Optical Engineering

OpticalEngineering.SPIEDigitalLibrary.org

Exploration of blockchain-enabled decentralized capability-based access control strategy for space situation awareness

Ronghua Xu
Yu Chen
Erik Blasch
Genshe Chen

SPIE.

Ronghua Xu, Yu Chen, Erik Blasch, Genshe Chen, "Exploration of blockchain-enabled decentralized capability-based access control strategy for space situation awareness," *Opt. Eng.* **58**(4), 041609 (2019), doi: 10.1117/1.OE.58.4.041609.

Exploration of blockchain-enabled decentralized capability-based access control strategy for space situation awareness

Ronghua Xu,[a] Yu Chen,[a,*] Erik Blasch,[b] and Genshe Chen[c]
[a]Binghamton University, Department of Electrical and Computer Engineering, Binghamton, New York, United States
[b]US Air Force Research Laboratory, Rome, New York, United States
[c]Intelligent Fusion Technology, Inc., Germantown, Maryland, United States

Abstract. Space situation awareness (SSA) includes tracking of active and inactive resident space objects and assessing the space environment through sensor data collection and processing. To enhance SSA, the dynamic data-driven application systems framework couples online data with offline models to enhance performance by using feedback control, sensor management, and communications reliability. For information management, there is a need for identity authentication and access control (AC) to ensure the integrity of exchanged data as well as to grant authorized entities access right to data and services. Due to decentralization and heterogeneity of SSA systems, it is challenging to build an efficient centralized AC system, which can either be a performance bottleneck or the single point of failure. Inspired by the blockchain and smart contract technology, we introduce blockchain-enabled, decentralized, capability-based access control (BlendCAC), a decentralized authentication, and capability-based AC mechanism to enable effective protection for devices, services, and information in SSA networks. To achieve secure identity authentication, the BlendCAC leverages the blockchain to create virtual trust zones, in which distributed components can identify and update each other in a trustless network environment. A robust identity-based capability token management strategy is proposed, which takes advantage of the smart contract for registration, propagation, and revocation of the access authorization. A proof-of-concept prototype has been implemented on both resources-constrained devices (i.e., Raspberry Pi nodes emulating satellites with sensor observations) and more powerful computing devices (i.e., laptops emulating a ground network) and is tested on a private Ethereum blockchain network. The experimental results demonstrate the feasibility of the BlendCAC scheme to offer a decentralized, scalable, lightweight, and fine-grained AC solution for space system toward SSA. *© 2019 Society of Photo-Optical Instrumentation Engineers (SPIE)* [DOI: 10.1117/1.OE.58.4.041609]

Keywords: blockchain; smart contract; decentralized access control; dynamic data-driven application system; space situation awareness.

Paper 181412SS received Sep. 30, 2018; accepted for publication Jan. 31, 2019; published online Feb. 19, 2019.

1 Introduction

Recent advances in big data (BD) have focused research on the volume, velocity, veracity, variety, and value of dynamic data. These developments enable new opportunities in information management, visualization, machine learning, and information fusion that have potential implications for space situational awareness (SSA).[1] In SSA systems, the space environmental data can be collected and processed to determine object motions and models updates.[2,3] A common example is space tracking using electro-optical sensors.[4] The key aspect for SSA is to track the many resident space objects (RSO), either satellites, debris, or space transportation support.[5] To enable the space environment assessment, many attributes are considered, including communications, space weather, and conjunction analysis as well as using nontradition data.[6] As one of the most critical research areas in SSA, there is a need for network management of the space surveillance network.[7,8] One development for resource management includes the dynamic data-driven applications system (DDDAS) framework whereby measurements are injected into the execution model to enhance system performance. A DDDAS-based system integrates online data with the offline models creating a positive feedback loop, where the model judiciously guides the sensor selection and data collection, from which the sensor measurements improve the accuracy of the model.[6]

DDDAS technology can enhance the space network, where components interact and cooperate with each other to build a BD platform to provide a wide range of services.[9] Thus a huge number of entities, e.g., physical RSOs and virtual services, connect and produce space environment data that can be retrieved by users regardless of their location. To reduce data security risks such as information theft and data alteration, systems dictate that only authenticated and authorized entities are allowed to access the data and use the services provided by the system. The conventional access control (AC) approaches have been widely used in the Internet technology (IT) ecosystem. However, the existing security solutions are not fully adapted to a space network ecosystem due to the constrained resources of space objects and heterogeneity of the platforms. The combination of multiple security technologies and solutions leads to an extraordinary high overload on the system. Furthermore, today's AC solutions often rely on a centralized architecture,

*Address all correspondence to Yu Chen, E-mail: ychen@binghamton.edu

0091-3286/2019/$25.00 © 2019 SPIE

which not only demonstrates enormous scalability issues in an distributed environment composed of large number of nodes but also can be a performance bottleneck or the single point of failure. Consequently, it is necessary to propose new AC solutions for SSA systems.

The blockchain protocol has been recognized as the potential candidate to revolutionize the fundamentals of IT technology because of its many attractive features and characteristics, such as supporting decentralization and anonymity maintenance,[10] as well as a fundamental protocol of Bitcoin,[11] the first digital currency. In this paper, a blockchain-enabled, decentralized, capability-based access control (BlendCAC) scheme is proposed to enhance the security of space applications. BlendCAC provides a decentralized, scalable, fine-grained and lightweight authentication, and AC mechanism to protect devices, services, and information in space networks. To achieve secure identity authentication, a decentralized authentication mechanism is implemented on the blockchain and aims at creating virtual trust zones to allow all distributed entities to identify each other and communicate securely in the trustless network environment. An identity-based capability token management strategy is presented and the federated authorization delegation mechanism is illustrated. A proof-of-concept prototype has been developed and evaluated on a private Ethereum blockchain network, and the experimental results demonstrate the feasibility and effectiveness of the proposed BlendCAC scheme.

The major contributions of this work are as follows:

1. Leveraging the blockchain and smart contract technologies, a decentralized AC solution is proposed to address both the identity authentication and access authorization issues in the distributed space network environment.
2. Using a virtual trust zone, the authentication mechanism ensures that only authenticated entities in same domain could communicate with each other, meanwhile the capability-based AC model provides a scalable, flexible, fine-grained, and lightweight access scheme for space applications.
3. A complete architecture of a blockchain-enabled decentralized AC system is properly designed, which includes identity authentication, capability token management, and access right (AR) validation. The data structures of identity certificate and capability token are explained. The identity authentication algorithms and AR verification process are discussed in detail.
4. A concept-proof prototype based on smart contracts is implemented both on resource-constrained edge devices and more powerful devices and deployed on a local private Ethereum blockchain network, to emulate satellites with collection sensors, a ground-based system network, and satellite communications (SATCOM), respectively. A comprehensive experimental study has been conducted that evaluates the computational and the timeliness performance of using the public blockchain.

The remainder of this paper is organized as follows: Section 2 gives a brief review on the state-of-the-art research in AC and blockchain technology. Section 3 illustrates the details of the proposed BlendCAC system and Sec. 4 explains the implementation of the proof-of-concept prototype. The experimental results and evaluation are presented in Sec. 5. Finally, the summary, current limitations, and ongoing efforts are discussed in Sec. 6.

2 Background Knowledge and Related Work

2.1 *SSA and DDDAS*

Space situation awareness (SSA) is considered as an important frontier because of the increased congestion of satellites vital for strategic decisions, communications, and weather/terrestrial observations.[12] Since the number of satellites in orbit continues to grow exponentially, it is required to ensure that all spacecraft on-orbit work as intended to successfully accomplish their missions. The SSA environment generally consists of two major areas: satellite operations and space weather. The satellite operations are focused on the local perspective to enable continuous operations by understanding the space environment and build models to support satellite health monitoring (SHM). SSA is a systems design, which utilizes data collected from ground and other space assets for RSO tracking, imaging, and collision avoidance.[6,13,14] The key components of SSA include RSO tracking and characterization, SHM and communication, information management, sensing, navigation, and data visualization.[6,15] To address SATCOM challenge that requires cognitive spectrum management and agile waveform adaptation solutions, a game theory-enabled high-level anti-interference strategy was proposed to solve interference in congested space environment.[16] To support space defense analysis and mission trade-off investigations, a satellite orbital testbed for space sensor resource allocation is developed and evaluated for pursuit-evasion game theoretic sensor management.[12]

DDDAS is a conceptual framework that synergistically combines models and data in order to facilitate the analysis and prediction of physical phenomena.[6,9] In an SSA application, DDDAS is a variation of adaptive state estimation that uses computational feedback rather than physical feedback to enhance the information content of measurements. The feedback loops in DDDAS include a data assimilation loop and a sensor reconfiguration loop. The data assimilation loop calculates the physical system simulation using sensor data error to ensure that the trajectory of the simulation more closely follows the trajectory of the physical system. As a fundamental aspect of DDDAS, the sensor reconfiguration loop seeks to manage the physical sensors in order to enhance the information content of the collected data. The simulation based on computational feedback process guides the sensor reconfiguration and the data collection, and in turn, improves the accuracy of the physical system environmental assessment (e.g., space weather and RSO tracking). For sensor management, DDDAS develops runtime software methods for real-time control such as AC.

2.2 *Access Control Mechanism*

An AC mechanism, which specifies admittance to certain resources or services, contributes to the protection, security, and privacy for IT systems. As a fundamental mechanism to enable security in computer systems, AC is the process that decides who is authorized to have what communication rights on which objects with respect to some security models and policies.[17] An effective AC system is designed to satisfy the main security requirements, such as confidentiality, integrity, and availability. In general, a comprehensive AC system addresses three main security issues: authentication, authorization, and accountability.[18] Authentication is the method of validating identity based on registered entity's information. Authorization involves the following phases: defining a security policy (set of rules), selecting an AC model to encapsulate the defined policy, implementing the model, and enforcing the access rules.[18] Accountability employs audit logs to associate subjects with functions.

