Research on Service Composition Strategies in Cloud Computing Environments Based on Blockchain Mechanisms (Postprint)

Authors: Wang Lei, Zhao Xiaoyong

Date: 2018-05-20T00:00:00+00:00

Abstract

Service composition in cloud computing environments exhibits characteristics of large scale, high complexity, diverse failure types, and dynamic resource variations; consequently, secure and trustworthy service composition solutions constitute the foremost challenge for the flexible deployment and on-demand delivery of massive service resources in cloud settings. Initially, we present an in-depth analysis of blockchain consensus mechanisms, smart contracts, and programmable features, draw analogies with service composition characteristics and workflows, and propose a service composition architecture with enhanced trustworthiness and reliability that employs blockchain as the underlying infrastructure, grounded in service overlay networks (SON) theory. Moreover, considering the security policies inherent to blockchain consensus mechanisms and leveraging the functionality and utility of smart contracts, we propose an efficient service path generation algorithm across service overlay layers based on optimal strategies for link-predicted survival time and service intensity. Furthermore, we devise a trustworthy service quality model, ToS (trust of service), incorporating metrics for single-point trustworthiness and interaction closeness, along with a ToS-driven service path selection algorithm. Simulation results demonstrate that the blockchain mechanism-based service composition implementation strategy substantially enhances the success rate and availability of composite service execution.

Full Text

Preamble

Research on Service Composition Strategy Based on Blockchain Mechanism in Cloud Computing Environment

Wang Lei, Zhao Xiaoyong

(School of Information Management, Beijing Information Science & Technology University, Beijing 100192, China)

Abstract: Service composition in cloud computing environments is characterized by large scale, high complexity, diverse failure types, and dynamic resource changes. Consequently, secure and trustworthy service composition schemes represent a major challenge for flexible deployment and on-demand provisioning of massive service resources in cloud environments. This paper first conducts an in-depth analysis of blockchain consensus mechanisms, smart contracts, and programmable features, drawing analogies with service composition characteristics and processes. Based on service overlay networks (SON) theory, we propose a novel service composition architecture with enhanced credibility and reliability that uses blockchain as its underlying infrastructure. Furthermore, considering the security strategies derived from blockchain consensus mechanisms and leveraging the functions and roles of smart contracts, we propose an efficient service path generation algorithm across service overlay layers based on predictive route lifetime and service intensity optimization strategies. Additionally, we design a trusted quality of service model, ToS (Trust of Service), incorporating single-node trust and interaction tightness metrics, along with a service path selection algorithm driven by this ToS model. Simulation results demonstrate that the blockchain-based service composition implementation strategy significantly improves the success rate and credibility of composite service execution.

Keywords: blockchain; smart contract; service composition; credibility

CLC number: TP302.1 **doi:** 10.3969/j.issn.1001-3695.2017.07.0688

0 Introduction

The contradiction between the complexity of requirements and the expected reliability and availability of service composition has become increasingly pronounced. With the continuous maturation and development of cloud computing, big data, and other technologies, information processing architectures have evolved from loosely-coupled single-point computing models in early Web service composition to "X as a Service"cloud resource frameworks, transforming into distributed, collaborative, dynamically reconfigurable, and ubiquitous multi-state fusion frameworks. All resources required to support upper-layer computing—including computation, storage, network, and data—are provided to legitimate terminals in the form of services or service compositions. While this brings abundant foundational resources for upper-layer applications, it also introduces significant challenges: the heterogeneity and dynamism of resources, combined with user demands, create a complex environment where most service composition research achievements cannot be directly applied to distributed cloud computing environments due to the high dynamism of service resources, service links, and diverse failure types. Therefore, exploring a service composition technology framework with strong universality and high credibility has become

one of the key scientific problems urgently needing resolution in the service computing domain.

