# Paradigm Network:
# A Scalable FAAS Network based on DAG

Paradigm Lab[a,1]

[a]*Paradigm Blockchain Research Center, Melbourne, Australia*
[b]*Paradigm Blockchain Development Center, Beijing, China*

---

**Abstract**

The Internet has undergone great changes in the past 50 years. One of the major changes was the introduction of global networking. Since the birth of ARPANET in 1969, the mainstream countries have gradually accessed the Internet. The second major change is global applications online. Since the advent of World Wide Web in 1989, Internet applications have been blooming. Paradigm believes that blockchain is on the way to third great change, and aims to establish a high performance, flexibly scalable and development-friendly distributed FAAS network. Paradigm reorganizes data structure into directed acyclic graph and proposes a two-layer hybrid Binary-Star Consensus (BSC) mechanism for concurrent transaction processing, and puts forward a Quantum Channel Protocol (QCP) for short time network delay. In order to be widely scalable, Paradigm proposes a pluggable VM designed by Instruction-set Hub Protocol (ICP) and elastic scheduling distribution to make the DAPPs easier to be spread. For secure deployment, Paradigm proposes a WebAssembly-based PVM which supports multi-languages and an event-driven Function Execution Engine (FEE) which splits the application development into units. To maintain privacy and security for all stakeholders, Paradigm provides an optional ZKP-based homomorphic encryption for the privacy of important data; an automated AI-based security rating engine for the security of smart contract and the anti-quantum crypto components to prevent the potential attacks in future. This paper focuses on the technical description of Paradigm, including system architecture, key modular design and un-negligible details.

*Keywords:* BSC, DAG, FAAS, PVM, QCP, ICT

---

*Email address:* `lab@paradigm.fund`; *Website*: `https://www.paradigm.fund`;
*Github*:`https://github.com/paradigm-network` (Paradigm Lab)

Contents

**1 Research status** **4**

1.1 DAG . . . . . 4

1.2 Sever-less . . . . . 5

**2 Paradigm** **6**

2.1 Terminology . . . . . 6

2.2 About Paradigm . . . . . 7

2.3 Mission and Vision . . . . . 7

**3 FAAS+DAG** **7**

**4 Technique Overview** **9**

4.1 Architecture . . . . . 9

4.2 Features . . . . . 11

**5 Core modules** **12**

5.1 Binary-Star Consensus . . . . . 12

5.1.1 Accretion . . . . . 13

5.1.2 Sequentia . . . . . 14

5.2 PVM . . . . . 17

5.2.1 WebAssembly-based Execution . . . . . 17

5.2.2 Instruction-set Hub Protocol . . . . . 18

5.2.3 Function Naming Protocol . . . . . 19

5.3 Quantum Channel Protocol . . . . . 21

5.4 Decentralized FAAS Network . . . . . 22

5.4.1 Distributed Gateway . . . . . 23

5.4.2 Accretion Network . . . . . 24

5.4.3 Function Compute Engine . . . . . 25

5.4.4 Sequentia DAG network . . . . . 26

**6 Plan in Long Term** **26**

6.1 PVM Safety Rating Engine based on deep learning . . . . . 26

6.2 Functional Privacy Protection . . . . . 28

6.2.1 General construction . . . . . 28

6.2.2 Modular construction . . . . . 29

6.2.3 Work Flow . . . . . 30

6.3 Anti-quantum Crypto-components . . . . . . . . . . . . . . . . . . . . . 31
6.3.1 Sponge-based Hash Algorithm . . . . . . . . . . . . . . . . . . . 31
6.3.2 Lattice-based Signature Algorithm . . . . . . . . . . . . . . . . . 31

**7 Comparison** **32**

## 1. Research status

### *1.1. DAG*

Instead of linear chain-based design, the upcoming next generation of blockchain employs a graph-based architecture called Directed Acyclic Draph (DAG). A DAG is a directed graph that has a topological ordering, a sequence of the vertices where every edge is directed from earlier to later in the sequence. Combined with blockchain, each vertex of the DAG represents a new block at a certain time, and each edge represents the propagation path.

More specifically, Byteball [GBYTE17] put forwards the witness mechanism to form the main chain. Whenever a witness sees a vertex, he More specifically, Byteball [GBYTE17] put forwards the witness mechanism to form the main chain. Whenever a witness sees a vertex, he openly declares the claim, and the system selects the mostly seen vertices to form a main chain. Following this principle, the selection security can be guaranteed only under the condition that most of the witnesses would not commit conspiracy attack with the attackers. However, in order to motivate witnesses, Byteball charges a certain transaction fee to witnesses as a reward. IOTA brings the tangle concept to describe the topology of arbitrary main chains. When the new one generated, each new vertex will select two foregoing vertices as the parent vertex to start PoW mining. IOTA would distribute more weights to sub verticals according to the support of parent vertex relations. However, there exist a certain vulnerability inside the anti-quantumn hash algorithm, which may result in the leakage of secret keys at certain probability. Nano [NANO17] arranges a separate chain for each account, whose relevant information will be updated when the transactions are sent or received, so that each chain only processes transactions related to itself, greatly reducing the cost of processing time. Nano uses a hybrid consensus mechanism of DPoS plus PoW, and the ontroversy will be decided upon voting or arbitration. In practice, however, Nano's account-chain architecture makes the token transfer inconvenient for the centralized exchange, since the exchange's account requires a very large number of transactions.

In the meanwhile, a large part of the academic research on DAG consensus has been carried out. DAG Labs team in Israel, has proposed several protocols on top of DAG chains. Inclusive blockchains [LSZ15] extends the Nakamoto consensus to DAG and specifies a framework to include off-chain transactions. PHANTOM [SZ18] specifies the total order in DAG-based blockchains. SPECTRE [SLZ16] specifies a non-transitive partial orders for all pairs of blocks. At the same time, Conflux [LLZ18], a team from China, furthermore provides a total order over all transactions on DAG to build up a scalable decentralized blockchain system.

## *1.2. Sever-less*

The development of cloud computing has come along the process from IaaS, PaaS and SaaS to FaaS nowadays, in which server-less stands out ever more vividly. LaaS is Infrastructure as a Service. Service providers provide the fundamental infrastructure for underlying physical layer (including server, data center, environmental control, power source, server room), while Users need to purchase virtual resources, select operating systems, install software, deploy programs, and monitor applications through the platform provided by IaaS.

Here we give an example by to describe the developing trend of cloud computing. May wants to run a couture, what services can she get form IaaS, PaaS, and Faas. 1) IaaS will spare one the need of building the Shoppe. While May is in charge of customizing the shoppe, making clothes and others. 2) PaaS can spare one the need of building a shoppe or customizing it While May still need to make clothes and take care of the others. 3) FaaS can take care of all the necessary preconditions, While May only needs to send over the clothes and decide the selling strategy. By the comparison above, the advantage of FAAS is pretty clear.

FAAS is the next generation of serverless, and also the future of cloud computing. It has following features:

- Application logic units in FaaS are considered as functions and developers only need to focus on the logical implementation of specific functions. By this way, they can focus on optimizing the method rather than lose focus over the entire application, which completes the optimization of the Macroscopic performance.

- FaaS is stateless, and naturally meets the requirements of the Cloud Native App. Stateless means that local memory and data on the disk are not available for subsequent operations. Most of the states depend on the external storage, such as databases, network storage, and so on.

- FAAS functions should be able to start execution quickly and have a short life cycle. They start and process in a limited time, and terminate when the result is returned. It should also be terminated if its execution time exceeds a certain threshold.

- FaaS needs to map the request route and the handler function through the API Gateway, and returns the response proxy to the caller.

With the popularization of network, mobile devices and the Internet, various Internet enterprises have sprouted up, and provides customized services for people from all industries. Of China's top 500 companies, Internet companies account for 15 % and contribute over 30% of China's GDP. The key supportment of this figure is cloud computing, the foundation of Internet companies. And it is expected to reach \$ 143.53 billion by 2020. The market of global public cloud service is growing rapidly, and the compound growth rate is expected to reach 16.44% in 2017-2020. Cloud computing service is becoming more elaborate, and more convenient for businesses use. As long as the enterprise is connected to the network, it can use the applications running on the cloud directly through the browser then, without the need to worry about such trivia as installation, and free of the prophase cost on investment of hardware and software . Therefore, it can be predicted that lower operation cost and finer granularity will be the first choice of enterprises in the future. FaaS is such a service.

## 2. Paradigm

### *2.1. Terminology*

**Paradigm** a professional concept in philosophy of science, put forward by Thomas, Kuhn, a famous American philosopher of science in 1970. The paradigm represents the world outlook and behavior patterns that agroup of researchers engaging in a certain science obey together, including three aspects: common basic theories, ideas and methods; common beliefs; a common view of nature. The paradigms answer the question of the truth of the existence of things, the relationship between the known persons and the educated persons, and the theoretical system of the research methods, and in turn perform the three levels of ontology, epistemology and methodology. As a recognized model or mode, we hope to extract potential invariants in complex abstractions and construct a complete paradigm of blockchain networks.

**Logo-Nautilus** the Nautilus can be abstracted in golden spiral of mathematics and expressed by in spherical coordinates. The function does not change its structural expression under the integration or the differentiation of infinite dimensions. The invariable paradigm extracted from the changing mathematical expression is our moral. Studying the general logic of the change structure of blockchain networks enables our structure to be the underlying paradigm.

**Consensus-Binary-star** Binary-star represents for ”twins” and it is associated with the twins attracted by gravity. In astronomy, Binary-star consists of two stars orbiting around their common barycenter. Binary star systems are very important in astrophysics because calculations of their orbits allow the masses of their component stars to be directly determined, which in turn allows other stellar parameters, such as radius and density, to be indirectly estimated. Paradigm names our specifies two-layer hybrid consensus mechanism as Binary-star, to vividly represent the mutual interactivity and promotion between the two layers. Accordingly, we also names these two layers with Accretion and Sequentia, which will be discussed in detail later.

### *2.2. About Paradigm*

Paradigm is a DAG-based, FAAS-oriented, distributed Blockchain system, is the platform for future DAPP development. Its design is combined of a specified two-layer hybrid consensus mechanism. It is of two mutually coupled layers – accretion and sequence, based on solid technical containing directed acyclic graph, Quantumn Channel Protocol and pluggable VM instruction set. Paradigm aims to implement a high performance DAG-based FAAS network with flexible scalability, high performance and friendly development. Through Paradigm, any person or any system will be able to achieve the real-time point-to-point value exchange. All enterprises can also deploy functions as a service architecture (FAAS) on the distributed blockchain network.

