Blockchain-based Content Name Search Mechanism in NDN

Jinshan Shi , Ru Li[+] , Jianghui Zhang

College of Computer Science, Inner Mongolia University, Hohhot 010021, China
csliru@imu.edu.cn

Abstract. Named Data Networking (NDN) is one of the strong competitors of the next generation network architecture, meeting the needs of today's users for the network. Open Data Index Name (ODIN) provides a domain name resolution service for content under Information Centric Networking. ODIN assigns a permanent name identifier to the content and provides a resolution service from the name identifier to the resource access point URL. However, the content in the NDN is identified by a human-readable name, so that the current NDN architecture does not require a name resolution system. Therefore, we designed a blockchain-based Content Name Search Mechanism (BCNSM) by binding content producers and human-readable content names as unique identifiers for content, and then mapping content names with storage locations. Provide users with a content name search and a name-to-content provider resolution service. Then the model was built for the BCNSM using the colored petri net. The model process is verified by the model simulation to meet the expectations, and the state space analysis proves that the BCNSM has no deadlock. Finally, the storage overhead and the cost of the smart contract were evaluated through experiments.

Keywords: Named Data Networking ; Blockchain; Content Name Search; Smart Contract ; Colored Petri Net .

1 Introduction

Users' demand for the network has been changed from computing resource sharing to content acquisition and distribution. Therefore, Named Data Networking (NDN) has been proposed as an implementation of Information Centric Networking (ICN) thinking [1][2]. However, there is a problem in the current NDN: the content consumer does not know whether there is any content in the network when sending the interest packet, and does not know where the target is, but passively searches through the routing process of the NDN architecture, that is, through name-based longest prefix match lookup in each NDN routers [3][4][5].

This problem has been solved by the Domain Name System (DNS) under the TCP/IP network architecture. The DNS provides a domain name directory for the user, and maps the IP address to the domain name so that the user can find the corresponding host according to the directory[6]. However, the lack of DNS is also obvious.

DNS is a distributed database with a central tree structure, so it is vulnerable to distributed denial of service(DDoS) attacks ,and the closer the attack target is to the center, the greater the impact. At the same time, DNS has the disadvantage that information is easily falsified and registration fees are high.

In view of the problems in the design of DNS centralization, two decentralized schemes proposed by researchers at the blockchain are proposed. Ethereum Name Service (ENS) [7] and Open Data Index Name (ODIN) [8]. The decentralized architecture of the blockchain greatly increases the difficulty of DDoS attacks. ENS is a distributed domain name system built on the Ethereum. It maps the addresses and domain names on Ethereum to each other. ENS is a decentralized application built on the ethereum blockchain. The purpose of the ODIN project is to create a decentralized DNS. ODIN is an open system for identifying and exchanging data content indexes on a Bitcoin blockchain architecture. Currently ODIN is used in ICN as a directory for data content .The core of ICN is information, or called content or data, that identifies each unit of information by its name. ODIN gives the content a unique name and is not readable, for example: ppk:305678.568/ISBN2890321345#1.0.

However, the content in the NDN is identified by a human-readable name, and the content is named using a hierarchical structured naming method, so that the current NDN architecture does not need to rename the content. Therefore, we designed a blockchain-based Content Name Search Mechanism (BCNSM) to solve the above problem. BCNSM creates a content name directory and maps the content name to its storage address. In BCNSM, the content name is not resolved to other types of permanent identifiers, but is directly identified with the content name and content producer as the unique identity of the content.

The main contribution of this paper is to design a content name search mechanism that is tightly coupled with the NDN without adding an additional identifier to the content. The content name is mapped to the content provider that stores the content, providing the user with the location of the target content. The irrelevance of the network address that NDN has is oriented to the application layer. For the bottom layer, the content still has a storage location.

