LAMBDA PROJECT

A Blockchain Infrastructure Providing
Unlimited Storage Capabilities

Lambda

lambda.im

Lambda

ABSTRACT

Lambda is a fast, safe, and scalable blockchain infrastructure project, which provides decentralized applications (DAPPs) data storage capabilities with unlimited scalability and fulfills services such as multi-chain data co-storage, cross-chain data management, data privacy protection, provable data possession (PDP), and distributed intelligent computing through logic decoupling and independent implementation of Lambda Chain and Lambda DB. In addition, as a blockchain infrastructure service, with the use of Sharding technology, Lambda Chain provides the capability of processing millions of requests per second (RPS), which increases with the expansion of the system scale; with the use the sub-chain technology, it provides future-oriented technical service capabilities with unlimited scalability.

OVERVIEW

In the past few years, the blockchain technology has flourished. By means of blockchain ledger technology and Proof of Work (PoW), human history witnessed for the first time that the consensus of decentralization has been reached among a large population. The subsequent appearance of bitcoins and other digital encrypted coins enables point-to-point value exchange issues to be perfectly solved. Ethereum, which was born afterwards, adopts Turing Complete smart contract technology to provide a convenient and accessible platform for the development of DAPPs, followed by the issuance of multiple tokens based on Erc20. The exchange of tokens lowers the cost of collaboration and value exchange, reduces the price cost to form a network effect in the project, and accelerates the development of the project.

Blockchains communities are developing fast and booming. However, so far, many DAPP projects have been still confronted with the risk of being unable to be implemented. The blockchain projects which have been implemented by now are bitcoin, and issuance of Token and Initial Coin Offering (ICO) based on Ethereum. Many blockchain projects in other fields failed to be implemented due to the incomplete infrastructures, which is represented in the following aspects:

The first one is the overall throughput of the blockchain system, namely, transaction per second (TPS) issue. The bitcoin network can process 7 transactions per second only and Ethereum can process 15, which is also far from the expectation since the processing capability of the centralized financial system is ten thousands of to millions of TPS. Projects such as EOS and Tron have been focused on solving the TPS issue with Delegated Proof of Stake (DPoS), which is still in progress. According to the existing testing data, there is a big gap between the implementation plan of super node mechanism and corresponding expectations. When promoting the assumption, EOS' target was a million of TPS, which turns out to be thousands of TPS in reality according to the current testing results. Moreover, general distributed systems can be scaled out, but the super node mechanism fails to let the throughput extend with the system scale. Algorand, Cardano, and Dfinity adopted Verifiable Random Function (VRF) consensus algorithms to resolve the transaction speed issue. On a short view, since VRF algorithms rely on the assumption of a strong synchronization network, it is unable to properly handle the forks caused by network latency.

The second one is data storage issue. As is known to all, the essence of Internet is the connection between human and information no matter for centralized applications or DAPPs. Without data, the Internet cannot work because itself is an inter-operation process based on data. However, today's DAPPs are deficient of the capability to implement data. Many DAPP development teams lacking of technical common sense misunderstand the blockchain system as a decentralized distributed database or even a cloud "magic database". However, in fact, all existing public blockchains are by nature decentralized distributed ledgers, which are simple linear lists of transaction records storing addresses of the vendors and purchasers, and transaction amount only. In the situation where data are unavailable, the actual application scenarios of blockchains are limited to a large extent. Electronic dropboxes provided by projects such as Sia, Storj, and MaidSafe were not aimed at accesses of programs, which are of comparable lower value. InterPlanetary File System (IPFS) and FileCoin projects have been dedicated to providing a distributed file storage plan, which is a meaningful attempt and exploration. But just as what had happened in the internet sector, files are unformatted data suitable for overall review only and hard to be inquired or accessed by programs. Large-scale data exchange and applications require a storage system or database system based on formatting or semi-formatting objects to enable accesses to object data, KV data, table data, and relational data. In addition, IPFS is an early design, whose concept was inspired by P2P network projects such as Kademlia, which is the reason why its MainChain project FileCoin has not developed as expected.

The third one is “isolated data island” and “isolated asset island” issue. In the earlier development stage of computer technology, data and applications are shared within an enterprise or even a department in an enterprise, lacking of interactions. Around 2008, service–oriented architecture (SOA) and Enterprise Service Bus (ESB) were introduced to enable inter–connection among data, achieving interoperability of data. In the current stage, inter–connection and verification capabilities are deficient among various blockchain systems. Although bitcoin side–chain projects such as RSK promised to enable assets transfer capabilities in the bitcoin sector and Polkadot and Cosmos have been also making more attempts in this sector, these projects are still in progress by now. The fourth one is the exploitability and accessibility issue. The bitcoin system adopts the script system with restricted functions to provide limited programmable support. Followed with the appearance of Ethereum, Ethereum Virtual Machine (EVM) has become de facto standard by providing a Turing Complete programming plan to compile smart contracts, based on which data are operated on the chains. However, Sodility language and other new blockchain languages require learning curves and cost, and many mature development technologies such as SSDLC and unit test cannot be directly implanted to the blockchain sector, which results in cer–tain issues in the exploitability of blockchain. Moreover, as assets are being directly programmed in the blockchain system, bugs in the program will definitely lead to asset loss, for example, projects such as BEC and SMT both exposed contract bugs. So far, no breakthrough has been made in the sectors such as virtual machines and formalization verification.

For the first three of the four issues mentioned above, Lambda project has been trying to provide a new solution to separately save different types of data in distinct chains and blocks. It is devoted to providing an infrastructure for DAPPs, on which it intends to offer a series of infrastructure capability services including storage with unlimited scalability, high throughput computing, and fast network transmission, making it easy for DAPPs to fulfill data generation, transmission, storage, retrieving, and computing. Moreover, the project is featured with flexible data structure, powerful programming interface, scalable system, and atomic operation. In the long run, it will make every effort to explore data index technology, multi–chain data co–processing, cross–chain data management, privacy protection, provable storage possession, distributed smart contract (storage process computing), and inquiry optimization technology.

Developers are allowed to freely use Lambda project codes and modify them under Apache License while users are permitted to download binary files of Lambda and deploy them in the private environment without connecting to the extranet. Under this circum–stance, Lambda project equals to a distributed database with automatic audition function and unchangeable audit data, which can be applicable to the settlement between banks, penetrating audit, and various financial scenarios. People can also choose to join in Lambda cloud network to be one part of the Lambda cloud and obtain our native Cryptocurrency: LAMB. With the development of Lambda project, an increasing number of nodes will be appealed to the network to ultimately form multiple self–organized and self–managed data storage clusters on the cloud covering hundreds of thousands of nodes, which can be simply used for data storage and inquiry of DAPPs through Application Programming Interfaces (APIs). What needs to be specially pointed out is that since it does not need to undertake the extremely high organization cost of decentralized Infrastructure as a Service (IAAS), this decentralized data cluster system is born with high cost–effectiveness and capabilities such as off–site disaster recovery and inter–continental data sharing, which preliminarily establish a decentralized IAAS with infrastructure capabilities in storage, computing, and network bandwidth. Taken the value exchange system provided by the blockchain network into account, Lambda will turn out to be a global infrastructure network with boundless space for imagination.

From the very beginning, Lambda project optimized the design for Internet of Things (IoT) and artificial intelligence (AI). As you know, IoT data are various metrics, which are magnanimous and strictly increasing by time with simple structures, and the AI sector has to provide a large number of incentives for the supply and annotation of data. In the future, these magnanimous data can be computed and processed on the decentralized infrastructures only. For IoT, Lambda project provides Agent and CEP technologies to support its data going on the chain, while for AI, it adopts currency incentives to assess the value and reach a consensus of the data created in the sector to build a great data exchange market, and therefore, push forward the development of IoT and AI.

Lambda

TABLE OF CONTENTS

I.INTRODUCTION TO LAMBDA PROJECT (DESIGN OF CHAIN)

1.1 Project Overview

Lambda project is dedicated to providing a data storage infrastructure for blockchains and DAPPs and based on which it intends to offer storage and access capabilities of the decentralized cloud database. Moreover, based on scale-out and Sharding technology, it enables high-speed transactions; based on the trunk chain technology and cross-chain transaction verification, it enables cross-chain transactions, and data access and verification within the system; based on the Bridge Control Protocol (BCP), it enables the cross-chain communications with other chain systems. By providing a series of infrastructure capability services including block storage, file storage, object storage, KV storage, and table storage, which can be extended without limit, and fast network transmission - remote synchronization (rsync), Lambda makes it possible for DAPPs to perform data generation, computing, transmission, storage, and retrieving in an easy manner. Moreover, attribute encryption and agent encryption technologies applied in the project protect data privacy. Still, Lambda project is featured with flexible data structure, powerful programming interface, and efficient backup.

With the development of Lambda project, an increasing number of service capabilities such as distributed cache, distributed memory-sharing computing based on non-volatile memory (NVRAM), distributed relational database, and distributed MapReduce, will join in the network as parallel sub-chains to provide external infrastructure services through the platform of Lambda. Ultimately, a self-organized and self-managed data management system covering tens of millions of nodes will be established on the cloud, where all DAPPs can easily use APIs to save and inquire data. What needs to be specially pointed out is that since it does not need to undertake the extremely high organization cost of decentralized IAAS, this decentralized cloud database is born with high cost-effectiveness and capabilities such as off-site disaster recovery and intercontinental data sharing, which preliminarily establish a decentralized IAAS with capabilities in storage, computing, and network bandwidth. Taken the value exchange system provided by the blockchain network into account, Lambda will turn out to be a global infrastructure network with boundless space for imagination.

As a decentralized blockchain data infrastructure, Lambda does not benchmark against performance metrics in the centralized database. Due to the peer-to-peer distributed networking, Lambda cloud database is limited to several application scenarios for now. Thanks to Lambda’s advantages in open source, community governance, and verifiable economy model and mutual-trust mechanism, for today’s technicians, it can be used as a KV database (such as redis), document database (mongodb), and time series database (TSDB) - druid, starting from backup and filing scenarios, to save unchangeable data. In terms of business scenarios, Lambda is more suitable for circumstances where transactions are performed frequently and where a large quantity of data flow in and out with small variations. Lambda cannot only be used to save IoT and AI data but also KV data, log data, metrics and Event data, and feed stream data. Thus, Lambda cloud database can be applied to the back-end storage of data for decentralized services such as decentralized lotteries, low-frequency gambles, video websites, feed stream applications, blogs,

1.2 Why We Need Lambda

The development of Internet applications requires a database system

Nowadays, Internet applications have penetrated into everyone’s life but normal users seldom sense the computer science behind. In fact, Internet relies on the development of computer technologies since its emergence, especially several key technologies such as HyperText Transfer Protocol (HTTP), Web servers, and database software.

In 1970, Edgar F. Codd from IBM Research Laboratory published a paper themed “A Relational Model of Data for Large Shared Data Banks” and created the history of database software, and therefore has been called “the father of relational database” ever since. In 1977, Larry Ellison founded Oracle and researched and developed a business database software - Oracle. In 1989, Tim Berners-Lee developed HTTP, Web server, and browser in the European Laboratory for Particle Physics (CERN). Based on the technologies mentioned above, Internet officially kicked off. As time goes by, the user group of Internet has been expanding to the every corner of the world and technologies have been continuously updating. No matter how they are changed, the development of Inter-net applications still requires a database system for data storage.

DAPPs on the blockchain require a decentralized data storage plan

With the development of blockchain technologies, the concept of decentralization has been deeply rooted in the heart of people. However, as DAPPs can save data in the centralized Internet data center (IDC) only during the software development and running, in essence, it is still a centralized system. Traditional database systems are usually managed and maintained by single institutions which have the supreme authorities over the entire database. This mode is not suitable for data management between institutions where a mutual trust has not been completely built, which stands out especially in the Internet application environment. As a decentralized, unchangeable, traceable, and multi-party maintained new distributed data management technology, the blockchain is tailored for the effective data management under these non-mutual-trust scenarios. Due to the significant differences between the old and new data management architectures, the blockchain data management technology must be innovated rather than copying the original technology.

Essentially, Lambda project is a decentralized database system which provides data storage and management services for DAPPs by fulfilling authorization, encryption, and communications among non-mutual-trust institutions with blockchain technologies. As far as we are concerned, only decentralized database can satisfy the data storage requirements of DAPPs.

Before the Lambda project, it is difficult to develop a DAPP and application chain because the public blockchains cannot provide a large number of services for data storage and retrieving, and DAPP needs to process data by itself, which is no different from the situation in 1990s when programs were developed to operate files.

Current blockchain storage plan is not a good choice for DAPP

For current decentralized blockchain applications which have to select targeted storage plans for data, the following decentralization choices are available:

Saving all content to the blockchains;

At present, there are about hundreds of DAPPs running on the Ethereum, enlarging the Ethereum ledger size to more than 100 GB, which requires each node has such a large storage space. It is estimated that thousands of applications will be established and the ledger required by each machine will be increasing by time. In the future, with the withdrawal of users lacking of disk memory space, Ethereum will gradually become centralized. Even worse, its transaction speed is 15 TPS, which is far too slow for a user to wait during a transaction since people can accept the waiting of 1minute for money transaction but not for launching a web page.

Point-to-point file system, such as IPFS

IPFS allows file sharing in the computers on the client and integration to the global file system. Based on BitTorrent protocol, to share a file with others, IPFS requires you to download it to your computer first and let others to download as required and Hash of the file contains IP addresses of both the downloader and publisher. Besides, the more frequently the file is downloaded, the much more bandwidth is provided and the faster the download is. But IPFS also has certain defects, for example, if you want to share a file, you have to keep online until at least one person interested in the file downloads it successfully. What's more, IPFS provides static files only, which cannot be modified or deleted after being uploaded, and it does not support any content-based search. AKASHA, a decentralized social application achieved through IPFS, cannot be downloaded after sending a message until the peer node receives it.

Distributed cloud file storage, such as Storj, Sia, and Ethereum Swarm;

Distributed cloud file storage can remove certain limitations on IPFS. From users' perspective, it is similar to cloud storage service such as Dropbox and the difference lies in that the content is saved in the personal computer offering the hard disk space for rent rather than in a data center. There are a lot of such kinds of projects including Sia, Storj, and Ethereum Swarm, in which users can share a file by simply uploading it to the cloud rather than keeping online all the time. These storage plans are featured with high reliability, fast access, and abundant storage space. However, they still provide static files only and cannot be searched with content nor directly accessed by programs.

The fast development of DAPPs requires high-speed data infrastructures with high availability.

In the past decade, informationization and digitization storms swept the world. Breaking up the territory boundary, mobile Internet

urged almost all services to be online. Many systems expanded from hundreds of thousands of users to tens of millions of users even hundreds of millions users. E-commerce platforms such as Taobao and Amazon provided services for tens of millions of customers in the peak hour, which required tens of thousands of servers in data centers all over the world supported by back-end services with high availability and low latency. In the centralized system architecture, the high availability was at the expense of expensive hardware, such as the ExaData system in Oracle. Nowadays, with the booming of blockchains, deep penetration of the concept of decentralized Internet, and continuous development of DAPPs, it is undoubtedly that the requirements on the availability of background database systems are increasingly high. That is to say, a cloud database system supporting continuous growth of data and providing a highly scalable platform is demanded. For the future DAPPs, high availability is one of the most important requirements because even the lightest system breakdown will result in huge economy loss and affect customers' trust. Even though the decentralized network can provide cheap PCs, mobile computing power, and IoT equipment only, we still need to establish a data infrastructure system with 100% high availability based on these unreliable devices.

Lambda project is a distributed data storage system without any central node, whose computing capability is relying on the computing, storage, and bandwidth capabilities shared by miners. Lambda database is an open source realization of the Amazon DynamoDB paper with time series data processing capability analogous to Druid and Clickhouse added, which is the unique open source implementation deeply integrated with blockchain technologies. Lambda cloud database provides simple and accessible data writing and reading services. By demonstrating the DynamoDB paper, we can guarantee that the whole system is AP-inclined, that is, apt to ensure fault tolerance and high availability of Sharding. Inside Amazon, the availability of DynamoDB ensures the proper operation of the entire Amazon e-commerce system. Through the open source implementation based on blockchains, we hope to provide fully decentralized distributed database services beyond the centralized IAAS experience for developer groups, open source communities, and DAPP developers.

The IoT in which everything is inter-connected requires a database with a low cost for storage of magnanimous data.

In the last 30 years, owing to the development of Internet, we marched from material age and analog signal age to the digital times where informationization and digitization storms swept the world and changed the productivity of every industry in human society. As requisite by-products of digitization, data are becoming increasingly sizable. We created 1 ZB data in 2010 and 16 ZB data in 2016, which is predicated to be 160 ZB in 2025. They would play significant roles in future economy activities and impact each enterprise, government body, and consumer individual. Large-scale organizations have realized that data themselves are assets with strategic value, which are gradually independent from physical world and constructing a digital world.

However, the value of data is not brought into full play today. According to the report of Mary Meeker, the Internet queen, among the data we produced by now, only 7% was saved and 1% was analyzed, while others were dumped. The primary reason for the situation is that the growth of centralized data storage and analysis computing power falls far behind that of the data, and that the price of centralized computing power is too high. Since machines have replaced human beings to be the principal production source of data, to take full advantage of the existed data, we must resort to the distributed decentralized fog computing architecture to connect with more dispersed storage and computing power resources to handle the growth. In the current decentralized architecture of blockchain, there are only several point-to-point file storage systems but no data-oriented storage and management system, which will be the first significance of Lambda project.

Lambda project can provide Cryptocurrency incentives to reward miners who offer computing and storage capabilities, and connect to a great number of weak computing power equipment to construct a distributed data storage and computing network with a low cost. Based on this architecture, Lambda platform can perform storage, computing, and transactions on data with low unit value and high gross value. In particular, data on this platform are in time series, whose value reduces by time. Therefore, the data owner is allowed to set storage duration corresponding to different prices through smart contracts.

Data trust - both individual consumers and enterprise consumers require trusty data cloud storage.

In the past decade, cloud computing services flourished, with AWS's annual income exceeding USD ten billion and SaaS becoming the powerful drive for American economic development. But in most times especially in China where a mutual trust has not been established successfully yet, many public cloud services have been used for private deployment and internal uses because many

enterprise customers were afraid that the SaaS enterprise background may result in data leakage. In terms of consumers, with the development of precision advertisement technology, tagging individual customers has become a primary technique of precision marketing, the information of which is usually obtained from transactions in the black market. Individual consumers' express data, e-commerce data, even social data are monetized by suppliers through various means. Consequently, as the owners of data, rather than benefiting from society and technology development, individual consumers even have to bear the harassment of all kinds of crank calls.

Depending on the eternally online Lambda data platform, data-oriented smart contracts can be signed among enterprises, individual consumers, and enterprise consumers to specify that data are saved in blockchains after being collected by Lambda Agent and desensitized during the transmission. As a result, enterprises are accessible to the calculated data on the Lambda platform through APIs while individual consumers can obtain Tokens.

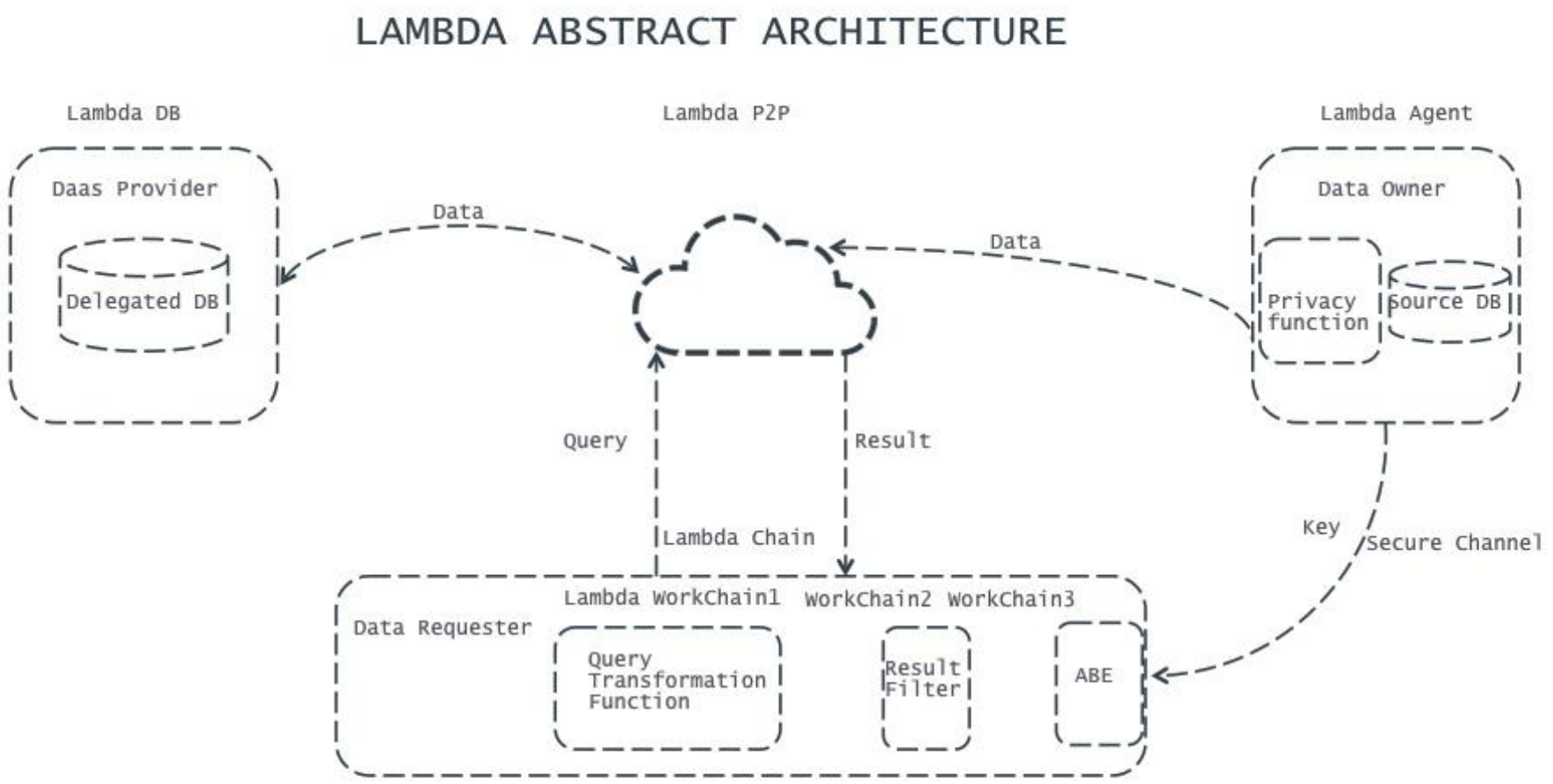
For AI, data push forward all algorithms.

AI technology has already impacted every aspect of our economy system, including advertisement, financial, health care, retail, automatic drive, energy, transport, and logistics industries. Till 2025, software and services related to AI technology would arrive at the scale of USD 60 billion covering 29 industries and more than 150 scenarios. The advancement of AI also requires the support of data, otherwise, its modelling will lose accuracy, which will backfire on the AI model. Presently, leading AI companies including Google and Facebook, all boast of a large number of data resources.

Primitive data and study result transactions can be conducted through Lambda between individuals and individuals, individuals and organizations, and enterprises and enterprises after the development is obtained on the platform, turning data to information, information to knowledge, and knowledge to wisdom of the human kind.

1.3 Design Architecture of Lambda Project

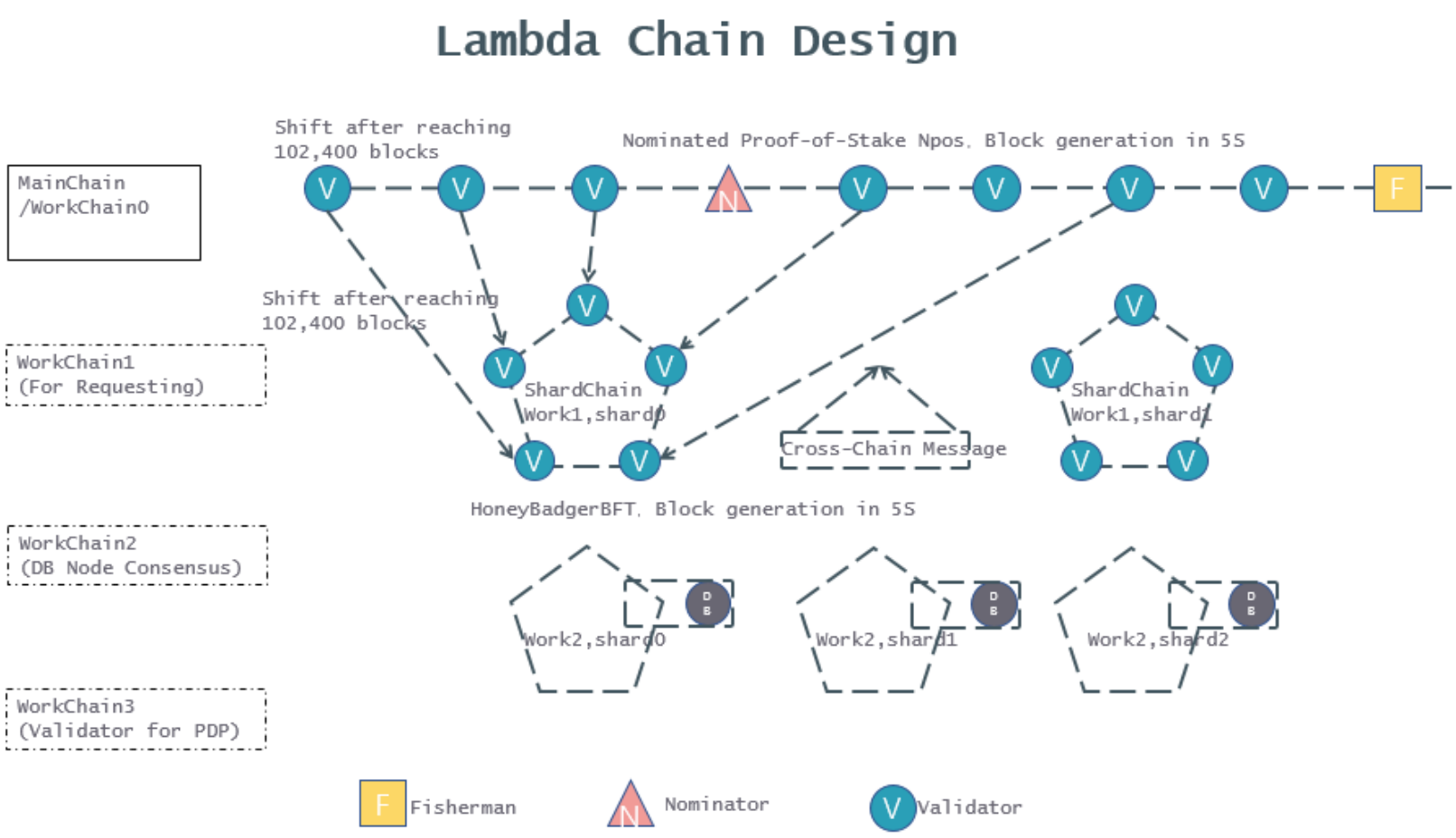
Lambda is a fast, safe, and scalable blockchain infrastructure project, which is able to process several millions of transactions per second with Sharding technology and provide storage capabilities with unlimited scalability for DAPPs through a secured decentralized cloud database. Lambda consists of the following components:



- Lambda Chain, a homogeneous multi-chain system providing high TPS access capabilities, Turing Complete smart contracts, and multi-chain transaction capabilities;
- Lambda P2P, a P2P network system providing addressing capabilities in the network layer;
- Lambda DB, a multi-database cluster system providing encrypted data storage capabilities with unlimited scalability;
- A structured support system at the bottom layer of Lambda DB, including a block storage system and a distributed file system - Lambda FS;
- Lambda ABE, an attribute-based encrypted authentication access system consisting of multiple nodes, playing the role of access control gateway for the database;
- Lambda TPA Chain (WorkChain3), a data integrity verification organization consisting of multiple validator nodes.
- Lambda Agent, a self-adaptive probe system providing memory data storage, performance monitoring, security moni toring, and upload of metrics data.