Blockchain, as the fifth-generation computing paradigm following mobile and social networks, focuses on ensuring trust as its core mission and promotes efficient operation in transactions, authentication, and multiple other aspects, bringing disruptive innovative ideas to the research of reliable, secure, and trustworthy service composition. This paper first investigates the basic architecture, fundamental principles, and characteristics of blockchain technology, then conducts feature comparison and in-depth analysis between blockchain operation mechanisms and critical aspects such as service resource discovery, selection, and composition. We identify common problems in the dynamic operation mechanisms of service composition and blockchain, abstracting the correspondence between service resources and blockchain nodes, service discovery and consensus algorithms, service selection and smart contracts, and service composition and block appending. Building upon this foundation and based on blockchain infrastructure and service overlay networks (SON) theory, we propose a blockchain-based service composition architecture. This framework employs blockchain' s inherently secure and reliable consensus strategies and highly programmable smart contracts to construct the service composition execution framework. Upon this architecture, we propose an efficient service path generation algorithm across service overlay layers based on predictive route lifetime and service intensity optimization strategies. Finally, we construct a trusted quality of service model, ToS, to drive the selection of highly credible service paths. Experimental results demonstrate that the proposed architecture, platform model, and algorithms possess correctness and applicability, providing beneficial supplements for research on distributed service resource composition in cloud computing environments.

---

1 Research Status of Service Composition

Since the early research on Web service composition, scholars have been dedicated to unified management, efficient scheduling, and trustworthy composition of service resources in service computing. Although no unified concept of service composition has been formed, in current cloud computing and big data domains, various resources—including computation, storage, network, and applications—are coordinated, synthesized, or orchestrated to meet user application requirements, ultimately providing upper layers with various IaaS/SaaS/PaaS/NaaS, and even DaaS cloud services that satisfy complex user tasks, achieving consistent and reliable service resource guarantee and support.

In fact, the contradiction between cloud computing' s prominent characteristics of openness, sharing, dynamism, and transparency, combined with resource heterogeneity, and the reliability and availability of composite services has become increasingly evident. In recent years, more and more researchers have focused

on the reliability and credibility of service composition, achieving remarkable results. For instance, in the direction of trustworthy service composition research, reference [1] proposed a trust-based distributed service composition method for mobile social networks, establishing a distributed service trust evaluation framework based on lattice models. Reference [2] proposed a cross-private-cloud reliability service composition method for big data applications, using K-means clustering for QoS history record selection to drive service component selection, significantly improving the effectiveness of cross-cloud service composition and the reliability and security of private clouds. Reference [3] proposed a Web service trust architecture based on secure interaction, federated identity, and distributed policies, providing an effective reference for Web service trust research.

In service fault tolerance mechanism research, a mainstream approach addresses benign or malicious faults in service execution based on the Byzantine Generals Problem to ensure normal service provision and operation. For example, reference [4] proposed an improved modular Web Service framework capable of satisfying Byzantine fault tolerance for commonly used fault tolerance techniques in Web Services, greatly enhancing the reliability and availability of fault tolerance in Web services. Reference [5] designed a Checkpoint protocol based on Byzantine fault tolerance for Web services, creating checkpoints periodically among replicas. These checkpoints provide historical data during replica state transitions, significantly improving the availability and reliability of service composition. Reference [6] proposed a cross-cloud Byzantine fault tolerance model that introduces Shamir' s secret sharing key distribution and recovery algorithm, substantially enhancing system reliability and cloud data security. Additionally, other research ideas and solutions for service composition trustworthiness continue to emerge, such as reference [7], which implements targeted online processing strategies based on the severity of runtime faults in Web service composition to maximize service execution guarantee, and reference [8], which proposes a heuristic algorithm-based multi-objective optimization service trust strategy for wireless Ad hoc networks. This strategy achieves optimal matching between trusted nodes and tasks under dynamic networks and personalized service demands while employing a blacklist mechanism to eliminate untrusted nodes, maximizing service availability and reliability.

---

2 Correspondence Analysis Between Blockchain Principles and Service Composition Process

With the groundbreaking application of digital cryptocurrency represented by Bitcoin, emerging blockchain technology has gradually become a hot research topic in academia and industry. Blockchain' s characteristics of decentralization, consensus mechanisms, and programmability give it broad application prospects in digital cryptocurrency, finance, and social systems [9~13].

In fact, as a universal underlying technology framework, blockchain will integrate into fields and industries beyond finance in the blockchain 3.0 model characterized by a programmable society, providing decentralized data structures and third-party-trust-free interaction mechanisms for distributed computing, distributed social systems, and distributed artificial intelligence, laying a solid credit foundation and feasible technical reference.