### *2.3. Mission and Vision*

The distributed function network based on Paradigm can address the problem in the blockchain industry: low performance, weak security, and poor privacy. Paradigm will focus on establishing a brand new underlying blockchain system, to gradually form a sound business environment; which will release the actual productivity of the public chain.

## 3. FAAS+DAG

Paradigm employs the DAG as the underlying structure. It turns to be stable and reliable for DAG-based constructions in many projects such as IOTA and Byteball. Mathematically, DAG is a directed graph that has a topological ordering, a sequence of the vertices such that every edge is directed from earlier to later in the sequence. The reachability relationship in DAG can be formalized as a partial order on the vertices of the DAG. In this partial order, two vertices $u$ and $v$ are ordered as $u \leq v$ exactly when there exists a directed path from $u$ to $v$ in the DAG. Therefore, we can employs the DAG to recognizes blockhain network. Each vertex of the DAG represents a new block event, at a certain time, and each edge represents the propagation path.

Different from chain-based linear structure, DAG-base structure may concurrently proceed with frequent intersection. But the parallel propagation path does not means a conflict. Each DAG can form an unique main chain including all paths, the orderings of the vertices such that the starting endpoint of every edge occurs earlier in the ordering than the ending endpoint of the edge. The existence of such main chain can be used to characterize DAGs. As the DAG goes along with time, the fork edge will also be contained in instead of being abandoned in traditional blockchain while the main chain will spontaneously form, although sometimes outside the main chain in DAG . In addition, similar to the confirmation of previous block in traditional blockchain, each new block event in the DAG also needs to confirm its parent and ancestral event. Tampering past records in confirmed transactions requires the modification of an increasingly large peer and ancestral events. These two processes make DAG-based network achieves the feature of chain-based network.

Furthermore, here we list several advantages of DAG-based structure. 1) Fast and Concurrent: The confirmation times reduce to millisecond, and the concurrent transactions can furthermore make it fast. 2) High-throughout: Paradigm allows for a large transaction throughput, limited only by the endpoints capacity. 3) Low-cost: Common transactions are almost free of fee, which perfectly supports the IoT data and micro-payment. 4) Anti-orphaning: It avoids the risk of orphaning, which comes with many additional benefits.

In combination with the FAAS structure, there are two similarities between FAAS and DAG. 1) **Scalability:** DAG is of good scalability and unlimited node growth. In FAAS, one function does only one thing, and can expands the volume independently without worrying about affecting other functions. And it can expands faster because of its smaller granularity. 2) **Event driven:** In FAAS, functions does not publish any services, and does not consume any resources when no request is proposed. Resources will be consumed to respond only when there is a request, and release immediately after the service. Due to this, FAAS naturally applies to any event-driven business scenario, such as advertisement bidding, identity verification, timed tasks, and the emerging IoT applications. Messages are also transmitted by events in DAG network, which can be used as the event propagation channel of

FAAS.

Besides above-mentioned similarities, FAAS and DAG also have natural complementarity. Functions in FAAS is stateless, and the IPO (input-process-output) of the function essentially determines the stateless nature of the function. The FAAS architecture can only use external storage for two reasons: First, whenever the function request is executed, the context needed for the function execution is cleared. Secondly, each launch of processing may be scheduled to a new server, any cache on local disk is no longer useful. However, DAG network is naturally a database, which can be in charge for the storage of the state data which may be required during the function execution in FAAS, which is also hard for attackers to tamper.

Generally, FAAS architecture is the direction of the future cloud computing, and the DAG-based distributed ledger is also the best solution for storage and expansion. Therefore, Paradigm firstly puts forward the theory of DAG+FAAS that integrates the data storage of blockchain and the server-less architecture of FAAS which realizes the function as a application on the chain, and creates a environment-friendly future ecosystem of minimize resources wasted. Such a combination not only reduces any development difficulties, but also meets different scenarios. It will be seen as the trend in the future.

## 4. Technique Overview

### *4.1. Architecture*

Paradigm aims at building a high-performance,flexibly scalable and develop-friendly heterogeneous network with the mission of serving commercial level applications. Paradigm network can be divided into four layers according to the architecture hierarchy including 1) Application access layer with API gateway $Gw^2$ and the supporting components of Autoscaler and Dispatcher 2)Functional computation layer with Function Execute Engine $Fn^2$ 3)Consensus layer with Binary Star Consensus $BSC$ 4) Network communication layer with Quantum Channel Protocol $QCP$.

- $Gw^2$**:** API GateWay is on the top of the architecture which is the window of the node providing services to the outside world, and the converter of the internal and external protocols.

- $Fn^2$**:** It consists of Function Execute Engine and the supporting components of Autoscaler and Dispatcher. The Function Execute Engine is the fundation

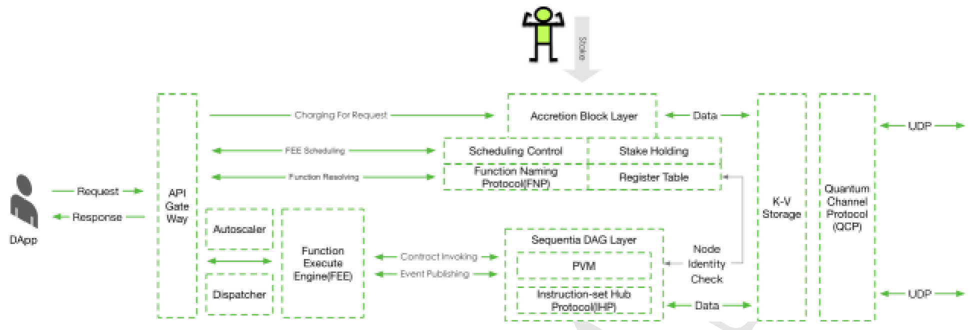


Figure 1: System Architecture

of FAAS, which is responsible for the allocation, addressing, and execution of Function calls .

- *BSC*:Binary Star Consensus that consists of the accretion and modular network.The accretion and modular network based on the binary star consensus algorithm provides the consensused status data for the entire computing platform, and a trust-worthy contract execution environment, which provides the distributed and reliable operation preconditions for the stateless function calculation.

- *QCP*:It enables a fast and low-latency communication between nodes, which can guarantee the performance of the communication between transoceanic nodes and IOT devices in a mobile network.

Ethereum, as a representive of existing underlying chain systems, its Virtual machine (EVM) provided to execute smart contract and application logic using and generating data, are all based on the datas stored in the smart contract accounts. In the traditional Internet applications, logics and datas are considered as a whole in which the update of the architecture will face a lot of inconvenience during the iteration. The raw data may face changes once the logic changes. Especially the constant addition of new functions may face exponentially growing difficulty while maintaining large smart contracts.

$$Smart\ Contract\ + Contract\, Execute\ Platform = DApp$$

With the latest software architecture model combined with the latest blockchain technology, we propose the following future DAPP development architecture:

$$Scalable\ DAG\ Network + Scalable\ FaaS\ Compute\ layer = Future\ DApp$$

We believe, this is the future DAPP development architecture.

## *4.2. Features*

Paradigm elaborately proposes the general architecture, the underlying construction modules, the upper layer components which will be detailedly stated in latter sections. Here we briefly summarize the significant features of Paradigm.

**Flexible scalability** Both the API GateWay and the Function Execution Engine in Paradigm network are capable of flexible scheduling, which can create and destroy FEE instance dynamically according to the request pressure of the network. Applications developed on aradigm in the FAAS architecture also employees the idea of seperation of logic and data, which ensures the flexibility of application's development, maintenance and upgrade to the greatest extent. DAG-based blockchain network ensures the bettering of transaction performance under the increase of network nodes, and the instruction set bus protocol provides maximum expansion capability for the self-evolution of virtual machines.

**Lightning fast with high TPS** Based on DAG-based network structure and two-layer hybrid Binary-star consensus mechanism, Paradigm presents a fast decentralized blockchain system which can optimistically process concurrent blocks without unpredictable forks. Paradigm allows multiple participants to join in the blockchain concurrently. In addition, the KCP-based Quantum channel reduces around three times transition delay while ensuring the data safety as TCP provided. The specified structure, hybrid consensus and fast transition channel enable higher throughputs and faster confirmations.

**Developer-friendly** FAAS splits the application into function units, which are driven by events or requests. Developers will be more focused on business logic than application deployment and environment maintenance. The function computing engine of Paradigm supports all development language which minimizes the cost of the application development based on our blockchain. In addition, Paradigm provides a fast and easy-to-use smart contract over the virtual machine, to reduce the difficulty of upper development, which supports multiple high-level languages, hot upgrade of the command set, and the command contract market.

## 5. Core modules

### *5.1. Binary-Star Consensus*

In paradigm, the communication between the nodes, which executes consensus process, is based on the asynchronous Full Information Transfer (FLT) protocol. When the nodes start to communicate with anyone else, the selected connection objects are random, so the time when the same information arrives at different nodes may also be different. Therefore, for achieving ultimate consistency, the consensus of Paradigm is going to maintain a complete communication process for all nodes in the network. Moreover, to make the participants recognizable, the consensus of Paradigm also plays a role in maintaining the register table (including the nodes public key and port information) and propagating it to all of the members. Therefore, based on elaborate design and analysis, Paradigm proposes a two-layer hybrid consensus mechanism called Binary-star Consensus, which can ensure the consistency of information, and also has good performance and security. The complete Binary-star consensus contains two parts – Accretion and Sequentia, to vividly represent the process and function of consensus along with the time.