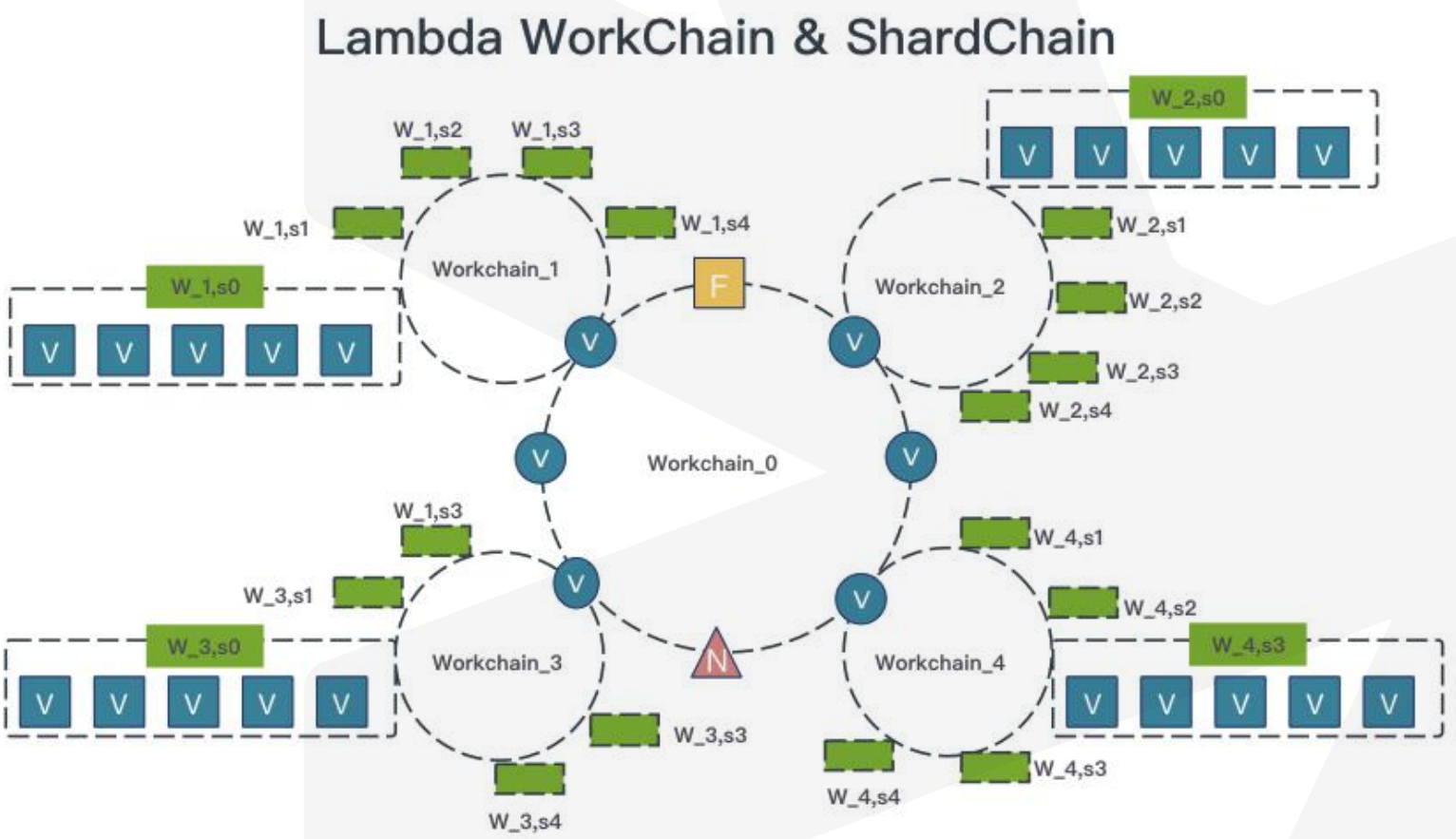
The core concept of Lambda project is a chain-database isolation mechanism and functional sub-chain design. Based on the trust and public verification grades of data, DAPPs can save data in different chains and database systems, to which Lambda project provides corresponding data collaborative management. In addition, as the entire Lambda DB is in a Permissionless environment, Lambda accomplishes the ABE access control mechanism authorized by multiple institutions and a complete PDP of storage data.

The chain-database isolation was mainly designed out of the concern of future system's upgrade and update because the update of the blockchain system would lead to forks and therefore impact the entire economy system irreversibly. To avoid the issue, we endowed the database system with the primary data processing capabilities and enabled the access control system of the database through sub-chains. Functional sub-chains were designed, for one aspect, to achieve future scalability, and more importantly, to fulfill two core functions of the decentralized storage system: privacy protection and PDP. Adopting the efficient ABE plan authorized by multiple institutions and PDP mechanism, we realized access control and encryption of cloud storage data, and data possession verification, respectively.



Lambda Chain is a homogeneous multi-chain design, which, in terms of concept, has three layers and three roles. These three layers are divided into two real chain layers and one virtual chain layer, which are MainChain (real chain layer, WorkChain0), WorkChain (virtual layer, from WorkChain0 to WorkChainN) and ShardChain (real chain layer). Except WorkChain0, all other WorkChains are virtual chains composed of multiple ShardChains, of which the consensus mechanism of MainChain is Nominated Proof-of-Stake NPos while that of ShardChain is HoneyBadgerBFT. MainChain is the master chain and trunk chain of all chains possessing all nodes including nominator nodes, validator nodes, and fishermen nodes. It designates validator nodes to be responsible for each WorkChain's transaction packing and block generation. The Sharding mechanism is achieved by Byzantine Fault Intolerance (BFT) consensus after validator nodes are grouped. Therefore, Lambda supports cross-chain transactions and communications. To better manage the database system, Lambda team realized the first three sub-chains, including WorkChain1 which autho-

rizes, records, and requests transfers for data requests, WorkChain2 which performs statistics of data responses and consensus management of database nodes, and WorkChain3 which verifies integrity of data. In the future, with the participation of more sub-chains, Lambda can realize more functions.



The three roles in Lambda are nominator, validator, and fisherman.

Validator: The validator, who has the supreme authorities, performs transaction packing and block generation in the entire Lambda network, therefore, it needs to pledge as many Tokens as possible. Besides, since we allow nominators with capital to recommend one or multiple validator representatives, the deposit is not fully possessed by the validator but also shared with the nominator. Hardware environment for the validator must be in accordance with corresponding requirements including the number of CPU cores, memory, and SSD hard disk, and the network environment must be featured with high availability, high network bandwidth, and low latency. The validator runs the general ledger client of the trunk chain, namely, the client of MainChain, in the machine environment as specified before. On each block, nodes are ready to verify and pack the Hash value of blocks which have already achieved consensus on sub-chains. Meanwhile, the validator nodes are randomly distributed to different WorkChains to perform packing and block generation for verification of transactions on the sub-chains. Either a MainChain or SubChain generates a block every 5 seconds and the validator node is shifted for every 1024 blocks. Under the consensus algorithms we selected, a validator failing to perform its responsibilities will be punished. For an accidental error, the reward for block generation will be withheld; for a repetitive error, the deposit of the validator may be deducted by burning tokens; for a severe error such as counter signing and collusion attack, all the deposits of the validator will be deducted, part of which is burnt and most are rewarded to honest validators and fishermen who provide information.

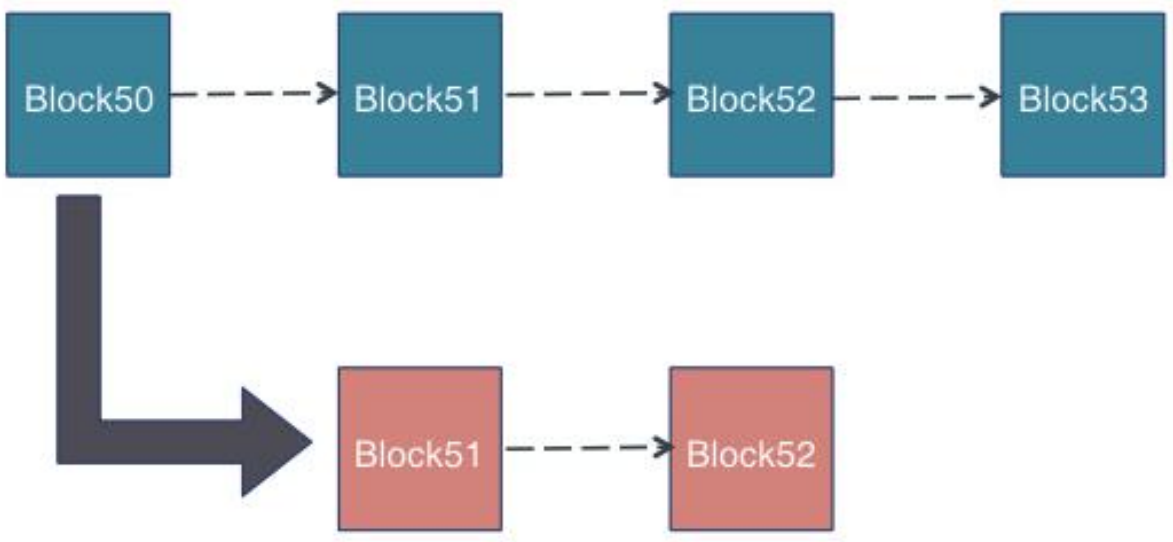
To a certain degree, validators are analogous to mining pools in the bitcoin system.
We estimates that hundreds of validator nodes will be contained in the system.

Nominator: Nominators are a group who owns stakes. They entrust security deposits to validators. Therefore, they have no more responsibilities except investing capital. In our design, each storage node or database node requires entrustment of security deposits. To this sense, each of them is a nominator.

Nominators are analogous to miners in the bitcoin system, which play the role of supervisor.
We estimate that thousands of nominators will be contained in the system.

Fisherman: Fishermen are not correlated to the block packing process since their roles are similar to “bounty hunters” in the real world appealed by a one-time rich reward. Because of the existence of fishermen, malicious behaviors are reduced. Once reporting promptly and proving that at least one participator with deposits has illegal behaviors, for example, signing on two blocks of the same parent block or packing an invalid block of a sub-chain, they can be rewarded. It requires the least to be a fisherman.

Short attack, long attack, and checkpoint mechanism: For a Proof of Stake (POS) mechanism, short attacks and long attacks from malicious validators are issues required to be solved. We demand validators to monetize capital to avoid them to sign on two blocks with the same height. In terms of long attacks, we plan to adopt a checkpoint mechanism similar to that in Polkadot and Casper projects to introduce the concept of compulsory finality.

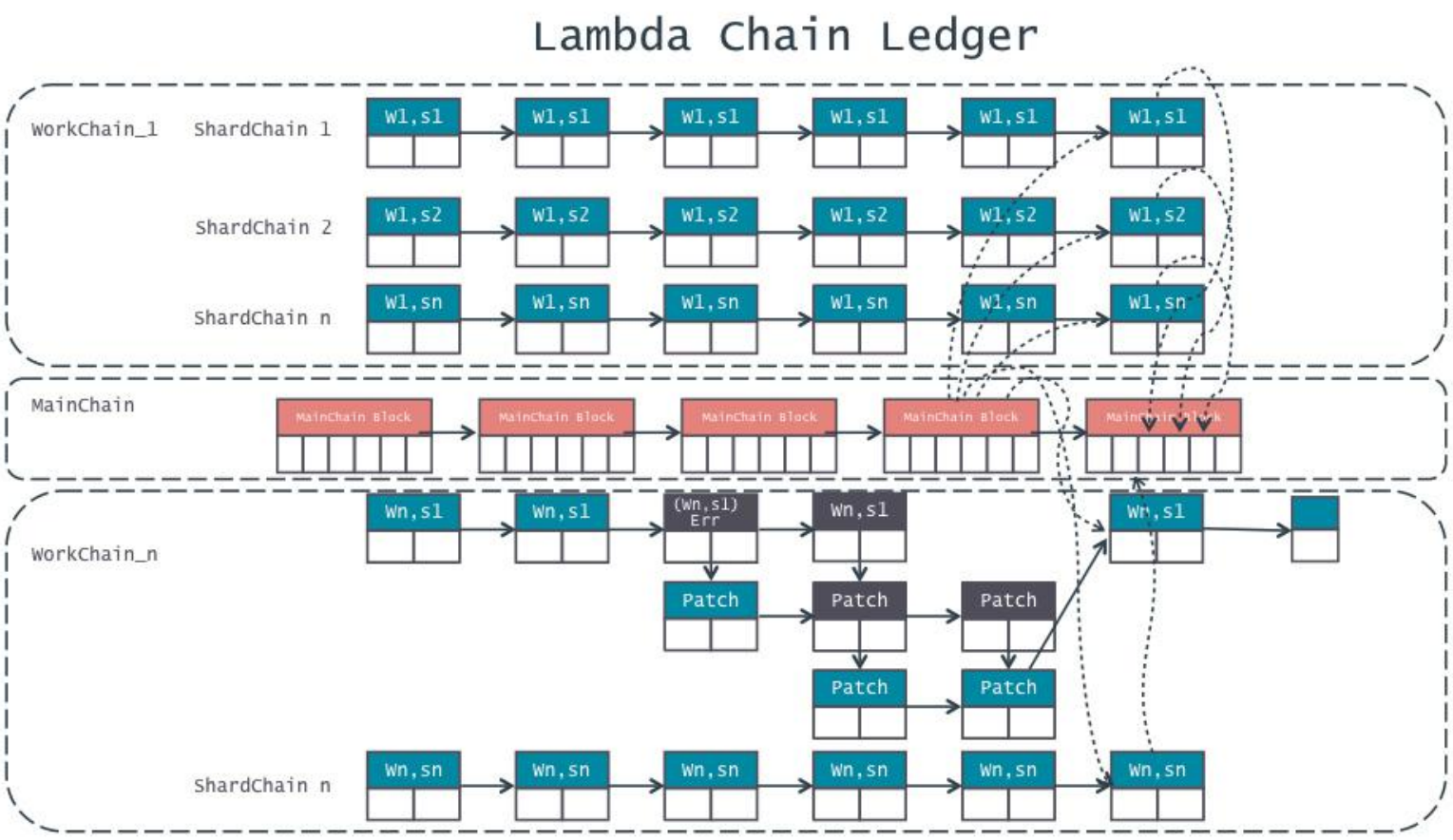


Stake contracts: we adopt stake contracts to manage the whole validator set, which mainly specify the following issues:

1. Which accounts are validators?
2. Which accounts can become validators in a short time?
3. Which accounts pledge Tokens to nominate validators?
4. Attribute state of each account, including account balance, acceptable mortgage ratio, and address list.

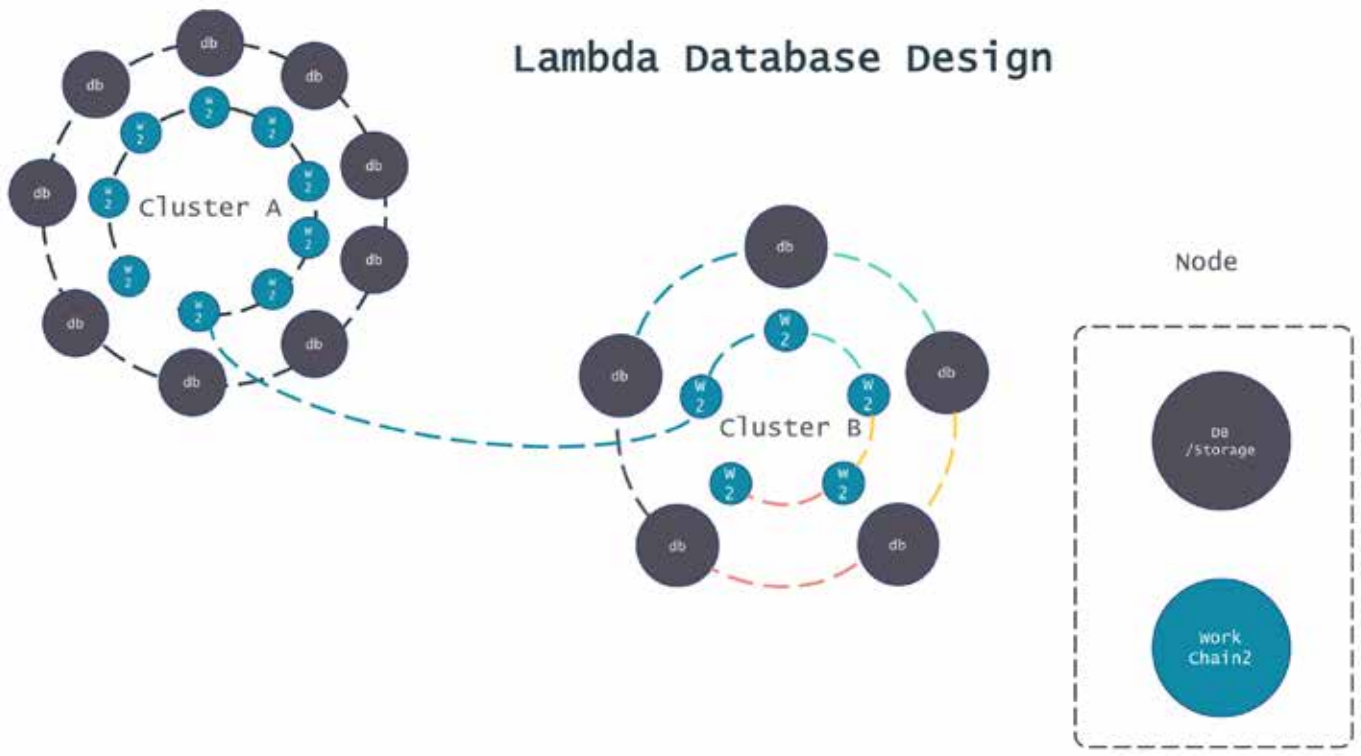
Ledger Structure: MainChain is the general ledger in the Lambda. Each WorkChain will be automatically split and combined according to the loading condition. Usually, a WorkChain, which is a complete blockchain, will be split into multiple SharChains based on the suffix of the account being processed. To enable the mutual-verification structure networking of main ledgers and Sharding ledgers in the Lambda, each new block of the MainChain contains the Hash of all new blocks on the ShardChain (unless part of ShardChain fails to generate a block within the duration required) while the new blocks on the ShardChain contain Hash of blocks on the last MainChain.

In Lambda, the role of fisherman is introduced to launch validity verification against a historical block. Once the historical block or any transaction on it is verified to be invalid, the fisherman will initiate more severe verification against it and restore it accordingly. Therefore, Lambda designs a blockchain for each block itself, which exists as a block of the blockchain in normal case. However, during the restoration, new blocks will be generated in its vertical direction to replace corresponding historical block as a whole or restore part of transaction information on the invalid block. Certainly, the introduction of other part of the invalid block at the same time also requires restoration until all the states are corrected. In this sense, Lambda blockchain ledgers will not produce any forks because even part of transactions are damaged, it can restore them with vertical chains and meanwhile maintain all the records of historical transactions.



Other issues not specified in this paper: Limited to the length of the paper, we do not list basic design issues such as WorkChain registration, messages and queues of cross-chain transactions, P2P network architecture design, vertical chain ledgers, and self-restoring capabilities, and business logic issues such as data request statistics, data response statistics, request response reconciliation, participation of database nodes in transactions, database node off-line consensus processing, and Lambda FS detailed design, which will be elaborated in the technical yellow book being compiled.

Thoughts about the architecture of the chain
The key feature of Lambda, which is also what Lambda is superior to IPFS, is the verification mechanism of PDP. Theoretically speaking, any storage project shall provide a provable possession mechanism and any realizable verification mechanism shall have a challenging party and an authentication party. Among the existed storage projects, Sia and Storj chose users as the challenging party, which requires the client to save the copy of the file or own the file copy in a certain duration. This mechanism can realize dropboxes only but cloud storage nor cloud databases. In IPFS, storage nodes play both the roles of challenging party and authentication party. It seems that it is unable to be realized since it introduce a lot of complicated logic such as zero-knowledge proof. As for us, we need chain nodes to act as the challenging party, which brings in two features: 1 Our verification sub-chain nodes need Sharding, that is to say, not all nodes (thousands of) are used to verify a file at the same time, since it will cause great waste of resources. Besides, verification nodes must reach strong consistency consensus with verification results, namely, forming a consensus with BFT. Thus, we need a Sharding chain of which the internal consensus mechanism is BFT, a strong consistency consensus. 2 We need another chain to achieve ABE and control the access to entire storage and database resource pool in the back end. Generally, ABE is based on a trusted code credit center, which is replaced by one chain and multiple institutions in the project. That’s the reason why the role of validator and BFT must be introduced. 3 We need a chain to reach a consensus in the participation and off-line state of database nodes through BFT algorithm, the nodes on which share the same run-time system with database nodes and must be protected by Trusted Execution Environment (TEE) and Intel Software Guard Extensions (SGX). 4 We need a high-speed chain to make statistics of users’ requests to the database system. In a word, the following requirements must be met: 1 Sharing 2 High-speed sub-chains 3 Homogeneous verification mechanism such as BFT 4 Inter-chain communications



During the design of the data system, to ensure the availability, performance, and scalability of services, we adopted the database cluster technology to provide services for DAPPs. Cluster is a basic concept in Lambda database. When providing a service, data always replicate a copy within a database cluster rather than crossing it. Additionally, the cluster technology can maintain a comparative balance of loading on all equipment.

In the design for the architecture of the database system, Lambda achieved the clustering-based and consistent hashing-aware data placement (CCHDP) algorithm which enables the balanced data distribution in the heterogeneous environment. At present, most placement algorithms are oriented at homogeneous storage systems but Internet equipment is mainly heterogeneous, which prompts a challenging issue about how to effectively place PB scale data. Improper data placement will result in huge waste of storage space, and a lot of energy consumption due to users' search for the data requested. Therefore, data placement algorithms are extremely important in the large-scale network storage system, and equipment volume of balanced use can balance I/O loading.

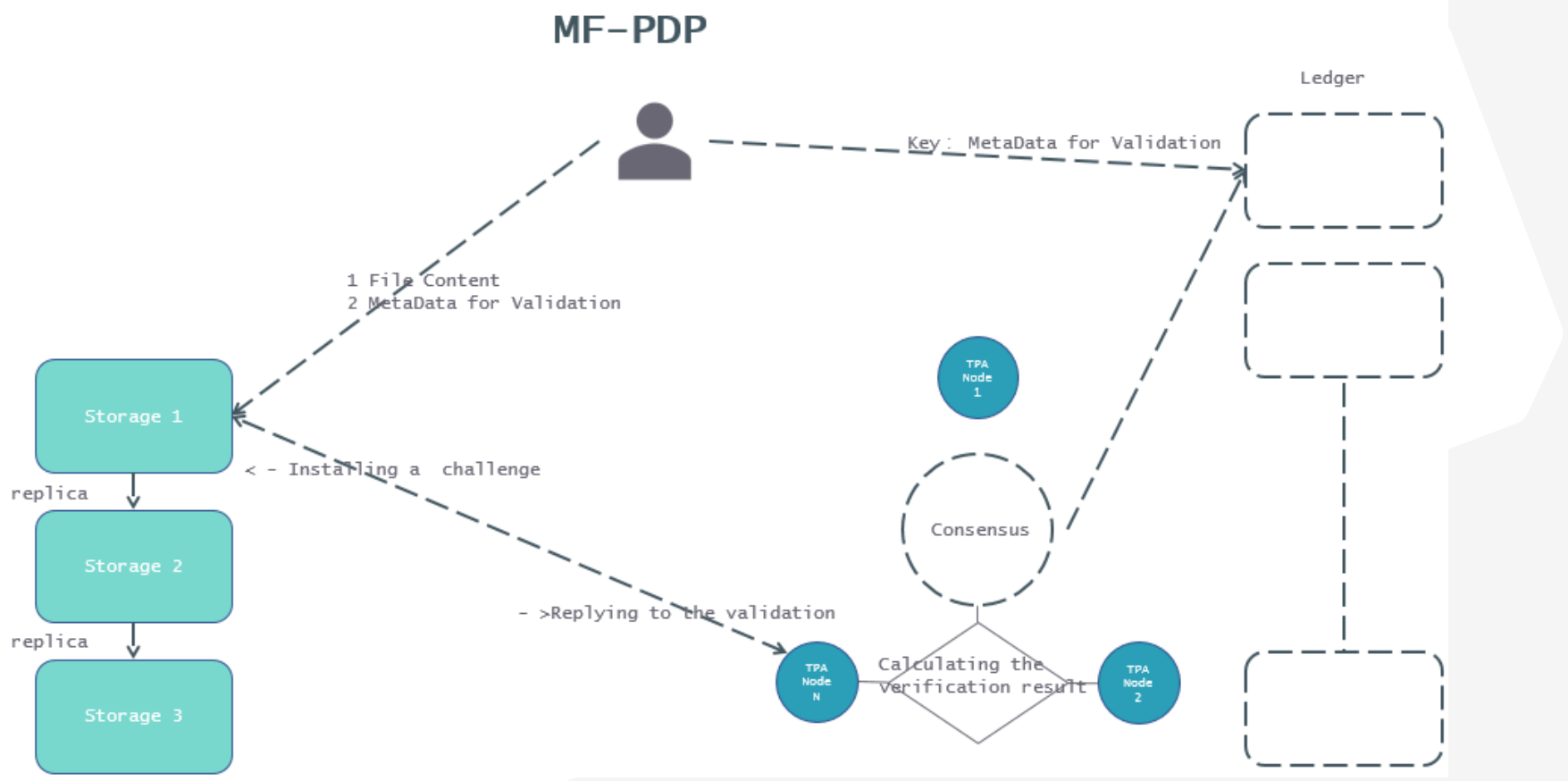
To make the full use of the storage capability of each equipment and maximize the overall performance of the storage system, Lambda distributes data fairly based on the volume of the equipment and bandwidth. The entire system will irregularly add new nodes or delete unqualified equipment or nodes of which the equipment is invalid. To maximize the performance of the storage system, data must be re-distributed equally. Since some applications in the Lambda database system are running for 24 hours, migration of mass data will definitely consume a lot of bandwidth, lower service quality, and even affect the availability of the whole system. To avoid the impact on the storage services of the system, Lambda algorithms ensure that data to be re-distributed are as few as possible. In this way, the data placement algorithms of the Lambda database system can automatically adapt themselves to the variation of storage scales, during which they migrate data as few as possible, guaranteeing that the storage positions of certain data can be calculated with little time and space information.

CCHDP, an algorithm combing clustering and consistent hashing technology, which is applied in Lambda, can equally distribute data based on the weight of the equipment in the heterogeneous equipment set, migrate the least data during the variation of a self-adaptive storage scale, and calculate the position of data in a short time with the introduction of a few amount of virtual equipment only, dramatically reducing the storage space required.

1.4 Basic Principles for Lambda Data Integrity Verification (Optional)

In the preliminary development stage of cloud computing technologies, academia regarded Provable Data Integrity (PDI) and PDP as edge fields of research and performed related explorations unceasingly. In the regular cloud computing environment, users are unable to download data and verify their correctness due to the huge communications cost may be caused by the large-scale data. Thus, with a few data at hand, cloud users need to verify the integrity of far-end data with high confidence possibility using certain knowledge proof protocols or probability analysis methods, which typically include Proof of Retrievability (POR), a user-oriented independent verification method, and PDP, a public and verifiable method. The Nippon Electronic Company (NEC) laboratory promoted the PDI method which not only improved the processing speed and verification object scale of the POR method but also supported public verification. Other typical verification methods include the plan based on the new tree structure MAC Tree promoted by Yun et al.; method based on algebraic signatures promoted by Schwarz et al.; method based on BLS homomorphic signatures and reed-solomon (RS) correction codes promoted by Wang et al.

The biggest difference between the centralized cloud and decentralized cloud storage and database lies in the Permissionless storage nodes. Moreover, cloud users cannot act as the verification initiator nor have a completely trusted TPA node in the former system. In Lambda design, we adopted two updated versions of PDP methods which support public verification: BLS-PDP and MF-PDP. We performed the job of a trusted TPA with the consensus of multiple validator nodes, that is, to fulfill PDP and PDI, and write the verification results on the chain, in case of fishermen's inspection. In the blockchain system, due to the time incrementality of data, the update and deletion logic do not exist and the modification mode of data is just append. In addition, in the cloud storage environment, in terms of dynamic data update issue, we discovered that cloud storage has a data update mode different from traditional storage (such as file system). Typical data update operations do not involve the insertion or modification of file content but change (usually adding) of the number of static files. In this case, we assume that tablespace files in the database be with characteristics of static files.



Algorithms of BLS-PDP scheme
Select the public parameter **e: G X G →G *r*** as a bilinear mapping, *g* as the generator of *G*; **H:{0,1}* —G** as BLS Hash function. Setup stage: Randomly select a private key *a*← *Zp*, calculate the corresponding public key $u=g^a$; public key of private key *sk = (a)* is *pk = (v)*. Randomly representing the unique file as $\upsilon \xleftarrow{R} \{0,1\}\{0,1\}^{\mathcal{K}}$ and select an auxiliary variable $\upsilon \xleftarrow{R} G$ to deblock the file: *F = {b1,…, bn}*, and generate an verification metadata set: $\Phi = \{\sigma_i\}_{1<i<n}$, where $\sigma_i = (h(v\|\|i) \cdot u^{m_i})^{\alpha}$. Save *F* and the verification metadata *O* to the remote server. Challenge stage: 1.A validator randomly selects **c** block indices from block index *[1,n]* and grants a random number $v_i \xleftarrow{R} Z_{p/2}$ for each block index to establish a challenge request $chal = \{i, v_i\}_{s_1<i<s_c}$, and sends it to the authenticator. 2.Upon receiving the chal request, the authenticator calculates $\sigma = \Pi_{i=s_1}^{s_c} \sigma_j^{v_i} = \Pi_{i=s_1}^{s_c} H(v\|\|i)^{v_i} u^{v_i m_i}$ first and then calculates $\mu = \sum_{i=s_1}^{s_c} v_i m_i = v_1 m_{s_1} + \cdots + v_s m_{s_c}$ and returns *{σ, μ}* as a proof to the validator. 3.After receiving the proof *{σ, μ}*, the validator checks whether the outsourcing data is integrated according to the equation: $e(\sigma, g) \stackrel{?}{=} e\left(\Pi_{i=s_1}^{s_c} H(v\|\|i)^{v_i} \cdot u^{\mu}, v\right)$

Algorithms of MF-PDP scheme
KenGen()→(pk,sk) 1. Generate public key *pk = (N, e, g)* and private key *sk = (N, d, g)*; 2. Output *(pk, sk).* Add *(usk,F, a,GID) →(F' ,M, a')* 1. Command *(N, d, g) = usk, (B) = a.* 2. Split file *F* into **t** blocks - *F[1],...,F[t]*, the length of each block is **L** bits. 3. For *1 ≤ i ≤ t* 1)Calculate the homomorphic verification signature of data block *F[i]*: $A[i]= ((H(GID)\|\|B+i) \cdot g^{F[i]})^{d} \bmod N.$ 4. Record the subscript range of F in the file group: *R=(B,B+t)*. 5. Update the largest subscription in the file group: *B' =B+t*. 6. Output $(F' =F, M=(A[1],\ldots,A[t],R), \alpha' =(B'))$.

