This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3135414, IEEE Internet of Things Journal

Resource Sharing and Trading of Blockchain Radio Access Networks: Architecture and Prototype Design

Yuwei Le, *Student Member, IEEE*, Xintong Ling, *Member, IEEE*, Jiaheng Wang, *Senior Member, IEEE*, Ruiwei Guo, *Student Member, IEEE*, Yongming Huang, *Senior Member, IEEE*, Cheng-Xiang Wang, *Fellow, IEEE*, Xiaohu You, *Fellow, IEEE*

***Abstract*—Recently, blockchain radio access network (B-RAN) arises as an innovative paradigm for the sixth-generation (6G) wireless communications to build cooperative trust, aggregate wireless resources, and schedule inter- and intra-network tasks among independent network entities. It establishes an open platform based on blockchain to provide diverse wireless services and applications, such as radio access, Internet of Things (IoT), and mobile edge computing, via trusted interactions with enhanced security and efficiency. As a distinctive feature, B-RAN enables secure and efficient resource sharing and trading by aggregating, pooling, and coordinating resources from multiple resource hosts and owners across subnetworks. Therefore, an implementable architecture along with various functional modules shall be delicately designed. This work aims to establish a unified architecture with enhanced efficiency, security, compatibility, and flexibility for resource sharing and trading in B-RAN. Specifically, we develop a six-layer architecture that incorporates a number of novel features such as enhanced blockchain structures, secure interaction methods, efficient service mechanisms, and scalable transaction patterns. We design a number of pluggable functional modules in each layer to support diverse functions, services, and applications of resource sharing and trading. At last, we implement a practical prototype based on the layered architecture for resource-limited devices. Multiple experiments are presented to verify the performance of the proposed architecture from different aspects.**

***Index Terms*—Blockchain, network architecture, radio access network (RAN), resource sharing and trading.**

I. Introduction

The exponential growth of diverse wireless devices and increasing network scale, density, heterogeneity, and complexity have brought tremendous pressure on wireless networks with limited scarce resources, which has accelerated the deployment of the fifth-generation (5G) mobile communication system and expedited the evolution towards the sixth-generation (6G) era. Due to the separation and isolation, both physically and logically, between various network entities such as communication infrastructures, Internet of Things (IoT) devices, and wireless service equipment, many inter- and intra-network tasks, such as resource sharing and trading, data interactions, device access and authentication, information tracking and supervision, are hindered by a number of security and trust problems [1]. Recently, blockchain has been introduced to establish trust between independent network entities, enforce secure interactions of network participants, and flexibly coordinate and manage wireless resources [2]–[4].

Blockchain technology, initiated in [5], inherently lends itself to store, secure, and operate data in a distributed way while having high credibility in guaranteeing the information integrity and security and the interests of participants [6], [7]. Naturally, blockchain arises as a remedy to solve trust problems of multi-sided interactions in various areas, such as finance, logistics, voting, healthcare, and also wireless networks [1]. Indeed, the Federal Communications Commission (FCC)[1] and a number of whitepapers and reports [1], [8]–[12] have foreseen the critical role of blockchain in future 6G wireless communications. The recent advancements of blockchain-as-a-service (BaaS) [13] and blockchain market [14], [15] have propelled blockchain to transform into a service-oriented multi-sided platform (MSP) [16], which makes blockchain eligible for coordinating multi-lateral business and managing large-scale services. In [17], [18], the authors proposed edge computing solutions for blockchain applications in mobile networks and provided practical and economic insights for resource management and pricing in mobile blockchains. The work [19] developed a blockchain-based trust mechanism and a deep reinforcement learning based computation resource allocation algorithm to improve network security, edge utility, and computational performance.

Notably, towards a trustworthy and secure MSP for 6G wireless networks, blockchain radio access network (B-RAN) [2] was proposed to establish cooperative trust among inherently untrusted network entities without any middleman and create an extensive network of subnetworks [1]. B-RAN manages network access, authentication, authorization, and accounting via trusted interactions with enhanced security and efficiency.

This work was supported in part by the National Key R&D Program of China under Grant 2018YFB1801103, in part by the National Natural Science Foundation of China under Grants 61901111, 61971130, and 61720106003, in part by the Natural Science Foundation on Frontier Leading Technology Basic Research Project of Jiangsu under Grants BK20212001 and BK20192002, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20190331, in part by the Huawei Cooperation Project under FA 2019051081-2021-01, and in part by the Frontiers Science Center for Mobile Information Communication and Security, the EU H2020 RISE TESTBED2 project under Grant 872172.

Corresponding Authors: Jiaheng Wang, Xintong Ling, and Xiaohu You.

Y. Le, X. Ling, J. Wang, R. Guo, Y. Huang, C.-X. Wang, and X. You are with the National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China (e-mail: ywle@seu.edu.cn, xtling@seu.edu.cn, jhwang@seu.edu.cn, leoguo@seu.edu.cn, huangym@seu.edu.cn, chxwang@seu.edu.cn, and xhyu@seu.edu.cn). Y. Le, X. Ling, J. Wang, Y. Huang, C.-X. Wang, and X. You are also with the Purple Mountain Laboratories, Nanjing 210023, China.

[1]https://www.mwcbarcelona.com/, accessed May, 2021.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3135414, IEEE Internet of Things Journal

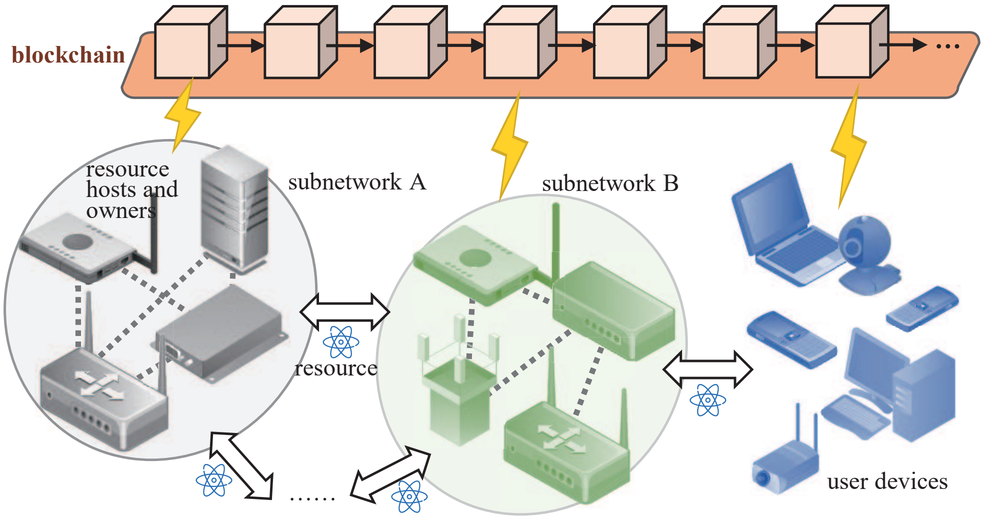


Fig. 1. Overview of resource sharing and trading in B-RAN.

By now, a lot of literature has studied and analyzed B-RAN regarding its features, applications, and performance. The B-RAN paradigm for 6G and its potential application scenarios were discussed at length in [1], [20]. The security and latency trade-off of B-RAN were analytically characterized in [21]. A basic B-RAN service workflow was designed in [22] with some prototype tests. The work [23] proposed a secure blockchain-based multi-hop data routing protocol. A secure grant-free access mechanism named hash access was developed and analyzed in [24], [25].

This work focuses on how to construct B-RAN to facilitate aggregating, pooling, and coordinating wireless resources, e.g., spectrum, computation, and storage or even infrastructures, from multiple hosts and owners across networks, and enabling secure and efficient sharing and trading in wireless networks. As shown in Fig. 1, the resource sharing and trading in B-RAN involves massive interactions both inside and between a group of resource hosts and owners and heterogeneous user devices, which, consequently, requires a practical and efficient architecture. However, a number of challenges have to be overcome: 1) When the blockchain is applied in wireless networks with heterogeneous devices and limited resources, a lightweight architecture design should be considered to reduce the overall energy consumption and increase efficiency [26]; 2) The resource hosts and owners in B-RAN require secure methods to aggregate, pool, and coordinate resources; 3) In B-RAN, resource requests can be conveniently fulfilled via smart contracts, but power-limited mobile devices cannot handle the intensive computation for creation, transmission, validation, and storage of smart contracts; 4) Resource services and applications are built upon trust between network entities, whereas their efficiency is still constrained by lengthy block confirmations and redundant blockchain structures; 5) The scalability of B-RAN is limited by complex on-chain operations of traditional blockchains; 6) The consensus in B-RAN should be flexibly configured to adapt to diverse resource services and applications with different security, latency, and complexity requirements; 7) A suitable mechanism is required to assure and supervise the reliable and honest actions of all network participants. Due to the above problems, most existing blockchain architectures and techniques are not readily applicable to B-RAN. That is why a dedicated and practical architecture design is needed.

This work aims to establish a unified architecture with enhanced efficiency, security, compatibility, and flexibility for resource sharing and trading in B-RAN. Specifically, we develop a six-layer architecture that incorporates a number of novel features such as enhanced blockchain structures, secure interaction methods, efficient service mechanisms, and scalable transaction patterns. The layered architecture can simplify the functionalities of resource sharing and trading in B-RAN by breaking them into smaller and more manageable units, offering greater flexibility and interoperability. Further, we design a number of pluggable functional modules in each layer to support diverse functions, services, and applications of resource sharing and trading. The contributions of this work are summarized as follows.

- We use a lightweight, pluggable, and efficient design in the six-layer architecture of B-RAN, and reduce the extra capital and operating expenditure for deployment.
- We introduce the concept of the *virtual service provider* (VSP) that unites multiple *service providers* (SPs) in B-RAN into one virtualized blockchain-based entity for load balancing and efficient resource sharing.
- We develop a novel smart contract structure, namely the *key-value smart contract*, to reduce the transmission and storage overhead in services.
- We propose a *fast smart contract deployment* (FSCD) mechanism to shorten the block confirmation period of resource services and safeguard the trading history.
- We design an *enhanced hash time-locked contract* (eHTLC) mechanism based on off-chain technologies for scalable and efficient resource sharing and trading in B-RAN.
- We encapsulate consensus mechanisms of B-RAN as *pluggable consensus* modules to achieve more flexible configurations for diverse resource services and applications.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3135414, IEEE Internet of Things Journal

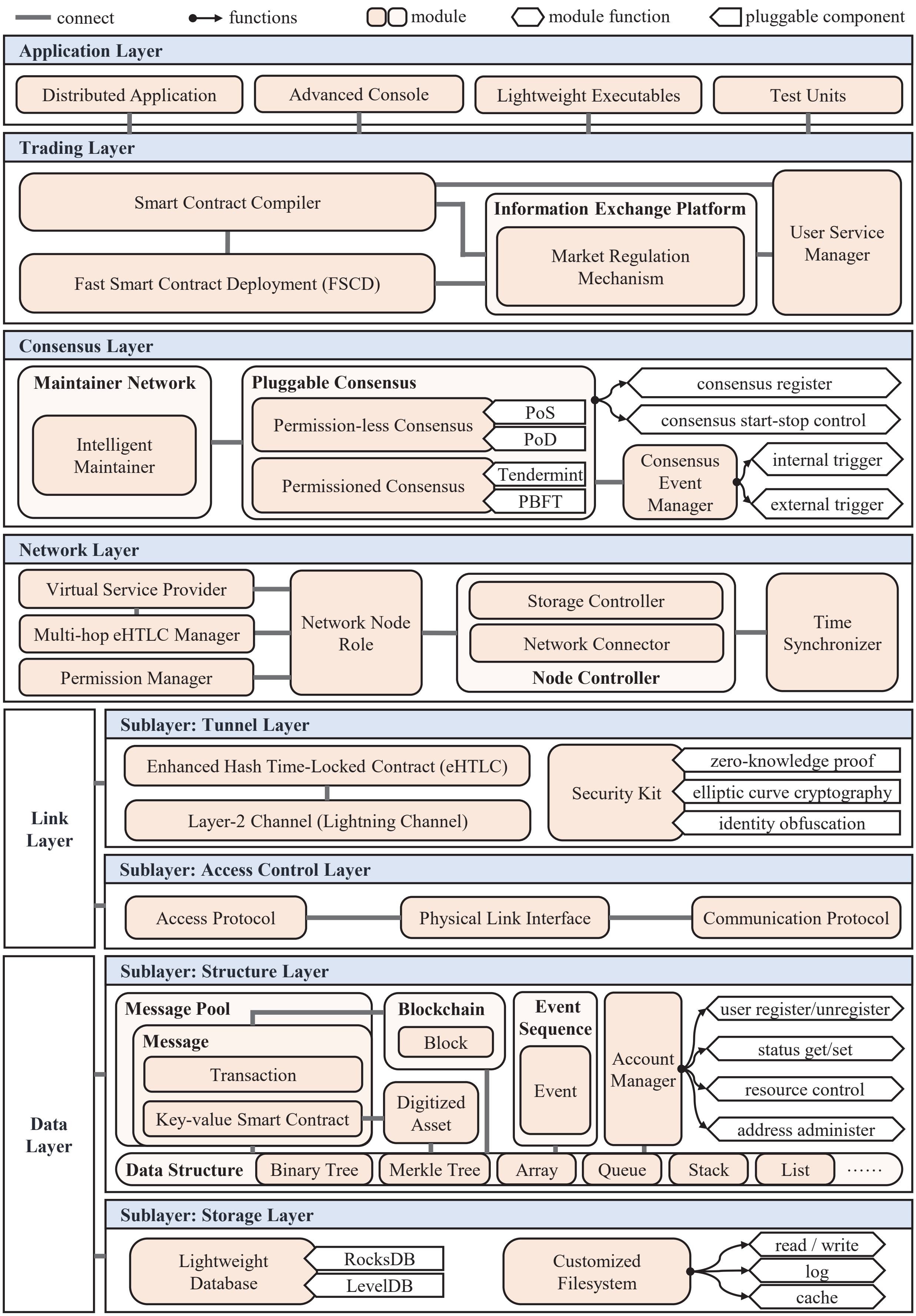


Fig. 2. Technical layers of the designed architecture.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3135414, IEEE Internet of Things Journal

- We categorize the B-RAN participants into four roles and use a blockchain-based method to supervise their behavior, and we involve a group of *intelligent maintainers* in multiple complex B-RAN tasks.
- We implement a portable and energy-efficient prototype to evaluate the overall performance of the designed architecture against an Ethereum-based implementation as a benchmark.

