sys_{\text{commit}}, sys_{\text{get}}, \dots\}$ where:

(a) $sys_{\text{commit}}(e)$ uses the system validation function $V(e, v)$ to add e to χ, and if successful calls $dht_{\text{put}}(H(e); e)$.

(b) $sys_{\text{get}}(k) = dht_{\text{get}}(k)$.

(c) See additional system functions defined in B.

15. Allow the functions in F_{app} defined in the DNA to call the functions in F_{sys}.

16. Let m be an arbitrary message. F_{sys} is included in the function $sys_{\text{send}}(A_{\text{to}},m)$ which when called on n from will trigger the function $app_{\text{receive}}(A_{\text{from}},m)$ in the DNA on the node n_{to}. Name this mechanism as ***node-to-nodemessaging***.

17. Allow that the definition of entries in DNA can mark entry types as ***private***. Enforce that if an entry σ_x belongs to this type, then $\sigma_x \notin \Delta$. Note however that entries of such type can be sent as node-to-node messages.

18. Let the system processing function $P(i)$ be a set of functions in F_{app}, which are to be registered in the system as callbacks based on various criteria, e.g.

Notification of rejected puts to the DHT, passage of time, etc.

D. SYSTEMIC INTEGRITY THROUGH VALIDATION

The appeal of the data-centric approach to distributed computing mainly relies on that if you can prove that all nodes reliably have the same data, strong general basis is provided for the proof of the integrity of the system as a whole. In the case of Bitcoin, the χ holds the transactions and the unspent transaction outputs, which allows nodes to verify future transactions against double-spend. In the case of Ethereum, χ holds what ammounts to pointers to machine state. Proving the consistency across all nodes of those data sets is fundamental to the integrity of those systems.

However, because we have started with the assumption (see III A) of distributed systems of independently acting agents, any *proof* of $\forall n, m \in N : \chi_n \overset{!}{=} \chi_m$ in a blockchain based system is better understood as a *choice* (hence our use of the $\overset{!}{=}$), in which the nodes use their agency to decide when to stop interacting with other nodes based on the detection of that the χ state no longer matches. This might also be called "proof by enforcement," which is also appropriately known as a ***fork*** because essentially it results in partitioning of the network.

The heart of the matter is about trusting any single agent involved in the system. In [EIP-150] Section 1.1 (Driving Factors) we read:

I want to provide a system that users can be guaranteed that no matter what individuals, systems or organizations they are interacting with, they can have absolute confidence on the possible outcomes and how those outcomes might come about.

The idea of "absolute confidence" here seems important. We attempt to understand it more formally and generally for distributed systems.

1. Let Ψ_α be a measure of the confidence an agent has in various aspects of the system it participates in, where $0 \le \Psi \le 1$, 0 represents no confidence, and 1 represents absolute confidence.
2. Let $R_n = \{\alpha_1, \alpha_2, \cdots\cdots\}$ define a set of aspects about the system with which an

agent $n \in N$ measures confidence. Name R_n as the ***requirements*** of n with respect to Ω.

3. Let $\varepsilon_n(\alpha)$ be a threshold function for node $n \in N$ with respect to αsuch that when $\Psi_\alpha < \varepsilon(\alpha)$ then n will either stop participating in the system, or reject the participation of others (resulting in a fork).

4. Let R_A and Let R_C be partitions of R where

$$\begin{array}{l} \forall \alpha \in R_A : \varepsilon(\alpha) = 1 \\ \forall \alpha \in R_C : \varepsilon(\alpha) < 1 \end{array} \qquad (2.5)$$

so any value of $\Psi \neq 1$ is rejected in R_A and any value $\Psi < \varepsilon(\alpha)$ is rejected in R_C. Call R_A the ***absolute requirements*** and R_C the ***considered requirements***.

So we have formally separated system characteristics that we have absolute confidence in (R_A) from those we only have considered confidence in (R_C). However, it still remains unclear about how to measure a concrete confidence level Ψ_α. In real-world contexts and for real-world decisions, confidence relies intensively on the vantage point of the agent, set of data at hand, and maybe even intuition. Thus we find it more adequate to call it a soft criterion. In order to comprehend this concept objectively and relate it to the notion conveyed by Woods in the quote above, we proceed by defining the measure of confidence of an aspect α as the conditional probability of it being the case in a given context:

$$\Psi_\alpha \equiv P(\alpha | C) \qquad (2.6)$$

where the context C models all other information available to the agent, including the basic and intuitive assumptions.

Consider the fundamental example of cryptographically signed messages with asymetricasymmetric keys as applied throughout the field of cryptographic systems (basically what coins the term crypto-currency). The central aspect in this context is called $\alpha_{signature}$, which provides us with the ability to *know with certainty* that a given message's real author $Author_{real}$ is the same agent indicated solely via locally available data in the message's meta information through the cryptographic signature

$Author_{local}$. We gain this confidence because it is assumed to be very hard for any agent not in possession of the private key to create a valid signature for a given message.

$$\alpha_{signature} \equiv Author_{real} = Author_{local} \qquad (2.7)$$

The appeal of this aspect is that we can check authorship locally, i.e., without the need of a 3rd party or a direct trusted communication channel to the real author. However, the confidence in this aspect of a certain cryptographic system depends on the context C:

$$\Psi_{signature} = P\left(Author_{real} = Author_{local} \,|C \right) \qquad (2.8)$$

If we constrain the context to remove the possibility of an adversary gaining access to an agent's private key and also exclude the possible (future) existence of computing devices or algorithms that could easily calculate or brute force the key, we might then assign a (constructed) confidence level of 1, i.e., "absolute confidence". Without such constraints on C, we must admit that $\Psi_{signature} < 1$, which can make the real world events quite clear, such as the hack of Mt. Gox in 2014.