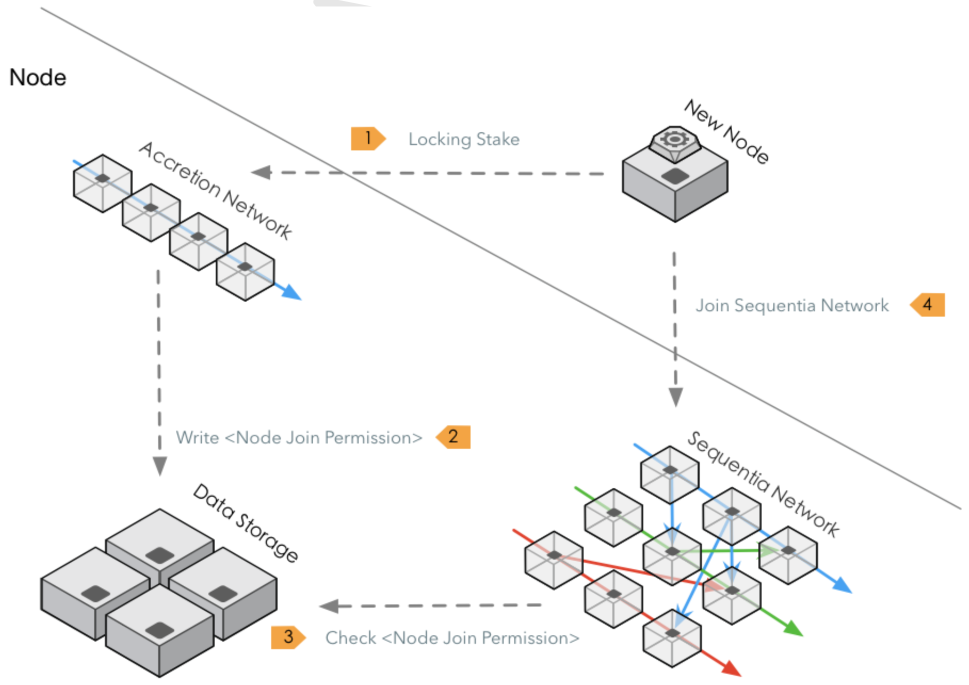


Figure 2: Paradigm Binary-star Consensus

### *5.1.1. Accretion*

Accretion, in astrophysics, is the accumulation of particles into a massive object by attracting more matter gravitationally, typically gaseous matter, in an accretion plate. Most astronomical objects, such as galaxies, stars, and planets, are formed by accretion processes. We adopt accretion process to represent our first phase, since the authority nodes in PoA (Proof-of-authority)-based network acts as a gathering role among nodes.

Accretion consensus network can be seen as a combination of PoA mechanism, an algorithm used in blockchains (such as Ethereum, POA network) that delivers and processes transactions comparatively fast through a consensus mechanism based on identity as a certificate, and stake pledge. In PoA-based networks, transactions and blocks are validated by approved accounts, known as validators, who run consensus process allowing them to put transactions in blocks. Different from the traditional PoA algorithm, when we generate the block, the blocker of the block is temporarily determined by the hash of the previous blocks, which adds some randomness to the order of the participants packing the block, so that the attackers outside cannot know which authority node is going to package in next round while inside ones can achieve an agreement.

Since there is a certain degree of trust in the participants in the Sequentia layer, we need a control mechanism to perform admission screening on the access nodes. If any node wants to participate in the Sequentia consensus process, it needs to pre-save some tokens to get a lease for a period of time, and the tokens will be returned if the node is still in the whitelist at the end of the lease period. Permission nodes can be dynamically connected and left, and they will be given some tokens as reward based on their contribution online. To be more specific, we describe the workflow and motivations of Accretion consensus network as follows.

- **Node identification** Each participant should be registered in a reliable table maintained by authority nodes. Regular nodes are identified by an address, created with an asymmetric cryptosystem, whose private key is also controlled by the node itself. To successfully join the Accretion network, nodes must submit their individual information such as Public key, Mac, IP port, etc., for later use in network connection and signature verification.

- **Register Table maintaining** Nodes already in Accretion net, which is also called authority nodes, play an essential role in access control. Every regular node can submit his information to the authority nodes as his passport to

DAG-basd Sequentia network. When the verification is passed, the authority nodes will add it to the member list. Each of the consensus nodes in Sequentia network needs to maintain this list and ensure its consistency in order to make some accurate decisions in consensus process (details will be introduced in next section).

- **Preventive Deposit** The deposit from participants acts as an preventive way to avoid doing evil. The deposit brings a threatening constraint to every nodes. According to different lease periods, the exact deposit amount varies. At the first phase, the regular nodes submit their information, which not only contains the identification but also the deposit tokens. The complete rules and process are coding in smart contract to ensure its transparency. The program will be automatically executed whenever triggered. The deposit will keep the node honest in maximum.

- **Financial incentive** Paradigm also includes incentives as a reward for contributing to consensus nodes and encouraging more nodes to participate in system maintenance. The incentive mechanism is designed in the Accretion layer, sharing state with Sequentia layer. Who do not engage in evil behavior will receive a certain amount of tokens. In terms of quantity, it will consider the number and frequency of the Comets generated.

- **Anti-Sybil attack** A reputation system's vulnerability to a Sybil attack depends on how cheaply identities can be generated, the degree to which the reputation system accepts inputs from entities that do not have a chain of trust linking them to a trusted entity, and whether the reputation system treats all entities identically. The Accretion network forces every participant provide some deposit, which increase the cost of faking identities in network. The costy identity reduces the motivation of cheating and tampering, which effectively avoids Sybil attack.

#### *5.1.2. Sequentia*

Sequentia, driven from music term, refers to proceedingly repeat the theme melody or phrases. Each shift has a direction targeting forwards or downwards and each round will accumulate along with time, which is similar to the rapid shift in musical sequentia. We adopt sequentia to represent our second phase, since the DAG-based network are orderly proceeding during the seemingly disordered propagation. By the way, all transactions and smart contract mainly created, packaged, and executed in the Sequentia layer.

At present, many blockchain projects use PoX system algorithms, such as PoW, PoS, or proof of other attributes instead of work or stake. In such algorithms, nodes generate blocks by competing for accounting rights, but there is no guarantee that only one block will be generated per round. When two or more blocks of the same height are born, the fork appears. Actually, conflicting blocks are often effective. Selecting one to add in main chain creates unfairness and waste of resources, and the handling of the fork will also result in inefficiency, delay in transaction confirmation, etc. To avoid these issue, Sequentia adopt graph instead of the traditional chain structure. Without any pruning, miners can generate blocks at any time according to their needs, regardless of stale. In this way, the scalability of the system can be improved and the transaction can be confirmed more quickly.

Another commonly used scheme is the improved algorithm of PBFT; a leader-based algorithm is vulnerable to attacks on the leader, which causes the delay of the eventual consistency. In addition, the consensus nodes often need to vote multiple times on the leader's proposal, which may lead to $O()$ or even $O()$ messages. Therefore, considering the communication burden of the network, this algorithm cannot bear too many nodes. While, Sequentia maintains the entire message graph so that nodes can calculate decisions locally without additional communication. And because the data stored in nodes is same, the algorithm used in nodes is same, so the results of decisions are guaranteed to be the same.

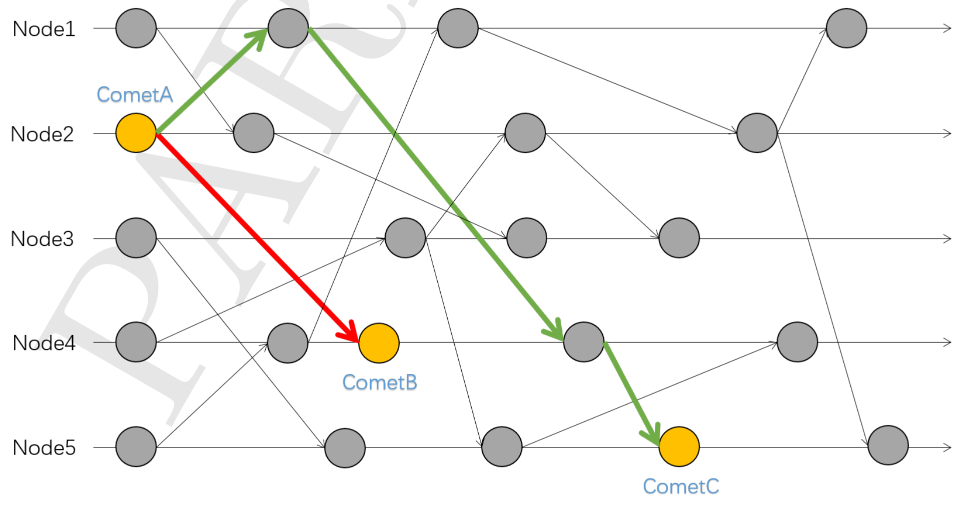


Figure 3: An example of Cometgraph

In conclusion, Sequentia consensus network is a leaderless, asynchronous and DAG-based decisive system. The concurrent feature it owned can make it possible for processing thousands TPS with sub-second latency. The gossip protocol is used to propagate information between nodes, each node transmits its own known information to others as much as possible. When a node receives information from another, it then creates a Comet as a record point. The Comet is a data structure containing:

Table 1: The structure of Comets

| | |
|---|---|
| *parent_Index* | Index of Comets that is directly related to the one when it created. |
| *Version* | Version number |
| *Index* | Usually represents the number of Comet generated by the node. |
| *Timestamp* | Creation time |
| *Signature* | Signature information of the node |
| *Transactions* | Zero or more transactions |

The *parent_Index* represents a relationship, as shown by the edge in the figure. A beveled edge represents a network communication from remote node to local. The vertical edge is a chronologically stored Comets created by the same node, structurally similar to chains. Each Comet usually connects two parent Comets, which are generated earlier and connected directly with himself, and one Comet can be the parent of multiple Comets. Through the signature information and parent index recorded internally, the constantly generated Comets form an irreversible Cometgraph. Each consensus node needs to store a complete Cometgraph to facilitate subsequent processes. Although the latest part of the graph stored by each node may be slightly different, new communication will quickly eliminate this difference over time.

Through the indexing of *parent_Index*, the generated Comet can prove the existence of the early generated. As shown in the figure 9, we can find *CometA* from *CometB* by backtrackingthe red line, thus we can say that *CometB* can prove *CometA*. When this proof path passing through more than 2/3 nodesthe green line from CometC, it can be considered that this proof has been known by enough nodes, and the proof becomes a strong proof, which is Byzantine fault tolerance.

By proving and strong proof, we can further determine whether a Comet has been successfully received by most nodes, in other words, we can ensure that most nodes have established a partially complete and consistent Cometgraph near the Comet. For a Comet in acknowledgment state, we can calculate the relative creation time of this Comet by calculating the median of the receiving time at different nodes. All confirmed Comets are sorted by timestamp, and finally each node can execute all the transactions contained in them and change the corresponding state. This process will be repeated in Sequentia to provide consistency for the Paradigm system.

## *5.2. PVM*

The smart contract was first proposed by Nick Szabo. He defines smart contracts as "a set of promises defined in digital form, including agreements on which contract participants can execute promises." The smart contract has not received much attention since its submission due to its highly unpredictable risk. Until the emergence blockchain, smart contract becomes credible since the entire states and transactions based on chain can't be modified or deceived.