The following sections are organized as follows. Section II overviews the six-layer B-RAN architecture, and Sections III-VIII describe the key modules in each layer. We implement a prototype based on the proposed architecture and present experimental results in Section IX. Finally, Section X summarizes the article.

II. Architecture Overview

As illustrated in Fig. 2, we design a practical and efficient architecture for resource sharing and trading in B-RAN from a systematic perspective. In this section, we will overview the functionalities and components of these six layers.

A. Data Layer

The data layer includes two sublayers. The storage sublayer provides basic operations, such as storing, reading, and writing in B-RAN. These data operations are physically related to the memory chips inside the user equipment (UE), and thus the storage sublayer is an interface connecting the hardware devices to the other layers. We exploit a *lightweight database* in this sublayer by adopting the key-value storage database (KVSDB) such as LevelDB[2] and RocksDB[3]. Differing from commonly used relational databases, KVSDBs consume fewer memory and computation resources and thus are suitable for mobile scenarios. Furthermore, a *customized filesystem* is included in B-RAN as an encapsulation of the interfaces provided by KVSDBs for supporting ordinary database operations, fault recovery, broken/irrelevant data identification.

The other sublayer, the structure sublayer, defines the basic data structures in B-RAN. It interprets the data from the storage sublayer and organizes them into several predefined structures for the upper layers. We set up six modules in the structure sublayer. 1) The *data structure* module defines a plural of fundamental structures such as array, queue, stack, and so forth. 2) We define an efficient *blockchain* structure formed by a chain of properly designed *blocks* (see Section III-B). 3) The wireless resources in B-RAN are digitized and virtualized to be *digitized assets* (DAs) by the resource module to facilitate quick trading via smart contracts (see Section III-A). 4) The *account manager* preserves identity information (e.g., account registration, address, and status) and relevant properties (e.g., user-owned resources and trading tokens) in the blockchain. 5) *Messages* (e.g., transactions and key-value smart contracts) in B-RAN, before recorded by the blockchain, are temporarily stored in the *message pool* (see Section III-C). 6) The *event* and *event sequence* define standard formats for blockchain consensus and delivery between modules and entities. (Please see Section III for more details.)

B. Link Layer

The link layer is composed of the access control sublayer and tunnel sublayer. The access control sublayer handles communications between the UEs of B-RAN participants. We set up the *physical link interface* for communications within a subnetwork in B-RAN. The physical link interface module supports data transmissions between B-RAN participants according to pre-defined *communication protocols* that may vary among subnetworks, and decides the *access protocol* based on the features of the scenario or the distributed content. For example, the hash access proposed in [24] can be adopted for short-packet grant-free random access in the IoT scenario.

The tunnel sublayer is a medium between the network layer and access control sublayer, providing a safe and efficient tunnel for transactions and resource authorizations in B-RAN. B-RAN participants shall first request resources through the tunnel sublayer before requesting access via the access control sublayer. The tunnel sublayer is responsible for specifying the basic transaction procedures and securing the resource authorizations between two entities. In this sublayer, we design a *security kit* including many effective security solutions, e.g., the zero-knowledge proof, elliptic curve cryptography, and identity obfuscation, to safeguard the transmission data. We also set up the *layer-2 channel* for scalable transactions in B-RAN (Section IV-A). Based on the layer-2 channel, we propose the eHTLC mechanism to achieve secure and efficient B-RAN resource authorization (Section IV-B).

C. Network Layer

The aim of the network layer is to maintain data consistency among network nodes, regulate the nodes' behaviors, and define how they connect to and communicate with each other. It provides the networking support for the consensus and trading layers, and delivers messages from the upper layers to the link layer. In the network layer, we categorize the network nodes in B-RAN into four *roles* and introduce the *role contract* to declare the rights and obligations of different roles, which are supervised by the *permission manager* (Section V-A). We further introduce the concept of SP virtualization to benefit B-RAN from the pooling principle (Section V-B). We develop an *eHTLC-based multi-hop manager* for resource authorization and aggregation through multiple nodes in B-RAN.

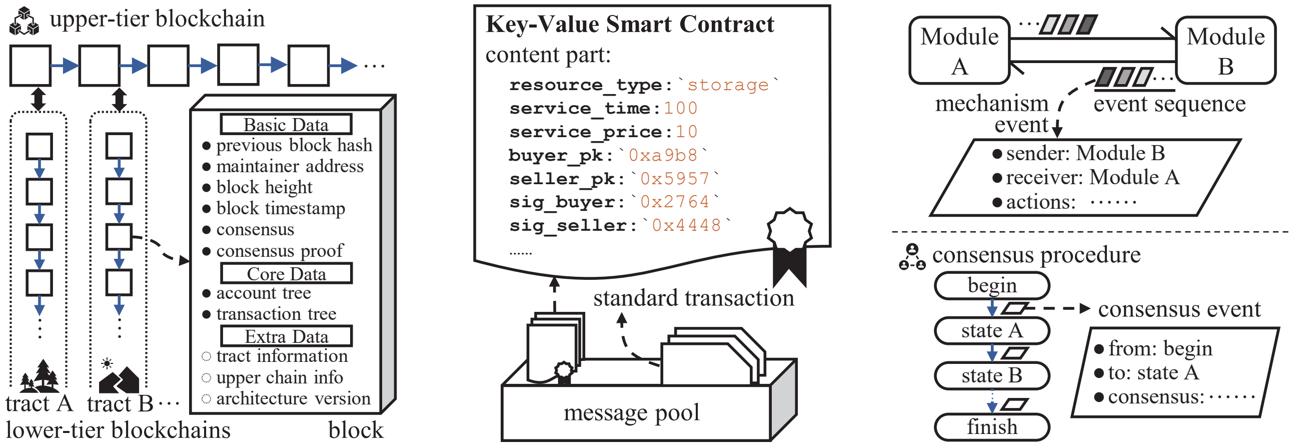
Furthermore, we design a *node controller*, including two submodules named the *storage controller* and the *network connector*, to define how network nodes store data and connect to each other. The storage controller stores messages and account data by Merkle-Patricia [27] trees within blockchains, and compresses the aged blocks (the blocks that have received a large number of confirmations) for saving the storage space on resource-constrained mobile devices. In the network connector, we use a transmission protocol based on RLPx[4] to broadcast

[2]https://github.com/google/leveldb/blob/master/doc/index.md, accessed May, 2021.

[3]https://rocksdb.org/, accessed May, 2021.

[4]https://github.com/ethereum/devp2p/blob/master/rlpx.md, accessed May, 2021.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3135414, IEEE Internet of Things Journal



(a) block and blockchain (b) message and message pool (c) event and event sequence

Fig. 3. Composition of blockchain components.

messages. The protocol prioritizes the node discovery and data transmission with network nodes at the same tier with appropriate roles in order to expedite the transmission and enhance network connectivity. Additionally, we specify a *time synchronizer* to calibrate the global time for the whole architecture and reduce exceptions caused by time misalignment. It can adopt a time calibration method based on trusted satellite broadcast [28].

D. Consensus Layer.

The consensus layer, along with the above trading layer, supports the basic operation of blockchains and the core functionalities of B-RAN services. In the consensus layer, intelligent maintainers of B-RAN blockchains not only work with the consensus compiler to enforce consensus processes and validate the data and requests, but also optimize resources assignment and allocation from proper SPs to requested clients for network efficiency (see Section VI-B). In this layer, we design the pluggable consensus so that B-RAN blockchains can flexibly switch to suitable consensus mechanisms. The pluggable consensus module is also responsible for registering, starting, and stopping the pluggable consensus operating on UEs. We introduce a *consensus event manager* to help the pluggable consensus module handle communications during consensus processes (see Section VI-A).

E. Trading Layer

Collaborating with the consensus layer, the trading layer is designed to safeguard resource exchanging, accelerate B-RAN services, and provide B-RAN participants with reliable and open access resource information. In this layer, we design four smart-contract-based modules namely the *smart contract compiler*, the *information exchange platform* (IEP), the *user service manager*, and the FSCD mechanism. The smart contract compiler (Section VII-A) is a key module that compiles the B-RAN key-value smart contracts into machine-readable codes. For quick contract deployment, we propose the FSCD in the trading layer (Section VII-B). Meanwhile, to facilitate resource trading, we establish the IEP as an open, public, and regulated market (Section VII-C). Furthermore, we introduce the user service manager for handling client-oriented businesses (e.g., service request, schedule, validation, and complaint) and connecting the application layer with the IEP, to protect the rights and interests of B-RAN participants.

F. Application Layer

The application layer connects the B-RAN users and the underlying layers. In this layer, we design four user-oriented tools, including *distributed application*, *advanced console*, *lightweight executable*, and *test units*, to better operate the user inputs (e.g., service requests, preference settings, etc.) and visualize critical information and real-time status of B-RAN for both ordinary users and developers.

III. Data Layer: Fundamental Units

A. DA

First of all, we introduce the concept of the DA as the basic element in B-RAN to represent the resources, e.g., spectra, computation, storage, energy, infrastructure, and data. Participants in B-RAN can quickly generate and easily virtualize network resources via classifying, tagging, and serializing DAs. Participants could generate an access code, a fixed-length serial code linked to a specific DA, to authorize resources to the other entities in B-RAN. During the trading, the DA access codes are recorded and secured by the blockchain.

B. Block and Blockchain

The block and blockchain are the backbone of the proposed architecture. As shown in Fig. 3, we categorize the block data into three fields, i.e., basic data, core data, and extra data fields. The basic data field defines the fundamental information of a block, such as the block height, block timestamp, and hash pointer to the previous block. To accommodate multiple pluggable consensus mechanisms (Section VI-A), we also define the consensus name and consensus proof in this field.[5] The core data field records user account and transaction

[5]If a proof of work (PoW) consensus is adopted, the consensus name is PoW and the consensus proof is the nonce.

data in the form of Merkle-Patricia trees [27]. The extra data field includes, e.g., current B-RAN version, geolocation information, etc., and can be neglected by some resource-constrained UEs.

In B-RAN, we adopt a novel two-tier blockchain structure [20]. As shown in Fig. 3(a), the lower-tier blockchains are set up according to geographic locations and mainly process service requests and tackle regional businesses, while the blockchain at the upper-tier records the digest data of lower-tier blockchains and dispatches inter-regional businesses.

C. Message and Message Pool

As shown in Fig. 3(b), messages in B-RAN include transactions and smart contracts. Smart contracts are important tools for automating resource sharing, allocating, and trading in B-RAN, and they can record service details, enforce services, and provide traceable receipts. In order to minimize the cost of storing, verifying, and executing contracts, we design a new smart contract structure for B-RAN named the key-value smart contract (which will be further discussed in Section VII-A). As shown in Fig. 3(b), the key-value smart contract is written in a simple syntax that only requires several critical information fields, e.g., resource type, resource price, traders' information from the service requesters. The transactions in B-RAN are based on standards in Ethereum [29] and Hyperledger Fabric [30], and are also vital to the procedures in Sections IV-B and VII-B. We use the message pool to store and manage messages that have not been recorded by the blockchain.