2 Background

The content search in the NDN relies on passive search with the router and its forwarding rules, so the user cannot know whether there is a corresponding data packet in the network before sending the interest packet. The same problem exists under TCP/IP. The thin waist of the TCP/IP structure is IP, and the user needs to find the target host conveniently. Therefore, the researchers designed the DNS to provide users with a domain name directory, mapping the IP address and the domain name so that the user can find the corresponding host according to the directory [6].

However, due to the serious security problems caused by its centralized design, some researchers have proposed to use a decentralized blockchain instead of DNS. The open source project ODIN released by the PPkPub development team, the purpose of ODIN is to create a decentralized DNS. ODIN is a completely open and de-

centralized naming identification system based on the blockchain. It is an open system for autonomously naming and exchanging data content indexes in the network environment [8]. At present, ODIN is used in the ICN network environment as a directory of data content in the ICN network, and a blockchain location in which the content name is registered is used as a unique identifier.

Another solution in the blockchain environment is the ENS, a distributed, open naming system based on the Ethereum. ENS maps the user address and account address in Ethereum to a domain name that is simple and easy to remember. ENS is a decentralized application provided by the Ethereum Foundation. Its purpose is to convert a series of complicated and difficult hash addresses in Ethereum into human-readable domain names, which facilitates the use of Ethereum. The ENS consists of three main modules, the registry, the resolver, and the registrar. The registry is the immutable part of the system core, and the resolver is finally implemented by the user. The registrar is a smart contract that assigns a subdomain according to the rules[7]. Hirai Y conducted formal verification for the contracts in ENS [9].

3 Blockchain-based Content Name Search Mechanism

3.1 Application Scenario

The application scenario is an NDN environment, which provides a content name search service for the user, so that the user can know whether there is content that the user wants in the network and the location of the node that can provide the content before sending the interest packet. Did not consider converting the content name to a domain name, because the naming of the content in the NDN itself is readable, not a bunch of meaningless hash characters. nor does it consider creating a unique identifier for the content, because we believe that a content can be uniquely identified through the content producer and content name. Although NDN is content-centric and does not require end-to-end connectivity, we believe that transparently providing users with the location of content caching nodes can make content routing more efficient than blind passive routing.

Table 1. Functional description of the main role

Name	Functional description
Blockchain node	Provide search services to users, store and execute smart contracts
Content producer	The node that generates the content, generates the smart contract corresponding to the content, and publishes the smart contract to blockchain.
Content consumer	The node requesting the content, searching for the node of the content name to the blockchain.
Content cacher	The node that stores the content in the cache and also initiates a transaction to update the information of the smart contract. Generally assumed by the NDN router.
Content provider	Consisting of content producers and content cachers, can provide

content for consumers

There are five types of roles in the network. Among them, the blockchain node consists of three main functional modules: blockchain database, Ethereum Virtual Machine (EVM), and content name search module. The blockchain database is used to store the complete blockchain, the EVM is used to execute the smart contract, and the content name search module is used to query whether there is content named by the name in the network. The content name search module is not a smart contract running on the blockchain, so the search rate is not affected by the blockchain consensus speed. The description of other roles is shown in Table 1.

3.2 Architectural Design

The network architecture of the BCNSM is shown in Figure 1. The content producer publishes the content name in the form of a smart contract to the blockchain before publishing the content to the NDN. Smart contracts automatically maintain a list of content cacher addresses.

After the smart contract is deployed in the blockchain, the content name search module running on the blockchain node collects all the content names stored in the blockchain to construct a mapping of content names to smart contract addresses. When the content consumer wants to search for the content of the name in the network, the content name search module can be used to quickly find out whether the content of the name exists. If the content exists, the content name search module will give the address of the corresponding smart contract. And then get a list of content provider addresses through the smart contract.