Blockchain is essentially a cloud-like distributed infrastructure where blocks are chained together. Nodes participate in the blockchain network based on value exchange protocols, forming a transaction database shared by all nodes within the system. Similar to this distributed consensus mechanism, the research approach to service composition in service computing also discovers and selects specific service nodes from distributed service resources to join the service execution path, forming composite services that meet user needs, thereby achieving unified management, integration, and scheduling of service resources to support complex upper-layer applications.

Drawing analogies between the decentralized and distributed computing paradigm characteristics of service resource composition in cloud computing and blockchain technology, this paper extracts and analyzes features from both technologies, yielding the conclusions shown in Table 1 .

Table 1 Comparison of Blockchain Technology and Service Composition Features

Blockchain Technology	Service Composition
Decentralization: P2P nodes with equal rights and obligations	Distributed resources equally dispersed
Collaborative autonomy: Nodes jointly maintain consistent data	Service components jointly maintain service paths
Consensus mechanism: Computing power competition ensures data consistency	Adaptive self-organization meets upper-layer application requirements
Smart contracts: Automatically executable contract algorithms	Automatically execute service selection and optimization algorithms

a) Both blockchain and service composition embody the idea of decentralization. In blockchain systems, there is no centralized database; each node maintains complete blockchain information with equal rights and obligations. In service composition, distributed service resources are dispersed and equal, with service components discovering and selecting based on functional and non-functional quality of service (QoS) attributes, and service nodes carrying various functions are independent and equal.

b) Both blockchain and service composition embody autonomous collaboration. Blockchain systems are operated and maintained jointly by all nodes

in the network without central control units. Similarly, in service composition, distributed service resources are dynamically scheduled, deployed, and composed according to established service composition strategies.

c) Both blockchain and service composition have consensus mechanisms. The core advantage of blockchain technology is that through encapsulating various consensus algorithms of network nodes, it enables nodes in highly decentralized systems with dispersed decision-making power to reach consensus on block data validity. The consensus in service composition lies in nodes jointly maintaining various dimensions of QoS attributes required by upper-layer applications through service links. Once a failure occurs during service provision, service resources update or reconstruct service paths in an adaptive and self-organizing manner.

d) Both blockchain and service composition have intelligent and contractual characteristics. Blockchain systems can provide flexible code scripting systems that support users in creating advanced smart contracts to implement complex functions executed and deployed automatically and intelligently across all nodes in the network. Service composition, as the execution and deployment of service paths, also has programmable characteristics. By setting discovery, selection, and execution path optimization processes for underlying service components according to different application requirements, flexible and programmable service resource combination strategies are realized.

The comparative analysis of features between blockchain and service composition is illustrated in Figure 1 [Figure 1: see original paper].

---

3 Blockchain-Based Service Composition Architecture

3.1 Design of Service Composition Architecture Based on Service Overlay Networks (SON)

Drawing on the design philosophy of service overlay networks (SON), this research logically divides the blockchain-based service composition architecture into three layers: physical network layer, service instance layer, and service abstraction layer. By establishing mapping relationships among functions, services, and routing, and adopting dynamic overlay deployment strategies, we abstract a fourth layer: the service overlay layer. The architecture is shown in Figure 2 [Figure 2: see original paper].

a) **Service Abstraction Layer**: Defines the interface between user requests and the distributed service composition system, decomposing a user' s functional requirements into a series of functional services.

b) **Service Instance Layer**: Implements dynamic selection and binding from abstract functional services to concrete service instances during ser-

vice execution, differing from traditional QoS-driven service instance selection and service path generation. This paper constructs a trusted quality of service-driven ToS model and uses smart contracts to encapsulate heuristic trusted service search algorithms to determine trusted service composition paths.

c) **Network Layer**: Encapsulates elements such as blockchain system networking methods, message propagation protocols, and data verification mechanisms, providing basic physical layer transmission, network communication, and network layer routing functions. Each node undertakes functions including network routing, block data verification, block data propagation, and new node discovery. To improve computing performance, nodes can be divided into full nodes and lightweight nodes—the former stores complete blockchain data, while the latter mainly provides routing functions to complete data verification. It provides necessary communication nodes, communication protocols, and physical connection link support to upper layers (service instance layer) and carries underlying node mobility and location information.

d) **Service Overlay Layer**: The service overlay layer is a virtual layer constructed through feature extraction and abstraction of the physical network layer, service instance layer, and service abstraction layer. It generates and maintains overlay network topology while providing various application layer services and physical layer node mobility information for the service composition process, ultimately determining service composition execution paths on the service overlay layer.