The AC mechanism incorporates two aspects: the AC model and architecture. The role-based access control (RBAC)[19] model provides a framework that authorizes user's access to resources based on functions. In an RBAC model, permissions are assigned to each agent's role according to organizational functionality definition, and ARs are indirectly granted by associating a user with certain specified role. The functional role acts as an intermediary to bring users and permissions together. The RBAC model supports principles, such as least privilege, partition of administrative functions, and separation of duties, and has widely been used in computer systems.[20] For example, the RBAC model implemented in Internet of things (IoT) networks adopts a Web of things framework,[21,22] which is a service-oriented approach, to enforce AC policies on the smart things via a web service application. However, current RBAC models are not able to address the key issues of implementing RBAC in a highly distributed network environment, such as self-management to handle the explosion of roles in complex and ambiguous space scenarios and autonomy to support physical objects through device-to-device communication.

Compared to the RBAC model, the attribute-based access control (ABAC)[23,24] model defines permissions based on any security relevant characteristics, known as attributes. In the ABAC, AC policies are defined by directly associating predefined attributes with subjects, resources, and conditions, respectively. Given all the attributes assignments, a policy rule, which is a Boolean function, decides whether to grant the subject's access to the resource under specific conditions. The ABAC model eliminates the definition and management of static roles used in the RBAC model. Hence, ABAC also eliminates the need for the administrative tasks for user-to-role assignment and permission-to-role assignment.[23] To address the weaknesses of the RBAC model in a highly distributed network environment, an ABAC extension to the AWS-IoTAC model was proposed to enhance the flexibility of AC in cloud-enabled IoT platform.[25] An efficient elliptic curve cryptography-based authentication and the ABAC policy together was proposed to solve the resource-constrained problem of a perception layer.[26] Although the ABAC is more manageable and scalable than the RBAC by providing finer-grained AC policies that involve multiple subject and object attributes, specifying a consistent definition of the attributes within a single domain or across multiple domains could significantly increase the effort and complexity of policy management as the number of devices grow, and a user-driven and delegation strategies are not supported with the ABAC model. Hence, the attribute-based proposal is not suitable for large-scale distributed network scenarios.

Capability-based access control (CapAC) utilizes the concept of capability that contains rights granted to the entity holding it.[18] The capability is defined as tokens, tickets, or keys that give the possessor permission to access an entity or object in a computer system.[27] The CapAC has been implemented in many large-scale projects, such as IoT@work.[28] However, the direct application of the original concept of CapAC model in a distributed network environment has raised several issues, such as capability propagation and revocation. To tackle these challenges, a secure identity-based capability (SICAP) system[17] was proposed to provide a prospective capability-based AC mechanism in distributed networks. Using an exception list, the SICAP enables monitoring, mediating, and recording capability propagations to enforce security policies as well as achieving rapid revocation capability.[17] By introducing a delegation mechanism for the capability generation and propagation process, a capability-based context-aware access control (CCAAC) model was proposed to enable contextual awareness in federated devices.[29] A federated delegation mechanism enables the CCAAC model has great advantages in terms of addressing scalability and heterogeneity issues in IoT networks. As a user-driven AC model, the CapAC supports machine-to-machine communication and presents great scalability and flexibility to deal with spontaneous and dynamic changes in distributed network environment. However, management of capability propagation becomes difficult without a proper delegation and revocation mechanism.

At the architecture level, the AC solutions are categorized as either the centralized or the decentralized approach. In a centralized AC architecture, the key components, such as authorization policy management and policy decision-making, employ a centralized authority. Outsourcing computational intensive tasks to a back-end cloud or a gateway relieves smart device from the burden of handling AC related functionalities. The approaches in Refs. 25, 26, 29, and 30 are centralized methods. The centralized AC solutions present many advantages, such as easy to adapt existing AC model and simple security policy management. However, enforcing AC on the centralized architecture also suffers from several data management drawbacks, such as single point of failure, performance bottleneck, and privacy issues.

To address the drawbacks in the centralized AC solutions, designing a decentralized AC mechanism supports SSA performance. A distributed capability-based access control (DCapAC) mechanism was proposed that directly deploys AC on resource-constrained devices.[31] The DCapAC allows smart devices to autonomously make decisions on ARs based on an authorization policy, and it has advantages in scalability and interoperability. To address challenges such as scalability, granularity, and dynamicity in AC strategies for SSA systems, a federated capability-based access control (FedCAC) model is proposed to enable an effective AC mechanism to devices, services, and information in large-scale SSA systems.[32] Migrating part of the centralized authorization validation tasks to local devices helps the FedCAC to be lighter and context-awareness enabled. The decentralized AC approach presents advantages, such as supporting user-driven security mechanism and not relying on centralized trust authority. Nonetheless, the decentralized approach also brings many issues, such as requiring a complex AC mechanism and introducing overhead such as in SATCOM.

2.3 *Blockchain and Smart Contract*

The blockchain technology, which was initially introduced by Nakamoto in 2008,[11] has demonstrated its success in decentralization of digital currency and payment like bitcoin. A blockchain is a replicated public database (ledger) that records, stores, and updates all data as chained blocks. It is a public ledger that provides a verifiable, append-only chained data structure of transactions. Enforcing the consensus mechanism on a peer-to-peer network framework, the blockchain is essentially a decentralized architecture that does not rely on a centralized authority. The transactions are approved by a large amount of distributed nodes called miners and recorded in timestamped blocks, where each block is identified by a cryptographic hash and chained to preceding blocks in a chronological order. Blockchain uses a consensus mechanism, which is enforced on miners, to maintain the sanctity of the data recorded on the blocks. Thanks to the trustless proof mechanism running on miners across networks, users can trust the system of the public ledger stored worldwide on many different nodes maintained by "miner-accountants," as opposed to having to establish and maintain trust with a transaction counterparty or a third-party intermediary.[33] Thus blockchain is considered an ideal decentralized architecture to ensure distributed transactions between all participants in a trustless environment.

Emerging from the smart property, a "smart contract" allows users to achieve agreements among parties and supports a variety of flexible transaction types through blockchain networks. Using cryptographic and security mechanisms, smart contract combines protocols with user interfaces to formalize and secure relationships over computer networks.[34] A smart contract includes a collection of predefined instructions and data that have been saved at a specific address of a blockchain as a Merkle hash tree, which is a constructed bottom-to-up binary tree data structure. Since smart contracts are developed as scripts and stored on the blockchain, each smart contract has a unique address that resides on the blockchain. Through exposing public functions or application binary interfaces (ABIs), a smart contract interacts with users to offer a predefined business logic or contract agreement.

Through encapsulating operational logic as a bytecode and performing turing complete computation on distributed miners, a smart contract allows the user to trans-code more complex business models as new types of transactions on a blockchain network. A smart contract provides a promising solution to allow the implementation of more flexible and complex applications far beyond cryptocurrencies, such as data provenance, resource sharing, and dynamic spectrum access. The blockchain and smart contract-enabled security mechanism for applications has been a hot research topic and some efforts have been reported recently, e.g., smart surveillance system,[35,36] social credit system,[37] identity authentication,[38] and AC.[39,40] The research reported by Mital et al.[41] presents the potential role, capabilities and value of blockchain, and smart contract usage within constellation and swarm satellite architectures. The blockchain and smart contract together are promising toward providing a decentralized solution to allow more flexible and fine-grained AC models on space applications.

3 BlendCAC: A Blockchain-Enabled Decentralized CapAC Mechanism

To support sensors and systems for SSA applications, there is a need for resilient, reliable, and robust designs. The space system comprises sensors to monitor the environment, communications to transfer information, and algorithms to process the data such as information fusion.[42] Examples of functions include: evaluating health of the system, tracking of space objects both on-orbit optical observations and ground observations, as well as passing messages with secure communication. Figure 1 demonstrates a sketch of the research scenarios of SSA. There are four geostationary (GEO) satellites (GEO 1, GEO 2, GEO 3, and GEO 4) for the blue orbit and three low earth orbit (LEO 1, LEO 2, and LEO 3) satellites for the yellow orbit. One ground site (GS) is used for ground observations, which provides spectrometers analyzing optical emissions from space object thrusters. Optical observations are performed on satellites using an on-board camera to confirm plume emission and take images of the thruster in operation. While ground observations are conducted on GSs to determine the emission spectra of actively firing Hall thrusters in vacuum chambers, the data transmission among satellites and the GSs is carried out on different SATCOM channels.

As Fig. 1 demonstrates, LEO uses the X-band SATCOM communication channel (yellow lines), and GEO utilizes the K-band SATCOM communication channel (blue lines). The collected data from both on-orbit optical observations and ground observations is shared to provide services for SSA applications. Data integrity and ccess security are significant to ensure data integrity in SSA applications. Thus a flexible and fine-grained AC scheme is necessary to ensure that data sharing among authenticated space objects and cooperative operations are performed by authorized entities. Furthermore, the SATCOM infrastructure includes heterogeneous SATCOM technologies, hybrid space-terrestrial systems, and a decentralized AC architecture.

3.1 *BlendCAC System Architecture for SSA*

Inspired by the smart contract and blockchain technology, a decentralized federated capability-based AC framework for SSA systems, called BlendCAC, is introduced in this paper, and a prototype of the proposal is implemented in a physical network environment to verify the efficiency and effectiveness on a simulated space network scenario. Figure 2 illustrates the proposed BlendCAC system architecture for SSA, which is intended to function in a scenario including multiple isolated space service domains without preestablishing a trust relationship.