We aim to describe these relationships in details in order to point out that any set R_A of absolute requirements can't reach beyond trivial statements about the content and integrity of the local state of the agent itself. Following Descarte's way of questioning the confidence in every thought, we project his famous statement *cogito ergo sum* into the reference frame of multi-agent systems by stating: Agents can only have honest confidence in the fact that they perceive a certain stimulus to be present and whether any particular abstract a priori model matches that stimulus without contradiction, i.e., that an agent sees a certain piece of data and that it *is possible to interpret it in a certain way*. Every conclusion being drawn a posteriori through the application of sophisticated models of the context is dependent on assumptions about the context that are inherent to the model. This is the heart of the agent-centric outlook. What we claim must always be considered in the design of decentralized multi-agent systems, which as it shows that any aspect of the system as a whole that

includes assumptions about other agents and non-local events must be in R_C, i.e., have an a priori confidence of $\Psi < 1$. As for the truth about multi-agene systems, we find little value in trying to force an absolute truth $\forall n, m \in N : \chi_n \stackrel{!}{=} \chi_m$ and we instead frame the problem as:

We wish to provide generalized means by which decentralized multi-agent systems can be built so that:

1. fit-for-purpose solutions can be applied in order to optimize (optimize the fit-for-purpose solutions or optimize the application of contextualized confidences Ψ_α, if it is the latter one, you need to delete "for". If is the former one, you need to use passive voice, "get changed") for application contextualized confidences Ψ_α,
2. violation of any threshold $\varepsilon(\alpha)$ through the actions of other agents can be detected and managed by any agent, such that
3. the system integrity is maintained at any point in time or, when not, there is a path to regain it (see ??).

We perceive the agent-centric solution to these requirements to be the holographic management of system-integrity within every agent/node of the system through application specific validation routines. These sets of validation rules lie at the heart of every decentralized application, which vary across applications based on the context. Every agent carefully keeps track of their representation of that portion of reality that is of importance to them - within the context of a given application that has to manage the trade-off between having high confidence thresholds $\varepsilon(\alpha)$ and a low need for resources and complexity.

For example, two different using cases of transactions need to be considered:

1. receipt of an email message where we are trying to validate it as spam or not and
2. commit of monetary transaction where we are trying to validate it against double-spend.

These contexts have different consequences that an agent may wish to evaluate differently and may be willing to expend differing levels of resources to validate. We designed IPS to allow such validation functions to be set contextually per application and expose these contexts explicitly. Thus, one could conceivably build a IPS application that deliberately makes choices in its validation functions to implement either all or partial characteristics of Blockchains. In light of this, IPS can be understood as a framework that opens up a spectrum of decentralized application architectures, in which Blockchain happens to be one specific instance at one end of this spectrum.

The following section presents what categories of validation algorithms exist and how these can be stacked on top of each other in order to build decentralized systems that are able to maintain integrity without introducing an absolute truth that every agent would be forced to accept or consider.

1. INTRINSIC DATA INTEGRITY

Every application except the low-level routines utilize the non-trivial structured data types. Structured data types imply the existence of a model describing how to interpret raw bits as an instance of a type and how pieces of the structure relate to each other. Generally, this includes certain assumptions about the set of possible values. Certain value combinations might not be meaningful or violate the intrinsic integrity of this data type.

Consider the example of a cryptographically signed message m={*body, signature, author*}, where *author* is given in the form of their public key. This data type conveys the assumption that the three elements, namely *body*, *signature* and *author,* correspond to each other as constrained by the cryptographic algorithm that is assumed to be determined through the definition of this type. The intrinsic data integrity of a given instance can be validated just by looking at the data itself and checking the signature by applying the cryptographic algorithm that constitutes the central part of the priori model of the type. The validation yields a result $\in$ {*true*,

false}which means that the confidence in the intrinsic data integrity is absolute, i.e. $\Psi_{intrinsic}=1$.

Generally, we define the intrinsic data integrity of a transaction type ϕ as an aspect $\alpha_{\phi,\ intrinsic} \in R_A$, which is expressed through the existence of a deterministic and local validation function $V_\alpha(t)$ for transactions $t \in \phi$ that does not depend on any other inputs but t itself.

It should be noticed about how the intrinsic data integrity of the message example above does not make any assumptions about the real author of any message, which is as the aspect $\alpha_{signature}$ from the previous section does. With this definition, we focus on aspects that don't make any claims about system properties non-local to the agent under consideration, which is rooted in the sequence of inferences that constitutes the validity and therefore the confidence of the high-level aspects and integrity of the system in consistent environmental inputs.

2. MEMBRANES & PROVENANCE

Distributed systems must rely on mechanisms to restrict participation by nodes during the process in which the systemic integrity would be compromised without such restriction. Systems where the restrictions are based on the nodes' identity, whether that be as declared by type or authority, or collected from the history of the nodes' behaviors, are know as ***permissioned*** [Swanson15]. Systems where these restrictions are not based on properties of the nodes themselves, which are known as permissionless. In permissionless multi-agent systems, a principle threatening the systemic integrity comes from *Sybil-Attacks*[Douceur02], where an adversary tries to overcome the validation rules of the system by spawning a large number of compromised nodes.

However, for both permissioned and permissionless systems, mechanisms existsexist to gate participation. Formally:

Let $M(n,\phi,z)$ be a binary function that evaluates whether transactions of type ϕ

submitted by $n \in N$ are to be accepted, where z serves as the arbitrary extra information needed to make that evaluation. Call M the *membrane* function, and note that it will be a component of the validation function $V(t, v)$ from the initial formalism5.

Under the cases of Ω_{bitcoin} and $\Omega_{ethereum}$, M ignores the value of n and makes its determination solely on whether z demonstrates the "proof" in proof-of-X be it *work* or *stake* which is a sufficient gating to protect against Sybil-Attacks.

Giving up the data-centric fallacy of forcing one absolute truth $\forall n, m \in N : \chi_n \underline{!} \chi_m$ reveals that the transaction provenance cannot be discarded. Agent-centric distributed systems instead must rely on two central facts about data:

1. it is originated from a source
2. its historical sequence is local to that source.

For this reason, Ω_{hc} splits the system state data into two parts:

1. each node is responsible to maintain its own entire χ_n or ***source chain*** and be ready to confirm that state to other nodes when asked.
2. all nodes are responsible to share portions of the transactions of other nodes and the meta data of those transactions in their **DHT shard** with the meta data includes validity status, source and optionally the chain headers of the source, which provide historical sequence.