Challenge(a)→chal

1. Generate a random challenge private key k_1 and k_2: $k_1 \xleftarrow{R} \{0,1\}^k$, $k_2 \xleftarrow{R} \{0,1\}^k$
2. Set the number of blocks to be challenged to **c** according to the target security grade definition.
3. Output $\text{chal}=(c, k_1, k_2)$.

$\text{Prove}(upk, chal, F', M, \alpha) \rightarrow P$

1. Command *(N,e,g)=upk,(c, k_1, k_2)=chal,(B)= a.*
2. For *1 ≤ j ≤ c*

1)Calculate the virtual subscription value of the data block to be verified: $i_j = \pi[B]_{k_i}(j);$

2)Calculate coefficient: $b_j = e_{k_2}(j)$.

3)Locate the data blocks to be inspected and their homomorphic verification metadata:

$$f[i_j] = F'[i'_j] = F'[i_j - F'.R.start], a[i_j] = M.A[i'_j] = \text{M.A}[i_j - F'.\text{R.start}].$$

3. Calculate $a = a[i_1]^{b_1} \cdot \ldots \cdot a[i_c]^{b_c} \bmod N$, the value of which shall be equal to that of $(H(GID||i_1))^{b_1} \cdot \ldots \cdot H(GID||i_c)^{b_c} \cdot g^{b_1 f[i_1]+\cdots+b_c f[i_c])^d} \bmod N$.
4. Calculate $f=b_1 f[i_1]+\ldots+b_c f[i_c]$.
5. Output *P=(a,f)*.

Verify *(upk,chal,P,α)→{0,1}*

1.Command *(N,e,g)=upk,(c, k_1 , k_2)=chal,(a,f)=P,(B)=α;*

2.For *1 ≤ j ≤ c*

1)Calculate $i_j = \pi[B]_{k_1}(j)$ and $b_j = e_{k_2}(j)$.

2)Calculate $\tau_1 = a^e \bmod N, \tau_2 = H(GID||i_1)^{b_1} \cdot \ldots \cdot H(GID||i_c)^{b_c} \cdot g^f \bmod N$.

3.If $\tau_1 = \tau_2$, output **1**, otherwise, output **0**.

1.5 Lambda ABE Data Access Control Plan Authorized by Multiple Institutions (Optional)

Currently, data on the blockchains are public and accessible, which limits the application scenarios to a large extent. To extend application scenarios, Lambda provides an access control plan based on the ABE authorized by multiple institutions, along with data encryption capability and the capability to encrypt removal of attributes through agents. With attribute as public keys, ABE associates cipher texts with user private keys and attributes, which can flexibly indicate access control strategies, greatly reducing the processing overhead of network bandwidth and nodes sending due to the data sharing fine-grained access control.

In the ABE plan, the composition of a private key is related to the attribute set while that of the cipher texts is related to the access structure. If the attribute set can meet the access structure requirement in the cipher text, a clear text can be obtained. ABE does not only have the one-to-multiple feature, but also can be used for deciphering of an unconfirmed number of users, and therefore widely applied to the fine-grained access control for cloud storage data. But regular ABE mechanisms cost a lot for computing and the expense increases linearly with the growing number of attributes in the access structure, which is not suitable for direct use in the mobile terminals with limited power resource, let alone in the blockchain sector.

The online-offline and private key conversion technology can lower the computing expense in encryption and deciphering on the client by pre-processing and outsourcing deciphering. However, the former method must be aware of the access structure in the off-line encryption stage, but since the access structures of different data are varied, it is difficult to know it in advance; while the latter one outsources the deciphering to a third party without a completely mutual trust, which cannot ensure the correctness of deciphering. Although the plan promoted by Shao et al. does not need to make sure the access structure in advance, the attribute set is managed by one attribute authorization institution only, which is not beneficial for the future system expansion nor be able to verify the correctness of the outsourced deciphering.

Facing the challenges mentioned above, Lambda adopted an Online/Offline Multi-Authority Attribute Based Encryption (OO-MA-ABE) plan, the primary thought of which is to transfer the online computing cost of the client to the off-line stage or cloud servers. The primary advantages of the plan are listed as follows:

1) With the use of online-offline and outsourced deciphering technology, Lambda formulates an efficient mobile cloud storage data access plan. In the encryption stage, it pre-processes a large number of matching operations in advance; in the deciphering stage, it outsources the matching operations to the cloud storage server. Consequently, the client can fulfill encryption and deciphering with simple calculation operations only, significantly decreasing the online computing expenses of the client.

2) Also, Lambda promotes a method to verify the correctness of outsourced deciphering. In the encryption stage, the user generates Hash values by encrypting private keys and clear texts as the verification token of data; in the deciphering stage, the user adopts the token to verify the correctness of the deciphering result, and therefore check whether the deciphering of cloud storage server is correct. Meanwhile, the plan specified in the paper can resist from the identity information acquisition by single institutions, protecting the privacy of users.

3) Lambda team performs the security analysis and simulation experiments for the plan, the results of which indicate its security and efficiency.

Formalization definition of ABE mechanism
Definition **1** (access structure). Assumed that $\{P_1, P_2, P_3, P_4 \dots P_n\}$ is the participator set, $P = 2^{\{P_1, P_2, P_3, P_4 \dots P_n\}}$., and the access structure *A* is the non-void sub-set of $\{P_1, P_2, P_3, P_4 \dots P_n\}$, that is, $A \subseteq P \backslash \{\varnothing\}$; if access structure *A* is monotonous, thus, $\forall B,C$; if $B \in A$ and $B \subseteq C$, thus, $C \in A$. Definition **2** (bilinear pair). If the mapping $e: G_1 \times G_1 \to G_2$ has the following features, it is a bilinear pair: (1) bilinear: if $e(f^a, f^b) = e(f,h)^{ab}$ is in $\forall a,b \in Zq, \forall f,h \in G1$, thus $e: G_1 \times G_1 \to G_2$ is bilinear; (2) non-degenerate: if $\exists f \in G_1$, thus $e(f,f) \neq 1$; (3)calculable: if $\forall f,h \in G_1$, thus, there is an effective algorithm to calculate $e(f,h)$. Notes: e(*,*) is a symmetric operation, in which $e(f^a, h^b) = e(f,h)_{e(f,h)}^{ab} = e(f^b, h^a)$.. Definition **3** (calculating **Diffie-Hellman(CDH)**). After randomly selecting $a,b,c \in Z_q^*$, and specifying the triad (g, g^a, g^b),, it calculates g^{ab}. Definition **4** (determining bilinear **Diffie-Hellman**(**DBDH**)). After randomly selecting $a,b,c \in Z_q^*$, $R \in G_2$, and specifying the tuple (g, g^a, g^b, g^c, R), it determines whether the equation $e(g,g)^{abc} = R$ can be established. Definition **5** (determining linear (**D-Linear**)). After randomly selecting the generators *g, f, v* of group *G* with order *q*, and indices $a,b \in Z_q$,$R \in G$,, and specifying the tuple (g,f,v,g^a,f^b,R), it determines whether $v^{a+b} = R$ can be established.

Definition **6** (selecting cipher attack (**IND-CCA**) game). The opponent and the challenged have following interactions: (1) the challenged systematically establishes an encryption plan, outputs the public and private key pair, and hands it in to the opponent; (2) the opponent can inquire the challenged about deciphering while the challenged deciphers the cipher text and returns the result to the opponent; (3) the opponent selects two clear texts M_0, M_1, and sends them to the challenged, who casts a fair coin $b \in \{0,1\}$, encrypts the clear text M_b and sends the obtained cipher text *C** to the opponent. (4) the opponent can continue inquiring the challenged as step (2) but the inquiry cipher text shall not be *C**; (5) finally, the challenged must answer **0** or **1** (marked as *b′*) for the supposition of *C**. If $b'=b$, the opponent wins the game. The advantages of the opponent in the game are defined as $\Pr[b'=b] ! 1/2$.

1.6 Typical Case Scenarios - IoT

With the gradually decreasing of sensor cost and fast improvement of network speed, IoT embraces a rapid development. However, during the implementation of technologies, there are still a lot of problems, such as scalability, security, and value liquidity. Little by little, people discover that blockchain technologies can resolve many problems in the IoT sector, such as identity recognition and inter-connection and inter-networking. Practically speaking, a lack of awareness in the data level is the reason why the existing IoT projects encounter a lot of difficulties in implementation.

First of all, the data cochain in the IoT is deficient of proper carriers. IoT data are various metrics, which are magnanimous and strictly increasing by time with simple structures, whose generation process can ensure the reality. Logically, they do not need any modification nor update. However, the current blockchain system does not have a database suitable for them because most TSDBs used now are established based on the traditional data storage such as HBase and Rethinkdb, where final data files are saved in the decentralized databases, in the danger of being tampered, disclosed, or lost.

Secondly, IoT data storage occupies huge space and the data scale are usually calculated in TB or even PB. The value of single data is little but when being put together, it is incredible. Value extraction from data is a long-term work similar to gold washing, which consumes a lot of storage and computing resources.

Time series data generated in IoT have a time stamp. In logic, the two features mentioned above do not need to be changed, which makes blockchain technologies meaningful in the IoT data storage field. Tokens in the blockchain can measure slight value, encouraging people to share and contribute computing and storage resources to obtain the Token reward. The value and imagination space of these two fields are immense, such as satellite remote sensing data, hydrology data, and air data. Based on the factors such as period, region, and event, data collected by sensors can be sold through smart contracts to any individual or institution requiring or interested in them.

Meanwhile, in the blockchain sector, existing basic chains, such as Ethereum protocol, have not properly resolved data storage issues since many storage projects, such as Sia and Storj, are still based on the early bitcoin protocols, which are already out of time. The original design of Ethereum protocol focused on the financial system mainly considering the realization of Turing Machine but few of data input, and completely ignored the requirement of database type.

Lambda project provides an IoT-oriented system for data collection, data transmission, data storage, data computing, and data transaction, which consists of Lambda Agent, Lambda CEP, Lambda DB, and Marketplace. Among which, Lambda DB establishes a TSDB system based on the blockchain technology. Different from BigChainDB, the bottom layer of which is based on the realization of traditional database RethinkDB, Lambda DB is based on the Amazon Dynamo paper [1], distributing data files in the bottom layer to corresponding nodes, and saving Hash information in blockchains. With the establishment of Lambda DB, the storage of IoT data is as simple as that in the traditional programming sector, and meantime scalability, availability, and security issues are solved. Lambda Agent guarantees applications' capabilities of fast deployment, development, monitoring, and security safeguard because all applications have to invoke it to transmit information to Lambda DB.

To ensure that developers and investors share the value of distributed computing and storage, Lambda project will independently develop and initially adopt Cryptocurrency:LAMB, which is based on the native blockchains. Presently, online suppliers all over the world need to improve the computing capabilities of their products, including IoT, Internet of Everything (IoE), big-data corporations, and various popular applications, especially in the blockchain sector, where computing power is mainly supplied by miners. But Lambda project provides another more practical and widely-used solution, in which all individuals or suppliers relying on Internet to operate businesses all around the world can use LAMB Cryptocurrency to meet their active demands of computing capabilities. Even better, all Internet users can increase incomes by renting computing resources to LAMB project.

Particularly, the fog computing architecture of the project is more suitable for IoT scenarios. As a significant component of presently world Internet computing capabilities, IoT/IoE is also what Lambda project's breakthroughs and applications are aimed at. Rather than adopting cloud computing which consumes a lot of resources, we introduce fog computing with high cost-effectiveness. Compared with cloud computing in which all data, data processing, applications are saved in the cloud, fog computing adopts a distributed architecture and saves these content in the network edge equipment, data storage and processing of which depends more on local equipment rather than on servers.

IoT Use Case

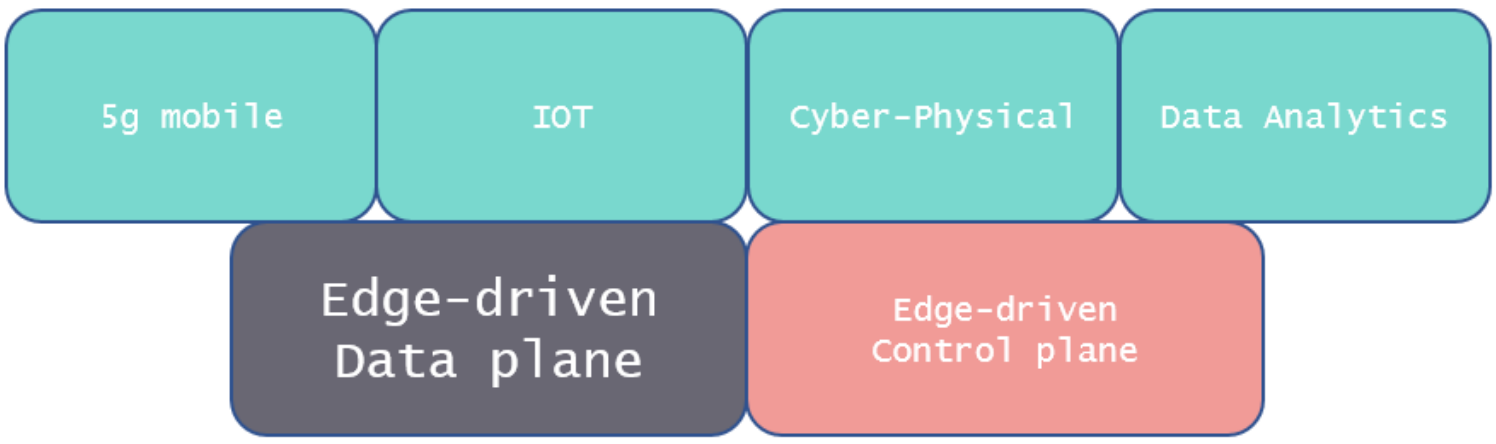


Figure 1: Data plane and control plane of Fog networks enable different applications

1.7 Typical Case Scenarios - AI

AI technology has already impacted every aspect of our economy system, including advertisement, financial, health care, retail, automatic drive, energy, transport, and logistics industries. Till 2025, software and services related to AI technology would arrive at the scale of USD 60 billion covering 29 industries and more than 150 scenarios. The advancement of AI also requires the support of data, otherwise, its modelling will lose accuracy, which will backfire on the AI model. Presently, leading AI companies including Google and Facebook, all boast of a large number of data resources.

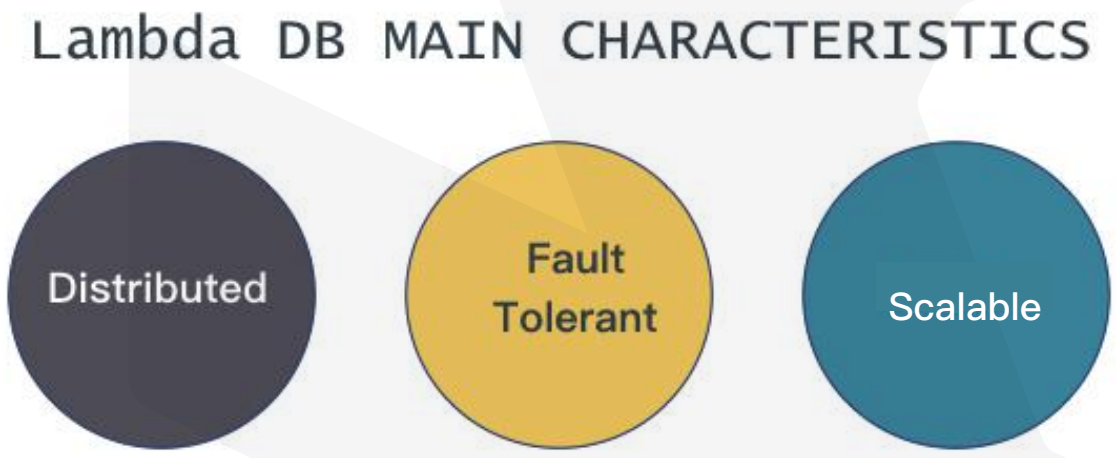
In the AI sector, the consensus is that the effect of data is multiple times that of the model. However, it is difficult and expensive to obtain AI data. The lack of data, restricts the development of AI. Due to the reasons such as lack of mutual-trust and data abuse, free individuals are reluctant to provide data for AI owners and many data institutions attempt to obtain data with gray techniques.

Ultimately, Lambda project will establish a network integrating all-time series data throughout the world, where data owners can sell their data using smart contracts in the Marketplace. Since the value of time series data lies in the insight after analysis, data owners can sell analysis results or even insights of the results instead of primitive data on the platform.

In the future, Lambda project will adopt technologies such as zero-knowledge proof to further develop smart contracts related to core scenarios of data transactions.

II. LAMBDA DB TECHNICAL PRINCIPLES (DESIGN OF DATABASE)

Lambda DB is completely decentralized, distributed, partitioning fault tolerant, and extremely scalable.



Decentralization: there are no central nodes.

Symmetry: all nodes are symmetric.

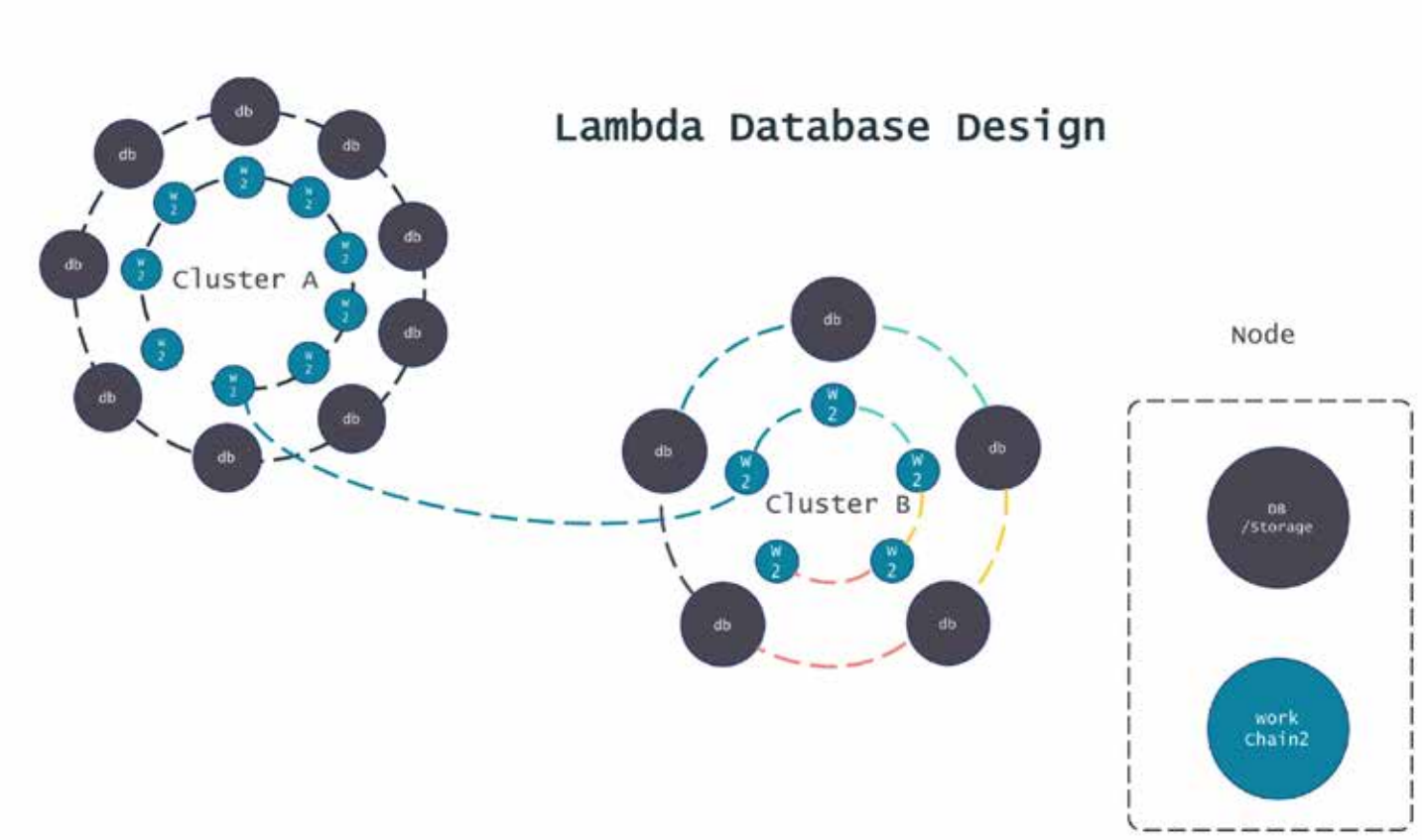
Scalability: incremental scalability strategy is adopted;

Fine-grained: computing capabilities of nodes can be refined.

Problem	Technique	Advantage
Partitioning	Consistent Hashing	Incremental Scalability
High Availability for writes	Vector clocks with reconciliation during reads	Version size is decoupled from update rates.
Handling temporary failures	Sloppy Quorum and hinted handoff	Provides high availability and durability guarantee when some of the replicas are not available.
Recovering from permanent failures	Anti-entropy using Merkle trees	Synchronizes divergent replicas in the background.
Membership and failure detection	Gossip-based membership protocol and failure detection.	Preserves symmetry and avoids having a centralized registry for storing membership and node liveness information.

Table 1: Summary of techniques used in Lambda DB and their advantages.

2.1 Overall Architecture of Lambda DB

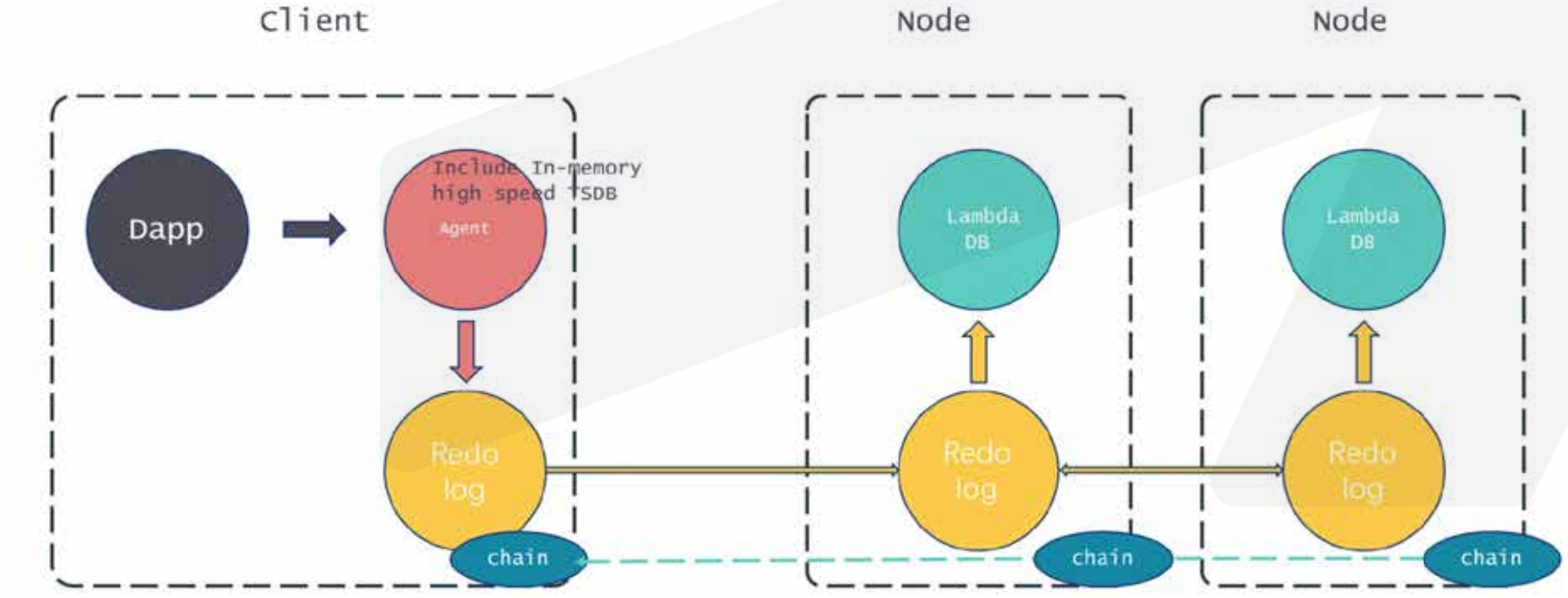


Lambda DB is a peer-to-peer distributed network, where, logically, each node contains a ShardChain (WorkChain) ledger and a run-time system of the database. Database nodes form a small network cluster according to certain rules, whose seed routing information is saved in the chain while the routing table is saved in the P2P system. When using the system, the client pays based on the blockchain system and obtains the supreme authority in the database system. In this sense, the entire MainChain is a low-frequency transaction system, WorkChain1 is a high TPS system billing continuously, and the database is a data access system giving considerations to both high-frequency and low-frequency transactions, which calculates data responses with

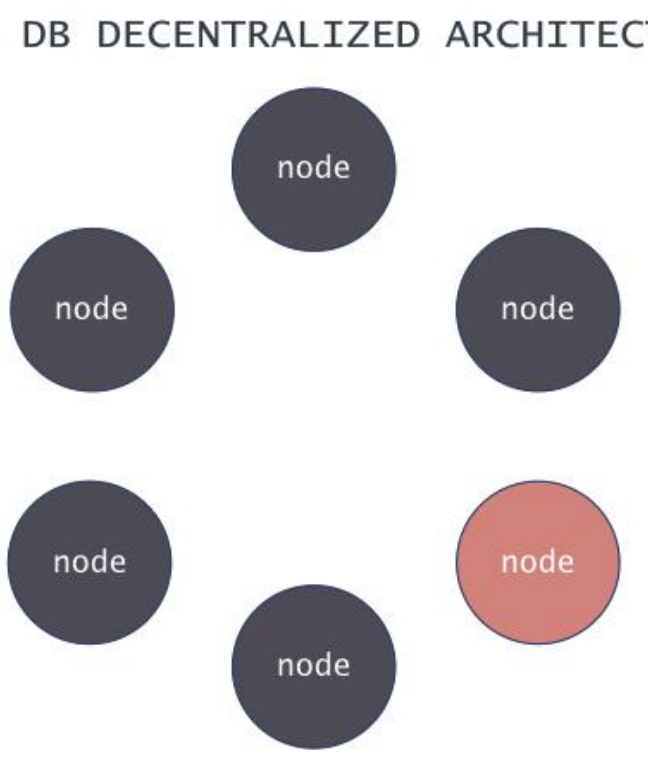
WorkChain2 and checks accounts of request and response information at the checkpoint. In terms of design thoughts, the chain provides a billing system analogous to a public cloud and also the information connection capability across multiple physical IDCs.

In the practical scenarios, all Lambda Clients are deployed with Agent. The embedded Agent helps Facebook to enable the high-performance time series data storage engine Beringei, which saves data within 10 minutes to the local, providing instant access capability. As a higher efficient engine compared with Facebook's previous Gorilla system, Beringei is designed to seek for an extreme speed, and improvement of reading efficiency, whose time stamp compression algorithm adopts delta-of-delta coding and data value adopts XOR to compress storage volume by nearly 10 times. Different from the traditional database storage time series data method based on HBase, Beringei puts data in the memory, which shortens inquiry latency by 73 times and improves throughput by 14 times.

Data generated longer than 10 minutes are transmitted to Lambda database cluster for saving through Remote Procedure Call (RPC).



There are no centralized management nodes but data storage nodes in Lambda DB. All storage nodes have the same responsibilities and provide the same services, therefore, no single point fault will occur in the entire system. The decentralized architecture makes it easy for the system to scale out. A node can join in the entire Lambda DB cluster at anytime, which will result in migration of a few data in the cluster.



2.2 Virtual Nodes, Sharding, and Replication

The core concept of Lambda DB is the Incremental Scalability principle, which requires a mechanism, named distributed hash table (DHT), to enable dynamic Sharding in a group of nodes. With DHT, Lambda TSDB can properly distribute loads to different storage nodes. The Sharding strategy adopted in Lambda DB relies on consistent hashing. For heterogeneity of nodes, we use

CCHDP algorithm to make a further adjustment by following the procedures as below:
1) Adopt the clustering algorithm to classify equipment sets and ensure that the variation of equipment weight in each class is within the preset range;

2) After finishing clustering, inter-class placement mechanism divides [0, 1] into multiple sub-sections according to the weight of the class, distributes one sub-section to each class, and allocates data falling in a certain sub-section to corresponding class.

