CYBERLAND: AN ARCHITECTURE DESIGN FOR DISSIPATIVE STRUCTURE ECONOMY TOKEN SYSTEM USING ABC BLOCKCHAIN

Hannah Lan, David Fong, Steve Lau, Wyman Lee, Robby Kwok

December 20, 2022, v1.0

Abstract

In this paper, we come up with a new solution called ABC blockchain based system which is tailor-made for Web3 applications, the ABC system features a decentralized infrastructure and application platform with improved self-management properties and secutity. The design of ABC blockchain is to increase the transaction throughput by adopting various optimizations in storage layers and network transport, and to enhance smart contracts with AI algorithm support. In this whitepaper, we try to introduce all major technologies adopted in our system, including distributed storage, blockchain, P2P network, service application framework, and data encryption.

To properly provide a cohesive, concise, yet comprehensive introduction to the CyberLand ABC system, we mainly focus on describing the unique definitions and features that guide the system implementation.

INTRODUCTION

Blockchain technology is the base component in the ABC technology stack. Blockchain technology combines peer-to-peer network computing and cryptography to create an immutable decentralized public ledger, which is considered as the fifth innovative computing paradigm. It has brought dramatic changes to the way of storing and exchanging data, allowing the Internet enter the era of trust economy. Blockchain technology is promising to make the world more secure and autonomous. However, the existing blockchain-based applications are still in their early stage of development, lacking a favorable environment for technical advancement. There are few blockchain applications that are mature or attractive to the public, i.e., most existing application scenarios have not significantly benefited from this technology. This indicates that there are many aspects need be further optimized for both blockchain technology and applications, these foundations will pave the wave for the mass adoption of Web3 era.

The current blockchain projects on the market mainly have the following limitations and disadvantages:

1. Low transaction processing throughput, low data storage capacity, no parallelism support: In Bitcoin, for example, due to the constraints on block size and block time, only 7 transactions can be processed per second and achieving a high confidence that a transaction has been confirmed requires about an hour long wait until the transaction is 6 blocks deep into the Bitcoin blockchain. Such a low transaction rate is far from satisfying the application demand. Moreover, the long transaction latency will reduce the willingness of new users to join the network, and hence lower the competitiveness of blockchain-based products against those from traditional industries. Although there are some research advancements,

such as the lightning-network, we still lack a comprehensive approach to solving the efficiency issue.

2. "User unfriendly", hard to use: The blockchain is indeed a synthesis of many existing technologies, which results in high learning cost and implementation difficulty. The current blockchain applications are mostly designed for those technical experts who know how to use them, but not for mainstream consumers. In particular, almost all blockchain applications require users to install and run blockchain full nodes or "lightweight nodes". Its long learning curve and massive efforts lower the willingness of participation for ordinary users. For example, the game Axie might be one of the most user-friendly distributed application on Ethereum, but users still have to install the Metamask Wallet browser extension by themselves. Besides, users also need to know how to securely purchase cryptocurrencies from others and use them in Metamask. To attract more users, blockchain applications should be designed as simple as the modern web or mobile applications. In addition, the blockchain technology also needs to lower the learning and usage costs, support rapid deployment, and provide close-to-business interfaces.
3. Poor extensibility, incomplete functional support: Most of current blockchain applications or services provide only limited functionalities, and are short of features that encourage code contribution from open source communities. The flexibility and extensibility of these applications need to be improved. We hope that, by providing a solid technical infrastructure and a complete set of corresponding API/SDK components, the third-party application developers can easily develop and extend their own applications. Hence, we allow developers to build customized blockchain applications based on available sub-components in the ABC ecosystem.

This project aims to address the problems and challenges discussed above arising from the development of blockchain technology and applications, including: the support of efficient data storage and concurrent transaction

processing; the improvement of application usability, functional completeness, robustness and scalability. The goal of the ABC system is to build an open, secure and autonomous application service platform based on the trustful relationships among all participates. By integrating advanced technologies in areas of blockchain, artificial intelligence (AI), big data, and Internet of thing (IoT), the system is able to provide secure and flexible communication channels for all participates and thus establish a security system based on trust economy. The ABC system can be viewed as a distributed super-computer or a value conversion platform, on which anyone can communicate and trade with others freely. Besides, the ABC system can provide native identity authentication and DNS services, which are of great significance for building the next generation Internet infrastructure. Overall, the vision of the ABC system is to serve all the rational Web3 applications that have currently been proposed and conceived.

This paper is for informational purposes only. All contained technical details are not finalized, and may change during the future development and testing phases.

CONTENTS

1 ABC Blockchain Network – ABN

In this chapter, we will introduce one of the core designs in the ABC system – ABC Blockchain (Slim Dragonfly) Network – ABN. In order to achieve a flexible, extensible and decentralized P2P solution, we propose a "Double Ring" topology called the ABC Blockchain Native (Slim Dragonfly) Network — ABN, which consists of a Manager Ring, a Worker Ring, and an Origin (starting point). **In Section 1.1** we will present the system topology of ABN. Then we will are going to introduce Worker Ring in **Section 1.2**.

1.1 System Topology - ABN

In this section, we introduce the network topology design in our ABC system – "Double Ring". The main design principle is to separate the data management logic from the application business logic layer, such that it is easy to achieve a flexible and extensible decentralized P2P solution. Moreover, our system adopts the "multi-layer DHT ring" technology for distributed data lookup.

Figure 1.1 shows the overall structure of the "Double Ring" topology, which Consists of two rings. We call this topology the ABC Blockchain Network — ABN. The Ring 0 (inner ring) is the *Manager Ring* composed of management server nodes, while the Ring 1 (outer ring) is the *Worker Ring* composed of worker nodes. In the ABN Ring topology, each node maintains the information of its sibling nodes and the corresponding head node in each group. With an optimized routing strategy, ABN provides fast data lookup and delivery within 2-3 hops, which effectively improves the efficiency of message broadcasting in the network. ABN is an improved Slim Fly topology [3] based on the Hoffman-Singleton Graph [2].

Figure 1.1 also demonstrates the data flow of the Ring topology and how the Manager Ring manages the groups of the Worker Ring. In the following, we discuss the design and implementation of each sub-components in detail.

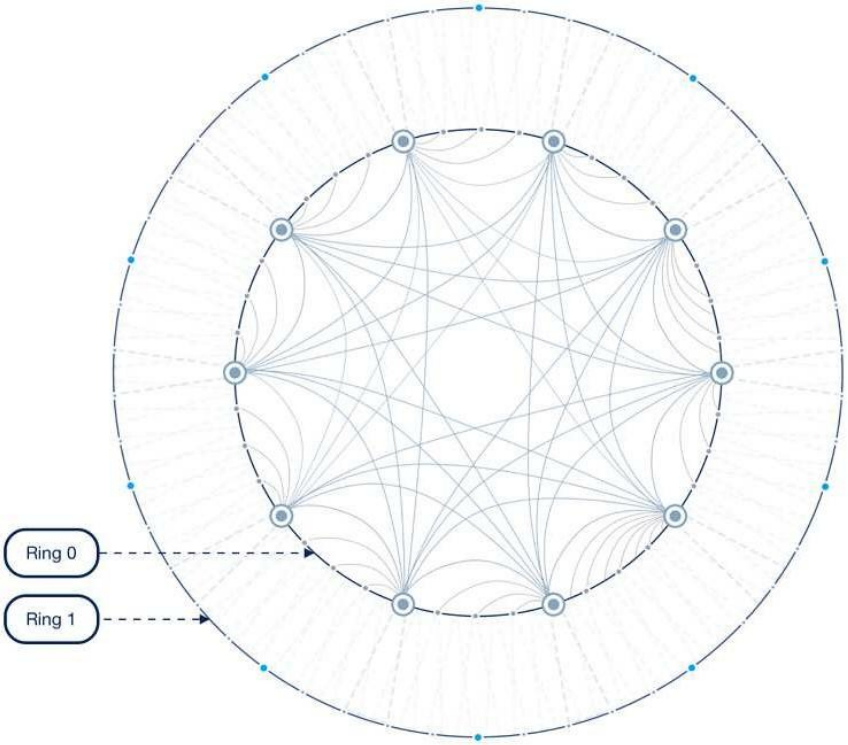

Figure 1.1 : ABC Blockchain Network — ABN

Figure 1.1 also demonstrates the data flow of the Ring topology and how the Manager Ring manages the groups of the Worker Ring. In the following, we discuss the design and implementation of each sub-components in detail.

1.1.1 Manager Ring – Ring 0

As shown in Figure 1.2, each server node in the Manager Ring is a full node, which stores the complete information of the system (including blockchain data, Worker Ring shard information, terminal node information, user account list information, etc.). In addition, the Manager Ring also embeds distributed databases with fast query support. A query request from a client will be submitted to the Manager Ring first, and the client has to pay a certain amount of tokens (e.g. 0.01 CC, where CC is short for CyberCoin). Here we propose the concept of "Unity": each Manager Ring node (i.e., a Unity) actually consists of N server nodes ($N \leq 6$)

stacked vertically, which are named from server 0 to 6. The server 0 is responsible for message management, i.e., send management commands (CMD) and check message integrity (Error Check). The servers 1 and 2 are responsible for sending messages within the ring, while the servers 3 and 4 are responsible for computation and transmission of request responses. The server 5 is reserved and will take over others' work when they are down. Overall, the entire Unity structure consists of 6 servers, each of which guarantees strong data consistency with others. The **Figure 1.2** shows the message flow direction inside each layer: counterclockwise for server 0-2, and clockwise for server 3-4.

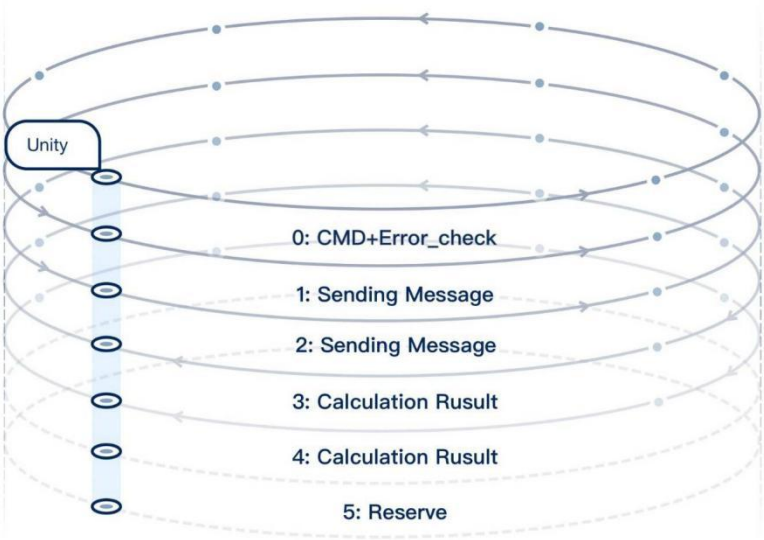

Figure 1.2: Manager Ring Structure

As shown in **Figure 1.3**, multiple Unities logically form a ring structure, which is divided into multiple Groups. Assume that there are N Unities on the ring, which is then divided into $\sqrt{N}$ Groups with $\sqrt{N}$ Unities each. For instance, the outer Worker Ring consists of 256 "Unities", named as S_1, S_2 ... S_{256}. It will be divided into 16 Groups, namely *GROU* P_1, *GROU* P_2,..., *GROU* P_{16}, each with 16 Unities.

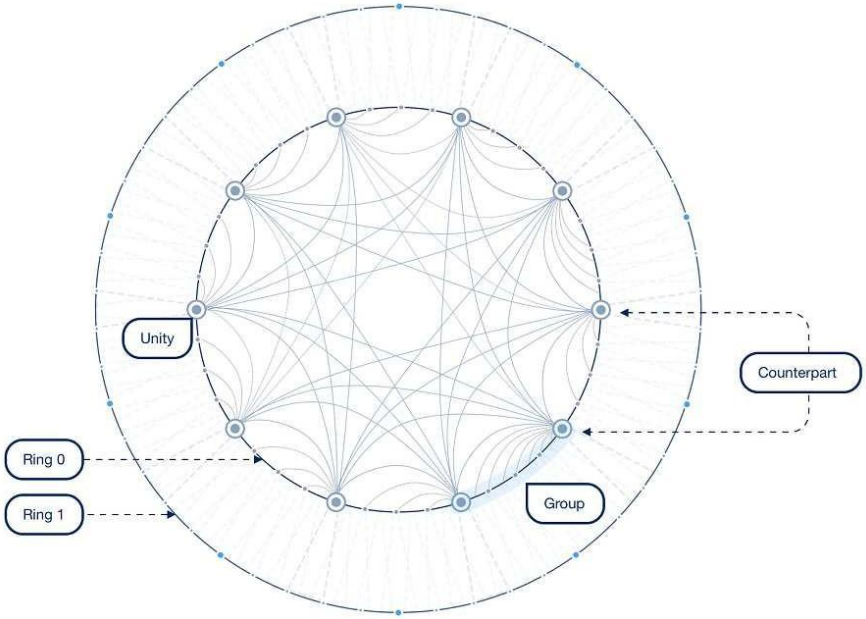

Figure 1.3: Unity, Group, and Counterpart with Annotation

Under this partition scheme, the header node of each group (or shard) is called a "Counterpart" of other headers. Note that the roles of all servers/nodes on the ring are identical, i.e., any Unity can be a header node. We propose a *Dynamic Sharding Paradigm*, which means that each Unity treats itself as a group header and its sibling node as the succeeding node. Besides, the corresponding group headers are treated as Counterparts. As a result, the partition scheme viewed from Unities are different and may be dynamically changed.

Whenever a terminal (or client) issues a service request, it needs to first submit a transaction request to the Manager Ring, and then the Manger Ring assigns this request to the corresponding Worker server with respect to the requesting terminal. Such a service can support, but not limited to, IM (Instant Messaging) or storage services. During the process of a request, the terminal can apply for a server replacement to ensure the service qualify, by issuing an appeal request. Upon the completion of the request, the Manager Ring distributes the tokens spent by the client to each participating server, according to the service time and quality. To protect the system security, the IP address of a Manager Ring Server is not publicly broadcasted. Instead, only some public email

addresses are available, and the Manager Ring server sends messages to outer Worker Ring server in a loosely coupled manner, i.e., by sending emails.

At the same time, a server on the Manager Ring correspondingly manages one or more Groups on the Worker Ring, and maintains all index information of these Groups. As shown in **Figure 1.4**, the Manager Ring shards and manages the Groups on the Worker Ring. Each Group synchronizes its group information at a fixed frequency (to the Manager Ring).

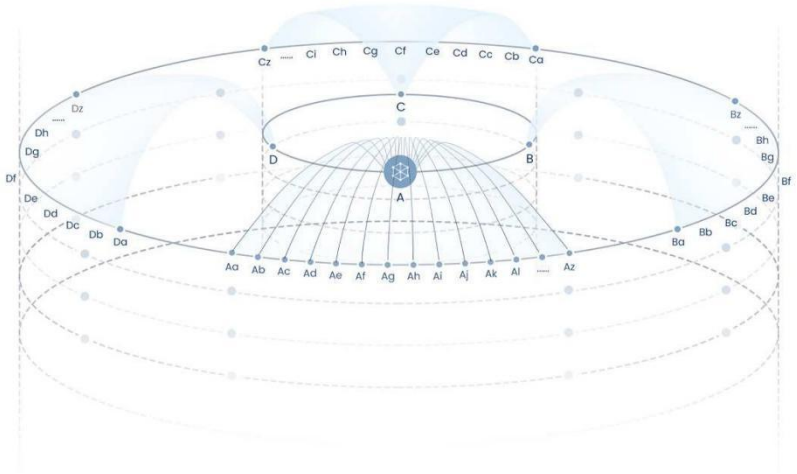

Figure 1.4: Group Management in Manager Ring

In principle, the Manager Ring servers only operate the management logics, but are not involved in specific application services. In this manner, the scalability of the ABC system can be significantly improved. We encourage developers to develop more distributed applications on top of ABC systems, such as instant messaging services, cloud storage services, App Engine services, and CDN services. The servers on the Manager Ring are responsible for executing blockchain bytecode, and store transaction data in the system. The transactions are generated by the Worker Ring or terminals, and then submitted to the Manager Ring. The Manger Ring servers commit transaction data into the ABC blockchain in the form of mining.

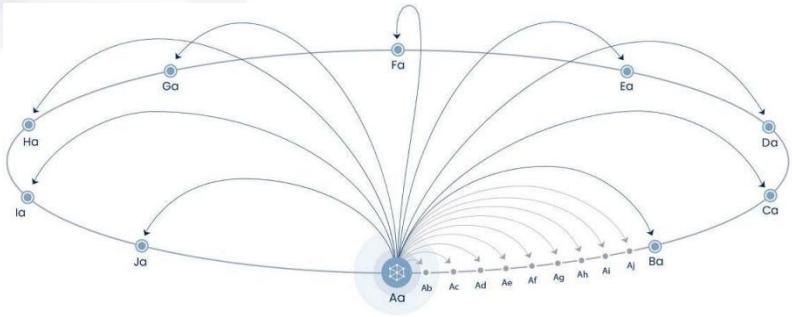

Figure 1.5: Data Flow on the Ring

Figure 1.5 shows the data flow on the Ring. Whenever a node needs to broadcast a message (e.g., block data), it sends the message to all neighboring nodes in the same group, as well as the header nodes (i.e., counterpart) in other groups. The message transmission pattern is actually identical to a Hamming graph (**Figure 1.6**) if we view the **Figure 1.5** from another dimension. All nodes in a same Group are fully connected, while a node and its corresponding counterparts are also fully connected.

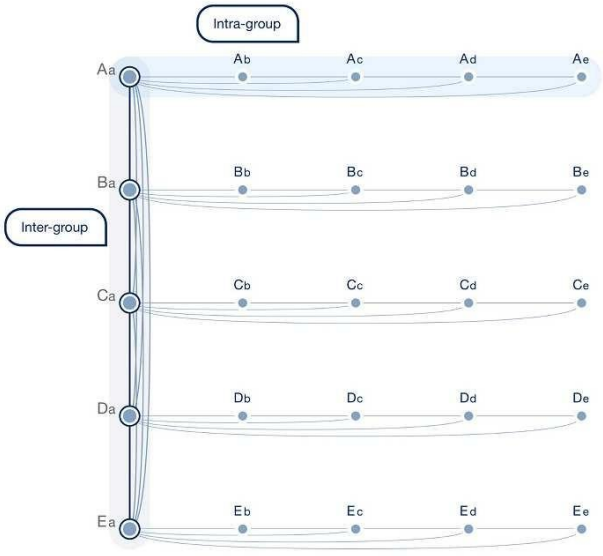

Figure 1.6: Hamming Graph

1.1.2 Worker Ring – Ring 1

The outer ring in the system is the "Worker Ring" composed of worker servers. We use the "DHT Ring" approach to manage the sharding and naming for the Worker Ring servers (the DHT in this context is just a rule for naming).

Similar to the Manager Ring Unity, the Unity of the Worker Ring is also a stereoscopic concept, which means that each Unity contains multiple servers stacked vertically (e.g., S_1 consists of S_{1a}, S_{1b} ,..., S_{1e}). Most data storages commonly apply 3-way data replication. Since our system only maintains the naming information requiring small storage space, we are able to use 5-way replication instead. All these replicated servers maintain the identical set of data, which are used as the backups to improve the system robustness. If one of the servers (e.g., S_{1a}) crashes or goes offline, the system can still operate normally, as other backup servers are able to provide the same services. To improve the system security, such as prevent external malicious traversal search, interception or spam delivery to the ABC system, the server on the Worker Ring does not keep any information related to the entire network. In our DHT algorithm, each Unity maintains the information about all other servers in the same Group (e.g., S_1 maintains the information of GROUP1 servers S_1, S_2,..., S_{16}), as well as the information about the header servers (i.e., counterparts) in other Groups (e.g., S_{17}, S_{33}, S_{49}, ...). In this manner, we guarantee that any query in the system can be handled within "2 to 3 hops", hence significantly improving system efficiency and reducing query overhead. The account information of the end user is maintained in the server that matches the first two characters of the hash derived from the account name .

Worker Ring servers earn tokens from providing application services to users, and the transactions are recorded on the blockchain of the Manager Ring. In addition, the Worker Ring servers are required to keep the log of the provided services, which might be audited by the Manager Ring periodically.

1.1.3 AI Self-Management Research and Strategy Scheduling

We deploy an automatic service scoring system on each client node, so that the client is able to evaluate the services it received. The Manager Ring rates the Worker Ring and the provided services, by recording the service scores of the outer Ring and periodically reading the service logs. With the adoption of AI reinforcement learning algorithms (including service node upgrade/downgrade, and suspicious behavior tracking, etc.), the ABC system can support intelligent self-management and eventually achieve system autonomy. The designs of related AI algorithms will be introduced in Chapter 6 in detail. Besides, the system also provides a management mechanism based on user threat assessment: The Manager Ring identifies potential malicious behaviors (e.g., sniffing nodes of firewalls) via user threat assessment (UTA) provided by the AI component. If a potential threat is detected in the network, the Manager Ring will kick that user out of the network, in order to protect the system security.

Recall that the Worker Ring is composed of worker servers, and the number of Groups can be dynamically adjusted according to the capacity of the actual services. In each Group, we can add a number of index servers, each of which acts as a backup for others and maintains the available server information in that Group. The Manager Ring maintains all index server addresses for each Group. When a client issues a service request, it first sends a transaction request to the Manager Ring, which returns the address of an index server. After that, the client connects to that server for obtaining the address of the corresponding worker server. Finally, the worker server can provide the user specific services, such as message forwarding or storage. The worker server internally provides group-based encrypted data backup mechanism that guarantees the reliability and availability of the stored data. When a sniffing client threatens the network, it has to first obtain the index server address before further affecting the index or worker server (e.g., blocking IPs). Once the AI confirms the detection from the UTA, the malicious user is immediately kicked out from the network, which ensures the network security. Meanwhile, the supplemental index servers will be re-elected within the affected Group, and the data on the lost worker servers will be replicated on new backup servers.

1.1.4 The Bootstrap

The Bootstrap server is a backup server for disaster recovery. When the Manager Ring fails to operate or is out of control due to internal errors, the origin server will launch the recovery process for taking over and reorganizing the entire system. This is to ensure that the system is able to work normally even in the occurrences of accidental failures or catastrophic errors.

1.1.5 Temporary Worker Server and Small Hardware Device

Apart from the "Double Ring" and the Worker Ring 1, our system also allows "temporary workers" to join the network. Temporary workers can be mounted in any Group on the Worker Ring to provide additional computation power, bandwidth and storage service, such that they can absorb workloads from the overloaded worker servers. A temporary worker is able to apply for promotion to join the Worker Ring as a normal worker node, after it serves a certain amount of service requests. In addition, we will release a "temporary worker" hardware, and users can also earn tokens by running this hardware and providing services.

The node expansion strategy in ABC is that the newly promoted temporary worker is assigned to the corresponding Group as a vertically parallel server. Once the number of servers in a group reaches the threshold, this Group splits into two Groups, then broadcasts this Ring expansion event to each Worker Ring Group and reports to the Manager Ring. When the Manager Ring wants to remove an offending or frequently failed worker server from the network, the temporary worker has the priority to take over the role of the removed server.

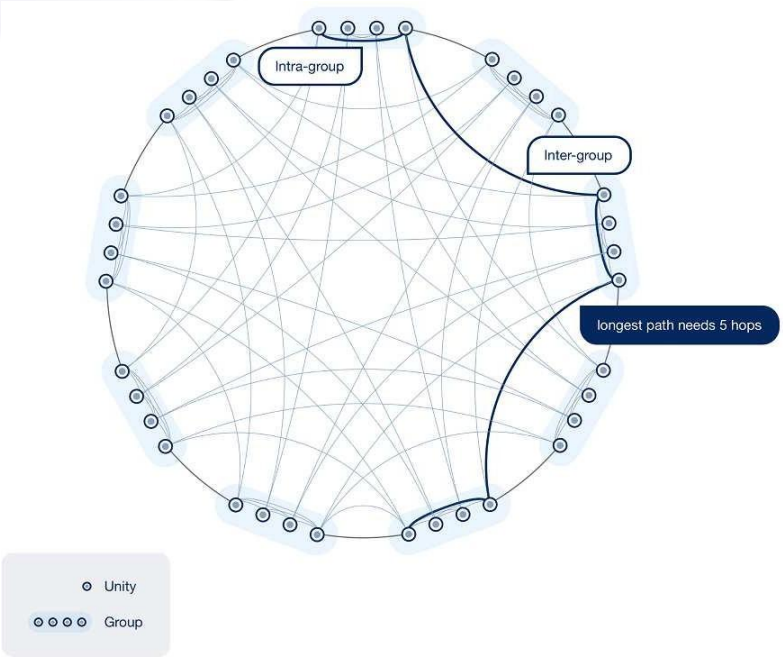

Figure 1.7: Dragonfly Networks

1.1.6 Comparison of ABN Network Topology with Dragonfly Network

We compare the ABN network topology with the Dragonfly Network Topology. It uses an optimized topology for the blockchain application scenarios. **Figure 1.7** illustrates the Dragonfly Topology, in which the Dragonfly router randomly selects an intermediate group to transmit a datagram each time, such that the network links can be evenly utilized. Therefore, it takes up to 5 hops to complete a data transmission. With an optimized routing strategy, ABN provides fast data retrieval and delivery within 2-3 hops, which significantly improves the efficiency of message broadcast in the network. Besides, ABN adjusts the tradeoff between transmission efficiency and routing storage: in the ring topology of ABN, each node maintains the information of its sibling nodes in the same Group and of the corresponding counterpart nodes in other groups. In this manner, the system increases the storage consumption slightly in exchange for the improvement of

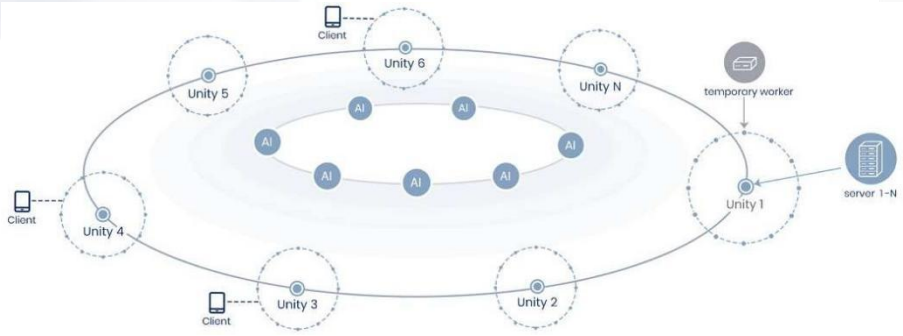

Figure 1.8: The Structure of Worker Ring (Ring 1) Sub-System

transmission efficiency. We believe that ABN is by far the most suitable technological application scenarios for the Dragonfly topology.

1.2 Worker Ring DHT Network

The Worker Ring (Ring 1) is the actual provider of various resources in the system. These servers provide services to clients who use the network, and charge a certain amount of service fee. As a decentralized network, the Worker Ring network cannot rely on a group of permanent central servers as in traditional C/S architectures, such as centralized e-commerce and IM networks. Instead, in the ABC system, even though a client has to connect to the Worker Ring server similar to the client-server manner, it actually can connect to an alternative worker server to be served. In this decentralized Worker Ring, there are several major issues that need to be addressed, as follows:

- How to deploy servers on the Worker Ring for network construction?
- How to efficiently lookup the required resources on the Ring?
- How does an application use these servers on the Worker Ring to provide its service?

1.2.1 Worker Ring Service Sub-System

The Worker Ring Service Sub-system is the service provider in the network. It

resides in the outer Ring (i.e., Worker Ring), which is composed of N adjacent Unities connected end to end as a ring. The Unity of Worker Ring is a stereoscopic concept, which consists of multiple servers and internally provides high availability and load balancing. There are a number of temporary worker servers around the Worker Ring, which are assigned by the Manager Ring and can help serve a part of the services/businesses. The Worker Ring Service Sub-system provides the concept of "permanent" virtual Unity, which can "permanently" exist and work on the Ring. The Worker Ring adopts a fully distributed structural topology design. Each Unity maintains only part of the Worker Ring network resources and provides a DHT-based resource lookup functionality.

A resource on the Worker Ring is just an abstract concept, which could be a single server, a stored file, or anything else. These resources are uniformly mapped to abstract addresses on the Worker Ring, and stored in the DHT routing table on the Unity server. Any server on the Worker Ring can locate the corresponding resource in the network by querying the table.

The service quality of the Worker Ring is monitored by the Manager Ring. The AI component in the Manager Ring will monitor the service status of each Worker Ring server, in order to change the role of these servers when necessary. A server can be either a temporary worker on the Worker Ring or a server in a Unity.

The Worker Ring servers provide bandwidth, computation and storage resources to the clients. When a client requests for a service, it issues a request to the Manager Ring, which then returns an available server according to the client request. Once the request is completed, the Manager Ring sends the ABC coins paid by the client to the account of the participating server and commits a transaction.

1.2.2 Abstract Address Network on Worker Ring

The fundament for establishing the Worker Ring is the abstract datagram in the network layer. It allows each Unity or even each server in these Unities to represent its exact "network identity" as a 256-bit "abstract network datagram",

and communicate with this 256-bit network address to identify the sender and receiver. In particular, we do not need to be aware of the underlying protocols, such as IPv4 or IPv6 addresses, UDP port signals, etc., which are completely hidden behind the abstract network layer.

In the ABC Worker Ring, there exists an overlay network address encapsulation (i.e., Abstract Resource Network Layer), which can (unreliably) send datagram from one abstract address to another. In principle, the Abstract Resource Network Layer (ARNL) can be implemented atop different network technologies. Therefore, we implements it over UDP protocol on the IPv4/IPv6 network. We can also choose TCP protocol, when UDP is not an option.

Option 1: ARNL is an unreliable (small) datagram protocol based on a 256-bit resource address abstraction, and it can be used as the base for more complex network protocols. For example, we can use ARNL as an alternative to the IP abstraction and build a TCP-streaming-like protocol.

Option 2: A reliable variable-sized large datagram protocol (called RLDP) can be built on ARNL to replace the TCP-like protocols. For example, we can use this reliable protocol to send RPC requests to remote hosts and receive responses.

1.2.3 Worker Ring Unity and Group

The Worker Ring network consists of a large number of servers, which are all acting as resource providers and data forwarders in the network. They are allocated in an address space, and mapped to different Unities on the Worker Ring. Multiple Unities logically form a ring structure.

Assume that there are N Unities on the Worker Ring; then each Unity should have following properties:

- Each Unity consists of 6 to 32 servers;
- Each Unity on the ring is of the same importance;
- All Unities on the ring are divided into $\sqrt{N}$ partitions, each of which is called a Group and contains $\sqrt{N}$ Unities;
- Each Unity maintains the index server information of $\sqrt{N}$ neighboring

Unities starting from itself;

- Each Unity maintains all Group server information that are at the distance of *N* Unities;
- Each Unity server acts as a Group server, an index server, and a resource server at the same time.