Thus, the DHT provides distributed access to the transactions of others and their evaluations of the validity of those transactions. This resembles how knowledge gets constructed within social fields and through interaction with others, which are as described by the sociological theory of *social constructivism*.

The properties of the DHT in conjunction with the hash function provide us with a deterministically defined set of nodes, i.e., a neighborhood for every transaction.

One cannot easily construct a transaction such that it lands in a given neighborhood. Formally:

$$\begin{aligned} &\forall t \in \Delta : \exists \eta : H \to N^r \\ &\eta\left(H\left(t\right)\right) = \left(n_1, n_2, \cdots, n_r\right) \end{aligned} \qquad (2.9)$$

where the function η is mapped from the range H of the hash function H to the r nodes that keep the r redundant shards of the given transaction t (see 12i).

The list of nodes $\eta(H(t))$ allows an agent to compare the third-party viewpoints regarding t, with its own and that of the transaction's source(s). The randomization of the hash function H ensures that those viewpoints represent an unbiased sample. The constraints of the application and the chosen trade-off between costs and system integrity can decide the adjustment of r. These properties provide sufficient infrastructure to create system integrity by detecting nodes that don't play by the rules - like changing the history or content of their source chain. In appendix C, it presents the tool that is appropriate for different contexts in details, including ones where detailed analysis of source chain history is required - for example financial transaction auditing.

Depending on the application's domain, neighborhoods could become vulnerable to Sybil-Attacks because a sufficiently large percentage of compromised nodes could introduce bias into the sample used by an agent to evaluate a given transaction. IPS allows applications to handle Sybil-Attacks through domain specific membrane functions. Because we chose to inherently model agency within the system, permission can be granted or declined in a programmatic and decentralized manner and hence to allow the applications to appropriately land on the spectrum between permissioned and permissionless.

In appendix D, we provide some membrane schemes that can be chosen either for the outer membrane of that application that nodes have to cross in order to talk to any other node within the application or for any secondary membrane inside the application. The latter one means that nodes could join permissionless and participate in aspects of the application that are not integrity critical without further condition but need to provide certain criteria in order to pass the membrane into application crucial validation.

Thus, IPS applications maintain systemic integrity without introducing consensus and therefore (computationally expensive) absolute truth because any single node uses

provenance to independently verify any single transaction with the sources involved in that transaction and 2) because each IPS application runs independently of all others, they are inherently permissioned by application specific rules for joining and continuing participation in that application's network. These both provide the benefit that any given IPS application can tune the expense of that validation to a contextually appropriate level.

3. GOSSIP & WORLD MODEL

So far, an analysis is made only focusing on those parts of the validation function V used to verify elments of χ. However, maintaining system integrity in distributed systems also requires that nodes have mechanisms sharing information about nodes that have broken the validation rules so that they can be excluded from participation. There exist, additionally, forms of bad-acting that do not live in the content of a transaction but in the patterns of transacting that are detrimental to the system, for example, denial of service attacks.

IPS uses gossip for nodes to share information about their own experience of the behavior of other nodes. Informally, we call this information the node's ***world model***. This section describes the nature of the gossip protocols of the IPS and how they build and maintain a node's world model.

we described one such part of the world model, the *uptime* metric and how it is used for maintain redundant copies of entries. we defined a membrane function that determines if a node shall accept a transaction and allowed that function to take arbitrary data z. The major data source is the world model.

More formally:

1. Recall that each node maintains a set M of metrics m about other nodes it knows about. It should be noticed that in terms of our formalism, this world model is part of each node's non-chain state data D.
2. Let m be a tuple of tuples $((\mu, c)_{self}; (\mu, c)_{others})$, which records an experience μ

of a node with respect to a given metric and a confidence c of that experience, both as directly experienced or as "hearsay" received from other nodes.

3. A class of entries stored in χ_n is allowed to be used also as a metric m_w, which acts as a signed declaration of the experience of n regarding some other node. Call such entries *warrants*. These warrants allow us to use the standard tooling of IPS to make provenance based, verifyable claims about other nodes in the network, which propagates orthogonally from the usual DHT methods, via gossip to nodes that need to "hear" about these claims and hence to make decisions about interacting with nodes.

4. $\forall m \in M$ let the function Gwith(m) return a set of nodes important for a node to gossip with defined by a probabilistc weighting that information received from those nodes will result in changing m_{other}.

5. $\forall m \in M$ let the function $G_{\text{about}}(m)$ return a set of nodes important for a node to gossip **about** defined by the properties of m.

6. Define subsets of $G_{\text{with}}(m)$ according to a correlation with what it means to have low vs. high confidence value c:

(a) **Pull**: consisting of nodes about which a low confidence means a need for more frequent gossip to raise a node's confidence. Such nodes would include those for which, with respect to the given node, hold its published entries, hold entries it is also responsible for holding, are close the then node (i.e. in its lowest k-bucket), and which it relies on for routing (i.e.a subset of each k-bucket)

(b) **Push**: consisting of nodes about which a high confidence implies a need for more frequent gossip to spread the information about that node. Such nodes would include ones for which a given node having high confidence is a bad actor, i.e. it directly experienced bad acting, or has received bad actor gossip from nodes that it has high confidence in being able to make that bad actor evaluation.

The computational costs of gossip depend on the set of metrics that a particular application needs to keep the track of the maintenance of the system integrity. For an application with a very strong membership membrane perhaps only uptime metrics are necessary to gossip about to balance resilience. However, this relies intensively on

the priori knowledge of the nodes involved in the application. Applications with very loose membership membranes may have a substantial number of metrics and complex membrane functions using those metrics which may require substantial compute effort. The IPS design intentionally leaves these parameters only loosely specified so that applications can be built fit for purpose.

III.COMPLEXITY IN DISTRIBUTED SYSTEMS

In this section, we discuss the complexity of our proposed architecture for decentralized systems and compare it to the increasingly adopted Blockchain pattern.