3) Internal placement mechanism of each class uses consistent hashing to distribute data again and places data to specific equipment.

CCHDP algorithm

CCHDP

Definition 1 (Equality). Assumed that storage system equipment set is $D_0=\{d_1,\ldots,d_n\}$, comparative weight of equipment $d_i \in D_0$ is wi, data set is $X_0=\{x_1,\ldots, x_m\}$, function $f_0:X_0 \rightarrow D_0$ maps data set X_0 to equipment set $D_0=\{\}$, $\forall x \in X_0$, and the possibility that placement algorithm A distributes data x to $d_i \in D_0$ is $p(A(x,f_0)=d_i)$; $\forall \varepsilon>0$, if $|p(A(x,f_0)=d_i)-w_i|<\varepsilon$, thus, the algorithm A is fair.

The equality of the algorithm enables the possibility that data $x \in X_0$ are distributed to equipment $D_i \in D_0$ infinitely approaches the comparative weight of d_i, ensuring that data are equally distributed to each equipment.

Definition 2 (self-adaptability). Assumed that storage system equipment set is $D_0=\{d_1,\ldots,d_n\}$, the comparative weight of $d_i \in D_0$ *is* w_i, data set *is* $X_0=\{x_1,\ldots,x_m\}$, function $f_0:X_0 \rightarrow D_0$ maps data set X_0 to the equipment set D_0; if the placement algorithm A satisfies the two conditions listed below, thus, it is regarded self-adaptive:

*(1)*If the current equipment set D_0 is changed to $D^+=\{d_1,\ldots,d_{n+1}\}$, and provided that function $f+:X_0 \rightarrow D^+$ maps data set X_0 to the current equipment set D^+; $\forall x \in X_0$, *if* $A(x,f+)=f+(x) \in D_0$, thus, $A(x,f+)=A(x,f0)$;

*(2)*If the current equipment set D_0 is changed to $D^+=\{d_1,\ldots,d_{n+1}\}$, and provided that function $f-:X_0 \rightarrow D^-$ maps data set X_0 to the current equipment set D^-; $\forall x \in X_0$, *if* $A(x,f_0)=f_0(x) \in D^-$, thus, $A(x,f_0)=A(x,f^-)$

Definition 3 (distance between equipment). If weights of the two factors d_i and d_j in the equipment set D_0 are w_i and w_j, respectively, thus, the distance between d_i and d_j is $w_{ij}=|w_i-w_j|$.

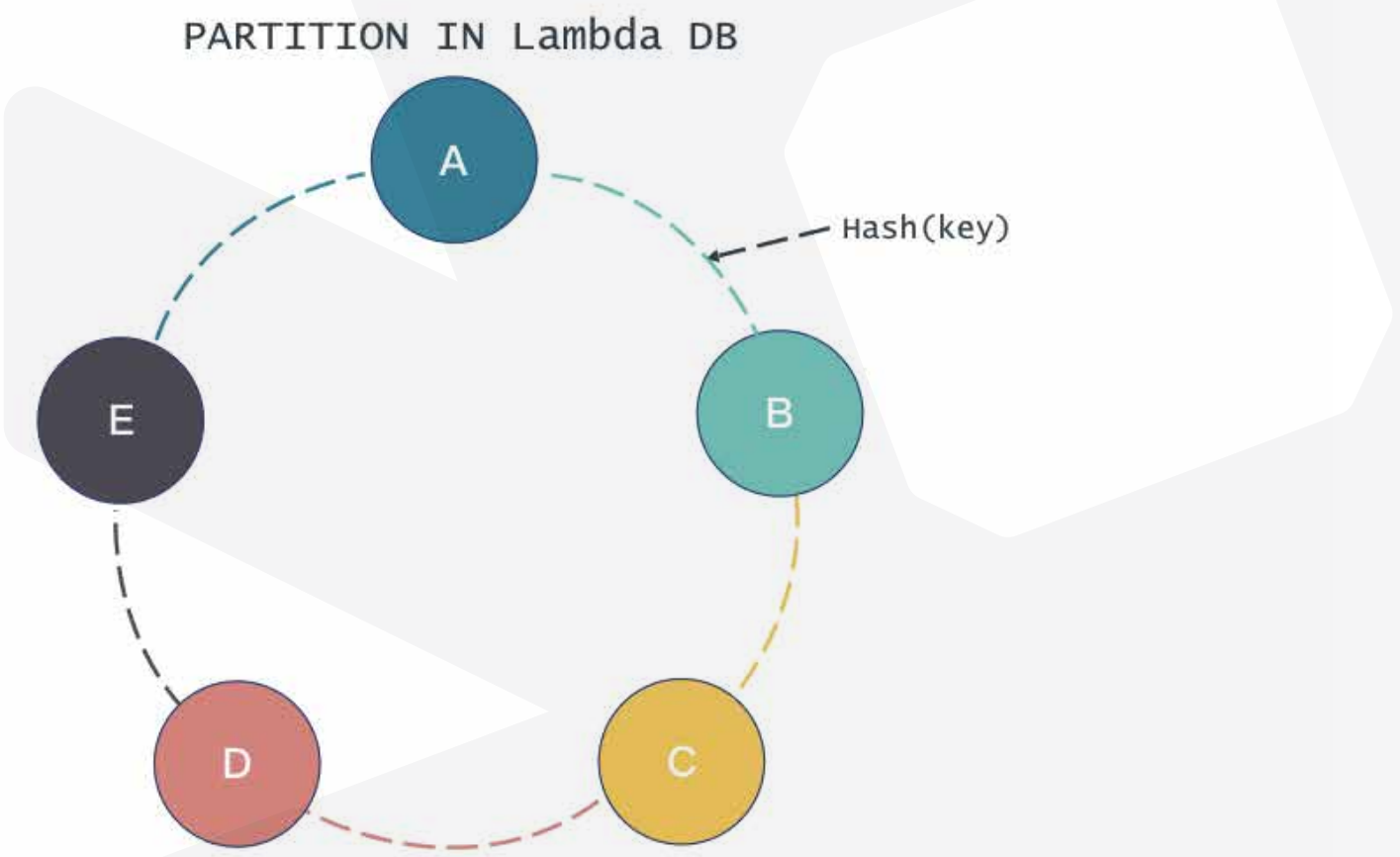
Definition 4 (distance between equipment and class). Assumed that equipment class is S and its clustering center is d_i, and $d_i \in S$, thus, distance between d_j and class S is the same as that between d_i and d_j.

The target of clustering algorithm is to ensure that the distance between the equipment of each class and clustering center is smaller than a preset value. The distance between the equipment and clustering center is calculated based on Definition 4. Assumed that inner-class distance threshold is T, compare T with the distance between the clustering center and equipment, thus, determine the class of the equipment or regard the equipment as the center of a new class: firstly, select any equipment d_i as the clustering center of the first class S_1, for example, choosing d_1 as the center of class S_1; calculate the distance between the next equipment d_2 *to* d_1 according to Definition 4; if the value is less than T, classify d_2 to class S_1; otherwise, d_2 will be regarded as the center of the new class S_2. Assumed that there are k clustering centers, calculate the shortest distance between the unclassified equipment d_i and k clustering centers; if the value is greater than T, equipment d_i will be regarded as the center of new class S_{k+1}; otherwise, distribute equipment to the class nearest to the k clustering centers. Repeat the above procedures until distributing all equipment to corresponding class.

The clustering algorithm divides equipment set $D_0=\{d_1,\ldots,d_n\}$ into g class sets $C=\{D_1,\ldots,D_g\}$.Suppose that class is D_j, of which $1 \le j \le g$, weight of class D { ; 1,2,..., } j i j d_i n= 1 g j j n n = = Σ_j is w_j; then, 1j n j i i w = = Σ and total weight of g classes is 1 1 g j i w = = Σ.

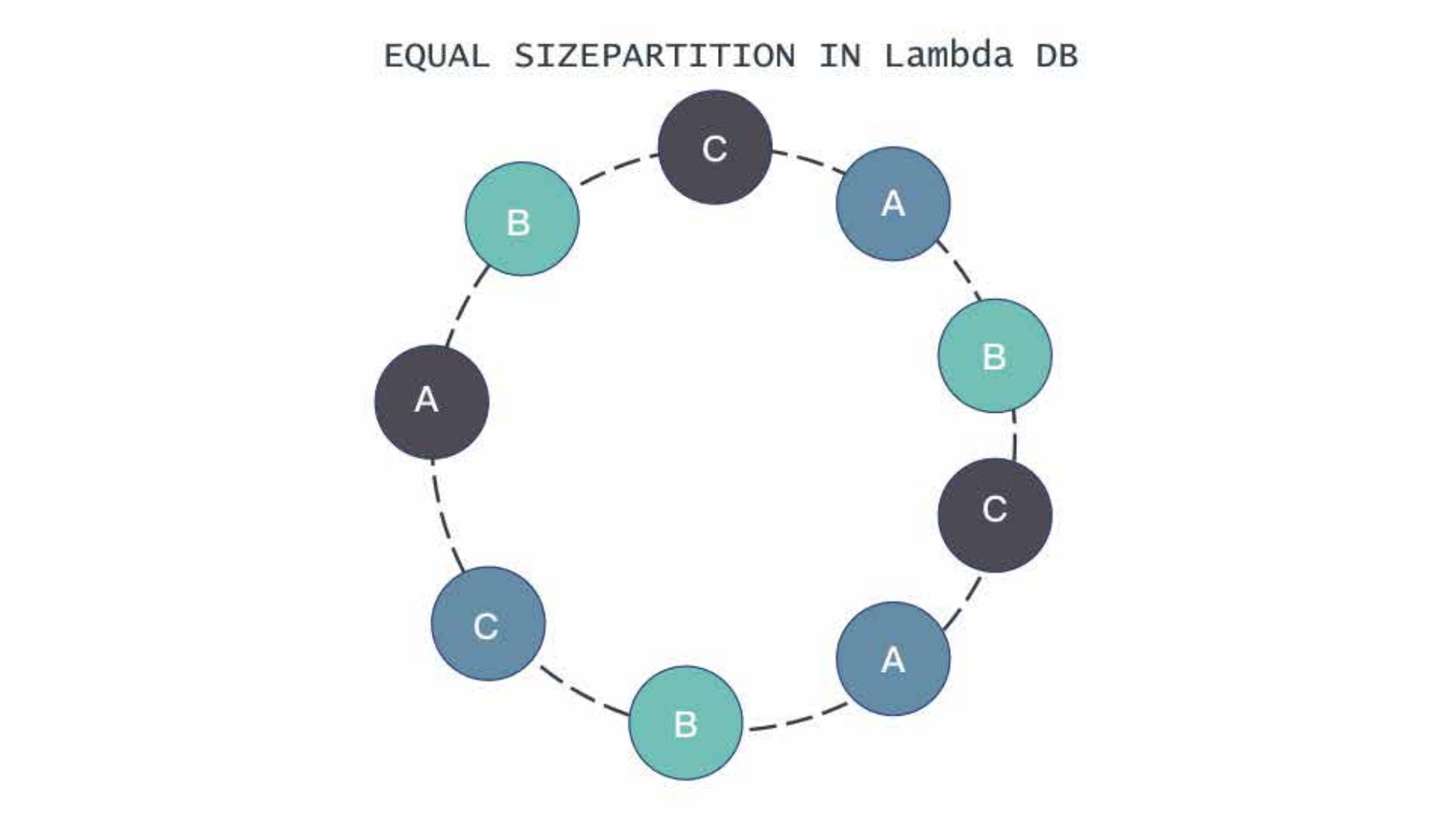
The issue is transformed to data distribution in the heterogeneous class set C of w.

We can simply summarize above algorithm that all keys will obtain unique values through hash function before being saved. Lambda DB establishes a special logic structure, where all component units are connected to form a loop with a fixed length, whose maximum value unit is connected with the minimum value unit.



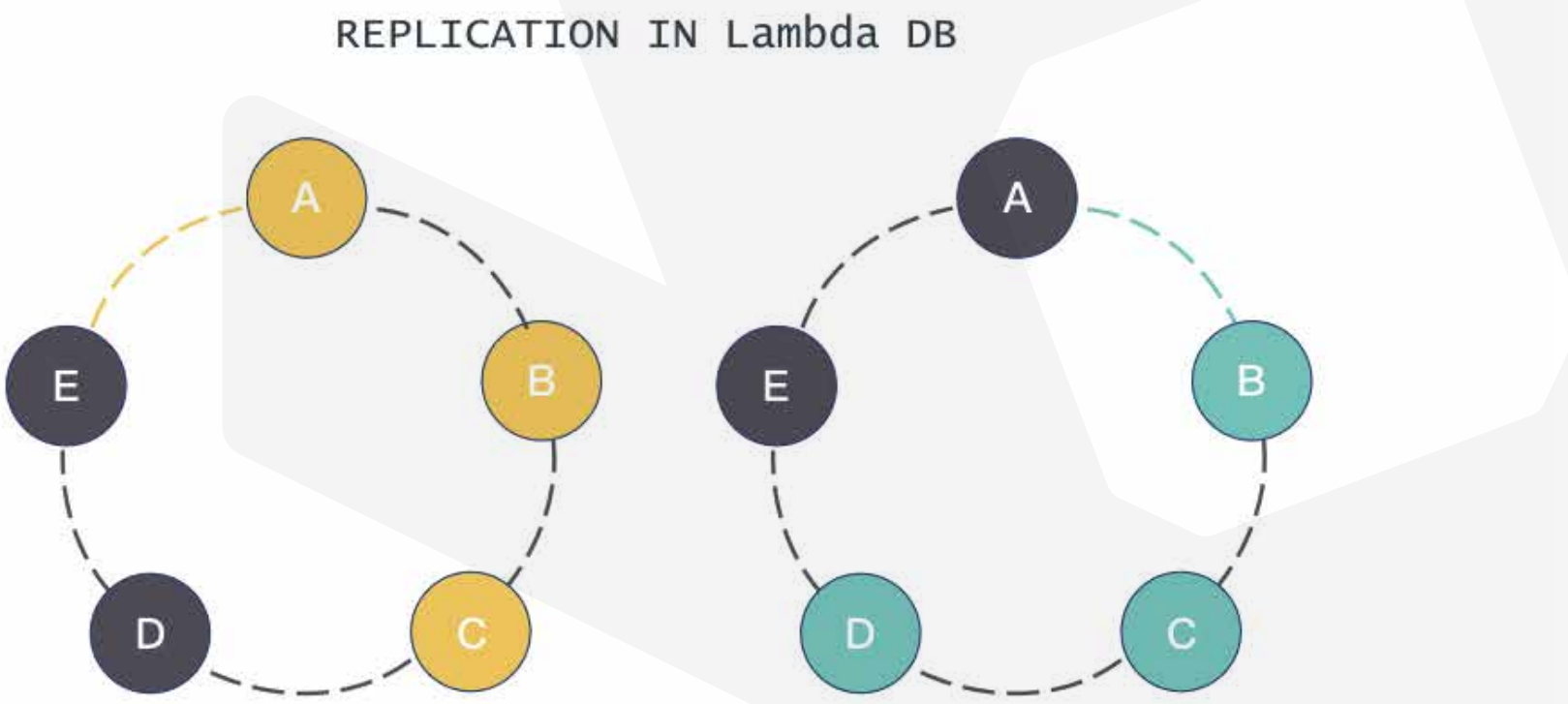
Lambda DB randomly allocates a place to each node in the loop, which processes all keys output by hash function before the current node. Assumed that there is a key value pair (key, value), and Hash (key) result is in the green part in the figure above, node B, the first node found by calculating clock-wisely from the hash result position in the loop, will become the coordinator, who is responsible for processing current key value pair. Each node in the figure above will be in charge of the part in the same color.

Each node will be allocated a random place when joining in the system. However, the processing range responsible for by different nodes may be varied due to the randomness of the algorithm and final loads are distinct. To solve the issue, Lambda DB adopts the variant of consistent hashing algorithm to distribute one physical node to different places in the loop, becoming multiple virtual nodes.



Besides the virtual node strategy, Lambda DB provides another strategy to solve the unbalanced loading, by equally dividing hash of data into m regions with the same size and enabling each of the n nodes to process m/n shardings. Once a node withdraws from the cluster due to a fault or other reason, its owned processing data sharding will be randomly distributed to other nodes; if a node joins in the system, it will take over corresponding data sharding from other nodes.

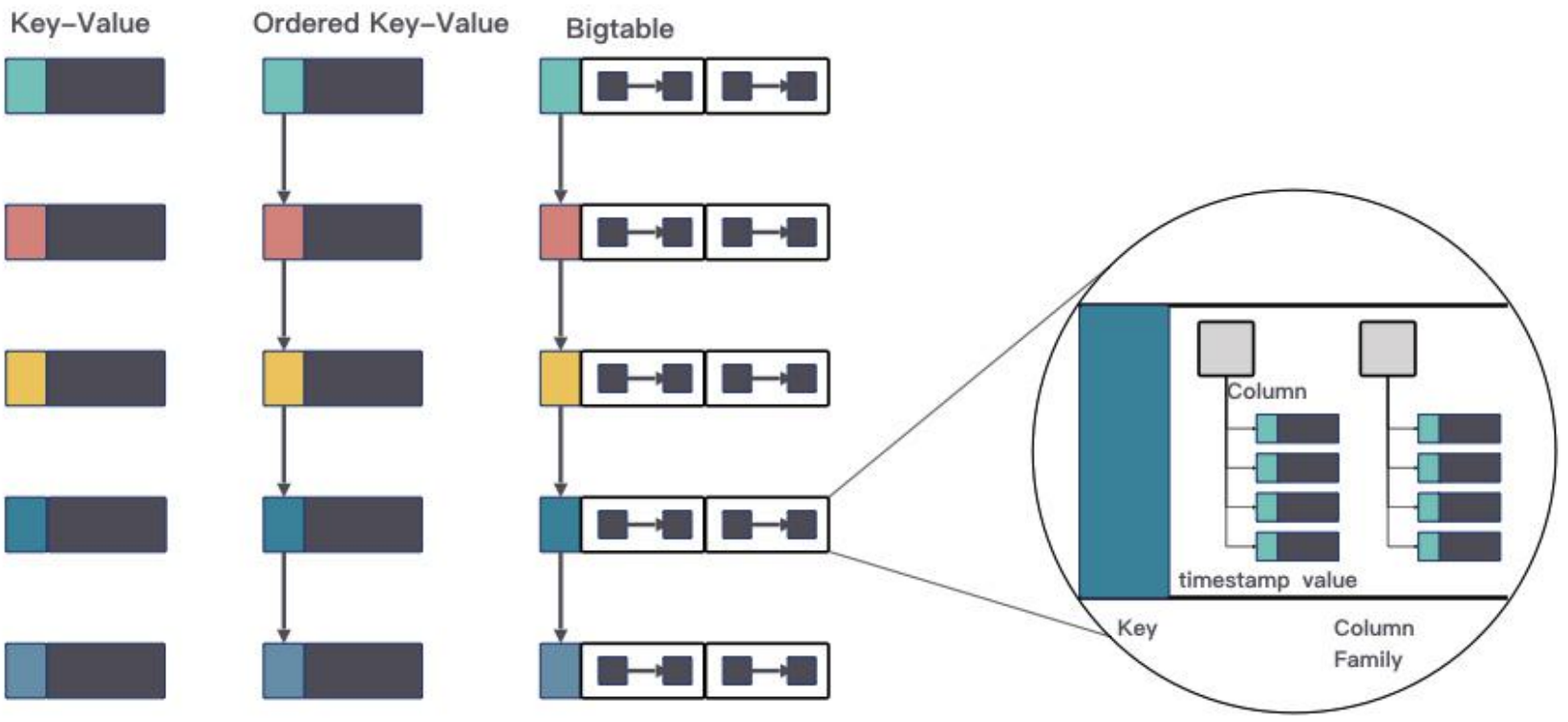
To maintain high availability and persistence, and avoid data loss caused by any node down-time fault, Lambda backs up a copy of data in the coordinator and the subsequent N-1 nodes, of which N is a configurable value usually set to 3.



Namely, the value of the yellow part in the figure above will be saved in A, B, and C nodes, while green part will be processed by B, C, D. In other words, node A processes values in (C, A] while node B processes values in (D, B].

In Lambda DB system, the node table in charge of saving a certain key value pair is called preference list. Because virtual nodes randomly exist in the loop, all nodes in the Lambda DB preference list are non-physical, which ensures that availability and persistence are not affected during a node fault. Since the preference list is saved in the chain, we can simply understand it as that routing information is saved in the chain.

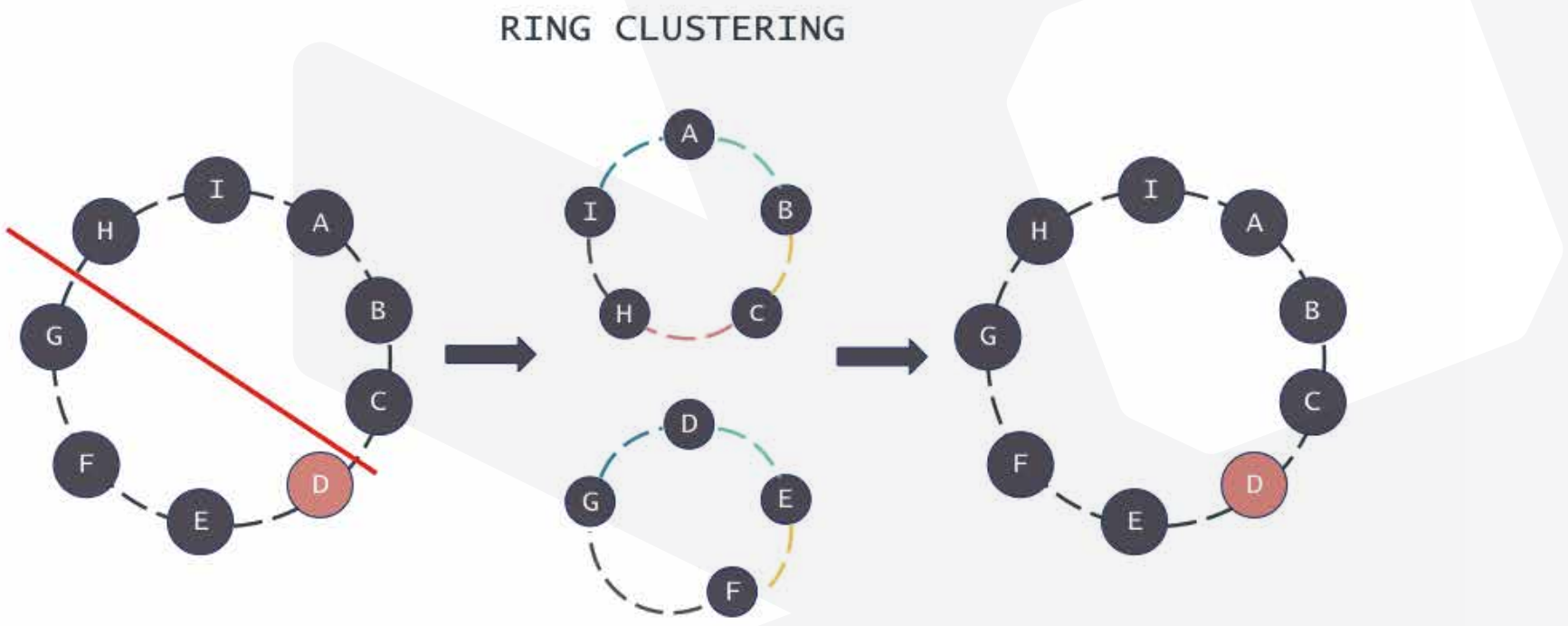
In the key-value model, different processing methods can result in the storage of distinct types of data. The storage methods supported are listed in the figure below:



2.3 Clustering and Partitioning

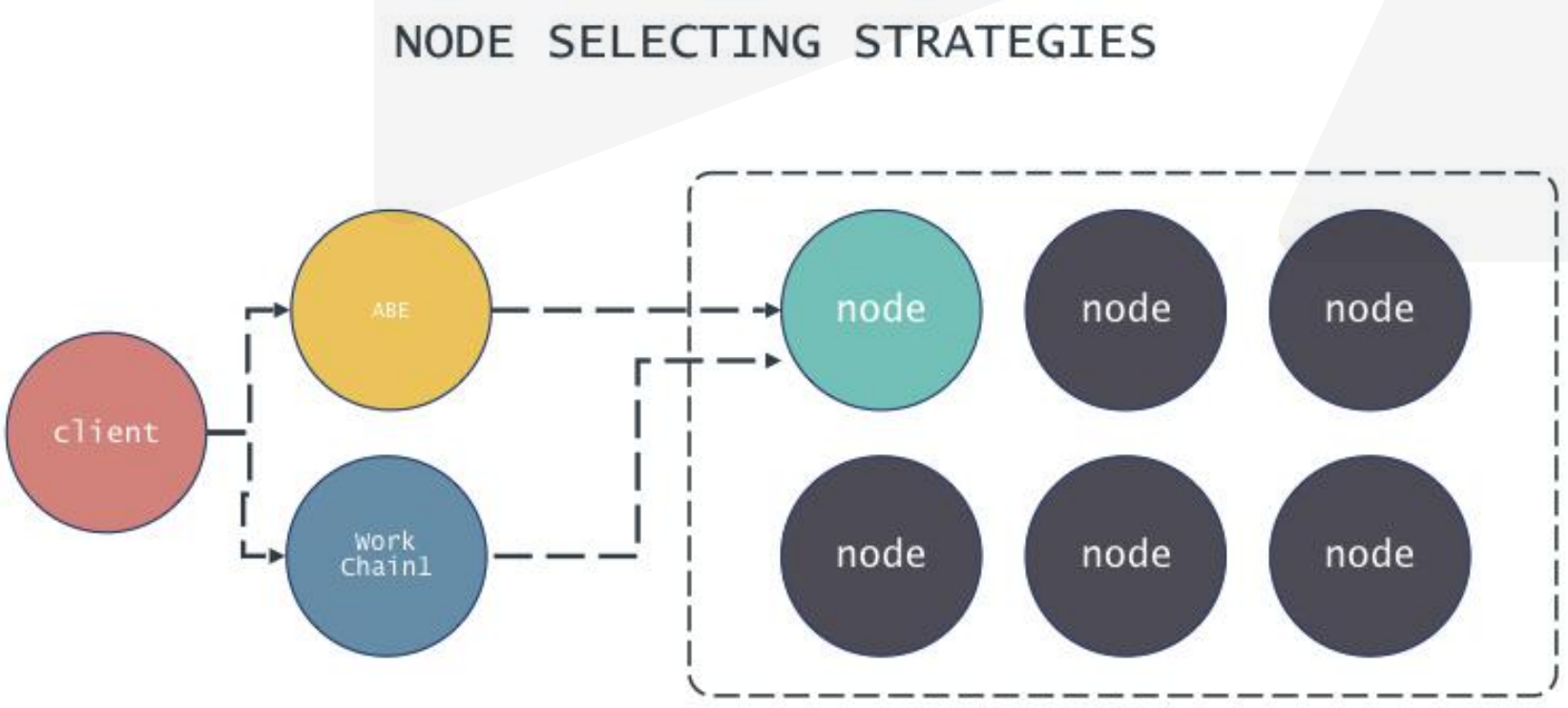
In Lambda DB, the clustering and partitioning features are also provided. Lambda DB is a unique database system at the cloud. In this distributed database system, possible network-wide broadcasting of data files may cause propagation and congestion of network messages due to oversized single cluster in extreme situations, Lambda DB has provided geographic location-based clustering and shard capabilities. Regardless of whether the node is located in the same computer room or in different computer rooms, network faults may result in partitioning of virtual rings. Upon partitioning, the nodes of Lambda DB join in the exit arrangement, resulting in the formation of new virtual ring around the nodes of each partition. The new virtual ring is formed completely automatically, needing no participation of the O&M personnel.

Suppose that there is a virtual ring, some of whose nodes are distributed in the European data center, while others are distributed in the Asian data center. Where there is an interruption of network connection between the two data centers, Lambda DB will form a new virtual ring in each data center. After a period of time, communication between the European data center and the Asian data center will be recovered, while the two virtual rings need to be merged. We can designate some special landmark nodes. All nodes in the ring without landmark node leave the ring at which they are located firstly and rejoin in the ring at which the landmark nodes are located. After a period of time, there is only one virtual ring remaining in the entire system.