3.2 Design of SON+Blockchain Infrastructure-Based Service Composition Architecture

Building upon this foundation, we present the blockchain basic architecture and interaction mechanisms in the service overlay layer architecture, as shown in Figure 3 [Figure 3: see original paper]. Node1 and node2 are distributed nodes existing in the Internet or Ethernet, which can be virtual machines, servers, or cloud nodes of different types. We design a blockchain-based hierarchical overlay architecture for the routing connection between node1 and node2, from top to bottom as follows:

a) **User Information Layer**: Contains basic information of users A and B, including public and private keys for user A and public and private keys for user B. When sender A encrypts information using receiver B' s public key before sending to B, decryption is performed using their own private key.

b) **P2P Network Layer**: The distributed network includes distributed networking mechanisms, data transmission mechanisms, and data verification mechanisms. It uses broadcast forms to send query requests to nodes in the network, serving as the basic implementation layer of underlying

routing mechanisms to support upper-layer service discovery and service composition.

c) **Service Composition (Smart Contract Layer)**: Mainly encapsulates various scripts, algorithms, and smart contracts. This is the foundation of blockchain' s programmable characteristics and an important core module for completing trusted service composition. This paper places the critical steps of service component selection and service execution path determination in on-chain code execution.

d) **Consensus Layer**: Uses consistency protocols to maintain data consistency requirements in distributed environments. The consensus layer mainly encapsulates various consensus algorithms of network nodes to implement blockchain consensus mechanisms. This paper uses Proof of Work (PoW) to complete consistency guarantees.

e) **Data Layer**: Encapsulates underlying data blocks and related data encryption, timestamps, and some necessary smart contract execution code, including transaction caching, block caching, and runtime data caching. This paper assumes that in low-speed environments, transactions are processed serially.

---

4 Service Composition Strategy and Algorithm Design

4.1 Smart Contracts Based on Trusted Quality of Service (ToS)

In the aforementioned SON+blockchain architecture, this paper designs service composition strategies and algorithms, proposing a trusted service composition quality ToS model to drive service path selection and execution. Based on the programmable characteristics of blockchain, we construct smart contracts by encapsulating service composition strategy and algorithm-related program code, predefined states, transition rules, scenarios that trigger contract execution, and corresponding actions for different scenarios. Blockchain monitors smart contract status in real-time and activates and executes contracts after verifying external data sources and confirming that specific trigger conditions are met. Various nodes in the network either automatically execute or simply forward contracts until the service lifecycle ends and service resources are withdrawn.

The trigger condition for smart contracts is the user' s service composition request. The service abstraction layer parses and decomposes the service composition request, mapping it to a logical composition scheme that satisfies functional and non-functional requirements, generating a service path set. The trusted service composition process is shown in Figure 5 [Figure 5: see original paper].

The operational mechanism of smart contracts is illustrated in Figure 4 [Figure 4: see original paper]. Typically, after being signed by all parties, smart contracts are attached to blockchain data in the form of program code, recorded in specific

blocks of the blockchain after P2P network propagation and node verification. Blockchain can monitor the status of smart contracts in real-time.

4.2 Analysis and Design of Trusted Quality Model (QoT)

This paper models trusted quality QoT (quality of trust) as an evaluation of the trust level of service paths during service component selection. Drawing on existing research results [14], trusted quality can be simplified and modeled as a trust description comprising two attribute sets: single-node trust and interaction tightness.

a) **Single-node Trust**: The ability of a single service component (excluding source and destination nodes) to provide trustworthiness level guarantees after fusing various credible evidence, denoted as $tetToS$. Aggregated along the service path direction, the impact of a single point on the entire composite service is arg,$tetToSQQoTrgtetQQoT$.

b) **Interaction Tightness**: Defined between service components participating in service path construction, representing the compatibility level determined based on component development information. Component tightness is denoted as $tetToS$, and arg,$tetToSQQoTrgtetQQoT$ on the trusted service path should follow a nonlinear principle that weakens with hop count. The aggregation method is as follows:

We establish a utility function to measure the trust level construction of trusted service paths. The utility function based on QoT attributes is as follows, where α and β are the weight values for single-node trust and interaction tightness, respectively.