Formally describing the complexity of decentralized multi-agent systems is a non-trivial task, for which the more complex approaches have been suggested ([Marir2014]). This might be the reason why there are misunderstandings and unclarity within communities discussing complexity and scalability of Bitcoin for example [Bitcoin Reddit].

In order to be able to have a ball-park comparison between our approach and the current approaches in the application architecture, we proceed by modeling the worst-case time complexity both for a single node $\Omega_{SystemNode}$ as well as for the whole system Ω_{System} and both as functions of the number of state transitions (i.e.,transactions) n and the number of nodes in the system m.

A. BITCOIN

Let $\Omega_{Bitcoin}$ be the Bitcoin network, n be the number of transactions and m be the number full validating nodes (i.e., *miners*) within $\Omega_{Bitcoin}$.

For every new transaction being issued, the signature of every transaction needs to be checked by the given node (among other checks, see. [BitcoinWiki]) and especially check if this transaction's output is not used in any other transaction to reject double-spendings, resulting in a time complexity of

$$c + n \qquad (3.1)$$

per transaction. The time complexity in big-O notation per node as a function of the

number of transactions is therefore:

$$\Omega_{BitcoinNode} \in O\left(n^2\right) \qquad (3.2)$$

The complexity handled by one Bitcoin node does not depend on m the number of total nodes of the system. However, since every node has to validate exactly the same set of transactions, the time complexity of the system as a function of number of transactions and number of nodes results as

$$\Omega_{Bitcoin} \in O\left(n^2 m\right) \qquad (3.3)$$

It should be noticed that this quadratic time complexity of Bitcoin's transaction validation process is what creates its main bottleneck as this reduces the gossip band-width of the network since every node has to validate every transaction before passing it along. In order to still have an average transaction at least flood through 90% of the network, block size and time cannot be pushed beyond 4MB and 12s respectively, according to [Croman et al 16].

B. ETHEREUM

Let $\Omega_{Ethereum}$ be the Ethereum main network, *n* be the number of transactions and *m* the number of full clients within in the network.

The time complexity of processing a single transaction on a single node is a function of the code that has its execution being triggered by the given transaction plus a constant:

$$c + f_{tx_i}\left(n, m\right) \qquad (3.4)$$

Similarly to Bitcoin and as a result of the Blockchain design decision to maintain one single state ($\forall n, m \in N : \chi_n \stackrel{!}{=} \chi_m$,"*this is to be avoided at all costs as the uncertainty that would ensue would likely kill all confidence in the entire system*." [EIP-150]). Every node needs to process the transaction being sent, resulting in a time complexity per node as

$$c + \sum_{i=0}^{n} f_{tx_i}(n,m) \qquad (3.5)$$

that is

$$\Omega_{EtheremNode} \in O(n \cdot f_{avg}(n,m)) \qquad (3.6)$$

whereas users are incentivized to hold the average complexity favg(n;m) of the code being run by Ethereum small since execution has to be paid for in gas and which is due to restrictions such as the block gas limit. In other words, considering that the complexity $\sum_{i=0}^{n} f_{tx_i}(n,m)$ being burdened upon all nodes of the system, other systemic properties have to keep users from running complex code on Ethereum so as to not bump into the limits of the network.

Again, since every node has to process the same set of all transactions, the time complexity of the whole system then is that of one node multiplied by *m*:

$$\Omega_{Ethereum} \in O(nm \cdot f_{tx_i}(n,m)) \qquad (3.7)$$

C. BLOCKCHAIN

Both examples of Blockchain systems above do need a non-trivial computation overhead in order to work at all: the proof-of-work, hash-crack process also called *mining*. Since this overhead is not a function of either the number of transactions nor the direct number of the nodes, it is often omitted in complexity analysis. With the total energy consumption of all Bitcoin miners today being greater than the country of Iceland [Coppock17], it would be a silly mistake to neglect the complexity of the consensus algorithm of Blockchain.

Blockchains set the block time, the average time between two blocks, as a fixed parameter that the system keeps in homeostasis by adjusting the difficulty of the has-crack based on the total hash rate of network. For a given network with a given set of mining nodes and a given total hash-rate, the complexity of the hash-crack is constant. However, as the system grows and more miners come on-line, which increases the networks total hash-rate, the difficulty increases correspondingly in

order to keep the average block time constant.

With this approach, the benefit of a higher total hash-rate x_{HR} is an increased difficulty of an adversary to influence the system by creating biased blocks (which would render this party able to do double-spend attacks). This is why Blockchains have to subsidize mining, depending on a high x_{HR} as to make it economically impossible for an attacker to overpower the trusted miners.

In light of this, there is a direct relationship between the total trusted hash rate of the network and its level of security against mining power attacks. This means that the confidence $\Psi_{Blockchain}$ any agent can have in the integrity of the system is a function of the system's hash-rate x_{HR}, and more precisely, the cost/work $cost(x_{HR})$ needed to provide it. Looking only at a certain transaction t and given any hacker acts economically rationally only, the confidence in t being added to all χ_n has an upper bound in

$$\Psi_{Blockchain}(t) < min\left(1, \frac{cost(x_{HR})}{value(t)}\right) \qquad (3.8)$$

In order to keep this confidence unconstrained by the mining process and the architecture of Blockchain itself, $cost(x_{HR})$ (which includes the setup of mining hardware as well as the energy consumption) has to grow linearly corresponding with the exchange of the value in the system.

D. IPS

Let Ω_{IPS} be a given IPS system, let n be the sum of all public (i.e., *put* to the DHT) state transitions (transactions), let all agents in Ω_{IPS} trigger in total, and let m be the number of agents (= nodes) in the system.

A new entry is put into the DHT, involving a node that is responsible for holding that specific entry, which in our case according to [Kademlia] has a time complexity of

$$c + \lceil log(m) \rceil. \qquad (3.9)$$

After receiving the state transition data, this node will gossip with its q neighbors, which will result in r copies of this state transition entry being stored throughout the system - on r different nodes. Each of these nodes has to validate this entry, which serves as an application with specific logic. Its complexity is called $v(n,m)$.