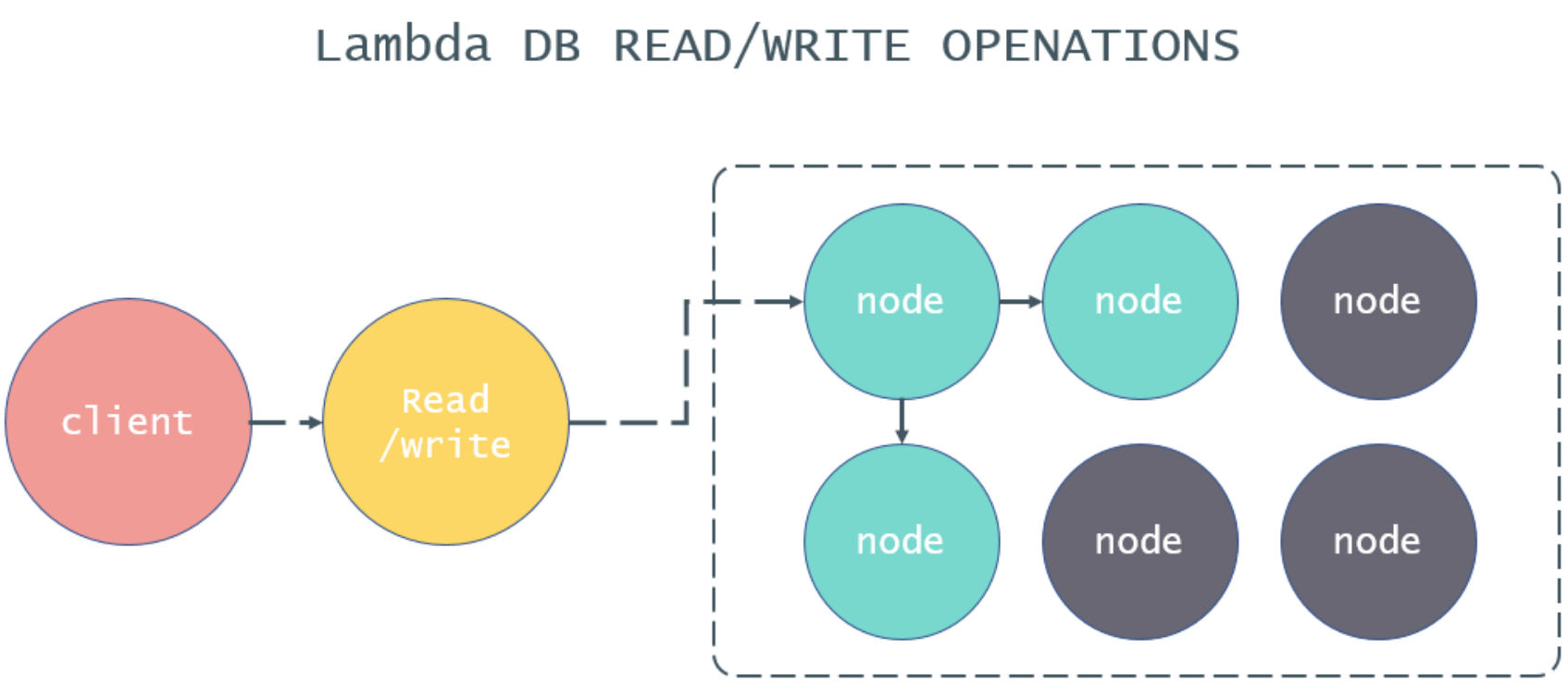


2.4 Read/Write of Data

Read-write requests for any Key from the client can be accepted by any node in a specified cluster of Lambda DB, which can be executed through RPC calls. When the client selects a node, it can obtain the authorization required for access via WorkChain3, and carry out route forwarding of communication via WorkChain1.



The node responsible for processing the read-write requests is called the coordinator, and the first N “healthy” nodes will participate in the processing of the read-write requests. Lambda DB ensures consistency in the system by using the Quorum consistency protocol, in which there are two configurable values: R and W, where R is the minimum number of nodes that successfully participate in a read request, whereas W is the minimum number of nodes that successfully participate in a write request.

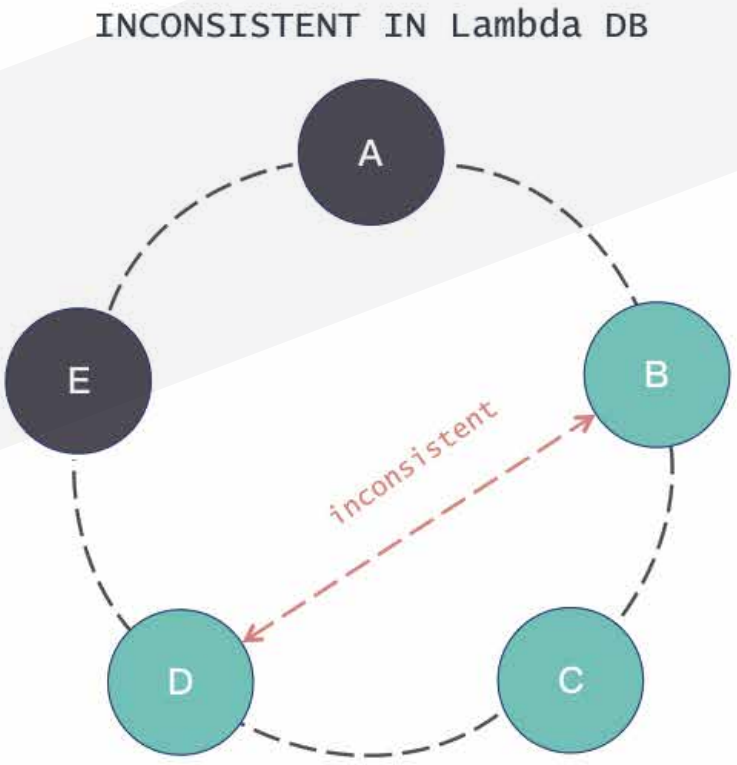


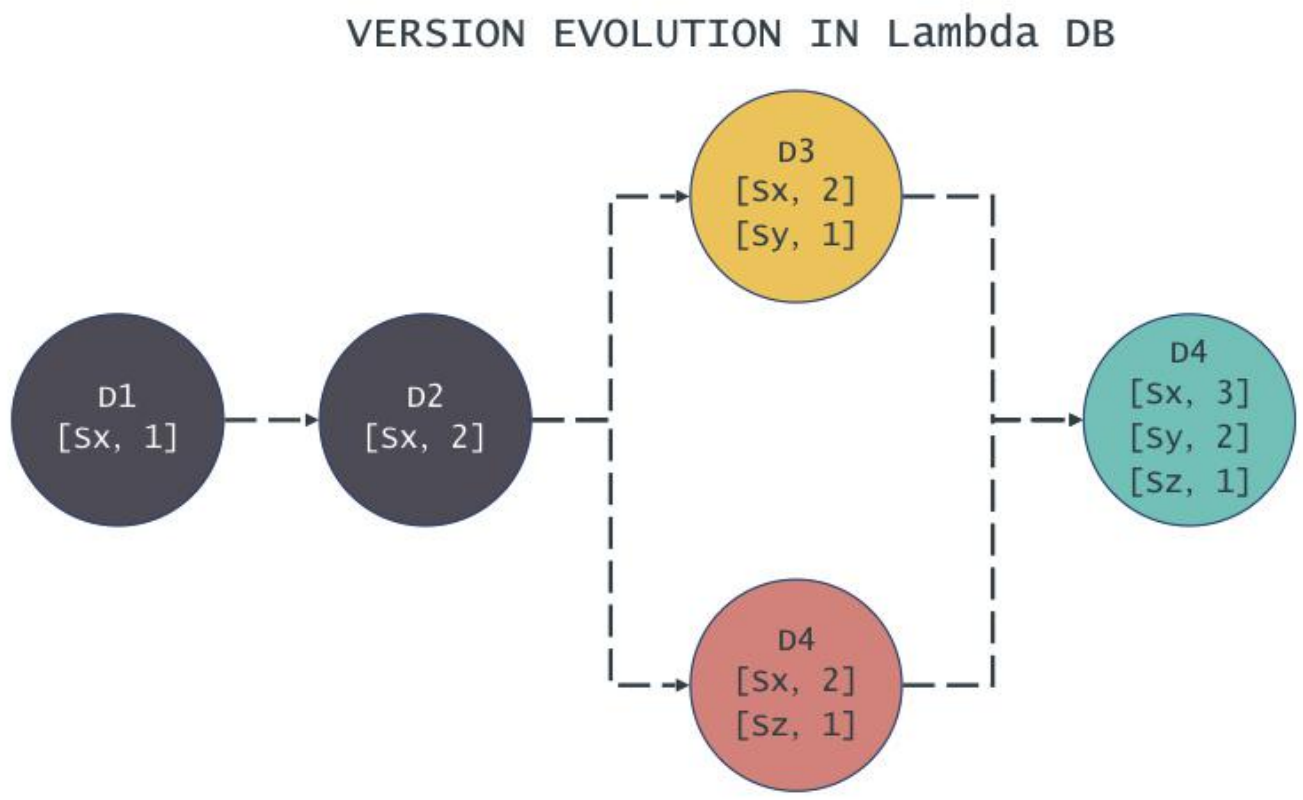
In case that R = 2, all read requests must wait until two nodes successfully return the results of the corresponding key, and the time for the read request depends on the node which returns the result the slowest. It is exactly the same for the write requests. When receiving the write request put() from the client, the coordinator will create a new vector clock, and then store the information about the new version locally. Later, the coordinator will send messages to the first N-1 nodes in the preference list until W-1 nodes of them return. Till then, the request ends successfully. The only difference between the read request get() and the above request is that if the coordinator finds that there is a conflict for data in the nodes, it will try to resolve the conflict and rewrite the results back to the corresponding nodes.

In case that R + W > N, Lambda DB ensures that read operation can always read the latest data. In the absence of node faults, it can be approximately understood as strong consistency.

2.5 Data Conflict and Vector Clock

The Lambda DB system provides final consistency, allowing us to asynchronously update nodes in the cluster.



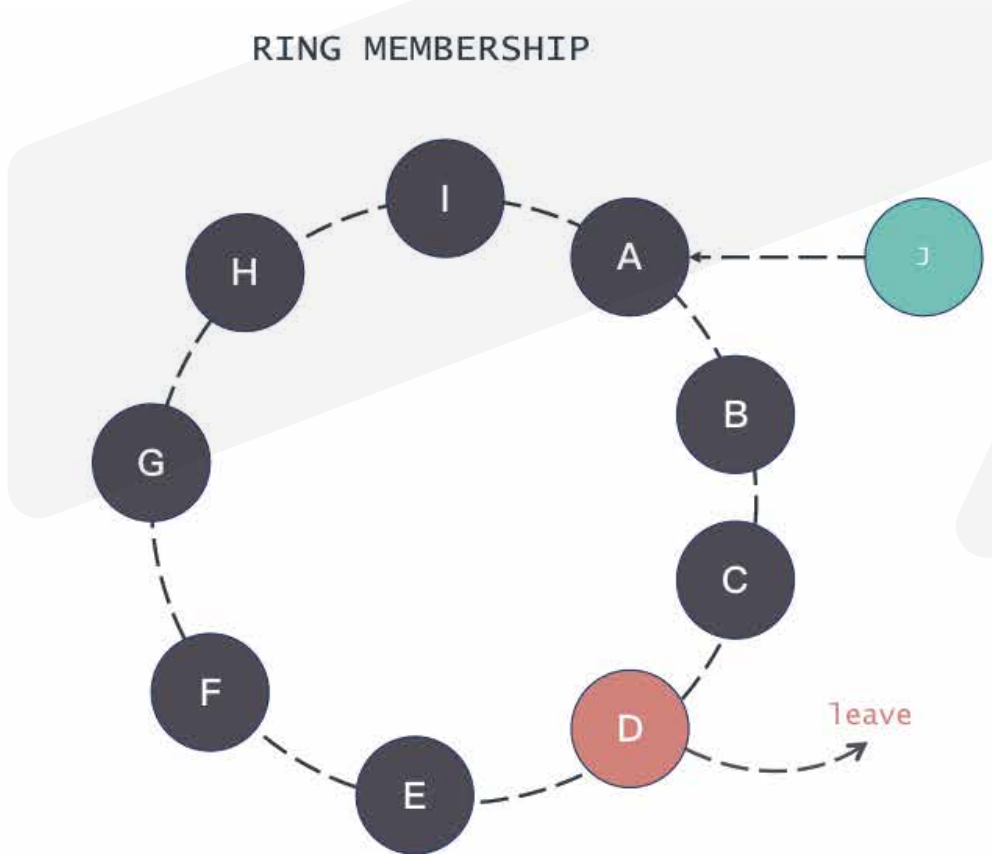


One or more vector clocks [Sn, N] are stored in the edition Dx of each object in the figure above. When writing data every time, Lambda DB will update the version number of the vector clock. The vector clock is [Sx, 1] when the node is written by Sx for the first time, and is [Sx, 2] when the node is written by Sx for the second time. At this time, suppose that neither the node Sy nor the node Sz knows that Sx has already written the nodes, but they receive requests from other clients, which write the same Key and generate different clocks [Sy, 1] and [Sz, 1] locally respectively. When the get() request is used in the client again, there will be data conflicts, and at this time, last write wins can be selected, which depends on the synchronization of the node clock.

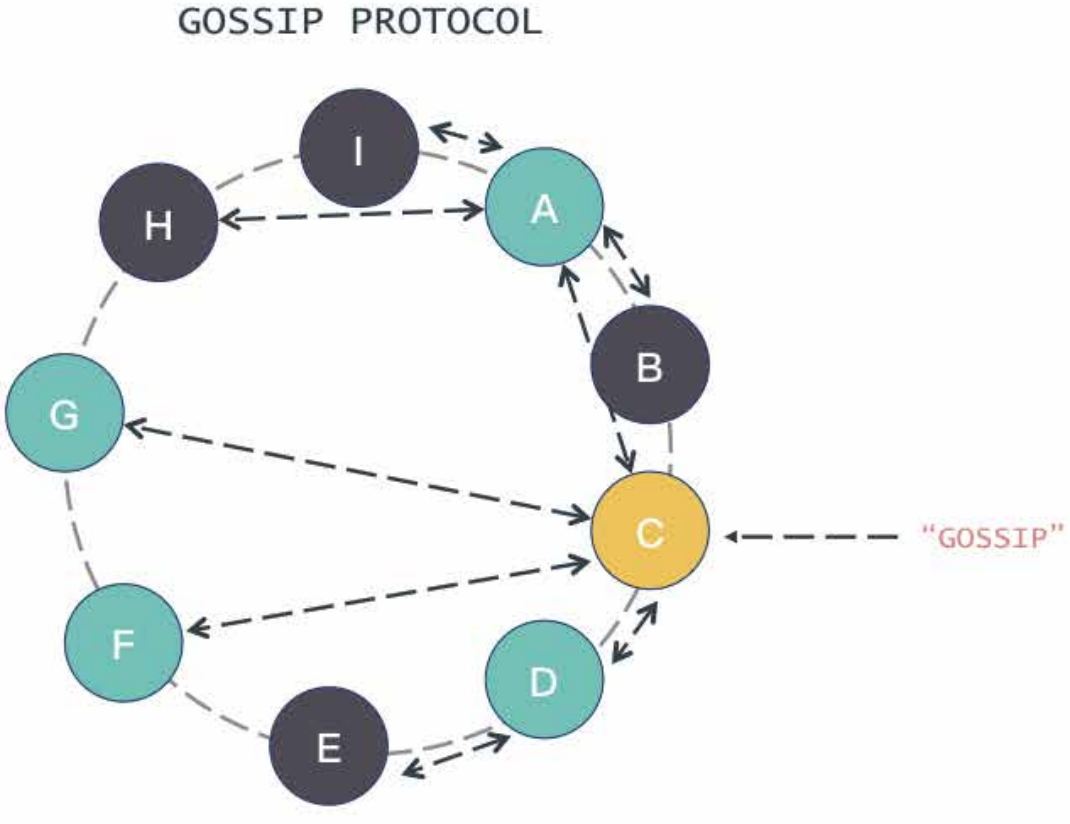
2.6 Addition and Deletion of Nodes

Node failure is common in a distributed system, while permanent failure seldom occurs to nodes for some reasons. Most of nodes usually suffer from system breakdown temporarily, and then quickly rejoin the system. Based on the above reasons, Lambda DB chooses to add nodes to or remove nodes from the system and update the changed information to the chain based on the pledge and consensus mechanism. Specifically, the mechanism is as follows:

1) When a node joins the system, the chain address corresponding to the storage node will launch a pledge transaction to the chain, taking the transaction time as the time when the node goes online.
2) When the node goes offline, the validator of the cluster where the node is located reaches a consensus, taking the consensus time as the time when the node goes offline.
3) The unit storage reward for the node is the average reward of the storage pool where the node is located, but the minimum online duration shall be reached.
4) The storage node gets the online reward by regularly guessing the block generation node of BFT.

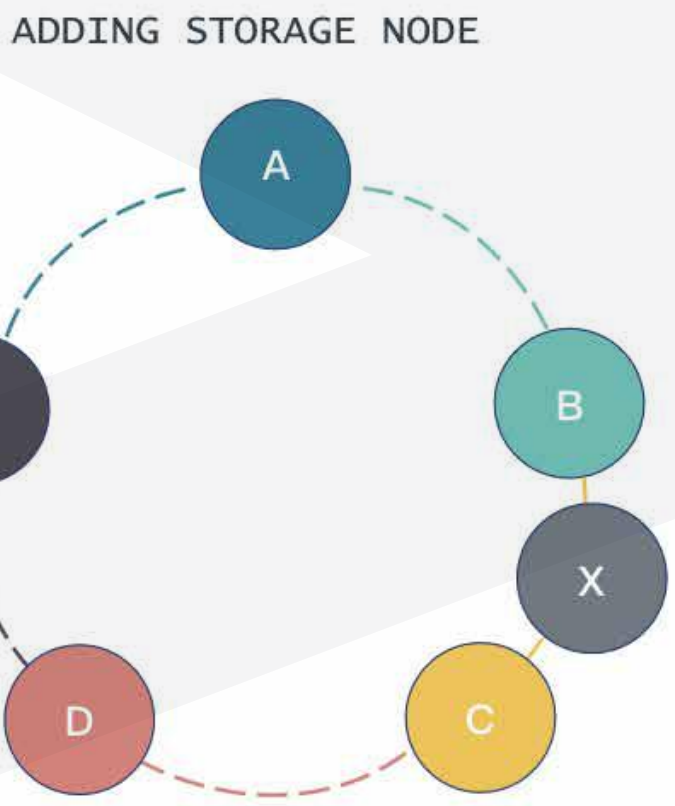


When added, any node in Lambda DB can be connected using the command-line tool or the browser to trigger a member change event, in which a new node will be removed from the current ring or added to the ring. Where there is a change for the node information, the node will notify the most nodes it can notify through the Gossip protocol.



In the Gossip protocol, two nodes for each communication will reach a consensus on node information in the current system. All nodes in the cluster of the entire Lambda DB will reach a consensus on member information eventually by transferring member information with each other. As shown in the figure above, node C will receive “gossip”, and then transfer it to the four nodes A, D, F and G to which it can be most accessible. Then, “gossip” will be transferred to the grey nodes in the system under secondary propagation. Up to this point, all nodes in the system have received the latest “gossip” message.

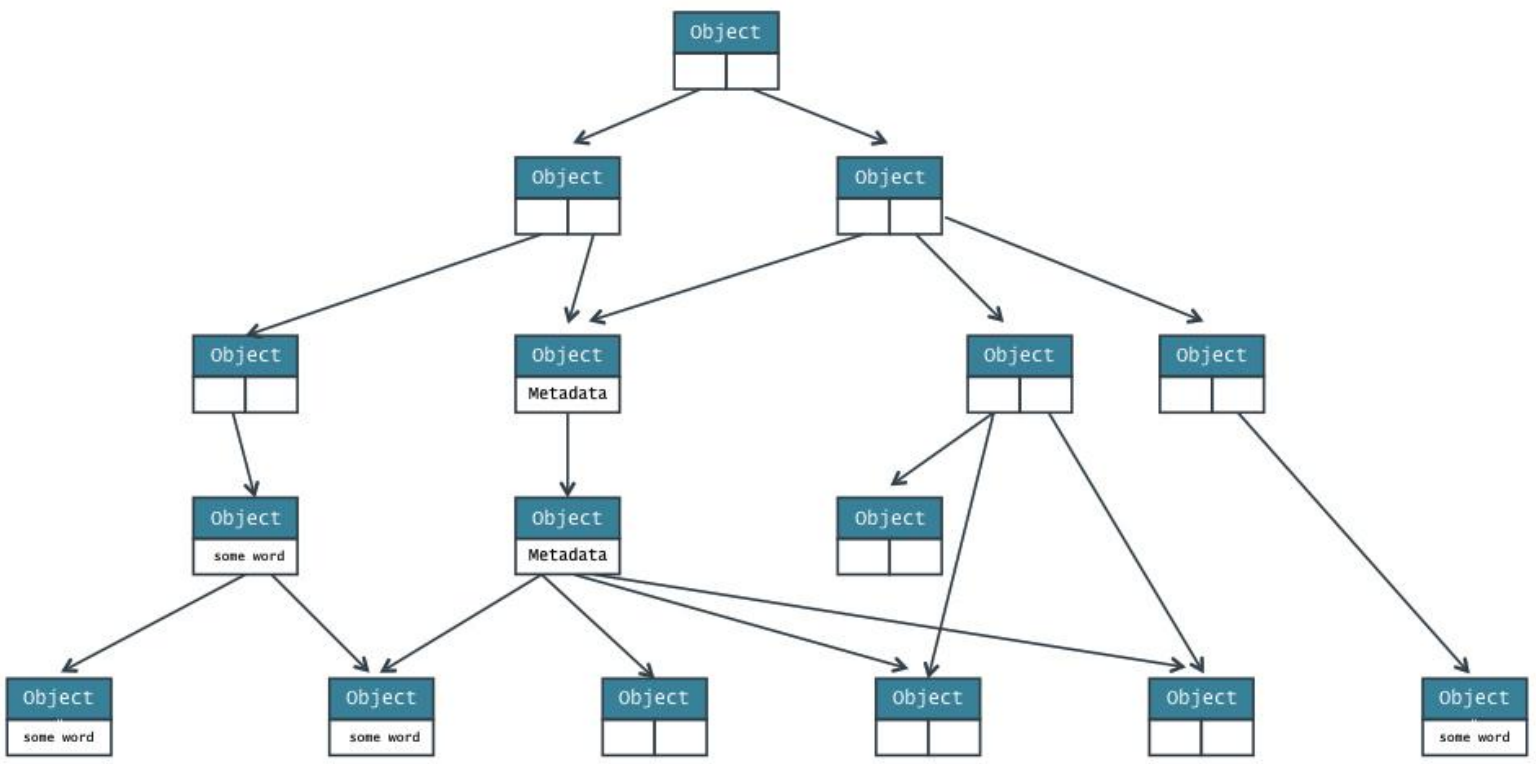
When we add a new node to Lambda DB, shard transfer will occur between the nodes. The new node joins in the X node and is connected to the Lambda DB database, which is allocated to back of the nodes A and B shown in the figure below.



The newly-introduced node X will accept one part of shards that three nodes C, D and E manage therefrom, i.e., the colored (E, A], (A, B] and (B, X] parts in the figure above, which will be subordinated to the nodes with the colors corresponding to the respectively colored parts before the X node joins in the cluster.

2.7 Duplicate Synchronization

During the operation of Lambda DB, there will inconsistency of data amidst different nodes in some cases, Lambda DB will ensure that information is stored synchronously in all duplicates using an anti-entropy strategy. In order to quickly confirm the consistency of data among multiple duplicates and avoid large amount of data transmission, Lambda DB carries out fast verification and transmission on data in different nodes by Merkle DAG, and at the same time, DAG can easily eliminate duplicated data to save the storage space. The flexible data storage method for DAG can be widely used for addressing, FS block exchange, KV data exchange, exchange of documents and other rich media in the Lambda system.



2.8 Implementation and Principle of Top-k Query of Data

In the scenario where time series data is processed, what are queryed are mostly the maximum and minimum values of data within a range of time. This belongs to a range query, which is also known as Top K query. Generally speaking, it is very difficult to support complex range query in DHT, which has become the theme of the front porch of scientific research in the distributed system. In our time series database, we have implemented support on range query, concatenate query and Top-K query, which is also why Lambda^2 TSDB is valuable. Due to limited space, we describe only the principle of implementing TOP k query. In the centralized and distributed systems, regarding the threshold algorithm TA [nepal and Ramakrishna, 1999] as the effective implementation of TOP k query, we have implemented a changed algorithm of TA algorithm in the DHT case, i.e., the DHTop algorithm, which is from the paper APPA [Akbarinia et al., 2007c].

In Lambda DB, the nodes store their tuples in DHT in two complementary ways: Tuple storage and attribute value storage. Regarding tuple storage, each tuple can be stored in DHT by taking its identifier (i.e., primary key) as a storage key. This enables it to find a tuple with the identifier like a primary index. Regarding attribute value storage, the attributes are stored in DHT separately, which may appear in equal predicate for a query or a query rating function. Therefore, it allows finding tuples by using the attribute values just like in the secondary index. There are two important features in attribute value storage:

(1) After receiving an attribute from the DHT, a node can easily obtain the tuple corresponding to the attribute value.
(2) The relatively close attribute values are stored in the same node.

In order to satisfy the first feature, the same attribute values will be stored together with those keys that are used to store the entire tuple. The second feature will be implemented by the domain partitioning concept as described below. Consider an attribute a and set Da as the domain of its value. Suppose that there is a total order < on Da, Da is divided into n non-null child domains d1, d2..., dn, making their union set equal to Da. The intersection set of any two different child domains is null, and for each child domain, v1 di and v2 dj; and if i<j , v1<v2, and the hash function is applied to the child domain of the attribute value. Therefore, the stored keys are the same for the attribute values that fall within the same child domain, which are also stored in the same node. In order to avoid the divergence of attribute storage, domain partitioning is accomplished by evenly distributing the attribute values in the child domains. In this technology, histogram-based information that describes the distribution of the attribute values is employed.

When this storage mock-up is used, the Top-k query processing algorithm, known as DHTop, will function as follows: Set Q as a given Top-k query, f as its rating function, and p0 as the node that initiates query for Q. Set the rating attribute as a set of attributes that are ready to be transferred to the rating function as parameters. DHTop begins with p0 and is processed in two phases: Firstly, it prepares the sequence listing of candidate child domains, and then it continuously obtains the attribute values of the candidate set and their tuples until it finds the first k tuples. The details of the two steps are as follows:

Algorithm: DHTop
Input:*Q*:top-k query; *f*:scoring function; A:set of m attrbutes used in f Output:*Y*:list of top-k tuples
Phase 1. For each scoring attribute α do Create a list L_α and add all sub-domains of α to it; Remove from L_α the sub-domains which do not satisfy *Q*'s condition; Sort L_α in descending order of its sub-domains;

Phase 2. end-condition := false;
For each scoring attribute α do in parallel
i := 1;
n := number of sub-domains in L_α;
While (end-condition = false) and ($i \le n$) do
Send *Q* to the peer *p* that maintains the α values whose sub-domain is $L_\alpha[i]$. *p* returns to p_{int} its values of α which satisfy *Q*'s condition, one by one in descending order, along with their corresponding tuple storage key;
v := the first α value returned by *p*;
While ($v \neq$ null) and (end-condition = false) do
Retrieve the corresponding tuple of *v* and compute its score. If it is one of the *k* highest scores, then record the tuple in a list *Y*;
If there are *k* tuples in *Y* whose scores are higher than the Threshold then set end-condition to true and return to the user these *k* tuples;
If end-condition is false then set *v* to the next α value returned by *p*;
If *v* is null (*i.e.* all values returned by *p* have been received) then set *i*:= *i* + *1*;

We have proved that DHTop can correctly process monotonic functions and a main category of non-monotonic functions.

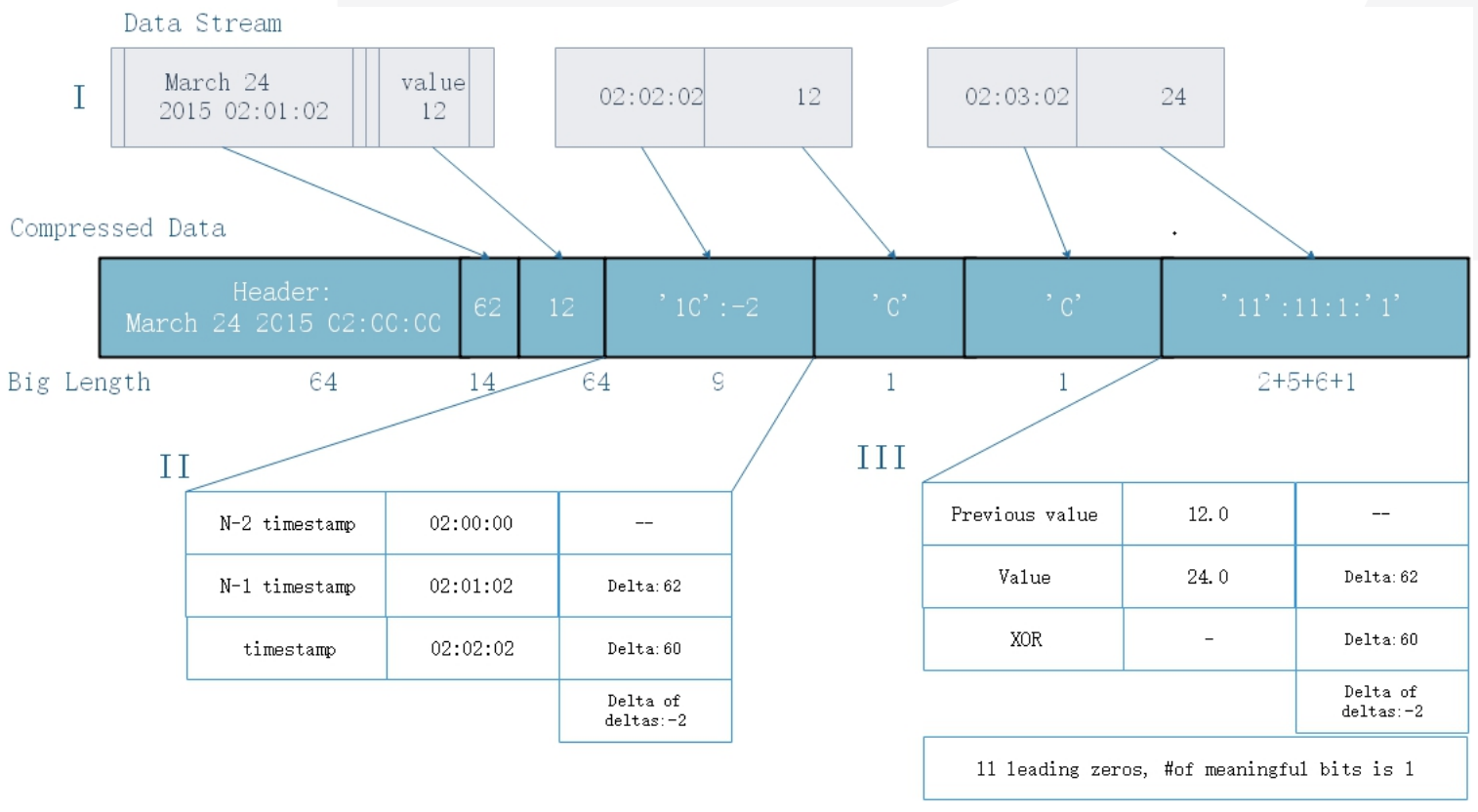
2.9 Data Collection Terminal-Lambda Agent

All distributed applications can implement data access by calling Lambda Agent.. Lambda Agent is a program running at the operating system level, which, essentially, is a Lambda's wallet where Lambda's blockchain ledgers and routing tables are stored. Under the simultaneous operation of multiple Lambda Agents, a fog network will be constituted, and at the same time, certain data verification capability can also be provided.