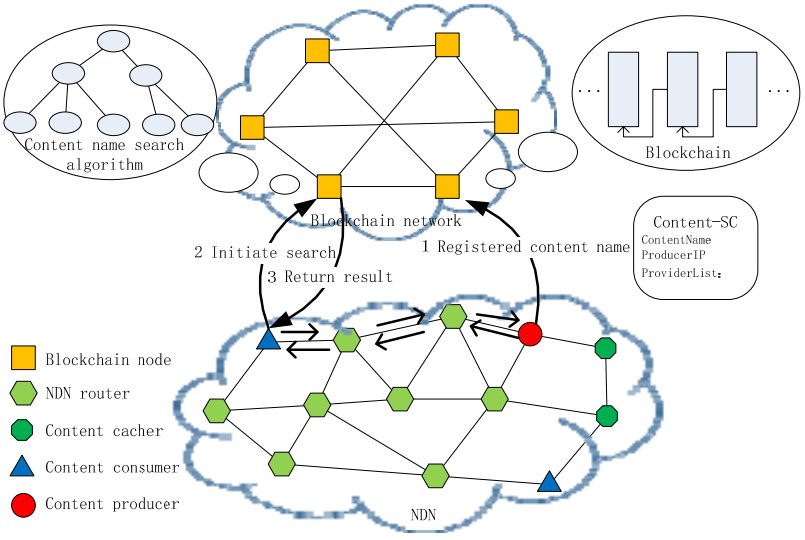


Fig.1. Schematic diagram

In this way, the content consumer can know whether there is a named content in the network and the location of the data source of the content.

The BCNSM is divided into three parts: the publish-subscribe mechanism, the content cache address update mechanism and the content name search algorithm. These three parts are combined by the blockchain.

3.3 Publish-Subscribe Mechanism

The publish-subscribe mechanism provides a registration service for content names and a content name search service. The blockchain acts as a message manager in the publish-subscribe mechanism. It mainly consists of two key parts: the design of the smart contract for registered content name, and the publish and subscription process.

Smart Contract Design. When the content producer generates the content, it also generates a smart contract for the content. The content producer publishes the content name and related information by deploying the smart contract on the blockchain. Choosing each content to generate a smart contract instead of processing all content through a smart contract is mainly due to the following factors: First, the network will generate and disappear a large amount of content at the same time, so the update and maintenance of database is a great burden for single smart contract. Second, it is more secure, content producer can be signed in smart contracts, while preventing malicious users from forging information.

A smart contract can be a collection of code and data. The information and functions stored in the smart contract are as follows:

1. The name of the content, because the content is determined by name in the NDN.
2. Content producer address, because the name is not mandatory globally unique in the NDN, so the producer's address needs to be recorded while the name is recorded to make a distinction.
3. Producer signature, for security reasons, producers will sign smart contracts to ensure that smart contracts are not forged by malicious users.
4. The address list of the content cachers. Associate the content name with all content providers for that content. Automatically update the list by initiates transactions through the content cacher.

Publish Subscription Process. The process of publishing in the publish-subscribe mechanism is to register the content name with the blockchain. The content producer publishes the content name and other related information through the smart contract, and the content cacher publishes the location where the content copy is stored by the cache address update mechanism. The process of subscribing is the content consumer requesting content information from BCNSM. The specific process is as follows:

The First Step: Publish Smart Contract. As shown in Figure 1. The content producer does not directly publish the content, but generates the corresponding smart contract at the same time. Then publish the smart contract in the blockchain. After the smart contract is deployed successfully, the address of the smart contract and the contract interface are added to the content data package, so that the NDN router can send the transaction synchronization information to the smart contract after receiving the content data package.

The Second Step: Subscription Content. The content consumer sends a subscription request message to the blockchain node to request a content provider address of a

content, and the subscription request message is represented as a dual group (content name, source address), wherein the content name is used to identify the subscribed content, and the source address is passed to the smart contract lets it return a message to the content consumer.

The Third Step: Search Content Name. After receiving the subscription request message, the blockchain node reads the content name in the message, and then uses the content name search algorithm to search for the smart contract of the content in the blockchain, and if so, reads the address of the content provider stored in the smart contract, generate a subscription reply message and return it to the content consumer.