The first successful application of the blockchain is Bitcoin, but it is a non-turing complete mechanism (derived from Alan Turing, a mathematician who introduced the Turing machine concept) which cannot support programmable operations. Ethereum embeds a specified virtual mechine, called EVM, into the chaincode and achieves the programmable operations. However, subject to language restrictions, it cannot inherently support conventional high-level languages such as Java and C++. The Solidity language intrinsically lacks a standard library. Although the stack-based architecture is easy to optimize, but it requires more *OP_CODE*s. NEO, as the first large-scale public chain in China, implements a lightweight virtual machine compatible with multiple languages. But it cannot support parallel processing itself, and the performance of contract execution is poor.

The formal definition of PVM is the Paradigm Virtual Machine for deploying smart contracts. PVM can smoothly interact with the existing developer ecosystem and meet the requirements of the consensus mechanism. During the contract operation, the PVM has various features such as Turing complete and multi-language support and pluggable instruction sets. And during contract development and deployment, Paradigm provides AI-based security rating engines which is embedded in the Web IDE and upper layers to protect digital asset threats caused by smart contract vulnerabilities.

### *5.2.1. WebAssembly-based Execution*

PVM provides a completely isolated environment to process the smart contracts inside blockchain. The execution process includes transaction processing, transaction preservation, and contract processing of a complete state machine. The virtual machine completes the corresponding code logic by executing the binary sequence code loaded by the engine , storing it in the stack, fetching instructions from the top of the stack, and then at last executing the stack data operations step by step according to the instructions. As the stack-based virtual machine, PVM employs WebAssembly (abbreviated wasm) as the binary instruction format. Wasm is designed as a portable target for compilation of high-level languages like C/C++/Rust/Go, enabling deployment on the web for client and server applications. Therefore, the developers can easily create smart contracts in Go/C++/Java and other programming languages to develop on-chain applications. In addition, the wasm-based PVM brings significant advantages. For example:

- **Fast, efficient, and portable**: PVM smart contract code can be executed at a almost native-run speed across different platforms.
- **Secure**: The smart contract code runs in safe sandbox. The isolated environment protects threats from attackers.
- **Readable and debuggable**: The wasm have a human-readable text fomat which allows us to view, check and debug the code.

### *5.2.2. Instruction-set Hub Protocol*

With the development of distributed ledger technology (DLT) and the implementation of decentralized application based on distributed ledger technology, enterprises and developers are more and more into DApp which is developed underlying the public chain such as Ethereum and EOS. It is estimated that there are about 500,000 to 800,000 developers on Ethereum, at least 20,000 open source code repositories, and at least 5,000 DApps across all industries.

As a DApp developer, various libraries and basic components of the software itself will be implemented in smart contracts, which are often inefficient to iterate the components. We now introduce to you a new virtual machine protocol that is self-evolving called the Instruction-set Hub Protocol (IHP).

A set of instructions is stored in the CPU which will be used to guide the CPU to execute operations and control the operating system. The blockchain virtual machine instruction set refers to a set of interfaces which indicates underlying functions that are applied to the virtual machine. There are many shortcomings in existing blockchain instruction set, for example the update of the instruction set relies on a hard fork of the whole network. After analyzing the implementation of several technical solutions, Paradigm decided to cut off the complicated designs and build a more general and agile instruction set. IT abstracts the general instructions of operations under different application scenario, and realizes the pluggable design of the instruction set. The iteration is dynamical rather than simply relying on the hard fork.

Table 2: Pluggable Instruction sets

| **Operations** | **Instruction Set** |
|---|---|
| Arithmetic | *ADD*, *MUL*, *SUB*, *DIV*, *SDIV*, *MOD*, *SMOD*, *EXP* ... |
| Logic operation | *LT*, *GT*, *EQ*, *ISZERO*, *AND*, *XOR*, *OR*, *NOT* ... |
| Business funtions | *SHA*3, *ADDRESS*, *BALANCE*, *CALLER*, *GASPRICE*, *LOG* |
| Asset transactions | *TRANS*, *REG_ASSETS*, *FREEZE* ... |
| Privacy | *ENC*, *DEC*, *SIGN*, *VERIFY* ... |
| IoT | *PARSE_QRCODE*, *PARSE_NFC*, *PARSE_RFID* ... |

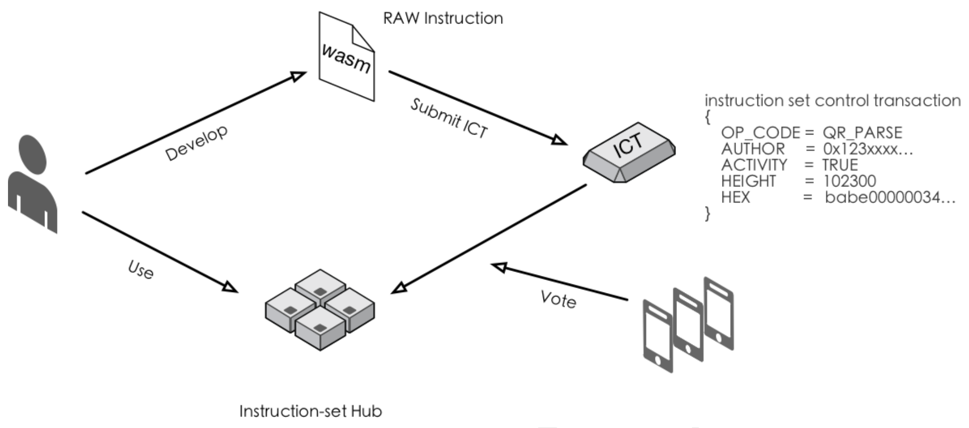
The pluggable instruction set is implemented by the registry, which is essentially a Hub. It can update the instruction set on the chain without forking, and encourages community maintenance. The Paradigm extendable instruction set will greatly facilitate the business applications and lower the threshold of real business.

The activation, iteration and inactivation of instruction set are processed based on "instruction-set control transaction (ICT)". The initiation of ICT can be proposed by any accretion consensus node. The ICT will be spread through the internet, votes will be counted to judge whether it is valid or not by a variable voting threshold. Any wallet with none-zero Paradigm tokens can make a vote to the operation of an instruction via the wallet's voting function. When the ICT is effective, developers or enterprises that initiate the ICTthat is, the vote is mostly passed, then the token bonus of the instruction invocation and community development incentive can be received in the future to maintain the healthy development of the IHP.

#### *5.2.3. Function Naming Protocol*

Currently, in the interaction of various block chain network, the address has been a huge drawback because of its long and unfriendly-to-remember characters. It has limited the transmissibility and practicality of the block chain. In order to solve the pain points of network value transmission, Paradigm comes up with the FNP solution. It is a build-in domain Name system, a distributed, extensible, multi-purpose, and easy-to-use naming system. Different from the ENS, one doesnt need to deploy a smart contract, but at the same time it can implement a unified identifier for the on-chain assets.

Inspired by the HTTP protocol, FNP generates a system of dot-separated hier-



```
instruction set control transaction
{
   OP_CODE =  QR_PARSE
   AUTHOR    =  0x123xxxx...
   ACTIVITY   =  TRUE
   HEIGHT     =  102300
   HEX          =  babe00000034...
}
```

Figure 4: Paradigm Instruction set Control Transaction

archical names called domain. It enables the connections between identifiers and assets, transactions and accounts. All registrars' operation is under Paradigm protocol. A legal domain should be like this, for example " paradigm://part/gipes.org/account/data ", " paradigm:// " is the default protocol header, " gipes.org " can be an organization, a person or a transaction. " par " can be the subdivision of the organization or the operator. We use slash to separate the method to call or account to operate. All the domains will be stored in the metadata.

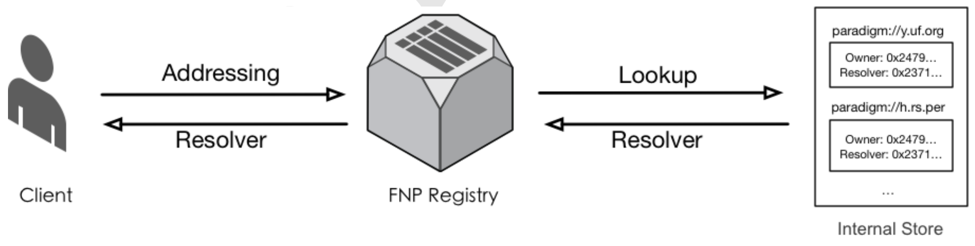


Figure 5: Paradigm Function Naming Protocol

FNP has two principal components: the registry, and resolvers. The FNP registry stores three critical pieces of information about each: The owner of the domain, The resolver for the domain, The time-to-live for all records under the domain. Here gives several FNP relative interfaces, which maps domain names to owners and resolvers.

- SetOwner(address []byte, domain string) bool
- SetResolver(domain string, resolver Resolver) bool

*5.3. Quantum Channel Protocol*

The transmission of data from node to node usually requires extremely high efficiency and security. In Paradigm, such transmission mode is called The Quantum Channel.