Our Lambda Agent SDK, in addition to accessing data, also provides a variety of free data collection and transmission capabilities, including performance monitoring, security analysis, data analysis, etc. Generally, such commercial products are very expensive, and the selling price of performance monitoring and security monitoring systems ranges from tens of thousands to millions of dollars. Now, we provide such products to users for free, who constitute a user group beyond distributed applications based on Lambda's development. We attract users to install our Agent system and provide computing and storage capabilities that users are willing to provide to obtain Lambda Cryptocurrency. This is a very good method to increase the overall computing power of Lambda, which also makes our wallet accessible to more users.

Memory-type TSDB

Built-in Agent implements a memory-type TSDB-Beringei which supports memory storage at a very high speed and ensures data constancy via the hard disk. The query of the storage engine is always processed in memory, providing extremely high query performance. Unless queries are required in the disk, disk operations are seldom required. Hence, reboot or migration can be implemented in case of short power-off and no loss of data. Beringei uses the extremely efficient data stream compression algorithm, by which the actual time series data can be compressed by more than 90%.



Performance monitoring

Agent includes the performance management module that meets the deployment requirements of cloud environment (public cloud, private cloud and hybrid cloud), data centers and large traditional IT infrastructure environment, and provides application performance monitoring solution for distributed, dynamic and agile environments which set strict requirements.

Upload of Metrics data

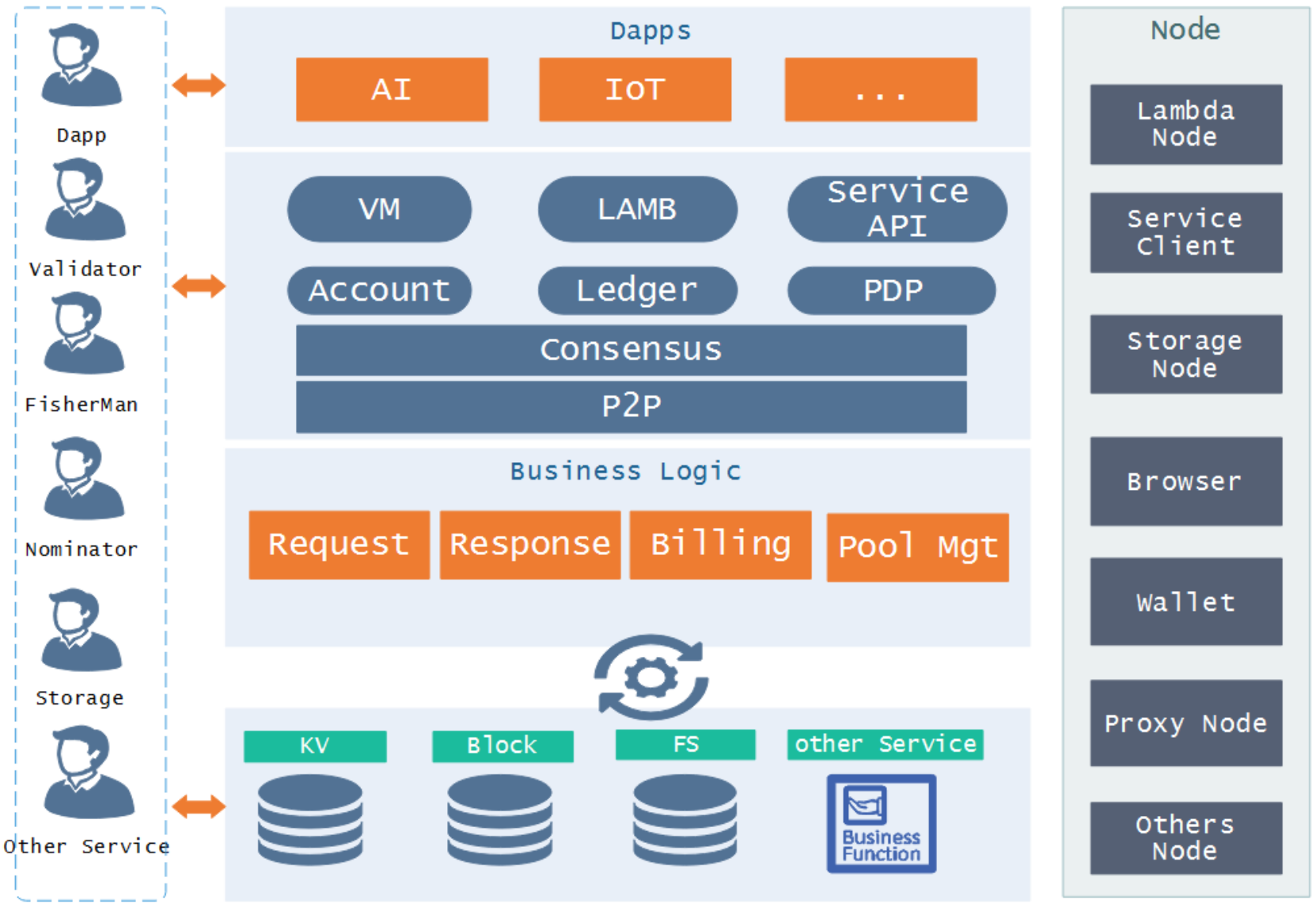
Any Dapp can upload Metrics data by calling Lambda Agent.

III. LAMBDA ECOSYSTEM

3.1 Lambda Economic Ecosystem

Unlike most blockchain applications, Lambda is a datastore infrastructure for blockchains, which is provided with a plurality of its own chains for charging, transactions, encryption and access control. The native token LAMB of the Lambda project creates memory and storage resources that consume the nodes. On the Lambda platform, resources that can be traded are mainly the addressing capability of format data. The fast addressing capability is a combination of the storage capacity of the hard disk and the memory size. Since the stored data is stored in the encrypted way, which is required to be deciphered when accessed by applications, CPU resources are also required to be consumed. When measuring the contributions of storage providers, we make decipherment on the chunk for a thousand times and return the deciphered chunk as the unit pricing unit.

In the Lambda ecosystem, the roles of participating parties are Dapp developers and project parties respectively, chain node participants, storage node participants and other participants, such as investors. The role of the chain nodes has been elaborated in the first chapter, mainly including nominators, validators and phishers. The difference between Lambda and other main chains is that we have our own business logic. We will implement processing and accounting on users' requests and returns and manage consensus and clearing within the storage resource pool.



The logic of the Lambda's data storage pool consists of four roles: Storage resource provider, storage resource buyer, community developer and data buyer, which constitute the ecosystem interdependent with Lambda. In addition, in terms of the business model, these four roles may be either individual users or enterprises.

Group	Lambda features	Incentive to participate
Tenant	Lambda offers tools to execute compute-intensive tasks.	Tenant get access to affordable and scalable solutions, which combine hardware and software.
Users	Lambda combines and utilizes (almost) any kind of existing computing hardware.	Users get paid for renting out their hardware
Software Developers	Lambda is a flexible platform to deploy and monetize software	Software developers use Lambda as a distribution channel, associated with access to hardware
Data analysts	Lambda DB is a time series database	Data analysts can search and subscribe data source from data producer,and then they can buy and analyze this data.

Storage facility provider

Storage capacity is mainly provided by the user group, which may be constituted by persons and organizations that have installed Lambda clients. From the renting of PC's idle resources by human users to the complete extraction of the computing power and storage by the large data center, we can drive countless computing and storage resources from small to large. Currently, both human users and many IDC cloud computing centers have already been confirmed to join in Lambda suppliers, including tens of thousands of NAS devices of NAS platform operators like Phicomn and hundreds of thousands of PC devices of many Internet cafés and Internet café platforms. These suppliers are motivated to join in Lambda as they can acquire the rewards of digital

currency LAMB from the completed tasks. For suppliers, the current low-configuration hardware and PC resources cannot obtain bitcoin-like digital currencies that require hash computations via proof of work. Moreover, the difference between the computing power of the conventional CPU and GPU and ASIC is getting farther and farther, and even the proof of work (POW) is also applied in storage tokens like Sia. It would be the optimum choice for PCs and IDCs to acquire digital currencies via the proof of work (POS) applied in Lambda and storage. Meanwhile, the Lambda's probe program can monitor the running of related infrastructures of infrastructure providers, resource consumption, and even the experience of end users in real time to ensure that the rental of devices will not pose any impacts on the original service.

Demands for data storage and transactions

Today's distributed application developers will increasingly feel the value of the Lambda project. On the Lambda platform, not only they can easily develop distributed applications by taking Lambda as a Baas service, enjoying a variety of cost-effective performance analysis and security assurance services in the price much lower than that of the conventional cloud hosting and cloud storage, but can also sell the value of data, and the return of data sales may be much higher than the costs of data.

The Lambda project is exploring a safe and efficient decentralized data sharing mock-up, in which Lambda firstly extracts multi-layer metadata information from the shared dataset and establishes domain indexes through each consensus node to find the connectable dataset efficiently. Secondly, starting from the format of transaction records and the consensus mechanism, Lambda establishes blockchain-based data transactions to achieve transaction transparency and avoid collusion and other corrupt conducts. Finally, Lambda writes a computing contract according to the computation demands of the data demand side, ensuring the computing and output privacy of data owners with the assistance of secure multi-party computation and differential privacy technology. The main technologies used herein include distributed hash tables and locality-sensitive hash.

Software and microservices

The Lambda team will introduce more features into the Lambda platform in future, but it is critical for the idea to develop its own Lambda applications together with other software developers. The quantity and quality of such applications are one of the key factors for Lambda's success in future. The data storage and data analysis services that we provide in future may go toe-to-toe with the existing mainstream cloud platforms. It is easy for other open-source DUs to open their own open-sourcing capabilities of software in the field of blockchains as the WorkChain of the Lambda system. We will drive the world's blockchain infrastructure to develop forwards through the Lambda Foundation.

3.2 Lambda Cryptocurrency LAMB

Mainchain consensus algorithm: POS

We choose Nominated Proof-of-Stake NPos as the consensus algorithm.

Signature algorithm: Variable signature type

Each public key has a specifier (an array with 16 bytes) and a byte slice containing public key encoding. The specifier is used to specify which one signature algorithm will be used when validating signatures. Each signature will be a byte slice whose encoding can be determined by examining the specifier for the corresponding public key.

Alternative signature algorithms include ed25519, ECDSA secp256k1 and Schnorr secp256k1

Lambda Cryptocurrency (LAMB)

- **Lambda Coin:** Lambda Coin (LAMB) is the core of the Lambda project and the main weapon of the Lambda project to subvert the development mode of the traditional commercial closed source software. Under the business model of traditional software companies, there are a lot of transaction costs and organization costs. Software companies have to employ a lot of labors to obtain users, provide services and solve problems for value transfer and delivery, which can be completed in the more opti mized way in the Lambda project.

- **POS consensus mechanism:** The POS proof of work is applied in the consensus mechanism for the Lambda chain, and the validator can win Lambda rewards after completing hash computations. In addition, the database storage nodes store data files with specified sizes at specified time, winning Lambda Coin rewards according to the contract;

- **UTXO mock-up:** The Lambda chain's ledger structure uses the UTXO mock-up while supporting the Account mock-up. The design is implemented in a manner similar to Qtum.

- **Cross-chain transactions and BCP protocol:** As a member of the UCE Economic Systems Alliance, our design follows the BCP protocol and supports atomic cross-chain transactions and atomic contract transactions within the UCE system.
- **EVM and smart contracts:** We will implement support on smart contracts on the chain and provide complete formalized verifi cation capability for contracts.
- **Total quantity:** The total quantity of Lambda Coin is 10 billion
- **Pledge:** Validators and nominators need to pledge LAMB, with the total amount of pledge approximate to 8-12% of the total quantity of LAMB.
- **Circulation:** In the first year, 40% of Tokens still wait for block generation rewards, 20% of Tokens are locked in the Founda tion's account, 10% of team Tokens and 20% of private equities have not been unlocked during the freeze period, and the maxi mum Token that is circulated on the market is about 10%.
- **Rental:** Users who rent cloud the Lambda databases need to purchase LAMB at the exchange.
- **Transactions:** For persons who carry out data transactions at Marketplace, the Buyer needs to purchase LAMB at the exchange.
- **Rule:** In the Lambda Coin's distribution rule, 10% for founding team, 30% for fundraising, 20% for foundation, and 40% for block generation rewards. Coils will be mined completely for 20 years. After 20 years, the total quantity of tokens keeps growing at a slow rate of about 0.5% per year.

3.3 Block Generation Rules

MainChain Block and ShardChain Block

- The size of the largest block is 32*10^6 (32e6) bytes, which is 32M. There is no limit on transaction sizes, but transactions must be able to be accommodated in the block, and the validator will limit the size of each transaction.
- Each block has a minimum allowable time stamp. It is determined by taking the intermediate time stamp of the first 10 blocks. If the number of the previous blocks is less than 10, the genesis timestamp is required to be used repeatedly.
- The rule for generating block ID is: H (parent block ID + 64-bit nonce + block Merkle tree root node).
- For the valid block, the block ID must be lower than a certain specific target.
- The block generation time for both MainChain and ShardChain is expected to be 5 seconds.
- MainChain contains the hashes of all ShardChain blocks, and ShardChain contains the block header of the previous MainChain block.

Transaction

A transaction is mainly composed of the following objects:

- LAMB Inputs
- LAMB Outputs
- File Contracts
- File Contract Revisions
- Storage Proofs
- Miner Fees
- Transaction Signatures

The sum of all LAMB inputs must be equal to the sum of all miner fees, LAMB outputs, and file contract payouts without surplus. Unlock hashes are stored in the above objects. The unlocked hash value is the root of the Merkle tree of the "unlock condition" object. The unlock condition contains a time lock, a plurality of required signatures and a set of public keys that can be used during signature. The root node of the Merkle tree of the unlock condition object is the root of the Merkle tree composed of the timelock, the public key (each public key corresponds to a leaf node) and the number of signatures. Enough signatures are required, and only by being at least equal to the value of the timelock, can the height of the blockchain meet the unlock condition.

The unlock condition contains a set of public keys, each of which can only be used once when the signature is provided. The same public key can be included twice, which means it can be used twice. The number of the required signatures indicates how many public keys are required to be used to verify the input. If the number of the required signatures is "0", the input means “anyone can consume.” If the number of the required signatures is more than that of public keys, the input is non-consumable.

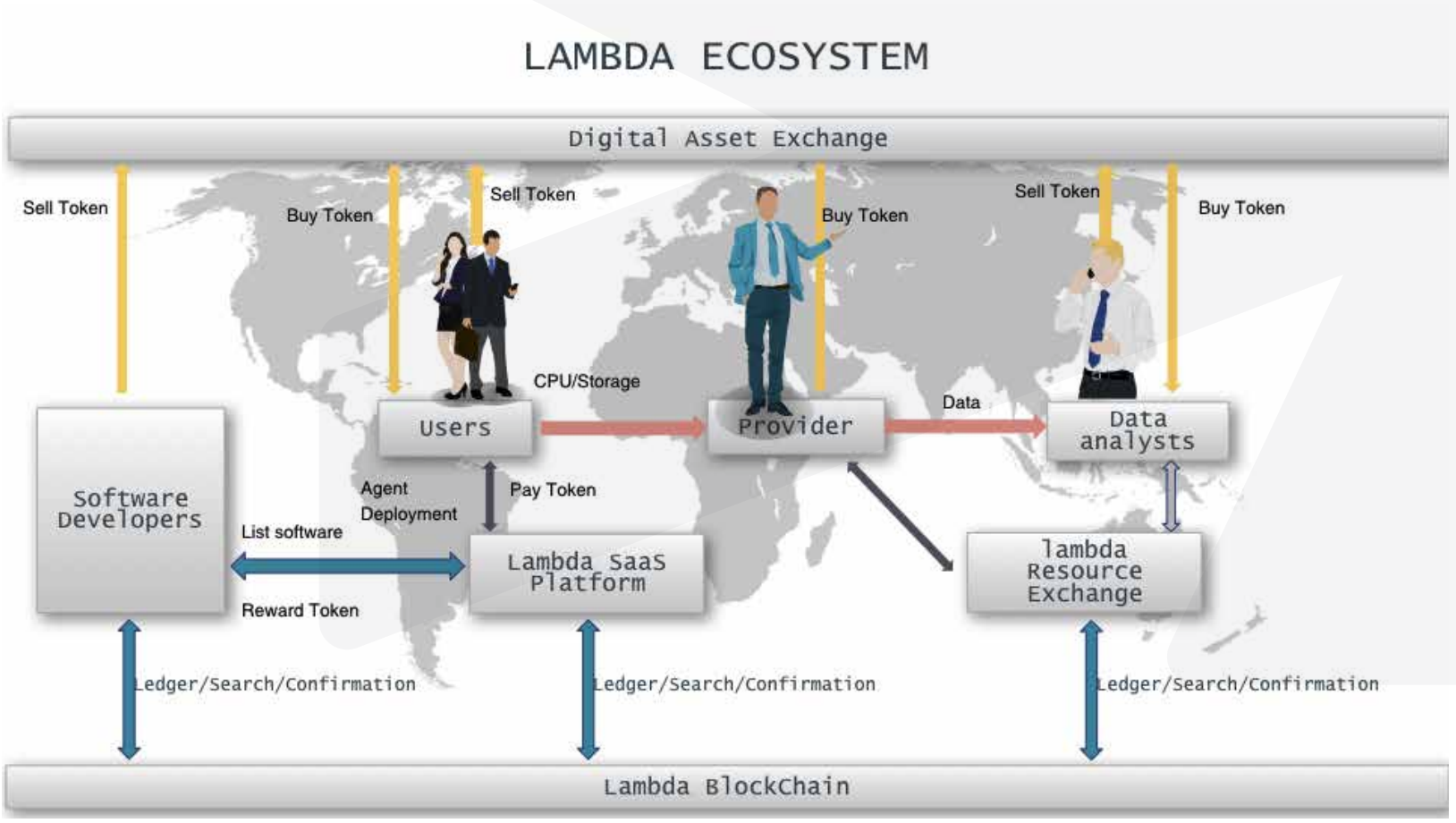
Contract

The data storage contract is a protocol between data storage providers and their customers. The customers choose the virtual

data nodes provided by Lambda to store data and carry out corresponding database retrieval and other operations. The protocol concluded and signed between the customers and the storage providers is a continuing protocol, and installment payment will be made according to certain conditions. Unlike the file storage, data stored in the database will be slowly increased, hence, the amount of each payment made by the customers to the database service provider for each node is:

1) Total convention amount* Change cycle* Size of current storage space/Total duration * Total convention space
2) The payment cycle depends on the cycle of proof of data retention of validator nodes on data storage nodes.
3) The core of a data contract is Merkle DAG for the data storage file.

3.4 Lambda Economic System



The role of Lambda's economic system is composed of the following:

1) Chain node

- The chain node is further divided into the validator, the nominator and the phisher. The chain node mainly provides the ability of routing, accounting, control and encryption to service access.
- The benefit of the chain node mainly comes from the mining revenue of the staking transaction of POS. In addition, the phisher can make verification on validator’s accounting information and the storage information of the storage node, gaining benefits

2) Storage node

- The main revenue of the storage node comes from the user's payment
- The storage node needs to pledge part of funds in advance and acquires a share of the shared benefits of POS
- The storage node needs to periodically send heartbeat information to the validator, and one who is fortunate can win rewards. This design is mainly to motivate the storage node to be online.

3) Users

Users are persons who purchase and use the storage node. Generally, they are other application chains and Dapps.

4)Investors

Investors invest in the Lambda project and share the benefits arising from LAMB rise

5)Exchange

Digital currency exchange

3.5 Platform Capability

3.5.1 Reward mechanism for fixing and submitting bugs

Software users encountering problems during use will submit Bugs to the community. The code developers in the community will sequentially fix Bugs according to the influence terrain and degree of importance of code bugs. The fixed Bugs be made into the installation package in the form of Patch, which is put into the community for downloading by users.

Under this model, because of inconsistency in the process and the dependency of fixing to code bugs on persons, infrastructure

software has not been able to be upgraded to the latest version yet, which will pose a great impact on the overall availability, performance, and security of the system. In the Lambda project, users can post rewards in the form of smart contracts. It is required for developers to fix upon the consumption of certain LAMBs. Upon fixing, Bug patch will be automatically installed in the user system through the Lambda Agent Release Automation module without manual operations.

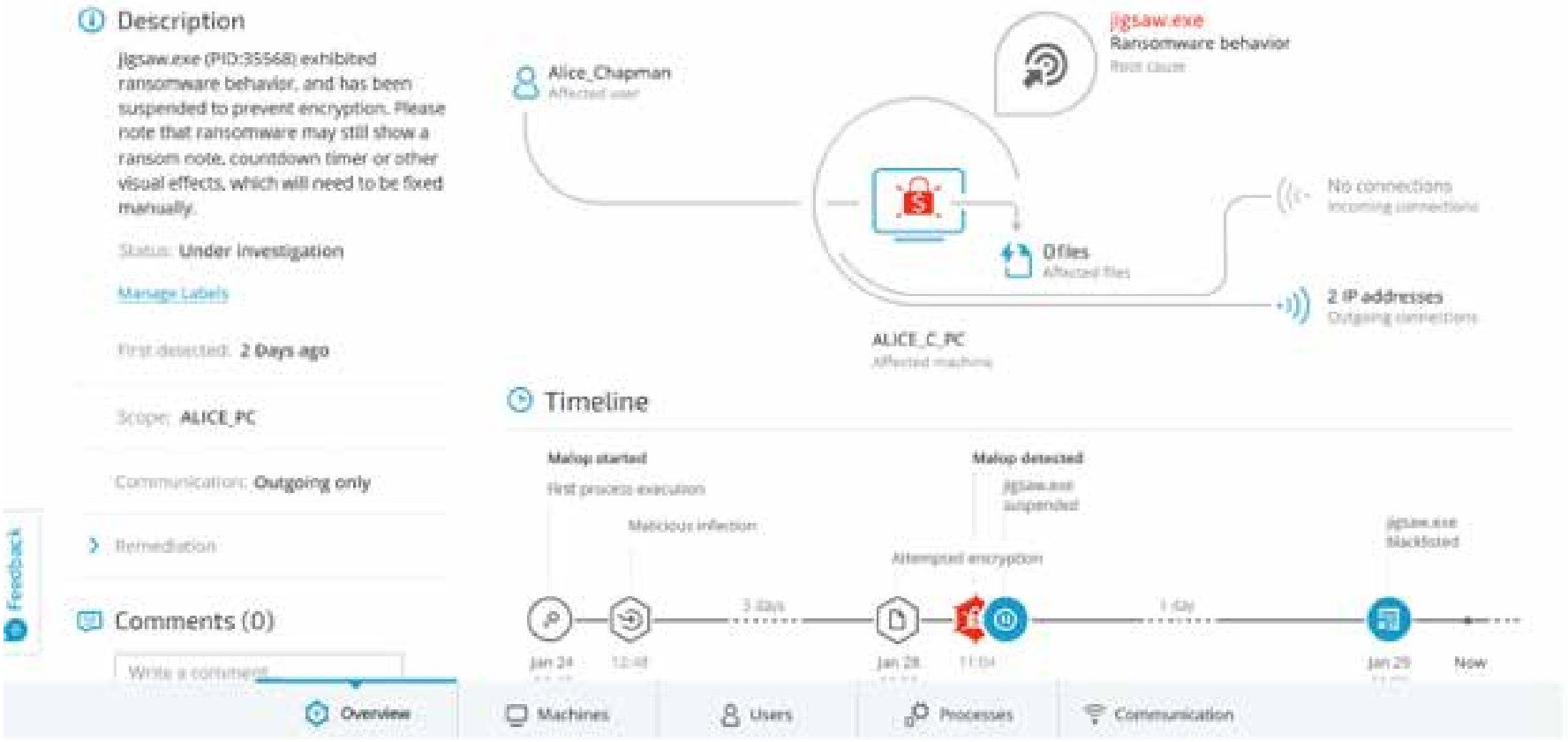
The users who originally submit the bugs will win LAMB rewards based on the subsequent use of Patch by other users.

3.5.2 Reward mechanism for performance improvement

Status	Priority	Summary	Last Changed By	Created	Duration
Resolved	SEV4	Provider failure in payments gateway SHOW DETAILS (30 resolved alerts) #495	Lucy Green	at 1:41 PM	00h 03m 19s
Resolved	SEV2	Clients cannot connect to payments API SHOW DETAILS (15 resolved alerts) #486	Frank Hamm	on Aug 25, 2017 at 4:37 PM	00h 04m 15s
Resolved	SEV3	Appdex low in payments gateway SHOW DETAILS (15 resolved alerts) #482	Lucy Green	on Aug 25, 2017 at 2:05 PM	00h 03m 00s
Resolved	SEV2	Appdex low in payments gateway SHOW DETAILS (27 resolved alerts) #462	Zayna Shah	on Aug 25, 2017 at 9:15 AM	00h 14m 19s
Resolved	--	Appdex low in payments gateway SHOW DETAILS (15 resolved alerts) #453	David Liu	on Aug 24, 2017 at 10:02 PM	00h 07m 28s

3.5.3 Automatic protection of security vulnerability

Lambda Agent further collects various security data. Then, it reports to the security community. We will check data with AI and summarize secure data. Where there are new or rare security issues, it will report to the company's eco-security experts. If the security issues are confirmed, i.e., a successful CVE being publicly announced, the first reporter will be rewarded with LAMB coins



IV. TECHNOLOGY ROADMAP

4.1 Cycle

As a group of programmers who are struggling in the research and development of infrastructure software and the frontline of the

open source community, we contribute codes to multiple open source programs under Apahce Foundation, and our team members get very deeply involved in Camel, Akka, Drill and other programs. We further actively drive the development of open source and community in China, establishing the Druid China subscriber group and Clickhouse China subscriber group.

Our development team has long been committed to developing software that collects massive data for storage, analysis and presentation under the Agent mode. Over the past few years, we have successfully released several products based on the probe technology. For example, Cloud Insight and Application Insight. In the coming days, we will gradually apply technologies accumu lated for many years to the Lambda project.

Phase I Lambda Technological Demonstration and Research (as of Q4 of 2017)

- To make deep research on the technical framework of each public blockchain and protocol layer, and the core team has read a large number of codes and papers (see appendix)
- To demonstrate the computing and storage mock-up for massive data under Permissionless environment
- To demonstrate the combination of OAS business model and Token economy

Phase II Lambda Technological Demonstration and Research (Q1 of 2018)

- To choose the appropriate blockchain consensus mechanism and determine the high-speed chain design using the Sharding scheme as well as the HoneyBadgerBFT consensus algorithm for subchains
- To determine multi-subchain design for keeping accounts respectively with Request and Reponse.
- To complete structural design for the separation of chains from libraries
- To determine provable data possession (PDP) and the access control and encryption mechanism for ABE
- To determine the technical route and the technical scheme
- To complete technical white paper

Phase III Lambda Core Component Development (Q1, Q2 and Q3 of 2018)

- To carry out evaluation test on the performance of libp2p and devp2p and modify codes
- To implement the underlying chunk storage system
- To implement the closed-source development of Lambda Chain
- To carry out technical verification on Lambda FS
- To carry out community development on Lambda FS and open source in real time
- To implement the Lambda database based on Lambda FS
- To develop Lambda Agent based on hotspot languages to facilitate integration with the existing applications.