The Group server maintains the partition table, which can provide the index server address closest to the target resource; the index server maintains an intra-group lookup table, which maintains the addresses of resource servers; the resource server provides resources, such as computation power and storage space.

1.2.4 DHT Routing Table in Unity

Any Unity server in the Worker Ring DHT usually maintains a DHT routing table. The Worker Ring routing table consists of n (n = 2) buckets, numbered from 0 to 1. The first table (i.e., Intra-group Routing Table) contains information about known Unities, whose distances to the Unity address a are from 1 to $\sqrt{N}$ Unities. The second table (i.e., Inter-group Routing Table) contains information about counterparts, whose distances to the current Unity are of an integer multiple of the Group size $\sqrt{N}$. The Unity information includes the address, IP, UDP port and other useful information, such as the response latency of the last ping.

The Worker Ring Unity follows a stereoscopic design, in which multiple servers collaboratively work in the same Unity and hence can also conduct timely data synchronization (such as user online and offline events).

When the Manager Ring assigns a server to a user, multiple servers can be provided simultaneously, and the user selects the "optimal" server considering its actual scenario requirements. Note that all provided servers are from the same Unity on the Worker Ring.

1.2.5 Resource Indexing in Unity DHT – Key and Value

The resources on the Worker Ring DHT are all indexed as key-values. The keys

are all of 256-bit bytes, i.e., the SHA256 hash of the serialized resource object in most cases. The serialized object is also called the key description. In some cases, the abstract address of a Worker Ring Unity server is also used as the key on the Worker Ring DHT, since such an abstract address is also 256-bit hash from serialized objects. For example, if a Unity server does not want to publish its own IP address, others can locate it as long as they know the key of that server.

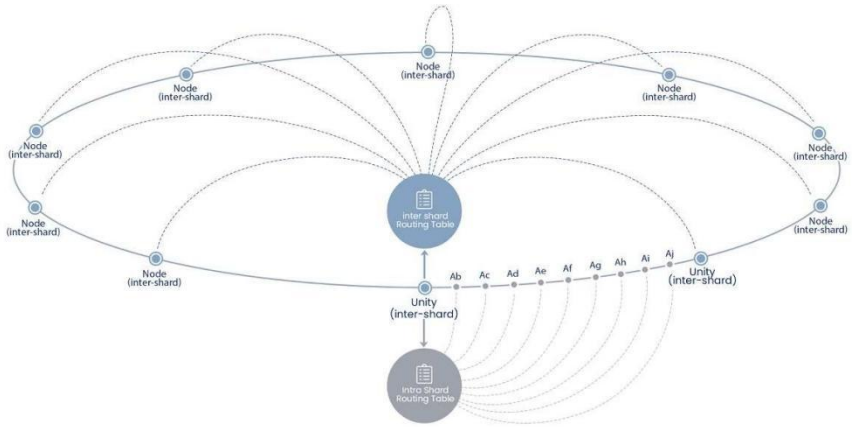

Figure 1.9: Unify Group and Inter-Group Connection

The values associated with these 256-bit keys are arbitrary byte strings with bounded lengths. The content of a byte string is determined by the corresponding application requirements. It can be either the account login information of an IM server, or the digital fingerprint of a stored file. The value of a Worker Ring network resource is usually acquired by locating and querying the Unity that maintains the target key.

The key-value mappings of the Worker Ring DHT are maintained on DHT Unity servers, and are essentially resource address mappings in the network. Therefore, any resource in the Worker Ring network (e.g., account login information in the IM service, and the stored file) has at least one address to ensure that it is an accessible resource. Such a DHT address should not be frequently changed, otherwise other Unities will not be able to locate the keys they are looking for. If a Unity server does not want to reveal its "real" identity, it

can generate an independent abstract address only for participating in the DHT lookups. However, the abstract address of the communication channel between Worker Ring servers have to be publically available, as it is associated with Unity server information, such as server id, IP address, and port.

1.2.6 DHT Network Lookup Algorithm with O(1) Complexity

The Worker Ring DHT adopts a more efficient lookup algorithm compared to classic DHT algorithms, such as Chord and Kad. In our algorithm, only two hops (i.e., O(1) complexity) are required to lookup the corresponding resource, which improves resource lookup efficiency on the Worker Ring.

The Worker Ring DHT resources mentioned above are indexed by their keys. A resource first generates a key, and hashes it to a Unity on the Worker Ring, which then maintains the resource.

When a query is issued to the network, the resource server where the client is located becomes a query proxy, and first looks up for local Group information. If it hits in the table, the server further locates the target resource server in one step. Otherwise, the server checks inter-group information and looks-up those resource servers near the target server Group. After that, we further look up for local Group information on that server, and search for the target resource server nearby. In this manner, the lookup can be completed within two hops. The object or resource stored on the target server is just abstract, which can be either the login information of a user (i.e., ABC IM user login), or a file.

Compared to Kad, the Worker Ring DHT lookup algorithm optimizes the search path in the lookup process, making it more efficient.

1.2.7 Dynamic High Availability and Load Balancing in Unity

In order to ensure the high availability of Unities on the Worker Ring, servers in the same Unity adopt a backup mechanism that supports hot backup composed of multiple physical servers. When a Unity server goes down or leaves, other servers can quickly switch their roles and take over the work belonging to the faulty Unity server.

When a Unity contains a relatively small number of servers, it takes over some Unity servers from neighboring Unity (considering its available resources) as backup servers. This is to ensure that each Unity has enough physical servers to support its workloads.

The servers inside the Worker Ring Unity has a load balancing mechanism, which helps achieve load balancing within the Unity by distributing the workloads properly based on the business capacity of each server.

1.2.8 Capacity Assumption for Worker Ring

In the Worker Ring network, we configure the minimum number of Unity to be 64. Consider that there are normally 6 servers in each stereoscopic Unity. The total number of concurrent connections that the entire ABC network can accept in the initial phase is approximately 24,960,000. It can accommodate up to 24 million online users, meeting the initial capacity requirements for the ABC IM system.

Now we discuss the upper limit of the ABC network service, where a Unity consists of up to 32 servers. According to this design, the number of total workers in these Unities can reach 323 (32,768), and the total number of system servers is around 1 million. Consequently, the maximum number of concurrent connections that the entire ABC network can accept is approximately 68,157,440,000, which can accommodate about 67 billion online users at any moment!

1.2.9 High Frequency Transaction Processing in Worker Ring

Since almost all events in the ABC IM system have to be paid, such as logging in and sending a message, we can foresee that the transaction frequency will be very high, resulting in heavy workloads on the chain. In order to solve this problem, a sharded transaction mechanism similar to the 'Lightning Network' (see ABC Blockchain Technology "Lightning Transaction" for details) is provided between the Worker Ring servers and the clients. This mechanism can batch the user spends following certain strategies (such as timed payment, one-time login,

etc.), which greatly reduces the transaction pressure on the main chain.

1.2.10 Unity Server Booting

Unity Server Booting in Unrestricted Network

When a Unity server on the Worker Ring DHT goes online, it first requests the Manager Ring for the information of joining the Unity. After obtaining this information, it joins the network and synchronizes data with other servers. As the Worker Ring has a stereoscopic node structure, there are usually multiple physical servers working for the same Unity. The new server can download all existing (key, value) pairs from them to populate its own DHT table.

In practice, each Unity server on the Worker Ring also maintains a "adjacency list", which contains information about other known Unities, such as their resources, abstract addresses, IP addresses and ports. Therefore, by issuing an initial query and obtaining adjacency lists from other servers, the new server is able to gradually expand its own list, and periodically remove obsolete entries as well.

However, when a Worker Ring Unity server has just launched, it may not know other existing Unity servers. This may happen, and hence the server cannot access any previously cached Unity servers or software hard-coded Unity servers. In this case, the Unity will send the datagram to a special "channel" from some relevant Unities. This approach does not need to know the recipient's public key in advance (but the sender's identity and signature should still be attached in the message), and thus the sent message will be transmitted without encryption (e.g., by sending an email). This approach is usually only used to request for the identity of the recipient (possibly creating a one-time identity for this purpose), and then conduct future communication in a more secure way.

Once at least one Unity is known, we can obtain more information by sending a special request to these known servers, and hence it is straightforward to populate more entries to the "adjacency list" and "routing table". Not all Unity needs to handle the datagrams sent to that channel, but those responsible for booting have to support this functionality.

Unity Server Booting in Restricted Network

The Worker Ring DHT network provides the solution for connections from restricted networks, which is useful for handling scenarios where servers in some particular areas are unable to connect to the network, due to the physical firewall blocking. When there exists a physical firewall barrier, the ABC system provides dedicated encryption and traffic obfuscation plug-ins to enable communications between Unities. Please refer to the appendix for the details about the review evasion.

1.2.11 Application Support with IM Example

Here we take the IM application as an example to illustrate how the Worker Ring works during the lookup process. Suppose that Alice and Bob are two accounts that have logged- in on the Worker Ring network. If Alice wants to send a message to Bob, he needs to lookup Bob's login information on the Worker Ring, which ensures that the IM message can be correctly delivered. Alice then sends server A (where he is located) a lookup request with RB as the key and B as the value. This value can be the ID (i.e., a 256-bit hash value) of the target object that the IM message is sending to. The server A then checks whether B is in its own

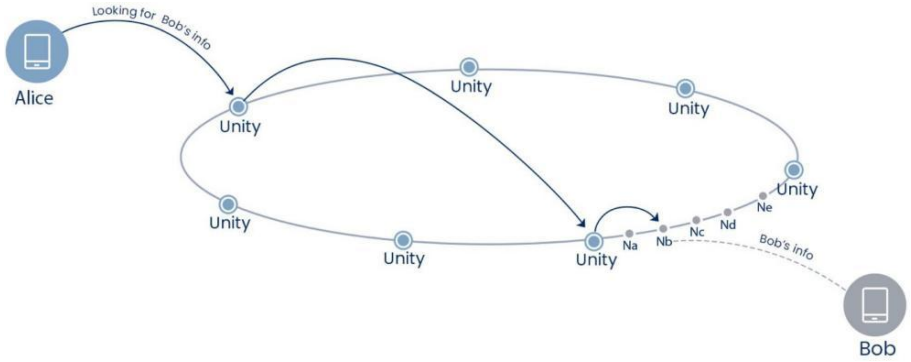

Figure 1.10: Example: How Alice Finds Blob on the ABC Network

Group: if yes, the target resource server can be directly located in the intra-group table within the Group; otherwise, the server finds the index server responsible for the target server Group from its inter-group table. After that, the index server helps to provide the corresponding intra-group table that can locate the target resource server. This resource server then retrieves and returns the information (i.e., the value B) about Bob's address, such as IP and port. Finally, the Alice's login server starts to connect with Blob, and Alice's message can be delivered to Bob's login server.

2 ABC Consensus Mechanism

One of the fundamental building blocks in blockchain system is the consensus mechanism which creatively tackles the problem of replicated agreement on transaction recording in an open-access, weakly synchronized network with the tolerance to arbitrarily behaving nodes. The consensus mechanism is not simply the consensus for algorithms and data sharing among computers, but more importantly the consensus for collaborations among partners. The consensus mechanism enables participants in the blockchain to collaboratively manage a shared ledger with an agreement, which guarantees the correctness, consistency and continuity of accounting among all collaborators. We will discuss the existing consensus mechanism in Section 2.1 and then propose our novel consensus protocol in Section 2.2.

2.1 Consensus in Blockchain Network

Networks in the context of distributed system, maintaining the canonical blockchain state across the P2P network can be seemed as a fault-tolerant deterministic replicated state machine that hosts all transaction. An agreement on a unique common view of the blockchain is expected to be achieved by the consensus nodes in the condition of Byzantine failures. In blockchain networks, Byzantine failures cause faulty nodes to exhibit arbitrary behaviors including malicious attacks/collusions (e.g., Sybil attacks and double-spending attacks), node mistakes and connection errors. In other words, the consensus is the process of determining the transaction sequence and filtering illegal transactions ensuring that transactions are objectively recorded over the entire network and cannot be tampered. A blockchain updating protocol is said to achieve the consensus in a Byzantine environment if the following properties are satisfied:

Validity: If all the honest nodes activated on a common blockchain state propose to expand the blockchain by the same block, any honest node

transiting to a new local replica state adopts the blockchain headed by that block.

Agreement: If an honest node confirms a new block header, then any honest node that updates its local blockchain view will update with that new block header.

Termination: All transactions originated from the honest nodes will be eventually confirmed.

Total Order: All honest nodes accept the same order of transactions as long as they are confirmed in their local blockchain views.

The consensus protocols vary with different blockchain networks. Since the permissioned blockchain networks admit tighter control on the synchronization among consensus nodes, they may adopt the conventional Byzantine Fault-Tolerant (BFT) protocols to provide the required consensus properties. In a network of *n* consensus nodes, the BFT-based protocols are able to conditionally tolerate faulty nodes up to $\frac{n}{3}$. On the contrary, permissionless blockchain networks admit no identity authentication or explicit synchronization schemes. Therefore, the consensus protocol therein is expected to be well scalable and tolerant to pseudo identities and poor synchronization. Since any node is able to propose the state transition with its own candidate block for the blockchain header, the primary goal of the consensus protocol in permissionless networks is to ensure that every consensus node adheres to the "longest chain rule". namely, when the blocks are organized in a linked list, at any instance, only the longest chain can be accepted as the canonical state of the blockchain. Due to the lack of identity authentication, the direct voting based BFT protocols no longer ensure the consensus properties in permissionless blockchain networks. Instead, the incentive based consensus schemes such as the Nakamoto consensus protocol are widely adopted.

2.1.1 Existing Consensus Mechanism

To ensure proper functioning of a permissionless blockchain network, Satoshi Nakamoto innovatively combines a consensus protocol based on a framework of cryptographic block-discovery racing game with economic incentives to probabilistically award the consensus participants based on an embedded mechanism of token supply and transaction tipping in the Bitcoin system, which is known as the Proof of Work Scheme. The Proof of Work requires miners to perform a moderate amount of computational work to solve the hash puzzle, which raises the cost of being a malicious node (e.g., publishing a fake block) and ensures that any consensus node will suffer from finical loss whenever it deviates from truthfully following the protocol. Inspired by Satoshi Nakamoto, many blockchain consensus mechanism have been proposed. In this section, we discuss the consensus protocols used by mainstream blockchain projects.

1. **Proof of Work (PoW)**

   To obtain the eligibility of a block commitment, each node has to solve a computation puzzle involving the hash of the previous block, the hash of the transactions in the current block, i.e., finding a random nonce that satisfies the constraints. That node can then add a valid block and broadcast it to all other nodes in the network after being verified. Other miners adopt and add block to the longest chain, which has the greatest Proof-of-Work effort invested in it. The advantage of PoW is that it is completely decentralized, and nodes can freely join and leave the network. The disadvantages and limitations are also obvious: Bitcoin has already attracted most of the computation resources in the world, which makes other PoW-based blockchain applications difficult to obtain comparable resources and achieve similar security levels; the block mining wastes huge amounts of electrical energy and other relevant resources; it takes a long period to reach the global consensus, which is not suitable for commercial applications.

2. **Proof of Stake (PoS)**

   In the Proof of Stake system miners do not compete, instead a validator set is maintained. Anyone, who owns blockchain's coins, can join this set by

locking all his coins, called the stake, into a deposit. The validators participate then in the block creation process, where two major types of consensus algorithms are used. In Chain-based PoS the validator, who has the right to create the block, is periodically pseudo-randomly selected. In Byzantine-fault-tolerant-style PoS the validators can propose blocks, the right to do so is randomly assigned to them, further the validators then agree or disagree on the proposed blocks by voting. The block creator gets transactions fees instead of block rewards. Therefore, all coins are created in the beginning, and their number never changes. Advantages of PoS are that less energy is needed for consensus and the increased protection against attacks.

3. **Delegated Proof of Stake (DPoS)**

   DPoS is similar to the board voting, where the stakeholders select a certain number of nodes as their delegation for verification and accounting. In EOS, for example, there is a new block produced every 3 seconds, and only one delegated node can produce the block at any point of time. A block will be skipped if it is not produced within a specified time period. There are 21 delegated nodes taking turns to produce the blocks. At the beginning of each round, 21 unique nodes are selected by the system as block producers. The selected producers then start to produce blocks following a pseudo-random sequence. In general, there will not be any forking in the DPoS-based blockchain, since block producers work in a collaborative way rather than competitively. Therefore, this might be a better solution for our system. Advantages: significantly reduce the number of participating nodes for block verification and accounting, and achieve second-level consensus verification. Disadvantages: the consensus protocol entirely relies on tokens, while many commercial applications do not require tokens.

4. **Proof of Elapsed Time (PoET)**

   The Proof of Elapsed Time consensus protocol is proposed by Intel, and it randomly elects a leader node from a number of validators to produce the new block. The election method relies on a secure timer (from the Intel SGX

Secure Guard Extensions framework) running on each node. The first node that has its timer expired is elected as the leader for submitting the next block. In the PoET protocol, the secure timer is just a simple counter, which means that it requires only a small amount of computation power to achieve consensus between thousands of nodes.The Intel Sawtooth platform applies the PoET consensus protocol and achieves significant advantages in terms of performance and scalability. However, as PoET relies on the primitives in Intel chips and lacks incentives for non-business participants, it may not be widely adopted on public networks. For private and federal networks, PoET might be a feasible alternative to PoW.

5. Practical Byzantine Fault Tolerance (PBFT)

PBFT[6] is a message-based consensus protocol. In normal case, it runs a three-phase protocol: pre-prepare, prepare, and commit. A client sends a request to one of the peers, who in turn broadcasts pre-prepare messages to the other peers. In the prepare stage, a prepare message is multicasted to all other nodes. When a replica receives $2f$ prepare messages, it matches with the pre-prepare message and multicasts a commit message.

Once the replica receives commit messages, that match the pre-prepare message, it changes the state to committed and executes the message operation. Once the message is executed, a reply is sent to the client. The main purpose of a Byzantine fault tolerant consensus algorithm is to allow the system to be able to survive and continue work despite some of the machines exhibiting arbitrary faults. Although, PBFT is a consensus algorithm with proven security and liveness properties, the network overhead during consensus round does not allow scale the consensus protocol, limiting the throughput of the whole system. It was shown that PBFT can be attacked by an adversary using a simple scheduling mechanism, halting the consensus either completely, or forcing to wait long timeout when leader is partitioned and unsynchronized.

The following figure shows the workflow of 4 nodes reaching the agreement, in which node 0 is the leader, and the node 3 is a faulty node

that does not respond or send any message. When the last node's status becomes committed, it indicates that the current round of agreement has been successfully reached.

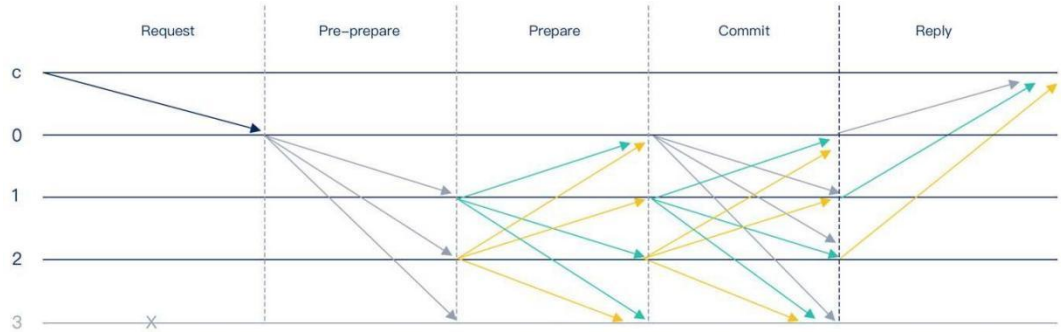

Figure 2.1: PBFT Protocol

In summary, the PoW (Proof of Work) consensus protocol used in Bitcoin is essentially a competition of computation resources. Since the total computation power in the Bitcoin network today is much higher than when the network was created, it is now more computationally difficult for a node to produce a block. As a result, miners have to consume a huge amount of computation resources for mining, which inevitably wastes lots of electrical energy. This is a severe waste of global resources, and is absolutely environmental-unfriendly. Moreover, to improve the mining efficiency in the Bitcoin network, some special and dedicated computers – mining machines – are assembled, which further form large mining pools jointly. Due to the existence of miners and mining machines, the community has a substantial concern about the centralization trend of the decentralized algorithm, i.e.,. It is difficult for normal clients to compete with these mining pools. This phenomenon results in the consequent that the Bitcoin network is becoming more and more centralized, and the network security is therefore declining. Besides, the existence of PoS is mainly supported by considerations and innovations from an economic point of view, such as the concepts of equity and interest. Although the PoS (Proof of Stake) consensus protocol solves the resource wasting problem in PoW, it has its own limitation: if a client only holds a small amount of coins, the probability for him to mine a

block is also very small, leading to a Matthew effect. Thereafter, the DPoS appears, which does not require massive computation power to distribute equity. However, as an alternative protocol, it is unlikely that the DPoS is capable of replacing PoW, PoS or PoW + PoS. After all, what is actual is rational: each protocol has its own technical and business considerations (or implications) during a specific historical time period.

2.2 ABC Consensus

In this section we describe ACP, an efficient blockchain consensus protocol for partial synchronous setting tolerating $f < n/3$ fault. ACP targets both high-performance in fault- free executions and correctness if Byzantine nodes exhibit arbitrary behavior. There has been many existing distributed consensus protocols that tolerate faulty nodes and later the ones that tolerate Byzantine nodes. However, their application was limited to a small scale. And a stable leader is required to facilitate the agreement in classic distributed consensus protocols from Multi-Paxos [8], Raft [9] to ZAB [45] protocol. In the permissionless blockchain setting, any node in the system may be Byzantine to exhibit arbitrary malicious behaviors. Therefore, the assumption that there will be a stable leader is vulnerable. Comparing with others blockchain consensus protocols, the key aspects of ACP can be summarized as follows:

1. **Early Stopping Consensus**

   An attacker powerful enough to control up to $1/3$ of the nodes are commonly assumed in the Byzantine threat model. However, the assumption is rather pessimistic. In the design of ACP, leveraging the governance of the AI module of ABC blockchain, we can assume a relative steady composition of ABC network without frequent join and leave of temporary nodes. Therefore, ACP aims to achieving the agreement as soon as possible in the case of Byzantine nodes far less than $1/3$, while guarantees the safety and liveness when up to $1/3$ nodes are Byzantine. That is, the consensus algorithm should have the "early-stopping" feature.

2. Parallel Byzantine Agreement Instances

Classic PBFT protocol relies on a stable leader to begin each instance. In case the leader node is Byzantine, it can slow down or stall the system by causing frequently view changes. To address this challenge, ACP runs multiple Byzantine agreement instances in parallel for a single block proposal. The overall efficiency is improved since a small scale is selected to run the PBFT protocol.

3. Prior or Posterior Strategies

In the blockchain consensus, some posterior techniques, such as VRF [35], are usually employed to improve security; on the contrary, in order to improve efficiency and reduce complexity, some prior techniques such as well-known static grouping and round-robin scheme are needed. ACP combines the characteristics of other components in ABC to make reasonable trade-offs, with the prior and posterior strategies, to ensure the safety, liveness and efficiency.

System Model

Node Each node represents a full node in ABC Blockchain and can communicate with other nodes in a peer-to-peer fashion. We use the term non-faulty to refer to nodes in the network that follow the protocol's instructions without error and obey the ABC governance strategy, and are perfectly capable of sending and receiving messages. Conversely, a node is Byzantine if he can deviate from the protocol in a completely arbitrary way, from simple crashes, to malicious behavior aimed at disturbing the consensus, fully coordinated between all Byzantine nodes.

The system consists of n nodes, out of which up to $t < n/3$ may be Byzantine, i.e., behave arbitrarily and collude together. Denote by $f \leq t$ the actual number of Byzantine nodes in a given run. A few types of synchronous environments we refer to throughout the paper are given hereafter.

Synchronous Network A network is said to be strongly synchronous if there exists a known fixed bound Δ such that every message delays at most Δ time when sent from one point in the network to another.

Partial Synchronous Network A network is said to be partially synchronous if there exists a fixed upper bound Δ on a message's traversal delay and a fixed upper bound Φ on relative processor speeds and one of the following holds:

1. Δ always holds, but is unknown.
2. Δ is known, but only holds starting at some unknown time.

We assume that the communication is partially synchronous in this work and state the goals of consensus for the ACP Protocol below.

Byzantine Consensus

The Byzantine consensus problem consists of each node i having an initial value v_i from a finite set V (i.e., $v_i \in V$). Each node i also has an output value $o_i \in O$. Two properties should hold:

1. **Agreement**: $o_i = o_j$ for any two non-faulty nodes i, j (thus we can talk about the output value of the algorithm);
2. **Validity**: if all non-faulty nodes start with the same initial value v, then the output value of the algorithm is v.

It is well-known that consensus can not be solved in asynchronous systems [5]. Dis-tributed consensus is implemented in partially synchronous systems.

Formal Consensus Goals

Assume a system of n nodes, where each node n_i has a private value v_i, and the following must be achieved:

1. **Agreement**: All non-faulty nodes must agree on the same value.
2. **Validity**: If all non-faulty nodes have the same initial value v, then the

agreed upon value by all non-faulty nodes is *v*.

3. **Termination**: All non-faulty nodes must eventually decide on a value.

2.2.2 Notations and ParABC ters

r: the current round number.

N_{all} : the total number of nodes in the system at the beginning of round *r*.

N_{pc}: the participating numbers of Potential Committee.

N_{fc} : the participating numbers of Final Committee.

$N_{fc-valid-leader}$: the expected numbers of nodes issue PBFT instance with valid block as initial value.

$N_{fc-empty-leader}$: the expected numbers of nodes issue PBFT instance with empty block as initial value.

RS_r : the random seed of round *r*.

PK_i : the public key of node *i*, which are known to all nodes in ABC Blockchain.

SK_i : the secret key of node *i*, which are stored locally.

SE_i^r : the secret string of node *i* in round *r*. *SE* is a fixed-length bit string updated periodically by each node, and used for generating the random seed *RS*.

rep_i^r : the reputation of node *i* in round *r*.

pw_i^r : the weight of node *i* in round *r*.

RI : the secret string refresh interval.

τ_{pc} : the expected numbers of Potential Committee.

τ_{fc} : the expected numbers of Final Committee.

σ_i^r : the credential of node *i* in round *r*.

λ_{pc}: the upper-bounds to the time needed to broadcast a message to the

whole Potential Committee.

λ_{fc}: the upper-bounds to the time needed to broadcast a message to the whole Final Committee.

λ_{all}: timeout to broadcast to the whole network.

B_i^r: the block proposed by node *i* in round *r*.

B_{cb}^r: the candidate block in round *r*.

$B_\emptyset^r$: the empty block in round *r*.

$B_{pbft-input}^r$: the block passed to PBFT in round *r*.

H: a cryptographic hash function

SBR: a synchronization barrier.

2.2.3 ACP Overview

In this section, we provide an brief overview of ACP protocol. We begin by presenting an figure showing the basic structure of ACP Protocol and then describe the individual building blocks. Each round of ACP consists of 4 stages as illustrated below:

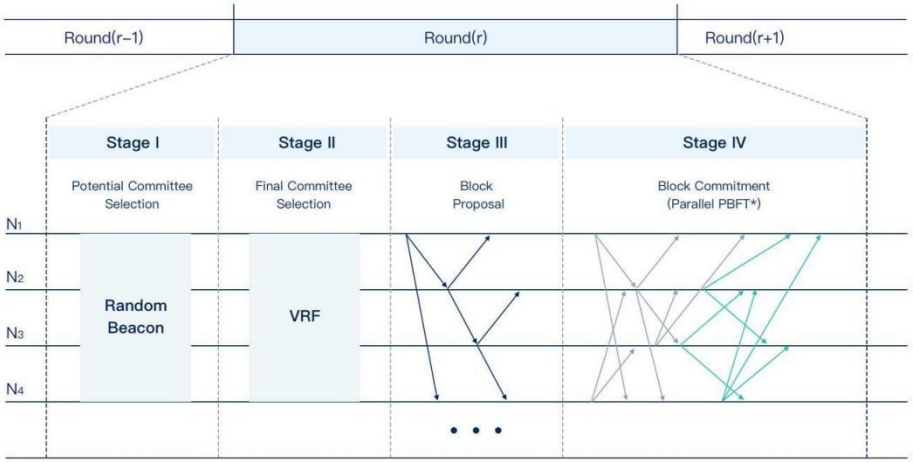

Figure 2.2: An Overview of ACP Protocol

The ACP protocol proceeds in round. At the end of each round, either a valid non-empty block containing a set of transactions or an empty block is agreed

among all participating nodes and appended to the blockchain. Each node identifies current round number from local newest block. Each round of ACP consists of the following 4 stages as illustrated in **Figure 2.2**.

Stage I This stage is to select Potential Committee from ABC Blockchain, in order to provide a balance between efficiency and resource usage, avoiding broadcasting messages to the whole network. This is one of the key techniques we exploit to overcome scalability challenge. The selection is based on a randomness beacon. Each node computes hash values (the outputs of a pre-specified hash function H_{rb} of all nodes and selects a subset of nodes to proceed to next stage as the Potential Committee according to a given threshold N_{pc}.

Stage II This stage is to select a much smaller subset Final Committee of the set Potential Committee, aiming at increasing resilient against adaptive attacks. Instead of adopting a hash function to determine eligibility in the Potential Committee Selection, we rely on a Verifiable Random Function (VRF) [35] instead. The VRF ensures that the adversary cannot predict in advance which nodes are the block proposers. After the selection, each node selected in Final Committee propose their candidate block proposals from local pending transaction pool and then broadcast a signed message including their respective candidate block proposals, signatures, selection hashs and the hash proofs (B_i^r, $sig(B_i^r)$, σ_i^r) to all members in Potential Committee.

Stage III After waiting an amount of time λ_{pa}, each node i in Final Committee chooses a
candidate block from his received block proposals $B^r_{i\epsilon\{0,\ldots.fc-1\}}$, denoted by B_i^r. And then Final Committee members start a two-step Reduction procedure [37]. At the end of Reduction Procedure,each Final Committee member outputs a valid candidate block B_{cb}^r that received at least $2N_{fc}/3 + 1$ votes in the second step of Reduction procedure or an empty block B_ϵ^r if no hash received enough votes. The Reduction procedure convertsthe problem of reaching consensus on an arbitrary value (the hash of a block) to reaching consensus on one of two values: either a

specific proposed block hash, or the hash of an empty block.

Stage IV Each Final Committee member i acts according to his output value at the end of stage III as follows.

- If i outputs B_{cb}^{r} and $H(\sigma_i^r)$ is among the $N_{f\ c-valid-leader}$ least value in $H(\sigma_{i\epsilon\{1,\ldots.f\ c-1\}}^{r})$, then i runs the optimized PBFT* instance as leader node with B_{cb}^{r} as initial value. At the same time i runs the others optimized PBFT* instances with the least $N_{f\ c-valid-leader}$ hash values as initial value in parallel.
- If i outputs B_{ϱ}^{r} and $H(\sigma_i^r)$ is among the $N_{f\ c-empty-leader}$ largest value in $H(\sigma_{i\epsilon\{1,\ldots.f\ c-1\}}^{r})$, then i runs the optimized PBFT* instances as leader node with B_{ϱ}^{r} as initial value.At the same time i runs the others optimized PBFT* instances with the largest $N_{fc-empty-leader}$ hash values as initial value in parallel.