In the proposed AC framework shown in Fig. 2, each domain master works as a certificate authority to provide identity authentication services and enforces delegated security policies to manage domain related space object functions and services. Both satellites and GSs could become domain masters. The identification authentication and access authorization policies are transcoded to the smart contracts and deployed across the blockchain network, and identity validation, authorization delegation, and AR verification are developed as service applications running on both domain masters and RSOs in the space network. The operation and communication modes are listed as follows.

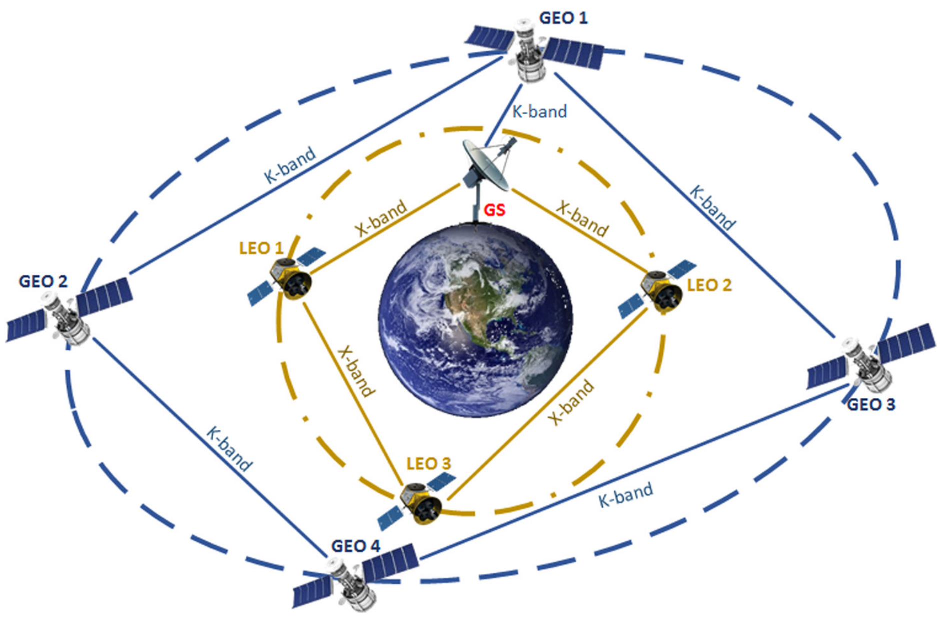


Fig. 1 Research scenarios of SSA.

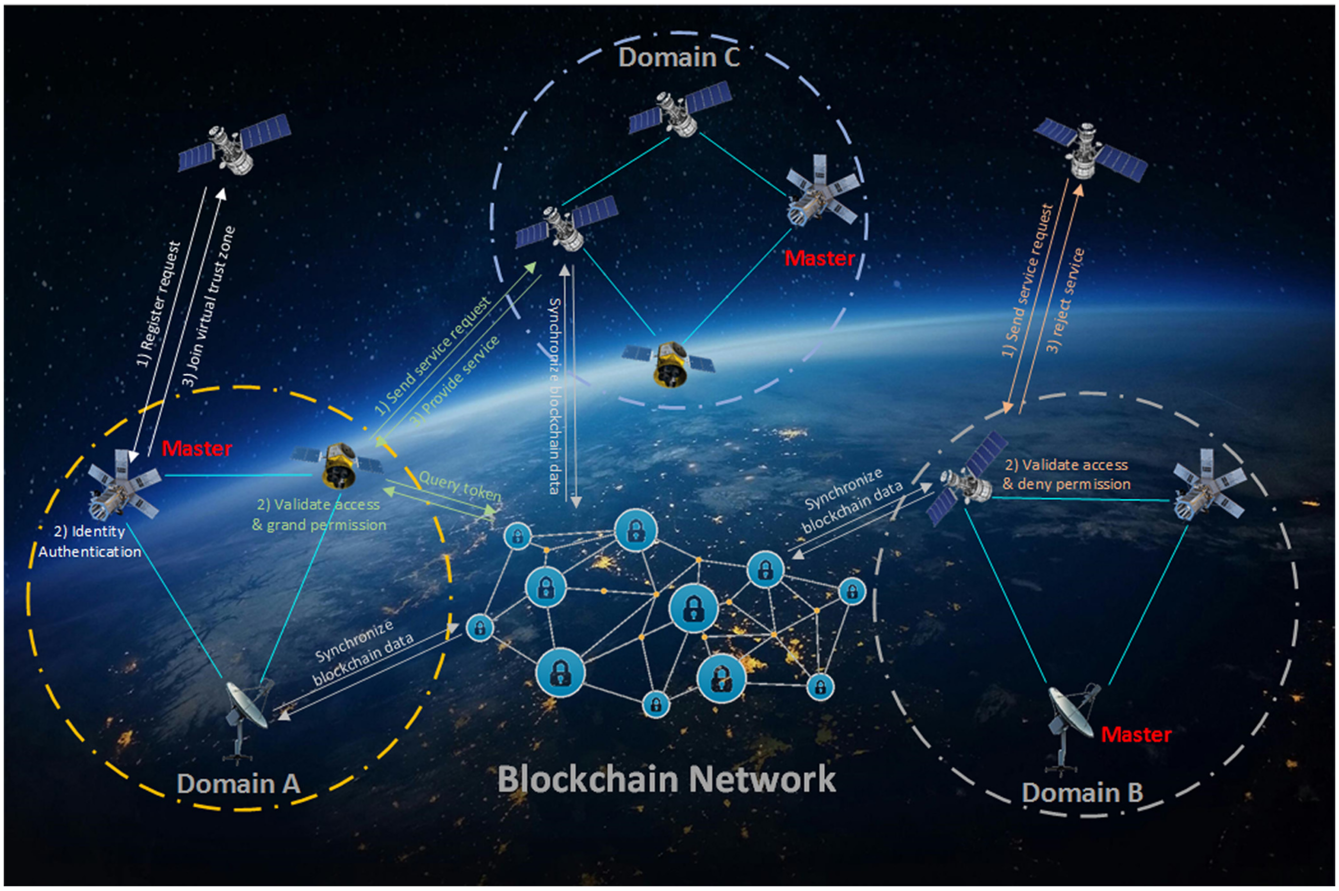


Fig. 2 System architecture of BlendCAC.

3.1.1 *Identification authentication*

All entities must create at least one main account defined by a pair of keys to join the blockchain network. Each account is uniquely indexed by its address that is derived from his/her own public key. Thus account address is ideal for identity authentication in the BlendCAC system given assumption that authentication process is ensured by a blockchain network. In this scenario, a virtual trust zone is created by the domain master, such that each object is allowed to communicate with objects in the same virtual trust zone. Entity registration process uses account address as a virtual identity (VID), which are recorded in the profile database that is deployed on the domain master. A new entity must send authentication request to the domain master in order to join the virtual trust zone. Once the identity information related to requester is verified, the domain master will create the ticket for the registered entity by recording his/her blockchain account address and group ID in the blockchain for authentication process when an service request happens. As a result, the domain masters not only are responsible for identification authentication, but also are able to enforce delegated authorization policies and perform decision-making to directly control the objects or resources in the virtual trust zone instead of depending on third parties.

3.1.2 *Smart contract deployment*

The smart contract, which carries out authentication and manages federated delegation relation and capability tokens, must be developed and deployed on the blockchain network by the policy owner. In the BlendCAC framework, the administrators of the DDDAS, who act as the data and policy owners, could deploy smart contracts encapsulating authentication and authorization algorithms. After smart contracts have been deployed successfully on the blockchain network, they become visible to the entire network owing to the transparency and publicity properties of the blockchain protocol, which means that all participants in the blockchain network can access transactions and smart contracts recorded in the chain data. Thanks to the cryptographic and security mechanisms provided by the blockchain network, smart contracts can secure any algorithmically specifiable protocols and relationships from malicious interference by third parties under a trustless network environment. After synchronizing the blochchain data, all nodes could access all transactions and the recent state of each smart contract by referring local chain data. Each node interacts with the smart contract through the provided contract address and the remote procedure call (RPC) interface.

3.1.3 *Capability authorization*

To successfully access services or resources at service providers, an entity initially sends an AR request to the domain master to get a capability token. Given the entity's ticket, which is the authenticated identity information saved in the blockchain, a decision-making policy module running on the domain master evaluates the access request by enforcing the authorization policies. If the access request is granted, the domain master issues the capability token encoding the AR, and then launches a transaction to update the token data in the smart contract. Once the transaction has been approved and recorded in a new block, the domain master responds to the requester by providing a smart contract address for the querying token data. Otherwise, the AR request is rejected by returning denied acknowledgement.

3.1.4 *Access right validation*

The authorization validation process is performed at the space object who works as the local service provider to receive space service requests from subjects, such as querying satellite sensors of observed spectrometer optical emissions and images of the thruster operations. Through regularly synchronizing the local chain data with the blockchain network, a service provider just simply checks the current state of the contract in the local chain to get a capability token associated with the entity's address. Considering the capability token validation and access authorization process result, if the AR policies and conditional constraints are satisfied, the service provider grants the access request and offers services to the requester. Otherwise, the service request is denied.

To enable a scalable, distributed, and fine-grained AC scheme for space networks, the proposed BlendCAC is focused on three issues: the identification authentication, the identity-based capability management, and the AR authorization.