D. Event and Event Sequence

We introduce the concept of the event as a trigger for state transitions of every procedure in B-RAN. As demonstrated in Fig. 3(c), a typical event contains the identity of a sender, a receiver, and an abstract of the procedure. We divide events into two classes: the consensus event and the mechanism event. The consensus event, organized and handled by the consensus event manager (see Section VI-A), is designed to propel and synchronize the consensus process across the B-RAN network. The mechanism event is used to handle the other tasks such as inter-layer or intra-layer function calls, transaction initiation, smart contract creation, and so forth. Meanwhile, we define the events occurring in UEs as internally triggered events and those requiring inter-entity transmissions as externally triggered events, which will be sent to other participants after created. All received events are stored and processed in an event sequence in order.

IV. Link Layer: Entity Interactions

A. Layer-2 Channel

We introduce the layer-2 channel to improve the scalability of resource sharing and trading in B-RAN. The scalability issue has hindered the development of blockchain towards large-scale applications. Financial systems like VISA can reach tens of thousands of transactions per second (TPS)[6], whereas Bitcoin and Ethereum are usually below 7 TPS and 15 TPS, respectively [31]. To solve this problem, we use the layer-2 channel, also known as an off-chain trading technique, to accelerate transactions and service requests of resource sharing and trading in B-RAN for higher TPS.

We adopt the lightning channel to retain the decentralization benefits and compatibility of blockchain protocols [32]. To facilitate fast trading, the lightning channel requires both participants to initiate a standard multi-signature blockchain transaction (also known as a blockchain-based vault), where both sides deposit a certain amount of tokens. Afterward, both participants may conduct off-chain trades by reaching a tamper-resistant revocable sequence maturity contract (RSMC) between them. Note that the "contract" here does not refer to a blockchain smart contract but an off-chain mutual agreement enabled by scripts. The RSMC is a dual commitment involving two paired commitment transactions that include the latest balance of two parties. The RSMC held by both sides must appear in pairs at each time of their off-chain trade, and the withdrawal conditions of tokens in each RSMC are more beneficial to the other participant. The malicious behaviors of one side would be punished by the withdrawal conditions in the other side's holding RSMC. Since the RSMCs can limit dishonest behaviors without relying on on-chain operations, participants do not need to upload RSMCs to the blockchain immediately.

By deploying lightning channels into B-RAN, we enable B-RAN to process massive transactions and service requests of resource sharing and trading with enhanced efficiency. The layer-2 channel in the link layer can control lightning channels' status between nodes and even provide available multi-hop routing formed by several lightning channels.

B. eHTLC

In B-RAN, resource authorization often involves two security concerns. 1) During the trading, both participants want their trading resources protected from fraud and theft. 2) Trades are expected to be swiftly settled and fully traceable. To address these concerns, we propose the eHTLC mechanism as an upgraded version of HTLC [32]. When a client requests a DA from an SP or a VSP, the eHTLC can be used to establish trust and enforce a secure exchange between them.

Fig. 4 shows the structure and workflow of eHTLC in B-RAN. In the original HTLC [32], when a payer promises some tokens to a payee, the payee cannot retrieve the fee in the HTLC unless it correctly solves a puzzle asked by the payer within a given time. Otherwise, the trade is canceled. Now, the eHTLC replaces the meaningless puzzle with a DA-related riddle to secure resource trading in B-RAN. As shown in Fig. 4, the eHTLC has a hash lock (HL) represented by an irreversible value of a hash riddle $\mathrm{h}(\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}}))$. The tokens in eHTLC can only be retrieved by using the correct key $\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}})$ to the HL within a time lock (TL). Otherwise, the tokens would be refunded to the buyer after the TL expires. The TL, represented by an expired time t_d, could be a period of time (e.g., one hour from now) or a number of succeeding blocks (e.g., six blocks from now).

[6]https://usa.visa.com/run-your-business/small-business-tools/retail.html, accessed May, 2021.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3135414, IEEE Internet of Things Journal

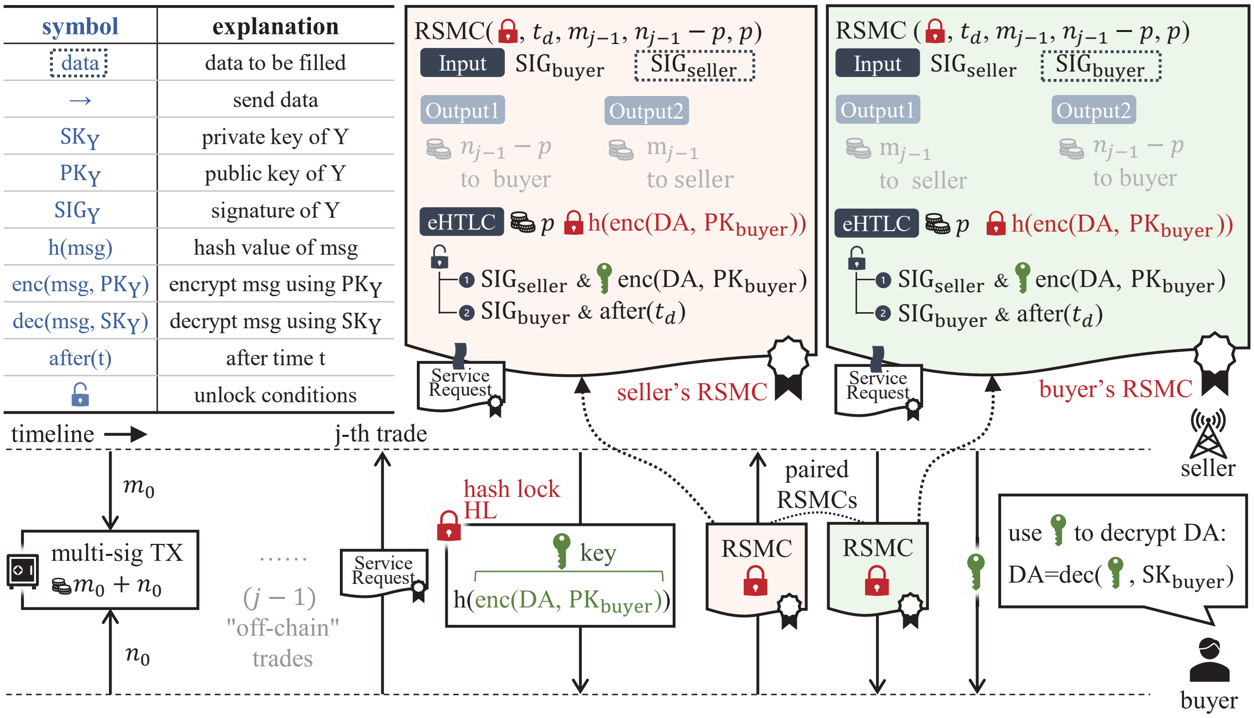


Fig. 4. Structure and workflow of the eHTLC in B-RAN.

Both HL and TL are related to the trading DA. Therefore, we can express an RSMC with eHTLC output as

$$\mathrm{RSMC}(\mathrm{HL}, \mathrm{TL}, \mathrm{B}_{\mathrm{seller}}, \mathrm{B}_{\mathrm{buyer}}, \mathrm{B}_{\mathrm{eHTLC}}),$$

where $\mathrm{B}_{\mathrm{seller}}$ and $\mathrm{B}_{\mathrm{buyer}}$ are the balances of the seller and buyer, respectively, and $\mathrm{B}_{\mathrm{eHTLC}}$ is the rest of tokens in the eHTLC. In the following, we show a detailed procedure of the resource trade based on the eHTLC.

- **Step 1.** Before an off-chain trading, the seller and buyer have to establish a lightning channel between them by funding m_0 and n_0 tokens, respectively, to their multi-signature address. Denote m_j and n_j ($m_j + n_j = m_0 + n_0$), as the balance of seller and buyer after the j-th off-chain trade.
- **Step 2.** In the j-th trade, the buyer requests an available DA with price p via eHTLC. Then, the seller uses the public key of buyer to generate a hash lock HL=$\mathrm{h}(\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}}))$, and sends it back to the buyer. The HL can only be unlocked by the key $\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}})$.
- **Step 3.** Both the buyer and seller create a new RSMC that transfers p tokens from the buyer's balance to the eHTLC output. The RSMC gets updated and becomes

$$\mathrm{RSMC}(\mathrm{h}(\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}})), t_d, m_{j-1}, n_{j-1} - p, p),$$

where the HL is $\mathrm{h}(\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}}))$ and the TL is t_d. Then, they attach the service request in step 2 and exchange the paired RSMCs. The exchange enforces both sides to behave honestly. As shown in the upper part of Fig. 4, the tokens in the eHTLC output can be withdrawn by the seller via the first unlock condition (announcing the key $\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}})$ to HL) or by the buyer via the second condition (waiting until the TL expires).
- **Step 4.** The seller provides the key of the HL, $\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}})$, to the buyer before the TL runs out. Then, the seller can get the resource fee p by signing either RSMC and publish the correct key $\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}})$ to claim the ownership of the tokens in the eHTLC output.
- **Step 5.** The buyer uses its private key $\mathrm{SK}_{\mathrm{buyer}}$ to decrypt the DA by:

$$\mathrm{DA} = \mathrm{dec}(\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}}), \mathrm{SK}_{\mathrm{buyer}})). \tag{1}$$

- **Step 6.** The buyer can upload its RSMC to the blockchain to finish the trade, update the two participants' online balance, and shut down the channel. Also, they can keep the channel active and confirm their trading results offline.

Now we highlight two special situations: 1) If the seller fails to give the key $\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{buyer}})$ before the expired time t_d, the buyer can get a refund using the second condition in the RSMC. 2) In case that the buyer has received the key but declares it has not or just fails to respond, the seller should announce its RSMC as soon as possible to get the fee via the first condition; otherwise, the buyer may cheat as the TL expires.

The eHTLC shows great potential to achieve safe and fast DA authorization between sellers and buyers. It is also critical to the functions such as data sharing and collaborative task computing. For example, in eHTLC-based data sharing, participants can share their data efficiently and without fear of data tampering or eavesdropping. In all, the advantages of eHTLC can be summarized as follows: 1) The eHTLC-based authorization is executed off the chain, leading to a more efficient resource trading (in milliseconds) than on-chain approaches. 2) The eHTLC mechanism can enforce both participants to behave honestly and follow the pre-defined rules. 3) The key to the HL is encrypted using the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3135414, IEEE Internet of Things Journal

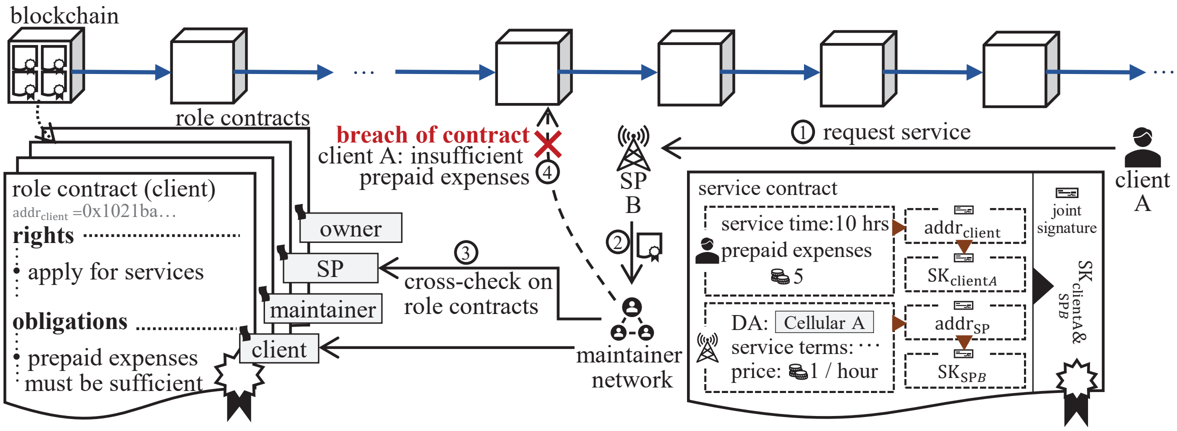


Fig. 5. Example of a failed service request detected by permission manager.

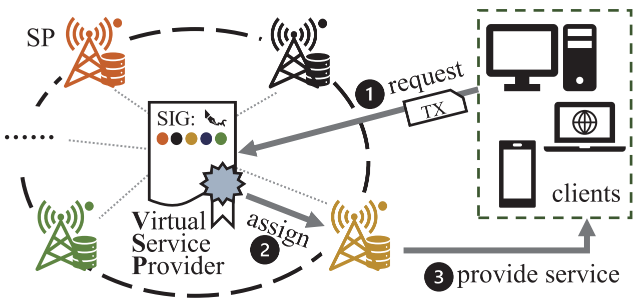


Fig. 6. Pooling of SPs in B-RAN and its application paradigm.