Combined, this results in a system-wide complexity per state transition as given with

$$\Omega_{IPS} \in O(n \cdot (log(m) + v(n,m)) \qquad (3.10)$$

Now, this is the overall system complexity. In order to enable comparison, we argue that in the case of that the IPS has none loss of generality (i.e., dependent on the specific IPS application), the load of the whole system is shared equally by all nodes. Without further assumptions, for any given state transition, the probability of it originating at a certain node is $\frac{1}{m}$. Therefore, the term for the lookup complexity needs to be divided by m to describe the average lookup complexity per node.

Other than in Blockchain systems where every node has to see every transaction, for the vast majorityvast majority of state transitions one particular node is not involved at all. The stochastic closeness of the node's public key's hash with the entry's hash is what triggers the node's involvement. We assume the hash function H to show a uniform distribution of hash values which results in the probability of a certain node being one of the r nodes that cannot discard this entry to be $\frac{1}{m}$ times r. The average time complexity being handled by an average node then is

$$\Omega_{IPSNode} \in O\left(\frac{n}{m} \cdot (log(m) + v(n,m))\right) \qquad (3.11)$$

where it should be noticed that the factor $\frac{n}{m}$ represents the average number of state transactions per node (i.e., the load per node). Even though this is a highly application specific value, it is an *a priori,* which is expected to have lower bound since nodes have to process at least the state transitions they produce themselves.

The only overhead that is added by the architecture of this decentralized system is the node look-up with its complexity of $log(m)$.

The unknown and also the application specific complexity $v(n,m)$ of the validation routines is what could still drive up the complexity of the whole system. It is indeed conceivable to think of IPS applications with a lot of complexity within their validation routines. It is basically possible to mimic the consensus validation requirement of Blockchain by enforcing that a validating node communicates with all other nodes before adding an entry to the DHT. It could as well only be half of all nodes. Surely, there is a host of applications with only little complexity - or specific state transitions within an application that involve only little complexity. *In a IPS app one can put the complexity where it is needed and keep the rest of the system fast and scalable.*

We proceed by providing real-world use cases and showing how non-trivial IPS applications can be built, which gets along with a validation complexity of $O(1)$, resulting in a total time complexity per node in $O(log(m))$ and a high enough confidence in integrity without introducing proof-of-work at all.

IV.FULLY REDUNDANT vs ZERO REDUNDANT

In a classical blockchain system, such as bitcoin, ethereum, etc., the entire block data is stored by each node. However, none reward is received for saving the data. The redundant blockchain architecture is not suitable for large-scale commercial scenarios.

Imagine if there is a virtual currency that has become the medium of exchange used in people’s daily life, taking Trump as an example who needs 30 transactions a day to buy daily necessities, then the resulting decentralization system needs to incorporate the following features, including low latency, preferably second-level validation, which is almost unrealizable in both bitcoin and Ethereum. Imagine if you're going to buy a hamburger with bitcoins, it will take you an hour to get it, which is unacceptable for ordinary people.

To support small-sum transactions, transactions inevitably require the lowest transaction fees or no fees, which is difficult to realize in both bitcoin and Ethereum.

The average transaction cost of Ethereum is 0.2 US dollars. How to deal with transactions below $ 0.2 dollars? Once a decentralized system is commercially available and the huge transaction data is sufficiently redundant, the storage of every single point can then easily reach the upper limit.

There are lots of systems, such as the EOS with the method of super node, to solve the aforementioned problems. However, but all of the solutions can not overcome impossible triangle---coexist of security (decentralization) and efficiency.

We adopt a zero-redundancy solution which stores all the chain data into DHT. There will only be one formal copy of the data in the system that ensures permanent reservation of all data.

V.SPECIAL POINTS

A. DHT BASED

Instead of trying to repair or improve the traditional blockchain architecture to solve the performance bottleneck and other problems, we are going to fundamentally change the design concepts of traditional blockchain systems, such as Bitcoin and Ethereum. The team has carefully studied DAG, DHT and other technologies that could completely change the traditional blockchain. Finally, we choose the DHT based technology architecture considering that only DHT technology can achieve this without undermining decentralization, which can greatly improve the operational efficiency of the system and have all of the features of the traditional blockchain and even more.

One important reason, of course, is that members are IPFS devoted followers. However, the Tangle design concept of IOTA is not recognized by everyone yet. We believe that DHT will bring the biggest change. Therefore, we get together to speed up the arrival of the revolution.

Compared with traditional blockchain system, DHT Based Decentralized system has the following advantages:

1.UNLIMITED EXPANSION OF BLOCKS GREATLY REDUCES TRANSACTION DELAY.

Since the block data is stored in the DHT and does not need to be broadcast to every node in the system, it is not necessary to consider block expansion bottleneck caused by the network bandwidth. Therefore, the interval block is packaged. Afterwards, its has value is broadcasted. The length of the hash is always fixed, and the network bandwidth required is always fixed. This delay in each transaction is basically the interval between block packing. Blocking interval is controlled in 1 second so that the delay of each transaction is no more than 2 seconds.

2.THE STORAGE EXPANSION OF THE SYSTEM IS NO LONGER CONSTRAINED BY THE UPPER LIMIT OF SINGLE NODE STORAGE CAPACITY.

The storage scalability of the system is no longer constrained by the upper limit of single-node storage capacity

Since each node is no longer required to store a complete block data, the block data is scattered into the DHT network and generates the unique identifier (Hash value) of the data. Each node only needs to store the data partially. Theoretically, the storage capacity of the system can be extended indefinitely and horizontally...

3.MAKE FULL USE OF MORE FRAGMENTED STORAGE.

Low-storage portable devices, such as mobile phones, cannot be used as nodes of conventional blockchain systems. Instead, they are used for DHT-based decentralized systems, low-storage portable devices can be fully involved and serve as a storage node.

B. DATA SAFETY & SECURE NODE

1. SECURE STORAGE NODE

DHT technology itself has data loss prevention and backup mechanism. This mechanism has been successfully running in many decentralized systems for many years, such as Ceph and BitTorrent, which actually not only achieved data security theoretically, but also can scrutinize by time. But as designers of the system, we always need to consider the worst case, even though we know it's almost impossible. Therefore, we design 100 security nodes on the basis of DHT to deal with the worst case. Once the data in DHT is incomplete due to some factors, it will be synchronized from the security node in real time.