3.2 *Identification Authentication*

Authentication is the mechanism of validating identity information of entities. The overall purpose of an authentication strategy is to allow multiple entities to communicate with each other in a trustworthy way in a trustless network environment. Inspired by the idea of bubble of trust, all members in a bubble zone can trust each other.[38] The scheme in Fig. 3 illustrates the proposed authentication approach and all the phases of the ecosystem's life cycle. The involved phases in an authentication process is as follows:

- *Initialization*: As shown in Fig. 3(a), the connected space objects that belong to different areas freely communicate with each other. All the space objects in the network could be categorized as either domain masters or service nodes. All the entities should use their main blockchain account addresses as the virtual ID for authentication and AC purposes.
- *Creation of Virtual Zones*: The smart contract for authentication is created by administrators and deployed through transaction, as shown in Fig. 3(b). The master has to communicate with the administrator to create the virtual trust zone for their domain. The master sends a transaction that contains the master's identifier as well as the identifier of the group to be created. The administrator verifies the identity information of the master and sends the transaction to the smart contract to create the virtual trust zone for the master. The blockchain verifies the uniqueness of both of the group ID and the master's virtual ID. If all conditions are satisfied, the smart contract generates a new virtual trust zone with a unique virtual zone ID for the master and returns the smart contract address and authorized RPC for the master to interact with.
- *Join Virtual Trust Zone*: Figure 3(c) demonstrates how nodes are associated with the virtual trust zone. After a virtual trust zone has been created, the nodes in turn, send transactions to the master in order to join their respective virtual trust zones. The domain master checks the applicant's identifier based on the registration policy. If all conditions are satisfied and the applicant has never joined the zone before, the master interacts with the smart contract to add the node to the virtual trust zone. As miners have verified the transaction and generated a new block, the node joins the virtual trust zone successfully with a unique group ID.

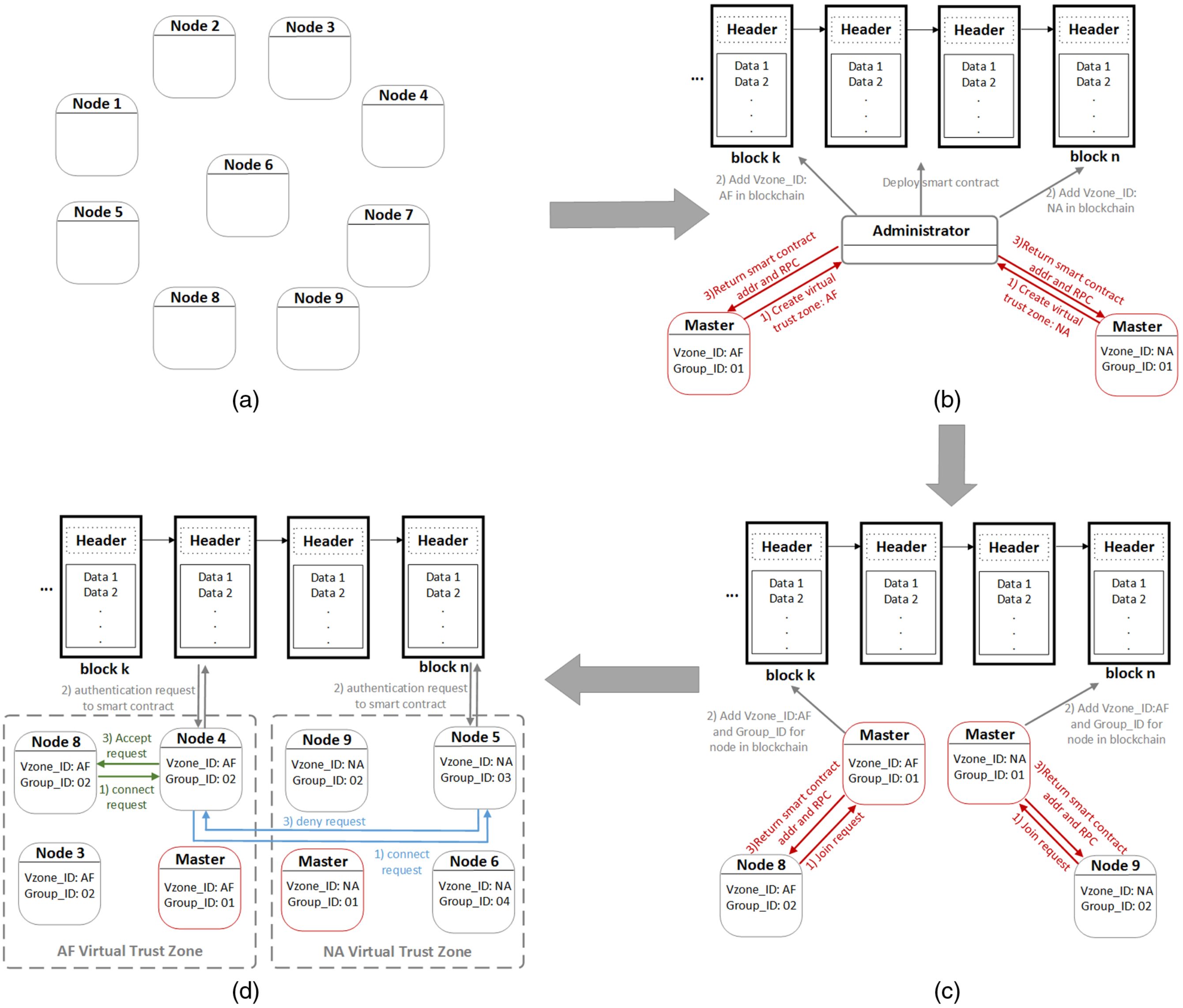


Fig. 3 Virtual trust zone mechanism for authentication: (a) initialization, (b) create virtual trust zone, (c) join virtual trust zone, and (d) identity authentication.

- *Identity Authentication*: As all nodes' group information are recored in the blockchain and the identifier verification process is ensured by the smart contract, the identity authentication is no longer relying on a third-party centralized trust authority. In Fig. 3(d), node 8 could successfully talk to node 4 owning to the fact that they have the same Vzone_ID. However, the node 5 denies the connect request from node 4 because they do not share the same Vzone_ID, which means that they actually do not belong to the same virtual trust zone.

Through clustering the nodes into different virtual trust zones, the application domains become isolated. Only those authenticated entities are allowed to communicate with group members of their zones, whereas any entities outside of the zones are considered as suspicious and prevented from being connected to any group members in the zones.

3.3 *Capability Token Structure*

In the access authorization scenarios, the entities are categorized as subjects or objects. Subjects are defined as entities who request services from the service providers, whereas objects are referred to as entities who offer the resources or services. Entities could be either human operators or RSOs like satellites. Since the identity registration and authentication processes are mainly performed on domain masters, a profile database that is used for recording the profile of each group member is constructed and maintained by the domain master. In the profile databased, all registered entities are associated with a globally unique VID, which is used as the prime key for searching entities' profile information. As each entity has at least one main account indexed by its address in the blockchain network, the blockchain account address is used to represent the VID for profiling register entities.

In general, the capability specifies which subject can access resources at a target object by associating subject, object, actions, and condition constraints. The identity-based capability structure is defined as a hash table, which is represented as follows:

$$\text{ICap} = f(\text{VID}_S) \rightarrow (\text{VID}_O, \text{OP}, C), \tag{1}$$

where the parameters are: f is a one-way hash mapping function to establish relation between a subject and authorized internal capability set; VID_S is the virtual ID of a subject that requests an access to a service or resource; VID_O is the virtual ID of an object that provides a service or resource; OP is a set of authorized operations, e.g., read, write, and execute; and C is a set of context awareness information, such as time and location.

In the AC system, the elements in OP set can be represented as actions, such as $\{\text{read}\}$, $\{\text{write}\}$, $\{\text{read; write}\}$, or $\{\text{NULL}\}$. If $\text{OP} = \{\text{NULL}\}$, any operation conducted on the resource is not allowed. C is defined as a context constraints set, such as $C = \{C1, C2\}$ or $C = \{\text{NULL}\}$. If $C = \{\text{NULL}\}$, no context constraint is considered in the AR validation process.

3.4 *Capability-Based Access Right Authorization*

The capability token structure and the related operations are transcoded to a smart contract that is deployed on the blockchain network, and the AR authorization is implemented as a policy-based decision-making service running on the domain master. As shown in Fig. 4, a comprehensive capability-based AR authorization procedure consists of four steps: capability generation, AR validation, capability delegation, and revocation.

1. *Capability Generation*: As one type of meta data to represent the AR, the capability ICap could be generated by associating a VID with an AR, thus the ICap has the identified property to prevent forgery. After receiving an access request from a user, the domain master generates a capability token based on the AR authorization policy, and launches transactions to save a new token data to a smart contract. A large number of ICaps are grouped into the capability pools on the smart contract, which could be proofed and synchronized among the nodes across the blockchain network.
2. *Access Right Validation*: After receiving the service request from a subject, the service provider first fetches the capability token from the smart contract using the subject's address, then makes decisions on whether or not to grant access to the service according to the local AC policy. Implementing AR validation at the local service provider allows smart objects to be involved in the AC decision-making task, which is suitable to offer a flexible and fine-grained AC service in distributed space networks.
3. *Capability Revocation*: The capability revocation considers two scenarios: partial AR revocation and ICap revocation. In the system design, only the administrator or domain masters are allowed to perform revocation operation on the capability tokenized smart contract. In the partial AR revocation process, the authorized entities could remove part of the entries from AR to revoke the selected ARs. In case of ICap revocation, through directly clearing the AR in ICap, the whole capability token becomes unavailable to all associated entities.