Phase IV Lambda Chain Test Network Development (Q4 of 2018)

- P2P network development is used to synchronize blockchain ledgers and store data synchronously.
- Development of virtual machine and development and verification of built-in contracts
- API interface, RPC interface, command line and development of peripheral tools
- Open source Lambda Chain

Phase V Test Network Construction and Development of Lambda-based Example (Q1 of 2019)

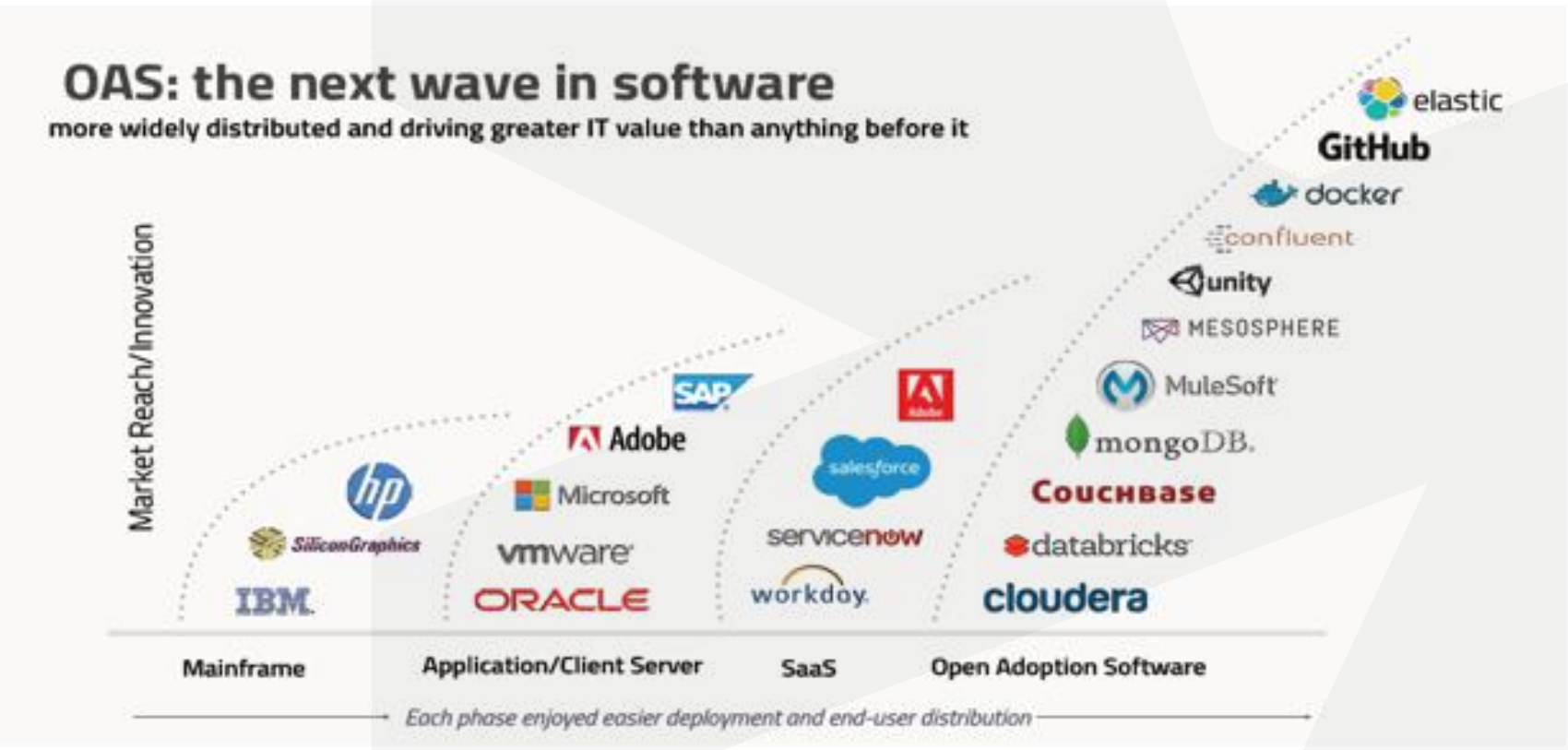
- To construct a large-scale test network based on the community for all-round testing.
- To issue LAMBs for encouraging early community enthusiasts to develop Lambda-based application examples
- To carry out rewarding test and amendment on security and service vulnerability on application examples based on test networks.

Phase VI Expansion of other data service capabilities and data transactions (Q2 of 2019)

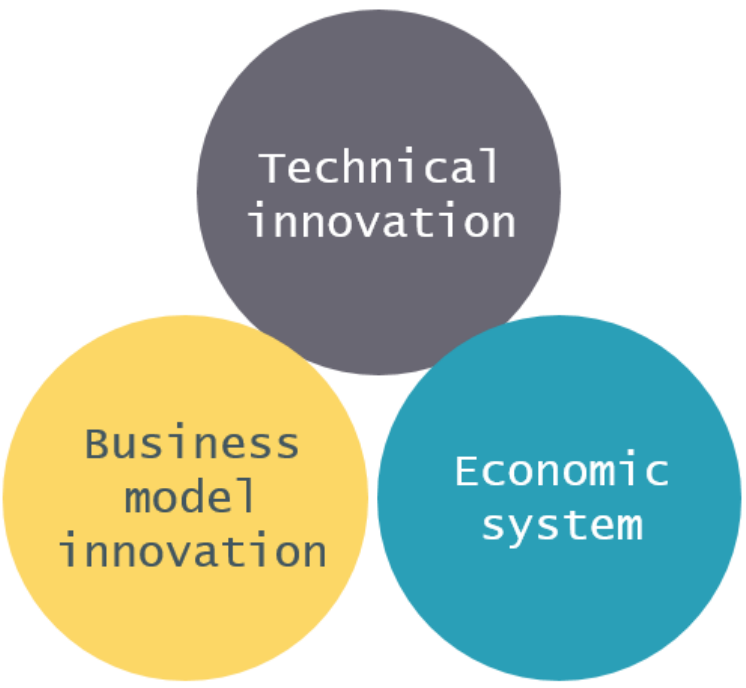
- It will be launched by the Foundation for encouraging more database types and service capabilities to access Lambda ecology
- To explore data transaction capability based on Lambda ecolog

4.2 Our Idea

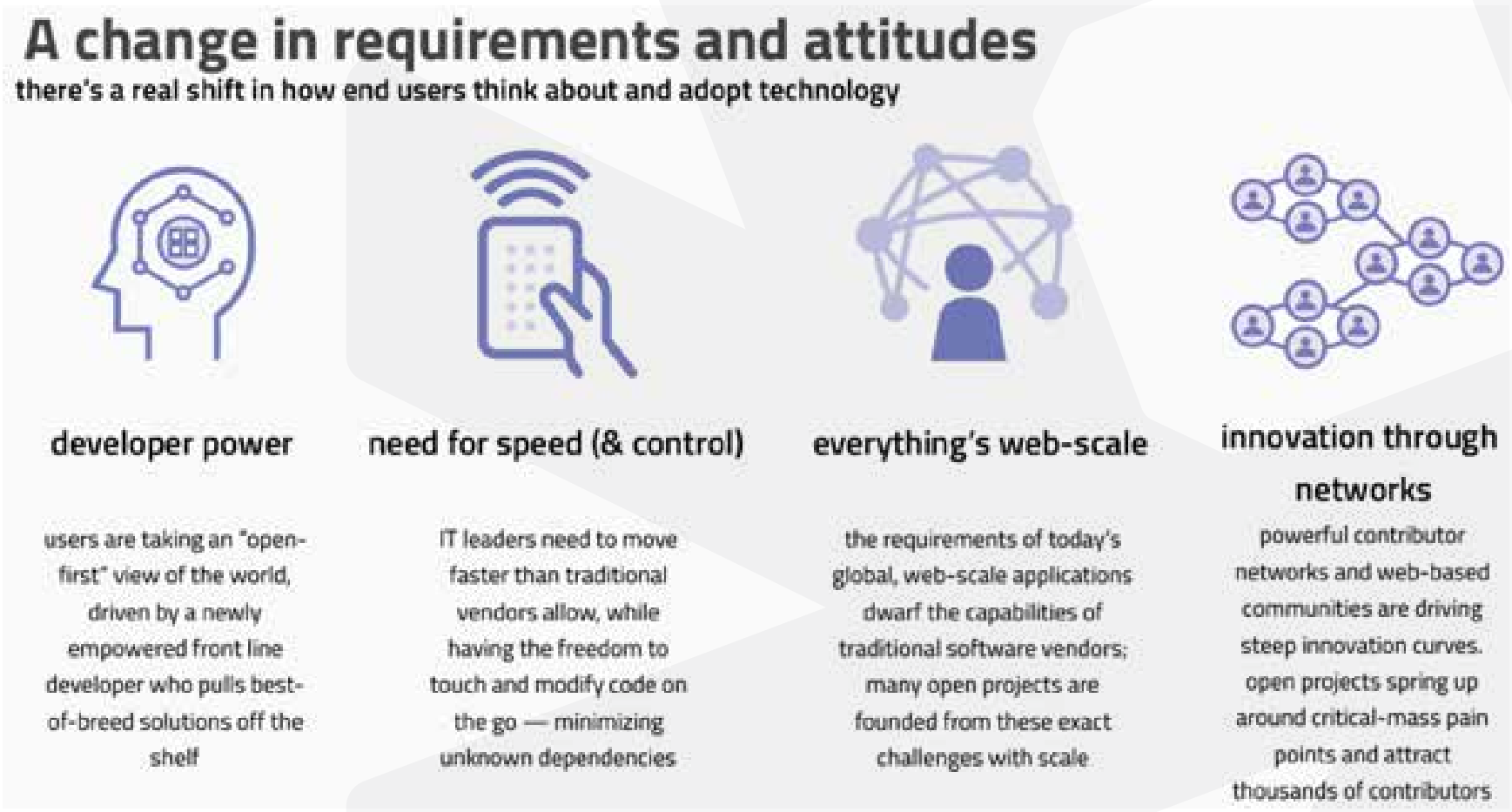
We firmly believe that the OAS mode will be the mainstream development way of all sorts of infrastructure software in future. Under the mode of community collaboration and Cryptocurrency incentive, the software field will certainly usher in major changes. But it is not enough for the OAS mode, and OAS plus blockchains will be the cooperation way with perfect integration for large-scale developers.



Currently, most of the public blockchain projects have only one public blockchain, such as Ethereum or EOS, including the follow-up Polkadot, etc. In addition, the infrastructures, if necessary, is also provided around the chain, i.e., providing decentralized domain name resolution, web disk, files, etc. At this time, the chain itself is the purpose, but our project is not like this, or not only that. The concept contained in this project goes beyond the current blockchain project. Our project is not merely subjected to technical innovation in the blockchain field, but more is the combination of technical innovation, innovation of business model and economic system. If the economic mock-up is not introduced in bitcoin's system, the consistency of the distributed system will never go beyond the raft, paxos and pbft mock-ups. Therefore, if we, only from a technical perspective, regard the implementation of the distributed database, this issue will always be controversial. We are going to forged ahead a new road rather than going to establish a database software company.



The starting point for our thinking is not the technical facilities for the blockchain, what we will consider more is how to establish a Unicorn Corporation with valuation of more than USD 1 billion in the field of China’s infrastructure software. The new mock-up has been born around the world, i.e., the community-based OAS mock-up whose the full name is Open Adoption Software. Just as bitcoin introduced the economic system into the distributed field, the OAS mock-up brings a new term to the field of global software for the first time, i.e., network effect.



We hope we can forge ahead a new road under such new model in the field of China's infrastructure software.

V. LAMBDA MANAGEMENT

5.1 Lambda Creation

Lambda Foundation was established in Singapore. The Foundation will dedicate in Lambda project. Oversee the project development. The General Meeting of Members shall be at the highest authority of this Association.

5.2 Lambda Council

The Council consists of 5 members. The Council:

(a) To carry out the Rules of this Foundation and the resolution of the General Meeting.

(b) To appoint or to remove the sub-committees.

(c) To take a lease of any equipment or premises or any place in the Republic of Singapore as may be required on behalf of this Association.

(d) To determine from time to time the amount of entrance fee and monthly subscriptions.

(e) To manage all urgent and important affairs or matters.

(f) To manage the Foundation's properties and shall file a list of members with the registrar, whenever changes occurred.

5.3 Community Governance

Owing to limited space, we will not discuss the community governance rule in details, but we promise to:

I. Issue the white paper on community governance rule as soon as possible

II. Ensure the open and transparent ledgers throughout the Foundation

VI. CORE TEAM

Origin

We are a group of programmers who have extensive experience in software R&D and open source community. We contribute codes to various open source projects under the Apahce Foundation, such as Camel, Akka, Drill and so on. Our team members are deeply involved in all of the projects. We are also actively promoting the development of open source and communities in China and have established China Users Groups of Druid and Clickhouse.

We met at OneAPM's Infrastructure Department. OneAPM is an APM SaaS company and has won a number of well-known VC investments. The business feature of OneAPM is application performance monitoring. Therefore, there will be massive data sourc-

es converging from various mobile terminals, browsers, and servers to the OneAPM server in 7X24 hours. The server needs to support the real-time writing, calculation, and query of massive data. At peak time, the OneAPM SaaS system needs to process 100 billion pieces of data every day.

In order to process these data, we built an analytical database druid cluster which is at the forefront in the world. This cluster deeply rewrites the Clickhouse database, creates a Chinese druid user group, compiles a number of technical books, and promotes practical experience of real-time analysis of massive data throughout China. Hence, the team has also produced a number of famous committers and montors of open source projects.

Since 2017, we have begun to create a completely decentralized and highly available database software using the open source community based on our own experience and business characteristics. In the process, we have gained great help from the Apache Foundation, Akka Community, Druid Community, and the Clickhouse Team, and three well-known software companies: OneAPM, Billion Cloud, and Jushu Database have also joined in as partners, all of which form today's Lambda project.

The Lambda project will be operated in a completely community-based manner, similar to the well-known open source projects such as Linux, Docker, and Kafka. The difference is that through the combination of our efforts and block chain, we will spontaneously connect the localized single point deployment of the original open source community into an automatically formed cloud. This is the wonder of the block chain.

6.1 Partners

He Xiaoyang	A well-known infrastructure software expert in China and a member of the JVM community, who worked at BEA Systems and Oracle as an engineer. In 2008, he built Blue Ocean Communications OneAPM and launched a SaaS-mode application performance management product in 2014. Within one year of 2014, OneAPM completed four rounds financing of Jingwei China, Chengwei Capital and Qiming Venture Partners, with a total investment of 300 million RMB. OneAPM was elected in the Forrester and Gartner reports and was named the leading ITOM manufacturer in the Asia Pacific region. OneAPM's SaaS products serve hundreds of thousands of developer users and thousands of companies in China and have extensive influence in finance, telecommunication, government, and large-scale Internet industries. OneAPM landed on the New Third Board innovation layer in 2016 with stock code 838699. He Xiaoyang is known as “China's first person in the APM industry”. He Xiaoyang was selected as an entrepreneur 35 under 35 in 2015. The Wechat's public account“He Xiaoyang's Reading Notes”has great influence and considerable range of readers in the programmer community and corporate services.
Li Monan	China's top programmer and founder of Coreseek and Log Insight As an expert of search engine, file system, storage system, and log system, the CoreSeek Chinese Word Segmentation System created by Li Monan was widely used in the Chinese Internet BBS community. Served as a technical consultant for many well-known internet companies to provide technical consulting and consulting services, familiar with the full set of technology stack from the operating system kernel to the browser to the database, has rich theoretical and practical experience on full text retrieval, OLAP database, compiler theory, virtual machine design, file system and data compression. At present, mainly studies distributed intelligent storage.

He Bingqing	OneAPM's co-founder & CTO, assembly language expert, jointly founded OneAPM in 2013 and responsible for the development of OneAPM core product Application Insight.
Guo Hongqiang	A data scientist with more than 10 years of experience in machine learning and artificial intelligence. He worked as a data science manager at GNS Healthcare in Boston, USA and supervised data science projects of supplier network optimization and severe disease intervention optimization. The products he developed have saved $40 million annually for customers. Before GNS, he spent 5 years at Cornell University in the United States. His research in data science was sponsored by the U.S. government with 6 million dollars.
Gao Haiqiang	A master of computer science, graduated successively from Computer department of Tianjin University and Inner Mongolia University. His main researches include P2P network design and optimization. He used to be vice president of technology for a network optimization company ACSNO, and founded OneASP, a security subsidiary of OneAPM in 2015 and served as CEO. OneASP got the investment leadership of three banks in 2015, with following investments of Jingwei, Chengwei and Qiming for 15 million RMB in total.
Haijun Zhao	Technical VP at OneAPM, previously worked in Qunar and Kongzhong, joined OneAPM in 2013 and set up the R&D team responsible for product development & managemen

6.2 R&D and Team Members

R&D Team	Huang Dong	Wang ZiMing	Zhang Xinyong	Zhang Chao
	Owen Yang	Liu Pai		
Marketing Staff	Lucy Wang	Patti Lin		

6.3 Technical Consultants

Tian Jia	M.S./B.S., Tsinghua University, Distributed SystemInvestor of Bitfinex & Limited Partner of BitfundConsultant of FBG CapitalConsultant of Zcash Jia Tian has served as senior dev at Baidu, Inc. and Alibaba, Inc. During the period he has developed large scale computer systems including the technology behind so.com, supporting >100 million page views per day, and large scale recommendation systems. Jia Tian is also a serial entrepreneur. He has joined the founding team of several companies focusing in AI and related areas. The first company Wolongyun has been acquired by Alibaba, Inc. Then he has joined the beijing machine learning information technology company as CTO, designed and built AI systems including recsys, chatbot, medical image recognition system. After that he has joined pony.ai to build an autonomous driving system, invested by Sequoia and IDG in angel round. Serves as the Chief Scientist in bitfundpe.com, a bitcoin fund dedicated in supporting the bitcoin community since 2013, founded by Xiaolai Li. Jia Tian is also advisor to multiple blockchain tech startups.
Zhongying Sun	ex technical expert at Alibaba and ex CTO at OKEX, was the technical advisor for OneAPM and OneASP
Liu Chunhua	The top hacker and security expert in China, founder and CEO of MAJORSEC and has been worked in Silicon Valley for many years. SEP Product founder of Symantec with annual sales of $2 billion, one of the Chinese technical experts with world's highest-ranking in Symantec. Graduated from the junior class of University of Science and Technology of China
Wang Tao	Co-founder and CTO of Jushan Database Former global consultant expert of IBM DB2, jointly designed IBM's new generation computer platform and former R & D Member of IBM North American Database Research Center
Cheng Yuan	Founder and CEO of Fang Cloud Worked at BOX R&D center in USA
BMAN	BMAN starts his career in global investment firm Blacksmith Global Ltd. Having a wealth of experience in both secondary market and venture investment, BMAN entered the area of blockchain as early as 2013 as an active investor in early-stage technology companies. BMAN also has a wide range of entrepreneurial experience. He cofounded Lijiaoshou (Shoujiao Technology, Inc.) In 2015, which was the top marketing content company in China and was acquired by Baidu, Inc. in 2017. Combined with artificial intelligence and big data technology, BMAN has developed Baidu AOD platform and served more than 500,000 business customers. BMAN served as consultant of dozen Internet listed companies, such as Alibaba, Inc, Lenovo, Inc and Qihoo, Inc, etc. and has accumulated strong relationships with some of the most promising entrepreneurs and top investors in both blockchain and internet industry.

6.4 Partners

ONE APM	OneAPM is a leading infrastructure software company in China.
亿方云	Fang Cloud is a leading cloud document and cloud collaboration SaaS company in China.

REFERENCES OF BLOCK CHAIN

[1] Amazon. Amazon Web service. http://aws.amazon.com/products/

[2] Houstn D, Ferdows A. Dropbox. https://www.dropbox.com/

[3] Microsoft. Windows azure. http://www.windowsazure.com/en-us/

[4] Schroeder B, Gibson GA. A large-scale study of failures in high-performance computing systems. IEEE Trans. on Dependable and Secure Computing, 2010. 337—350. [doi: 10.1109/TDSC.2009.4]

[5] http://games.qq.com/a/20110503/000013.htm

[6] http://www.chinanews.com/it/2011/09-05/3305704.shtml

[7] http://tech.163.com/07/ 0813/11/3LP86PEM000915BD.html

[8] Lamport L, Shostak R, Pease M. The Byzantine generals problem. ACM Trans. on Programming Languages and Systems, 1982, 4(3):382—401. [doi: 10.1145/357172.357176]

[9] Clement A, Kapritsos M, Lee S, Wang Y, Alvisi L, Dahlin M, Riche T. Upright cluster services. In: Proc. of the ACM SIGOPS 22nd Symp. on Operating Systems Principles. New York: ACM Press, 2009. 277—290. [doi: 10.1145/1629575.1629602]

[10] Castro M, Liskov B. Practical Byzantine fault tolerance and proactive recovery. ACM Trans. on Computer Systems, 2002,20(4): 398—461. [doi: 10.1145/571637.571640]

[11] Cowling J, Myers D, Liskov B, Rodrigues R, Shrira L. HQ replication: A hybrid quorum protocol for Byzantine fault tolerance. In: Proc. of the 7th Symp. on Operating Systems Design and Implementation. Berkeley: USENIX Association, 2006. 177—190.

[12] Kotla R, Alvisi L, Dahlin M, Clement A, Wong E. Zyzzyva: Speculative Byzantine fault tolerance. In: Proc. of the 21st ACM SIGOPS Symp. on Operating Systems Principles. New York: ACM Press, 2007, 45—58. [doi: 10.1145/1294261.1294267]

[13] Serafini M, Nokor P, Dobre D, Majuntke M, Suri M. Scrooge: Reducing the costs of fast Byzantine replication in presence of unresponsive replicas. In: Proc. of the 2010 IEEE/IFIP Int'l Conf. on Dependable Systems and Networks. 2010. 353—362. [doi: 10.1109/DSN.2010.5544295]

[14] Yin J, Martin JP, Venkataramani A, Alvisi L, Dahlin M. Separating agreement from execution for Byzantine fault tolerant services. In: Proc. of the 19th ACM Symp. on Operating Systems Principles. New York: ACM Press, 2003. 253—267. [doi: 10.1145/945445.945470]

[15] Wood T, Singh R, Venkataramani A, Shenoy P, Ceeehet E. ZZ and the art of practical BFT execution. In: Proc. of the 6th Conf. on Computer Systems. New York: ACM Press, 2011. 123—138. [doi: 10.1145/1966445.1966457]

[16] Wester B, Cowling J, Nightingale EB, Chen PM, Flinn J, Liskov B. Tolerating latency in replicated state machines through client speculation. In: Proc. of the 6th USENIX Symp. on Networked Systems Design and Implementation. Berkeley: USENIX Association, 2009. 245—260.

[17] Hendricks J, Sinnamohideen S, Ganger GR, Reiter MK. Zzyzx: Scalable fault tolerance through Byzantine locking. In: Proc. of the 2010 IEEE/IFIP Int'l Conf. on Dependable Systems and Networks. 2010. 363—372. [doi: 10.1109/DSN.2010.5544297]

[18] Pease M, Shostak R, Lamport L. Reaching agreement in the presence of faults. Journal of the ACM, 1980,27(2):228—234. [doi: 10.1145/322186.322188]

[19] Matin JP, Alvisi L. Fast Byzantine consensus. IEEE Trans. on Dependable and Secure Computing, 2006,3(3):202—215. [doi: 10.1109/TDSC.2006.35]

[20] Baker MG, Hartman JH, Kupfer MD, Shirriff KW, Ousterhout JK. Measurements of a distributed file system. In: Proc. of the 13th ACM Symp. on Operating Systems Principles. New York: ACM Press, 1991,25(5):198—212. [doi: 10.1145/121132.121164]

[21] Leung AW, Pasupathy S, Goodson G, Miller EL. Measurement and analysis of large-scale network file system workloads. In: Proc. of the USENIX 2008 Annual Technical Conf. on Annual Technical Conf. Berkeley: USENIX Association, 2008. 213—226.

[22] Amir Y, Coan B, Kirsch J, Lane J. Byzantine replication under attack. In: Proc. of the DSN. 2008.

[23] Malkhi D, Reiter M. Byzantine quorum systems. Jorunal of Distributed Computing, 1988,11(4):203—213.

[24] Martin JP, Alvisi L, Dahlin M. Minimal Byzantine storage. In: Proc. of the 16th Int'l Conf. on Distributed Computing. London: Springer-Verlag, 2002. 311—325. [doi: 10.1007/3-540-36108-1_21]

[25] Abd-EL-Malek M, Ganger GR, Goodson GR, Reiter MK, Wylie JJ. Lazy verification in fault-tolerant distributed storage systems. In: Proc. of the 24th IEEE Symp. on Reliable Distributed Systems. 2005. 179—190. [doi: 10.1109/RELDIS.2005.20]

[26] Goodson GR, Wylie JJ, Ganger GR, Reiter MK. Efficient Byzantine-tolerant erasure-coded storage. In: Proc. of the 2004 Int'l Conf. on Dependable Systems and Networks. 2004. 135—144.

[27] Hendricks J, Ganger GR, Reiter MK. Low-Overhead Byzantine fault-tolerant storage. In: Proc. of the 21st ACM SIGOPS Symp. on Operating Systems Principles. New York: ACM Press, 2007. 73—86. [doi: 10.1145/1294261.1294269]

[28] Hendricks J, Granger GR, Reiter MK. Verifying distributed erasure-coded data. In: Proc. of the 26th annual ACM Symp. on Principles of Distributed Computing. New York: ACM Press, 2007. 139—146. [doi: 10.1145/1281100.1281122]

[29] Abd-El-Malek M, Ganger G, Goodson G, Reiter M. Fault-Scalable Byzantine fault-tolerant services. In: Proc. of the 20th ACM Symp. on Operating Systems Principles. New York: ACM Press, 2005. 59—74. [doi: 10.1145/1095810.1095817]

[30] Luo XH, Shu JW. Summary of research for erasure code in storage system. Journal of Computer Research and Development, 2012, 49(1):1—11 (in Chinese with English abstract).

[31] Li MQ, Shu JW, Zheng WM. GRID codes: Strip-based erasure codes with high fault tolerance for storage systems. ACM Trans. on Storage, 2009,4(4):1—22. [doi: 10.1145/1480439.1480444]

[32] Patterson D, Chen P, Gibson G, Katz R. Introduction to redundant arrays of inexpensive disks (RAID). In: Proc. of the Spring COMPCON'89. 1989. 112—117. [doi: 10.1109/CMPCON.1989.301912]

[33] Cachin C, Tessaro S. Asynchronous verifiable information dispersal. In: Proc. of the 24th IEEE Symp. on Reliable Distributed Systems. 2005. 191—201. [doi: 10.1109/RELDIS.2005.9]

[34] Alvisi L, Malkhi D, Pierce E, Reiter MK. Fault detection for Byzantine quorum systems. IEEE Trans. on Parallel and Distributed Systems, 2001,12(9):996—1007. [doi: 10.1109/71.954640]

[35] Chandra TD, Toueg S. Unreliable failure detectors for reliable distributed systems. Journal of the ACM, 1996,43(2):225—267. [doi: 10.1145/226643.226647]

[36] Lima M, Greve F, Arantes L, Sens P. Byzantine failure detection for dynamic distributed systems. SANTOSDELIMA-2010-584597, RR-7222. Paris, 2010.

[37] Delporte-Gallet C, Fauconnier H, Guerraoui R, Hadzilacos V, Houznetsov P, Toueq S. The weakest failure detectors to solve certain fundamental problems in distributed computing. In: Proc. of the 23rd Annual ACM Symp. on Principles of Distributed Computing. New York: ACM Press, 2004. 338—346. [doi: 10.1145/1011767.1011818]

[38] Liu G, Zhou J, Sun Y, Qin L. A fault detection mechanism in erasure-code Byzantine fault-tolerance quorum. Wuhan University Journal of Natural Sciences, 2006,11(6):1453—1456. [doi: 10.1007/BF02831796]

[39] Reiser HP, Kapitza R. Hypervisor-Based efficient proactive recovery. In: Proc. of the 26th IEEE Int'l Symp. on Reliable Distributed Systems. 2007. 83—92. [doi: 10.1109/SRDS.2007.25]

[40] Sousa P, Bessani AN, Correia M, Neves NF, Verissimo P. Highly available intrusion-tolerant services with proactive-reactive recovery. IEEE Trans. on Parallel and Distributed Systems, 2010,21(4):452—465. [doi: 10.1109/TPDS.2009.83]

[41] Distler T, Kapitza R, Reiser HP. Efficient state transfer for hypervisor-based proactive recovery. In: Proc. of the 2nd Workshop on Recent Advances on Intrusiton-Tolerant Systems. New York: ACM Press, 2008. [doi: 10.1145/1413901.1413905]

[42] Paulo E. Travelling through wormholes: A new look at distributed systems models. Journal of SIGACT News, 2006,37(1):66—81. [doi: 10.1145/1122480.1122497]

[43] Chun BG, Maniatis P, Shenker S, Kubiatowicz J. Tiered fault tolerance for long-term integrity. In: Proc. of the 7th Conf. on File and Storage Technologies. Berkeley: USENIX Association, 2009. 267—282.