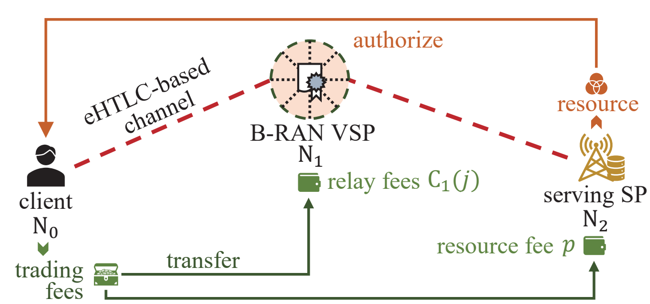


Fig. 7. Illustration of a client-VSP-SP service based on eHTLC-based multi-hop mechanism.

buyer's secret key, and thus the DA cannot be obtained by other participants. 4) The unlock conditions of the eHTLC can prevent the participants from cheating and protect both participants' legitimate interests.

V. Network Layer: Nodes and Connections

A. Network Nodes and Attributes

In B-RAN, multiple nodes collaborate for diverse network functionalities via proper network orchestration. To better describe the behaviors of different B-RAN participants, we categorize the network nodes in B-RAN into four basic roles with their attributes according to their rights and obligations as follows.

- **Owner**. They have the ownership of resources in B-RAN. Owners can lease or sell the resources in the form of DAs.
- **SP**. They purchase or lease DAs from owners, and further re-allocate, share, and distribute these resources to clients via signing contracts.
- **Client**. Ordinary users of wireless resources and services.
- **Maintainer**. Data validators and blockchain consensus executors in B-RAN, similar to the "miners" in traditional blockchains.

Note that the network nodes in B-RAN can play more than one role and flexibly change the roles if necessary. For example, a traditional network operator plays both roles of spectra owners and SPs.

To regulate the rights and obligations of participants, we define a special type of smart contracts, namely role contracts. As shown in Fig. 5, the role contracts are recorded in the genesis block of blockchain with public addresses. Moreover, considering that the role contracts might be updated periodically, we introduce an amendment smart contract. The amendment smart contracts are linked to the role contracts as complements and only take effect after getting a certain amount of approvals from B-RAN participants. All participants should refer to the latest amendment smart contracts as the preference to avoid possible ambiguities.

Furthermore, the behaviors of different roles are supervised and regulated by the permission manager. The permission manager is designed to identify and stop digital actions that violate the rights and obligations in role contracts. It can also reduce potential risks such as maliciously tampering with the service contract and man-in-the-middle attacks. Participants should declare their roles and the corresponding address of role contracts when creating smart contracts and transactions in B-RAN.

We show an example of detecting a flawed service request in Fig. 5. After both the client and SP fill in the service information, they have to add the address of their corresponding role contracts and then sign their respective parts using private keys. Next, they send the contract to the maintainer network with their joint signature. Under the control of the permission manager, maintainers identify that the client violates its obligation that all clients have to prepay sufficient service fees, and therefore discard this request rather than assembling it into a new block.

Algorithm 1 eHTLC-based multi-hop resource trading process in B-RAN

Input: a selected multi-hop route consisting of node accounts $\mathbf{N} = \{\mathrm{N}_0, \mathrm{N}_1, ..., \mathrm{N}_i\}$, nodes' balance after the $(j-1)$-th trade $\mathbf{B}(j-1) = \{\mathrm{B}_0(j-1), \mathrm{B}_1(j-1), ..., \mathrm{B}_i(j-1)\}$, relay fees that nodes decide to charge at the j-th trade $\mathbf{C}(j) = \{\mathrm{C}_0(j), \mathrm{C}_1(j), ..., \mathrm{C}_i(j)\}$, process time that nodes need at the j-th trade $\mathbf{D}(j) = \{\mathrm{D}_0(j), \mathrm{D}_1(j), ..., \mathrm{D}_i(j)\}$, and request $\mathrm{REQ}(t,p)$, where t is the ordered service time, p is the service price.

Output: nodes' balance after the j-th trade $\mathbf{B}(j) = \{\mathrm{B}_0(j), \mathrm{B}_1(j), ..., \mathrm{B}_i(j)\}$, DA.

N_0 requests a service from N_i by sending request $\mathrm{REQ}(t,p)$;
N_i allocates a DA satisfying $\mathrm{REQ}(t,p)$;
N_i generates the HL of RSMC as h(enc(DA, $\mathrm{PK}_{\mathrm{N}_0}$)) and sends to N_0;
for $k = 0$ **to** $i-1$ **do**
  N_k sends an RSMC to N_{k+1} and N_{k+1} sends back a mirror RSMC with same parameters:
$$\mathrm{RSMC}(\mathrm{h}(\mathrm{enc}(\mathrm{DA}, \mathrm{PK}_{\mathrm{N}_0})),\ \sum_{l=k}^{i-1}\mathrm{D}_l(j),\ \mathrm{B}_{k+1}(j-1),\ \mathrm{B}_k(j-1) - \sum_{l=k+1}^{i-1}\mathrm{C}_l(j),\ p + \sum_{l=k+1}^{i-1}\mathrm{C}_l(j));$$
end for
for $k = i$ **to** 1 **do**
  N_k gives the key of the HL, enc(DA, $\mathrm{PK}_{\mathrm{N}_0}$), to N_{k-1};
  if $k = i$: $\mathrm{B}_k(j) \leftarrow \mathrm{B}_k(j-1) + p$;
  else : $\mathrm{B}_k(j) \leftarrow \mathrm{B}_k(j-1) + \mathrm{C}_k(j)$;
end for
$\mathrm{B}_0(j) \leftarrow \mathrm{B}_0(j-1) - p - \sum_{l=1}^{i-1}\mathrm{C}_l(j)$;
N_0 retrieves the DA by performing dec(enc(DA, $\mathrm{PK}_{\mathrm{N}_0}$), $\mathrm{SK}_{\mathrm{N}_0}$).

B. SP Virtualization

Now we develop the concept of the VSP as a virtual entity that connects a group of SPs in B-RAN and pools and shares the wireless resources among these cooperative SPs. The VSP breaks the physical and logical barriers between SPs and makes available resources accessible to all B-RAN participants. When a client requests services from the VSP, it could use the resource from a suitable SP of the VSP under unified management. The VSP decouples physical infrastructures and digital representations to provide a virtual viewpoint to the network participants and facilitates the SPs in B-RAN to better cooperate with enhanced efficiency and security. From a global point of view of B-RAN, the VSP can balance traffic loads among all SPs. The VSP is also helpful to handle burst traffics by rearranging and offloading traffic loads.

Technically, the VSP is established by a union of SPs through a special smart contract of SP virtualization. Each SP in B-RAN reaches a uniform service agreement and adds its signature to the VSP smart contract before it becomes a member of VSP. We demonstrate the working paradigm of a VSP-enabled service request in Fig. 6. When a client requests a service, the client would first contact the VSP and indicate its demand for the service, e.g., its preferred resource, service time, price, and information of peripheral SPs, via a transaction. The transaction also contains the client's prepaid service fees and is sent to the VSP smart contract. Then, the VSP smart contract would assign an eligible SP to provide service for the client. After the service, the VSP would transfer the service fees to the service SP's account.

By leveraging the pooling principle that a larger shared network formed by multiple subnetworks is more efficient than the collection of smaller non-shared ones [33], the VSP can establish strong cooperation among SPs, simplify their resource management, enhance service performance, reduce service costs, and better cope with burst traffics. Meanwhile, clients can receive better quality of service from a union of SPs without subscribing to a specific SP. The VSP endows B-RAN with the capability of efficient resource coordination, fair load balancing, and intelligent service scheduling. (An application is introduced in Section V-C.)

C. Multi-Hop eHTLC Manager

According to the previous section, when requesting services from the VSP, the client would deposit the service fee in the VSP account. To let the serving SP get paid and the client get its required resource, we provide a multi-hop eHTLC manager for multi-hop DA authorizations. The manager can implement the aforementioned client-VSP-SP services and is also applicable to trading scenarios such as multi-hop resource leasing. During multi-hop trading, one or more intermediate B-RAN network nodes would participate based on several eHTLCs. Assume that node N_0 requests for a specific DA from nodes connected to it via existing eHTLC-based channels, and N_i $(i > 1)$ is the only node who has the DA that N_0 requires but does not have a direct eHTLC-based channel connecting to N_0. In this case, for node N_0 to get the DA, multiple intermediate nodes have to be involved in the following two tasks. 1) Finding and creating a route based on eHTLC-based channels from N_0 to N_i. 2) Relaying the DA from N_i to N_0, and transferring the service fee from N_0 to N_i based on eHTLCs. We summarize the multi-hop eHTLC mechanism in Algorithm 1.

Further, we show an example where a client requests a service from the VSP and illustrate the involved steps in Fig. 7. Assume that the client and all available SPs in B-RAN are connected to the VSP via the eHTLC-based channels introduced

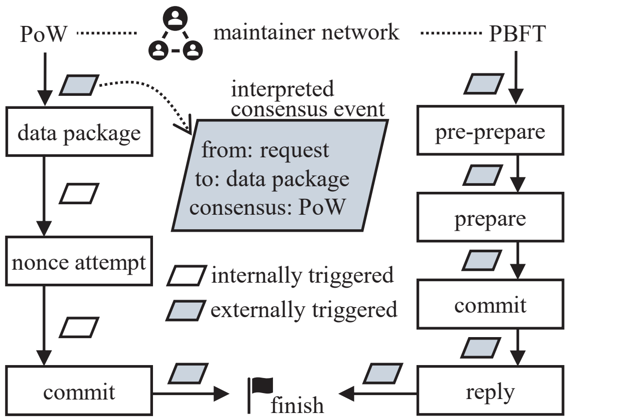


Fig. 8. Workflows of the pluggable PoW and PBFT consensus enabled by consensus event manager.

in Section IV-B. Note that SPs can establish eHTLC-based channels with the VSP when they first register as its members. The request would go through the following steps. 1) The client first makes a resource request to the B-RAN VSP, which then assigns an eligible serving SP to the client. 2) The client prepares the resource fee for the serving SP and the processing reward for the VSP. 3) The SP authorizes its resource to the client via two existing eHTLC-based channels, and the client transfers fees to the VSP and serving SP. Moreover, as introduced in Section V-B, the multi-hop eHTLC manager can work with the VSP to establish secure routes for offloading through several SPs according to Algorithm 1.

The multi-hop mechanism retains all merits of the eHTLC mechanism and establishes firm cooperative trust among the client, VSP, and SPs in B-RAN, and is flexible to provide more intelligent and diversified services.

VI. Consensus Layer: Blockchain Maintenance

A. Pluggable Consensus

We design the pluggable consensus to configure consensus protocols in B-RAN flexibly. The pluggable consensus module covers permissioned and permission-less protocols. The permissioned consensus, e.g., Tendermint [34] and PBFT [35], has low energy consumption and high efficiency and can be adopted to handle regional services and small-scale businesses. The permission-less consensus, e.g., PoW [5], PoS [36], PoD [24], usually highlights its openness and enhanced security. It is suitable to be applied in large-scale and traffic-intensive environments. In B-RAN, the lower-tier and upper-tier blockchains can choose from these two types of consensus based on security needs and traffic load in their deployed areas. We use the consensus and consensus proof fields in blocks to identify the adopted consensus. Besides, the pluggable consensus can also be changed according to pre-defined rules, e.g., changing to the PBFT consensus if the average block size of the last ten blocks exceeds a given value.

Further, we define a sequence of consensus events for each pluggable consensus to identify the state transitions between each consensus state. For example, the pluggable PoW consensus in B-RAN contains five states and requires

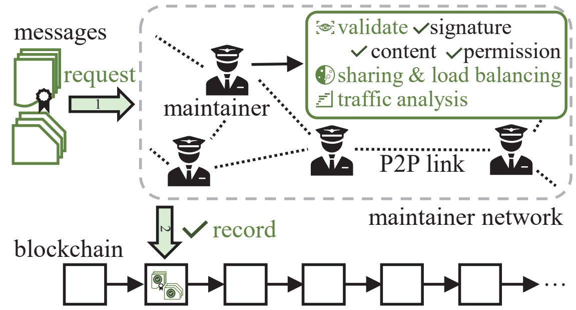


Fig. 9. Illustration of B-RAN intelligent maintainers.

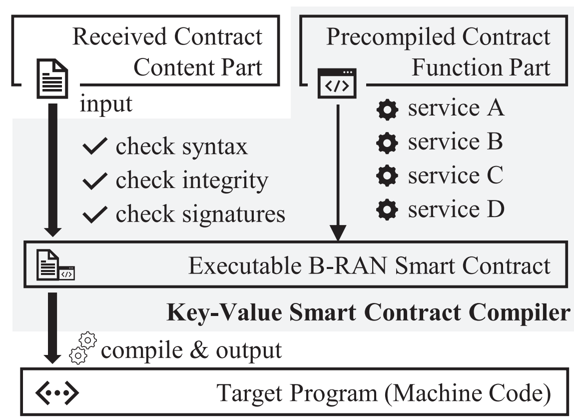


Fig. 10. Working paradigm of key-value smart contract compiler.

four consensus events. The consensus events can assist B-RAN maintainers in identifying and tracking the progress of an ongoing consensus. We use the consensus event manager to interpret those events and trigger state transitions on UEs. In Fig. 8, we demonstrate the workflows of the pluggable PoW and PBFT consensus. Each transition, marked as black arrows, is raised by a consensus event, and the permissioned consensus mechanisms usually have more externally triggered events than permission-less ones.