4 Prototype Design

A proof-of-concept prototype system has been implemented in a real private Ethereum blockchain network environment extending an SSA study using a cloud architecture.[43] As the second biggest ledger in the world, Ethereum is robust against attacks and data falsifications. In addition, transactions in Ethereum adopt the elliptic curves cryptography

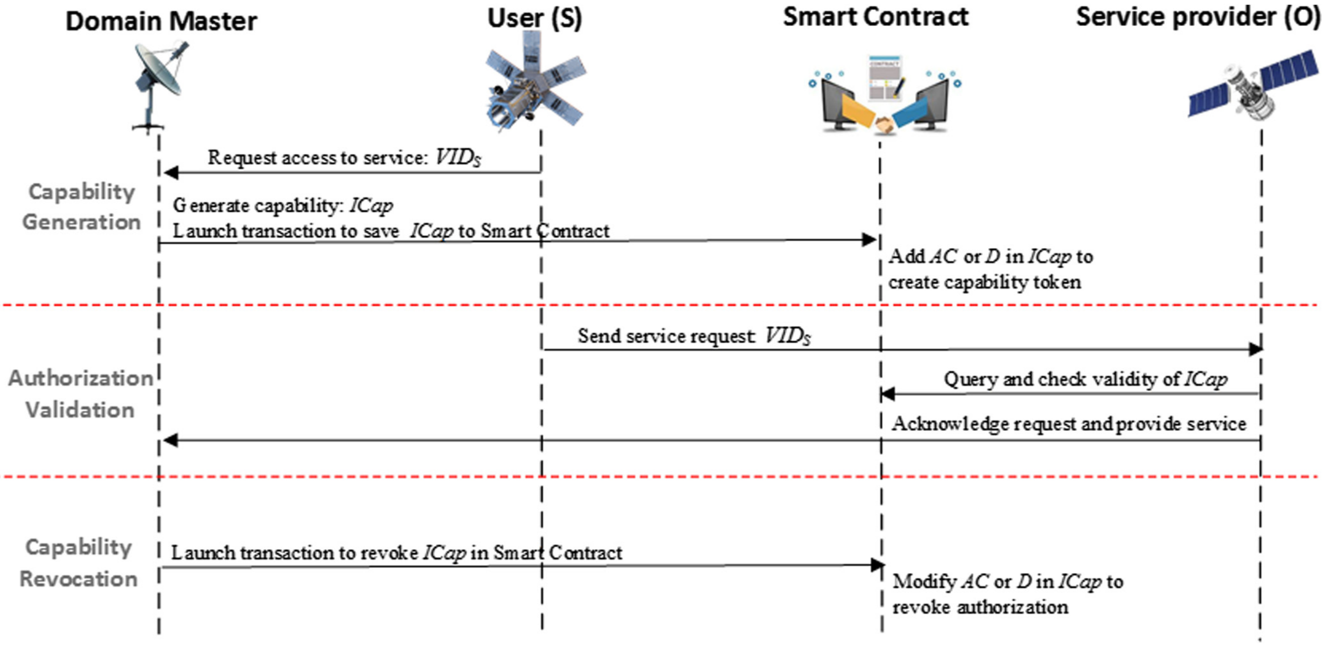

Domain Master
User (S)
Smart Contract
Service provider (O)
Capability Generation
Request access to service: VID_S
Generate capability: $ICap$
Launch transaction to save $ICap$ to Smart Contract
Add AC or D in $ICap$ to create capability token
Authorization Validation
Send service request VID_S
Query and check validity of $ICap$
Acknowledge request and provide service
Capability Revocation
Launch transaction to revoke $ICap$ in Smart Contract
Modify AC or D in $ICap$ to revoke authorization

Fig. 4 Flowchart of the capability-based AR authorization.

as the signature scheme, which represents robust and lightweight properties for constrained devices. Furthermore, compared with other open blockchain platforms, such as Bitcoin and Hyperledger, Ethereum has a more matured ecosystem and is designed to be more adaptable and flexible for the development of a smart contract and business logic.[44]

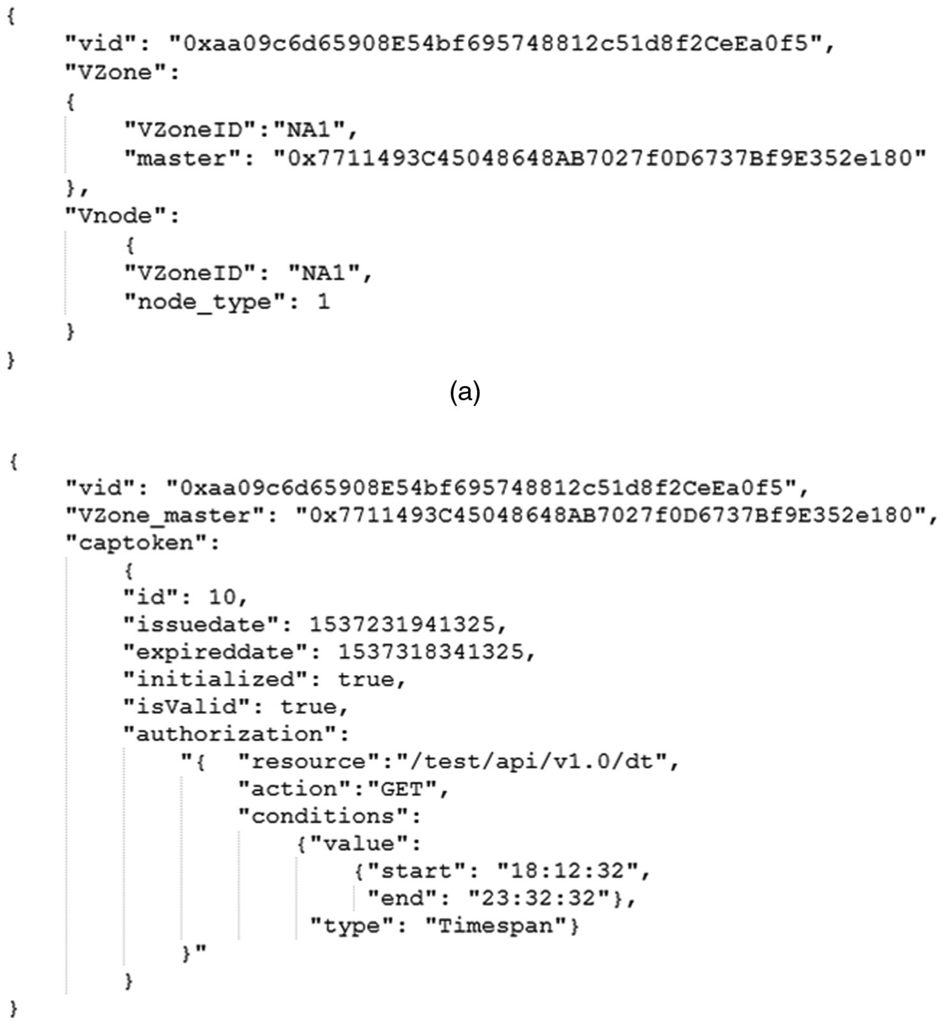
4.1 *Authentication Certificate and Capability Token Structure*

The proposed identity authentication and AC models have been transcoded to smart contracts using solidity,[45] which is a contract-oriented, high-level language for implementing smart contracts. With Truffle,[46] which is a world class development environment, testing framework, and asset pipeline for Ethereum, contract source codes are compiled to Ethereum virtual machine bytecode and migrated to the Ethereum blockchain network.

To implement a BlendCAC system on RSOs without introducing significant overhead over SATCOM and computation, delegation certificate and capability token data structure is represented in JSON[47] format. Compared to XML-based language for AC, such as XACML and SAML, JSON is lightweight and suitable for resource constrained platforms.

Figure 5(a) demonstrates an example of the authentication certificate, and the data fields in the data structure are described as follows:

- vid: a 20-byte value to represent address of the certificate owner in the blockchain network;
- VZone: virtual trust zone data that has been created by the master, including
  - VZoneID: a string that is used for a virtual trust zone data uniquely represented; and
  - master: a 20-byte value used to represent the blockchain account address of the entity who created the virtual trust zone.
- Vnode: a set of identity information that has been associated to the node for authentication, including
  - VZoneID: a string that is used to record the unique ID of a virtual trust zone, in which the entity has participated; and



```
{
    "vid": "0xaa09c6d65908E54bf695748812c51d8f2CeEa0f5",
    "VZone":
    {
        "VZoneID":"NA1",
        "master": "0x7711493C45048648AB7027f0D6737Bf9E352e180"
    },
    "Vnode":
        {
        "VZoneID": "NA1",
        "node_type": 1
    }
}
```

(a)

```
{
    "vid": "0xaa09c6d65908E54bf695748812c51d8f2CeEa0f5",
    "VZone_master": "0x7711493C45048648AB7027f0D6737Bf9E352e180",
    "captoken":
        {
        "id": 10,
        "issuedate": 1537231941325,
        "expireddate": 1537318341325,
        "initialized": true,
        "isValid": true,
        "authorization":
            "{  "resource":"/test/api/v1.0/dt",
                "action":"GET",
                "conditions":
                    {"value":
                        {"start": "18:12:32",
                         "end": "23:32:32"},
                     "type": "Timespan"}
            }"
        }
}
```

(b)

Fig. 5 Token data structure used in BlendCAC: (a) authentication certificate and (b) capability token.

  - node_type: an integer to specify the role that the entity has been assigned in the virtual trust zone, either the master or the follower.

Figure 5(b) presents an example of the capability token data used in the AC mechanism. A brief description of each field is provided as follows:

- vid: a 20-byte value to represent address of the capability token owner in the blockchain network;
- VZone_master: a 20-byte value used to record address of the master in the virtual trust zone that entity has joined;
- id: the autoincremented prime key to identify a capability token;
- initialized: a bool flag used for checking token initialized status;
- isValid: a boolean flag signifying the enabled status to show whether or not the token is valid;
- issuedate: for identifying the date and time when the token was issued;
- expireddate: the date and time when a token expires;
- authorization: a set of AR rules that the issuer has granted to the subject, including
  - action: to identify a specific granted operation over the resource;
  - resource: to grant the operation in the service provider; in this case, the resource is defined as the granted REST-ful API; and
  - conditions: a set of conditions that must be fulfilled locally on the service provider to grant the corresponding operation.

After a smart contract has been successfully deployed on the blockchain network, all nodes in the network could interact with the smart contract using address of the contract and the ABI definition, which describes the available functions of the contract.