In the Binary Consensus algorithm adopted by Paradigm Network, in order to ensure the stability and efficiency of Network communication in the process of binary consensus, if the traditional TCP communication is adopted, the connection between nodes should be established before communication and disconnected after communication, which will consume more computer resources when ensuring the reliability of transmission, so that there will be relatively high resource consumption on network processing when the node size increases. If UDP communication is adopted, the reliability of communication is relatively reduced even though the resource loss required for connection establishment is not achieved. We need a protocol between UDP and TCP that has both UDP transmission speed and TCP reliability. Quantum channel has such characteristics, which is essentially a fast and reliable ARQ protocol. We will insert an encryption layer between the QCP layer and UDP layer to encrypt the data inputted by the UDP layer. At the same time, when the data packet is lost, it will be selectively re-transmitted to ensure the security and reliability of the data packet. QCP can provide a reliable transport while preserving UDP efficiency. As a conclusion, QCP has the following technical features:

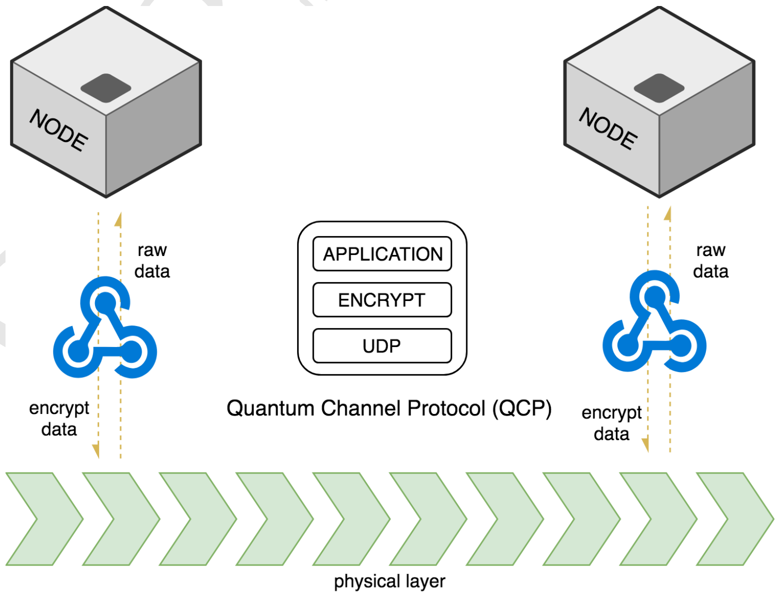


Figure 6: Paradigm Quantumn Channel Protocol

- A single server can support 10k concurrent connections, which makes it possible to support massive concurrency on a single machine thanks to UDP's stateless and flexible and efficient thread scheduling.
- Supporting the transmission layer encryption protocol based on Diffie-Hellman, the encryption simultaneously achieves forward security, that is, the existing key leakage does not cause the information on the previous channel to be cracked.
- The forward error correction is added, which greatly guarantees the data reliability under UDP.

With the development of portable devices and Internet of things, the interaction between nodes and nodes will be more in 3G and 4G network environments. Therefore, we conducted a simulation test, and compared Enet and QCP with the random addition of delay and network jitter. QCP can still perform well.

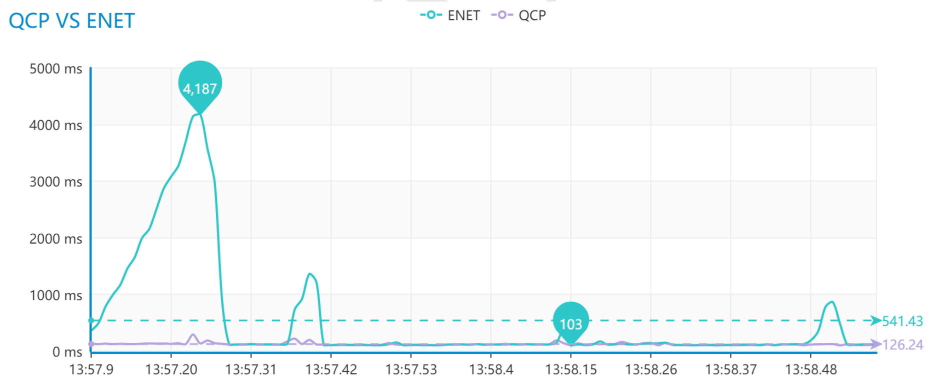


Figure 7: Paradigm QCP Test

## *5.4. Decentralized FAAS Network*

A application is mostly made up of many small components. Each of these components contains a small set of input and the context. It needs about 10 to 100 milliseconds to complete the work. In the traditional Internet development, we must deploy corresponding servers to support the application which often fails to achieve the load balancing. In order to solve this, draw on the experience of Amazon AWS, we have developed a serverless computing service, FAAS (Function as a Service). Paradigm FAAS enables processes to run beyond their current runtime and environment, which leads to a faster accessing and persistenting to the world state of th blockchain. Compared with other enterprises, users on Paradigm possess a more beneficial site, such as price, market timing, etc.

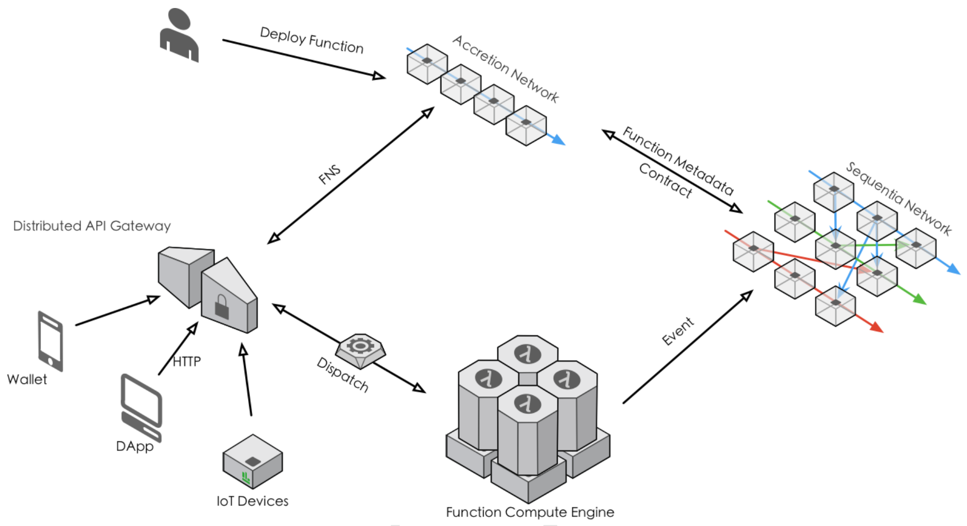


Figure 8: Paradigm FAAS Network

Traditional FAAS software architecture based on public cloud technology is usually divided into API Gate-Way layer, function computing layer and data persistence layer. Similarly, in the distributed network, we will build a completely distributed FAAS network based on Paradigm's Binary-star consensus algorithm and DAG technology.

The FAAS network based on Paradigm is composed of parallel nodes, which communicate through QCP. Within the nodes, there are four modules: distributed gateway, function compute engine, Sequentia DAG network, and accretion consensus chain network. Paradigm FAAS has following outstanding features

- Event driven
- Less fee
- Elastic scheduling
- Developer friendly
- High performance

### *5.4.1. Distributed Gateway*

The distributed gateway $Gw^2$ acts as the gate of the whole system, in essence, it is used to translate the HTTP request of external application into specific function to execute the request, as well as basic traffic control and security check. After receiving the request, the distributed gateway will perform FNP addressing with the incoming request parameters, find the specific function address by the function domain, deserialize the parameters, and re-package the request into function execution request then pass it to the function execution engine. At the same time, the request times of statistical function are regularly submitted to the Accretion network with special transaction.

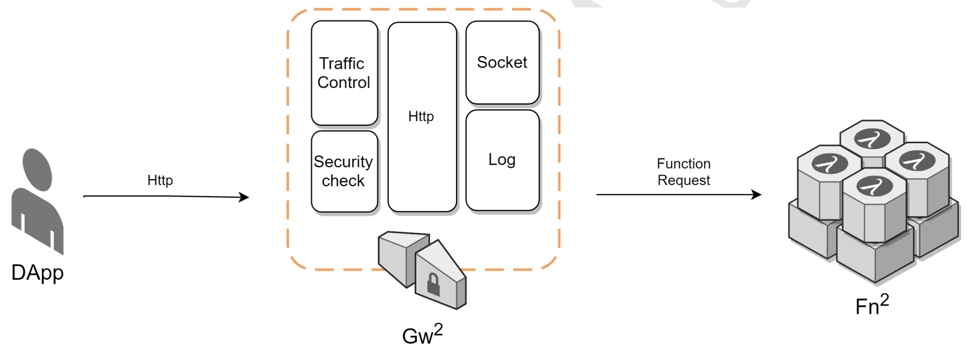


Figure 9: $Gw^2$

*5.4.2. Accretion Network*

The Accretion network maintains the function registration service of the whole network. The developer registers the developed function to the network through the function registration transaction, and the whole network will synchronize all the information of this function. Meanwhile, the mapping relation of FNP in the network is also maintained to support the FNP search operation. At the same time, Accretion network dynamically sends the flow control instructions of the gateway according to the scalable-capacity algorithm after receiving the statistical information from the distributed gateway, so as to achieve the dynamic change processing capability with the change of traffic.

When the network faces the situations such as Crypto-Kitties, the trading volume on the entire network had increased sharply. This oftens causes congestion on the blockchain network, resulting in higher fees and longer transaction and confirmation time. Traditional Internet applications, most of the time the number of requests is very sustainable, CPU, memory, and other computing resources utilization rate is low. But for some reason situations, such as staff off at 6 PM or certain activities, will result into huge requests, and cause surging computing resource utilization, network jitter, RT, which leads to the problem such as quality of service. We combined the thinking of Internet FAAS to apply flexible scheduling in the blockchain network for the first time. Elastic scheduling means that when a number of function calls , transactions in the network surges, the node will calculate the number of theoretical capacity expansion through the expansion algorithm, and create the corresponding function compute unit instance to cope with the processing pressure. When the internal detection thread finds the pressure reduction, the number of function compute unit to be maintained is calculated by the shrinkage algorithm, and the redundant unit instances are gracefully exited, finally destroyed to release the computing resources.

### *5.4.3. Function Compute Engine*

Function Compute Engine is denoted as $Fn^2$. AutoScaler is an automatic expandable component that enables it with expansibility for FEE cluster. It provides configurable thresholds of its upper-lowerlimit for FEE's capacity, calculates the pressure and status of the API Gate Way and FEE cluster, and reports to the accretion network. AutoScaler will calculate the mathematical median pressure of the period to keep up with the demand, to adjust the expansion capacity according to the preset thresholds, and adjust the scale of the FEE claster according to the feedback from the accretion layer, which may trigger the destructionoperation if it's blew the lower limit. The expansive capacity function of AutoScaler provides an effective solution for sudden traffic peak, and provides effective resource allocation on demand which is more reasonable. Dispatcher is a high-speed, unlocked, circular queued cache dispatcher, that loads requests coming from the API Gate Way to FEE in a more balanced way.