B. Intelligent Maintainers

Regulated by the permission manager, the B-RAN maintainers can do more than running consensus and managing blockchains. They are a group of intelligent miners who can also validate node identities and service requests and identify low-load or busy service areas.

This feature makes them not only driven by mining incentives but also engage in network activities as B-RAN participants. As illustrated in Fig. 9, the intelligent tasks that can be done by the intelligent maintainers include checking signatures and contents of received requests, validating requesters' permissions, sharing and balancing network load, perceiving and analyzing real-time traffic, and so forth. Intuitively, this may prevent a significant portion of false and invalid messages from being further spread to other network nodes, reduce the network transmission overhead, and further improve the overall efficiency of resource sharing and trading in B-RAN.

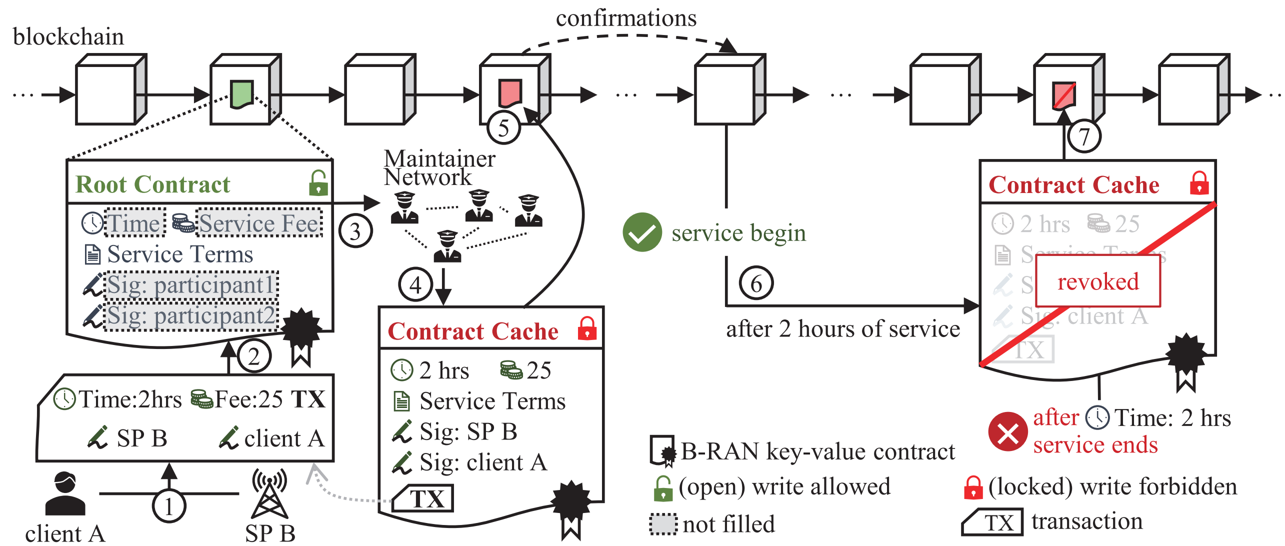


Fig. 11. Example of an SP's providing the service to a client using the FSCD.

VII. Trading Layer: Contract-Based Trading

A. Key-Value Smart Contract

In this section, we develop the key-value smart contract to reduce the cost of storing, verifying, and executing smart contracts in B-RAN. The key-value smart contract consists of the function part and content part (introduced in Section III-C). The function part of a key-value contract is precompiled and stored on the UEs of participants locally.[7] Only the content part is announced to specify the service details in a request.

To support the content verification and service calls in key-value smart contracts, we use a smart contract compiler to combine the content and function parts together and translate the restored contract to machine-readable target programs, i.e., machine codes. The working paradigm of the compiler is shown in Fig. 10. For the varying contents of each received contract, the key-value smart contract compiler will first check their syntax, integrity, and signatures before combining them with the function part. Then, the compiler will merge the precompiled function part with the input content, and generate a fully-restored executable B-RAN smart contract. Finally, based on different execution environments, the compiler will choose a compatible interpreter to convert the restored contract to machine codes. Since the key-value smart contracts only transmit the critical information of a request and the function part of contracts is precompiled and stored on UEs beforehand, the network transmission and contract compilation overhead can be remarkably reduced. Also, the content parts are transmitted in their original format without any compression, which can facilitate quick examination and validation by B-RAN participants.

B. FSCD

The FSCD manager is an important mechanism for quick contract deployment. In existing blockchain platforms, participants may worry that a service smart contract is flawed or invalid, so an efficient way to protect their interests is to pay expenses after the content of smart contract has been confirmed by the blockchain, which will lead to a lengthy confirmation process. The FSCD manager combines with the key-value smart contract and can accelerate contract deployment. To achieve this goal, we first set up two types of smart contracts, namely the root contract and contract cache. The root contract is an unfilled content part of a key-value smart contract. We elaborate multiple root contracts as service request templates of various B-RAN services, in which the service descriptions and some required fields are given. The contract cache is a filled root contract with complete request content.

Next, in Fig. 11, we illustrate the full working process of the FSCD manager via a scenario where an SP provides service to a client. The service can be establishing an eHTLC-based channel, trading a resource, updating a role contract, and so forth. The process includes the following steps.

1) Client A requests SP B for service. They jointly create a transaction in which they declare the request information, including service time, digital signatures, and an advance service payment from the client.
2) The transaction activates the root contract, which the blockchain has previously recorded.
3) The root contract generates a filled contract cache with the request information and transaction and is then sent to intelligent maintainers.
4) Maintainers thoroughly examine the signatures and content of the contract cache, confirm the client and SP's permissions, and check the validity of the transaction.
5) After checking, the maintainer network assembles the contract cache into a new block and commits it to the B-RAN blockchain.
6) The service begins immediately after the block is confirmed in the blockchain. Then, the SP provides the client with a service that meets the content of the contract cache.
7) When the service time is over, the service fee is auto-

[7]The function part defines all the digital operations of services and it can be recorded in a block in the upper-tier blockchain. New B-RAN participants may get the function part from this block after joining B-RAN.

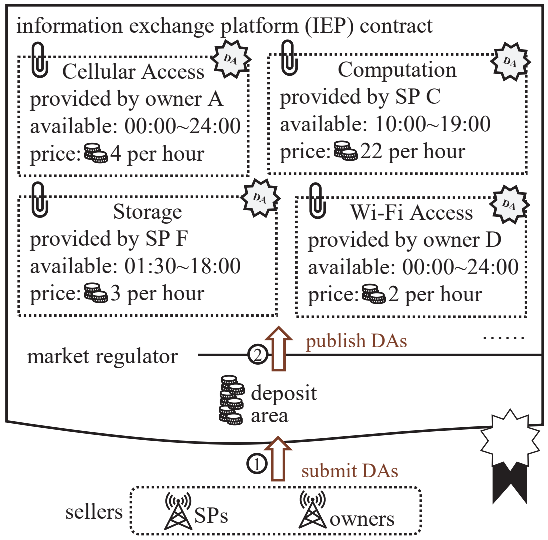


Fig. 12. Structure of the IEP and the registration procedure of new DAs.

matically transferred from the contract cache to the SP. Next, the contract cache creates a copy of itself with zero balance and a "revoked" tag, and commits the copy to the blockchain to declare the end of this service.

In B-RAN, we use the FSCD on various services to reduce service waiting time and upgrade user experience. In the above process, the FSCD manager requires fewer block confirmations, and therefore the service between two participants can be reached more rapidly. If such services are implemented by old ways with no service request template, the participant has to commit its request and then pay the service fee after confirming the validity and security of the request content. Since the root contracts can be created and recorded by the blockchain before service requests arrive, the service contents declared therein can hardly be tampered with by rogue nodes, and B-RAN participants should have ample time to review its content and ensure the legitimacy of services. Furthermore, suppose a malicious entity misbehaves during a service, e.g., overcharging service fees. In that case, the B-RAN intelligent maintainers would identify such behaviors by comparing the content of the root contract and contract cache, and discard the contract cache to block the service. The victim could get back its prepaid expenses without delay since the blockchain has not recorded the transaction and contract cache.

C. IEP

In B-RAN, we build the IEP as a public resource market [37], [38], shown in Fig. 12. In the IEP, owners and SPs can publish their resources and sell them to clients. The selling resource item in the IEP contains critical information such as the seller's identity, available time, and price. SPs may also purchase or lease DAs from multiple owners to provide convergent services.

The IEP uses multiple wireless resource pricing algorithms [17], [39], [40] to help SPs and owners decide the selling price for their DAs according to the market trend. It can prohibit sellers from unfair competitions such as selling overpriced resources, and protect the legitimate interests of all market participants. Besides, we devise a market regulation mechanism to declare market rules and impose penalties on network nodes who violate the rules, e.g., selling invalid DAs, serving resources inconsistent with descriptions, initiating hostility complaints, and spending tokens with suspicious origins. To this end, we require all market participants to pay a certain amount of tokens to a deposit area in the IEP contract before they engage in any market transaction. For sellers, the amount of deposit must be higher than the total price of their selling DAs, and for buyers, they must use their deposit to purchase resources. If a participant violates the market rules, the mechanism would launch a public arbitration via the user service manager to confirm the violation and deduct from the deposit as a penalty. Besides, once a participant's deposit is lower than a certain amount, it will be temporarily banned from market trading.

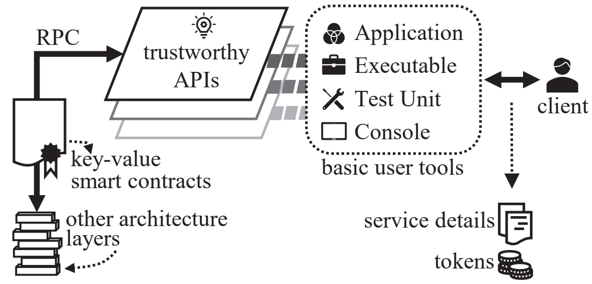


Fig. 13. Structure and connections in application layer.

VIII. Application Layer: Trustworthy Client Services

We design several user-oriented modules in the application layer, such as basic user tools, trustworthy application programming interfaces (API), and standard remote procedure calls (RPC) [41]. The trustworthy APIs implement a number of back-end operations for the front-end user tools, and convert between user commands and smart contracts. The RPCs interpret the smart-contract-based data flow between the application layer and other layers. For example, when initiating a resource request in B-RAN, participants may use the basic user tools to fill out service details. Then, the trustworthy APIs will translate these data into a key-value smart contract and forward it to the RPCs. The RPCs finally translate the contract into a sequence of bits and send it to the lower layers. To better describe the usage of each basic user tool, we categorize the tools into two groups.

- For B-RAN developers. 1) Test units: we devise a set of test cases for every module in this architecture so that the developers may get a quick start and spot the vulnerabilities in the codes. 2) Advanced console: the advanced console with command-line interfaces is designed for developers to interact directly with the trustworthy APIs.
- For ordinary users. 1) Distributed application: the distributed application is a program with graphical user interfaces (GUI) and can provide ordinary users with a full set of visualized B-RAN data. 2) Lightweight

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3135414, IEEE Internet of Things Journal

[Ethereum-based Implementation] B-RAN Smart Contract

Solidity

```
struct rele {
    uint req_time;
    address payable addr_user;
    address payable addr_server;
    string details;
    uint price;
    uint serv_time;
    uint deposit;
    bytes sig_user;
    bytes sig_server;
    bytes32 final_hash;
}

rele private _r;
uint private start_time;
string private appeal_reason;
uint private jury_ub = 5;
uint private jury_num = 0;
uint private jury_withdraw = 0;
uint private jury_on_user = 0;
address payable public requestor;
uint private status;

constructor(
    address payable _addr_user,
    address payable _addr_server,
    string memory _details,
    uint _price,
    uint _serv_time,
    uint _deposit,
    bytes memory _sig_user,
    bytes memory _sig_server
) public payable{...}
```

Service Parameters

```
function payDeposit()public payable {...}
function close() public {...}
function start_service() public {...}
function end_service() public {...}
function appeal_service(string memory _appeal_reason) public {...}
function public_jury(bool _is_in_trouble, uint _support_side) public {...}
```

Service Functions

[Layered Prototype] B-RAN Key-Value Smart Contract

Python

```
class kv_contract:
    types_resource = ['spectrum', 'storage', 'computing']
    types_request = ['fscd']
    def __init__(self, time_created=None, pubkey_seller_raw=b'', pubkey_buyer_raw=b'', signature_seller_raw=b'', signature_buyer_raw=b'',
                 length_service=1, type_resource='spectrum', type_request='fscd', price_service=1, comments='', status=True,
                 funding_amount=1, funding_account_pubkey_raw=b''):
        self.time_created = time.time() if time_created is None else time_created
        self.pubkey_seller_raw = pubkey_seller_raw
        self.pubkey_buyer_raw = pubkey_buyer_raw
        self.signature_seller_raw = signature_seller_raw
        self.signature_buyer_raw = signature_buyer_raw
        self.length_service = length_service
        self.type_resource = type_resource.lower()
        self.type_request = type_request.lower()
        self.price_service = price_service
        self.comments = comments
        self.status = status
        self.funding_amount = funding_amount
        self.funding_account_pubkey_raw = funding_account_pubkey_raw
        self.time_begin_service = None
```

Content Part

```
    def sign(self, privkey_raw):...
    def validate(self, pubkey_raw, signature):...
    def validate_self(self):...
    def start_service(self):...
    def is_finished(self):...
    def finish_service(self):...
```

(Pre-compiled) Function Part

Fig. 14. Outlines of the B-RAN resource service smart contracts in the layered prototype and Ethereum-based implementation.