Final Committee members propagate the agreed-upon block once they received the agreed-upon result from any optimized PBFT* instance. In this case, the node reaches Final Consensus. On the other hand, tentative Consensus means that it has not yet received the agreed-upon hash and broadcast an empty block B_{ϱ}^{r} after waiting a large amount of time SBR.

2.2.4 ACP Details

We now describe each of the components of the ACP protocol in more detail.

Initialization of The Protocol

The protocol starts with $r = 0$. The initial random seed RS_0 is generated by a (Public)-VSS Coin Tossing scheme [46] and hard-coded into the genesis block.

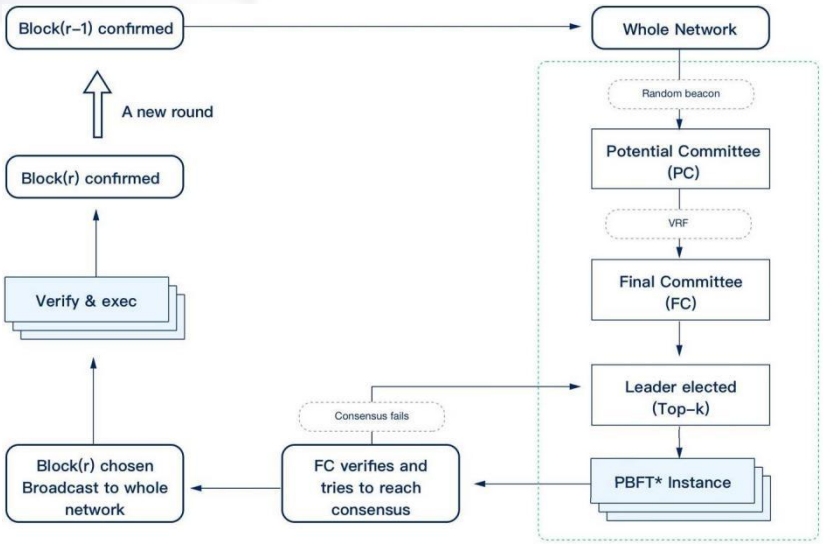

Figure 2.3: The Flowchart of ACP

Potential Committee(PC) Selection

Each Node in ABC Blockchain calculates for all nodes the Potential Weight(PW) [36]:

$$pw_j^r = \frac{H(RS^{r-1}, r, PK_j)}{Rep_i^r} \tag{2.1}$$

Where $j = \{0, 1, \ldots, n\ 1\}$ is the indicator of nodes, RS^{r-1} is the random seed of previous round, r is the current round number, PK_j is the corresponding public key of node j, rep_i^r is the reputation of node j in round r, H is the pre-specified hash function. According to the weights and the expected Potential Committee size m, we select the m nodes with lowest potential weight into Potential Committee.

Final Committee(FC) Selection

Final Committee Selection is implemented using VRF [35] to randomly select Final Committee Members in a private and non-interactive way. A VRF is a triple of algorithms Keygen, Evaluate, and Verify.

VRFGen: On a random input, the key generation algorithm produces a

public key PK and a private key SK pair.

VRFEvaluate: The evaluation algorithm takes the private key SK, a message X as input and produces a pseudorandom output string Y and a proof ρ.

VRFVerify: The verification algorithm takes the public key PK, the message X, the output Y and the proof ρ as input. It outputs 1 if and only if it verifies that Y is the output produced by the evaluation algorithm on inputs SK and X.

In the Final Committee Selection Stage, each node in Potential Committee checks whether he is selected in Final Committee. If this is the case, he collects a block of transactions from pending transaction pool as his candidate block proposal BP and then broadcast a signed message to Potential Committee which includes the candidate block proposal BP, the selection output hash and its proof of selection.

After waiting an amount of time

, each Final Committee member chooses a candid ate block

from his received block proposals $B^r_{i \in \{0,....f\ c-1\}}$

The chosen mechanism is described below:

- Chooses B^r_i with the largest transaction size among $B^r_{i \in \{0,....f\ c-1\}}$.
- Chooses B^r_i with the least hash value among $H(\sigma^r_{i \in \{0,....f\ c-1\}})$ in case of multiple nodes with the same transaction size.

And then each node in Final Committee starts a two-step Reduction procedure [37]. The Reduction procedure satisfies two properties:

- If agreement is alert = true, there are non-faulty processes with different initial values from *V*. In this case, all non-faulty processes use a predefined default value from *V* as the result of the following steps.
- If agreement is alert = false, then all non-faulty processes have the same initial value from *V*. This value is the result of the following steps.

In the first step of Reduction procedure, each Final Committee member votes

for the hash of the candidate block chosen by the above mechanism . In the second step, Final Committee members vote for the hash that received at least $2N_{fc}/3 + 1$ votes in the first step, or the hash of the default empty block if no hash received enough votes. After the second step, each Final Committee member outputs a valid candidate block B_{cb}^{r} that received at least $2N_{fc}/3 + 1$ votes in the second step or an empty block B_{ϱ}^{r} if no hash received enough votes.Note that a valid block B_{cb}^{r} output and an empty block B_{ϱ}^{r} output corresponds to the above properties with true alert and false alert, respectively. Let us denote the output of Stage III by $B_{pbft-input}^{r}$

Reach Consensus

After Final Committee members having outputting their candidate block respectively, reaching agreement on the final block among all non-faulty Final Committee members remains the main problem. Classic Byzantine fault tolerance consensus protocols requires a stable leader node to facilitate the agreement. A Byzantine leader can cause frequent view changes which would prevent forward progress [47,48]. In ACP, each Final Committee runs multiple PBFT* instances in parallel to circumvent the Byzantine leader problem.

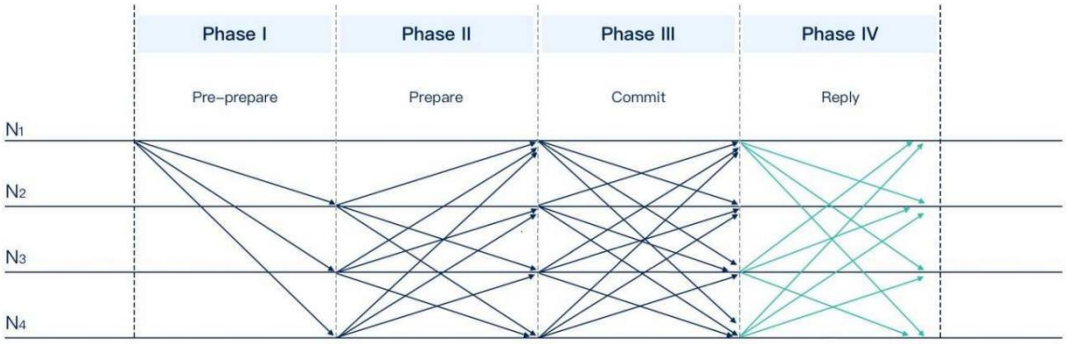

Figure 2.4: PBFT* Protocol

PBFT* : Each round of classic PBFT [6] consists of 4 phases: Pre-prepare, Prepare, Commit and Reply. In the Reply phase, nodes send the committed result to the client and the client is aware of the agreement with replied result. In ABC

blockchain network, each node is an equal amongst others and hence, whether the agreement is achieved should be known to every node as soon as possible. We modify the Reply phase to achieve this goal by broadcasting committed result to the whole Final Committee instead. As a result, each node in Final Committee can verify whether the agreement is achieved and enter next round as early as possible.

Parallel Instances: Most of the existing PBFT-based consensus protocols perform single PBFT instance to achieve consensus in each round. If the leader node itself is a Byzantine node, the communication complexity will boost to a staggering $O(n^3)$ [49]. In ACP protocol, multiple nodes are selected from Final Committee as leaders to execute multiple PBFT* instances in parallel. Although the leaders chosen by each node may be different, it will not hamper the safety of the protocol. The process is described below.

P1: Each node in Final Committee outputs either B^r or B^r_ϱ B^r_{cb}.

P2: Each node in Final Committee runs multiple PBFT* instances in parallel. If a node's
local alert is equal to false, it rejects to run the PBFT* instance with B^r_ϱ as initial value.

P3: Denote the set of instances with B^r_{cb} and B^r_ϱ as initial value by α and β, respectively. Use 1 to represent the state of reaching agreement, 0 otherwise. We enumerate the following potential agreement scenarios:

(1) $\alpha = 1, \beta = 0$;
(2) $\alpha = 0, \beta = 1$;
(3) $\alpha = 0, \beta = 0$;

P3.1 In case (1) and (2), all non-faulty nodes with successful completion of any optimized PBFT* instance will broadcast the agreed-upon result to the whole network. This scenario is defined as final consensus.

P3.2 In case (3), each node can not distinguish this scenario from the others two scenarios. So each node has to wait a relative large amount of time *SBR* and broadcasts an empty block B_{ϱ}^{r} if it has not yet received the agreed-upon result till the completion of wait. This scenario is defined as tentative consensus.

P4: The agreed-upon block will be appended to blockchain and transactions contained in B_{cb}^{r} will be confirmed instantly in the case of achieving final consensus. Transac-tions from a tentative block will be confirmed only if and when a successor block reaches final consensus.

P5: ACP produces two kinds of consensus: final consensus and tentative consensus. The introduction of these two notion is to guarantee the liveness. With a negligible probability, the presence of network partition can create a fork in the network and forks will severely impact the liveness in turn. To mitigate this problem, a recovery protocol is proposed below.

P5.1 Each node becomes aware of potential forking by checking whether the previous block hash in current block proposal is the same with the last local block hash.

P5.2 A node aware of potential forking proposes an empty block whose predecessor hash is the last final consensus block $B_{last-final-block}$ observed by him so far.

P5.3 Finally, the node invokes ACP to reach consensus on forked blocks, choosing the block with highest $B_{last-final-block}$ as his candidate block in the Stage III instead.

In the process of executing PBFT* protocol, each node in Final Committee signed the message broadcasted by itself. Once reaching agreement, Final Committee members will broadcast the agreed-upon block with these signatures, allowing any nodes to validate the correctness of a block.

Stage 1: Potential Committee (PC) Selection

Instructions for every node in the network: Node i calculates for all nodes the Potential Weight(PW):

$$pw_j^r = \frac{H(RS^{r-1}, r, PK_j)}{rep_i^r} \tag{2.2}$$

where $j = \{0, 1, ..., n - 1\}$ is the indicator of nodes, RS^{r-1} is the random seed of previous round, r is the current round number, PK_j is the corresponding public key of node j, rep_i^r is the reputation of node j in term r.H is the pre-specified hash function.

According to the weights and the expected Potential Committee size m, node i checks where $i \$ PC^r$ or not.

- If $i \$ PC^r$, then i stops his own execution of ACP right away.
- If $i \in PC^r$, then i moves to Stage 2.

Stage 2: Final Committee (FC) Selection

Instructions for every node in PC: Node i computes its hash output σ_i^r with round number r as input string and checks whether $i \in FC^r$ or not according to the expected Final Committee size.

- If $i \$ FC^r$, then i stops his own execution of Stage 2 right away.
- If $i \in FC^r$, then i broadcasts $(1, B_i^r, sig(B_i^r))$ to all Potential Committee members and moves to Stage 3.

Stage 3: Candidate Block proposal

Instructions for every node in FC:

3.1 After waiting an amount of time λ_{pa}, node i votes the hash of the candidate block chosen from his receive block proposals $B^r_{i\epsilon\{0,\ldots.f\ c-1\}}$ by the following mechanism:

- Chooses B^r_i with the largest transaction size among $B^r_{i\epsilon\{0,\ldots.fc-1\}}$.
- Chooses B^r_i with the least hash value among $H(\sigma^r_{i\epsilon\{0,\ldots.fc-1\}})$ in case of multiple nodes with the same transaction size.

3.2 After waiting an amount of time λ_{fa}, i votes for the hash that received at least $\frac{2N_{fc}}{3}$ + 1 votes in the Step 3.1.

3.3 After waiting an amount of time λ_{fa}, i outputs a valid candidate block B^r_{cb} that received at least $\frac{2N_{fc}}{3}$ + 1 votes in Step 3.2 or an empty block B^r_{ϱ} if no hash received enough votes.

Stage 4: Reach Consensus

Instructions for every node in FC:

4.1 Node *i* acts according to his output value at the end of Stage 3 as follows.

- If *i* output B_{cb}^{r} and $H(\sigma_i^r)$ is among the $N_{f\ c-valid-leader}$ least value in $H(\sigma_{i\epsilon\{1,\ldots.f\ c-1\}}^{r})$,then *i* runs the PBFT* instance as leader node with B_{cb}^{r} as initial value . At the same time *i* runs the others PBFT* instances with the least $N_{f\ c-valid-leader}$ hash values as initial value in in parallel.

- If *i* outputs $B_{\varrho}^{r}$and $H(\sigma_i^r)$ is among the $N_{fc-empty-leader}$ largest value in $H(\sigma_{i\epsilon\{1,\ldots.f\ c-1\}}^{r})$, then *i* runs the PBFT* instances as leader node with B_{ϱ}^{r} as initial value. At the same time *i* runs the others PBFT* instances with the largest $Nfc-empty-leader$ hash values as $H(\sigma_{i\epsilon\{1,\ldots.f\ c-1\}}^{r})$, then *i* runs the PBFT* instances as leader node with B_{ϱ}^{r} as initial initial value in parallel.

4.2 After waiting a relative large amount of time *SBR*, *i* checks whether it has received an agreed-upon block from any PBFT instance or not.

- If *i* has received an agreed-upon block, then *i* propagates the received final consensus block B_{cb}^{r} or B_{ϱ}^{r}.
- If *i* has not received an agreed-upon block, then *i* broadcasts an empty tentative consensus block B_{ϱ}^{r}.

Stage P: Propagate Consensus

Instructions for every node in the network:

- If node *i* has received an final consensus block, then *i* appends the agreed-upon block to the blockchain.
- If node *i* has received an tentative consensus block, then *i* keeps the received tentative consensus block as pending state until it received a successor final consensus block. After the reception of a successor final consensus block, it appends the tentative consensus block and the successor final consensus block to the blockchain.

Stage R: Recovery Protocol

With a negligible probability, the presence of network partition can create a fork in the network and forks will severely impact the liveness in turn. To mitigate this problem, a recovery protocol is proposed below.

- Each node monitors potential forking by checking whether the previous block hash in current block proposal is the same with the last local block hash.
- A node aware of potential forking proposes an empty block whose predecessor hash is the last final consensus block $B_{last-final-block}$ observed by him so far.
- The node aware of potential forking invokes ACP to reach consensus on forked blocks, choosing the block with the highest $B_{last-final-block}$ as his candidate block in the Stage 3 instead.

2.2.5 Proof of Safety Property

Assume that the numbers of Final Committee is *n*. After Reduction stage, there are two possible states of the output of each Final Committee member.

- alert = false, which corresponds to a non-empty block proposal, denoted by B_{cb}^{r}.
- alert = true, which corresponds to a empty block proposal, denoted by B_{ϱ}^{r}.

We now enumerate the possible value of the numbers of Final Committee members with non-empty block initial value, denoted by *p*, and the numbers of Final Committee members with empty block initial value, denoted by *q*.

Case1 If $p \geq \frac{2n+1}{3}$, then only the PBFT* instance with B_{cb}^{r}initial value will complete successfully,due to the fact that the majority nodes with alert=false will reject the PBFT* instance with a B^{r} initial value .

Case2 If $\frac{2n+1}{3} > p > 0$, then all of the instances may complete with a failure result for the presence
of both instances with B_{cb}^{r} initial value and with B_{ϱ}^{r} initial value.

Without loss of generality, we can assume that instances with B_{cb}^{r} initial value are PBFT_{x1}^{*}, PBFT_{x2}^{*}, PBFT_{x3}^{*} and instances with B_{ϱ}^{r} initial value are PBFT_{y1}^{*}, PBFT_{y2}^{*}, PBFT_{y3}^{*}.

PBFT instances with B_{ϱ}^{r} initial value, that is PBFT_{y1}^{*}, PBFT_{y2}^{*}, PBFT_{y3}^{*}, will abandon their running instances in any cases once they are aware of the existence of instances with B_{cb}^{r} initial value,that is PBFT_{x1}^{*} , PBFT_{x2}^{*} , PBFT_{x3}^{*}.

The only case that PBFT_{y*}^{*} has achieved agreement while PBFT_{x*}^{*} have not sent the first message yet is that the number of PBFT instances with B_{cb}^{r} initial value is very few.And furthermore PBFT_{x*}^{*} and PBFT_{y*}^{*} are located in different network partition. In this case, we let each node wait an amount of time λ after sending a broadcast and tag a logic index for each received message *r* − *s* : 4-phase:*i*, which means the *r*-th round, 4-stage in*i*-th phase. Assume that the start time of a phase is $T_{phases-i}$, then in [$T_{phases-i}$, $T_{phases-i}$ + λ], a node will receive most message sent from honest node and discard the message after $T_{phases-i}$ + λ. With this

method, We can ensure that only one instance can achieve agreement.

Case3 p=0, then all PBFT instances are with B_{ρ}^{r} initial value, and thus all instances will complete successfully. We lists the instructions for each node *i* in Final Committee:

- If alert=false, then *i* rejects PBFT instances with B_{ρ}^{r} initial value.
- If alert=true and there exists running PBFT instances with B_{cb}^{r} initial value, then i rejects PBFT instances with B_{ϵ}^{r} initial value if no agreement has achieved, otherwise, abandon the current running instance.
- Node *i* has to wait an amount of time to receive most of the message sent from honest node after broadcasting a message in $T_{phases-i}$ and discard all overtime messages.

2.2.6 Security Analysis

To achieve scalability and keep resilient against large scale DDOS attacks, two committees-the Potential Committee and the Final Committee are randomly selected from the total set of nodes. The parallel PBFT* protocol is then run within Final Committee.

The agreement is achieved in the Final Commit and broadcast to the entire network. Therefore, when the Potential Committee and Final Committee can be normally selected and the Byzantine nodes is less than a third of the selected committee members, the nature of safety and liveness can be guaranteed by PBFT*. The selection of Potential Committee is driven by the decentralized random beacon which ensures that the generation of random seed is provably immune from manipulations and unpredictable. After revealing the random seed, the membership of Potential Committee are known to all nodes. As to the membership of Final Committee, only after each node in Final Committee constructs his block proposal and broadcasts his identity, the membership of Final Committee are known to all nodes in Potential Committee. There is

therefore potential attacks towards Potential Committee and Final Committee members. Below we will analyze the various types of attack and the economic incentives of these attacks to demonstrate that large-scale attacks on members of the PC cannot be performed successfully in the ABC network.

Nothing at Stake The Nothing at Stake attack is when the validator casts different blocks in the voting stage without penalty, which may causes the ACP consensus protocol to generate an empty block. In response to this attack, the ACP consensus solution is to provide economic incentives to validators in Potential Committee and Final Committee only in the case that valid block are generated without forking.

Selfish Mining Selfish mining refers to the behavior of the selected block proposer prioritizing or only packaging the transaction that is beneficial to itself, so as to complete the confirmation of the self-interest transaction. Since all transactions are valid, the verification nodes cannot distinguish this behavior. In addition, because of its small size, the broadcast speed is faster and easier to be confirmed by other nodes. From this perspective, Final Committee members tend to selfish mining in order to prioritize their own interest-related transactions, reduce waiting time, and have a greater chance of becoming a final miner. To address this issue, we associate the volume of transactions in the blocks proposed by Final Committee members with the gains they can earn as validator.

Sybil Attack Sybil attack in blockchain network is an attack where a single adversary is controlling multiple nodes in the network. Each node in ABC network has unique identity, and the AI governance module in ABC network also requires each node to report the corresponding identity, ip and other network attributes, which will make Sybil Attack difficult to appear in the ABC network.

Eclipse Attack Eclipse attack, which targets a specific node and sends them blocks of a private fork, while attempting to eclipse them from the rest of the network so that they don't see the main blockchain. According to the design of

ABN, each node has a large number of neighbor nodes and counterpart nodes, which mitigates the eclipse attack in ABC network. In addition, to defend against Eclipse Attack, the AI governance module in ABC network detects whether the interconnection graph of each node is abnormal.

Adaptive Adversary The AI governance module in ABC will adjust the reputation of each node according to his long-term performance. Once an malicious behavior is detected by the governance module, the Byzantine node will be punished by reducing its reputation. Therefore, it is controlling a large number of nodes over the threshold in the Potential Committee and Final Committee to manipulate the generation of the block for a long time can be eliminate in ABC system.

Incentive as Countermeasure of Threat

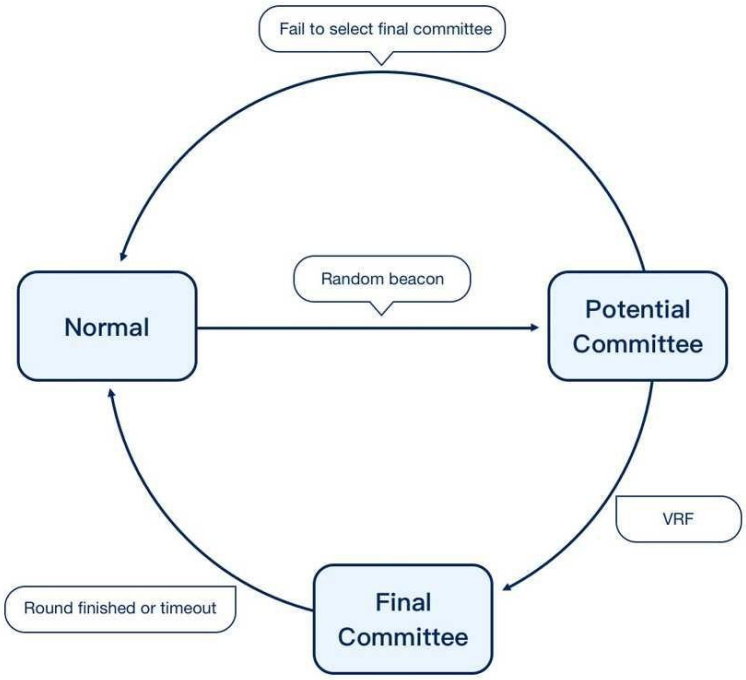

Figure 2.5: Node State Transition

PC Incentives The Potential Committee is selected in the first stage of ACP pro- tocol. Potential Committee members further selected Final Committee

through VRF calculations. Economic incentives should be granted for their devoted works. More importantly, Potential Committee members are motivated to ensure that they do not maliciously broadcast Final Committee membership to other nodes in the network, causing Final Committee members to be attacked by Byzantine nodes. When a node in Potential Committee knows that it has not selected in Final Committee in this round and has no chance to get the "mining" reward, there may be malicious attack toward Final Committee members. Therefore, we propose that Potential Committee members will be rewarded in next round given that the round they participated has successfully completed. Thus, ACP can avoid the Nothing at Stake attacks.

FC Incentives Final Committee is selected in the second stage of ACP protocol. Final Committee members bear the responsibility of constructing block and establishing consensus through the PBFT* algorithm, which is also the "mining" in the traditional sense. In order to encourage the miners to work honestly and reduce malicious behavior, economic incentives should be granted to Final Committee members after reaching an agreement on a valid block in each round.

2.2.7 Complexity Analysis

In this section we analysis the communication complexity and latency of ACP protocol in each round.

In normal case operation, there are following message transmissions in each round of ACP protocol: the message of block proposals broadcasted by selected Final Committee members; the 2 message delay for running Reduction procedure; the 4 message delay for running PBFT; the message of agreed-upon result broadcasted to the whole system. That is, there are 8 message delay in each round of ACP protocol and the message volume is:B_{cb}^{r}

$$N_{pc}^2 + 2N_{fc}^2 + 3N_{fc}^2 N_{fc-valid-leader} + 3N_{pc}^2 N_{fc-empty-leader} + N_{fc} N_{all} \quad (2.3)$$

There are multiple PBFT* instances running in parallel. Each instance has a leader

node respectively. Unless all of the leader nodes running PBFT* are Byzantine nodes, causing an extra waiting periods *SBR*, the protocol can completes at the end of any PBFT* instance process.

2.2.8 Evaluation

We provide our estimation of ABC throughput and latency in this section. We start with some assumptions:

1. 1000 ms is needed to calculate the hash values of all nodes in the system.
2. 500 ms is needed to broadcasting a message to Potential Committee with 512 nodes.
3. 200 ms is needed to broadcasting a message to Final Committee with 16 nodes.
4. 3000 ms is needed to broadcast a message to the whole system with 100, 000 nodes.
5. 200 ms is needed to a message transmission between two nodes.

The agreement time of ACP protocol can be estimated as below:

$$1000 + 500 + 6 * 200 + 3000 = 5700\text{ms} = 5.7\text{s}.$$

The throughput and latency of ACP can be estimated as below:

	Block size	With a 3s agreement time (tps)	With a 5.7s agreement time (tps)
Bitcoin	4M	1028	541
	8M	2056	1082
Ethereum	4M	2608.4	1372.8
	8M	5216.8	2745.6

In order to estimate the most suitable hop number for specific network setting, we estimate the block time under various setting by following equation:

$$\left(\frac{8 \cdot BS_{Block-size} \cdot \sqrt[h]{N}}{B_{bandwidth}} + \frac{RTT}{2}\right) \cdot h \leq T_{block} \quad T_{block} \tag{2.4}$$

Where N is the numbers of whole network; $B_{bandwidth}$ is the bandwidth; RTT is the round-trip time; h is the hop number; $BS_{block\ size}$ is the block size; T_{block} is the block time. The equation is deduced following steps below:

1. $\sqrt[h]{N}$ is the number of receiving end to which each node has to send at each hop.

2. $BS_{Block-size} \cdot \sqrt[h]{N}$ is the amount of data to be sent at each hop. $\frac{8 \cdot BS_{Block-size} \cdot \sqrt[h]{N}}{B_{bandwidth}}$ is the time required for sending all data at each hop.

3. Per-hop time equals to the sum of network transmission time and the time required for sending all data locally.

4. The block time equals to the multiplication of the number of hops and per-hop time. We list the estimated block time under various setting below:

RTT=50ms		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=100	h=2	12.85	4.32	2.61	1.65	1.33
	h=3	8.99	3.05	1.86	1.19	0.97
	h=4	8.20	*2.80*	*1.72*	*1.11*	*0.91*
	h=5	*8.16*	2.80	1.73	1.13	0.93
	h=6	8.42	2.91	1.80	1.18	0.98
	h=7	8.82	3.06	1.90	1.26	1.04
	h=8	9.30	3.23	2.02	1.34	1.11
		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=1000	h=2	40.53	13.54	8.15	5.11	4.10
	h=3	19.28	6.48	3.92	2.48	2.00
	h=4	14.50	4.90	2.98	1.90	1.54
	h=5	12.86	4.37	2.67	1.72	1.40
	h=6	12.29	*4.20*	*2.58*	*1.67*	*1.36*
	h=7	*12.19*	4.18	2.58	1.68	1.38
	h=8	12.34	4.25	2.63	1.72	1.41
		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=10000	h=2	128.05	42.72	25.65	16.05	12.85
	h=3	41.44	13.86	8.35	5.25	4.21
	h=4	25.70	8.63	5.22	3.30	2.66
	h=5	20.32	6.86	4.16	2.65	2.14
	h=6	17.97	6.09	3.71	2.38	1.93
	h=7	16.87	5.74	3.51	2.26	1.84
	h=8	*16.39*	*5.60*	*3.44*	*2.22*	*1.82*

RTT=100ms		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=100	h=2	12.90	4.37	2.66	1.70	1.38
	h=3	9.06	3.12	1.93	1.26	1.04
	h=4	8.30	*2.90*	*1.82*	*1.21*	*1.01*
	h=5	*8.29*	2.93	1.86	1.25	1.05
	h=6	8.57	3.06	1.95	1.33	1.13
	h=7	9.00	3.23	2.08	1.43	1.21
	h=8	9.50	3.43	2.22	1.54	1.31
		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=1000	h=2	40.58	13.59	8.20	5.16	4.15
	h=3	19.35	6.55	3.99	2.55	2.07
	h=4	14.60	5.00	3.08	2.00	1.64
	h=5	12.99	4.50	2.80	1.84	1.52
	h=6	12.44	*4.35*	*2.73*	*1.82*	*1.51*
	h=7	*12.37*	4.36	2.75	1.85	1.55
	h=8	12.54	4.45	2.83	1.92	1.61
		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=10000	h=2	128.10	42.77	25.70	16.10	12.90
	h=3	41.52	13.94	8.42	5.32	4.29
	h=4	25.80	8.73	5.32	3.40	2.76
	h=5	20.44	6.98	4.29	2.77	2.27
	h=6	18.12	6.24	3.86	2.53	2.08
	h=7	17.05	5.92	3.69	2.44	2.02
	h=8	*16.59*	*5.80*	*3.64*	*2.42*	*2.02*

RTT=150ms		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=100	h=2	12.95	4.42	2.71	1.75	1.43
	h=3	9.14	3.20	2.01	1.34	1.12
	h=4	8.40	*3.00*	*1.92*	*1.31*	*1.11*
	h=5	*8.41*	3.05	1.98	1.38	1.18
	h=6	8.72	3.21	2.10	1.48	1.28
	h=7	9.17	3.41	2.25	1.61	1.39
	h=8	9.70	3.63	2.42	1.74	1.51
		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=1000	h=2	40.63	13.64	8.25	5.21	4.20
	h=3	19.43	6.63	4.07	2.63	2.15
	h=4	14.70	5.10	3.18	2.10	1.74
	h=5	13.11	4.62	2.92	1.97	1.65
	h=6	12.59	*4.50*	*2.88*	*1.97*	*1.66*
	h=7	*12.54*	4.53	2.93	2.03	1.73
	h=8	12.74	4.65	3.03	2.12	1.81
		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=10000	h=2	128.15	42.82	25.75	16.15	12.95
	h=3	41.59	14.01	8.50	5.40	4.36
	h=4	25.90	8.83	5.42	3.50	2.86
	h=5	20.57	7.11	4.41	2.90	2.39
	h=6	18.27	6.39	4.01	2.68	2.23
	h=7	17.22	6.09	3.86	2.61	2.19
	h=8	*16.79*	*6.00*	*3.84*	*2.62*	*2.22*

RTT=200ms		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=100	h=2	13.00	4.47	2.76	1.80	1.48
	h=3	9.21	3.27	2.08	*1.41*	*1.19*
	h=4	*8.50*	*3.10*	*2.02*	*1.41*	1.21
	h=5	8.54	3.18	2.11	1.50	1.30
	h=6	8.87	3.36	2.25	1.63	1.43
	h=7	9.35	3.58	2.43	1.78	1.56
	h=8	9.90	3.83	2.62	1.94	1.71
		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=1000	h=2	40.68	13.69	8.30	5.26	4.25
	h=3	19.50	6.70	4.14	2.70	2.22
	h=4	14.80	5.20	3.28	2.20	1.84
	h=5	13.24	4.75	3.05	*2.09*	*1.77*
	h=6	12.74	4.65	*3.03*	2.12	1.81
	h=7	*12.72*	*4.71*	3.10	2.20	1.90
	h=8	12.94	4.85	3.23	2.32	2.01
		$B_{bandwidth}$=100Mbps	$B_{bandwidth}$=300Mbps	$B_{bandwidth}$=500Mbps	$B_{bandwidth}$=800Mbps	$B_{bandwidth}$=1000Mbps
N=10000	h=2	128.20	42.87	25.80	16.20	13.00
	h=3	41.67	14.09	8.57	5.47	4.44
	h=4	26.00	8.93	5.52	3.60	2.96
	h=5	20.69	7.23	4.54	3.02	2.52
	h=6	18.42	6.54	4.16	2.83	2.38
	h=7	17.40	6.27	4.04	*2.79*	*2.37*
	h=8	*16.99*	*6.20*	*4.04*	2.82	2.42

2.3 Conclusion and Future Work

In this chapter, we have described the limitation of existing blockchain consensus and proposed a novel blockchain consensus protocol ACP with economic

incentive compatible. The ACP consensus protocol offers the ability of instant transaction confirmation and the state-of-the-art throughput performance, while preserving high security and scalability and maintaining decentralization. In future, we plan to upgrade the protocol in following aspects:

1. Design a novel transfer mechanism that message and block are transferred sepa- rately. Further, we will combine the push/pull and multi-level node model in the transmission process to improve network transmission performance.
2. Employ threshold-signature technology in the last stage of ACP to accelerate the process of achieving agreement.
3. Conduct study on dynamic replacement in Final Committee to improve the security of ACP.