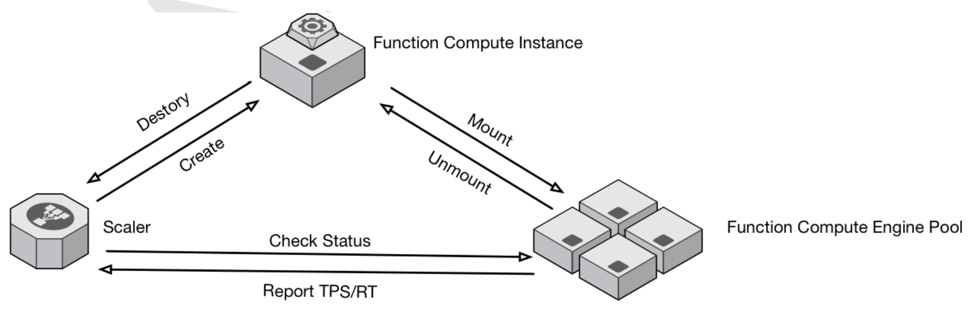


Figure 10: Paradigm FAAS Engine

The function execution engine is the core processor of FAAS. It loads the function in the function call request passed from the upper layer into the task pool with the execution, which is dynamically distributed to the idle processor through the load balancing algorithm. In addition, the function execution engine will calculate the cost of the function execution based on the code of each function execution and the time consumption. The functional execution engine theoretically supports all development languages, so any language developer can use our FAAS network.

Because the function is stateless, internal function should not keep any data, or maintain applications so when internal logic function need to manipulate data for a user, or business use, it will trigger an event which includes the function, developers, business all of the unique identification of the organization, as well as the related data operation command and parameters, and will distribute this event to mold into the network.

### *5.4.4. Sequentia DAG network*

Sequentia network is a distributed network composed of event streams based on DAG, on which events spread and finally reach a consensus. The purpose of an event is to transfer value, which is captured by listening to the uniquely labeled smart contract in PVM, thus triggering data modification in the contract.

The operation of DApp is inseparable from logical control and data storage. In traditional architectures such as Ethereum, data and logic are maintained in a smart contract and executed by EVM serially. When it comes to applications such as CryptoKitties, the processing capacity is seriously insufficient, resulting in the congestion of the entire network. Transactions and applications cannot be processed and confirmed quickly. In the FAAS network of Paradigm, there is no such problem, because we innovatively separate the logic control from the data storage, the application logic is processed quickly by the efficient, flexible , expansive function execution engine, and the data changed will come to consensus and then be confirmed by the rapid DAG network.

## 6. Plan in Long Term

### *6.1. PVM Safety Rating Engine based on deep learning*

A smart contract is a piece of code on blockchain. Its structure is often single, but it is extremely secure. New potential attack points and attack methods may emerge. Paradigm develops a smart contract security rating system based on deep neural network and NLP technology, which can adaptively evaluate the security performance of smart contracts.

The figure above shows the rating framework, which mainly includes two modules of training and recognition. The purpose of the training module is to learn the

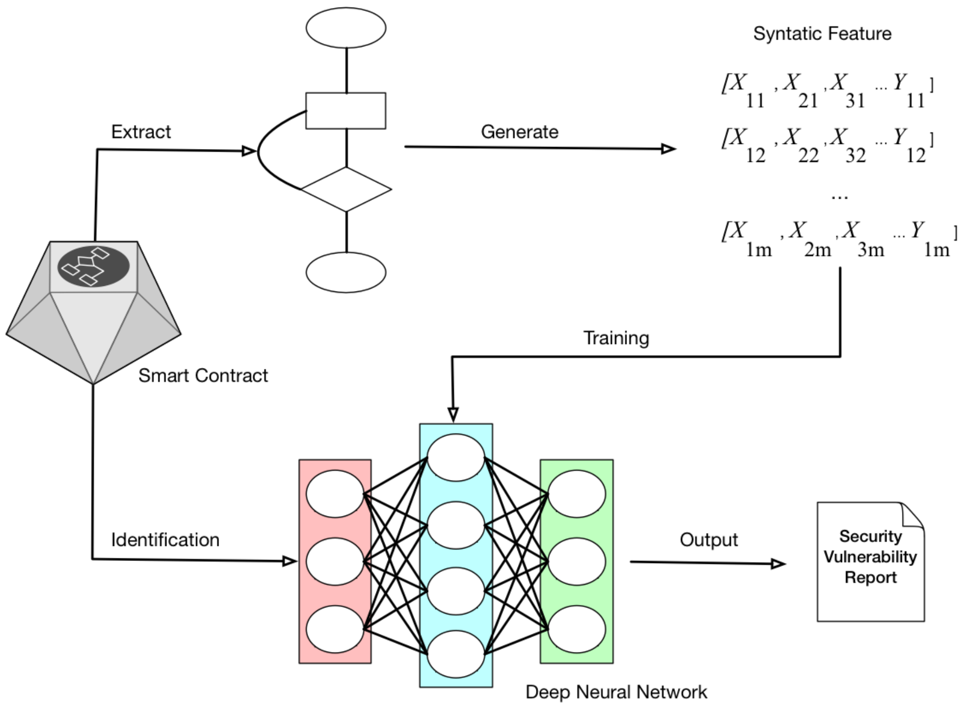

Syntatic Feature
$[X_{11}, X_{21}, X_{31} \ldots Y_{11}]$
$[X_{12}, X_{22}, X_{32} \ldots Y_{12}]$
...
$[X_{1m}, X_{2m}, X_{3m} \ldots Y_{1m}]$
Extract
Generate
Training
Smart Contract
Identification
Output
Security Vulnerability Report
Deep Neural Network

Figure 11: AI-based Security Rating Engine

semantic features of different smart contracts through deep neural network; the recognition module applies the trained neural network model to the detection of defective contracts.

In order to enable the neural network can process the contract code and learn its semantic features, we use a similar NLP semantic analysis technique to first transform the control flow graph of the smart contract into a semantic vector form. Specifically, we believe that the semantic vector of smart contracts should be composed of static features and dynamic features: static features are related to basic functional blocks and directed edges of control flow graphs, and dynamic features are related to execution paths and dependencies out of from the symbolic execution. Each smart contract is represented as a list of digital vectors. In the model training, the cross entropy loss function form is used and optimized by the stochastic gradient descent algorithm. Once the model is trained, it can be deployed in the contract development environment to rate the smart contracts across the network and locate the potential attack point where the contract confronts.

Due to the use of deep neural network algorithms, smart contract samples for model training should contain as many types of security flaws as possible, so that the neural network can learn different types of errors, resulting in greater. With the continuous expansion of Paradigm ecology, the number and types of smart contracts that can be acquired will increase, which makes the deep learning based security rating system also become more intelligent and more adaptable. Compared to symbolic execution relying on observing the execution trajectory of the program to find problems, the deep neural network approach is to proceed from a global perspective, and potential unknown attack points may be found ever before.

### *6.2. Functional Privacy Protection*

As to the technical realization, the difficulty of the encryption system of Paradigm lies in the problems of composite-order residue class and the security of anti-chosen-plaintext attack under the standard model. The features of the transaction scheme for Paradigm structure are as follows.

- **Being homomorphic**: It can realize the addition and subtraction operation under plaintext through the multiplying operation under the encrypted ciphertext.
- **Zero-knowledge**: It can ensure the correct verification without disclosing any valid information.
- **Effective and efficient**: The proposed scheme can be implemented in an highly efficient way.

After being encrypted according to the relevant cryptosystem algorithm, the ciphertext generated will be present in each transaction sheet. Paradigm employs zero-knowledge proof to verify whether the plaintext hidden by the relevant ciphertext satisfies the requirements of positiveness and equality. With relatively simple steps to be taken, zero-knowledge proof can ensure a smaller expansion field and also the positive number of secrets during the transaction process. According to the feature of hash value, it can be verified that the two commitment values contain one same secret without disclosing the value inside commitments. In other words, the total input should be equal to the total output during the transaction. On the premise of ZKP features, it can ensure that the value after the encryption should be accurate without being disclosed during the process of verification.

#### *6.2.1. General construction*

The general construction for Paradigm can be defined as:

$$\Pi = (HEnc, RangeVer, BalanceVer, Tx)$$

$\Pi_i$ means the construction of relevant algorithms, namely homomorphic encryption, range verification, balance verification and Paradigm system. The definitions of core modules will be given as follows.

### *6.2.2. Modular construction*

**Homomorphic Cryptosystem** Homomorphic encryption is a type of public key encryption. It is a cryptology technique based on the computational complexity theory aimed at mathematical puzzles. The encryption system has the feature of being homomorphic. The plaintext operation is hidden as the ciphertext operation. To be more specific, the encrypted data will be processed and then lead to an output for decryption. The result is the same as that of the same method in processing the original unencrypted data. Homomorphic encryption system can be classified as addition homomorphic algorithm, multiplication homomorphic algorithm or hybrid homomorphic algorithm according to the operation mode.

**Range Verification Proof** Range verification proof is used to prove that the commitment verification value falls within a certain range. The commitment scheme means that the sender sends a secret value to the receiver who is not informed of the secret value. After that, the sender will open the secret value and the receiver will verify it. The commitment scheme can be divided into two stages, namely promising stage and opening stage (or revealing stage). According to the range verification proof, the general range of the ciphertext value will be verified without opening the commitment value. Hence, it is highly confidential.

**Balance Verification Proof** Balance verification proof is a special commitment scheme. Traditional commitments aim to calculate a certain commitment value according to trapdoor information. The operation of two commitment values will be carried out by the trapdoor characteristic commitment. The characteristic feature only allows the people owning the trapdoor information to judge whether the secret values of the two commitments are equal to each other. They are not allowed to open the commitment. In other words, the characteristic feature of the concept lies in the judgment of the output result being 1 or 0 rather than the specific commitment value. Only when the secret key of the trapdoor is owned and the secret values of the commitments are equal, the output of 1 will be made. Otherwise, the output of 0 will be made.

**Paradigm System** With the structure of DAG, Paradigm system can be used to conduct the reliable distributed transaction of the data with the measurement unit of token. It consists of transaction sending, transaction receiving, transaction broadcasting and block confirmation. After it passes the confirmation from the validator, the whole-network broadcasting will be made. The original plaintext information on the transaction processed by the scheme will be hidden as the ciphertext which is not intelligible. Hence, it can ensure the privacy which is the only object for potential analysis and processing during the transaction.

### *6.2.3. Work Flow*

The highly-private transaction workflow of Paradigm can be described with the sequence diagram.