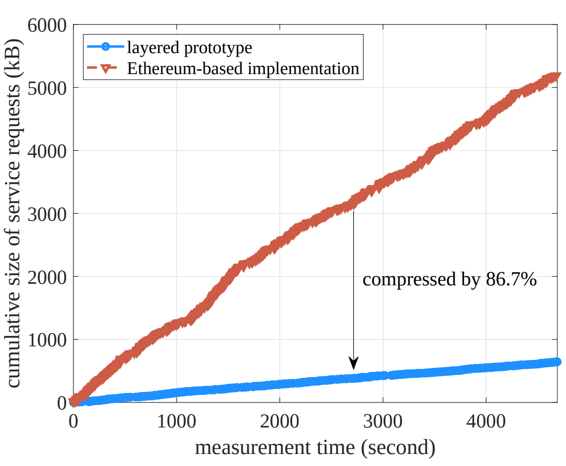


Fig. 15. Size of service requests accumulates over the measurement period.

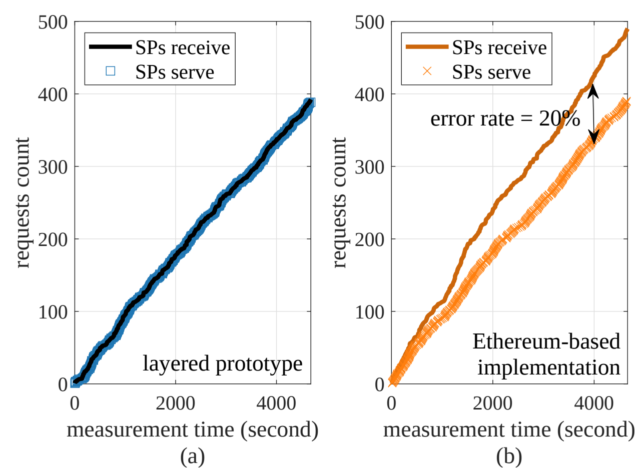


Fig. 16. Comparison between the amount of services received and actually processed by SPs. (a) Layered prototype. (b) Ethereum-based implementation.

executable: the lightweight executable is designed for resource-constrained mobile devices. It is a compressed single-file application that allows devices to use resource services and applications in B-RAN with minimal computing and storage costs.

IX. Prototype Experiments and Results

A. Experiment Environment

According to the proposed B-RAN architecture, we implement a layered prototype using Python. Meanwhile, we also realize an Ethereum-based implementation as a benchmark, which uses a traditional blockchain architecture [42].[8] Yet, due to the limitations of Ethereum development tools, the Ethereum-based implementation lacks some critical modules, such as the key-value smart contract, FSCD, eHTLC, and intelligent maintainers. We demonstrate the outlines of service smart contracts in the experiments in Fig. 14, where the layered prototype divides the contract into an editable content

[8]The codes can be found in https://github.com/leyuwei/BRANChain

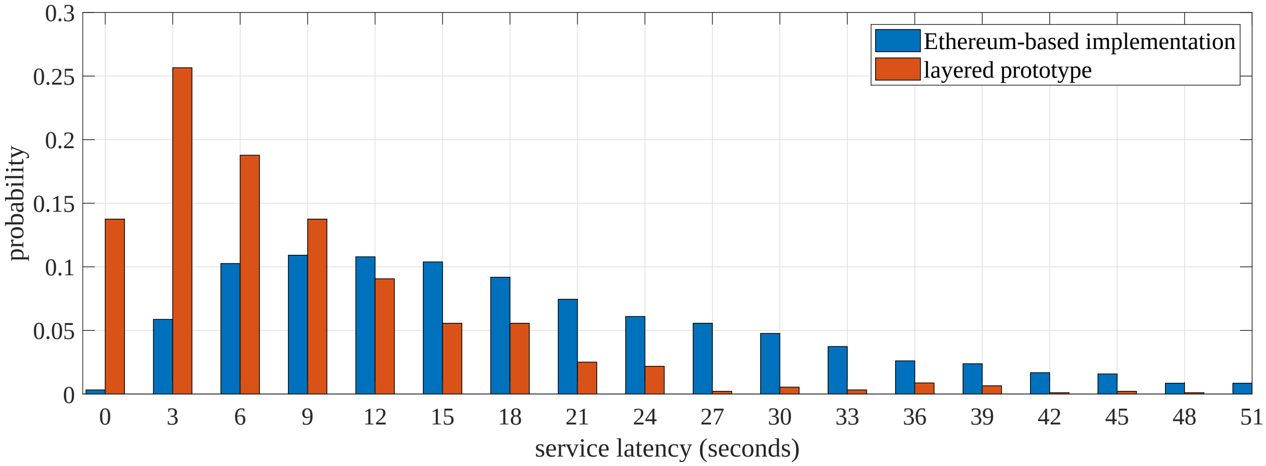


Fig. 17. Distribution of the service latency in the layered prototype and Ethereum-based implementation.

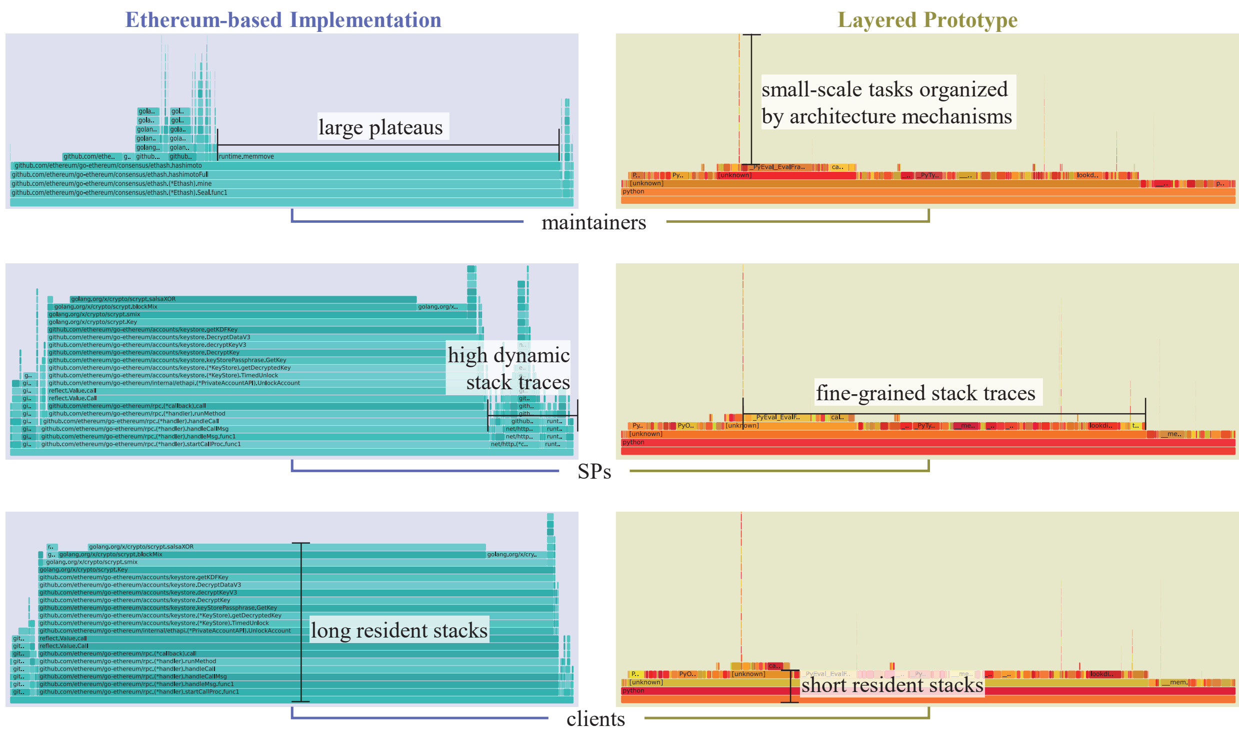


Fig. 18. Comparison of CPU flame graphs of the layered prototype and Ethereum-based implementation.

part and a fixed precompiled function part. However, all codes in the Ethereum-based implementation have to be involved in every transmission and compilation process. All the experiments in this work are based on a single workstation equipped with an Intel I7-8700K processor, 16 gigabytes RAM, and 2 terabytes storage.

In the experiments, we set two clients, two maintainers, and two SPs in a local network, and all nodes are fully connected to each other. We adopt the PoW consensus to fairly compare the performance of the layered prototype and Ethereum-based implementation.[9] We use RLPx transmission protocol and LevelDB for both two implementations and let all nodes separately work on local network ports. We set the average block time T^b to 2 seconds and average requested service time T^c to 10 seconds, and vary the network parameters, such as the average request interarrival time T^a, number of block confirmations N, number of SPs κ, maximum number of clients that each SP can provide services for simultaneously ι, and traffic load $\rho = \frac{T^c}{\kappa \iota T^a}$, to illustrate the performance.

[9]Note that, the permissioned consensus mechanisms (e.g., PBFT and Tendermint) are also available and pluggable for the layered prototype.

B. Architecture Design Evaluations

In this section, we evaluate the performance of several key architecture designs through the comparison between the layered prototype and Ethereum-based implementation. First, we show network traffic represented by the cumulative size of service requests in Fig. 15. As shown in Fig. 15, the cumulative size of service requests grows at significantly different rates in the layered prototype and Ethereum-based implementation. After 4000 seconds, the accumulated size in the Ethereum-based implementation becomes nearly eight times larger than

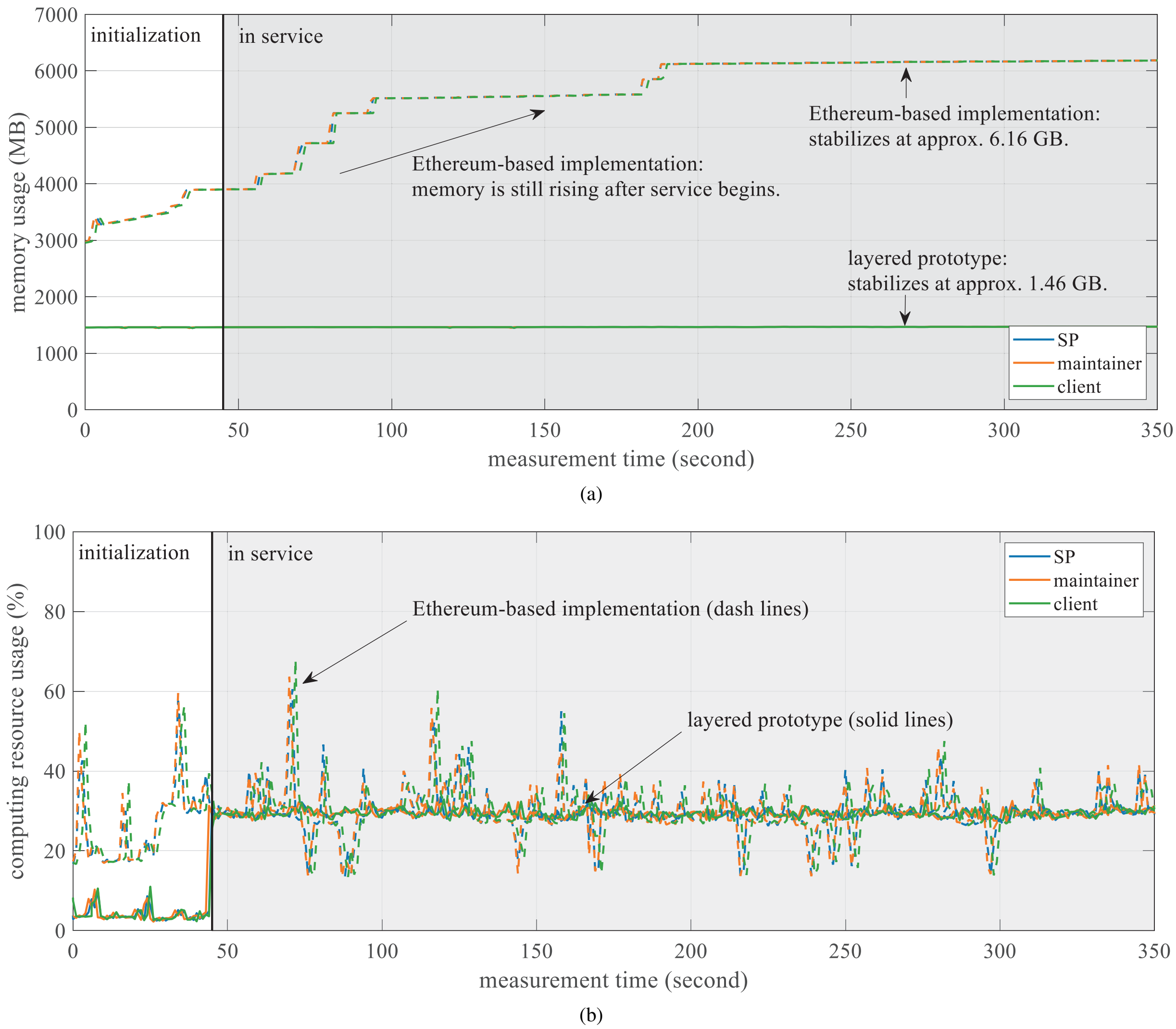


Fig. 19. Resource consumption of different node roles over the measurement period. (a) Memory usage. (b) Computing resource usage.

that in the layered prototype. Because of the key-value smart contracts, B-RAN resource service contracts are optimized to record essential request information. In contrast, the Ethereum smart contracts consist of complex application binary interfaces and byte codes, inevitably resulting in extra overhead during request generation and transmission.