The Fourth Step: Return Messages. The subscription reply message is represented as a five-tuple (content name, content producer, requester address, target node address list). The content name and content producer are combined to be used as the unique identifier of the content, because the name in the NDN is not mandatory to be globally unique, so there may be cases where the content names are the same. The requester address is used to locate the target node of this message. The target node address list provides possible destination node addresses for user.

3.4 Content Cache Address Update Mechanism

After the NDN route node caches a certain content, the smart contract address and the contract interface in the content data package are read, and then the transaction that updates the "content cacher address list" in the content smart contract is sent to the blockchain. Add the address of the content cacher. When the node that caches the content deletes the content, a transaction is sent to the blockchain to delete the address of the node in the "content cacher address list" in the smart contract.

3.5 Content Name Search Algorithm

The content name search algorithm uses the blockchain as the data source of the content name, reads the smart contract stored in the blockchain, and collects all the content names to construct a mapping of the content name to the smart contract address. The data structure of the index database is selected using the prefix tree, because the purpose of the search is to find the address of the smart contract based on the content name, so the keyword is the content name, and the hierarchical naming method is used in the NDN, so the search only needs to match the longest common prefix. Provided that an exact match is found, the content exists in the network and then directly returns the corresponding smart contract address to publish-subscribe mechanism.

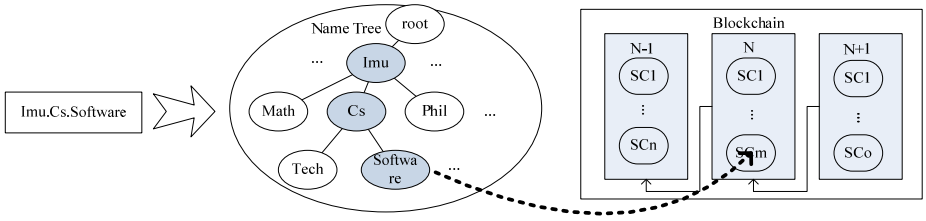


Fig.2. Schematic diagram of content name search algorithm

Figure 2 shows a search schematic of an application using a hierarchical naming method. The content name is the path of the content of the Imu/Cs/Software search in

the name tree. Each node represents a mapping of a smart contract, and finally the target contract is found based on the smart contract address mapped by the Imu/Cs/Software node. The specific algorithm is as follows Algorithm 1:

Algorithm1. Content name search algorithm

Input:	BlockChain(t) ,cn_i // Blockchain at t time, content name
Output:	$ContractAddress_{sc}$ // Smart contract address
1	(1) Initialize the name tree:
2	NameTree(t)=TreeInitialise(BlockChain(t))// Construct a name tree based on the blockchain at time t
3	(2)Periodically update the name tree
4	While *Blockchain consensus* do
5	Scan new blocks and update the name tree
6	End while
7	(3) Content name search:
8	Longest prefix match on the name tree
9	If (*the node exactly matches the content name* cn_i) then
10	Return $ContractAddress_{sc}$
11	Else
12	The content name is not in the blockchain
13	End if

4 Modeling and Analysis

We use formal verification of BCNSM through Colored Petri Net (CPN), which is modeled using hierarchical CPN due to the complexity of the mechanism. The top-level model is shown in Figure 3. It consists of four substitution transitions, nine places, and 16 arcs. The subpage model of the substitution transitions Consumer is shown in Figure 4. The substitution transitions Producer subpage model is shown in Figure 5. The subpage model of the substitution transitions provider is shown in Figure 6. The substitution transitions Blockchain subpage model is shown in Figure 7.

We modeled with CPN Tools and performed simulation and state space analysis. The simulation of the model verified that the BCNSM process was in line with expectations. Then the complete state space of the model is calculated by CPN Tools. The state space analysis report proves that our model has no deadlock.