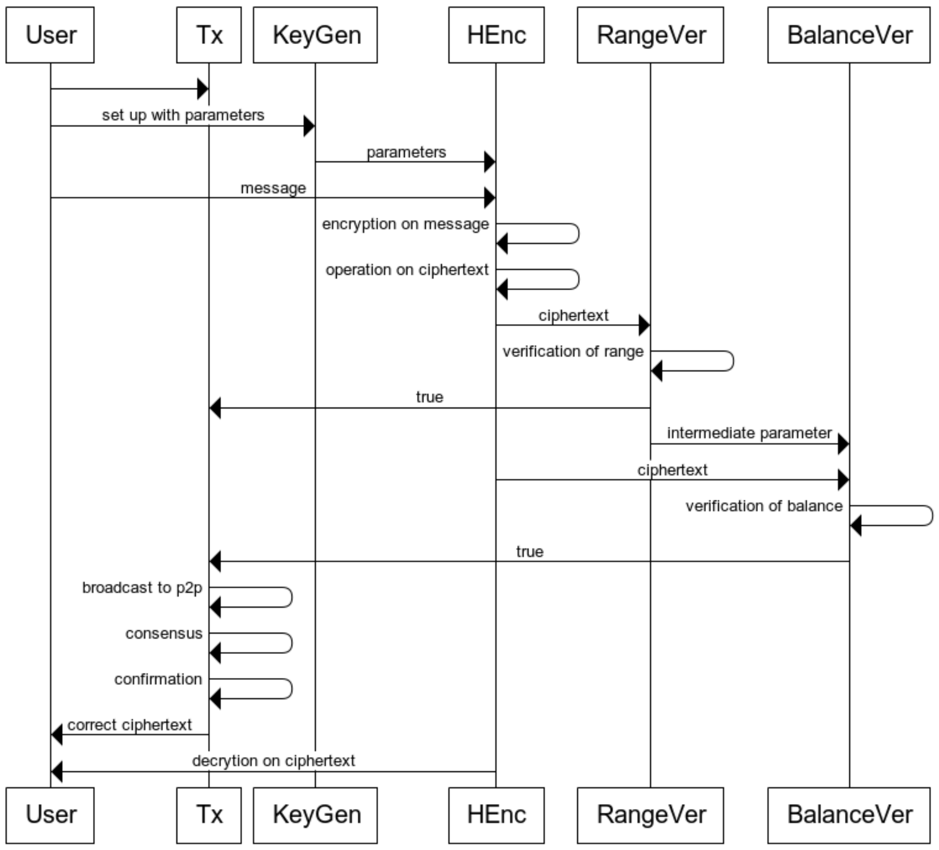


Figure 12: The workflow of Paradigm privacy scheme

## *6.3. Anti-quantum Crypto-components*

The approaching of quantum computers threatens the foundation of cryptography, and it is also a huge threat to blockchain-based cryptocurrencies. The atomic components of cryptography are closely related to the security in future. Paradigm aims to strength the robustness of the basic atomic components and improve the security level from the bottom to top. Therefore, Paradigm proposes the Sponge-based hash function and Lattice-based signature algorithm as our underlying design.

### *6.3.1. Sponge-based Hash Algorithm*

Hash algorithm is an open function that maps the arbitrary message into a short fixed-length value, called message digest. To keep the data from being tampered, the blockchain not only saves the original data, but also saves the hash function values. As a basic atomic component, hash algorithm is closely related to the security of the confirmed ledger on chain. At present, the most effective attack against hash function is GROVER algorithm under the quantum computer, which reduces the complexity from $O(2^n)$ to $(2^{n/2})$. Therefore, The RIPEMD160 algorithm in Bitcoin system can be easily attacked under quantum computer. One effective way to resist quantum attack is to increase the output length of the hash algorithm upto at least 256 bits. Therefore, selecting a strongly secure hash function requires 1)can avoid the traditional attacks such as differential analysis, message modification and so on. 2) can avoid the potential threat of quantum attack in the future.

Paradigm employs Keccak512, the official SHA-3 algorithm defined by NIST, as the basic atomic component to improve the chain security. Different from the traditional Merkle-Damgard (MD) structure, Keccak512 adopts the hardware-friendly Sponge structure, which achieves strong security level under the same cost. Sponge-based hash algorithm contains two phases, absorbing and squeezing. After iterated round function, the original inputs can be almost-perfectly confused. It turns that the keccak512 hash algorithm can resist quantum attack since the complexity of first preimage attack goes up to $2^{256}$.

### *6.3.2. Lattice-based Signature Algorithm*

Digital signature is a mathematical scheme for presenting the authenticity of digital messages or documents. It encrypts the messages with the sender's private key and verifies under the recipient's public key. A valid digital signature makes a recipient to believe that the message was created by a known sender, that the sender cannot deny having sent the message, and that the message was not altered in transit. Therefore digital signature possesses two features: 1) provide the authentication of sender; 2) ensure the non-reputation of event; 3) confirm the integrity of message. Existing blockchain system mostly adopts ECDSA as their digital signature algorithm. As the most widely used basic component, signature algorithm plays an essential role inside the blockchain. At present, the most effective attack against signature algorithm is SHOR algorithm under the quantum computer. SHOR algorithm is pretty suitable for addressing hard mathematical problems such as large integer decomposition and discrete logarithm inversion.

There exist three types of public-key cryptosystems (PKC) against quantum attacks mainly including lattice-based PKC, McEliece-based PKC and MQ-based PKC. The security of McEliece-based PKC is based on error-correcting code, but it is inefficient. MQ-based PKC is based on the intractability of multivariate quadratic polynomial equations over a finite field, but it has obvious shortcomings in security. In contrast, the lattice-based PKC possess the advantages of strong security and efficient implementation.

Paradigm employs lattice-based signature algorithm NTRUSign-251 to resist quantum attacks. The security of NTRUSign-251 signature algorithm is ultimately equivalent to finding the shortest vector problem (SVP) in a 502-dimensional integer lattice. As traditional computer, there is no fast algorithm to solve SVP under quantum computer. The best heuristic algorithm is exponential complexity, which goes to at least around $2^{168}$.

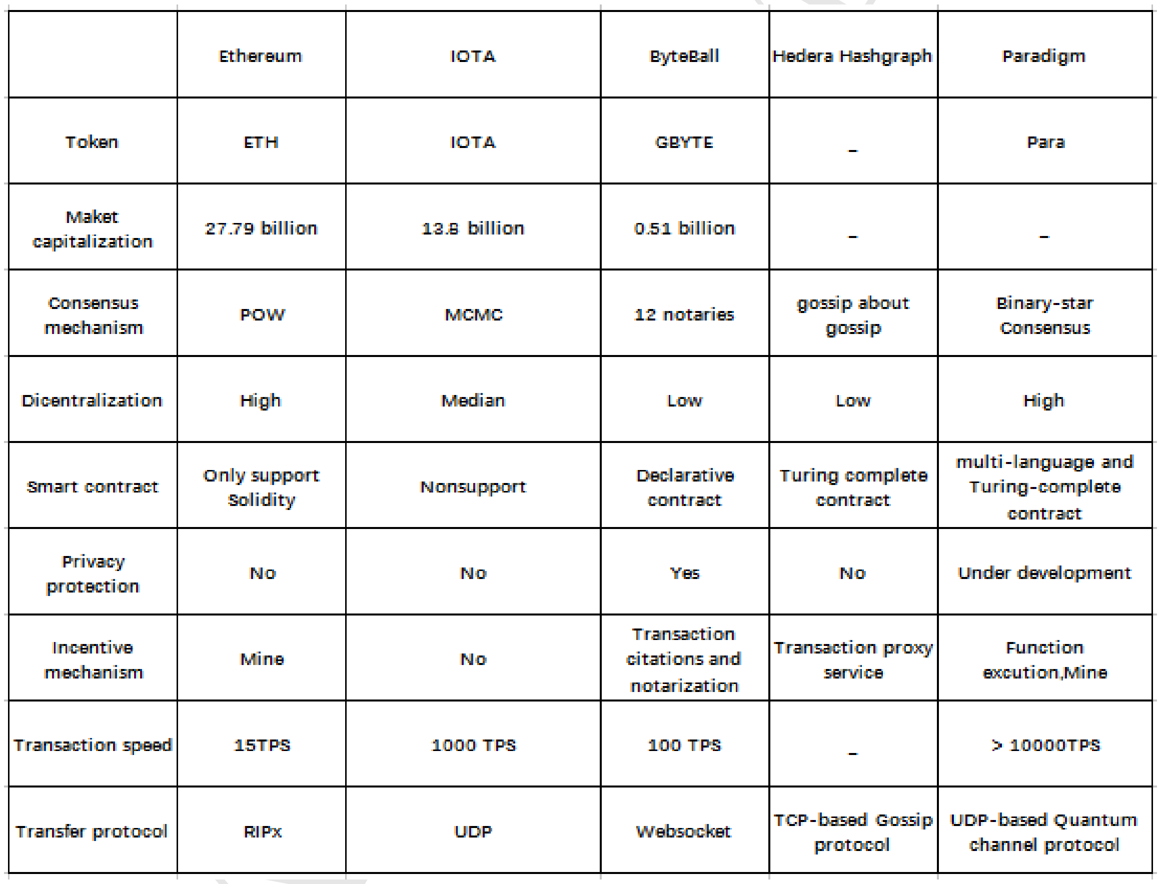
## 7. Comparison

With an eye to the future development of blockchain and cloud computing, Paradigm has proposed an epoch-making future DAPP development architecture by referring and absorbing to the advantages of current blockchain technology. The combination of FAAS and DAG is a fine solution of these following two problems:

1, How to maintain the scalability of DAG while meeting the requirement of decentralization?

2, How can DAPP get more users?2, How to realize the effective allocation of resources while in demand, and leave whenever the task is done?

For the first problem, our Binary-star consensus enables the flexbility of network capacity, solves the problem of centralization existing in traditional DAG solutions, as well as the contradiction between blockchain centralization and performance. And for the second one, by adopting our on-chain FAAS architecture, users do not need to pay for using DAPP, while the deployer calculates will be charged by a specific formula according to the number of calls, duration and other factors. The deployment costs are lower and less than that of a single DAPP call in ethereum by at least an exponential order. Such an architecture can realize not only the point-to-point value transmission under different scenario, but also achieve the efficient allocation of resources while aiso being developer-friendly .



| | Ethereum | IOTA | ByteBall | Hedera Hashgraph | Paradigm |
|---|---|---|---|---|---|
| Token | ETH | IOTA | GBYTE | _ | Para |
| Maket capitalization | 27.79 billion | 13.8 billion | 0.51 billion | _ | _ |
| Consensus mechanism | POW | MCMC | 12 notaries | gossip about gossip | Binary-star Consensus |
| Dicentralization | High | Median | Low | Low | High |
| Smart contract | Only support Solidity | Nonsupport | Declarative contract | Turing complete contract | multi-language and Turing-complete contract |
| Privacy protection | No | No | Yes | No | Under development |
| Incentive mechanism | Mine | No | Transaction citations and notarization | Transaction proxy service | Function excution,Mine |
| Transaction speed | 15TPS | 1000 TPS | 100 TPS | _ | > 10000TPS |
| Transfer protocol | RIPx | UDP | Websocket | TCP-based Gossip protocol | UDP-based Quantum channel protocol |

Figure 13: The comparison with other projects

**Acknowledgment**

This whitepaper is supported by the PARADIGM Development Centre, PARADIGM Research Centre and PARADIGM Foundation.