3 ABC Blockchain Native Storage – ABS

In this chapter, we will introduce a storage solution ABS (ABC Blockchain Native Storage) in the ABC system optimized for blockchain applications. We present the architecture of our solution in Section 3.1 and challenges of blockchain storage in Section
3.2. Then we will discuss in detail our implementation in Section 3.3.

3.1 Architecture

From the technical perspective, the blockchain is a large distributed storage system with Byzantine fault tolerance. However, the emerging blockchain applications expose unprecedented challenges to traditional distributed storage, due to its unique data characteristics and application models. To address the challenges and optimize the distributed storage, we look into two layers, i.e., architecture layer and storage engine layer. From the view of architecture, the optimization idea is relatively simple, similar to common approaches used in other system designs: extract and move storage related operations that cause high overhead out from the system critical path. This is also a common approach for handling data storage in mainstream blockchain application frameworks, and is called off-chain storage. From the view of storage engine, we propose the design of ABC Blockchain Native Storage (ABS), based on our deep understanding of blockchain storage that combines cutting-edge research achievements and practices.

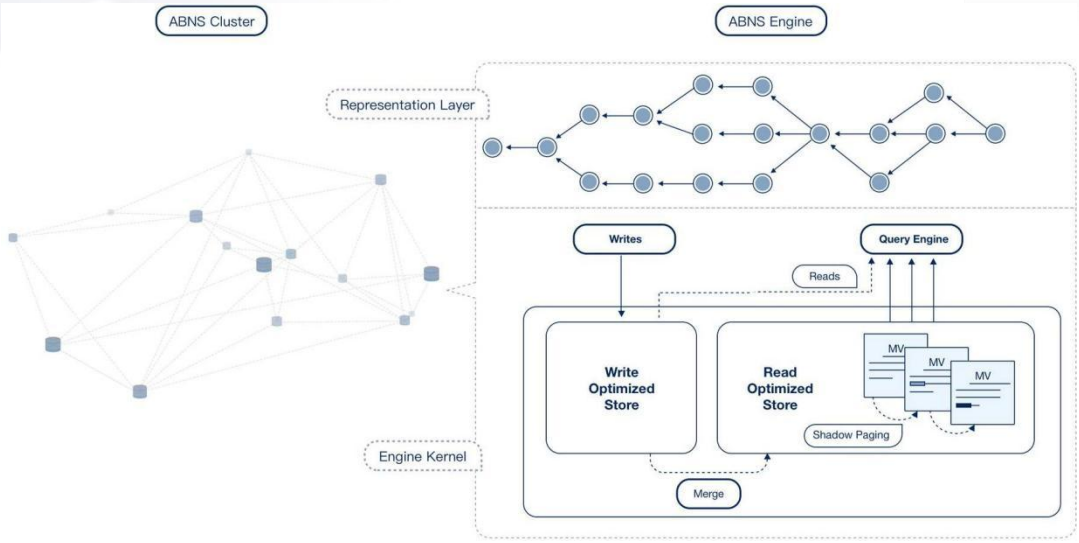

Figure 3.1: ABS Architecture

As shown in Figure 3.1, the ABS cluster on the left represents the entire storage cluster. It is a decentralized storage system consisting of multiple nodes, each of which is an ABS Engine. On the right, we illustrate the internal design of the ABS Engine, composed of two layers: the bottom layer (i.e., Engine Kernel) is a high-performance, high-concurrency storage engine implemented with cutting-edge database and distributed system technologies; the upper layer (i.e., Representation Layer) achieves many blockchain-friendly features, such as multi-versioning, forking and immutability.

3.2 Challenges and Requirements for Blockchain Storage

3.2.1 Challenges for Traditional Storage Systems

First, the data stored in the blockchain system is very special compared to data in other systems. Currently, the mainstream blockchain systems store data as key-value pairs, where the key is usually the hash of the value part (also known as data fingerprint or digest). The keys stored in the key-value stores are therefore scattered in the entire key space. This trait poses a huge challenge for LSM-tree-based [50] key-value storages, because the insertion of a new key-value pair

may result in huge amounts of sorting workloads. Since "real-time global ordering" is one of the core features in the storage system, these sorting workloads cannot be avoided, though might be delayed to a certain extent. Moreover, as a single transaction usually involves reading and writing multiple key-value pairs, the looseness in the key distribution will cause requested keys to be scattered over different locations in the underlying physical storage, i.e., with poor data locality. This phenomenon brings about a huge number of random reads and writes, which leaves a heavy burden on the design of internal cache policy in the storage.

Second, the blockchain system usually needs to maintain different types of data, including blocks, transaction details and global states. The blockchain has special access patterns for these stored data. For example, the blockchain data access follows such a pattern that has high read ratio, low write ratio and writes cannot be blocked by reads. Therefore, the conventional optimizations in databases have no obvious improvement for data accesses in the blockchain.

Last but not least, due to the emergence of blockchain-based applications, the analytical queries over blockchain data become increasingly demanding. However, the widely used LSM-tree-based key-value stores have very limited functionalities and performance for such analytical queries, e.g., the lack of concurrent query or range query capabilities.

Besides the data storage challenges brought by the blockchain, there are also other limitations from the mainstream storage engine themselves. For example, LevelDB [51] is one of the best stand-alone storage engines today, but it still has following limitations or disadvantages:

- Almost all data files need be rewritten when conducting the full compaction, and the storage requires at least double space compared to the actual data size;
- The compaction is an extremely time-consuming operation and cannot be aborted, as re-execution of an aborted compaction needs to start from scratch;

- The read/write amplification effect is very severe, which directly degrades the performance of normal read and write operations.

According to relevant research findings [52]: In the experimental setting that a LevelDB manages records with 16-byte keys and 1K-byte values, the write and read amplifications are about 14 and 327 times respectively when the data reaches 100GB.

3.3 Storage Requirements for Data Models

In the previous section, we discussed the challenges faced by storage systems for handling blockchain data characteristics. In this section, we discuss the blockchain storage requirements from the data model perspective.

3.3.1 Multi-Versioning with Traceability

The blockchain data model consists of two core concepts (i.e., blockchain = block + chain), which together form a weakly centralized "global state" as "blockchain". A "Block" records a specific change in the "global state", and the "chain" guarantees the traceability of the "global state" changes. From a macro perspective, the blockchain and the traditional storage can be conceptually mapped: a block contains multiple operations (i.e., transactions) that change the system status. In fact, each operation can also be packed as a single-record block. Packing multiple operations as a block is actually a specific optimization. This can be viewed as batch operations or group submissions in traditional storages. In addition, the blockchain generates a new version after a transaction is committed in the new block, which is similar to the transaction processing in traditional storages. However, one major difference is that most storages only maintain the latest version of data, while the blockchain maintains all historical transactions that are visible and verifiable from the user side.

In summary, when designing the storage engine for a blockchain system, we need to take the particularity of the data model into consideration. It is desirable to support such data models natively in the underlying storage engine, which

might lead to an optimal technical solution. If blockchain data is maintained by a generic storage engine, it requires additional data transformation and adaptation at the application layer, apart from the schema design and storage engine.

3.3.2 Multi-branched Data Versioning

In the blockchain, there is a very important module called the consensus protocol, which is also called consistency protocol in the traditional distributed storages. Since the Byzantine problem is not considered in traditional storages, the mainstream strong consistency protocols are Paxos [53], Raft [54], Viewstamped [55], etc. In contrast, Byzantine fault tolerance is a necessity in blockchain systems, and their consistency protocols are mainly categorized into three classes as follows:

Category	Basic Principle	Typical Implem entation
Algorithm Class	Achieve consensus by passing messages between nodes and verifying constraints	PBFT [6], Tangaroa [56], etc
Engineering Class	Achieve consensus from an engineering or sociological perspective, using economic game theory and complex computations	PoW, PoS, DPoS, etc.
Synthesis Class	Achieve consensus using synthesis of algorithm, cryptography, hardware and engineering	Algorand [57], ByzC in [58], Chainspace [59],etc.

The "Algorithm Class" consistency protocols are rigorous and have been well proven with mathematical derivation. The "Synthesis Class" consistency protocols are similar to those from "Algorithm Class". They are mainly designed to address the scalability issue in "Algorithm Class" protocols, and all provide strong consistency. On the contrary, the "Engineering Class" protocols mainly provide eventual consistency, proven by existing theoretical analysis. For example, it is

possible for two miners to simultaneously find the hash values meeting the requirements, hence the blockchain might be temporary forked in normal PoW protocols. Consequently, as a distributed storage, the blockchain has a unique property that its historical states might not be linear, i.e., forks may occur. Such scenarios seldom occur in traditional strong storages.

3.3.3 Easy Detection of Historical Data Tampering

Another characteristic of the blockchain is that the data cannot be tampered (or, more accurately, is extremely expensive to tamper). Therefore, for a blockchain-based application, the most critical problem is how to quickly identify whether the blockchain has been tampered with or not. From a technical perspective, nothing is impossible to tamper with. However, if tampering can be quickly detected and participants no longer trust the affected data, the blockchain becomes tamper-proof from the engineering point of view. Consequently, for a blockchain-friendly storage engine, it must have the ability to detect data tampering efficiently and timely.

3.3.4 Append-Only Data Property

Since the blockchain needs to maintain the entire evolution history of the global state, it is different from the conventional storages. A blockchain-friendly storage engine cannot conduct in-place-update on a historical version, instead it needs to transform each modification into a new version (i.e., new state). For example, a user executes following commands in Redis [60]: *set abc 'x'; set abc 'y'*. For this modification sequence, the response of querying *abc* will be *abc = 'y'*, after *set abc 'y'* command has been successfully executed. The user does not know that the previous version is *x*, and the provenance information that *y* is derived from *x*. However, in the blockchain, any transaction must be traceable, hence we need to maintain not only the latest state of the data, but also the entire evolution history of the state. The historical data is not only used to verify the validity of blocks and transactions, but also opens up the opportunity of supporting AI applications and analytical queries in the future.

In summary, we argue that the current mainstream storage engines do not have native Blockchain-oriented properties. In addition, due to the rapid development and industrialization of AI technology, we believe that in the future, the blockchain appli- cations must be deeply integrated with AI techniques, which further derive advanced blockchain applications that are weakly centralized or fully de-centralized. Moreover, the AI applications require efficient retrieval of various data stored in the blockchain, which forces the underlying storage engines to have efficient retrieval capabilities. In other words, the storage engines require multi-dimensional query functionalities to reflect and fulfill the uniqueness of the blockchain. Therefore, blockchain backend technology is the base for large-scale popularization of blockchain-based applications. In particular, under the premise of maintaining the main chain with blockchain characteristics, the improvement of scalability, robustness and performance of the main chain will become the core competitive strength of blockchain backend technology.

3.4 ABS Technical Solution

ABS is a blockchain native storage system with the complete support of ACID [73] properties. It is inspired by the design of *delta-main update* [61] concept, and provides excellent write performance while perfectly satisfying the read-intensive nature of the blockchain system. ABS adopts the MVCC (Multi-Version Concurrency Control) [62] mechanism based on the shadow paging [63, 72] technology, which implements latch- free write/read and latch-free read/read, and further improves the system performance via zero-copy read operations. In addition, ABS fully integrates widely used blockchain data models, such as MPT (Merkle Patricia Tree) [64]. For the indexing methods, ABS adopts the unique CoW [65] and speculative inheritance [66] GC approaches, which ensure that the scattered hash keys in the blockchain system still preserve good data locality and cache locality [74]. As far as we know, ABS is one of the few storage engines in current stage that focus on the core ecology for blockchain technology and have deep exploration for native support.

3.4.1 Design Principle

In this section, we briefly introduce the design principles behind the ABS. During the design of a general-purpose storage engine, we limit the possible ways of reading and writing data, once the physical data layout on storage devices (e.g., disk, flash, memory and cache) are designed and finalized. Therefore, in most cases, we have to consider the trade-off among three directions [67, 68], i.e., read optimized, write optimized and space optimized:

As shown in Figure 3.2, optimizations towards any two directions will have negative impact on the third direction. Therefore, the design priority of ABS is as follows: point lookup (read optimized), write, range lookup (read optimized), space optimized. Note that write is not the direction with highest priority, because we believe that the main bottleneck of the blockchain systems will still be in network communication and consensus mechanism for a long period of

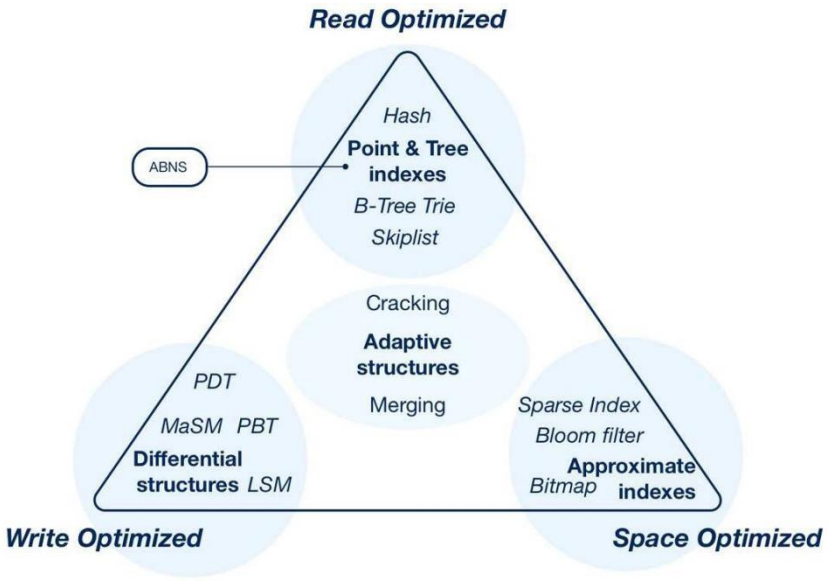

Figure 3.2: ABS Design Trade-off

time from now. Besides, the trade-off between write throughput and latency can be tuned by flexibly configuring the write batch size; moreover, not all data requires real-time synchronous persistency, and we can exploit page cache and memory to alleviate the write pressure of the entire system. The point lookup has the highest priority in our design, because in order to improve transaction

efficiency in the block system, it needs to conduct a large number of key-value queries during the execution of transaction processing and verification. The range lookup is of the third place, because ABS is designed to support rich features and provide various possibilities for analyzing underlying blockchain data in different upper-layer applications. The primary goal is to ensure the high efficiency of blockchain transaction execution and to prevent the storage engine from becoming a bottleneck of the entire system.

After all, the design priority is a relative measurement, and we need to carefully consider the whole picture when designing a specific component. Any obvious limitation or bad design may affect the applicability of the ABS. Therefore, the ranking of design priority does not mean that our design lacks the deep consideration or optimization on low priority directions.

3.4.2 Design and Implementation

The overall design of ABS mainly consists of four layers, from bottom to top respec- tively: Engine Kernel (EK), Blockchain Feature Representation (BFR), Data Access Pattern (DAP) and Semantic Views (SV). EK is our self-developed key-value storage engine, whose design details will be covered in subsequent chapters. BFR is designed for common data structures and features of the blockchain system generalized from the EK design, such as: Merkle tree model abstraction, generic data validation abstraction, etc. All these abstractions can be further extended continuously. DAP is a high-level API, including point query, range query, point write, batch write, etc., which is actually the SDK provided in this layer. For the SV layer at the top, since different blockchain applications have own focuses depending on their specific domain knowledge and models, they need to use the API provided in this layer to customize differentiated data views.

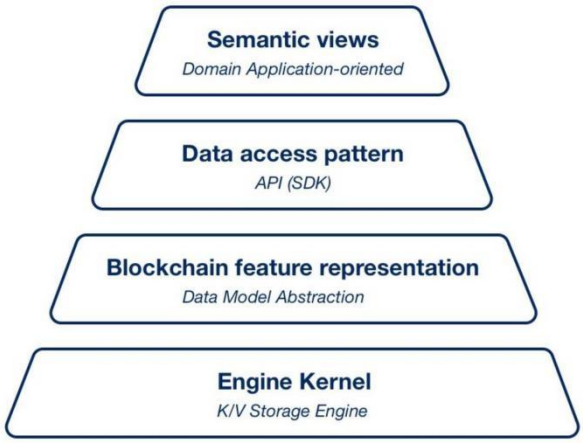

Figure 3.3: Layered Design of ABS

With the layered architecture like UStore [70] and Forkbase [71] shown in **Figure 3.3**, ABS is able to deeply and independently optimize each layer, which offers high horizontal scalability without internal dependencies.

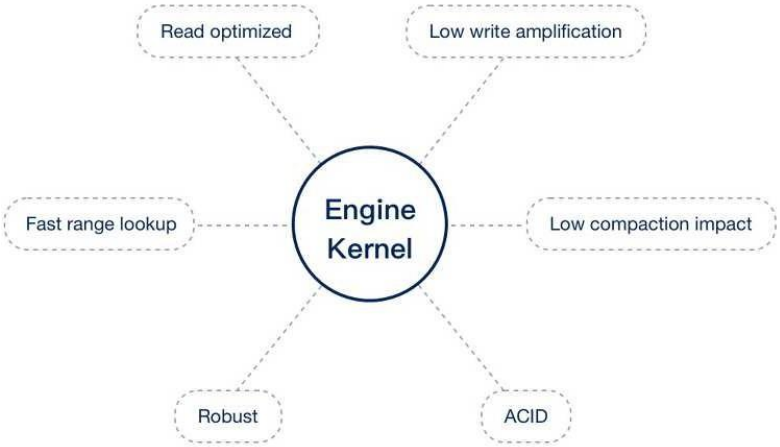

Figure 3.4: Engine Kernel Features

Engine Kernel Layer (EK)

EK is the most critical part of the ABS Engine and the base component of the entire ABS Engine system. First, EK is a generic key-value storage engine, as shown in **Figure 3.4**, which provides rich features: read-optimized, low write amplification, low compaction impact, ACID, robustness, fast range lookup, etc.

The core data structure of EK is an implementation of B+-tree with append-only and shadow paging concepts [72]. The index structure in B+-tree satisfies the requirement of intensive reads and fast range lookups. The conventional append-only approaches lead to very large storage overhead, and bring in large write amplification and compaction impacts. EK resolves the overhead and negative impacts by continuously recycling expired data pages via a unique speculative inheritance GC [66] technology.

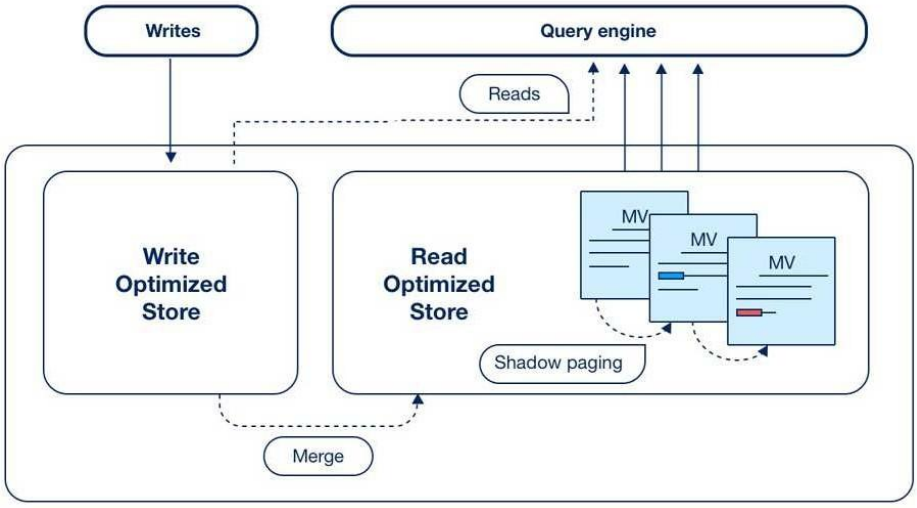

Figure 3.5: Architectural Design of ABS Engine Kernel

As shown in the figure above, EK offers latch-free write/read and latch-free read/read. Within each single partition, EK adopts a single-writer/multi-readers model, while for the entire system, it provides multi-writers/multi-readers model through partitioning. For data update, EK adopts the design of *delta-main update* concept, which lowers the storage cost and improves the capability of concurrent processing. For physical data layout, EK adopts row-oriented layout, which provides superior point get/put operations and is more suitable for blockchain data access patterns. For data version management, by applying the shadow paging technology, EK organizes different versions of the same key as a chained sequence, which is friendly for conducting GC on old versions.

Blockchain Feature Representation Layer (BFR)

Since append-only and multi-version features are already included in the underlying EK design, these features can be directly utilized at the Blockchain Feature Representation (BFR) layer. In our design, two most important abstractions in this layer are: Merkle Tree Abstraction and Fast Validation Abstraction. Merkle Tree plays a very important role in the blockchain technology. Both Bitcoin and Ethereum adopt Merkle Tree imple- mentations and optimizations, which are however heavily coupled with other modules. In addition, the underlying storage engines (e.g., LevelDB [51] and RocksDB [69]) do not natively support these features, i.e., append-only and multi-version. Therefore, in state- of-the-art blockchain systems, the support of these features has low efficiency and high complexity during the implementation.

In fact, For the Merkle Proof, its basic idea is similar to shadow paging, both of which aim to involve as less storage and computation related to the branch change as possible during the procedure. Hence, apart from the adoption of append-only and shadow paging in the B+-tree, we also consider the possible ultimate blockchain storage model in the long run [75]. Possibly, the most ideal approach is not the key-value model, but a more blockchain native model. That is why Blockchain Feature Representation layer needs to provide functionalities such as existence checking of a specific key and verification of the associated value.

Data Access Pattern Layer (DAP)

Data Access Pattern layer provides SDK for upper-layer application developers. The provided APIs in the SDK can be flexibly extended according to different application scenarios and requirements. The currently implemented APIs mainly include:

Put(key, base version, value) -> {version} Get(key, version) -> {value} GetPrevious(key, version) -> {values}

Semantic Views Layer (SV)

In Semantic Views layer, different blockchain applications focus on different aspects based on their domain knowledge and expert models. They can utilize the APIs provided in SV to achieve application-specific differentiated data views. The common data views include: fine-grained access control, data security, subscription/publishing of data updates, etc.

3.5 Conclusion and Future Work

In this chapter, we have outlined the ABS, an advanced blockchain native distributed storage system. ABS combines the cutting edge database technology with distributed system technology with focus on adapting blockchain feature in the design of ABS. It perfectly addresses the performance, security, capacity, query processing and scalability issues faced by blockchain storage. In future, taking the intrinsic problem of blockchain, i.e., scalability and capacity into consideration, ABS will focus on the following research directions:

1. ABS rebuilds genesis blocks by epoch, where the epoch interval is configurable
2. ABS stores complete blockchain data through technology of erasure coding, and any block can be reconstructed from other nodes to achieve low-consumption storage.
3. By means of rebuilding and erasure coding technique, ABS can purge cold-data safely, such that the scalability and capacity problem in blockchain is solved perfectly.

4 Service and Application

Here we mainly describe the service framework and application design based on ABC platform. In the second half of this chapter, we will use ABC-IM as a typical service case to introduce the specific content of services and applications.

4.1 Service Architecture

As a decentralized basic service platform, we summarize the service architecture of ABC as follows:

1. Decentralized application support platform based on Blockchain DNS, Open ID.
2. Above the basic support platform, it is the abstract resource layer of the system, including resources such as storage, bandwidth and computing power etc.
3. Applications are supported on top of the abstract resource layer.

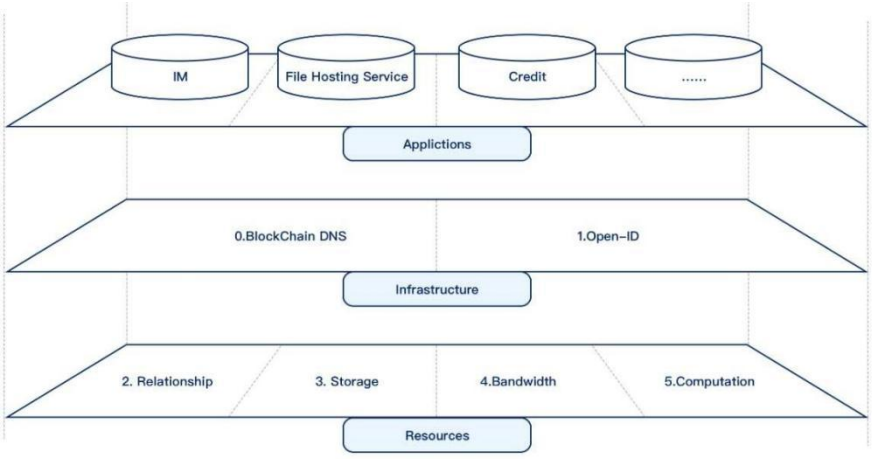

Figure 4.1: ABC Service Architecture

When a client requests a service from ABC system, whether it is storing files or hosting services, or even relying on nodes on the network to complete

computing tasks, it is essentially using three resources on the ABC system including storage, bandwidth and computing. Therefore, we abstract these three resources and provide the OpenID service and decentralized DNS service to enable a uniform resource identifier and user-friendly domain name mapping. We describe these two function provided to decentralized application by ABC system bellows.

4.2 Decentralized Application Support Platform

4.2.1 Blockchain DNS

It is an important aspect in a decentralized system about how to communicate. Earlier we mentioned that the Worker Ring can find the corresponding resource through the key. Meanwhile, the key is a 256-bit hash value which is unfriendly for users to use this public key (or its base58 or base64 encoding string) as an identifier to identify users. In order to address this problem, ABC provides a fully decentralized DNS service at the support layer. BC-DNS (Blockchain-DNS) is a blockchain-based domain name service that comes with the system. The domain name in the BC-DNS domain name resolution service is not limited to the particular formatting rule ending in a fixed format such as. com or .org. It provides a user-friendly, readable, variable-length custom identifier (domain name) to the system account as well as the mapping between this identifier and its corresponding account.

DNS Quick Query based on Manager Ring Servers

It is relatively slow to query the BC-DNS domain name mapping by retrieving persistent storage through the get method of the BC-DNS smart contract. The system needs to provide a faster method to response BC-DNS domain name queries. In this case, we build a distributed in-memory BC-DNS database on the Manager Ring servers. Each Manager Ring server contains all the domain name mapping data of the fragment that it is currently responsible for. By providing a paid (or free) fast domain name retrieval service through the Manager Ring servers, the system nodes can quickly retrieve the BC-DNS domain name

mapping through RPC or other means.

DNS Query Process

It is relatively slow to query the BC-DNS domain name mapping by retrieving persistent storage through the get method of the BC-DNS smart contract. The system needs to provide a faster method to response BC-DNS domain name queries. Therefore, we build a distributed in-memory BC-DNS database on the Manager Ring servers. Each Manager Ring server contains all the domain name mapping data of its corresponding shard. By providing a paid (or free) fast domain name retrieval service through the Manager Ring servers, the system nodes can quickly retrieve the BC-DNS domain name mapping through RPC or other means.

Assuming that the BC-DNS domain name query servers have completed the above service statement and are available, meanwhile, the client has obtained a list of servers that support the BC- DNS domain name query service. Below is a typical domain name retrieval process:

1. The client retrieves the local BC-DNS cache. If there is an entry corresponding to the target domain name , exits and completes the query. Otherwise go to step 2.
2. The BC-DNS dedicated server queries the cached local BC-DNS domain name mappings. If there is an entry corresponding to the target domain name, exits and completes the query. Otherwise, go to step 3.
3. BC-DNS dedicated server initiates BC-DNS domain name retrieval to the Manager Ring server. If the corresponding entry is found, it will be cached into BC-DNS dedicated server. Then go to step 4.
4. If the valid domain name mapping is obtained in step 3, it will be returned to the client. If there is no such domain name mapping, the "domain name does not exist" error is returned to the client.

Validity Period of BC-DNS Domain name Mapping Cache

Theoretically, the modification of domain name mapping data is very rare, but we still need a mechanism to handle BC-DNS domain name mapping updates in BC-DNS domain name query services.

Firstly, we set a timeout period for each domain name mapping on the BC-DNS domain name query server. For all queries, if the corresponding domain name mapping does not exceed the timeout period, it is considered valid and the mapping value is returned as the query result.

Secondly, the BC-DNS domain name query server must set the timeout period for returned results based on the current validity period of the domain name mapping. After receiving the result, the client considers that the domain name mapping is valid within the timeout period. Otherwise, delete the corresponding cache on the client and the client initiates a new BC-DNS domain name query request.

Based on the BC-DNS domain name resolution service, we can build a bunch of applications. We can treat the corresponding nodes IP address which appears in the URL https://yourdomain/index.html as the server IP by using BC-DNS domain name resolution service.

4.2.2 Open ID

The abstraction of accounts and resources in the ABC system can ultimately be mapped to the same address space and obtain a unique identifier. This is the OpenID, which is essentially a 256-bit string. The address space of ABC OpenID is about 2 to the power of 256, which is about 1.16 times 10 to the power of 77. This address space is large enough to ignore the problem of index conflict. The OpenID of each resource is unique in the ABC system. It is also an index. With OpenID, users can easily and quickly find various resource in ABC system including but not limited to an account, a file (digital fingerprint), as well as a service (address mapping), or even a node server (inter-node communication). For example, there is a service deployed in the ABC system. According to the characteristics of the service, the system finally assigns it an index of 'hlkGDG34fjglk-2kdC0djfTlkaDsAdaDfdQs', which is a kind of OpenID. With this OpenID, users can find and get the corresponding service easily.