4.2 *Identity Authentication Policy Service*

The identity authentication mechanism based on the virtual trust zone is implemented as a set of service interface functions, which are executed by the smart contract to enforce the authentication policy. Algorithm 1 illustrates the virtual trust zone construction process. The function createVZone() receives the inputs of the string VZoneID and returns the VZone creation result. The process first checks the entity address so that the supervisor or the valid masters are allowed to create a new VZone. The existing virtual trust zones could be deleted either by the supervisor or masters who have created the VZones, and the revocation process is illustrated in Algorithm 2.

After the virtual trust zones have been constructed by masters, other nodes could send registration requests to the masters for joining the virtual trust zone. Algorithm 3 describes the process of how a node becomes a member of a VZone. Once the Vnode of the applicant has been recorded in the blockchain, he/she could communicate with other nodes in the same VZone. The associated trust

Algorithm 1 Create virtual trust zone

```
Require: VZoneID
entityAddr = msg.sender
if (entityAddr == supervisor) or (isValidMaster(entityAddr) ==
true) then
    if if(Vzone[VZoneID].master == address(0)) then
        Vzone[VZoneID].uid += 1
        Vzone[VZoneID].master = entityAddr
        Vnode[entityAddr].VZoneID = VzoneID
        Vnode[entityAddr].node_type = 1
        returnTrue
    else
        returnFalse
    end if
else
    returnFalse
end if
```

Algorithm 2 Revoke virtual trust zone

```
Require: VZoneID
entityAddr = msg.sender
if entityAddr == supervisor then
    if (entityAddr == supervisor) or ((isValidMaster(entityAddr) ==
    true) and Vzone[VZoneID].master == entityAddr) then
        curr_master = Vzone[VzoneID].master
        Vzone[VZoneID].uid += 1
        Vzone[VZoneID].master = address(0)
        Vnode[entityAddr].VZoneID =“ ”
        Vnode[entityAddr].node_type = 0
        returnTrue
    else
        returnFalse
    end if
else
    returnFalse
end if
```

Algorithm 3 Join virtual trust zone

```
Require: VZoneID
Require: nodeAddr
entityAddr = msg.sender
if (entityAddr == supervisor) or (entityAddr == Vzone[VZoneID].master) then
    if Vnode[nodeAddr].nodetype == 0 then
        Vnode[nodeAddr].VZoneID = VzoneID
        Vnode[nodeAddr].node_type = 2
        returnTrue
    else
        returnFalse
    end if
else
    returnFalse
end if
```

relationship between a node and the VZone could be revoked either through leave request sent by the node, or directly be removed by the supervisor and the master of the VZone. Algorithm 4 explains the operation to remove a node from a virtual trust zone.

The identity verification is enforced based on service providing scenarios. As a service provider received a service request from an entity, he/she just queries entity's Vnode data in the blockchain and verifies the identification by simply checking whether or not the entity has the same VZoneID. The requests from non-VZone entities are directly rejected.

Algorithm 4 Leave virtual trust zone

```
Require: VZoneID
Require: nodeAddr
entityAddr = msg.sender
if (entityAddr == supervisor) or (entityAddr == Vzone[VZoneID].master) then
    if Vnode[nodeAddr].nodetype == 2 then
        Vnode[entityAddr].VZoneID = " "
        Vnode[entityAddr].node_type = 0
        returnTrue
    else
        returnFalse
    end if
else
    returnFalse
end if
```

4.3 *Access Authorization Service*

The access authorization and validation policy is enforced as a web service application, which emulates the space service scenarios among satellites and GSs. The test application is developed on the Flask framework[48] using Python. The Flask is a microframework for Python based on Werkzeug, Jinja 2, and good intentions. The lightweight and extensible micro-architectures make the Flask a preferable web solution on resource constrained devices.

Web service application in the BlendCAC system consists of two parts: client and server. The client performs operations on resource by sending data request to the server, whereas the server provides REST-ful API for the client to obtain data or perform operations on the resource at the server side. A capability-based AC scheme is enforced at the server side by performing AR validation on the service provider. The AR validation process is launched after a request containing the client's identity is received by the server. Figure 6 shows a block diagram with the steps to process an authorization request.

1. *Check cached token data*: After receiving a service request from a user, the service provider first checks whether or not the token data associated with user's address exists in the local database. If it is failed in searching the token data, the service provider can fetch the token data from the smart contract through calling an exposed contract method and save token data to the local database. Otherwise, the token data are directly reloaded from the local token database for further validation. The service provider regularly synchronizes the local database with the smart contract to ensure the token data consistency.
2. *Verify token status*: As a capability token has been converted to the JSON data, the first step of token validation is checking the current capability token status, such as initialized, isValid, issuedate, and expireddate. If any status of a token is not valid, the authorization process stops and sends a deny access request acknowledgement back to the subject.
3. *Check whether access is granted or not*: The service provider will go through all access rules in the AR set to guarantee that the request operation is permitted. The process checks whether or not the REST-ful method used by the requester matches the authorized action of current access rules and the value of the resource field is the same as the request-URI option used by the requester. If the current access rule verification failed, the process skips to the next access rule for evaluation. If none of the access rules could successfully pass the verification, the authorization validation process stops and denies the access request.
4. *Verify the conditions*: Even though the action on a target resource is permitted after the access validation, the context-awareness constraints are necessary to

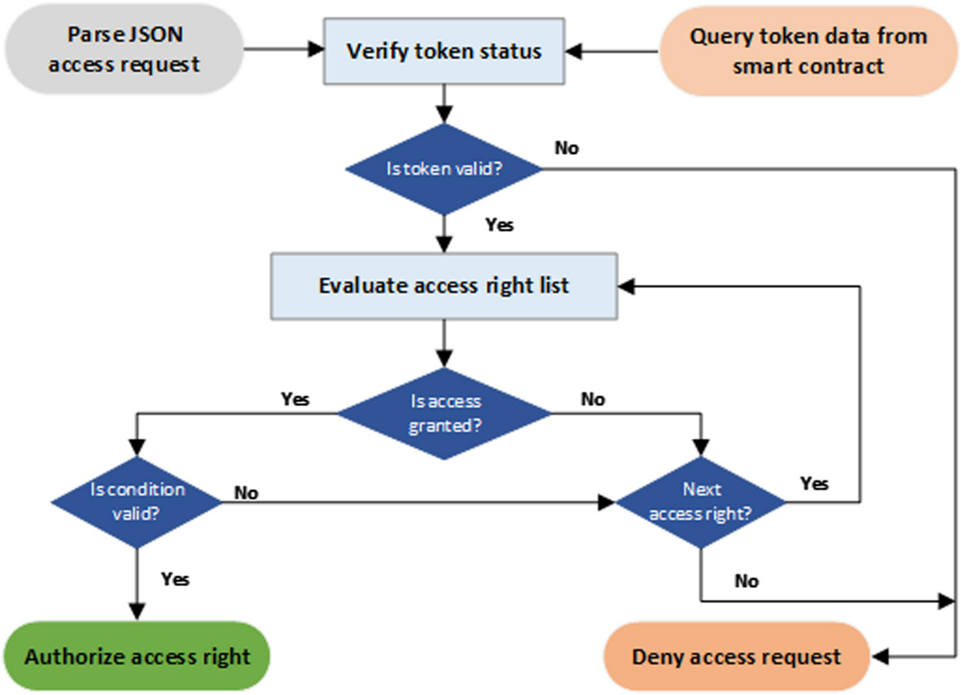


Fig. 6 Access authorization process of BlendCAC.

be evaluated on the local device by verifying whether or not the specified conditions in the token are satisfied. The condition verification process goes through all constraints in the condition set to find the matched ones. If no condition is fulfilled in the given local environment, the AR validation process stops and denies access request.

5 Experimental Study

In order to evaluate the performance and the overhead of our AC scheme, the identity authentication and access authorization are transcoded to separate smart contracts and enforced on the experimental web service system. The profiles and policy rules management are developed using an embedded structured query language (SQL) database engine, called SQLite.[49] The lower memory and computation cost make the SQLite an ideal database solution to resource constrained system like Raspberry Pi. All documents and source code are available on BC_DDDAS project repository on GitHub.[50]

5.1 *Testbed Setup*

The mining task is performed on a system with stronger computing power, such as a laptop or a desktop. Two miners are deployed on a laptop and other four miners are distributed on four desktops. Table 1 describes configuration of nodes used in the experiments. In our system, the laptop acts as a cloud computing server, whereas all desktops work as fog computing nodes to take role of domain masters. Each miner uses two CPU cores for mining. The edge computing services are deployed on two Raspberry Pi 3 Model B. Since the Raspberry Pi is not powerful enough to carry out mining task, so all Raspberry Pi devices function as nodes to participate in the private blockchain network without mining. All devices use Go-Ethereum[51] as the client application to work on the blockchain network.

5.2 *Experimental Results*

To verify the effectiveness of the BlendCAC approach against unauthorized access requests, a service access experiment is carried out on a simulated SATCOM network. In the

Table 1 Configuration of experimental nodes.