The security nodes serve as the last barrier threatening the system stability and the data security. Undoubtedly, the security nodes need very professional storage device. Therefore, it is necessary to set up reasonable incentive and punishment mechanism. Establishing security nodes is also a useful form of mining. At the same time, the qualification to establish the security node is shared by the community. Each participant can establish a security node as long as the hardware device to satisfy the basic requirements.

Although we preset 100 security nodes, the nodes are not always fixed, which are likely to switch at any time. Due to fixed safety node mining reward, if there are too many participants, the benefits of each node will inevitably decrease. Nodes with higher marginal costs will be naturally withdrawn from the security network. As the security node gradually withdraws, the mining revenue of each node will increase. The mining revenue will increase again and then there will be newly-added security nodes, which will eventually cause the number of security nodes to fluctuate around 100.

As to why the number is 100 instead of 50, the more the security nodes there are, the better performance there would be. However, security nodes need a reasonable incentive mechanism. The coin used to motivate actually has its limit the reward is

also required by other forms of mining. Therefore, the result of our calculation is that we can have almost 92.368 security nodes which we take an integer 100. We came to a conclusion of 100 that is valid, not because Nicolas always scores 100 points.

2. SECURE COMPUTING NODE

C. ANONYMOUS NETWORK & RELAY NODE

Up to now, many people still believe Bitcoin is anonymous. Actually, bitcoin cannot be completely anonymous at the network level. Zcash\Manero\Zencash and other currencies choose to set an anonymous network on the basis of the original blockchain network to ensure the anonymity of transactions and accounts. However, the establishment of an anonymous network requires enough security nodes.

Traditional and already-worked anonymous networks are Tor, i2P, etc., which are time-tested and can achieve certain anonymity. However, there are also many problems. Tor and i2p have been proved to be unable to achieve 100% anonymity. Minor operational errors will result in information leakage. Meanwhile, the using process is very complicated.

Therefore, we launched the anchor project, focusing on anonymity and ease of use, improving anonymity while simplifying the use for ordinary users. Fortunately, EDC has a member from the United States Navy trials who participated in the development of the TOR project. Besides, we have studied parts of the codes of I2P and Orchid Networks in details. We are sure that our solution is the most anonymous and easiest to use. However, this project is still under development and is not currently in use. Although we have already know how to develop such a project, it will still take some time.

D. REGULATION & DOUBLE SECRET KEY

Difficulties in the supervision of digital currencies have always been a major

obstacle hindering the development of the industry. Complete liberals hope that they will not be regulated and will be free from any constraint. In contrast, the government wants to well aware of every citizen. Therefore, seemingly, there is an irreconcilable conflict between the two. However, we believe that the rights should be given to the ordinary people themselves about whether to be regulated or protected, or be completely free which instead means not be protected at all. Based on such an idea, we designed a double-key mechanism for the system.

The double-key mechanism means that there are two keys for each account. The key with all the functions is called the operation key. By all functions here, they not only include initiating a transaction, signing the transaction, but also checking account status at any time. The other key called review key only has the function of querying account status.

In this case, if users choose to be supervised, users will give the review key to the regulatory authorities of the government so that the supervision department can check the status and changes of the account in real time. However, law enforcement can only find all the people off the line. The change of the account can only be used as evidence. If the user wants to have complete freedom, the user just needs to keep his two private keys from authority.

Some might argue that accepting supervision as a blockchain system is a violation of the ideal or that regulation is an integral part of the industry's progress. In this case, the ordinary users are endowed with the power of choice.

E. PRIVACY & ZERO KNOWLEDGE PROOF

Privacy has always been a serious issue for blockchain practitioners because there is really no good solution. After comparing the broadcast protocol and zero-knowledge proofs, we believe that zero-knowledge proof serves as a good solution. Zcash adopted zero-knowledge for the first time in the field of digital currency. EDC has told us many times how excited he was the night when Zcash

broke through 5BTC. Privacy is never the need for just a small number of people.

F. PARALLEL DOUBLE COIN CURRENCY SYSTEM

The basic function of the currency is transaction medium and value scale, while the basic function of assets is to maintain and increase value. We designed a parallel dual-currency economic system, stable currency BUT and digital asset IPST. This value stability is not like DAI, the anchored currency, nor USDT which is endorsed by the legal currency but achieved through a transparent issuance and destruction mechanism to strictly control the currency supply. There is no difference between IPST and the ordinary digital currency. It has both use value and price value. Since the total amount is limited, and there will be no issuance or destruction. IPST can be only used as a kind of digital asset and a hard currency in times of crisis.

VI.STORAGE NETWORK

Modularization is the basic principle of software development, which is considered at the beginning of the system design. The entire system is divided into storage network (DHT storage network, secure storage network), computing network, and pre-anonymous network. The storage network is divided into DHT storage network and secure storage network. Logically, DHT storage network can be further divided into block data storage network and business data storage network.

A. DHT STORAGE NETWORK

1. BLOCK DATA STORAGE NETWORK

The block data is stored in the DHT storage network. Actually, storage network is not physically separate. We just made a distinction on logic. Even the storage network, computing network and the pre-anonymous network are not completely

physically separate. They can exist on a machine at the same time.

2. BUSINESS DATA STORAGE NETWORK

One of the main reasons why there has been no large-scale commercial use in the decentralized systems is that they cannot carry a large amount of business data. These data cannot be stored in the blockchain, nor on centralized storage, which is against its original intention of decentralization. The DHT-based decentralized storage network solves these problems perfectly. It not only prevents the data storage loss or tampering but also ensures the anti-censorship of the data. The data no longer has any position. As long as the data is created, it will be there forever. The position of the data depends on the reader rather than the censor.

B. SECURE STORAGE NETWORK

The secure storage network consists of approximately 100 secure storage nodes. Ever since the creation of the block, all the block data is stored in each secure storage node. There is peer-to-peer communication between the secure nodes, security nodes and DHT storage networks, security nodes and client. When a data segment or data block is lost in the DHT storage network, the secure storage network will take over the DHT storage network to provide temporary services and simultaneously restore the data to the DHT network. Subsequently, the sever side will be switched after the restoration is completed.