4.3 Typical Case: CyberLand-IM as a Service

ABC-IM is an instant messaging application based on ABC blockchain platform. With unique decentralized distributed technology, it can provide highly secure service to ensure free global chat with perfect forward secrecy (PFS). It also provides built-in wallet function which makes sending red envelopes and transferring money as easy and fast as chat.

The main features of ABC -IM are:

- Global decentralized server which does not require VPN to guarantee communica- tion quality in various places.
- Support end-to-end encryption scheme to protect privacy.
- Support for sending and receiving most mainstream cryptocurrencies.
- Support Moments, Paid Group, Official Account and other functions.

4.3.1 Open ID-based IM Account System

In the ABC -IM instant messaging application, the mapping between the basic support platform user account key value and user-readable, friendly nickname (domain name in BC-DNS) is established through the OpenID system (see Section 4.2.1 BC-DNS). It enables users to build custom, recognizable personalized nickname s on the OpenID system. For the specific process of nickname registration and modification based on the open account system, please refer to Section 4.2.1 BC-DNS.

4.3.2 IM Backend as Decentralized Service

Based on the decentralized application support platform (see Chapter 4.2 Decentralized Application Support Platform), we treat the IM backend service as a special application on the support platform. The IM backend server joins the Worker Ring DHT and registers the IM service with the ABC blockchain platform. In this way, in the Worker Ring DHT network, each IM backend server is functionally independent on the IM backend service, and can provide services

independently. At the same time, through the DHT network interconnection, a decentralized ABC-IM background service group is formed.

4.3.3 IM Client Access Method

By querying from the ABC blockchain platform for the list of available servers support- ing the IM background service, the IM client selects an available IM server to access the ABC-IM network. Meanwhile, it ensures the long connection between the client and the current IM server to complete the accession of an IM client. It should be noted that when the IM client obtains the IM backend server list from the ABC blockchain platform, the client caches it in the local storage. Therefore, when the client is off and restarts, it can get the last active server address and initiate a connection to it. Moreover, when the currently active IM backend server is unavailable, another IM backend server can be selected to ensure the reliability of the background service.

5 ABC 's Security Technology Scheme

ABC system is based on mature cryptography technology and components. It uses the best security practice technologies which are widely used in industry to protect users' security and privacy. ABC adopts end-to-end encryption technology based on Signal protocol to protect users' communication content from being eavesdropped by any third party (including hackers, communication operators, national teams, etc.). ABC also uses zero-knowledge proof and Tor-based anonymous network communication technology to prevent network traceability, thus protecting users' privacy. ABC also employs quantum resistant encryption algorithm technology. With the coming of quantum computing, our encryption algorithm will still be able to withstand attacks from quantum computers. This section will describe three parts corresponding security techniques in details.

5.1 End-to-End Encrypted Communication

ABC uses the Signal protocol to encrypt the transmission channel between the two clients. In this case, any third party, hackers, communication operators, national teams, including ABC development team, cannot view the communication content. The security of the user's message content is not only guaranteed by the ethic of these third parties, but also by technical mechanisms. ABC applies Signal protocol in both end-to-end communication and group communication, which ensures the security of transmission of message, picture, audio, video and other files. The ABC system also provides forward security and backward security. Even if the key of a message is leaked, the hacker still cannot decrypt the previous and subsequent messages.

5.1.1 Transmitting Media and Other Attachments

Large attachments of any type (video, audio, images, or files) are also end-to-end encrypted:

1. The sender (ABC user who sent a message) generates an ephemeral 32 byte AES256 key, and an ephemeral 32 byte HMAC-SHA256 key.
2. The sender encrypts the attachment with the AES256 key in CBC mode with a random IV, then appends a MAC of the ciphertext using HMAC-SHA256.
3. The sender uploads the encrypted attachment to a blob store.
4. The sender transmits a normal encrypted message to the recipient that contains the encryption key, the HMAC key, a SHA256 hash, and a pointer to the encrypted message in the blob store.
5. The recipient decrypts the message, retrieves the encrypted blob from the blob store, verifies the SHA256 hash of it, verifies the MAC, and decrypts the plaintext.

5.1.2 Group Message Security Communication Technology

Traditional unencrypted messenger apps typically employ "server-side fan-out" for group messages. When a user sends a message to a group, the server distributes the message to each group member. And "client-side fan-out" is the client sends a message to each group member. ABC 's group message is build on the pairwise encrypted sessions outlined above to achieve efficient server-side fan-out for most messages sent to groups. This is accomplished using the "Sender Keys" component of the Signal Messaging Protocol.

The first time a ABC group member sends a message to a group:

1. The sender generates a random 32-byte Chain Key
2. The sender generates a random Curve25519 Signature Key key pair.
3. The sender combines the 32-byte Chain Key and the public key from the Signature Key into a Sender Key message.
4. The sender individually encrypts the Sender Key to each member of the group, using the pairwise messaging protocol explained previously.

For all subsequent messages to the group:

1. The sender derives a Message Key from the Chain Key, and updates the Chain Key.
2. The sender encrypts the message using AES256 in CbC mode.
3. The sender signs the ciphertext using the Signature Key.
4. The sender transmits the single ciphertext message to the server, which does server- side fan-out to all group participants.

The "hash ratchet" of the message sender's Chain Key provides forward security. Whenever a group member leaves, all group participants clear their Sender Key and start over.

5.1.3 Verifying Keys

ABC users additionally have the option to verify the keys of the other users with whom they are communicating so that they are able to confirm that an unauthorized third party (or ABC) has not initiated a man-in-the-middle attack. This can be done by scanning a QR code, or by comparing a 60-digit number. The QR code contains: a version, the user identifier for both parties and the full 32-byte public Identity Key for both parties. When either user scans the other's QR code, the keys are compared to ensure that what is in the QR code matches the Identity Key as retrieved from the server. The 60-digit number is computed by concatenating the two 30-digit numeric fingerprints for each user's Identity Key. To calculate a 30-digit numeric fingerprint: Iteratively SHA-512 hash the public Identity Key and user identifier 5200 times. Take the first 30 bytes of the final hash output. Split the 30-byte result into six 5-byte chunks. Convert each 5-byte chunk into 5 digits by interpreting each 5-byte chunk as a big-endian unsigned integer and reducing it modulo 100,000. Concatenate the 6 groups of 5 digits into 30 digits.

5.2 Privacy Protection

For ABC, protecting the privacy of users is always at the top of the list. We will protect the privacy of our users and prevent them from being traced back to the

network from both business logic and network aspects.

5.2.1 Privacy Protection Technology at The Business Logic Level

Bitcoin is designed as an anonymous currency. However, Bitcoin actually cannot achieve true anonymity (pseudonymity). Because the Blockchain is open and accessible to everyone. User identity information can be associated with certain addresses through techniques such as block browsers and data mining. If a merchant that supports Bitcoin transactions publishes its own Bitcoin address, by using the block browser, one can easily find the merchant's revenue, capital flow, and transaction details with the customer, etc. There are also many people who post their Bitcoin addresses in forums and blogs to accept donations, etc. This also directly links the address with personal identity, and further related transaction information about the identity is also leaked. Some countries, such as Australia, also require organizations that engage in Bitcoin-related businesses (such as exchanges) to provide KYC (Know Your Customer, understand user's real information) and AML (Anti- Money Laundering) for regulation. ABC adopts Zero-knowledge proof at the business level to protect user privacy and prevent network traceability.

Zero-Knowledge Proof

Zero-knowledge proof is to prove something to others without reveal the information of the specific thing. For example, if Alice wants to prove to Bob that she has the key to the room, but does not want to give the key to Bob, then she can bring something in the room (such as Bob knows that there is a silver iPad in the room) to Bob. And this can indirectly prove that she has the key to the room.

Of course, this example has certain requirements for business scenarios, as well as excessive interaction. The zero knowledge proof we need to achieve is bound to be universal. Thus, we use a non-interactive zero-knowledge proof – zkSNARKs which can hide the input and output address and transaction amount in the transaction details.

In fact, for miners, they don't care how much money a transaction spends, as well as the sender and the recipient. Miners only care about whether the system's

money is conserved. Then they just need to prove the following three questions:

1. The sender's money belongs to the person who initiated the transaction.
2. The money sent by the sender is equal to the money received by the receiver.
3. The sender's money is indeed destroyed after the transaction is over.

zkSNARKs technology can mathematically transform each problem that needs to be proved into a polynomial. In this case, the proof of a problem can be converted to: If you know this polynomial, you can prove the problem with this polynomial. Then, how can we prove we know this polynomial? We need to use blind evaluation of polynomials:

1. The system generates a parABC ter with a random number and publicize it.
2. A uses polynomial parABC ters to compute polynomial results and gives to B.
3. B verifies the correctness of polynomial results.

If the result is proved to be correct by B, then A can prove that he knows a polynomial, and at the same time prove that the problem corresponding to this polynomial is proved. The concrete implementation process of zero knowledge proof has the following steps:

1. Homomorphic Hiding
2. Blind Evaluation of Polynomials
3. From Computations to Polynomials
4. The Pinocchio Protocol

The specific steps are more complicated and involve a lot of mathematical knowledge. We will not give a detailed description here. However, zero-knowledge proof technology is a very important privacy protection technology in our products, and it is also a privacy protection technology that is highly valued in the Blockchain technology.

5.2.2 Network Level Privacy Protection Technology

It is still not enough to protect privacy at the business logic level. Users' IP will often reveal such kind of important information, for example, who the user is,

where they live, which will still lead to the user being traced back to the network. ABC uses Shadowsocks to prevent network traceability to protect user privacy.

Shadowsocks

Shadowsocks is a light weight and high performance socks5 proxy. It is very easy to set up a Shadowsocks. Typically, it just needs to install a client and configure a server. Shadowsocks splits the Socks5 protocol created by the original ssh into server and client. The request from the client communicates with the ss-local terminal based on the Socks5 protocol. Since this ss-local is generally a local machine or a router or other machine on the local area network which the firewall does not review, the problem of block by the firewall through feature analysis will be solved. Meanwhile, both ss-local and ss-server communicate through a variety of optional encryption methods. When the firewall is reviewed, it is a regular TCP packet. There is no obvious feature and the firewall cannot decrypt the data as well. Then, ss-server decrypts the received encrypted data, restores the original request, and then sends it to the service that the user needs to access. Finally, it obtains the response and sends back to the client. For the commonly keyword filtering system, a Shadowsocks plugin should be enough to bypass the censorship and access the free Internet.

5.3 Quantum-Resistant Encryption Algorithm

5.3.1 The Threat of Quantum Computers on Blockchain

Cryptographic algorithms, especially asymmetric cryptography, are the cornerstone of blockchain security. However, with the rapid development of quantum computer technology, this cornerstone is not strong enough anymore.

Currently, popular public key cryptography algorithms are based on three types of problems: (1)large integer decomposition problem (RSA); (2)discrete logarithm problem (El-Gamal, DSA, DH key exchange); (3)discrete logarithm problem based on elliptic curves (ECDSA). The algorithms based on these three types of problems are safe because their computational complexity is very high. Under the current computer power level, there is no way to brute force in a short

time (100 years). However, with the introduction and development of the concept of quantum computers, this situation will change.

In CES2018, Intel demonstrated a 49 qubit processor, which is considered a milestone. On March 5, 2018, Google Quantum A.I. Lab Team announced their latest achievement which is "Bristlecone", a 72 qubit processor.

In the near future, there may be quantum computers that can crack existing encryp- tion algorithms. Therefore, we must also respond to this to protect the security of our Blockchain technology.

5.3.2 Quantum-Resistant Encryption Algorithm Technology – Falcon

Falcon means Fast-Fourier Lattice-based Compact Signature over NTRU, which is a lattice-based digital signature scheme with the properties such as compactness, efficiency and provable security, etc. It has high practical value.

Falcon's construction idea is mainly to instantiate the GPV framework (Gentry, Peikert, Vaikuntanathan proposed in 2008) [81−84] to construct a lattice-based Hash-and-sign digital signature.

Instantiating this framework requires two parABC ters:(1)A class of cryptographic lattices; (2)Select a trapdoor sampler.

For these two parABC ters, Falcon selects the NTRU lattices and Fast Fourier Sampling. So the design of the Falcon scheme can also be simply described by the following formula:

$$Falcon = GPVf\ rABC\ work + NT\ RUlattices + FastFouriersampling \quad (5.1)$$

GPV framework

The GPV framework is used to construct a lattice-based digital signature, which can be described as below:

1. The public key contains a full-rank matrix $A \in Z_q^{nxm}$ (with $m > n$) generating a q-ary lattice Λ.The private key contains a matrix $B \in Z_q^{nxm}$ generating Λ_q^T, where Λ^T denotes the lattice orthogonal to Λ modulo q: for any $x \in \Lambda$ and $y \in \Lambda_q^T$, we have $(x, y) = 0 \bmod q$.Equivalently, the rows of A and B are

pairwise orthogonal:$B \times A^t = 0$.

2. Given a message m, a signature of m is a short value $s \in Z_q^m$ such that $sA^t = H(m)$,where $H : \{0, 1\}* \rightarrow Z_q^n$ is a hash function. Given A, verifying that s is a valid signature is straightforward: it only requires to check that s is indeed short and verifies $sA^t = H(m)$.
3. Computing a valid signature is more delicate. First, an arbitrary preimage $c_0 \in Z_q^m$is computed, which verifies $c_0A^t = c$. As co is not required to be short and $m\ n$, this ≥ can simply be done through standard linear algebra. B is then used in order to compute a vector $v \in \Lambda_q^T$ close to c_0. The difference $s = c_0 - v$ is a valid signature:indeed, $sA^t = c_0A^t - vA^t = c - 0$ = c,and if c_0 and v are close enough, then s is short.

The above description of the framework was not originally proposed by GPV, but was first proposed in GGH and NTRUsign. However, GGH and NTRUSign suffer of total break attacks, whereas the GPV framework is proven to be secure in the classical and quantum random oracle models assuming the hardness of SIS for some parABC ters. The difference between these frameworks is the way of computing v in the signing procedure. GGH and NTRUsign use the Round-off algorithm. The problem with this algorithm is that each signature will reveal some information about the matrix B. On the other hand, the calculation of v in GPV relies on a randomized variant of the nearest plane algorithm. This algorithm proves that it will not reveal information about the B matrix, so it is provably secure.

In addition, in order to prevent the same message from generating different signa-tures, GPV also introduces a fixed length of random salt in the signed message.

NTRU Lattices

After determining the framework, the next step is to instantiate the lattices. The main consideration for Falcon is the compactness. Because the NTRU lattices are chosen, and the performance is excellent in public key size and efficiency.

Let $\varphi \in Z[x]$ be a monic polynomial, and $q \in N$. A set of NTRU secrets consists of four polynomials $f , g , F , G \in Z[x]/(\varphi)$ which verify the NTRU equation: $fG - gF = q \mod \varphi$ Provided that f is invertible modulo q, we can define the polynomial $h \leftarrow g \cdot f^{-1} \mod q$.

Typically, h will be a public key, whereas f , g, F, G will be secret keys. Indeed, one can check that the matrices $[1 \quad h \quad 0 \quad q]$and $[f \quad g \quad F \quad G]$ generate the same lattice, but the first matrix contains two large polynomials (h and q), whereas the second matrix contains only small polynomials, which allows to solve problems mentioned before. The security of NTRU relies on solving two small polynomials f', g' such that $h = g' \cdot (f')^{-1}$ is a hard problem.

Instantiate the GPV framework over NTRU lattices:

1. The public basis is $A = [1 h]$
2. The secret basis is $B = [g \; -f \; G \; -F]$, the matrices A and B are orthogonal: $B \times A = 0 \mod q$.
3. The signature of a message m consists of a salt r plus a pair of polynomials (s_1, s_2), such that $s_1 + s_2 h = H(r \vee m)$. Since s_1 is completely determined by m, r and s_2, the signature can simply be (r, s_2).

Fast Fourier Sampling

When instantiating a GPV framework, we also need to select the Trapdoor Sampler to solve for the vector v. When selecting trapdoor the main measurement is efficiency, and how short the final signature s is, that is, how close is v to c_0. There are four main choices:

1. Klein algorithm which is nearest plane algorithm with randomization. It is superior in security but relatively low in time and space efficiency $O(m^2)$.
2. Peikert's round-off algorithm with randomization optimizes the space and time efficiency, but the security is worse than the Klein algorithm.
3. A highly efficient and simple trapdoor sampling proposed by Micciancio and Peikert, but the compatibility with NTRU is unclear.

4. The nearest planar algorithm similar to Fast Fourier Transform proposed by Ducas and Prest can be added with a randomization method, which is both safe and efficient, and can be used on the NTRU lattices.

After comprehensively comparing the above four schemes, Falcon chose the fourth trapdoor sampler with randomized Fast Fourier nearest plane.

In addition, it is necessary to pay attention to the setting of the parABC ter standard deviation of the trapdoor sampling. If it is too small, the key information (basis of the matrix B) will be leaked. But if it is too large, the generated signature will not be "short" enough. Both of these situations will make the constructed signature scheme unsafe.

ParABC ters Selection of Falcon

According to the security level requirements, the selected parABC ters of Falcon submitted to NIST are:

Level	Dimension n	Polynomial Φ	Modulus q	Acceptance bound β^2
1- AES128	512	$x^n + 1$	12289	43533782
2- SHA256 3- AES192	768	$x^n - x^{n/2} + 1$	18433	100464491
4- SHA384 5- AES256	1024	$x^n + 1$	12289	87067565

In the ABC account system, the parABC ters of the second security level (SHA256, AES192) will be selected. In addition, the hash algorithm that generates the signature message digest uses SHAKE-256.

5.3.3 Cryptographic Algorithm Scheme Considering Both Pre-Quantum and Post- Quantum

The ABC account system will support both ECDSA and Falcon's public key system. Before the quantum computer matures, normal users can use ECDSA as the public and private key of their wallet account. After the quantum computers

become mature, we can use the public and private keys generated by the new post-quantum signature scheme to protect the asset security of the wallet account.

Wallet Address Generation

Both ECDSA and Falcon will use Bitcoin-like wallet address generation process.

1. First is to generate a public key using a private key.
2. The public key generates a public key hash of 20 bytes in length through two Hash operations (SHA256 and RIPEMD160).
3. Performing the SHA256 operation twice on the public key hash, and takes the first 4 bytes of the operation result as the check code of the wallet address.
4. Select a prefix of the wallet address, and append the public key hash and the check code. Encoding it in Base58 to generate the final wallet address.

The specific process is as follows:

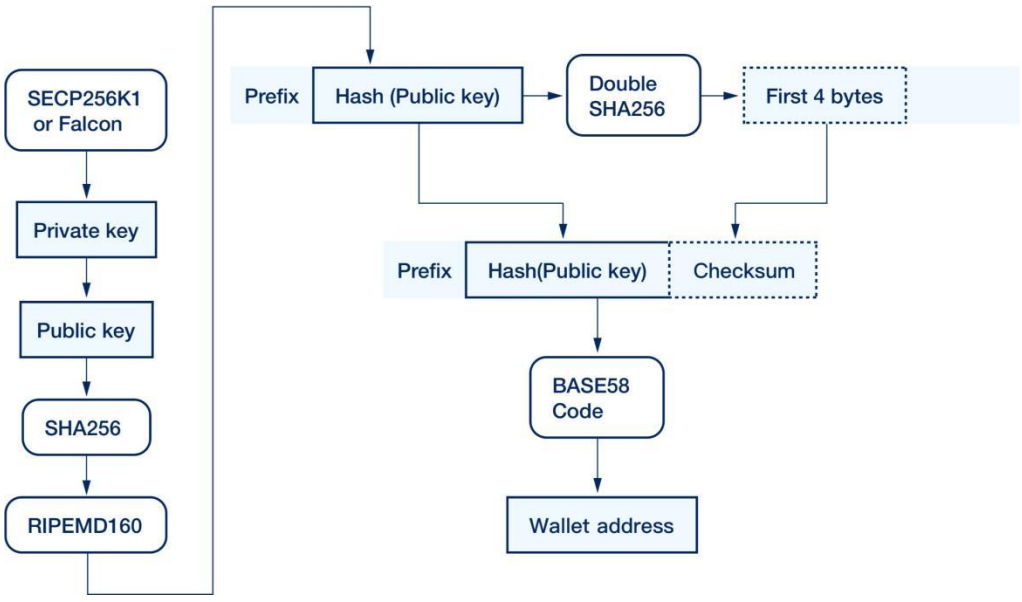

Figure 5.1: Wallet Address Generation Process

In order to distinguish between ECDSA and Falcon's wallet addresses, these two addresses will use different prefixes.

1. The wallet address generated using the ECDSA public key, will use prefix 'A', and the wallet address example is as follows:

   AJA6FuwhMzkriA8mk2zkuKFFb1MvvoCifX

2. The wallet address generated using the Falcon public key, will use prefix 'F', and the wallet address example is as follows:

   FJA6FuwhMzkriA8mk2zkuKFFb1MvvoCifX

Or we can use Bitcoin Cash's wallet format to distinguish two different addresses more clearly. We can use the word "falcon:" as a prefix. An example of a wallet address is:

falcon:JA6FuwhMzkriA8mk2zkuKFFb1MvvoCifX

Transfer Account Assets to Falcon Address

After the emergence of a mature quantum computer, the assets of the user on ECDSA wallet address become unsafe. Hackers can forge legal signatures to spend the balance on an account. In order to cope with this situation, users need to transfer their assets to an anti-quantum account. There are two ways:

1. **Use the RPC command (or curl, etc.)**

   Design an RPC command:

   sendalltofalconaddr [falcon wallet address],

   the ABC client uses this command to iterate over all ECDSA accounts in the local wallet file. And it uses the private key to sign and transfer all account balances which are greater than zero to the falcon address in the command by one transaction. Example is shown below:

   ./ABC -cli sendalltofalconaddr falcon: JA6FuwhMzkriA8mk2zkuKFFb1MvvoCifX You can also use a fine-grained

command to make a split transfer:

sendtoaddress [falcon wallet address] [amount] Example is shown below:

./ABC -cli sendtoaddress falcon:JA6FuwhMzkriA8mk2zkuKFFb1MvvoCifX 1000

2. Use the wallet client

Users can use the "Transfer to Falcon Wallet in One-click" feature button on the full- node wallet or other third-party light-node wallet. Users enter the Falcon Wallet address and transfer all of the user's assets to a secure account. Similarly, Users can also use split transfers.

6 ABC AI Governance System

In this chapter we will introduce the novel artificial intelligent based built-in governance mechanism in ABC, ABC AI Governance system, that we design to mitigate the security issues, maintain the system responsive and resilient to the continuously changing state. We will present the objectives of ABC AI Governance system in Section 6.1 and its architecture in Section 6.2. Then we are going to introduce of modules contained in ABC AI Governance system.

6.1 Objectives

The nature of openness of decentralized blockchains and the dynamical compositions of computing machinery resulting in design inherent weaknesses. Every user can submit transaction request to the network and every computer can join and leave the cluster. This results in inconsistency of the state of blockchain, including users dynamic online activities, the status of each node and the changing network composition, the potentially hostile machines residing within the corporate network. The objectives of ABC AI Governance system is to adapt to the above characteristics of blockchain. We list the main objectives of ABC AI Governance system below.

- Establish a monitoring infrastructure to constantly monitor the behavior of the network and automatically detect anomalies or failures affecting the network to avoid wide spread damage;
- Assign users jobs to nodes according to their historical performance to maintain ABC robust and resilient to changing workload;
- Control the admission, elimination and performance evaluation of nodes in ABC network to guard system against adversary participants;

To realize all of the objectives is hard due to multiple challenges stemming from varying state of the composition of blockchain network. In particular, a key

challenge is to design networks and distributed algorithms that guarantee reliable operation of the system in the face of faults or sophisticated adversarial attacks on certain participating node.

6.1.1 Threats

The participants in permissionless blockchain may be malicious (or compromised) and therefore should not be trusted with the confidentiality and integrity of data and relationship information. There are various attacking threats can happen in blockchain system, not excepting the ABC blockchain. It includes the blockchain network security threats targeting the weakness of underlying peer-to-peer network, the malicious online activity and attacks against machine learning leveraged for intelligent governance in ABC .

We list the threats that ABC AI Governance aims to mitigate below [99].

1. **Network Threats**

   The blockchain peer-to-peer nature of the network, which includes all the nodes who maintain and run the blockchain protocols and provides services come under the blockchain network. In case of the ABC there are two types of nodes included in its p2p network ABN: worker nodes in ring1, which are application servers offering instant messaging services, storage services and game services provided in ABC light app platform CyberCube; and the management nodes in ring0, which run the blockchain protocol and manage the worker nodes including job allocation, result verification and reputation shifting. The adversary in ABC may controls a number of nodes or peers in the network and can block or degrade the network itself and can feed malicious information into the network. The following are some attacks may happen at the ABC network layer.

   Sybil Attack, the attacker subverts the reputation system of P2P network by creating a large number of pseudonymous identities and then use them to gain a suspiciously large influence. A Sybil attack in blockchain network is an attack where a single adversary is controlling multiple nodes in the network. While economic incentives (rewards and punishments) can

mitigate Sybil attacks against a blockchain system, it is hard to prevent a Sybil attack from occurring.

Eclipse Attack, which targets a specific node and sends them blocks of a private fork, while attempting to eclipse them from the rest of the network so that they don't see the main blockchain.

Probing Attack, scanning and probing behavior, as well as the attempted exploita- tion of high-profile vulnerabilities.

2. Attacks against machine learning

Several studies show that machine learning models may be vulnerable to well crafted malicious input in an adversarial environment [102]. Researchers have investigated the vulnerabilities exposed by various types of attacks, e.g., adversarial attack, poisoning attack and membership against machine learning models. These types of attacks usually start with manipulating the input samples by adding certain noises or obfuscating features to baffle the model into misclassifying the malicious samples or misleading actions in case of reinforcement learning. Additionally, machine learning models can also leak various types of sensitive information contained in the training data.

Adversarial Attack, which is specially crafted inputs that have been developed with the aim of being reliably misclassified in order to evade detection. Adversarial inputs include malicious documents designed to evade antivirus, and emails attempting to evade spam filters.

Data Poisoning Attack, which involves feeding training adversarial data to the classifier. The most common attack type we observe is model skewing, where the attacker attempts to pollute training data in such a way that the boundary between what the classifier categorizes as good data, and what the classifier categorizes as bad, shifts in his favor. The second type of attack we observe in the wild is feedback weaponization, which attempts to abuse feedback mechanisms in an effort to manipulate the system toward misclassifying good content as abusive.

Membership Inference Attack, which is to determine the membership of a data record in the training data of the machine learning model, given just the data record and black-box access to the model. In some cases, it can directly lead to a privacy breach. More commonly, it is used to optimize opponent model against guardian models.

Moreover, the malicious entities are considered to be Byzantine, and can launch both active and passive attacks. We assume that the adversary is unable to decrypt the content and addressing information of the packets from the user's traffic. We also assume that the hardware capabilities of the adversary limit him from controlling a majority of peers in the network.

6.1.2 Solution Sketch

To achieve the objectives of ABC AI Governance system against these threats, multiple techniques for automatically monitoring the threat and maintaining the system robust and resilient with the presence of adversary are proposed. All of these techniques adapt to core ABC blockchain function and follows the design philosophy:

- Decentralized: does not rely on a functionality provided by a central server(s) to perform its tasks.
- Autonomous: can operate without user intervention or expert feedback.
- Privacy-Preserving:multi agent coordinate without reveal sensitive individual information.
- Adversary Resistant: can withstand dynamic adversarial attack. We sketch our solutions as the following.

Management Representatives Sampling

Management representatives sampling are introduced by choosing a committee—a small set of management representatives randomly selected from the total set of management nodes in ring 0—to run each round of node status monitoring, result verification and management protocol. It allows nodes which

are monitored to report their status only to their random chosen management nodes. Particularly, the management nodes can also gossip their received messages to others in the management committee. By repeating this process periodically, each management node is going to maintain a partial, yet continuously updating set of received other nodes profiles (i.e., a partial view of the network), that has participated in the system. After every task, the management nodes cast a vote on the proposed action according to the performance of the node and the state of network based on AI mechanism.

Reputation

Reputation system are proposed to guide the nodes to perform well in network. It can combine with the economic incentives to achieve the goal of encouraging participating node in ABC to provide expected services and not to act maliciously. Nodes can gain reputation by working on the job owners' tasks. When the service is delivered, the workers that follow service level agreements, measured by quality, are rewarded with increased reputation. The others that violate the service level agreement are considered deceitful and thus lose reputation. This measurement of quality of service is done at the time of validation. Before validation is allowed to occur, enough replicas must have been returned and the sum of the respective workers' reputation must be high enough. The shifting of reputation and node role reduces the attack surface in ABC blockchain system.

Decentralized Machine Learning

To achieve the goal of monitoring, diagnostic and taking response action to the state of ABC system automatically without the intervene of experts, two machine learning algorithms are used:(1) Graph based anomaly detection are used to spot potential attacks early without knowing attack patterns and collecting labeled data; Specifically, node centric graph partition algorithm are leveraged to find the anomaly community in the hypergraph in the context of decentralized network.(2) Given the context of adaptive policy-driven scheduling, we opt for reinforcement learning algorithms to implement the proposed strategy learning model for ABC governance. Reinforcement learning based strategy learning that

can dynamically adapt to traffic variation, and to various task specific reward functions set by network operators, to optimally.

6.2 Architecture of ABC AI Governance Module

The architecture of ABC AI governance system is displayed on Figure 6.1. The system is designed to be entirely decentralized. It is composed of four main components: data extraction layer, graph-based feature extraction layer, task specific feature representation layer and reinforcement learning based strategy learning layer. As a result, it does not burden any single machine with excessive workload and at the same time does not require all the data to be centralized for execution.