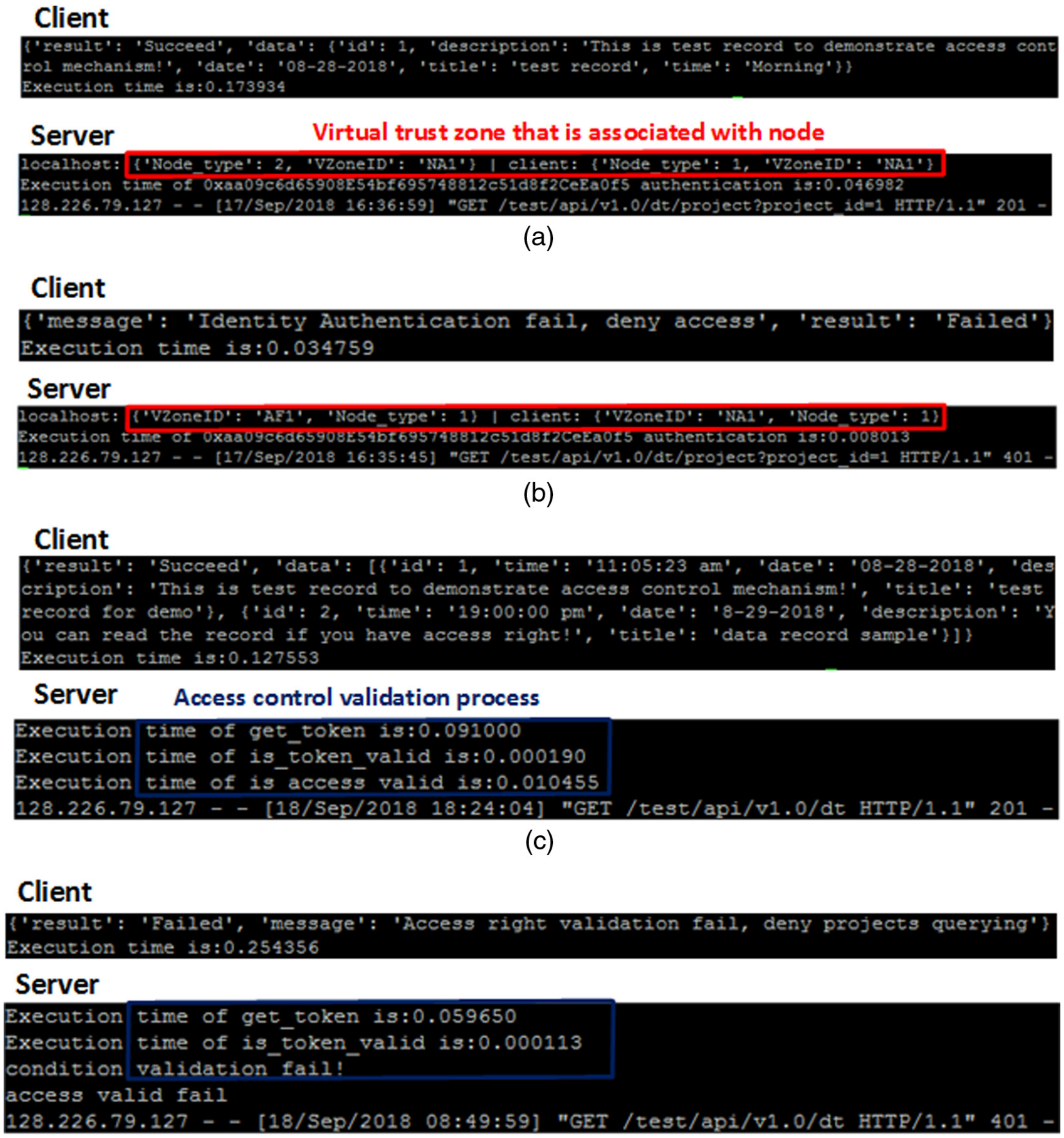
Node device	Lenovo P50	Dell Optiplex 760	Raspberry Pi 3 model B
CPU	2.3 GHz Intel Core i7 (8 cores)	3 GHz Intel Core™ (2 cores)	Quad-core ARM Cortex A53, 1.2 GHz
Memory	16 GB DDR3	4 GB DDR3	1 GB SDRAM
Storage	250G SSD + 500G HHD	250G HHD	32 GB (microSD card)
Operation system	Ubuntu 16.04	Ubuntu 16.04	Raspbian GNU/Linux 8 (jessie)

simulation environment, the edge devices represent satellites and the server is the ground communication receiving data, such as space imagery. In the test scenario, one Raspberry Pi 3 device works as the client and another works as the service provider. The identity authentication results are shown in Figs. 7(a) and 7(b). Figure 7(a) demonstrates that the node "0xaa09c6d65908e54bf695748812c51d8f2ceea0f5" successfully passed the authentication process executed on the server with the same VZoneID. Figure 7(b) shows a failed authentication scenario caused by communicating with entity who belongs to a different virtual trust zone.

Given the access authorization process shown in Figs. 7(c) and 7(d), when any of the steps in the authorization procedure fails, the running process immediately aborts instead of continuing to step through all the authorization stages. As shown in Fig. 7(d), the server stopped the authorization process due to the failure in verifying the granted actions or the conditional constraints that are specified in the AR list. Consequently, the client node received a deny access notification from the server and cannot read the requested data. In contrast, Fig. 7(c) presents a successful imagery data request example, in which the whole authorization process is accomplished at the server side without any error. Finally, the client successfully retrieves the imagery data from the service provider.

5.3 *Performance Evaluation*

In the simulation environment, two Raspberry Pi devices emulate satellites to provide on-orbit observations while a desktop computer serves as a GS to perform ground observations. One Raspberry Pi device is adopted to play the role of the client while another Raspberry Pi and desktop computer are service providers, and an Ethernet is used to simulate SATCOM communication channel. To measure the general cost incurred by the proposed BlendCAC scheme



```
{'result': 'Succeed', 'data': {'id': 1, 'description': 'This is test record to demonstrate access cont
rol mechanism!', 'date': '08-28-2018', 'title': 'test record', 'time': 'Morning'}}
Execution time is:0.173934
```

```
localhost: {'Node_type': 2, 'VZoneID': 'NA1'} | client: {'Node_type': 1, 'VZoneID': 'NA1'}
Execution time of 0xaa09c6d65908E54bf695748812c51d8f2CeEa0f5 authentication is:0.046982
128.226.79.127 - - [17/Sep/2018 16:36:59] "GET /test/api/v1.0/dt/project?project_id=1 HTTP/1.1" 201 -
```

(a)

```
{'message': 'Identity Authentication fail, deny access', 'result': 'Failed'}
Execution time is:0.034759
```

```
localhost: {'VZoneID': 'AF1', 'Node_type': 1} | client: {'VZoneID': 'NA1', 'Node_type': 1}
Execution time of 0xaa09c6d65908E54bf695748812c51d8f2CeEa0f5 authentication is:0.008013
128.226.79.127 - - [17/Sep/2018 16:35:45] "GET /test/api/v1.0/dt/project?project_id=1 HTTP/1.1" 401 -
```

(b)

```
{'result': 'Succeed', 'data': [{'id': 1, 'time': '11:05:23 am', 'date': '08-28-2018', 'des
cription': 'This is test record to demonstrate access control mechanism!', 'title': 'test
record for demo'}, {'id': 2, 'time': '19:00:00 pm', 'date': '8-29-2018', 'description': 'Y
ou can read the record if you have access right!', 'title': 'data record sample'}]}
Execution time is:0.127553
```

```
Execution time of get_token is:0.091000
Execution time of is_token_valid is:0.000190
Execution time of is_access_valid is:0.010455
128.226.79.127 - - [18/Sep/2018 18:24:04] "GET /test/api/v1.0/dt HTTP/1.1" 201 -
```

(c)

```
{'result': 'Failed', 'message': 'Access right validation fail, deny projects querying'}
Execution time is:0.254356
```

```
Execution time of get_token is:0.059650
Execution time of is_token_valid is:0.000113
condition validation fail!
access valid fail
128.226.79.127 - - [18/Sep/2018 08:49:59] "GET /test/api/v1.0/dt HTTP/1.1" 401 -
```

(d)

Fig. 7 Examples of experimental results of the BlendCAC system: (a) identity authentication succeed, (b) identity authentication failed, (c) access authorization succeed, and (d) access authorization failed.

both on the space (e.g., satellite) devices' processing time and the network communication delay, 100 test runs have been conducted based on the proposed test scenario, where the client sends a data query request to the server for an access permission. This test scenario is based on an assumption that the subject has joined the virtual trust zone and has been assigned a valid capability token when it performs the action. Therefore, all steps of identity authentication and authorization validation must be processed on the server side so that the maximum latency value is computed.

5.3.1 *Computational overhead*

According to the results shown in Fig. 8, the average total delay time caused by the BlendCAC operation of retrieving data from the client to server is 250 ms on satellites. Since the GSs have much more computation capacity than the satellites, the execution time of the whole data querying process on the ground communication is about 60 ms. The total delay includes the round trip time (RTT), time for querying capability data from the smart contract, time for parsing JSON data from the request, and time for identity authentication and the AR validation. The token processing task is mainly responsible for fetching token data from the smart contract and introduces the highest workload among the AC operation stages. As the most computing intensive stage, the execution time of token processing is about 60 ms on the satellite, and the same operation on the ground communication only needs 10 ms.

The entire AC process is divided into two steps, identity authentication and capability token verification. Since the identity authentication process needs to interact with smart contract twice for querying VZone and Vnode data separately, identity authentication processing time is 152 ms, which is almost twice as much as that of execution on capability-based AC stage: 63 ms. The execution time of the AC process is about 214.5 ms ($152 + 62.5$) on satellites, which accounts for almost 86% of the entire data service processing time.

5.3.2 *Communication overhead*

Due to the high overhead introduced by querying token data from the smart contract in token processing stage, a token data caching solution is introduced in the BlendCAC system to reduce the network latency. When the client sends a service request to the server, the service provider extracts cached token data from the local storage to valid authorization. The service providers regularly update cached token data by checking smart contract status. The token synchronization time is consistent with the block generation time, which is about 15 s in the Ethereum blockchain network. Simulating a regular service request allows us to measure how long it takes for the client to send a request and retrieve the data from the server.