VII.COMPUTING NETWORK

A. CONVENTIONAL COMPUTING NETWORK

Any system is composed of storage and computing. A calculation is actually logical processing. Our network also has great innovation. For a world computer like

Ethereum, every piece of code will run on all machines in the system. Although the maximum security (decentralization) is ensured in this way, it does not make full use of computing resources. Ethereum has solutions such as fragmentation and status channels. However, we do not think this is the ultimate solution.

The decentralized world needs a new computing framework that ensures security without wasting extra resources. DHT is a perfect solution to the problem that security and efficiency cannot coexist in the storage world. However, there is no mature and stable solution in the computing field. The team carefully studied distributed computing frameworks such as MapReduce, Storm, and Spark and gained the inspiration. In fact, as long as reasonable incentive mechanisms are added to these distributed computing frameworks, the concurrence of the diversity of the clients should be considered in the computing network. It will greatly improve the utilization efficiency of system computing resources on the premise of ensuring security (decentralization).

B. SECURE COMPUTING NETWORK

Undoubtedly, this concept has not yet been widely validated. We have also studied a lot of blockchain projects related to fog computing. Therefore, we think that we need to deploy a secure computing network in addition to the decentralized computing network, which consists of 100 secure computing nodes with each secure computing node retaining all smart contracts and logical application code. To prevent problems in the zero redundancy decentralized computing network, all contracts and logic codes are synchronized from the secure computing network, thus enhancing the robustness of the system.

VIII.CURRENCY SYSTEM

We have adopted a completely new dual currency system because Edison, the designer of this economic system, a well-known economist and sociologist do not

agree with the definition of currency in traditional economics. The most important point is that currency does not possess the function of media trading and value storage at the same time. The most essential function of currency can only be transaction medium, while and the storage of value is the function that only assets can possess. Therefore, we have designed two currencies in the system. The currency used as the trading medium is BUT (Block Universal token). The BUT itself is a stable currency, the stability of which is not achieved by anchoring the currency but by adjusting the supply of money. The other currency, IPST, is designed for value storage. Any item that serves as value storage can only be an asset. Besides, only the items that have limited supplies (or are generally considered to have limited supplies) have value storage capabilities.

A. BLOCK UNIVERSE TOKEN(BUT)

In the decentralized world, there has never been a greater need for a stable digital currency than this moment, with stability instead of fixed, which has been criticized by some of the so-called scholars for the excessive speculation of digital money, but not a global digital currency that penetrate into daily life. Fortunately, the underlying architecture of IPS makes it possible for the large-scale application of decentralized systems. It is only in this case that can make the emergence of a universal digital currency in the bit world possible.

In the mechanism design of BUT, we have not only designed a new currency supply adjustment mechanism but also designed a brand new credit system based on the trading day. BUT aims to become a daily digital currency and a universal digital currency which only has transaction medium functions with self-regulate, and also has a reliable credit system. For details, please refer to our currency whitepaper. Believe it will give you a bright feeling.

B. B.INTER-PLANETARY SYSTEM TOKEN(IPST)

IPST is the fuel that drives the normal operation of the IPS system. It is also a digital asset with value storage capabilities. Because it has both price value, usage value and limited total amount, it will never be issued additionally. For more details, please refer to our currency whitepaper.

IVV. DEEP THINKING IN PHILOSOPHY

Someone argues that the so-called decentralization of blockchain is nothing more than a new center, which is recognized by the Edison. However, the new center is an open, transparent, technology-supported center for all participants. Currently, it is still a pyramid structure, of which the top layer is a new center with technology as the core. The new pyramid is built from the bottom to up. The will of the lower level determines the shape and composition of the top. On the contrary, the traditional pyramid is a top-down one, the top level of which determines the form of the lower layer. At the same time, the new pyramid is completely created by code and performed by machines. Because the code is law, the code is always rational and the real rule of law. On the contrary, the traditional pyramid depends on the rule of people at every level, which inevitably leads to the problem of corruption because of human nature. However, the code will never be corruptive, but only keep running in accordance with established logic.

VV. DEEP THINKING IN ECONOMICS

We have always believed that Bitcoin is not a digital currency but a digital asset. Because the basic function of the currency is to serve as the transaction medium and value scale, while the basic function of assets is to maintain and increase value. People need assets as well as currency in their daily lives. We believe that the system can only be pushed forward with decentralized system currencies and assets as gas together to push the system continuously forward. However, the underlying economic logic of currency and assets is completely different and must be issued separately.

Articles or charts comparing Bitcoin to gold is already very common. Obviously, all options of Bitcoin have indicated higher value than gold. The end of the gold standard monetary system is not merely because of the difficulty of carrying higher circulation costs and cutting apart but more because the total amount of gold is limited. Although the gold trading volume in London gold trading market far exceeds the underground gold reserves, the paper gold can be manufactured at will and the precise total storage of gold is still unknown, in the public's perception, the total amount of gold is limited. Any goods with a limited total amount can only be an asset which may be used as a currency in a very short time, but definitely not in a long-term.

The historical evolution of gold to credit currencies, correspondingly to the digital currency, Thethe digital assets such as Bitcoin will inevitably transfer the functions of the transaction medium by a new global digital currency with stable and adjustable supply, only retaining the digital assets attributes, as well as the the hard currency function in critical time. Given such economic understanding, we designed a parallel dual-currency economic system, stable currency BUT and digital asset IPST. This value stability is not like DAI, the anchored currency, nor USDT which is endorsed by the legal currency but achieved through a transparent issuance and destruction mechanism to strictly control the currency supply. There is no difference between IPST and the ordinary digital currency. It has both use value and price value. Since the total amount is limited and there will be no issuance or destruction, IPST can only be used as a digital asset and a hard currency in times of crisis.