6.2.1 Data Collection Module

This module is responsible to collect information needed for monitoring, diagnostics and strategy learning. In the context of decentralized data collection, each node receives partial measurements of the state of nodes or users in ABC blockchain, and seeks to asymptotically estimate the entire state by exchanging information with its neighbors in the network. Due to the possibility of having malicious peers disrupting the monitoring infrastructure by sending its neighboring peers with fake data, the data collection mechanism should make this malicious behavior harder for the adversary as a complementary method to economic punishment. We deal with collaboratively

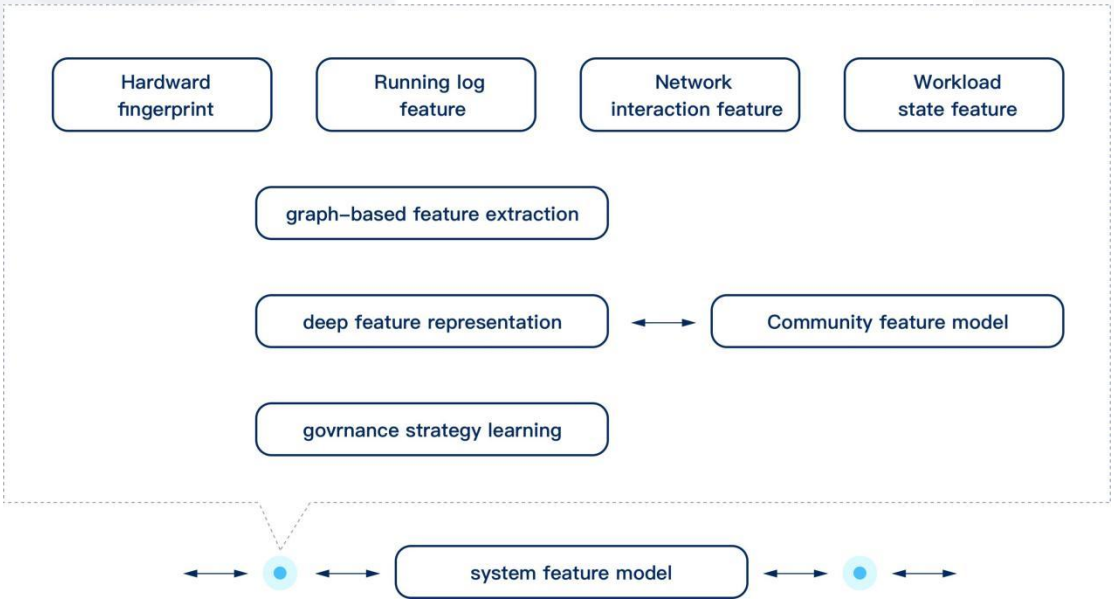

Figure 6.1: ABC AI Governance System Architecture

collecting and estimating the state of the system under the assumption that certain nodes are compromised by adversarial attacks. Specifically, we consider a Byzantine adversary model, where a compromised node possesses complete knowledge of the system dynamics and the network, and can deviate arbitrarily from the rules of any prescribed algorithm. We utilize a distributed filtering algorithm that enables each uncompromised node to asymptotically recover the entire state dynamics without explicitly detecting which nodes are under attack. We will discuss the algorithm and various type of data we collect in **Section 6.3.**

6.2.2 Graph Based Anomaly Detection

This module provides a big-data analysis framework to detect malicious and compro- mised nodes early without the need of relying on historical or labeled training data. The framework is based on large graph analysis and machine learning techniques. Graph Analysis and anomaly detection is performed locally on the peer side using the information collected by the nodes. Due to the dynamic nature of the observed network, it is difficult to determine a priori the expected values or behavior of the input data. Therefore, unsupervised machine learning techniques are required.

It first constructs a set of hyper-graphs to represent the activities of nodes and users. Each hypergraph node may correspond to a set of events or a set of users or server nodes in ABC network, with edge attributes specifying their connectivity relationship. On top of these constructed graphs, the system applies community detection algorithms and performs large-scale graph analysis to determine a subset of anomaly users or server nodes and their activities with high confidence. Particularly, in the context of ABC blockchain system, decentralized and scalable community detection mechanism are required. Random walks and diffusion-based techniques are adopted to extract disjoint communities [113], which can be implemented using node-centric programming model without requiring any global knowledge. The result set of detected high-confidence anomaly users or server nodes and activities are then used as self-generated training data to feed into the following machine learning components to derive a set of risk detection models or strategy learning models. Finally, these newly generated risk detection models can be used to detect the remaining set of undetected user accounts or account activities. In this framework, the graph analysis bootstraps the system to automatically generate training data on demand, without relying on historical training data obtained from manual labels or external detection components. As such, early detection of malicious users or server nodes and their activities in an unsupervised manner can be achieved. We will elaborates our graph based algorithms in **Section 6.4.**

6.2.3 Reinforcement Learning Based Strategy Learning

The ABC Governance system requires adaptive governance strategies to handle the unpredictable events occurring in nodes where various tasks are executing. To this aim, we adopt deep reinforcement learning to deal with the complicated scheduling and reputation shifting problems of the nodes in ABC network with large state space.

We can consider the management decision process as a Markov decision process (MDP) model [121]. MDP model can help selecting the possible actions from the current state and observing the derived reward/cost from each transition in order to find a better scheduling/reputation shifting decision in ABC

network. More concretely, we consider the scheduling and reputation shifting decision as a life cycle in which the node progresses through this life cycle and goes from one state to another. For instance, a node can go through the following states in the MDP model: idle, scheduled, finished, and failed with its reputation score. We consider the mapping between these states and the entire network states over the possible actions to find an appropriate scheduling and reputation shifting policy as the process of the scheduling and reputation shifting decision selection in ABC governance module. Following this approach, our proposed solution can estimate and compare all possible rewards that are earned when applying the actions from a given environment state. Furthermore, deep reinforcement learning based strategy learning framework allows to consider the dynamic events occurring in a system's environment and to adjust the decisions making procedures under uncertainty. We will discussed the reinforcement learning algorithm in **Section 6.5.**

6.2.4 Privacy Preserving

Given the decentralized nature of ABC system, sharing and working on sensitive data in distributed settings is a challenge due to security and privacy concerns. There are a number of different approaches, from homomorphic encryption to differential privacy. We choose to follow differential privacy scheme to preserve privacy in the process of distributed computation without losing our effectiveness and efficiency.

Differential privacy [104] is one of the most popular definitions of privacy today. Intuitively, it requires that the mechanism outputting information about an underlying dataset is robust to any change of one sample, thus protecting privacy and assuring resistant to membership inference attack. In the ABC Governance machine learning training implementation, we will follow a variation of the Laplacian mechanism to preserve privacy. Before the discussion of the mechanism we followed, we quote some definition of differential privacy.

A mechanism f satisfies (ϱ, δ)-differential privacy for two non-negative numbers ϱ and δ iff for all neighbors $d(D, D^{'})$, and all subset S of f's range, as long as the following probabilities are well-defined, there holds

$$P(f(D) \in S) \leq \delta + e^{\varepsilon} P(f(D') \in S) \quad (6.1)$$

where $d(D, D')$ denotes the minimum number of sample changes that are required to change D into D'. Intuitively speaking, the number δ represents the probability that a mechanism's output varies by more than a factor of e^{ϱ} when applied to a dataset and any one of its neighbors. A lower value of δ signifies greater confidence and a smaller value of ϱ tightens the standard for privacy protection. The smaller ϱ and δ are, the closer $P(f(D) \in S)$ and $P(f(D') \in S)$ are, and the stronger protection is.

Laplacian mechanism is a popular ϱ-differentially private mechanism for queries f with answers $f(D) \in R$, in which sensitivity plays an important role. The sensitivity is defined below:

Given a query f and a norm function $\|\cdot\|$ over the range of f, the sensitivity $s(f, \|\cdot\|)$ is defined as

$$s(f, \|\cdot\|) = \max_{d(D,D')=1} \| f(D) - f(D') \| \quad (6.2)$$

Usually, the norm function $\|\cdot\|$ is either L1 or L2 norm. The Laplacian mechanism: given a query f and a norm function over the range of f, the random function $\overline{f}(D) = f(D) + \eta$ satisfies ϱ-differential privacy. Here η is a random variable whose probability density function is $p(\eta) \propto e^{-\varrho \|\eta\| / s(f, \|\cdot\|)}$. There is a variation of the Laplacian mechanism, which replaces Laplacian noise with Gaussian noise. On one side, this replacement greatly reduces the probability of very large noise; on the other side, it only preserves (ϱ, δ)-differential privacy for some $\delta > 0$, which is weaker than ϱ-differential privacy. Variation of the Laplacian mechanism: given a query f and a distance function over the range of f, the random function $\overline{f}(D) = f(D) + \eta$ satisfies (ϱ, δ)-differential privacy. Here η is a random variable from distribution N.

6.2.5 Decentralized Optimization

In our decentralized setting, instead of sending all of their data to a centralized server, there is a set of workers, each of which collects data from different data sources. Therefore, the training of machine learning model used in ABC AI

Governance system requires a decentralized implementation, where nodes pass updates to every other data shard in the cluster without having a shared parABC ter server. There are many distributed algorithms such as the dual averaging-based algorithm and the subgradient methods. However, most of existing works are built on the hypothesis that the network is deployed in benign surroundings without any intruder. Under the presence of adversary, the existing distributed optimization algorithms become vulnerable or even invalid, which may lead to system paralysis. We define the decentralized optimization with adversary nodes problem as follows:

Assume that each agent i has its local cost function $f_i(x) \in \mathrm{R}$, where $x \in \mathrm{R}$ is the same variable owned by all agents. The local cost function $f_i(x)$, $\forall\ i \in V$ is a convex function. At the same time, it is supposed that the optimal point set arg min $f_i(x)$ is nonempty, bounded and closed. The derivative of the function $f_i(x)$, $\forall\, i \epsilon V$ is represented by $f_i'(x)$. The optimization problem can be written as,

$$\mathrm{Min}\ \frac{1}{n}\sum_{i=1}^{n} fi(x) \tag{6.3}$$

When adversary agents exist, for all normal nodes, the problem becomes,

$$\mathrm{Min}\ \frac{1}{n0}\sum_{i\epsilon V\setminus Va}^{n} fi(x) \tag{6.4}$$

Where n_0 is the normal agent. To address this problem of distributed optimization dynamics to failure and adversarial behavior, a resilient distributed filtering algorithm that guarantees that the non-adversarial nodes converge to the convex hull of the min- imizers of their local functions are used in the implementation of ABC AI Governance machine learning training, which we will discussed in **Section 6.3.1.**

6.3 Data Collection

In a fully-decentralized and highly dynamic P2P network, the absence of central or supernode like entities makes monitoring the network a challenge since any

information required for understanding the behavior of the network is distributed over possibly hundreds or even thousands of participating peers. Gathering information from all peers becomes unscalable as the network grows and privacy concerns limits the nature of information that can be collected from peers. In addition to these issues, we assume that there are adversary participants in the network. Therefore, a node may receive fake data from its peers. In the process of decentralized machine learning training, each agent receives gradient from neighbors to update its local variable value and the arbitrary fake data sent by adversary peers may lead system paralysis. To tackle these challenges, the distributed filtering algorithm are used to filter malicious data.

6.3.1 Distributed Filtering Algorithm

Consider the linear dynamical system $x[k+1] = Ax[k]$, where $k \epsilon Z$ is the discrete-time index, $x[k] \epsilon R^n$ is the state vector and $A \epsilon R^{n \times n}$ is the system matrix. The system is monitored by a network $G = (V, E)$ consisting of N nodes. The i-th node has partial measurement of the state

$$x[k] : y_i[k] = C_i x[k], \tag{6.5}$$

where $y_i[k] \in R^{r_i}$ and $C_i \in R^{r_i \times n}$. We denote

$$y[k] = [y_1^T[k], ..., y_N^T[k]]^T, \tag{6.6}$$

and

$$C = [C_1^T, ..., C_N^T]^T. \tag{6.7}$$

For any $\lambda_j \in sp(A)$, where $sp(A) = \{\lambda \in C \vee det(A - \lambda I) = 0\}$ denotes the set of all eigenvalues (modes) of a matrix A, let $z^{(jm)}[k]$ denote the m-th component of the vector $z^{(j)}[k]$, and let $z\hat{}^{(jm)}[k]$ denote the estimate of that component maintained by node $i \in V$, where V is the node set of the distributed network. For any node i, let the set of eigenvalues it can detect be denoted by O_i, and let $UO_i = sp(A) O_i$. Consider an unstable eigenvalue $\lambda_j \in UO_i$. For such an eigenvalue, node i has to rely on the information received from its neighbors, some of whom

might be adversarial, in order to estimate $z^{(j)}[k]$. To this end, a resilient consensus algorithm that requires each regular node $i \in V \setminus S_j$ to update its estimate of $z^{(j)}[k]$ using the following two stage filtering strategy:

1. At each time-step k, each regular node i collects the state estimates of $z^{(j)}[k]$ received from
only those neighbors that belong to a certain subset $N_i^j \subseteq N$ (to be defined later). For every
component m of $z^{(j)}[k]$, the estimates of $z^{(jm)}[k]$ received from nodes in N_i^j are sorted from
largest to smallest.

2. For each component m of $z^{(j)}[k]$, node i removes the largest and smallest f estimates (i.e., removes $2f$ estimates in all) of $z^{(jm)}[k]$ received from nodes in N_i^j , and computes the quantity:

$$z^{-(jm)}[k] = \sum_{l \in M_i^{(jm)}[k]} w_{il}^{(jm)}[k] \hat{z_l}^{(jm)}[k] \tag{6.8}$$

Where $M_i^{(jm)}[k] \subset N_i^j \subseteq N^j$ is the set of nodes from which node i chooses to accept estimates of $z^{(jm)}[k]$ at time-step k, after removing the f largest and f smallest estimates of $z^{(jm)}[k]$ from N_i^j. Node i assigns the weight $w_{il}^{(jm)}[k]$ to the i-th node at the k-th time-step for estimating the m-th component of $z^{(j)}[k]$. The weights are nonnegative and chosen to satisfy $\sum M_{l \epsilon m_i^{(jm)}[k]} w_{il}^{(jm)}[k] = 1, \forall \lambda_j \epsilon UO_i$ and for each component m of $z^{(j)}[k]$. With the quantities $\bar{z}^{(jm)}[k]$ in hand node i updates $\hat{z_i}^{(j)}$ as follows:

$$\hat{z_j}^{(j)}[k+1] = V(\lambda_j) \bar{z}_i^{(j)}[k] \qquad \text{if } \lambda_j \epsilon UO_j \text{ is real} \tag{6.9}$$

$$\hat{z_j}^{(j)}[k+1] = W(\lambda_j) \bar{z}_i^{(j)}[k] \qquad \text{if } \lambda_j \epsilon UO_j \text{ is not real} \tag{6.10}$$

where $\bar{z}_i^{(j)} = \bar{z}_i^{(j1)}[k] \ \ldots \ i^{(j\sigma_j)}[k]^T$, $\sigma_j = a_A(\lambda_j)$ if $\lambda_j \in UO_j$ is real and $\sigma_j = 2a_A(\lambda_j)$ if $\lambda_j \in UO_i$ is not real.

6.3.2 Types of Data

With the filtering algorithm in hand, various data can be collected and computed to estimate the state of network and to train AI based model. In order to distinguish the malicious nodes and cooperative nodes and give a evaluation of the quality of the user, we are using multi dimensional data include on-chain data and off-chain data to monitor the state of ABC network and to learn the historical malicious modes and cooperative nodes based on the historical logs.

Our goal is to provide detectors with network wide information collected at the same time that flow records are generated in order to minimize detection delay. We also wish to provide detectors with a combination of both volume and distribution information so that detectors can be generalized rather than specialized for specific attack signatures.

Information to be collected can be classified into:

- On-Chain Data: history transaction related data of nodes and users.
- Current State of Network: related to the underlying network and the interaction of each node. For example, number of neighboring peers, peer uptime, etc.
- Profile of Nodes: the reputation score, the workload and history performance of the node and the information related to the host. For example, memory and processor consumption.
- Running Log: history running log of tasks that nodes received.

These data will encode the information such as the individual state of each node and the partial view of the state of entire network. All these related features are fused into the anomaly detection model and various task specific strategy learning model to measure the reputation of a node in a multiple-dimensional way and make decision about task allocation, reputation shifting and privilege management.

6.4 Graph based Feature Representation

The governance system is responsible for the monitoring, diagnosis and dynamic scaling the network in order to robust and resilient to changing state. The governance system needs to ensure the user/nodes has not been compromised before assigning task. Hence, monitoring the state of the network is the first step towards secure system governance. In addition to monitoring the state of the network, anomaly detection is needed to spot unusual system behaviors such as failures, different attacks and anomalous communication patterns. To automatically and reliably detect anomalies, it is required to characterize and construct a model of normal network behavior and identify abnormal behavior as it occurs. The normal behavior of a node is expected to be constantly evolving and a present notion of normal behavior might not be valid in the future.

However, investigating individual resource behavior may not be efficient in detecting abnormal behavior in large and complex data centers. By leveraging Graph-Mining techniques [117], unusual behaviors in data centers could be detected not only based On a per-resource behavior, but using a holistic view of inter-dependency and inter- communication pattern between different resources.

The constructs and analyzes several types of activity graphs, referred to as hyper- graphs, to detect malicious (or compromised) accounts and malicious events without using training data. A global view of the connectivity structures among users and events allows the system to perform early detection of stealthy attack patterns that are difficult to identify when each user or event is examined in isolation. The hypergraph based detection can identify groups of malicious accounts without requiring labeled data provided by the customers. The labeled data are often hard to obtain, especially with new unseen attacks. With hypergraph analysis, the system can self-bootstrap the system with an initial list of malicious accounts or events. This step also has the ability to capture new attack campaigns automatically.

6.4.1 Graph Construction

To build the hypergraphs, the system first processes input data and derives a set

of features and statistics for each node (or each node event). The combination of all features or statistics is referred to as a profile. For each user or node, the system can compute a corresponding profile. In addition, for each of one or more groups of users or nodes, the system can compute a corresponding group profile. Collectively across an entire user population available to the system, a global profile can be computed.

The set of computed feature profiles will be used to construct hypergraphs for graph analysis. Each node on a graph corresponds to a feature profile. Each feature profile can be constructed from a set of correlated events or a set of correlated users or nodes. The set of correlated events or correlated user or node is identified by taking the set of events with similar behaviors. The edges of the graphs may be computed in multiple ways. The edges can be computed by adding an edge between node *A* and node *B*, if *A* and *B* share a similar feature. To determine if two features are similar, the system can perform the following procedures. If the feature corresponds to a numerical value, then the system can compare their respective values. In some implementations, the system checks whether the difference between two corresponding feature values is smaller than a pre-set threshold. Alternatively, in some other implementations, the system checks whether the ratio of two features value is smaller than a pre-set threshold.

The output graph component information can be combined with individual user or node or event information to generate an initial list of malicious users or nodes with a high confidence, as they have exhibited stronger global correlations in conducting malicious activities.

6.4.2 Suspicious Graph Node Detection

Once the system obtains a list of suspicious graph nodes, it proceeds to identify suspicious graph communities. Graph communities can be identified using several different graph algorithms. As mentioned in Section 6.2.2, random walk and diffusion- based techniques which can be implemented in node-centric model are adopted to adapt to the decentralized feature of ABC blockchain system.

Graph Diffusion based Community Detection

Classical community detection is formulated as a clustering problem. That is, given the full graph $G = (V, E)$, partition the vertex set into K subsets $S_1,\dots, S_k$, (a partitioning), such that $\mathrm{T}_1^k\ S_i = \varnothing$ and $\mathrm{S}_1^k\ S_i = V$. A quality metric $Q(\{S_1,\dots, S_k\})$ is defined over the partitions and a community detection algorithm will try to find a partitioning that maximize or minimize Q depending on its nature. This is for non-overlapping community detection and one can simply remove the constraint $\mathrm{T}_1^k\ S_i = \varnothing$ to get the overlapping version. Note that Q is only an artificial surrogate to the axiomatic notion of community.The maximum Q does not necessarily corresponds to the best community. However, the community detection problem becomes tractable via well-studied optimization frameworks by assuming a form of Q e.g. Modularity, Conductance. Now consider the decentralized scenario. One node (observer) is limited to its local view of the whole graph. It is unreasonable to ask for a global partitioning in terms of sets of nodes.

6.5 Fully Decentralized Reinforcement Learning

Reinforcement learning is a machine learning method used to tackle sequential decision-making problems through a trial-and-error technique to search for effective actions [121]. In the broadest sense, a machine learning system, using the reinforcement learning paradigm, interacts with a single environment; it observers the state of that environment, selects an action, and receives a scalar reward or feedback for the action. The process is depicted in Figure 6.2.

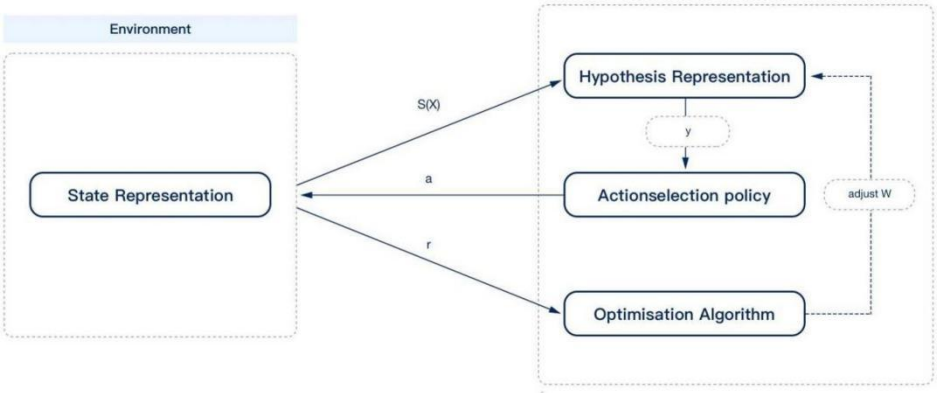

Figure 6.2: Reinforcement Learning Paradigm

The environment, in this paradigm, is characterized by a set of states, S, in which every state is constructed from a vector of features (called state features). The machine learning system consists of a set of actions, A, that are applicable to perform on the environment (**Figure 6.2**). A machine learning system interacts with its environment at each time of a sequence of discrete or continuous time steps, $t = 0, 1, 2, 3,$ The interaction takes place through a repeated cycle of three steps:

1. sensing the state of the environment at t, $s_t \in S$;
2. performing an action $a_t \in A(s_t)$, where $A(s_t)$ is a set of actions that are admissible for the state s_t; that is, $A(s_t) \subset A$;
3. receiving a scalar reward, which in general cases is defined as $R: S \times A \times S \rightarrow \mathrm{R}$, which specifies the reward obtained for a transition from one particular state to another, after performing an action.

At each time step t, the machine learning system interacts with its environment with the goal of building an action selection policy (denoted π_t), which maps the states to the actions $\pi: S\ A$, where $\pi_t(a \rightarrow s)$ is the probability of $A_t = a$ if $S_t = s$. The goal of this policy is to maximize the reward signal that represents a long-term objective. Thus, the policy is a fundamental step in understanding the characteristics of the reinforcement learning components before building any reinforcement learning application; the components are the environment state space, the machine learning system action space, and the reward space. These components of reinforcement learning can be formalised using a Markov decision process (MDPs) framework, especially if the state and action space are discrete. They are tuples (S, A, P, R) where:

- S is the set of environment states, which can take a broad range of forms. For instance, state spaces can be defined by continuous variables such as velocity, price, performance etc., called continuous state-spaces ($|S| \in \mathrm{N}$);

Alternatively, they can be defined by a discrete state-space if the number of states is discrete.

- A is the set of possible actions available to the machine learning system.
- P is the state transition function. It is defined as $P(s_t, a_t, s_{t+1}) \rightarrow [0, 1] \rightarrow [0, 1]$, where P represents the probability of reaching state $s_{t+1} \in S$ by applying action $a_t \in A(s_t)$ in state $s_t \in S$. A characteristics of this function is that it is deterministic. This refers to the probability of the learning system being in some state s_{t+1} after taking action at from state s_t, or $p^{at}_{ststst+1}$. State transition determinism occurs when $p^{at}_{ststst+1} = 1$.By contrast,if $p^{at}_{ststst+1} < 1$ the transition is non-deterministic or stochastic.
- R is the reward function: $R(s_t, a_t, s_{t+1}) \rightarrow$ R. It provides an immediate indication when an action $a_t \in A(s_t)$ is taken in state st and moves the machine learning system into a subsequent state $s_{t+1} \in S$.

6.5.1 State Space

The state space S consisting of all possible state vectors s^i. Where each s^i is the concatenation of a pair of sub-states such that $s^i = (c^i, o^i)$. The control sub-state helps inform the controller of which metric or metrics are in most need of correction. The operating sub-state gives the controller information about the current network operating environment. The control sub state is a three element vector representing the three performance metrics: precision, recall, and forwarding. Each element of the vector is set to 1 or -1 indicating necessity for improvement of a particular performance metric where 1 indicates improvement required and -1 indicates none. The calculation of c^i_t is accomplished by use of a artificial neural network (ANN) to map system performance and operator priorities to the possible sub-states. The ANN accepts as input, the value at time t of the three performance functions $F_q(t)$ and three operator defined performance goals $G(q)$. A control string indicating the priority order of the performance functions $G(q)$ is used to calculate weights. Where q = (1, 2, 3) =

(Precision, Recall, Forwarding).

The operation sub state o^i consists of a vector of state variables produced by the Network Preprocessor. This combination of information allows the system to respond to changes in network conditions while also providing the system information both specific to and independent from the underlying algorithm. The resulting state space is potentially high dimensional and continuous. In reinforcement learning continuous state spaces can be managed using function approximation methods.

6.5.2 Action Space

We define an action space A over a parABC ter space P as a set of action vectors where each element of an action vector represents a possible governance action on the available governance parABC ters in the underlying detection system (p^1, ..., p^n). Each governance action consists of a direction and magnitude of change for that governance parABC ter. We discretize the action space into a set of n dimensional action vectors $a = (a_1, ..., a_n)$ where n is the number of tunable parABC ters in the objective model and each element of $a_i = \pm 1$ where the sign represents adecision to increase or decrease p^k. We also maintain a single n dimensional vector M to track the magnitude of the tuning changes. Each m^k in M is a dynamic range variable that increases or decreases with the series of sign changes in a_k^i. Each m^k is a range limit in the interval $[0, \delta k]$, where $\delta^k = \max(p^k) - \min(p^k)$ betw een minimum and maximum values according to the underlying algorithm. Each time the sign of a_k^i stays the same, m^k is incremented toward δ^k. Each time the sign of Each time the sign of δ^k changes, m^k is reset to 0. Ceiling and floor limits of $\max(p^k)$, $\min(p^k)$ are applied to ensure parABC ters stay within algorithmic limitations. The result of this approach is that when parABC ters are tuned on consecutive intervals in the same direction, the magnitude of the adjustments continues to increase. When parABC ters are tuned on consecutive intervals in the different directions, magnitude of the change is initially reset to 0 and gradually increased as appropriate. Additionally, while the direction of the adjustment to p^k is determined by the discrete action output by the RL tunner, the magnitude of the change is independent of the other parABC ters in the

action vector. It is determined by the aggregate history of sign changes of p^k only.

6.5.3 Reward Definition

One of the primary objectives of this approach is to ensure that high priority metrics remain within established goals while lower priority metrics bear the tradeoff. The reward function is calculated to place emphasis on priority metric functions remaining within established goals. We first determine if the system is within operating parABC ters. If the weighted factor of H is used when the system is not within parABC ters to ensure the priority metrics are accounted for first. We calculate a system score at time t:

$$F_q(t) - G(q). \tag{6.11}$$

Where G(q), and $F_q(t)$ are as previously described. If over the interval $t \ldots t + 1$ the score has increased, then the reward is 1 and 0 otherwise. If the system has crossed the threshold of operating standards from in standards to not in standards, the reward is 0. If the system has crossed from not in standards to in standards, the reward is 1. Otherwise the weighted r non weighted score is used to determine reward.

Once the training phase is done, various decisions can be made by the algorithm. The decisions include whether the observed state is anomalous and the reputation shifting decision.

7 Economy System

7.1 Introduction

This chapter proposes an economic development plan for the ABC blockchain ecosphere. The purpose is to enable participants of the entire ecology to perform their duties under reasonable economic incentives and strong fiscal governance to ensure the healthy operation of the ABC blockchain's ecosystem. First of all, we will introduce the unique three-layer ecological structure, including the underlying ABC blockchain technology platform, the application layer-blockchain messenger, and the top-level CyberLand-based game platform CyberCube . Secondly, in order to link the entire three-layer ecological structure, we have established a set of economic operation systems, including the token issuance mechanism, consensus economics, and the economic cycle mechanism. We also illustrate how the economy within the entire ecology is running through practical examples. Thirdly, in order to ensure the healthy operation of the ecology and stabilize the value of the tokens, we have added several fiscal policies, including inflation, bond mechanism, and foundation system, forming a distinctive economic system with a combination of free markets and macro controls. Next, we will discuss some of the mechanisms used in the economic system, and make comparisons with other blockchain projects to elaborate in detail the trade-offs in the design of these mechanisms. The final section will summarize the ecosystems covered in this chapter and propose our vision for the future.

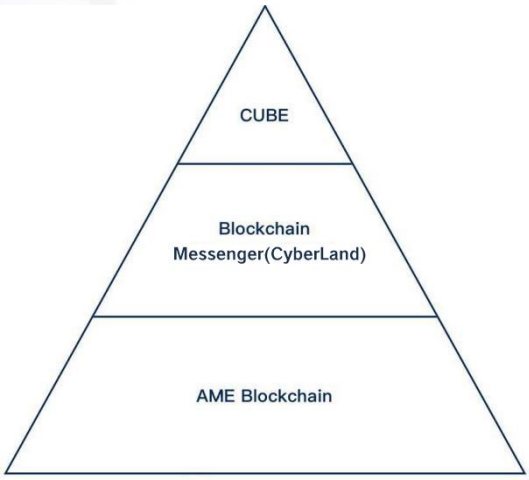

Figure 7.1: Schematic Diagram of The Three-layer Ecological Structure

7.2 Three-Layer Ecological Structure

The three-layer ecological structure splits the entire ecosystem into three parts, namely the underlying ABC technology platform, the middle level blockchain messenger (here-inafter referred to as CyberLand) and the top-level CyberCube. Among them, the ABC technology platform is the foundation of the entire ecology, as well as the provider of resources and services. CyberLand is an instant messenger based on the ABC technology platform that provides users with secure and encrypted communication services. CyberCube is a blockchain-oriented open application platform built on top of CyberLand, especially focusing on gambling games. The platform provides easy-to-use development tools and blockchain resources for developers. Developers only need to pay attention to the development of business logic. Meanwhile, the platform also provides comprehensive APIs, and welcomes as well as supports the third part games to access in CyberCube. Obviously, the three parts of the ecology progressively form a pyramid structure. We will introduce the three parts of the pyramid in details, focusing on the participants in each part and their code of conducts, rather than specific technical details.

7.2.1 ABC Technology Platform

As mentioned earlier, the ABC technology platform is a secure, self-managed, and de- centralized platform based on blockchain technology. Hence, the blockchain technology is the basic component of the ABC technology platform. It adopts a double-ring network topology design, manages the node servers participating in the blockchain system, and separates the management logic from the business logic. Thus, a flexible and scalable-decentralized P2P system solution is realized. For different kinds of server nodes, roles and benefits of them are discussed as below.

Manager Ring Node

"Mining" Rewards

In the blockchain, the server nodes in the Manager Ring, which assume the role of the manager, and is not responsible for the specific businesses, can respond to the client's transaction request, and assign the corresponding worker server to the client The transaction is collected and stored in the ABC blockchain by the Manager Ring nodes in the form of "mining".