4.3.1 Service Path Generation

This paper adopts the classic directed acyclic graph (DAG) theory to design the service path generation algorithm. Given the high dynamism of cloud computing service resources, we propose a path generation strategy based on service intensity and link lifetime to optimally select the next service instance. In the service overlay layer SON, we extract service node information and service link information, calculating the service intensity of service components that implement the next function i at the current node i and the predictive link lifetime of links where service components are located. After weighted averaging and sorting, we select the optimal and suboptimal service components as service selection return values to generate the final service path.

To improve service path stability and reliability, this paper considers selecting service components with longer link lifetimes and greater service intensity to complete corresponding functions. This constitutes a subgraph $G'(V', E')$, where vertices V' are instantiated service components njs in corresponding service areas, and edges E' represent input-output relationships between two service instances i and predecessor node j, while achieving optimal weighted

averaging of service intensity and link connectivity time and satisfying QoS consistency.

We first provide definitions for link lifetime and service intensity.

Definition 1: Link Lifetime. The route lifetime (RLT) is determined by the minimum value of link lifetimes maintained by adjacent nodes, i.e., $if\{\}ijRLTMinLETl =$, where ijl represents the wireless link between two adjacent nodes in the routing link, and $ijLETl$ is the predictive lifetime of link ijl. Link breakage occurs when a node moves out of another node' s communication range.

Definition 2: Service Intensity (SI). Service intensity SI is a quantitative expression of the ability of vehicle terminal i to acquire service component j, denoted as $iNjAbis$, where $iNjAbis$ is the capability of node i to provide service j, proportional to its transmission power, $disNN$ represents the hop count from current node i to node j, and α is a constant that does not affect service discovery decisions.

Based on the above dimensional descriptions, the specific steps of the efficient service instance selection algorithm based on predictive link lifetime and service intensity optimization are as follows:

a) Assume the composite service has automatic functional decomposition capability. The service composition scheme DAG graph generated based on user service composition requests contains m service components: $\{S_1, S_2, ..., S_m\}$. The system predefines the service area to which each service component belongs that has the same function: $S_j = \{s_{j1}, s_{j2}, ..., s_{jn_j}\}$, where s_{jk} represents specific service instances. As shown in Figure 6 [Figure 6: see original paper], nodes represent service areas that complete certain functions (tasks).

b) In the DAG graph, based on the following steps, determine whether the current node speed iV and predecessor node speed jV maintain QoS consistency: (a) Assume the service areas to which current node i and predecessor node j belong are $iSAS$ and $jSAS$, respectively.

c) Extract node service information, node mobility information, and physical network layer routing link status in the service overlay layer. Calculate the service intensity $iNjAbis$ of service components implementing the next function i at current node i and the predictive link lifetime ijt according to Definitions 1 and 2. After weighted averaging and sorting, select the optimal and suboptimal service components as service selection return values to generate the final service path.

The following is the service path generation algorithm proposed in this paper.

Algorithm: Generate service paths considering RLT and SI

The algorithm enhances service path stability and reliability while minimizing

the service path solution space, improving service composition success rate and execution efficiency by having the current state jointly decide and drive the instantiation process of service components in the functional graph.

4.3.2 Service Path Selection Algorithm

Among existing service paths that satisfy user functional requirements, we select the path with the strongest credibility and availability as the final execution scheme for service composition. Therefore, this section proposes a trusted quality of service ToS supporting the SON+blockchain structure to evaluate service path performance.

Assume a five-dimensional attribute set is used as the metric for service path performance: availability, price, and trusted quality. In fact, different users have different preferences for various service quality attributes. Therefore, a key step in the second-level scoring process is setting weights for each QoS indicator. Weight determination must consider both user subjective feelings and objective facts. Thus, this paper adopts multiple attribute decision making (MADM) theory and a combination of subjective and objective weighting methods to solve service quality attribute weights [15].