**References**

**NS08** Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System, 2008. https://bitcoin.org/bitcoin.pdf.

**Wod13** Gavin Wood. Ethereum: A Secure Decentralised Generalised Transaction Ledger, 2013. http://gavwood.com/Paper.pdf.

**But14** V. Buterin et al. A next-generation smart contract and decentralized application platform, 2014. https://github.com/ethereum/wiki/wiki/White-Paper.

**HB16** E Heilman, F Baldimtsi. Blindly Signed Contracts: Anonymous On-Blockchain and Off-Blockchain Bitcoin Transactions[M]. Financial Cryptography and Data Security. 2016.

**MGG13** I Miers, C Garman, M Green, et al. Zerocoin: Anonymous distributed e-cash from bitcoin[C]//Security and Privacy (SP), 2013 IEEE Symposium on. IEEE, 2013: 397-411.

**BCG14** E B Sasson, A Chiesa, C Garman, et al. Zerocash: Decentralized anonymous payments from bitcoin[C]. 2014 IEEE Symposium on Security and Privacy. IEEE, 2014: 459-474.

**BNM14** J Bonneau, A Narayanan, A Miller, et al. Mixcoin: Anonymity for Bitcoin with accountable mixes[C]//International Conference on Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2014: 486-504.

**Ben14** J. Benet. IPFS–content addressed, versioned, p2p file system, 2014. https://github.com/IPFS/IPFS/blob/master/papers/IPFS-cap2pfs/IPFS-p2p-file-system.pdf.

**BM07** I. Baumgart, S. Mies. S/kademlia: A practicable approach towards secure key-based routing, 2007. http://www.tm.uka.de/doc/SKademlia 2007.pdf.

**Max13** Gregory Maxwell, Proof of Storage to make distributed resource consumption costly. 2013. https://bitcointalk.org/index.php?topic=310323.0

**Bac13** A. Back. Bitcointalk. 2013. bitcoins with homomorphic value. https://bitcointalk.org/index.php?topic=305791.0. 2013

**Max15** G. Maxwell. Condential transactions. https://people.xiph.org/greg/condential-values.txt.2015

**Noe15** S. Noether. Ring signature condential transcation for monero. http://eprint.iacr.org/2015/1098. 2015

**SAL17** Sun SF., Au M.H., Liu J.K., Yuen T.H. RingCT 2.0: A Compact Accumulator-Based (Linkable Ring Signature) Protocol for Blockchain Cryptocurrency Monero. ESORICS 2017. Lecture Notes in Computer Science, vol 10493. Springer, Cham.

**LSZ15** Lewenberg, Y., Somplinsky, Y., Zohar, A. Inclusive block chain protocols. In International Conference on Financial Cryptography and Data Security 2015, Springer, pp. 528547.

**SLZ16** Sompolinsky, Y., Lewenberg, Y., Zohar, A. Spectre: Serialization of proof-of-work events: confirming transactions via recursive elections, 2016.

**SZ18** Sompolinsky, Y., Zohar, A. Phantom, a scalable blockdag protocol. https://eprint.iacr.org/2018/104.pdf.

**LLZ18** Chenxing Li, Peilun Li, Dong Zhou, Wei Xu, Fan Long, Andrew Chi-Chih Yao. Scaling Nakamoto Consensus to Thousands of Transactions per Second. arXiv:1805.03870v3 [cs.DC] 12 Aug 2018.

**GBYTE17** Byteball network. https://www.byteball.org

**IOTA17** IOTA network. https://www.iota.org

**NANO17** Nano network. https://www.nano.org

**DAGLab** DAG Labs. https://www.daglabs.com. Israel

**LSP82** Lamport L, Shostak R, Pease M. The Byzantine generals problem, 1982. ACM Trans. on Programming Languages and Systems,,4(3):382401.

**SG10** Schroeder B, Gibson GA. A large-scale study of failures in high-performance computing system, 2010. IEEE Trans. on Dependable and Secure Computing. 337350.

**CML06** Cowling J, Myers D, Liskov B, Rodrigues R, Shrira L. HQ replication: A hybrid quorum protocol for Byzantine fault tolerance, 2006. In:Proc. of the 7th Symp. on Operating Systems Design and Implementation. Berkeley: USENIX Association. 177190.

**SND10** Serafini M, Nokor P, Dobre D, Majuntke M, Suri M. Scrooge: Reducing the costs of fast Byzantine replication in presence of unresponsive replicas, (2010). In: Proc. of the 2010 IEEE/IFIP Intl Conf. on Dependable Systems and Networks. 353362.

**Mig00** Miguel Oom Temudo de Castro. Practical Byzantine fault tolerance, 2000.Massachusetts Institute of Technology, Cambridge, MA, USA.

**MA06** Matin JP, Alvisi L. Fast Byzantine consensus, 2006. IEEE Trans. on Dependable and Secure Computing, 3(3):202215.

**HGR07** Hendricks J, Ganger GR, Reiter MK. Low-Overhead Byzantine fault-tolerant storage, 2007 In: Proc. of the 21st ACM SIGOPS Symp. On Operating Systems Principles. New York: ACM Press. 7386.

**LAS10** Lima M, Greve F, Arantes L, Sens P. Byzantine failure detection for dynamic distributed systems, 2010. SANTOSDELIMA-2010-584597, RR-7222. Paris.

**SFK09** Singh A, Fonseca P, Kuznetsov P, Rodrigues R, Maniatis P. Zeno: Eventually consistent Byzantine-fault tolerance, 2009. In: Proc. of the 6th USENIX Symp. on Networked Systems Design and Implementation. Berkeley: USENIX Association. 169184

**Pla96** J. S. Plank. A tutorial on reed-solomon coding for fault-tolerance in raid-like systems, 1996. http://web.eecs.utk.edu/ plank/plank/papers/CS-96-332.pdf.

**Mao03** W. B. Mao. Modem cryptography: theory and practice. Prentice Hall Professional Technical Reference, 2003.

**BF01** Boneh D, Franklin M K. Efficient generation of shared RSA keys, 2001. Journal of the ACM, 48(4): 702-722.

**Sha79** Shamir A. How to share a secret, 1979. Communications of the ACM, 22(11): 612-613.

**DM08** Dehkordi M H, Mashhadi S. Verifiable secret sharing schemes based on non-homogeneous linear recursions and elliptic curves, 2008. Computer Communications, 31(9):1777-1784.

**BG09** D. Boneh, C. Gentry. A fully homomorphic encryption scheme, 2009. Stanford University.

**VGH10** Van Dijk M, Gentry C, Halevi S, et al. Fully homomorphic encryption over the integers, 2010. Theory and application of cryptographic techniques, 24-43.

**Pai99** Paillier P. Public-key cryptosystems based on composite degree residuosity classes, 1999. Theory and application of cryptographic techniques, 223-238.

**CCS08** Camenisch J, Chaabouni R, Shelat A, et al. Efficient Protocols for Set Membership and Range Proofs, 2008. International conference on the theory and application of cryptology and information security, 234-252

**FO97** Fujisaki E, Okamoto T. Statistical Zero Knowledge Protocols to Prove Modular Polynomial Relations, 1997. International cryptology conference, 16-30.

**FS87** Fiat A, Shamir A. How to prove yourself: practical solutions to identification and signature problems, 1987. International cryptology conference, 186-194.

**DF92** Desmedt Y, Frankel Y. Shared generation of authenticators and signatures, 1992 International cryptology conference, 457-469.

**JO08** Jarecki S, Olsen J. Proactive RSA with Non-interactive Signing, 2008. Financial cryptography, 215-230.

**FFS88** Feige U, Fiat A, Shamir A, et al. Zero-knowledge proofs of identity, 1988. Journal of Cryptology, 1988, 1(2): 77-94.

**CM99** Camenisch J, Michels M. Proving in zero-knowledge that a number is the product of two safe primes, 1999. Theory and application of cryptographic techniques, 107-122.

**KLL97** 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, 1997. In: Proc. of the 29th Annual ACM Symp. on Theory of Computing STOC97. El Paso: ACM Press. 654663.

**HM03** Honicky RJ, Miller EL. A fast algorithm for online placement and reorganization of replicated data, 2003. In: Dongarra J, ed. Proc. of the 17th Intl Parallel Dtributed Processing Symp. IPDPS 2003. Nice: IEEE Computer Society.

**HM04** Honicky RJ, Miller EL. Replication under scalable hashing: A family of algorithms for scalable decentralized data distribution, 2004. In: Bader DA, ed. Proc. of the 18th Intl Parallel Distributed Processing Symp. IPDPS 2004. Santa Fe: IEEE Computer Society.

**WBM06** Weil SA, Brandt SA, Miller EL, Long DDE, Maltzahn C. Ceph: A scalable, high-performance distributed file system, 2006. In: Bershad B, ed. Proc. of the 7th Symp.on Operating Systems Design and Implementation. Seattle: USENIX. 307320.

**BEH07** Brinkmann A, Effert S, Heide FMAD, Scheideler C. Dynamic and redundant data placement, 2007. In: Abdelrahman TS, ed. Proc. of the 27th Intl Conf. on Distributed Computing Systems. Toronto: IEEE Computer Society.

**SMK01** Stoica I, Morris R, Karger D, Kaashoek F, Balakrishnan H. Chord: A scalable peer-to-peer lookup service for internet applications, 2001. In: Cruz R, ed. Proc. of the Annual Conf. of the Special Interest Group on Data Communication, SIGCOMM 2001. San Diego: ACM Press. 149160.

**BSS00** Brinkmann A, Salzwedel K, Scheideler C. Efficient, distributed data placement strategies for storage area networks, 2000. In: Gary M, ed. Proc. of the 12th ACM Symp.on Parallel Algorithms and Architectures. Bar Harbor: ACM Press. 119128