VVI.USE CASES

A. CONTENT INDUSTRY

The centralized content platform controls a significant internet traffic, manipulating the content of the users based on its own interests rather than the quality of the content itself. The quality here is a comprehensive consideration of the value of the content itself and user demands. At the same time, the centralized content

platform can always arbitrarily delete and modify the content of the creator from its own position. It will be hard for the creator to have 100% control of the content once it is written and published on the Internet. In some countries that have no freedom of speech, all the speech of the public needs to be in the position of some minority. Meanwhile, all the statements are supervised, censored, controlled, modified, and deleted. A decentralized content platform will always be neutral, giving users maximum freedom.

At the same time, in the traditional centralized content platform, high-quality content creators cannot directly monetize high-quality works. It often needs free value output first, waiting for fans to accumulate to a certain extent and then realizing the flow through e-commerce or advertising. Unfortunately, there are many specialists in one field but few all-rounders in the world. A content creator or creative team is required not only to create high quality content, but also to consider the e-commerce or advertising-related issues, so accordingly cannot focus on content creation.

B. B.GAME

In the traditional centralized gaming platform, ordinary users only have the right to use the virtual property, not ownership. It is very common that the user account is frozen and deleted by the game developer. All the virtual properties in the decentralized game platform is protected by the asymmetric encryption algorithm. The private key represents the sole ownership and usage right of the virtual property in the game. Thus, it can be assumed that players can really have everything in the game for the first time.

At the same time, in traditional games, the transaction and conversion of virtual assets in different games is a serious problem because the centralization of the system will inevitably cause data isolation between different game systems. The liquidity only has solid state mobility, while the virtual property in the decentralized game based on the blockchain has a gaseous liquidity.

For those traditional game developers, it is the flow distribution and channel

rather than the game itself that is more important. Decentralized game system has a very strong bootstrap. Only reducing the intermediate link and returning to the quality of the game itself will make the whole game industry enter a virtuous cycle.

As the globalization becomes stronger and stronger, the users of a game may spread all over the world. Considering that currency of each country cannot be applied universally, the trade in the game has an urgent need to have a transaction medium, which is not controlled by any country with low transaction cost and high efficiency, accordingly result in low cost.

C.SHARING OF POWER

The era of big data has already come. To mine gold from data, corresponding centralized calculation support is necessary. The Cloud Center, Hadoop cluster, and Spark cluster are very expensive. In light of this, it is not worthwhile for users to build their own cluster who have no continuous demand for power but just deal with large numbers of computing tasks in short time. It is difficult to ensure the security of the data through the use of cloud service vendors' computing services. The decentralized power-sharing platform can just fill in the blanks to not only reduce the calculation cost but also ensure the security of the data.

The new generation of fog computing platform with incentive mechanism will undoubtedly be a strong competitor for AWS and Alicloud. Besides, it is superior to the centralized cloud computing platform in terms of cost, efficiency and data security. Confidentiality and security of data are the greatest advantages of decentralized computing platform and also are viewed as the highest risks of the centering cloud computing platform. In the foreseeable future, decentralized fog computing platforms will become mainstream.

D. VIRTUAL ASSETS CLAIMING SERVICE

In the era of information Internet, we only have the right to use and have no

ownership over all bit assets. However, in the era of value Internet, we can have unique ownership of all virtual assets. Through asymmetric encryption algorithms, such kind of ownership is protected. The private key is the only certificate for ownership and use rights. The virtual economy supported by value internet will fully grow into a brand-new empire, a complete competitive relationship with the real economy of the physical world, and will inevitably overcome the real economy. The ownership certification of virtual assets serves as the cornerstone of this new empire.

E. CURRENCY

IPS, a high-performance and high-security platform, has made daily decentralized digital currency possible. Because currency needs high requirements for strong security, the high stability of service and timeliness in transaction processing, only IPS, a platform that combines high security and high performance can bear world-class applications such as decentralized digital currency system.

F. DECENTRALIZED STORAGE SERVICES

Decentralized data storage is not suitable to adopt blockchain solutions because the cost of data storage is too high. IPS DHT-based data storage services make data decentralized and anti-censorship and meanwhile have high security and low redundancy.

The biggest advantage of decentralization is the anti-censorship of data storage. Without a center, there will be no mandatory control, and the sof data will be much higher.

VVII.OUR MISSION

A. EFFICIENCY SHOULD BE IMPROVED WITHOUT SACRIFICING DECENTRALIZATION

Decentralization is the premise of blockchain as value Internet. Any high efficiency at the expense of decentralization is undesirable in the era of value Internet. The perfect combination of Distributed Hash Table and Blockchain can achieve the coexistence of decentralization (security) and efficiency (high TPS).

B.IT SHOULD BE THE USERS WHO DECIDE TO ACCEPT THE SUPERVISION OR NOT.

Bitcoin, a digital currency with completely public ledger, does not have real anonymity and cannot truly protect users' privacy. At the same time, Zcash, a completely anonymous currency certified by zero knowledge proof, is highly resistant to censorship but difficult to comply with regulations. The dual secret keys are designed to protect user privacy and meanwhile can also make the supervision optional and hence to give the right to be supervised back to the common currency holder.

C. BOTH COMPUTING AND STORAGE SHOULD BE DECENTRALIZATION.

Existing blockchain architectures, such as Ethereum, can decentralize computing. However, the storage of massive business data is still centralized. Therefore, there is a great risk of data loss. Besides, the data storage is not resistant to censorship. In the real value Internet, the storage of business data needs to be decentralized.

D.THE DECENTRALIZED NETWORK SHOULD BE COMPLETELY ANONYMOUS.

The decentralized networks should also be anonymous at the network level. Only in this way can the privacy of users be completely protected. Therefore, any decentralized service requires a pre-layered anonymous network to protect user data security.

E.THE ECONOMIC SYSTEM SHOULD MEET THE DEMAND OF BOTH THE MEDIUM OF EXCHANGE AND THE STORE OF VALUE.

The tokens of any economic system should satisfy the requirements of exchange media and value storage simultaneously. However, these two attributes are contradictory in the long run. Therefore, the best solution is to design a parallel dual token mechanism.

VVI. ACKNOWLEDGMENTS

We would like to show our gratitude to Jason Yip for his review of this paper.....