The parameter evaluation matrix is expressed as $M = [q_{ij}]_{5\times r}$, where each row represents the same service quality attribute across different service paths, and each column corresponds to a service execution path P_j in the available service path set $\{P_1, P_2, ..., P_r\}$. w_i is the weight of service attribute i, $w_i \in [0, 1]$, and $\sum_{i=1}^{5} w_i = 1$.

To unify measurement units for different ToS indicators, we need to standardize different types of ToS attributes so that quantitative values fall within $[0, 1]$. For benefit-type attributes (higher values indicate better quality) and cost-type attributes [15] (lower values indicate better quality), we provide the following standardization formulas:

For benefit-type: $a_{ij} = \frac{q_{ij} - q_j^{min}}{q_j^{max} - q_j^{min}}$ if $q_j^{max} \neq q_j^{min}$, otherwise $a_{ij} = 1$

For cost-type: $a_{ij} = \frac{q_j^{max} - q_{ij}}{q_j^{max} - q_j^{min}}$ if $q_j^{max} \neq q_j^{min}$, otherwise $a_{ij} = 0$

where $q_j^{max} = \max_i(q_{ij})$ and $q_j^{min} = \min_i(q_{ij})$.

After standardization, we obtain the standardized matrix $N = [a_{ij}]_{5\times r}$.

After standardization, we conduct a two-level scoring process for service execution paths: The first-level scoring is user constraint level, filtering out service execution paths that do not satisfy user condition constraints (e.g., delay cannot exceed 20 seconds). The second-level scoring further evaluates the filtered paths, with final performance determined by score magnitude to obtain the user service request response.

The simplified quality parameter matrix can be expressed as $M = [q_{ij}]_{n\times m}$. The

weight values can be determined by the following formula: $w_k = d_k / \sum_{j=1}^{m} d_j$, where d_k represents the weight determined by the subjective-objective model.

The solution process is as follows. We construct the following multi-objective programming model to solve:

$\min f = \alpha \sum_{k=1}^{m} \sum_{j=1}^{m} (w_k - w_j d_{kj})^2 + \beta \sum_{i=1}^{n} \sum_{j=1}^{m} (b_{ij} w_j - b_i^*)^2$

subject to: $\sum_{j=1}^{m} w_j = 1 \; w_j \geq 0, j = 1, 2, ..., m$

where α and β represent the relative importance of subjective and objective weighting modes, satisfying $\alpha + \beta = 1$, $0 < \alpha, \beta < 1$.

The Lagrange function is constructed as: $L(w, \lambda) = \alpha \sum_{k=1}^{m} \sum_{j=1}^{m} (w_k - w_j d_{kj})^2 + \beta \sum_{i=1}^{n} \sum_{j=1}^{m} (b_{ij} w_j - b_i^*)^2 + 2\lambda(\sum_{j=1}^{m} w_j - 1)$

The solution yields the weight vector $w^* = (Z^T Z)^{-1} Z^T e / (e^T (Z^T Z)^{-1} Z^T e)$, where Z is an $m \times m$ matrix with elements $z_{ij} = \alpha d_{ij} - \beta b_{ij}$.

The second-level screening results can be obtained accordingly. Notably, if multiple candidate execution paths remain after the second scoring, this paper employs a further optimal service elimination mechanism: by comparing the relative distance between service providing nodes and service requesting nodes on the service execution path, the path with the minimum distance is returned as the user request response.

Algorithm: Blockchain-based service path selection

Input: A set of candidate paths that each fulfills a given task, represented as a 5-element vector $C = \{c_1, c_2, ..., c_5\}$

Output: An optimal candidate path that fulfills task initialization

1. Create matrix M
2. I-level filtering: For $i = 1$ to 5 do if $q_{ij} > c_i$ then delete column j (i.e., path P_j) from M
3. A-level filtering: Let r be the number of columns in updated M
4. Generate normalized quality vector N from M using standardization formulas
5. Use formula $Score(P_i) = \sum_{i=1}^{5} w_i a_{ij}$ to generate an r-element score vector
6. Sort paths according to score values in descending order into V_1
7. Distance-based filtering & result return: Set k as the number of elements in V_2 that all have the highest score. Set $x = L$ (the maximum diagonal length of the network concerned). For each path, calculate the distance y between the requesting node and the service-providing node. If $y < x$ then $x = y$. Return the first path where $score(P_i) = elementV(2, 1)$.