Generally speaking, in other blockchain projects, the successful "mining" node receives the mining rewards which are distributed in the form of tokens. For example, a winner in Ethereum's blockchain "mining" wins 5 new Ethereum rewards [122]. However, it needs to be clarified that the "mining" rewards is not an endogenous rule of the blockchain, but a sociological means of economic incentives. For example, at the time of this white paper successful bitcoin miners grant 12.5 bitcoin rewards, but the total amount of bitcoins will reach 21 million when 2140. After that, miners will not be able to get "mining" rewards [1]. The source of revenue for miners will only be the transaction fee. In addition, in the Ripple protocol, the server node does not need nor can't "mine", and naturally cannot get the "mining" rewards [123]. In other words, when deciding whether to reward a server node for successful "mining", one must find the sociological basis.

The ABC blockchain is intended to issue a certain amount of token-rewards to server nodes that successfully "mine". This setting is mainly based on the

following considerations. First of all, the Manager Ring nodes competed in "mining", packaged the transaction information, paid the computing power, and contributed to the blockchain system. Secondly, the combined effect of "mining" cost and "mining" rewards will increase the cost of malicious behaviors; using the behavior of nodes pursuing profits to restrict nodes from behaving maliciously. Finally, the "mining" behavior of the ABC blockchain system exists only in the Manager Ring, and the nodes in the Worker Ring and the temporary Worker Ring are not eligible to participate in "mining". The ABC blockchain system provides a swap channel for nodes of the Worker Ring and the Manager Ring. If a Worker Ring node provides better service, then he has the opportunity to compete into the Manager Ring. If a Manager Ring node fails in the competition, then he may be downgraded into the Worker Ring. In order to benefit, server nodes tend to provide better resources and services to enter or remain in the Manager Ring and obtain "mining" rewards.

Transaction Fee

In addition to the "mining" rewards, the transaction fee is another source of revenue for the Manager Ring nodes. When the client initiates a transaction, it is required to pay a transaction fee, which is allocated to the service providers- the Manager Ring nodes and the Worker Ring nodes. The purpose of setting the transaction fee is to prevent the client from abusing the resources of the underlying blockchain, and also to quantify the work of the blockchain nodes.

Service Fee

In addition to the transaction behavior, the Manager Ring nodes can also provide non- transaction behavior for the client, that is, a service that does not need to be broadcasted, in order to obtain a "service fee". This is an affirmation of the labor value of the nodes. Even if the Manager Ring nodes are not qualified to "mine" in the current packaging process, they can still obtain certain benefits through the work, ensuring the enthusiasm of all nodes in the Manager Ring, preventing none of the mine-qualified nodes from being passively absent or cheating.

Transaction Fee Allocation

As mentioned earlier, the main sources of revenue for the Manager Ring nodes are "mining" revenue and transaction fee. And, the "mining" income belongs to the nodes that successfully propose blocks. There are two forms of transaction-fee allocation which are discussed separately below.

The first form is to grant all of the transaction fee to the successful "miners". All transaction fee generated within a block time, along with rewards for "mining", are issued to the node that successfully proposes the block. In other words, in addition to the "mining" rewards, the nodes with successful "mining" also received all the transaction fee, while the remaining Manager Ring nodes were unable to obtain transaction fee rewards. For the nodes involved in the "mining" competition, it is within their psychological expectation to not get the block and not receive the transaction fee rewards. But what is frustrating is that this kind of rewarding method makes the nodes without "mining" qualifications have no hope of obtaining transaction fee rewards in this block time. Then, there will be a sociological problem. Since I don't have the possibility of getting a transaction fee, why should I participate in the managing task of the transaction?

The second form solves this problem by allocating transaction fee to all nodes on the Manager Ring. Fairly, in a certain period of time, all nodes on the Manager Ring are involved in the processing and managing of transactions, and it is reasonable to obtain transaction fee based on their contribution. The first form of transaction fee is technically implemented without any threshold, but it is easy to cause the "non-miner" Manager Ring nodes to cheat, and it is not effective to stimulate all nodes on the Manager Ring to work hard. The second way is to actually provide a reward-stimulation for all Manager Ring nodes, regardless of whether the node participates in "mining", which can effectively improve the enthusiasm of the Manager Ring nodes. But the second solution poses a tricky technical question, namely, how to confirm that the nodes involved in the "mining" or not, and whether their workload is consistent with what they claimed to be? Even if this problem is solved, the system will face challenges from many nodes, resulting in a big amount of energy being spent on anti-cheating tasks.

Therefore, we propose a third solution and try to solve this problem by using sociological methods instead of technical methodologies. In this scenario, the "mining" rewards and the current round of transaction fee are still packaged in the form of "ABC" and rewarded to the successful "mining"-nodes, but nodes without "mining" qualification can still earn "service fee" in the form of "CC" by providing the client with non-transaction type of services. "ABC" and "CC", explained in Chapter 3, are both the tokens circulated in the blockchain ecosystem. It is worth noting that "ABC" can exchange into legal tender in the exchanges while "CC" can not, and hence, "ABC" is more valuable than "CC". This method ensures that the Manager Ring nodes have opportunities to profit at any moment, and prevents the nodes that are not qualified for "mining" from being negatively absent or cheating. Besides, the token issuance with two layers structure can make the influence on the whole economy brought by the profit allocation as little as possible.

Unworthy Law

"Unworthy law" is one of the ten major laws of economics. The most intuitive expression is that things that are not worth doing are not worthy to be done well. The law reflects people's psychology: if a person is engaged in something that he or she thinks is not worthwhile, they will often hold perfunctory attitudes towards it, resulting in not only the small success rate, but also lack of sense-of-achievement even if they succeed.

The original intention of the service-providing blockchain nodes is to earn revenue. If the nodes do not have the expectation of earning revenue in a round of service, according to the "unworthy law", the blockchain nodes will think that this round of service is not worth being done well, maybe not even worth doing at all. In addition, some nodes tend to behave maliciously for those that they think are "worthy to do" in order to profit. Therefore, as mentioned above, we have proposed the concept of "service fee" to solve this problem.

Worker Ring Node

The Worker Ring nodes are responsible for the specific business, perform the

work assigned by the Manager Ring, and obtain revenue by providing services and resources. (Including computing power, network bandwidth and storage, etc.) As mentioned above, the Worker Ring nodes do not participate in "mining", so their income includes only transaction fee and service fee.

According to the "Double-Ring" topology of service nodes, one manager ring node manages N worker ring nodes. Therefore, the manager node should share part of incomes with its workers. Besides, the worker, which is subject to many managers simultaneously, can complain to all the managers if some of his managers do not share with him. The whole process will be completed with the help of AI self-manage module. Next, we will introduce the specific process of the profit allocation.

Interest-Distribution among Nodes

The interest-distribution among nodes refers to how the transaction fee between the Manager Ring nodes and the Worker Ring nodes are distributed in one block period. The transaction fee and "mining" rewards we discussed are issued in the form of tokens (ABC, ABC Blockchain Coin). Assume that successful "miner" is granted with M ABCs; All transaction fee in a block time are T ABCs; Manager Ring has N_1 nodes, and Worker Ring has N_2 nodes. Then,

$$E_m = (M + T \div 2) \div N_1 \tag{7.1}$$

$$E_w = T \div 2N_2 \tag{7.2}$$

where E_m is the revenue expectation of Manager Ring nodes, E_w is the revenue expecta- tion of Worker Ring nodes

Let T = 2M, that is, the "mining" rewards are 1/2 of the transactions rewards in one specific block period, then the formula above can be transformed into:

$$E_m = 2M \div N_1 \tag{7.3}$$

$$E_w = M \div N_2 \tag{7.4}$$

Among them, the upper limit of N_1 is 100, and $N_1 \ll N_2$, so $E_m \gg E_w$, that is, the revenue expectation of the Manager Ring nodes is far greater than the revenue expectation of the Worker Ring nodes. Therefore, to pursue revenue, the Worker

Ring nodes hope to enter the Manager Ring in order to obtain higher revenue.

In order to form a healthy competition-ecology and make blockchain nodes striving to provide better services, ABC blockchain provides a bidirectional channel between the Manager Ring and the Worker Ring. If the Worker Ring nodes provide better resources and services, they can be allowed to enter the Manager Ring and become Manager Ring nodes, leading to higher revenue. However, if the quality of service of the Manager Ring nodes are degraded or cannot meet the requirements of the Manager Ring nodes, they will be degraded to the Worker Ring to undertake the work of the business layer.

Specifically, the access mechanism of the Manager Ring and the Worker Ring includes the following three aspects: the workload, the stability of the service, and whether the node has a bad record. Regarding one of the three aspects, if the node has ever committed a bad behavior, it can never enter the Manager Ring to provide services. The dynamical adjustment mechanism is driven by the AI governance module.

Disbursement and Collection of Fees

When the client uses the service of the node, it needs to pay the real-time transaction fee or the service fee. The node cannot receive the fee immediately since the "lightning network" performs the timed settlement. The method reduces the pressure of the server.For nodes, the amount of service fee they can receive is related to the amount of work they provided in that service. For Manager Ring nodes, in a transaction or service, they often provide a one-time "introduction service", which is to help the client find a suitable Worker Ring node providing resource services. After the "introduction" is completed, Manager Ring nodes' work is often over, and will not continue with the client and Worker nodes to the end of the service period. Therefore, it is reasonable to use the number of transactions or the number of services to evaluate the workload of the Manager Ring nodes. The Worker Ring nodes are the real executor of the service. Ethereum uses the size of the transaction, i.e. the number of bytes, to calculate the transaction fee. This is a simplified solution that considers storage and ADSL services only, and eliminates the computing power service. Its advantage is easy

to carry out. If the computing power is also added to the evaluation criteria, then the computational complexity of the transaction needs to be evaluated. Therefore, the workload evaluation of the Worker Ring nodes includes two aspects: the number of bytes of the transaction and the computational complexity of the transaction.

7.2.2 **CyberLand IM**

Function Description

CyberLand is an instant messenger based on the ABC blockchain platform that can run on iOS and Android operating systems. CyberLand is committed to making itself a blockchain game platform, providing fast access methods and dedicated development tools for developers to support third-party developments. More importantly, the vision of CyberLand is to realize decentralized operation. That means CyberLand is an application run on the ABC blockchain independently and can not be controlled by anyone or any organization.

The Tragedy of The Commons

The Tragedy of the commons is a theoretical model proposed by Professor Garrett Harding in the United Kingdom in 1968 [124]. The core idea of the model is that the commons, as resource or property, has many owners. Each of them has the right to use, but has no right to prevent others from using it, resulting in excessive usage amount and exhaustion of resources. Excessively felled forests, overfished fishery resources, and heavily polluted rivers and air are typical examples of the "tragedy of the commons". The reason why it is called tragedy is that every party knows that resources are exhausted due to excessive use, but everyone feels powerless to prevent the situation from continuing to deteriorate. Moreover, because of greedy human nature, there is no easy way to recover the system without any external force. Therefore, the public property is, because of determination of property rights or the cost of defining is too high, inevitably used excessively or encroached competitively.

For CyberLand users, the ABC blockchain platform is actually a "commons". If

the CyberLand user use the ABC normally, the underlying blockchain resources are totally satisfactory for all users' demands. However, if a malicious attack is encountered, or a large number of fake accounts send messages at the same time, it is easy to cause blockchain resources to be abused, resulting in the "tragedy of the commons".

Problem Solving

From a certain perspective, the cause of the "tragedy of the commons" can be attributed to the fact that people do not need to endure any costs when using resources, and do not need to pay any fees to the resource providers. As an IM application, CyberLand provides users with functions of sending messages and using services. Considering the user experience, it is naturally unnecessary to ask users for payments. So, how to prevent malicious users from sending messages madly or abusing blockchain resources while ensuring the free experience of normal users?

In order to solve this problem, we require that users' chat for a certain period of time to be a transaction, and the transaction requires users to spend "CC". "CC" is the chat fund that the system regularly issues to the user, and the amount of each payment is sufficient to ensure the normal usage of the user during a certain period of time. If the user uses an excessive amount, the system will use the CAPTCHA method [125] to quickly verify the authenticity of the user to ensure the normal usage of real users. The "CC" spent by the user on the CyberLand service is transferred to the blockchain nodes providing the service and assigned as the way that's described in Section 2.1. For each block period, the system will measure the workload according to the "CC" received by each node, and then issue the corresponding amount of token rewards. These rewards are the transaction fee income part of the node.

7.2.3 CyberCube

CyberCube is a CyberLand-based platform for blockchain games. The platform provides easy-to-use development tools and blockchain resources for third-party developers who only need to pay attention to the development of application

logic. At the same time, the platform provides a convenient interface for third-party games, welcomes and supports games which have already existed on other markets to access into the platform.

Decentralized game Platform

Traditional game distribution channels, such as Facebook, AppStore, Wechat, etc., enjoy absolute authority as centralized platforms in the whole ecology of games. game developers need to pay different percentages of their profits to the platform. As a participant in the value of the platform and as a value input, the game players' labor is not recognized, and the value generated on the platform is mostly acquired by the platform. As a game platform based on CyberLand, CyberCube is committed to creating a free service platform that connects game developers and players with the idea of decentralization. Due to the investment properties and anonymity of cryptocurrency, CyberCube aims to embrace the gambling games and capital related games.

Game Developers

From game developers' perspective, the platform does not charge any fees, and all game revenue is owned by the developers themselves. CyberCube provides game developers with easy access. For third-party games, access to the platform is equivalent to an extra income. It is a very attractive policy for developers. At the same time, CyberCube also provides developers with convenient game development tools. Based on the development tools and the underlying server of the blockchain, the game can be quickly deployed on the CyberCube. Reasonably, if the developers use the blockchain service, they will have to pay a fee to the server.

Players

From the perspective of the gamers, while playing games through CyberCube, that a certain amount of digital currency can be harvested, and the value of labor is recognized by the platform. Using tokens rewarded by the platform, game players can continue to buy platform games or in-game purchases to meet their

daily gaming needs. For heavy rollers, the platform's rewards cannot meet their gaming needs. So the heavy rollers will buy tokens from exchanges for game purchases or in-game purchases.

Inflation

In the ABC blockchain ecosystem, the amount of money will increase with the development of the system. According to the Fisher equation, P will increase, that is, inflation will occur. According to classical economic theory, moderate inflation favors the functioning of the labor market (ie, the workers in the ABC ecosystem) and promotes the vitality of the economy. One reason is that it is difficult to renegotiate for price cuttings in certain cases, especially for wages and contracts. Therefore, if the price rises slowly, it would be easier for the relevant prices to adjust. There are a variety of commodity prices that will "resist to price cuts" and tend to keep rising. So trying to achieve zero inflation (prices are maintained) will lead to lower prices, profits, and numbers of employees in other industries. Therefore, the executive departments of several companies regard mild inflation as "lubricating the commercial ship". The pursuit of absolute price stability will lead to devastating deflation (continuing price dropping), which will lead to bankruptcy and economic recession (even the economic depression). As the ABC ecosystem will continue to have blockchain nodes, users and other currency value injections, it is necessary to issue a certain amount of currency to ensure ecological operation. But it needs to be ensured that the additional currency will stimulate the entire ecology mildly, so that the system will maintain a moderate 2% inflation-policy every year.

In principle, we expect the amount of money deposited in the option pool to account for 20% of the total currency in the same period to ensure a stable demand for currency within the system. In addition, annual inflation is expected to account for 20% of the option pool, then the inflation of the entire system will be controlled at around 4%. It is worth noting that the amount of tokens that is forgotten, discarded or lost in circulation each year is expected to account for about 2% of the total amount of tokens. Therefore, the annual inflation rate should be controlled at 8% to meet the demand of these three aspects, including

option pool deposit, token destroyed, and additional issuance.

Token Demands

In the entire ecosystem, there are three demand parties for currency, namely, the option deposit of nodes, the game developers of CyberCube, and the users of CyberCube. The currency demand of CyberCube game developers lies in the use of the underlying resources of the blockchain, which requires a certain fee to be paid to the blockchain nodes. The recharging/circulation currency on the CyberCube game platform is ABC, and players need to spend a lot of ABCs to play games, so players will create a big amount of currency demand. In addition to the 20% monetary option deposit, we expect the CyberCube gaming platform to generate approximately another 20% of the currency demand.

7.2.4 Consensus Economy

Chapter 2 has elaborated and analyzed the ACP consensus mechanism. This section will analyze the distribution of economic benefits and the reasons for each role in the consensus process based on ACP mechanism.

ABC Incentive

The FC is the final committee for the second round of elections, which bears the responsibility of packaging and generating consensus through the PBFT * algorithm, which is also the "mining" in the traditional sense. In order to encourage the miners to work honestly and reduce bad behavior, FC members should be granted with economic incentives after they have made a block and be confirmed successfully. Since the miner's "mining" motivation lies in the ABC that can be exchanged, the economic incentives at this time are issued using ABC.

As mentioned earlier, FC members can be divided into three roles:

- Final Miner – A successful packager.
- Final Leader – The proponent of the block in PBFT* .

- Final Verifier – The verifier of the block in PBFT* . All FC members are verifiers.

According to the contributions of the three parties, the overall income ratio of the three parties is defined as:

$$name_{-miner} : name_{-leader} : name_{-verifier} = N_1 : N_2 : N_3 \quad (7.7)$$

Therefore, the revenue model for different role in FC is:

- $name_{-miner} = N_{r-eco-abc} \div (365 \times 24 \times 60 \times 6) \times N1 \div 10$
- $name_{-leader} = N_{r-eco-abc} \div (365 \times 24 \times 60 \times 6) \times N_2 \div 10 \div N_{fc-leader}$
- $name_{-verifier} = N_{r-eco-abc} \div (365 \times 24 \times 60 \times 6) \times N_3 \div 10 \div N_{fc}$

The income expectation model of the FC node is:

$$N_{fc-node} = (name_{-miner} + name_{-leader} \times N_{fc-leader} + name_{-verifier} \times N_{fc}) \div N_{fc} \quad (7.8)$$

Where N_{fc} is the number of nodes of the FC.

In addition, assuming that the upper limit of the capacity of one block in the ABC blockchain network is C_{limit} MB, and the size of the package proposed by Final Miner is C_{final} MB, then the ABC reward that can be obtained is determined by the following formula:

$$name_{-miner-non-selfish} = name_{-miner} \times (C_{limit} - 1 \div (C_{final} + k)) \div C_{limit} \quad (7.9)$$

Among them, k is a constant, which is used to control the convergence speed of the benefit; $name_{-miner-non-selfish}$ is the actual benefit that can be obtained according to the size of the package every time the node is successfully mined.

In order to encourage all members of the FC to package local transactions into blocks as much as possible, it is necessary to associate the interests of the remaining two roles in the FC, Final Leader and Final Verifier, with the size of the package, namely:

$$name_{-leader-non-selfish} = name_{-leader} \times (C_{limit} - 1 \div (C_{final} + k)) \div C_{limit} \quad (7.10)$$

$$name_{-verifier-non-selfish} = name_{-verifier} \times (C_{limit} - 1 \div (C_{final} + k)) \div C_{limit} \quad (7.11)$$

CC Incentive

The PC was elected to the committee in the first round of elections. Its members further elected FC through VRF calculations. They paid for their power and labor and should receive corresponding economic incentives. More importantly, the incentive for the PC to ensure that its members do not maliciously broadcast FC to the entire network, causing FC members to be attacked by malicious nodes. Obviously, before the FC wants to carry out the PBFT consensus, it is necessary to broadcast its identity to the PC to find all FC members, and the PC knows the identity of the FC. When a non-FC PC knows that it can't enter the FC in this round, leading to no chance of "mining" reward, and then they may act maliciously because the success of this round of packaging has nothing to do with itself. Therefore, we propose that if the last round is completed successfully, PC will be rewarded at the beginning of current round, thus ensuring that each round of PCs hope the success of the last round, so as to avoid the economic interests of evil behavior.

Since the number of PC nodes is relatively large, and in the long run, each node is equally likely to assume PC work. Therefore, the incentive for PC is a wide range of airdrop behavior, which requires the use of CC tokens with relatively low value and limited circulation.

Assuming that the blockchain has 100,000 participating nodes, according to the ACP consensus protocol, the theoretical period of one block is about 10 seconds. According to the CC unlock plan in Section 3.1, the number of CC issued in this round should be T times the ABC issued in the previous round. Therefore, FC members can receive CC rewards for each successful block cycle, which can be calculated by the following formula:

$$N_{CC-pc} = N_{r-abc} \times T \div (365 \times 24 \times 60 \times 6 \times N_{pc}) \quad (7.12)$$

Where N_{r-abc} is the total amount of ABC rewarded to the nodes in the last round.

As a common blockchain node in the ABC blockchain network, the daily CC revenue expectation is:

$$N_{CC-node/day} = N_{CC-pc} \times N_{pc} \div N_{all} \times 6 \times 60 \times 24 \quad (7.13)$$

7.2.5 **Economic**

Cycle Background

In this section, we provide a comparative analysis on the token systems of Bitcoin, Ethereum and EOS with a focus on their economic models in order to design a better token system for the ABC ecosystem.

Figure 7.3 shows a simplified bitcoin flow model. When the blockchain node mines, bitcoin flows from the miners to the exchange, and then flows to the investors. The bitcoin that flows out does not return to the bitcoin network, re-creating value for the system. Therefore, the economy of Bitcoin is a non-closed, open flow model.

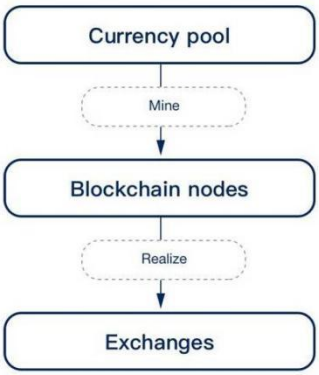

Figure 7.3: Bitcoin Economic Operation Model

Figure 7.4 shows a simplified Ethereum economic operation model. Compared with Bitcoin, Ethereum's economic model has made significant progress which mainly reflected in its recycling of fuel Gas, making the economic flow forming a simple closed loop.

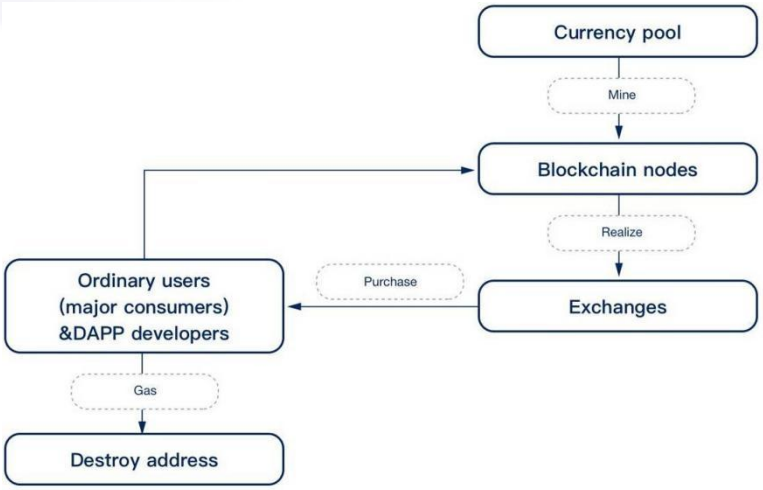

Figure 7.4: Ethereum Economic Circulation Model

Figure 7.5 shows the expected economic transfer model for EOS. On the supply side of the currency, it is still miners producing the money. On the demand side of the currency, the fuel consumption of Ethereum was replaced with the deposit of the blockchain accounts. Compared with fuel, deposit depends on the activity level of the entire blockchain community, the number and size of developers that are unpredictable at the beginning of the project. Once the project performs poorly, the demand for currency drops sharply. In other words, the recovery method of deposit is not direct or effective enough. Despite the success of these three projects, there are some notable flaws in their economic models. From the comparative analysis described above, it can be seen that their ecologies do not form a closed loop of tokens, which is vital to any economic model. Therefore, we propose the CFE token system for the ABC ecosystem to address this common pain point of existing blockchain projects.

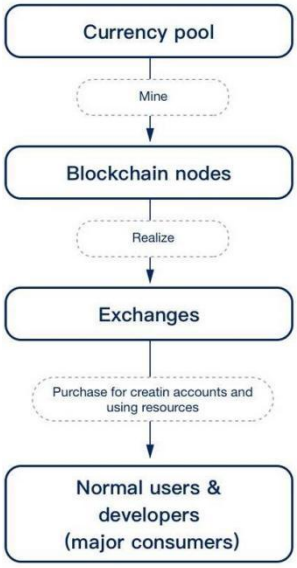

Figure 7.5: EOS Economic Circulation Model

CFE Token System

For ease of understanding, the CFE is first disassembled into two parts, the CyberLand layer and the CyberCube layer.

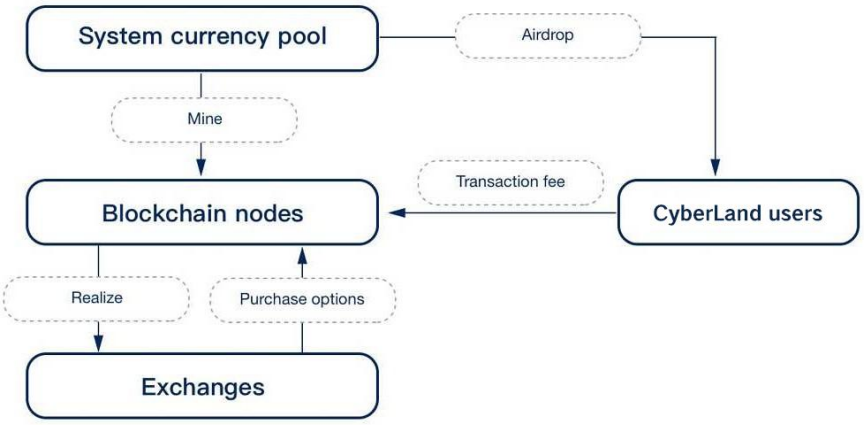

Figure 7.6: CyberLand Layer Economic Cycle Model

CyberLand Layer

As shown in **Figure 7.6**, the CyberLand forms a closed loop of currency circulation around the blockchain nodes. The blockchain nodes obtain the currency rewards by "mining" and collecting the transaction fee of the CyberLand

users. The nodes can sell the acquired currency to the exchange to realize the cash, or can save and convert it into options to obtain more rewards. If it is needed by the nodes, it is also feasible to buy the currency directly from the exchange and deposit it as options.

Figure 7.7 shows the economic cycle model of the CyberCube layer, which forms a closed loop of currency around gamers and game developers.

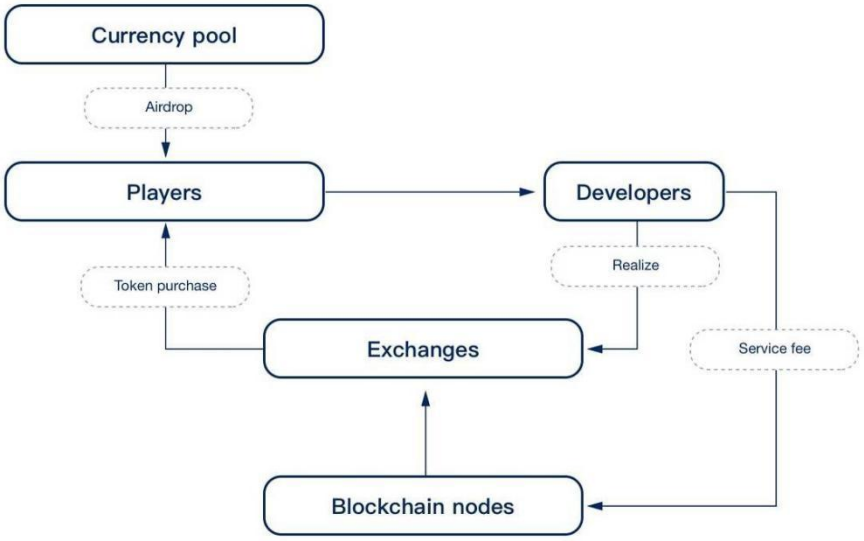

Figure 7.7: CyberCube Layer Economic Cycle Model

For the players, they can get a certain amount of "CC" airdrops by playing games, but this part of the currency cannot meet the demand of heavy rollers. The heavy rollers will choose to buy "ABC" from the exchanges and change them into "CC" to pay the game developers. The "ABC" that the developers received can be realized as cash in the exchanges.

For game developers who use services and resources of blockchain, they also need to pay a portion of the currency to blockchain nodes.

CFE Token System

Figure 7.8 combines the CyberLand and CyberCube to give a fully ecological economic cycle model. As can be seen from the model, the entire ecosystem

forms a complete closed loop around blockchain nodes, CyberCube and exchanges.

As shown in **Figure 7.8**, the ABC Ecosystem builds a complete closed-loop economic model around blockchain nodes, CyberCube and exchanges, providing reasonable access and exit for each economy participant. Compared with the open economic models of Bitcoin and Ethereum, ABC's economic model can enhance the liquidity and stability of tokens.

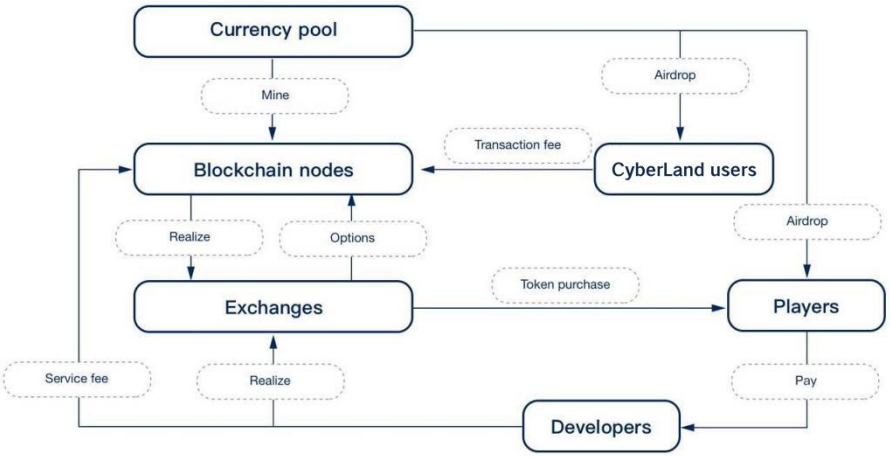

Figure 7.8: Full Ecological Economic Cycle Model

7.2.6 Foundation Mechanism

Throughout the ecology, participants, especially CyberCube participants, are extremely sensitive to the market price of tokens. Stable currency price is particularly important in ABC's ecology. We utilize the foundation reserve pool pricing and market pricing to ensure that the token price is stable. The source of the Foundation Reserve Pool consists of two parts, namely a portion of the legal currency cash reserve for initial financing and a portion of the token reserved for the issuance of the token (about 20% of the total circulation).

When there are n ABCs in circulation in the market, if the market price of ABC is higher than the expected theoretical price, the market will purchase ABC from the foundation to achieve a price equilibrium; When the ABC price is lower than the theoretical price, the demand for all tokens will be completed through open market transactions. The system will not sell. When the market price is less than

0.5 of the expected theoretical price, the foundation will buy back some of the ABCs, reduce the number of ABCs circulating in the market, and make the market price return to the expected price.