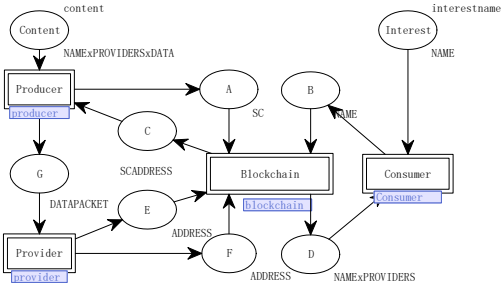


Fig.3. Top-level module

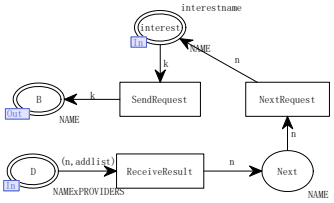


Fig.4. Content consumer module

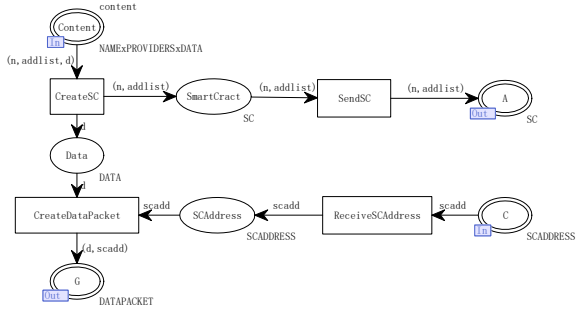


Fig.5. Content producer module

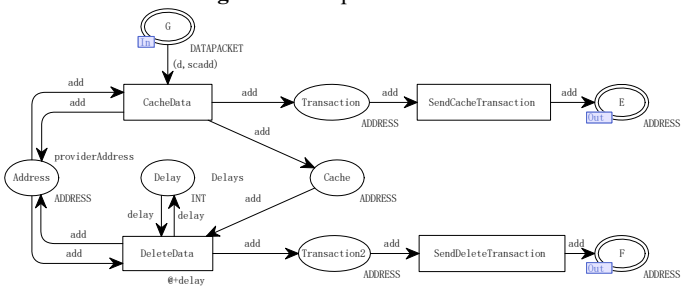


Fig.6. Content cacher module

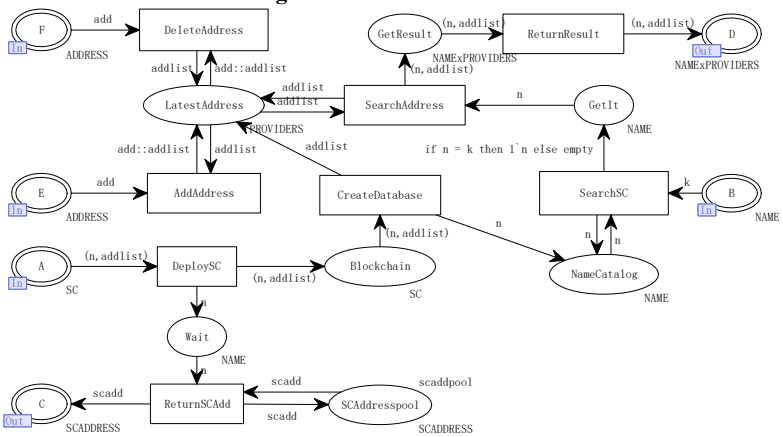


Fig.7. Blockchain module

5 Cost Assessment and Experiment

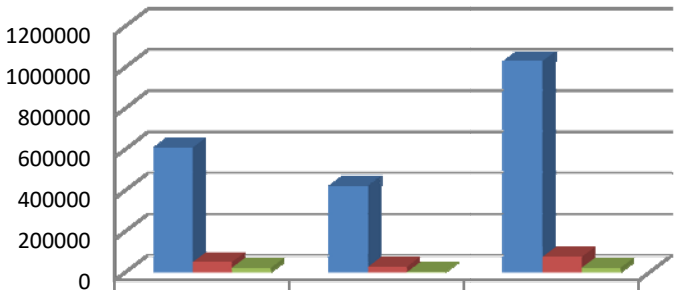
In Ethereum, the cost of calculation and storage is relatively large. Therefore, the average cost of publishing smart contracts and storing data through experimental testing is required. We use the Solidity language to write a smart contract issued by the content producer to register the content name.