[44] Chun BG, Maniatis P, Shenker S, Kubiatowicz J. Attested append-only memory: Making adversaries stick to their word. In: Proc. of the 21st ACM SIGOPS Symp. on Operating Systems Principles. New York: ACM Press, 2007. 189—204. [doi: 10.1145/1294261.1294280]

[45] Felten EW. Understanding trusted computing: Will its benefits outweigh its drawbacks? Journal of IEEE Security Privacy, 2003, 1(3):60—62. [doi: 10.1109/MSECP.2003.1203224]

[46] Levin D, Douceur JR, Lorch JR, Moscibroda T. TrInc: Small trusted hardware for large distributed systems. In: Proc. of the 6th USENIX Symp. on Networked Systems Design and Implementation. Berkeley: USENIX Association, 2009. 1—14.

[47] Trusted Computing Group. Trusted platform module. http://www.trustedcomputinggroup.org/developers/trusted_platform_module

[48] Singh A, Fonseca P, Kuznetsov P, Rodrigues R, Maniatis P. Zeno: Eventually consistent Byzantine-fault tolerance. In: Proc. of the 6th USENIX Symp. on Networked Systems Design and Implementation. Berkeley: USENIX Association, 2009. 169—184.

[49] Aiyer AS, Alvisi L, Clement A, Dahlin M, Martin JP, Porth C. BAR fault tolerance for cooperative services. In: Proc. of the 20th ACM Symp. on Operating Systems Principles. New York: ACM Press, 2005. 45—58. [doi: 10.1145/1095810.1095816]

Lambda

REFERENCES OF CRYPTOGRAPHY

[1] Fiat A, Naor M. Broadcast encryption. In: Stinson DR, ed. Advances in Cryptology–CRYPTO'93. Berlin, Heidelberg: Springer–Verlag, 1994. 480—491.

[2] Naor D, Naor M, Lotspiech J. Revocation and tracing schemes for stateless receivers. In: Kilian J, ed. Advances in Cryptology–CRYPTO 2001. Berlin, Heidelberg: Springer–Verlag, 2001. 41—62.

[3] Boneh D, Gentry C, Waters B. Collusion resistant broadcast encryption with short ciphertexts and private keys. In: Shoup V, ed.Advances in Crytology–CRYPTO 2005. Berlin, Heidelberg: Springer–Verlag, 2005. 258—275. [doi: 10.1007/11535218_16]

[4] Shamir A. Identity–Based cryptosystems and signature schemes. In: Blakley GR, Chaum D, eds. Advances in Cryptology–CRYPTO'84. Berlin, Heidelberg: Spring–er–Verlag, 1984. 47—53.

[5] Boneh D, Franklin M. Identity–Based encryption from the weil pairing. In: Kilian J, ed. Advances in Cryptology–CRYPTO 2001.LNCS 2139, Berlin, Heidelberg: Springer–Verlag, 2001. 213—229. [doi: 10.1007/3–540–44647–8_13]

[6] Sahai A, Waters B. Fuzzy identity–based encryption. In: Cramer R, ed. Advances in Cryptology–EUROCRYPT 2005. Berlin,Heidelberg: Springer–Verlag, 2005. 457—473.

[7] Goyal V, Pandey O, Sahai A, Waters B. Attribute–Based encryption for fine–grained access control of encrypted data. In: Proc. ofthe 13th ACM Conf. on Com–puter and Communications Security. New York: ACM Press, 2006. 89—98. [doi: 10.1145/1180405.1180418]

[8] Yu SC, Ren K, Lou WJ. Attribute–Based content distribution with hidden policy. In: Proc. of the 4th Workshop on Secure NetworkProtocols (NPSec). Orlando: IEEE Computer Society, 2008. 39—44. [doi: 10.1109/NPSEC.2008.4664879]

[9] Traynor P, Butler K, Enck W, Mcdaniel P. Realizing massive–scale conditional access systems through attribute–basedcryptosystems. In: Proc. of the 15th Annual Network and Distributed System Security Symp. (NDSS 2008). San Diego: USENIXAssociation, 2008. 1—13.

[10] Cheung L, Newport C. Provably secure ciphertext policy ABE. In: Proc. of the ACM Conf. on Computer and CommunicationsSecurity. New York: ACM Press, 2007. 456—465. [doi:10.1145/1315245.1315302]

[11] Cheung L, Cooley JA, Khazan R, Newport C. Collusion–Resistant group key management using attribute–based encryption.http://eprint.ia–cr.org/2007/161.pdf

[12] Yu SC, Ren K, Lou WJ. Attribute–Based on–demand multicast group setup with membership anonymity. Computer Networks, 2010,54(3):377—386. [doi: 10.1016/j.comnet.2009.09.009]

[13] Baden R, Bender A, Spring N, Bhattacharjee B, Starin D. Persona: An online social network with user–defined privacy. In: Proc. ofthe ACM SIGCOMM 2009 Conf. on Data Communication. New York: ACM Press, 2009. 135—146. [doi: 10.1145/1592568.1592585]

[14] Bethencourt J, Sahai A, Waters B. Ciphertext–Policy attribute–based encryption. In: Proc. of the 2007 IEEE Symp. on Security andPrivacy. Washington: IEEE Computer Society, 2007. 321—334. [doi: 10.1109/SP.2007.11]

[15] Beimel A. Secure schemes for secret sharing and key distribution [Ph.D. Thesis]. Technion: Israel Institute of Technology, 1996.

[16] Liang XH. Research on attribute–based cryptosystem [MS. Thesis]. Shanghai: Shanghai Jiaotong University Press, 2009 (inChinese).

[17] Lewko A, Sahai A, Waters B. Revocation systems with very small private keys. In: Proc. of the IEEE Symp. on Security andPrivacy. Washington: IEEE Computer Society, 2010. 273—285. [doi: 10.1109/SP.2010.23]

[18] Shamir A. How to share a secret. Communications of the ACM, 1979,22(11):612—613. [doi: 10.1145/359168.359176]

[19] Pirretti M, Traynor P, Mcdaniel P, Waters B. Secure attribute–based systems. In: Proc. of the ACM Conf. on Computer andCommunications Security. New York: ACM Press, 2006. 99—112. [doi: 10.1145/1180405.1180419]

[20] Baek J, Susilo W, Zhou J. New constructions of fuzzy identity–based encryption. In: Proc. of the ASIAN ACM Conf. on Computerand Communications Security (ASIACCS 2007). New York: ACM Press, 2007. 368—370. [doi: 10.1145/1229285.1229330]

[21] Ostrovsky R, Sahai A, Waters B. Attribute–Based encryption with non–monotonic access structures. In: Proc. of the ACM Conf. onComputer and Communica–tions Security. New York: ACM Press, 2007. 195—203. [doi: 10.1145/1315245.1315270]

[22] Naor M, Pinkas B. Efficient trace and revoke schemes. In: Frankel Y, ed. Proc. of the Financial Cryptography. Berlin, Heidelberg:Springer–Verlag, 2001. 1—20. [doi: 10.1007/978–3–540–68914–0_7]

[23] Nishide T, Yoneyama K, Ohta K. Attribute–Based encryption with partially hidden encryptor–specified access structures. In:Bellovin SM, Gennaro R, Keromytis A, Yung M, eds. Proc. of the Applied Cryptography and Network Security. Berlin, Heidelberg:Springer–Verlag, 2008. 111—129. [doi: 10.1007/978–3–540–68914–0_7]

[24] Emura K, Miyaji A, Nomura A, Omote K, Soshi M. A ciphertext–policy attribute–based encryption scheme with constant ciphertextlength. In: Bao F, Li H, Wang G, eds. Proc. of the InformationSecurity Practice and Experience (ISPEC 2009). Berlin, Heidelberg:Springer–Verlag, 2009. 13—23. [doi: 10.1007/978–3–642–00843–6_2]

[25] Canetti R, Halevi S, Katz J. Chosen–Ciphertext security from identity–based encryption. In: Advances in Cryptology–EUROCRYPT2004. Berlin, Heidelberg: Springer–Verlag, 2004. 207—222.

[26] Boneh D, Waters B. Conjunctive, subset, and range queries on encrypted data. In: Proc. of the Theory of Cryptography Conf.(TCC). Berlin, Heidelberg: Spring–er–Verlag, 2007. 535—554. [doi: 10.1007/978–3–540–70936–7_29]

[27] Goyal V, Jain A, Pandey O, Sahai A. Bounded ciphertext policy attribute based encryption. In: Aceto L, Damgård I, Goldberg LA,Halldórsson M M, Ingólfsdóttir A, Walukiewicz I, eds. Proc. of the ICALP 2008. Berlin, Heidelberg: Springer–Verlag, 2008.579—591. [doi: 10.1007/978–3–540–70583–3_47]

[28] Liang XH, Cao ZF, Lin H, Xing DS. Provably secure and efficient bounded ciphertext policy attribute based encryption. In: Proc.of the ASIAN ACM Symp. on

Information, Computer and Communications Security (ASIACCS 2009). New York: ACM Press,2009. 343—352. [doi: 10.1145/1533057.1533102]

[29] Ibraimi L, Tang Q, Hartel P, Jonker W. Efficient and provable secure ciphertext-policy attribute-based encryption schemes. In: Proc.of the Information Security Practice and Experience. Berlin, Heidelberg: Springer-Verlag, 2009. 1—12. [doi: 10.1007/978-3-642-00843-6_1]

[30] Waters B. Ciphertext-Policy attribute-based encryption: An expressive, efficient, and provably secure realization. http://eprint.iacr.org/2008/290.pdf [doi: 10.1007/978-3-642-19379-8_4]

[31] Attrapadung N, Imai H. Conjunctive broadcast and attribute-based encryption. In: Shacham H, Waters B, eds. Proc. of thePairing-Based Cryptography-Pairing 2009. Berlin, Heidelberg: Springer-Verlag, 2009. 248—265. [doi: 10.1007/978-3-642-03298-1_16]

[32] Lewko A, Okamoto T, Sahai A, Takashima K, Waters B. Fully secure functional encryption: Attribute-Based encryption and(hierarchical) inner product encryption. In: Advances in Cryptology-EUROCRYPT 2010. LNCS 6110, Berlin, Heidelberg:Springer-Verlag, 2010. 62—91. [doi: 10.1007/978-3-642-13190-5_4]

[33] Waters B. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In: Halevi S, ed. Advances inCryptology-CRYTO 2009. Berlin, Heidelberg: Springer-Verlag, 2009. 619—636. [doi: 10.1007/978-3-642-03356-8_36]

[34] Lewko A, Waters B. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In: Proc. of the 7thTheory of Cryptography Conf. (TCC 2010). Berlin, Heidelberg: Springer-Verlag, 2010. 455—479. [doi: 10.1007/978-3-642-11799-2_27]

[35] Attrapadung N, Imai H. Attribute-Based encryption supporting direct/indirect revocation modes. In: Parker MG, ed. Proc. of theCryptography and Coding 2009. Berlin, Heidelberg: Springer-Verlag, 2009. 278—300. [doi: 10.1007/978-3-642-10868-6_17]

[36] Boldyreva A, Goyal V, Kumar V. Identity-Based encryption with efficient revocation. In: Proc. of the ACM Conf. on Computerand Communications Security. New York: ACM Press, 2008. 417—426. [doi: 10.1145/1455770.1455823]

[37] Ibraimi L, Petkovic M, Nikova S, Hartel P, Jonker W. Mediated ciphertext-policy attribute-based encryption and its application. In:Proc. of the 10th Int'l Workshop on Information Security Applications-WISA 2009. LNCS 5932, Berlin, Heidelberg:Springer-Verlag, 2009. 309—323. [doi: 10.1007/978-3-642-10838-9_23]

[38] Yu SC, Wang C, Ren K, Lou WJ. Attribute based data sharing with attribute revocation. In: Proc. of the ASIAN ACM Conf. onComputer and Communications Security (ASIACCS 2010). New York: ACM Press, 2010. 261—270. [doi: 10.1145/1755688.1755720]

[39] Li J, Ren K, Kim K. A2BE: Accountable attribute-based encryption for abuse free access control. http://eprint.iacr.org/2009/118.pdf

[40] Li J, Ren K, Zhu B, Wan ZG. Privacy-Aware attribute-based encryption with user accountability. In: Proc. of the InformationSecurity Conf. 2009. LNCS 5735, Berlin, Heidelberg: Springer-Verlag, 2009. 347—362. [doi: 10.1007/978-3-642-04474-8_28]

[41] Yu SC, Ren K, Lou WJ, Li J. Defending against key abuse attacks in KP-ABE enabled broadcast systems. In: Proc. of the Securityand Privacy in Communication Networks. Berlin, Heidelberg: Springer-Verlag, 2009. 311—329. [doi: 10.1007/978-3-642-05284-2_18]

[42] Boneh D, Sahai A, Waters B. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In: Vaudenay S,ed.Advances in Cryptology-EUROCRYPT 2006. LNCS 4004, Berlin, Heidelberg: Springer-Verlag, 2006. 573—592. [doi:10.1007/11761679_34]

[43] Chase M. Multi-Authority attribute based encryption. In: Proc. of the Theory of Cryptography Conf. (TCC). Berlin, Heidelberg:Springer-Verlag, 2007. 515—534.

[44] Božović V, Socek D, Steinwandt R, Villányi VI. Multi-Authority attribute based encryption with honest-but-curious centralauthority 2009. http://eprint.iacr.org/2009/083.pdf

[45] Lin H, Cao ZF, Liang X, Shao J. Secure threshold multi authority attribute based encryption without a central authority. In:Chowdhury DR, Rijmen V, Das A, eds. Proc. of the Cryptology in India-INDOCRYPT 2008. Berlin, Heidelberg: Springer-Verlag,2008. 426—436. [doi: 10.1007/978-3-540-89754-5_33]

[46] Chase M, Chow SSM. Improving privacy and security in multi-authority attribute-based encryption. In: Proc. of the ACM Conf. onComputer and Communications Security. New York: ACM Press, 2009. 121—130. [doi: 10.1145/1653662.1653678]

[47] Gennaro R, Jarecki S, Krawczyk H, Rabin T. Secure distributed key generation for discrete-log based cryptosystems. Journal ofCryptology, 2007,20(1):51—83. [doi: 10.1007/s00145-006-0347-3]

[48] Goyal V, Lu S, Sahai A, Waters B. Black-Box accountable authority identity-based encryption. In: Proc. of the ACM Conf. onComputer and Communications Security. New York: ACM Press, 2008. 427—436. [doi: 10.1145/1455770.1455824]

[49] Kapadia A, Tsang PP, Smith SW. Attribute-Based publishing with hidden credentials and hidden policies. In: Proc. of the 14thAnnual Network and Distributed System Security Symp. (NDSS 2007). USENIX Association, 2007. 179—192.

[50] Li J, Wang Q, Wang C, Ren K. Enhancing attribute-based encryption with attribute hierarchy. In: Proc. of the Mobile Networks andApplications. Berlin, Heidelberg: Springer-Verlag, 2010. 1—9. [doi: 10.1007/s11036-010-0233-y]

[51] Agrawal S, Boneh D, Boyen X. Efficient lattice (H) IBE in the standard model. In: Gilbert H, ed. Advances in Cryptology-EUROCRYPT 2010. Berlin, Heidelberg: Springer-Verlag, 2010. 553—572.

[52] Boneh D, Boyen X, Goh EJ. Hierarchical identity based encryption with constant size ciphertext. In: Cramer R, ed. Advances inCryptology-EUROCRYPT 2005. Berlin, Heidelberg: Springer-Verlag, 2005. 440—456. [doi: 10.1007/11426639_26]

[53] Boyen X, Waters B. Anonymous hierarchical identity-based encryption (without random oracles). In: Dwork C, ed. Advances inCryptology-CRYPTO 2006. Berlin, Heidelberg: Springer-Verlag, 2006. 290—307. [doi: 10.1007/11818175_17]

[54] Horwitz J, Lynn B. Toward hierarchical identity-based encryption. In: Proc. of the Theory and Applications of CryptographicTechniques. Berlin, Heidelberg: Springer-Verlag, 2002. 466—481.

[55] Yao DF, Fazio N, Dodis Y, Lysyanskaya A. Id-Based encryption for complex hierarchies with applications to forward security andbroadcast encryption. In: Proc. of the ACM Conf. on Computer and Communications Security. New York: ACM Press, 2004.354—363. [doi: 10.1145/1030083.1030130]

[56] Attrapadung N, Imai H. Dual-Policy attribute based encryption. In: Abdalla M, Pointcheval D, Fouque P A, Vergnaud D, eds. Proc.of the Applied Cryptography and Network Security. Berlin, Heidelberg: Springer-Verlag, 2009. 168—185. [doi: 10.1007/978-3-642-01957-9_11]

REFERENCES OF DATABASE AND STORAGE

[1] Gray J. What next? A few remaining problems in information technology. 1998. http://research.microsoft.com/~gray/talks/ Gray_Turing_FCRC.pdf

[2] Welch B, Unangst M, Abbasi Z, Gibson G. Scalable performance of the Panasas parallel file system. In: Baker M, ed. Proc. of the 2008 Conf. on File and Storage Technologies (FAST 2008). San Jose: USENIX, 2008. 17—33.

[3] Karger D, Lehman E, Leighton T, Levine M, Lewin D. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the World Wide Web. In: Proc. of the 29th Annual ACM Symp. on Theory of Computing (STOC'97). El Paso: ACM Press, 1997. 654—663.

[4] Tang H, Gulbeden A, Zhou JY, Strathearn W, Yang T, Chu LK. A self-organizing storage cluster for parallel data-intensive applications. In: Huskamp JC, ed. Proc. of the 2004 ACM/IEEE Conf. on Supercomputing (SC 2004). Pittsburgh: IEEE Computer Society, 2004. 52—63.

[5] Stoica I, Morris R, Karger D, Kaashoek F, Balakrishnan H. Chord: A scalable peer-to-peer lookup service for internet applications. In: Cruz R, ed. Proc. of the Annual Conf. of the Special Interest Group on Data Communication (SIGCOMM 2001). San Diego: ACM Press, 2001. 149—160.

[6] Brinkmann A, Salzwedel K, Scheideler C. Efficient, distributed data placement strategies for storage area networks. In: Gary M, ed. Proc. of the 12th ACM Symp. on Parallel Algorithms and Architectures. Bar Harbor: ACM Press, 2000. 119—128.

[7] Brinkmann A, Salzwedel K, Scheideler C. Compact, adaptive placement schemes for non-uniform distribution requirements. In: Maggs B, ed. Proc. of the 14th ACM Symp. on Parallel Algorithms and Architectures. Winnipeg: ACM Press, 2002. 53—62.

[8] Schindelhauer C, Schomaker G. Weighted distributed hash tables. In: Paul S, ed. Proc. of the 17th ACM Symp. on Parallelism in Algorithms and Architectures (SPAA 2005). Las Vegas: ACM Press, 2005. 218—227.

[9] Honicky RJ, Miller EL. A fast algorithm for online placement and reorganization of replicated data. In: Dongarra J, ed. Proc. of the 17th Int'l Parallel & Distributed Processing Symp. (IPDPS 2003). Nice: IEEE Computer Society, 2003.

[10] Honicky RJ, Miller EL. Replication under scalable hashing: A family of algorithms for scalable decentralized data distribution. In: Bader DA, ed. Proc. of the 18th Int'l Parallel & Distributed Processing Symp. (IPDPS 2004). Santa Fe: IEEE Computer Society, 2004.

[11] Weil SA, Brandt SA, Miller EL, Maltzahn C. CRUSH: Controlled, scalable and decentralized placement of replicated data. In: Miller BH, ed. Proc. of the 2006 ACM/IEEE Conf. on Supercomputing. Tampa: IEEE Computer Society, 2006.

[12] Weil SA, Brandt SA, Miller EL, Long DDE, Maltzahn C. Ceph: A scalable, high-performance distributed file system. In: Bershad B, ed. Proc. of the 7th Symp. on Operating Systems Design and Implementation. Seattle: USENIX, 2006. 307—320.

[13] Gobioff H, Gibson G, Tygar D. Security for network attached storage devices. Technical Report, TRCMU-CS-97-185, Pittsburgh: Carniege Mellon University, 1997.

[14] Sun Microsystems, Inc. Lustre file system: High-Performance storage architecture and scalable cluster file system. 2007. https://www.sun.com/offers/docs/LustreFileSystem.pdf

[15] Nagle D, Serenyi D, Matthews A. The Panasas ActiveScale storage cluster—Delivering scalable high bandwidth storage. In: Huskamp JC, ed. Proc. of the 2004 ACM/IEEE Conf. on Supercomputing. Pittsburgh: IEEE Computer Society, 2004.

[16] Schmuck F, Haskin R. GPFS: A shareddisk file system for large computing clusters. In: Long D, ed. Proc. of the 2002 Conf. on File and Storage Technologies (FAST 2002). Monterey: USENIX, 2002. 231—244.

[17] Liu Z, Zhou XM. A data object placement algorithm based on dynamic interval mapping. Journal of Software, 2005,16(11): 1886—1893 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/1886.htm [doi: 10.1360/jos161886]

[18] Wang D, Shu JW, Xue W, Shen MM. Self-Adaptive hierarchical storage management in SAN based on block-level. Advanced Technology Communication, 2007,17(2):111—115 (in Chinese with English abstract).

[19] Wang F, Zhang SD, Feng D, Zeng LF. Hybrid object allocation policy for object storage systems. Journal of Huazhong University of Science & Technology (Nature Science Edition), 2007,35(3):46—48 (in Chinese with English abstract).

[20] Tan HF, Sun LL, Hou ZF. An effective algorithm of data placement in a mass storage system. Computer Engineering, 2006,32(10): 47—49 (in Chinese with English abstract).

[21] Brinkmann A, Effert S, Heide FMAD, Scheideler C. Dynamic and redundant data placement. In: Abdelrahman TS, ed. Proc. of the 27th Int'l Conf. on Distributed Computing Systems. Toronto: IEEE Computer Society, 2007.

ANNEX I: EVENT SOURING

We can query an application's state to find out the current state of the world, and this answers many questions. However there are times when we don't just want to see where we are, we also want to know how we got there.

Event Sourcing ensures that all changes to application state are stored as a sequence of events. Not just can we query these events, we can also use the event log to reconstruct past states, and as a foundation to automatically adjust the state to cope with retroactive changes.

The fundamental idea of Event Sourcing is that of ensuring every change to the state of an application is captured in an event object, and that these event objects are themselves stored in the sequence they were applied for the same lifetime as the application state itself.

Let's consider a simple example to do with shipping notifications. In this example we have many ships on the high seas, and we need to know where they are. A simple way to do this is to have a tracking application with methods to allow us to tell when a ship arrives or leaves at a port.

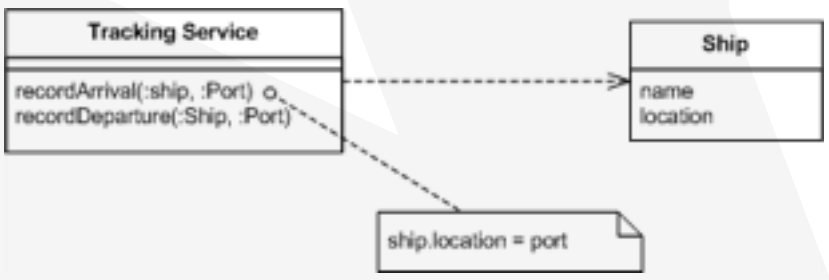


Figure 1: A simple interface for tracking shipping movements.

In this case when the service is called, it finds the relevant ship and updates its location. The ship objects record the current known state of the ships. Introducing Event Sourcing adds a step to this process. Now the service creates an event object to record the change and processes it to update the ship.

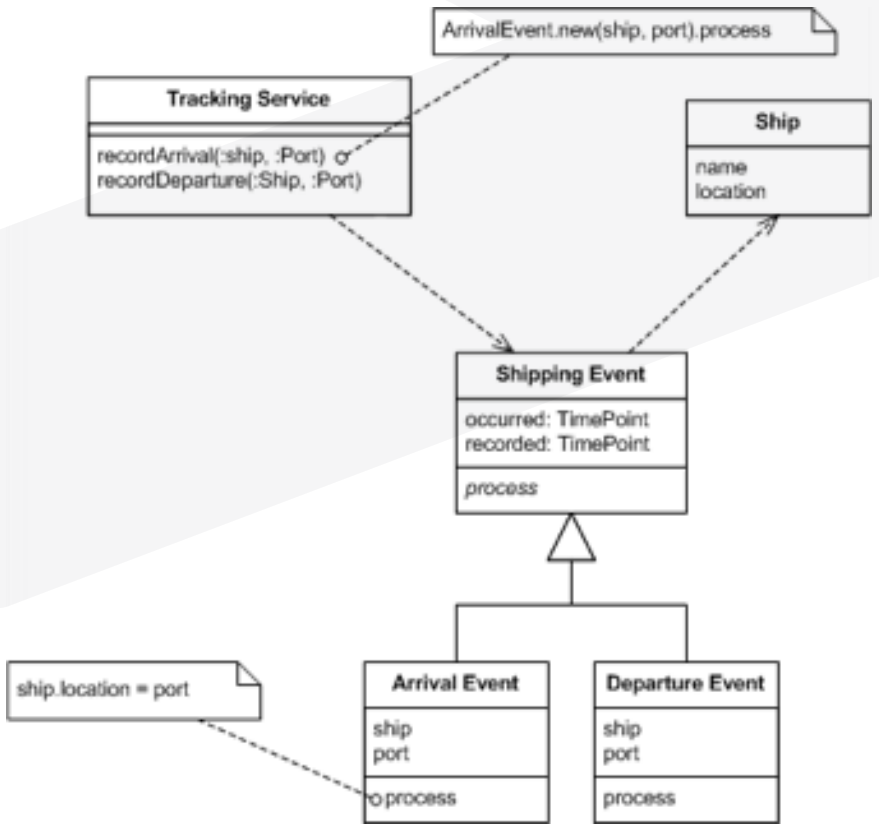


Figure 2: Using an event to capture the change.

Looking at just the processing, this is just an unnecessary level of indirection. The interesting difference is when we look at what persists in the application after a few changes. Let's imagine some simple changes:

- The Ship 'King Roy' departs San Francisco
- The Ship 'Prince Trevor' arrives at Los Angeles
- The Ship 'King Roy' arrives in Hong Kong

With the basic service, we see just the final state captured by the ship objects. I'll refer to this as the application state.

Figure 3: State after a few movements tracked by simple tracker.

With Event Sourcing we also capture each event. If we are using a persistent store the events will be persisted just the same as the ship objects are. I find it useful to say that we are persisting two different things an application state and an event log.

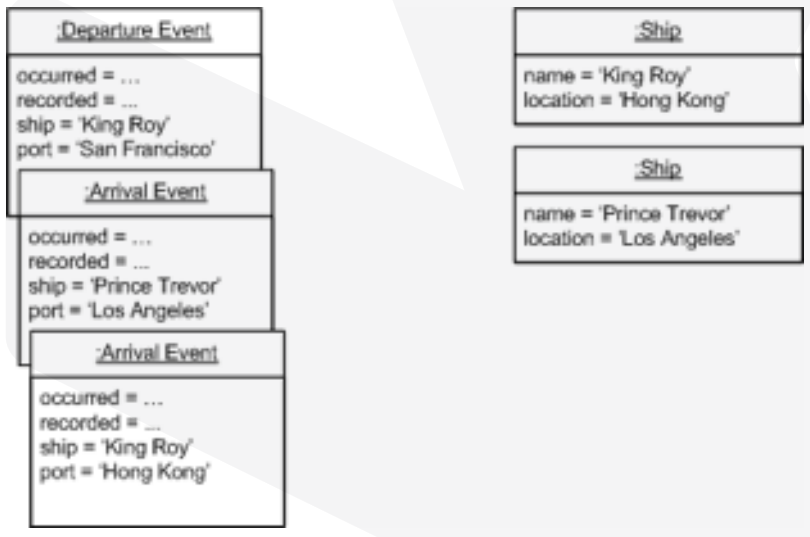


Figure 4: State after a few movements tracked by event sourced tracker.

The most obvious thing we've gained by using Event Sourcing is that we now have a log of all the changes. Not just can we see where each ship is, we can see where it's been. However this is a small gain. We could also do this by keeping a history of past ports in the ship object, or by writing to a log file whenever a ship moves. Both of these can give us an adequate history.

- The key to Event Sourcing is that we guarantee that all changes to the domain objects are initiated by the event objects. This leads to a number of facilities that can be built on top of the event log:
  Complete Rebuild: We can discard the application state completely and rebuild it by re–running the events from the event log on an empty application.

- Temporal Query: We can determine the application state at any point in time. Notionally we do this by starting with a blank state and rerunning the events up to a particular time or event. We can take this further by considering multiple time–lines (analogous to branching in a version control system).

- Event Replay: If we find a past event was incorrect, we can compute the consequences by reversing it and later events and then replaying the new event and later events. (Or indeed by throwing away the application state and replaying all events with the correct event in sequence.) The same technique can handle events received in the wrong sequence – a common problem with systems that communicate with asynchronous messaging.

A common example of an application that uses Event Sourcing is a version control system. Such a system uses temporal queries quite often. Subversion uses com–plete rebuilds whenever you use dump and restore to move stuff between repository files.

A Blockchain Infrastructure Providing
Unlimited Storage Capabilities

Lambda

lambda.im