The link among currency, inflation, and currency-policy

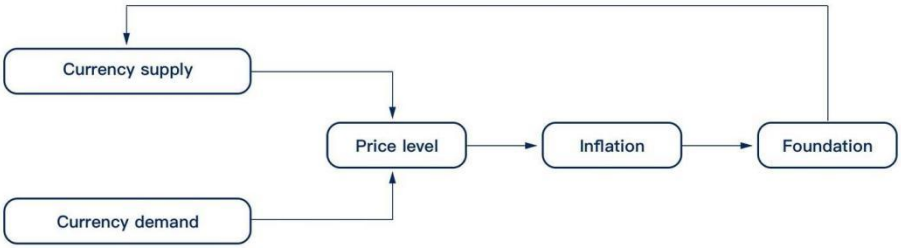

Figure 7.9: Relationship among Currency, Inflation, and The Foundation Mechanism

According to the classical economic theory, money supply and demand jointly determine the current price level, and the price change determines the inflation rate [127]. In the ABC ecosystem, the currency supply side is supplemented by 4% of the additional currency each year, in addition to the currency that was first issued and circulated in the market. The currency demand side relies on the option deposit of blockchain nodes, the needs of CyberCube game developers and gamers. This supply and demand relationship determines the purchasing power of the token, which in turn affects the price level.

The price level determines the ecological inflation rate. In the blockchain ecosystem, the money supply side can be regarded as a basically constant value. If there is a situation of excessive demand or weakness on the demand side of the currency, then the hyperinflation or deflation happens resulted in a serious impact on the ecology of the system. Therefore, it is necessary to establish a foundation mechanism and use a reasonable monetary policy to influence the amount of money on the money supply side when necessary, so that the entire ecology maintains a moderate inflation rate.

7.3 Analysis

A healthy blockchain economic operation model needs to include at least the following 4 aspects: transaction fee, network security, price stability mechanisms, and economic cycles. Next, we will use the method of comparison to illustrate the specifics of the ABC blockchain ecology in these four aspects.

7.3.1 Transaction Fee

The transaction fee is the fee that the blockchain user needs to pay to the blockchain nodes or the system when the transaction is initiated. The fee may be transferred to the blockchain nodes and may also be consumed as fuel. The original intention of transaction-fee design is to allow users to use blockchain resources as a paid service, prevent the abuse of resources, and maintain the stability of the system. Typical representatives of this type of design are the Bitcoin network and Ethereum. Users must pay a certain transaction fee or fuel for the transaction. However, the problem is that charging the transaction fee will cause the blockchain usage threshold to be pulled higher, causing many potential users to be rejected. Next, EOS tries to solve the problem by cancelling the user's transaction fee. However, if you want to use the blockchain resources, you need to freeze a certain amount of tokens in the account. The amount of resources available and the amount of frozen tokens are positively related [128]. This approach seems not requiring users to spend transaction fee. However, if users want to use blockchain resources, they still need to have a certain amount of tokens, which is a threshold itself. Therefore, EOS did not solve this problem very well.

In the ABC ecosystem, CyberLand is truly free to users. The system issues "CC" to users, and users chatting paying "CC" to nodes. The amount of "CC" received by nodes measures the workload of nodes. Finally, the system issues the corresponding amount of tokens to nodes. For resource abuse issues, we use a secure resource allocation pool and a game-like human-machine identification scheme to circumvent, separating the transaction fee from the use of resources and making the user layer truly free.

7.3.2 Network Security

The challenge of blockchain network security mainly comes from the blockchain node itself. In the absence of a clear penalty mechanism and sufficient positive incentive stimulus, blockchain nodes tend to behave maliciously for their own interests. There are usually two solutions to this problem: first, the deposit mechanism. That is, when the server is registered as a node, it needs to pay a certain amount of deposit to the blockchain system. If the node has a malicious error such as a packaging error, the deposit will be all or partially deducted. For example, both EOS and Steemit use this mechanism. Second, income turns into option partially. That is, the revenue of the blockchain node, whether it comes from mining or transaction fee, some of which are issued in the form of options that needed to be frozen for a period of time before exercising. We adopt the second method. Compared with the first one, there are two advantages of income partially turning into option.

1. Reduce node access costs and thresholds. Contrary to the design of the EOS super node, the node does not need to pay the deposit at the time of registration. The node can concentrate on access and service without worrying about how many tokens are self-owned, further ensuring the decentralization of the blockchain system.

2. Slowing down the rate of inflation. Under the ABC Ecology, the transaction fee and mining fee issued to the nodes are all new empty coins. Setting a part of the option currency not only guarantees the same economic benefits among nodes, but also slows down the inflation rate, which is conducive to firm currency prices.

7.3.3 Price Stability Mechanism

When electronic currency is issued, if it is simply adjusted by the market itself, it will often lead to instability of the currency price, and the magnitude is large. This is not a good thing for providers and users of blockchain resources, because the use of resources and becoming nodes often require consumption or possession of tokens. Therefore, Steemit proposed a fiscal policy similar to bonds [129]. Steemit issues bonds to investors. Bonds can be exchanged for equivalent tokens.

Community leaders are responsible for maintaining the 1:1 equivalence between bonds and the US dollar. By means of bonds, Steem is linked to the US dollar. In this way, the interest rate of the bond can be used as a financial means to adjust the amount of tokens in the market, thereby affecting the price of the tokens. In addition, the foundation mechanism is also a common method. The foundation mechanism refers to a pool of funds and tokens that are artificially established by the blockchain development team to ensure the health of the economic order. The entire ecological economy is regulated by issuing pricing and repurchasing tokens. According to the theory of sticky information [130], a transparency and simple fiscal policy can improve the efficiency of monetary policy. The bond system is obviously more complicated than the foundation mechanism. For the foundation mechanism, ordinary investors only need to know the value of the token and that there are real gold and silver backing it up. The foundation mechanism enhances the public's confidence in the tokens, and stabilize the currency. Therefore, we first use the foundation mechanism as a price stability tool for ABC Ecology.

Sticky Information Theory

The core idea of the sticky information theory is that economic entities need to bear certain costs in the process of collecting, understanding and absorbing information [131]. This has caused the economic entity not to continuously update the decision information set, still using the original plan and expired information. A simple understanding is the lag in information on economic decisions. Therefore, high transparency is conducive to enhancing the stabilizing effect of monetary policy. The central bank should communicate high-quality information to the public, and the central bank's intentions should be fully transparent and generally beneficial to social welfare. However, the increase in the transparency of sensitive information will have a negative impact on social welfare.

In the sticky information model, the transparency of monetary policy can improve the efficiency of monetary policy. If policy information is complex, people will ignore it because they think it is not worth spending too much effort

on this information.

In the sticky information model, central bank announcements are influential [132]. For example, because the central bank made an announcement, before the central bank adopted a tightening policy, the public might already have adjusted the plans. As a result, the central bank announced a reduction in the money supply growth rate before taking action, which would lead to a faster inflation response and a smaller loss of output comparing to a sudden reduction in the money supply growth rate.

As to the CFE token system, many designs are conducted under the guidance of the sticky information theory. For example, the reason of a foundation system rather than a bond mechanism is used to stabilize monetary value is, in the sticky information model, the transparency of monetary policy can improve the efficiency of monetary policy. If policy information is complex, people will ignore it because they think it is not worth spending too much effort on this information.

7.4 Closed-Loop Token System

This chapter proposes an economic plan based on the ABC technology platform. The goal is to enable all participants in the ecology to perform their duties under reasonable economic incentives and fiscal governance to ensure the healthy operation of the entire ecosystem of the ABC blockchain. The ABC blockchain platform, CyberLand instant messenger and the CyberLand-based CyberCube community are closely linked and functioning, by the mechanisms of the free economic market and macro-control policies. Finally, through the comparative economic analysis of ABC blockchain and other well-known blockchain projects, the completeness and liquidity of the ABC economic model is crystal clear.

CONCLUSION

In this paper, we provide a three-layer pyramid-shaped blockchain ecosystem, consisting of the underlying level ABC blockchain, the middle level CyberLand and the top-level CyberCube. The ABC blockchain is the foundation of the entire system. To extend scalability, ABC blockchain proposes a "Double-Ring" network topology design ABN. It decouples network management logic from business logic. Thus, a flexible and scalable decentralized P2P system solution is realized. In terms of consensus mechanism, ABC blockchain proposes a novel next-generation consensus protocol ACP. It uses randomness beacon and Verifiable Random Function to fairly select a random committee and then achieves agreement among the committee efficiently based on the PBFT* protocol. Beside the outstanding throughput and latency performance, the rigorous four-phase consensus protocol offers high security guarantees for ABC blockchain.

As a decentralized application platform, we offer an ABC blockchain based IM application and the blockchain-oriented gaming platform CyberCube, particularly for GameFi, on top of the IM. To ensure the healthy operation of the ecosystem and stabilize the value of the tokens, we integrate CFE economic system into free markets including token issuance mechanism, consensus economics, economic cycle model. Powered with the economic incentive of CFE model, ABC blockchain is able to link the participants in the three different layers and promote the healthy development of ABC ecology.

In addition, we offer a novel artificial intelligent based built-in governance mechanism in ABC. The ABC AI Governance Module is an intelligent module for privilege management, risk analysis and anomaly detection on the ABC. It comprehensively utilizes multiple AI technologies and deeply adapts the characteristics of ABC blockchain to automatically control the access, elimination and the role conversion of nodes on ABC; It effectively improves the overall efficiency of ABC systems. Moreover, it can reduce the risk of ABC being subject to external surveillance, attacks, and large-scale node failures. In spite of all these, there are many challenges faced by the enhancement of ABC blockchain

ecosystem. For instance, in order to make the ABC AI Governance module more secure and intelligent, we need to develop more efficient privacy preserving decentralized learning algorithms. In the aspect of ecological construction, we planned to combine AI self-management technology with ABC economic model to make token airdrops, option unlocking and the supply and demand adjustment more effectively and intelligent. These cutting-edge research into the fundamental problems of blockchain is the focus of our future work, which is also an inspiring exploration for the future of blockchain industry.

REFERENCES

[1] Nakamoto, S. Bitcoin A peer-to-peer electronic cash system. http://www.bitcoin.org/bitcoin.pdf, 2009.

[2] Kim, J., et al. Technology-driven, highly-scalable dragonfly topology. Computer Architec-ture, ISCA'08. 35th International Symposium on. IEEE, 2008.

[3] Wikipedia contributors. HoffmanSingleton graph, Wikipedia, The Free Encyclope- dia, https://en.wikipedia.org/w/index.php?title=Hoffman

[4] Besta, M., and Torsten H. Slim fly: A cost effective low-diABC ter network topology. Pro-ceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE Press, 2014.

[5] Fischer, M.J., Lynch, N.J. and Paterson, M. Impossibility of distributed consensus with one faulty process, in ACM SIGACT-SIGMOD, 1983.

[6] Castro M. and Liskov, B. Practical Byzantine fault tolerance and proactive recovery, ACM Trans. Comput. Syst., vol. 20, no. 4, pages 398 461, 2002.

[7] Dwork, C., Lynch, N.A. and Stockmeyer, L.J. Consensus in the presence of partial syn-chrony, J. ACM, vol. 35, no. 2, pages 288323, 1988.

[8] Lamport, L. Paxos made simple, SIGACT News, vol. 32, no. 4, pages 1825, 2001.

[9] Ongaro, D. and Ousterhout, J.K. In search of an understandable consensus algorithm, in USENIX Annual Technical Conference, 2014.

[10] Lamport, L., Shostak, R.E. and Pease, M.C. The Byzantine generals problem, ACM, vol. 4, no. 3, pages 382401, 1982.

[11] Boldyreva, A. Threshold signatures, multi-signatures and blind signatures based on the Gap-Diffie-Hellman-group signature scheme, Workshop on

Theory and Practice in Public Key Cryptography (PKC), 2003.

[12] Amir, Y., Coan, B.A., Kirsch, J. and Lane, J. Prime: Byzantine replication under attack, IEEE Trans.DependableSec.Comput., vol. 8, no.4, pages 564577, 2011.

[13] Shoker A. and Bahsoun, J.-P. BFT for three decades, yet not enough! http://haslab.uminho.pt/ashoker/files/ shokerbft3decadestr.pdf, 2009.

[14] Ho, C. Reducing costs of Byzantine fault tolerant distributed applications, [Online]. Availa-ble: http://ecommons.library.cornell.edu/ handle/1813/30754, 2011.

[15] Schneider, F.B. Implementing fault-tolerant services using the state machine approach: A tutorial, ACM Comput. Surv, vol. 22, no. 4, pages 299319, Dec. 1990.

[16] Yin, J., Martin, K.-P., Venkataramani, A., Alvisi, L. and Dahlin, M. Separating agreement from execution for byzantine fault tolerant services, in Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, ser. SOSP 03. New York, NY, USA, pages 253267, 2003.

[17] Kotla, R., Alvisi, L., Dahlin, M., Clement, A. and Wong, E. Zyzzyva: Speculative byzantine fault tolerance, ACM Trans. Comput. Syst., vol. 27, no. 4, pages 7:17:39, Jan. 2010.

[18] Veronese, G.S., Correia, M., Bessani, A. and Lung, L.C. Spin ones wheels? byzantine fault tolerance with a spinning primary, in 28th IEEE International Symposium on Reliable Dis-tributed Systems, 2009. SRDS 09, Sep. 2009.

[19] Mao, Y., Junqueira, F.P. and Marzullo, K.: Mencius building efficient replicated state ma-chines for WANs, in OSDI, vol. 8, 2008.

[20] Pease, M., Shostak, R. and Lamport, L. Reaching agreement in the presence of faults. J.ACM. 27(2), pages 228234, 1980.

[21] Ben-Or, M. Another advantage of free choice(extended abstract):Completely asynchronous agreement protocols. In: PODC 1983, pages 2730, ACM, New York, 1983.

[22] Bracha, G. An asynchronous [(n - 1)/3]-resilient consensus protocol. In: PODC 1984, pages 154162, ACM, New York, 1984.FF2

[23] Martin, J.-P. and Alvisi, L. Fast Byzantine consensus. IEEE Transactions on Depend- able and Secure Computing 3(3), pages 202215, 2006.

[24] Borran, F. and Schiper, A. A Leader-free Byzantine Consensus Algorithm. Technical report, EPFL, 2009.

[25] Attiya, H. and Welch, J. Distributed Computing: fundamentals, simulations, and advanced topics. John Wiley & Sons, Chichester, 2004.

[26] Moses, Y., Waarts, O. Coordinated traversal: (t + 1)-round byzantine agreement in polyno-mial time. In: FOCS, pages 246255, 1988.

[27] Guerraoui, R. and Vukoli, M. Refined quorum systems. Distributed Computing, 23(1): pages 142, 2010.

[28] Lamport, L. Fast byzantine paxos. United States Patent 7620680, filed August 15, 2002, issued November 17, 2009.

[29] Oki, B.M. Viewstamped replication for highly available distributed systems. Technical Re-port MIT/LCS/TR-423, MIT Laboratory for Computer Science, August 1988.

[30] Lamport,L. Generalized Consensus and Paxos,TechnicalReport, MSR-TR-2005- 33, Mar. 2005.

[31] Chen, J. and Micali, S. Algorand. Technical report, URL http://arxiv.org/abs/1607.01341, 2017.

[32] Brockman. G. Stellar, https://stripe.com/blog/stellar, July 2014.

[33] Douceur, J.R. The Sybil attack. In Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS02), Cambridge, MA, Mar. 2002.

[34] Micali, S. Fast and furious Byzantine agreement. In Proceedings of the Innovations in Theo-retical Computer Science (ITCS) Conference, 2017.

[35] Micali, S., Rabin, M.O. and Vadhan, S.P. Verifiable random functions. In

Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science (FOCS), New York, NY, Oct. 1999.

[36] Helix. https://orbs.com/wp-content/uploads/2018/07/Helix-Consensus-Paper- V1.2-1.pdf, 2018.

[37] Turpin R. and Coan, B.A. Extending binary Byzantine agreement to multivalued Byzantine agreement. Information Processing Letters, 18(2): pages 7376, Feb. 1984.

[38] Berman, P., Garay, J. A. and Perry, K. J. Bit optimal distributed consensus. Computer sci-ence: research and applications, pages 313321, 1992.

[39] Coan B.A. and Welch, J.L. Modular construction of a byzantine agreement protocol with optimal message bit complexity. Inf. Comput., 97(1): pages 6185, 1992.

[40] Fitzi, M. and Hirt. M. Optimally efficient multi-valued byzantine agreement. In PODC 06: Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed compu-ting, pages 163168, New York, NY, USA, 2006.

[41] Liang G. and Vaidya, N. Capacity of byzantine agreement: Complete characteriza- tion of the four node network. Technical Report, CSL, UIUC, April 2010.

[42] Damgard, I. and Koprowski, M. Practical threshold RSA signatures without a trusted dealer, in EUROCRYPT, 2001.

[43] Dolev, D., Dwork, C. and Stockmeyer, L. On the minimal synchronism needed for distribut-ed consensus, J. ACM, vol. 34, no. 1, pages 7797, Jan. 1987.

[44] Correia, M., Neves, F.F. and Verssimo, P. From consensus to atomic broadcast: Time-free byzantine-resistant protocols without signatures, Comput. J., vol. 49, no. 1, pages 8296, Jan. 2006.

[45] Reed, B. and Junqueira, F.P. A Simple Totally Ordered Broadcast Protocol. In Proceedings of the 2Nd Workshop on Large-Scale Distributed Systems and Middleware, pages 2:12:6, Sept. 2008.

[46] Cascudo, I. and David, B. SCRAPE: Scalable randomness attested by public entities, in Springer ACNS, 2017.

[47] Amir, Y., Coan, B., Kirsch, J., Lane, J. Byzantine replication under attack. In: DSN08. Pag-es 97206, 2008.

[48] Clement, A., Wong, E., Alvisi, L., Dahlin, M., Marchetti, M. Making Byzantine fault toler-ant systems tolerate Byzantine faults. In: NSDI09, Berkeley, CA, USA, USENIX Associa-tion, pages 153168, 2009.

[49] Yang, L. LinBFT: Linear-Communication Byzantine Fault Tolerance for Public Blockchains, https://arxiv.org/pdf/1807.01829.pdf, Date accessed: 2018-09-20.

[50] ONeil, P. Cheng, E., Gawlick, D. and ONeil, E. The log-structured merge-tree (lsm- tree). Acta Informatica, 33(4): pages 351385, 1996.

[51] Ghemawat, S. and Dean, J. 2012. LevelDB, A fast and lightweight key/value database li-brary by Google. http://code.google.com/p/leveldb/, 2012. Retrieved June 20, 2015.

[52] Lu, L., Pillai, T.S., Arpaci-Dusseau, A. and Arpaci-Dusseau, R.H. WiscKey: Separating keys from values in ssd-conscious storage. In 14th USENIX Con-ference on File and Storage Technologies (FAST 16). USENIX Association, Santa Clara, CA, 133148. https://www.usenix.org/conference/fast16/ technical- sessions/presentation/lu, 2016.

[53] Lamport, L. The part-time parliABC nt. Technical report, DEC SRC, 1989.

[54] Ongaro, D, and Ousterhout, J. In Search of an Understandable Consensus Algorithm. Draft on October 7, 2013.

[55] Liskov, B.,Cowling, J. Viewstamped replication revisited. Tech. Rep. MIT-CSAIL- TR-2012-021, MIT, July 2012.

[56] Copeland, C. and Zhong, H. BFTRaft.

[57] Chen, J. and Micali, S. Algorand: The efficient and democratic ledger, URL: https://arxiv.org/pdf/1607.01341.pdf, 2016.

[58] Kogias, E., Jovanovic, P., Gailly, N., Khoffi, I., Gasser, L. and Ford, B. Enhancing Bitcoin security and performance with strong consistency via collective signing. In 25th USENIX Security Symposium, pages 279296. USENIX Association, 2016.

[59] Chainspace. http://www0.cs.ucl.ac.uk/staff/M.AlBassam/uploads/chainspace- slides.pdf.

[60] Redis. https://github.com/antirez/redis.

[61] Sikka, V., Frber, F., Lehner, W., Cha, S.K., Peh, T. and Bornhvd, C. Efficient Transaction Processing in SAP HANA Database - The End of a Column Store Myth. SIGMOD, 2012.

[62] MVCC. https://en.wikipedia.org/wiki/Multiversion concurrency control.

[63] Ylonen, T. Concurrent Shadow Paging: Snapshots, Read-Only Transactions, and On-The-Fly Multi-Level Incremental Dumping, Technical Report 1993/TKO-B104, Helsinki Univer-sity of Technology, 1993.

[64] MPT. https://blog.ethereum.org/2015/11/15/merkling-in-ethereum/.

[65] CoW. https://en.wikipedia.org/wiki/Copy-on-write.

[66] Johnson, R., Pandis, I. and Ailamaki, A. Improving OLTP scalability using speculative lock inheritance. PVLDB, 2(1), 2009.

[67] Athanassoulis, M., Kester, M. S., Maas, L. M., Stoica, R., Idreos, S., Ailamaki, A. and Calla-ghan, M. Designing Access Methods: The RUM Conjecture. In EDBT, 2016.

[68] Athanassoulis, M. and Idreos. S. Design Tradeoffs of Data Access Methods. In SIGMOD, 2016.

[69] Facebook. RocksDB. https://github.com/facebook/rocksdb.

[70] Dinh, T. T. A., Wang, J., Wang, S., Chen, G., Chin, W.-N., Lin, Q., Ooi, B. C., Ruan, P., Tan, K.-L., Xie, Z. and Zhang, M. UStore: A distributed storage with rich semantics. CoRR, abs/1702.02799, 2017.

[71] Forkbase. http://www.vldb.org/pvldb/vol11/p1137-wang.pdf.

[72] Rodeh, O. B-trees, shadowing, and clones. ACM Transactions on Storage (TOS), 3(4), 2008.

[73] Gray, J. and Reuter, A. Transaction Processing: Concepts and Techniques. Morgan, Kauf-mann, 1993.

[74] Bender, M. A., Farach-Colton, M, Fineman, J. T., Fogel, Y. R., Kuszmaul, B. C. and Nelson, J. Cache-Oblivious Streaming B-trees. In SPAA, 2007.

[75] mLSM: Making Authenticated Storage Faster in Ethereum. https://www.usenix.org/conference/hotstorage18/presentation/raju.

[76] Marlinspike, M. and Perrin, T. The X3DH Key Agreement Protocol, https://www.signal.org/docs/specifications/x3dh/.

[77] Perrin, T. and Marlinspike M. The Double Ratchet Algorithm, https://www.signal.org/docs/specifications/doubleratchet/.

[78] [3] Perrin, T. The XEdDSA and VXEdDSA Signature Schemes, https://signal.org/docs/specifications/xeddsa/.

[79] Marlinspike, M. and Perrin, T. The Sesame Algorithm: Session Management for Asynchro-nous Message Encryption, https://signal.org/docs/specifications/sesame /.

[80] WhatsApp-Security-Whitepaper, https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf

[81] Maurer, F., Neudecker, T. and Florian, M. Anonymous Coin Join Transactions with Arbi-trary Values, IEEE Trustcom/BigDataSE/ICESS, 2017.

[82] Ruffing, T. and Moreno-Sanchez, P. Mixing Confidential Transactions: Compre- hensive Transaction Privacy for Bitcoin, International Conference

on Financial Cryptography and Data Security

[83] libsnark: a C++ library for zkSNARK proofs, https://github.com/scipr-lab/libsnark.

[84] Ben-Sasson, E., Chiesa, A. Tromer, E., Virza and M. Succinct, Non-Interactive Zero Knowledge for a von Neumann Architecture,USENIX Security, (23rd USENIX Security Symposium), 2014.

[85] Gennaro, R., Gentry, C., Parno, B. and Raykova, M. Quadratic Span Programs and Succinct NIZKs without PCPs, International Association for Cryptologic Research, 2013.

[86] Bender, A., Katz and J., Morselli R. Ring Signatures: Stronger Definitions, and Construc-tions without Random Oracles, Theory of Cryptography pages 60-79, TCC 2006.

[87] David, C. Blind signatures for untraceable payments, Advances in Cryptology, pages 199-203.

[88] Pointcheval, D. and Stern, J. Security Arguments for Digital Signatures and Blind Sig-natures, Journal of Cryptology Volume 13, Issue 3, pages 361396, June 2000.

[89] Dingledine, R., Mathewson, N., and Syverson, P. Tor: The Second-Generation Onion Rout-er, Security '04 Technical Program.

[90] https://github.com/Shadowsocks.

[91] https://shadowsocks.org/en/index.html.

[92] Deng, Z., Liu, Z., Chen, Z., and Guo, Y. The Random Forest based Detection of Shadow-sock's Traffic, 9th International Conference on Intelligent Human- Machine Systems and Cy-bernetics (IHMSC), 2017.

[93] Fouque, P., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., and Zhang, Z. Falcon: Fast-FourierLattice-based Compact Signature sover NTRU, NIST Post-Quantum Cryptography

Project on November 30th, 2017.

[94] About Falcon https://falcon-sign.info/.

[95] Aggarwal, D., Brennen, G., Lee, T., Santha, M. and Tomamichel, M. Ledger, Quantum at-tacks on Bitcoin, and how to protect against them, [S.l.], v. 3, oct. 2018.

[96] Johnson, D., Menezes, A. and Vanstone, S. The Elliptic Curve Digital Signature Algorithm (ECDSA), International Journal of Information Security Volume 1, Issue 1, pages 3663, August 2001.

[97] Secure Development Lifecycle, https://www.owasp.org/.

[98] Ward, R. and Beyer, B. BeyondCorp: A New Approach to Enterprise Security, https://cloud.google.com/beyondcorp/.

[99] Conti, M., Kumar, S., Lal, C. and Ruj, S. A Survey on Security and Privacy Issues of Bitcoin. In (IEEE) Communications Surveys and Tutorials, 20(4): pages 3416-3452, 2018.

[100] Conti. M., Li Q., Maragno A. and Spolaor R. The Dark Side (-Channel) of Mobile Devices: A Survey on Network Traffic Analysis. In (IEEE) Communications Surveys and Tutorials, 20(4): pages 2658-2713, 2018.

[101] Kloft, M. and Laskov, P. Online anomaly detection under adversarial impact. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pages 405-412. 2010.

[102] Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I. and Tygar, J.D. Adversarial machine learning. In Proceedings of the 4th ACM workshop on Security and artificial intelligence, pages 43-58. ACM, 2011.

[103] Mitra, A., Richards, J., Bagchi S., and Sundaram, S. Resilient Distributed State Estimation with Mobile Agents: Overcoming Byzantine Adversaries, Communication Losses, and In-termittent Measurements. Autonomous Robots, 2018.

[104] Dwork, C. and Roth, A. The algorithmic foundations of differential privacy. Foundations and Trends R in Theoretical Computer Science, 9(34): pages 211407, 2014.

[105] Dwork, C. Differential privacy. In Automata, Languages and Programming, 33rd Interna-tional Colloquium, ICALP 2006, Venice, Italy, Proceedings, Part II, pages 112. July 10-14, 2006.

[106] Han, E., Karypis G., Kumar, V. and Mobasher, B. Hypergraph Based Clustering in HighDi-mensional Data Sets: A Summary of Results, IEEE Bulletin of Technical Committee on Data Engineering, pages 1-8. 1998.

[107] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy, 2016.

[108] Omidshafiei, S., Pazis, J., Amato, C., How and J., Vian, J. Deep Decentralized Multi-task Multi-Agent Reinforcement Learning under Partial Observability, in The International Con-ference on Machine Learning (ICML), Sydney, Australia, August 2017.

[109] Boyd, S., Parikh, N., Chu, E., Peleato, B. and Eckstein, J. Distributed optimization and sta-tistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning, 3 pages 1122. 2011.

[110] Li, Y. Deep reinforcement learning: An overview. arXiv preprint arXiv:1701.07274.

[111] Puterman, M. L. Markov Decision Processes: Discrete stochastic dynamic programming. John Wiley & Sons.

[112] Shoham, Y., Powers, R. and Grenager, T. Multi-agent reinforcement learning: A critical sur-vey. Technical Report, 2003.

[113] Soliman, A., Rahimina, F., and Girdzijauskas, S. Stad: Stateful Diffusion for Linear Time Community Detection. Under review, submitted to ICDCS.

[114] Soliman, A. and Girdzijauskas, S. AdaGraph: Adaptive Graph-based Algorithms for Spam Detection in Social Networks. In International

conference On Networked Systems (NETYS 2017), Springer, pages 338-354, 2017.

[115] Soliman, A. and Girdzijauskas, S. DLSAS: Distributed Large-Scale AntiSpam framework for Decentralized Online Social Networks. Invited paper in the 2nd IEEE International Con-ference on Collaboration and Internet Computing, IEEE, pages 363-372, 2016.

[116] Biryukov, A., Feher, D. and Khovratovich, D. Guru: Universal Reputation Module for Dis-tributed Consensus Protocols. IACR Cryptology ePrint Archive 2017: 671. 2017.

[117] Akoglu, L., Tong, H., and Koutra, D. Graph based anomaly detection and description: a sur-vey. DMKD 29, pages 626688, 3, 2015.

[118] Yanardag, P. and Vishwanathan, S. Deep Graph Kernels. In the 21th ACM SIGKDD Inter-national Conference, pages 13651374, New York, New York, USA, ACM Press, 2015.

[119] Scarselli, F., Gori, M., Tsoi, A., Hagenbuchner, M. and Monfardini, G. The Graph Neural Network Model. IEEE Transactions on Neural Networks, 2009.

[120] Kipf, T. and Welling, M. Semi-supervised classification with graph convolutional networks. In ICLR, 2017.

[121] Sutton, R. S. and Barto, A. G. Reinforcement Learning: An Introduction (2nd Edition). MIT Press. 2018.

[122] Buterin, V. Ethereum (ETH): A next generation smart contract & decentralized application platform, 2017.

[123] Schwartz D., Youngs, N. and Britto, A. The ripple protocol consensus algorithm, 2014.

[124] Hardin, G. The Tragedy of the Commons, Science, Vol. 162, No. 3859, pages 1243- 1248, 1968.

[125] Ahn, L., Blum, M. and Langford, J. Telling humans and computers apart

automatically, Communications of the ACM, Vol. 47, No. 2, pages 57-60, 2004.

[126] Schneeweis, T. and Spurgin, R. The Benefits of Index Option-Based Strategies for Institu-tional Portfolios, The Journal of Alternative Investments, Vol.3, No.4, pages 44 52, 2001.

[127] Reisman, G. Capotalism: A treatise on economics, JABC son Books, Ottawa, Illinois, 1998.

[128] Grigg, I. EOS An Introdution, 2017.

[129] Steem: An incentivized, blockchain-based, public content platform, 2017.

[130] Mankiw, N. The Inexorable and Mysterious Tradeoff Between Inflation and Unemployment, Economic Journal, Vol. 111, No. 471, pages C45-C61, 2001.

[131] Mankiw, N. and Reis, R. Sticky Information: A model of monetary nonneutrality and structural slumps, 2001.

[132] Friedman, M. The Role of Monetary Policy, ABC rican Economic Review, Vol. 58, March, pages 1-17, 1968.