To demonstrate the impact of intelligent maintainers, we randomly involve 20% abnormal requests, including the ones with illegal service terms and the ones requesting unavailable resources. Fig. 16 shows the number of requests that SPs receive and actually serve. Due to the lack of node collaboration and request inspection capabilities, miners in the Ethereum-based implementation cannot recognize service details and perceive service information from SPs, and thus cannot stop abnormal requests from being spread. Therefore, after 4000 seconds, the gap between the received and served requests in the Ethereum-based implementation widens to 20%, whereas in the layered prototype, the abnormal requests are all intercepted and discarded by intelligent maintainers.

Furthermore, we measure and visualize the service latency distribution of the layered prototype and Ethereum-based implementation in Fig. 17. Because the FSCD mechanism can significantly shorten the confirmation delay in blockchain, one can see that the service latency in the layered prototype is remarkably lower than that in the Ethereum-based implementation.

C. Platform-Level Performance

1) Software flame graph analysis: To illustrate the performance improvement at the platform level, we capture the processor's stack traces of the maintainers, SPs, and clients, in the layered prototype and Ethereum-based implementation and visualize them as flame graphs. As shown in Fig. 18, the Ethereum-based implementation suffers from a number of issues. 1) The large plateaus indicate the existence of time-consuming tasks. 2) High dynamic stack traces show the presence of frequently varying function calls and intricate code branches. 3) Long resident stacks reflect complex daemon designs of the implementation. 4) Tedious message processing tasks occupy a substantial proportion of processor resources in SPs and clients. In contrast, the layered prototype solves these issues and ensures efficient operations. It has fine-grained stack traces and short resident stacks, indicating the flexible task scheduling and lightweight foundations of the

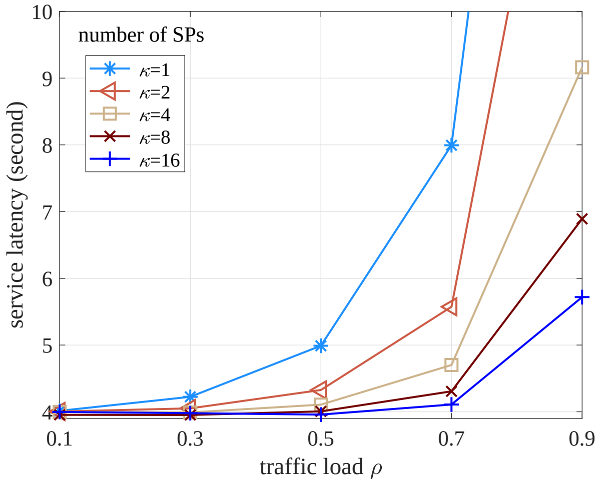


Fig. 20. Service latency under varying traffic load ρ.

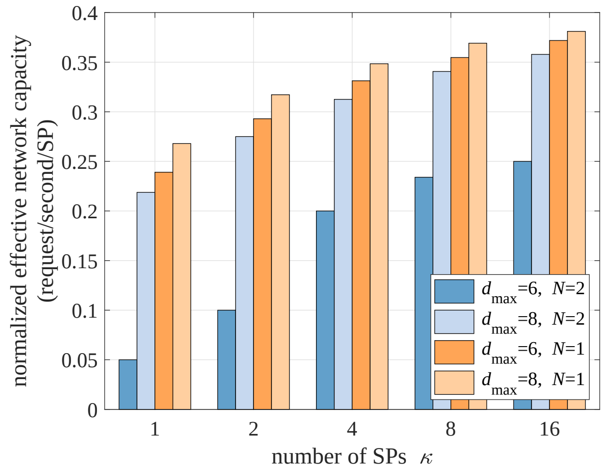


Fig. 21. Normalized effective network capacity under different number of SPs κ.

proposed architecture. Moreover, frequent small-scale tasks in the layered prototype, e.g., processing blocks and smart contracts, encrypting and decrypting messages, and signing transactions, are handled by architecture mechanisms and thus can be intelligently scheduled and rapidly completed without continually occupying processors.

2) Hardware resource consumption: In Fig. 19, we show the memory and computing resource consumption of the layered prototype and Ethereum-based implementation. We start the maintainer, SP, and client in tandem and monitor their resource consumption. In Fig. 19(a), after initiation, the memory usage of the Ethereum-based implementation keeps rising and finally stabilizes at 6.16 gigabytes, whereas the value of the layered prototype stabilizes at approximately 1.46 gigabytes which is almost a quarter of the Ethereum's. In terms of computing resource consumption shown in Fig. 19(b), the layered prototype occupies a lower amount of computing resources than the Ethereum-based implementation at the initiation stage. Since they both use the resource-consuming PoW consensus, their computing resource usage both float around 30% after B-RAN services begin. However, the consumption curve of the layered prototype is more "stable" than the Ethereum-based implementation, which verifies the conclusion in the above flame graph analysis.

3) Effect of SP virtualization: We present the pooling effect in B-RAN brought by the VSP with regard to the throughput and service latency, based on the layered prototype. Fig. 20 compares the service latency in the networks with different numbers of SPs, and each SP can provide services for at most four clients simultaneously. One can observe that more SPs, i.e., a larger network, can significantly reduce service latency, which shows the benefits from pooling in B-RAN.

Furthermore, in Fig. 21, we demonstrate the effective network capacity between networks with different numbers of SPs. The effective network capacity is defined as the maximum request arrival rate $\lambda^a = 1/T^a$ satisfying the constraint of delay-violation probability $\Pr\{D \geq d_{max}\} \leq \epsilon$, where $D \geq d_{max}$ means the client request waits for a delay longer than d_{max} and ϵ is set to 0.2. For comparing the performance between networks, we normalize the effective network capacity by the number of SPs. Under varying d_{max} and N, more SPs will distinctly increase the normalized effective network capacity. The results firmly support the effectiveness of a preliminary remark that *larger shared networks are more efficient and productive* [20].

X. Conclusion

In this work, we have established a unified architecture with enhanced efficiency, security, compatibility, and flexibility for resource sharing and trading in B-RAN for future 6G wireless networks. The developed architecture includes six layers and incorporates a number of novel features such as enhanced blockchain structures, secure interaction methods, efficient service mechanisms, and scalable transaction patterns. Starting from blockchain components and data storage to blockchain-based mechanisms and user interfaces, we have designed a number of pluggable modules in each layer to support diverse functions, services, and applications of resource sharing and trading for various network entities such as communication infrastructures, IoT devices, and wireless service equipment. We have proposed several lightweight designs such as portable databases and key-value smart contracts. We have also exploited several modules for improving the efficiency of resource sharing and trading, i.e., FSCD, multi-hop trading mechanism, and intelligent maintainers. To enhance the scalability and security, we have designed the eHTLC, VSP, IEP, pluggable consensus, and permission manager. Finally, based on the proposed architecture, we have implemented a portable, energy-efficient prototype using Python and conducted several experiments against an Ethereum-based implementation. The experiment results have shown a distinct superiority of the developed architecture in terms of transmission overhead, node collaboration, resource consumption, service latency, throughput, and network pooling of resource sharing and trading in B-RAN. The concepts of endogenous lightweight, inherent security, and demand-adaptive design adopted in the proposed architecture can be applied to future blockchain studies and projects.

References

[1] J. Wang, X. Ling, Y. Le, Y. Huang, and X. You, "Blockchain enabled wireless communications: a new paradigm towards 6G," *Natl. Sci. Rev.*, vol. 8, no. 9, Apr. 2021. [Online]. Available: https://doi.org/10.1093/nsr/nwab069

[2] X. Ling, J. Wang, T. Bouchoucha, B. C. Levy, and Z. Ding, "Blockchain radio access network (B-RAN): Towards decentralized secure radio access paradigm," *IEEE Access*, vol. 7, pp. 9714–9723, Jan. 2019.

[3] K. Kotobi and S. G. Bilén, "Blockchain-enabled spectrum access in cognitive radio networks," in *Proc. 2017 Wireless Telecommun. Symp. (WTS'17)*, Chicago, US, Apr. 2017, pp. 1–6.

[4] K. Kotobi and S. G. Bilen, "Secure blockchains for dynamic spectrum access: A decentralized database in moving cognitive radio networks enhances security and user access," *IEEE Veh. Technol. Mag.*, vol. 13, no. 1, pp. 32–39, Jan. 2018.

[5] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Tech. Report*, Oct. 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[6] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," *IEEE Trans. Serv. Comput.*, vol. 13, no. 2, pp. 289–300, Mar.-Apr. 2020.

[7] Y. Xu, C. Zhang, Q. Zeng, G. Wang, J. Ren, and Y. Zhang, "Blockchain-enabled accountability mechanism against information leakage in vertical industry services," *IEEE Trans. Network Sci. Eng.*, vol. 8, no. 2, pp. 1202–1213, Apr.-Jun. 2021.

[8] X. You, C.-X. Wang, J. Huang, X. Gao, Z. Zhang, M. Wang *et al.*, "Towards 6G wireless communication networks: vision, enabling technologies, and new paradigm shifts," *Sci. China Inf. Sci.*, vol. 64, no. 1, p. 110301, Nov. 2020.

[9] G. Hurlburt, "Might the blockchain outlive bitcoin?" *IT Prof.*, vol. 18, no. 2, pp. 12–16, Mar. 2016.

[10] S. Underwood, "Blockchain beyond bitcoin," *Commun. ACM*, vol. 59, no. 11, pp. 15–17, Oct. 2016.

[11] Y. Dai, D. Xu, S. Maharjan, Z. Chen, Q. He, and Y. Zhang, "Blockchain and deep reinforcement learning empowered intelligent 5G beyond," *IEEE Network*, vol. 33, no. 3, pp. 10–17, May 2019.

[12] M. Latva-aho, K. Leppänen, F. Clazzer, and A. Munari, *Key Drivers and Research Challenges for 6G Ubiquitous Wireless Intelligence*, Sep. 2019. [Online]. Available: https://elib.dlr.de/133477/

[13] M. Samaniego and R. Deters, "Blockchain as a service for IoT," in *Proc. 2016 IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData)*, Chengdu, CN, Dec. 2016, pp. 433–436.

[14] K. Malinova and A. Park, "Market design for trading with blockchain technology," *SSRN*, May 2016. [Online]. Available: https://ssrn.com/abstract=2785626

[15] P. Banerjee and S. Ruj, "Blockchain enabled data marketplace - design and challenges," *arXiv preprint*, Nov. 2018. [Online]. Available: http://arxiv.org/abs/1811.11462

[16] A. Hagiu and J. Wright, "Multi-sided platforms," *Int. J. Ind. Organ.*, vol. 43, pp. 162–174, Nov. 2015.

[17] Z. Xiong, S. Feng, W. Wang, D. Niyato, P. Wang, and Z. Han, "Cloud/fog computing resource management and pricing for blockchain networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4585–4600, Jun. 2019.

[18] Z. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, "When mobile blockchain meets edge computing," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 33–39, Aug. 2018.

[19] L. Xiao, Y. Ding, D. Jiang, J. Huang, D. Wang, J. Li *et al.*, "A reinforcement learning and blockchain-based trust mechanism for edge networks," *IEEE Trans. Commun.*, vol. 68, no. 9, pp. 5460–5470, May 2020.