5 Simulation and Analysis

5.1 Simulation Scenario Setup

The simulation environment in this paper adopts the Audit Simulation Cloud Platform of Beijing Information Science & Technology University. This platform is a high-performance computing and service sharing platform built based on the National Science and Technology Support Program project to meet IaaS virtual scheduling and management requirements. The platform uses three Dawning TIANKUO 1620-G10 servers and one Dawning TIANKUO 1840-GS server as the core architecture, along with 15 high-performance PCs as the hardware foundation of the entire cloud platform. The existing hardware environment can simulate a high-performance computing environment.

Simulation experiments are conducted on the open-source simulation framework CloudSim 3.0. This experiment simulates three medium-scale infrastructure service providers by writing and setting gateway components, constructs a multi-cloud, multi-domain virtual network system, and simulates the network service environment of blockchain. Based on this architecture design, the simulation data uses real-time streaming media files from the online video database of the Beijing Information Science & Technology University campus network. We assume that the transmission delay of smart contracts between sub-clouds is randomly generated within the range of [5~200] ms, and the QoS attributes of each atomic service are randomly generated based on the public dataset QWS (http://www.uoguelph.ca/~qmahmoud/qws/index.html). We assume that each node carries four basic services $\{s_1, s_2, s_3, s_4\}$, corresponding to functions $\{f_1, f_2, f_3, f_4\}$. These service resources can be randomly composed to form composite services containing n tasks that satisfy user requirements. Customer nodes are randomly selected in the network, generating service composition requests every 10 ms.

5.2 Experimental Simulation and Analysis

This section compares the blockchain-based service path selection algorithm proposed in this research with the classic discover-then-compose algorithm and random algorithm, primarily evaluating service composition success ratio (SCSR):

$$SCSR = \frac{\text{Number of successfully executed service paths}}{\text{Total number of service composition requests}}$$

First, we reveal the service composition success rate of each algorithm under different service densities. In this experiment, service density refers to the ratio of service replicas completing a certain function to the total number of nodes. We set it to increase from 5% to 30% in steps of 5%. As shown in Figure 7 [Figure 7: see original paper], whether in static or dynamic environments, the blockchain-based service composition strategy proposed in this paper demonstrates significant advantages in service composition success rate compared to other classic algorithms Dis-Comp and Random-Comp. For example, in dynamic environments with a service density of 30%, the service composition suc-

cess rates of Dis-Comp and Random-Comp are 81% and 42.5%, respectively, while the blockchain-based approach achieves a success rate as high as 96.5%. For each algorithm, the service composition success rate increases with service density, primarily because increased service density leads to more available service nodes, providing more opportunities to establish and select better service execution paths. Comparing Figures 7(a) and 7(b), we can see that the service composition success rate of the blockchain-based algorithm in dynamic environments is higher than in static environments. This is because node mobility brings more available service resources, and the blockchain-based service composition enhances the credibility and reliability of service paths, greatly improving the probability of establishing optimal service paths.

Next, we discuss the impact of node mobility speed on service composition availability in different service composition algorithms. We set the average node movement speed to 5, 10, ···, 30 m/s (with a step size of 5 m/s) and conduct simulation experiments on the blockchain-based service path selection algorithm, Dis-Comp, and Random-Comp algorithms in such dynamic environments. As shown in Figure 8 [Figure 8: see original paper], the availability of service execution paths established by all three algorithms declines with increasing node speed. In comparison, the service composition strategy proposed in this paper demonstrates better performance, achieving approximately 50% service availability even at a speed of 30 m/s. This is because the blockchain-based service composition algorithm using a two-level scoring strategy fully relies on trusted quality of service for trusted service component selection during the service execution path selection process, thereby enhancing the stability and reliability of composite services and ensuring the highest service availability.