Smart contracts are compiled in bytecode and deployed in Ethereum. Ethereum uses Merkle Patricia Trie as a data organization to store the status and data of smart contracts, so we can not directly test the storage overhead of smart contracts, but through the smart contract bytecode is used to represent it. Table 2 shows the storage overhead of the registered content name smart contract. A typical registere smart contract size is 13,033 bytes; the add content cacher transaction and the delete content cacher transaction are both 200 bytes.

Table 2. Storage overhead

Name	Storage overhead(bytes)
Smart Contract	13033
ACCT	200
DCCT	200

Next, the gas overhead of the smart contract is tested experimentally. As shown in Figure 8, the gas overhead is divided into transaction overhead and execution cost. The transaction overhead is the Gas required to initiate the transaction, and the execution cost is the gas required to execute the transaction. The execution cost of deleting a content provider transaction is only 382Gas because the transaction does not need to write additional data to the account store, only changes the state of the stored data in the smart contract, while the other two need to write data to the account store.



	Transaction cost gas	Execution cost gas	Total cost gas
Contract Deployment	607352	421488	1028840
ACCT	49158	26478	75636
DCCT	21846	382	22228

Fig.8. The gas cost of contract deployment and transaction

6 Future Work and Summary

The current work was to map the content name to the content provider that stores the content, providing the user with a content name search service. The next work plan will be to design a routing algorithm based on the content name search. Because the BCNSM content name search service can obtain the storage location of the required content, so the routing algorithm combined with the BCNSM does not need to search for the content data packet through the routing process.

This paper proposes a blockchain-based BCNSM, which provides users with content name search services in the NDN environment, so that users can know whether there is content in the network and the storage location of the content. Then, the BCNSM was modeled by colored Petri nets, and the correctness of the mechanism was proved by formal methods. Finally, the storage and Gas overhead of the smart contract in the mechanism were evaluated through experiments.

Acknowledgmesnts

This work has received funding from the National Natural Science Foundation of China(No. 61862046).

References

1. Zhang L, Afanasyev A, Burke J, et al. : Named Data Networking. ACM SIGCOMM Computer Communication Review 44(3): 66-73(2014).
2. Jacobson V, Smetters D K, Thornton J D, et al.: Networking Named Content. Proceedings of the 5th international conference on Emerging networking experiments and technologies, CoNEXT , pp. 1-12. ACM, Italy(2009) .
3. Wang Y, He K, Dai H, et al.: Scalable Name Llookup in NDN Using Effective Name Component Encoding. 2012 IEEE 32nd International Conference on Distributed Computing Systems, ICDCS ,pp.688-697 . IEEE, Macau, China(2012).
4. Wang Y, Zu Y, Zhang T, et al.: Wire Speed Name Lookup: A GPU-based Approach. USENIX Symposium on Networked Systems Design & Implementation, NSDI, pp.199-212 .ACM ,Lombard(2013).
5. Huang K, Wang Z, Xie G.: Scalable High-speed NDN Name Lookup. Proceedings of the 2018 Symposium on Architectures for Networking and Communications Systems, ANCS, pp.55-65 ACM, Ithaca(2018).
6. Mockapetris P, Dunlap K J.: Development of the Domain Name System. SIGCOMM '88 Symposium Proceedings on Communications Architectures and Protocols, vol.18,pp.123-133, ACM, New York(1988).
7. ENS Homepage, https://docs.ens.domains/en/latest/,last accessed 2018/10/05.
8. ODIN,https://github.com/ppkpub/docs/blob/master/PPT_PPk_ODIN_Introduce_20180322.pdf,last accessed 2018/10/05.
9. Hirai, Y.: Formal verification of Deed contract in Ethereum name service(2016).