[20] X. Ling, J. Wang, Y. Le, Z. Ding, and X. Gao, "Blockchain radio access network beyond 5G," *IEEE Wireless Commun.*, vol. 27, no. 6, pp. 160–168, Dec. 2020.

[21] X. Ling, Y. Le, J. Wang, Z. Ding, and X. Gao, "Practical modeling and analysis of blockchain radio access network," *IEEE Trans. Commun.*, vol. 69, no. 2, pp. 1021–1037, Feb. 2021.

[22] Y. Le, X. Ling, J. Wang, and Z. Ding, "Prototype design and test of blockchain radio access network," in *Proc. 2019 IEEE Intl. Conf. Commun. Workshops (ICC WKSP'19)*, Shanghai, CN, May 2019, pp. 1–6.

[23] X. Ling, P. Chen, J. Wang, and Z. Ding, "Data broker: Dynamic multi-hop routing protocol in blockchain radio access network," *IEEE Commun. Lett.*, Sep. 2021. [Online]. Available: https://doi.org/10.1109/LCOMM.2021.3114218

[24] X. Ling, Y. Le, J. Wang, and Z. Ding, "Hash access: Trustworthy grant-free IoT access enabled by blockchain radio access networks," *IEEE Network*, vol. 34, no. 1, pp. 54–61, Jan. 2020.

[25] X. Ling, B. Zhang, H. Xie, J. Wang, and Z. Ding, "Hash access in blockchain radio access networks: Characterization and optimization," *IEEE Internet Things J.*, Sep. 2021. [Online]. Available: https://doi.org/10.1109/JIOT.2021.3111915

[26] L. Zhang, H. Xu, O. Onireti, M. A. Imran, and B. Cao, "How much communication resource is needed to run a wireless blockchain network?" *arXiv preprint*, Jan. 2021. [Online]. Available: https://arxiv.org/abs/2101.10852

[27] D. R. Morrison, "Patricia-practical algorithm to retrieve information coded in alphanumeric," *J. ACM*, vol. 15, no. 4, pp. 514–534, Oct. 1968.

[28] X. Ling, Z. Gao, Y. Le, L. You, J. Wang, Z. Ding *et al.*, "Satellite-aided consensus protocol for scalable blockchains," *Sensors*, vol. 20, no. 19, p. 5616, Oct. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/19/5616

[29] V. Buterin, "A next-generation smart contract and decentralized application platform," *Tech. Report*, Jan. 2014. [Online]. Available: https://ethereum.org/en/whitepaper/

[30] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *Proc. 2016 Workshop Distrib. Cryptocurrencies Consensus Ledgers (DCCL'16)*, vol. 310, Zurich, CH, Jul. 2016.

[31] L. M. Bach, B. Mihaljevic, and M. Zagar, "Comparative analysis of blockchain consensus algorithms," in *Proc. 41st Int. Convention Inf. Commun. Technol. Electron. Microelectron. (MIPRO'18)*, Opatija, HR, May 2018, pp. 1545–1550.

[32] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," *Tech. Report*, Jan. 2016. [Online]. Available: https://lightning.network/lightning-network-paper.pdf

[33] K. Cattani and G. M. Schmidt, "The pooling principle," *INFORMS Trans. Educ.*, vol. 5, no. 2, pp. 17–24, Jan. 2005. [Online]. Available: https://doi.org/10.1287/ited.5.2.17

[34] J. Kwon, "Tendermint: Consensus without mining," *Tech. Report*, Aug. 2014. [Online]. Available: https://tendermint.com/static/docs/tendermint.pdf

[35] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *Proc. 3rd Symp. Oper. Syst. Des. Implementation (OSDI'99)*, vol. 99, no. 1999, New Orleans, US, Feb. 1999, pp. 173–186.

[36] S. King and S. Nadal, "PPCoin: Peer-to-peer crypto-currency with proof-of-stake," *Tech. Report*, Aug. 2012. [Online]. Available: https://decred.org/research/king2012.pdf

[37] J. Bae, E. Beigman, R. Berry, M. L. Honig, H. Shen, R. Vohra *et al.*, "Spectrum markets for wireless services," in *Proc. 3rd IEEE Symp. New Front. Dyn. Spectr. Access Networks (DYSPAN'08)*, Chicago, US, Oct. 2008, pp. 1–10.

[38] R. Berry, M. L. Honig, and R. Vohra, "Spectrum markets: Motivation, challenges, and implications," *IEEE Commun. Mag.*, vol. 48, no. 11, pp. 146–155, Nov. 2010.

[39] C. A. Gizelis and D. D. Vergados, "A survey of pricing schemes in wireless networks," *IEEE Commun. Surv. Tutorials*, vol. 13, no. 1, pp. 126–145, Mar. 2011.

[40] A. Asheralieva and D. Niyato, "Distributed dynamic resource management and pricing in the IoT systems with blockchain-as-a-service and UAV-enabled mobile edge computing," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1974–1993, Mar. 2020.

[41] B. J. Nelson, "Remote procedure call," Ph.D. dissertation, Carnegie Mellon Univ, Pittsburgh, US, May 1981.

[42] F. R. Yu, J. Liu, Y. He, P. Si, and Y. Zhang, "Virtualization for distributed ledger technology (vDLT)," *IEEE Access*, vol. 6, pp. 25 019–25 028, Apr. 2018.

Yuwei Le (S'19) received the B.E. degree in electrical engineering from Dalian Maritime University, Dalian, China, in 2017, and he is currently pursuing the Ph.D. degree with the school of information science and engineering, National Mobile Communications Research Laboratory, Southeast University, Nanjing, China. His research interests include wireless networks and blockchain technology.

Xintong Ling (S'15–M'18) is currently an Associate Professor with the National Mobile Communications Research Laboratory (NCRL), Southeast University, and also with the Purple Mountain Laboratories. He received the B.E. and Ph.D. degrees in electrical engineering from Southeast University, Nanjing, China, in 2013 and 2018, respectively. From 2016 to 2018, he was a visiting Ph.D. student with the Department of Electrical and Computer Engineering, University of California at Davis. His current research interests focus on future generation wireless communications and networks, including blockchain technologies, distributed networking systems, wireless communications, and signal processing.

Jiaheng Wang (M'10–SM'14) received the Ph.D. degree in electronic and computer engineering from the Hong Kong University of Science and Technology, Kowloon, Hong Kong, in 2010, and the B.E. and M.S. degrees from the Southeast University, Nanjing, China, in 2001 and 2006, respectively.

He is currently a Full Professor at the National Mobile Communications Research Laboratory (NCRL), Southeast University, Nanjing, China. From 2010 to 2011, he was with the Signal Processing Laboratory, KTH Royal Institute of Technology, Stockholm, Sweden. He also held visiting positions at the Friedrich Alexander University Erlangen-Nürnberg, Nürnberg, Germany, and the University of Macau, Macau. His research interests are mainly on communication systems and wireless networks.

Dr. Wang has published more than 150 articles on international journals and conferences. From 2014 to 2018, he served as an Associate Editor for the IEEE Signal Processing Letters. From 2018, he serves as a Senior Area Editor for the IEEE Signal Processing Letters. He was a recipient of the Humboldt Fellowship for Experienced Researchers and the best paper awards of IEEE GLOBECOM 2019, ADHOCNETS 2019, and WCSP 2014.

Ruiwei Guo (S'20) received the B.E. degree in information management and information system from China Jiliang University, Hangzhou, China, in 2019, and he is currently pursuing the M.S. degree with the school of cyber science and engineering, Southeast University, Nanjing, China. His research interests include blockchain technology and distributed system security.

Yongming Huang (M'10-SM'16) received the B.S. and M.S. degrees from Nanjing University, Nanjing, China, in 2000 and 2003, respectively, and the Ph.D. degree in electrical engineering from Southeast University, Nanjing, in 2007.

Since March 2007 he has been a faculty in the School of Information Science and Engineering, Southeast University, China, where he is currently a full professor. He has also been the Director of the Pervasive Communication Research Center, Purple Mountain Laboratories, since 2019. From 2008 to 2009, he visited the Signal Processing Lab, Royal Institute of Technology (KTH), Stockholm, Sweden. He has published over 200 peer-reviewed papers, hold over 80 invention patents. His current research interests include intelligent 5G/6G mobile communications and millimeter wave wireless communications. He submitted around 20 technical contributions to IEEE standards, and was awarded a certificate of appreciation for outstanding contribution to the development of IEEE standard 802.11aj. He served as an Associate Editor for the IEEE Transactions on Signal Processing and a Guest Editor for the IEEE Journal Selected Areas in Communications. He is currently an Editorat-Large for the IEEE Open Journal of the Communications Society and an Associate Editor for the IEEE Wireless Communications Letters.

Cheng-Xiang Wang (S'01-M'05-SM'08-F'17) received the B.Sc. and M.Eng. degrees in communication and information systems from Shandong University, China, in 1997 and 2000, respectively, and the Ph.D. degree in wireless communications from Aalborg University, Denmark, in 2004.

He was a Research Assistant with the Hamburg University of Technology, Hamburg, Germany, from 2000 to 2001, a Visiting Researcher with Siemens AG Mobile Phones, Munich, Germany, in 2004, and a Research Fellow with the University of Agder, Grimstad, Norway, from 2001 to 2005. He has been with Heriot-Watt University, Edinburgh, U.K., since 2005, where he was promoted to a Professor in 2011. In 2018, he joined Southeast University, Nanjing, China, as a Professor. He is also a part-time Professor with Purple Mountain Laboratories, Nanjing. He has authored 4 books, 3 book chapters, and more than 440 papers in refereed journals and conference proceedings, including 25 highly cited papers. He has also delivered 22 invited keynote speeches/talks and 9 tutorials in international conferences. His current research interests include wireless channel measurements and modeling, 6G wireless communication networks, and applying artificial intelligence to wireless communication networks.

Dr. Wang is a Member of the Academia Europaea (The Academy of Europe), a Fellow of the IEEE, IET, and China Institute of Communication (CIC), an IEEE Communications Society Distinguished Lecturer in 2019 and 2020, and a Highly-Cited Researcher recognized by Clarivate Analytics in 2017-2020. He is currently an Executive Editorial Committee Member of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He has served as an Editor for nine international journals, including the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, from 2007 to 2009, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, from 2011 to 2017, and the IEEE TRANSACTIONS ON COMMUNICATIONS, from 2015 to 2017. He was a Guest Editor of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, Special Issue on Vehicular Communications and Networks (Lead Guest Editor), Special Issue on Spectrum and Energy Efficient Design of Wireless Communication Networks, and Special Issue on Airborne Communication Networks. He was also a Guest Editor for the IEEE TRANSACTIONS ON BIG DATA, Special Issue on Wireless Big Data, and is a Guest Editor for the IEEE TRANSACTIONS ON COGNITIVE COMMUNICATIONS AND NETWORKING, Special Issue on Intelligent Resource Management for 5G and Beyond. He has served as a TPC Member, a TPC Chair, and a General Chair for more than 80 international conferences. He received 14 Best Paper Awards from IEEE GLOBECOM 2010, IEEE ICCT 2011, ITST 2012, IEEE VTC 2013-Spring, IWCMC 2015, IWCMC 2016, IEEE/CIC ICCC 2016, WPMC 2016, WOCC 2019, IWCMC 2020, WCSP 2020, CSPS2021, and WCSP 2021. Also, he received the 2020 & 2021 "AI 2000 Most Inuential Scholar Award Honourable Mention" in recognition of his outstanding and vibrant contributions in the field of Internet of Things.

Xiaohu You (Fellow, IEEE) received the M.S. and Ph.D. degrees in electrical engineering from Southeast University, Nanjing, China, in 1985 and 1988, respectively.

Since 1990, he has been with the National Mobile Communications Research Laboratory, Southeast University, where he is currently the Director and a Professor. From 1999 to 2002, he was the Principal Expert of the C3G Project, responsible for organizing China's 3G mobile communications research and development activities. From 2001 to 2006, he was the Principal Expert of the China National 863 Beyond 3G FuTURE Project. Since 2013, he has also been the Principal Investigator of China National 863 5G Project. He has contributed over 200 IEEE journal articles and two books in the areas of adaptive signal processing and neural networks and their applications to communication systems. His research interests include mobile communication systems and signal processing and its applications.

Prof. You was selected as an IEEE Fellow for his contributions to the development of mobile communications in China in 2011. He was a recipient of the National 1st Class Invention Prize in 2011. He served as the General Chair for the IEEE Wireless Communications and Networking Conference 2013, the IEEE Vehicular Technology Conference 2016, and the IEEE International Conference on Communications 2019. He is the Secretary General of the FuTURE Forum, the Vice Chair of the China IMT-2020 Promotion Group, and the Vice Chair of China National Mega Project on New Generation Mobile Network.