---

6 Conclusion

Currently, the most important applications of blockchain mainly appear in the digital cryptocurrency domain. Although theoretical research and applications are experiencing explosive growth, application scenarios and fields remain relatively limited, unable to provide many beneficial references.

This paper keeps pace with technological frontiers and experimentally migrates blockchain basic theory and architecture to the service computing domain. We first conduct in-depth analysis and draw on blockchain' s inherent characteristics of decentralization, consensus mechanisms, and smart contracts. Based on blockchain infrastructure and service overlay networks (SON) theory, we propose a blockchain-based service composition architecture that uses blockchain as the underlying framework to support and guide key processes of service composition, including service discovery, service selection, service path generation, and optimal service path determination. Simulation results prove that the proposed service composition framework, trusted quality of service model ToS, service path generation algorithm, and optimal service path selection algorithm possess

correctness, effectiveness, and rationality. This service composition strategy significantly improves the success rate and availability of composite service execution in cloud computing environments.

This research approach will provide beneficial supplements for research on credibility and reliability of service composition in cloud computing environments and offer more effective solutions for service resource composition and optimization in complex network environments, providing more effective routing strategies to support the operational efficiency of upper-layer applications.

References

[1] Zhang Tao, Ma Jianfeng, Xi Ning, et al. A trust-based distributed service composition method for service-oriented mobile social networks [J]. Acta Electronica Sinica, 2016, 44(2): 258-267.

[2] Dou Wanchun, Zhang Xuyun, Liu Jianxun, et al. HireSome-II: towards privacy-aware cross-cloud service composition for big data applications [J]. IEEE Trans on Parallel and Distributed Systems, 2015, 26(2): 455-464.

[3] Liu Lingxia, Wang Dongxia, Huang Minheng, et al. A Web service trust architecture [J]. Computer Science, 2014, 41(12): 30-32.

[4] Wang Jiaxiang. Research and application of Byzantine fault tolerance algorithm in Web Service provision [D]. Jinan: Shandong University, 2009.

[5] Zhou Wei, Chen Liu. Checkpoint protocol based on Byzantine fault tolerance [J]. Computer and Modernization, 2013, (11): 104-107.

[6] AlZain M A. A Byzantine fault tolerance model for a multi-cloud computing [C]// Proc of the 16th IEEE International Conference on Computational Science and Engineering. 2013: 130-137.

[7] Zou Fang, Gao Chunming. Fault-tolerant architecture in Web service composition operation [J]. Computer Engineering, 2008, 34(18): 89-92.

[8] Wang Yating, Chen I R, Cho J H, et al. Trust-based task assignment with multi-objective optimization in service-oriented ad hoc networks [J]. IEEE Trans on Network and Service Management, 2016, 14(1): 217-232.

[9] Beck R, Stenum C J, Lollike N, et al. Blockchain-the gateway to trust-free cryptographic transactions [C]// Proc of the 24th European Conference on Information Systems. 2016: 1-14.

[10] Brandvold M, Molnár P, Vagstad K, et al. Price discovery on Bitcoin exchanges [J]. Journal of International Financial Markets Institutions and Money. 2015 (36): 18–35.

[11] Buehler K, Chiarella D, Heidegger H, et al. Beyond the Hype: Blockchains in Capital Markets [J/OL]. 2015 [2016-09-23]. https://www.weusecoins.com/assets/pdf/library/McKinsey%20Blo

[12] Carminati B, Ferrari E, and Tran N H. Trustworthy and effective person-to-person payments over multi-hop MANETs [J]. Journal of Network and Computer Applications, 2016 (60): 1-18.

[13] Florian Glaser. Pervasive Decentralisation of Digital Infrastructures: A Framework for Blockchain enabled System and Use Case Analysis [C]// Procs of the 50th Hawaii International Conference on System Science. 2017.

[14] Wang Huiqiang, Zou Shichen, Lin Junyu, et al. A trusted service component selection method based on trust path search in distributed virtualization environments: HA-OTPS [J]. Acta Electronica Sinica, 2017, 45(1): 192-200.

[15] Fan Zhiping, Zhao Xuan. Subjective and objective weighting method for weight determination in multi-attribute decision making [J]. Decision Making and Decision Support Systems. 1997, 7(4): 87-91.

Note: Figure translations are in progress. See original paper for figures.