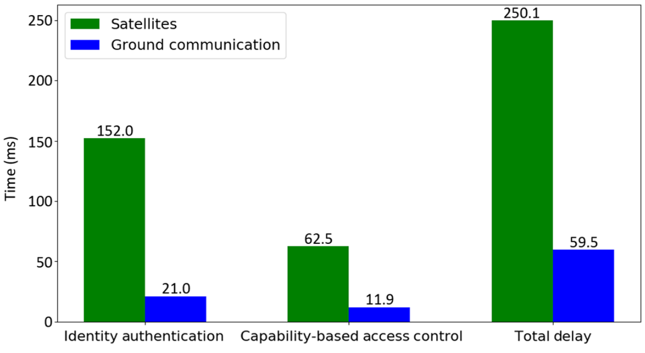


Fig. 8 Computation time for each stage in BlendCAC.

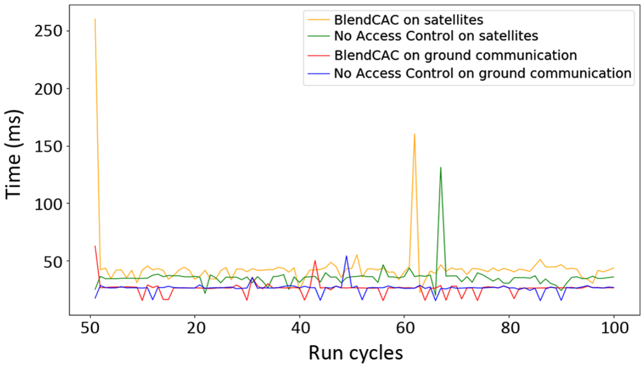


Fig. 9 SATCOM latency of BlendCAC.

Figure 9 shows the overall SATCOM communication latency incurred and compares the execution time of the BlendCAC and a benchmark without any AC enforcement (BwoAC). At the beginning, a long SATCOM delay is observed in the first service request scenario, in which the service provider communicated with the smart contract and cached the token data. However, by processing the local cached token data for authorization validation, the SATCOM latency decreases quickly and becomes stable during the subsequent service requests. At the satellite device, the BwoAC takes an average of 35 ms for fetching requested data versus the BlendCAC that has an average of 44 ms. The results demonstrate that the proposed BlendCAC scheme only introduces about 9 ms extra latency. The overhead in terms of delay by the AC enforcement is even more trivial on ground communication nodes. The average time for querying data with AC is about 26 ms, which is almost the same as the average time of querying data without AC. However, the benefit of the secure BlendCAC outweighs the small latency cost.

5.3.3 *Processing overhead*

The delegation certificate and capability token could be validated only if the related transactions to the smart contract have been approved by miners and recorded to new blocks. The transaction rate is proportional to the block generation time, which refers to the time consumed by the miners to verify new blocks. Table 2 demonstrates the impacts of the number of miners on the blockchain network as well as estimated financial cost for transactions. In each scenario, 60 blocks are appended to the blockchain and the average block generation time is calculated. In initial, only two miners run consensus algorithm. As more miners perform the proof of work, the block generation time drops down and finally becomes stable. As shown in Table 2, the optimal number of miners to get minimum block generation time is 6 in our private blockchain network. As a central part of the Ethereum network, gas is used to pay for the computing resources consumed by miners. To evaluate process

Table 2 Impact of block generation and financial cost.

Time consumption of block generation						
Number of miners	2	3	4	5	6	7
Time (ms)	16.07	15.65	13.58	9.37	7.73	7.95
Estimated financial cost of transaction						
Gas (Wei)	159544.25		ETC Price (USD)		212.77	
Transaction fee (ETC)	0.0010853		Transaction fee (USD)		0.22	

overhead resulting from gas cost, 100 transactions that assign delegate certificate and capability are created on the blockchain network, and the average gas cost for each transaction is 169576.15 Wei (in Ethereum, $1\ \text{ether} = 1.0 \times 10^{18}$ Wei), which is around \$0.22 considered ETC value during the writing of this paper (September 20, 2018) is 1 ETC = 212.77\$.

5.4 *Discussion*

The experimental results demonstrate that the proposed BlendCAC strategy is effective and efficient to protecting the space devices and services from an unauthorized access request. Compared to centralized AC solutions, the BlendCAC scheme has the following advantages:

- *Decentralized Architecture*: Due to the decentralization provided by the blockchain technique, the proposed BlendCAC scheme allows masters to control their devices and resources instead of depending on a centralized third authority to establish the trust relationship with unknown nodes; thus, the bottleneck effect and the risk of malfunction resulting from centralized architecture are removed. Even in the worst case that a master is out of service, it has limited impact on the authentication (apart from adding new nodes to a virtual trust zone).
- *Edge Computing Driven Intelligence*: Thanks to federated delegation mechanism and blockchain technology, the BlendCAC framework provides a device-driven AC strategy that is suitable for the distributed nature of a space communications network. Through transferring power and intelligence from the centralized cloud server to the space network edge, smart objects are capable of protecting their own resources, enforcing privacy, and securing user-defined data content, which is meaningful to distributive, scalable, heterogeneous, and dynamic space scenarios.
- *Fine Granularity*: In the BlendCAC system, each entity uses its unique block-chain address for identity authentication and joins the virtual trust zone, and a capability token is only assigned to the authenticated entity. It is difficult for attackers to access services by using fake identities. Enforcing AR validation on local service providers empowers those smart devices to decide whether or not to grant access to certain services according to the local environmental conditions. Fine-grained AC with lease privilege access principle prevents privilege escalation, even if an attacker steals capability token.
- *Lightweight*: Compared to XML-based language for AC, such as XACML, JSON is a lightweight technology that is suitable for resource constrained platforms. Given the experimental results, our JSON-based capability token structure introduces small overhead on the general performance.

Although the proposed BlendCAC mechanism has demonstrated these attractive features, using blockchain to enforce AC policy in space systems, it also incurs new challenges in performance and security. The transaction rate is associated with confirmation time of the blockchain data, which depends on the block size and the time interval between the generations of new blocks. Thus the latency for transaction validation may not be able to meet the requirement in real-time SSA scenarios. In addition, as the amount of transactions increases, the blockchain becomes large. The continuously growing data introduce more overhead on storage and computing resources of each client, especially for resource constrained devices. Furthermore, the blockchain is susceptible to majority attack (also known as 51% attacks), in which once an attacker takes over 51% computing power of network by colluding selfish miners, they are able to control the blockchain and reverse the transactions. Finally, since the blockchain data are open to all nodes joined the blockchain network, such a property of transparency inevitably brings privacy leakage concerns. More research efforts are necessary to improve the trade-off when applying the BlendCAC in practical scenarios.

6 Conclusions

In this paper, we proposed a partially decentralized capability-based AC framework leveraging the smart contract and blockchain technology, called BlendCAC, to handle the challenges in AC strategies for SSA applications. A concept-proof prototype has been built in an SSA emulated physical network environment to verify the feasibility of the proposed BlendCAC scheme. The identity authentication scheme and capability-based access model are transcoded to smart contracts and work on the private Ethereum blockchain network. The desktops and laptops serve as miners to maintain the sanctity of transactions recorded on the blockchain, and Raspberry Pi devices act as edge computing nodes to access and to provide services. Extensive experimental studies have been conducted using a space network emulator and the results are encouraging. It validated that the BlendCAC scheme was able to efficiently and effectively enforce AC authorization and validation in a distributed and trustless network. This work has demonstrated that our proposed BlendCAC framework is a promising approach to provide a scalable, fine-grained and lightweight AC for space network applications.

While the reported work has shown significant potential, there is still a long way toward a complete decentralized security solution for real-world space scenarios. Deeper insights are expected. Part of our on-going effort is focused on further exploration of the blockchain-based AC scheme for real-world space imagery access scenarios. Furthermore, designing an efficient consensus mechanism to address current issues in blockchain, such as lower transaction rate and 51% attack, is another effort to enhance the blockchain network toward reliable SSA applications of RSO tracking and space weather monitoring.

References

1. E. Blasch et al., "Big data for space situation awareness," *Proc. SPIE* **10196**, 1019607 (2017).
2. P. Wheeler et al., "Satellite propulsion spectral signature detection and analysis through hall effect thruster plume and atmospheric modeling," *Proc. SPIE* **9974**, 99740T (2016).
3. R. F. Teehan, "Responsive space situation awareness in 2020," Tech. Rep., Air War Coll Maxwell AFB AL Center for Strategy and Technology (2007).
4. B. Jia et al., "Cooperative space object tracking using space-based optical sensors via consensus-based filters," *IEEE Trans. Aerosp. Electron. Syst.* **52**(4), 1908–1936 (2016).
5. R. Oliva, E. Blasch, and R. Ogan, "Applying aerospace technologies to current issues using systems engineering: 3rd AESS chapter summit," *IEEE Aerosp. Electron. Syst. Mag.* **28**(2), 34–41 (2013).
6. E. Blasch et al., "DDDAS for space applications," *Proc. SPIE* **10641**, 1064108 (2018).
7. S. P. Chin, "Game-theoretic homological sensor resource management for SSA," *Proc. SPIE* **7330**, 73300O (2009).
8. J. A. Kennewell et al., "An overview of space situational awareness," in *Int. Conf. Inf. Fusion*, pp. 1029–1036 (2013).
9. E. Blasch, S. Ravela, and A. Aved, *Handbook of Dynamic Data Driven Applications Systems*, Springer, New York City, New York (2018).
10. M. Crosby et al., "Blockchain technology: beyond bitcoin," *Appl. Innovation* **2**, 6–9 (2016).
11. S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system" (2008).
12. D. Shen et al., "An orbital emulator for pursuit-evasion game theoretic sensor management," *Proc. SPIE* **10196**, 101960D (2017).
13. H. Chen et al., "Comparison of several space target tracking filters," *Proc. SPIE* **7330**, 73300I (2009).
14. W. Faber, S. Chakravorty, and I. Hussein, "A randomized sampling based approach to multi-object tracking with comparison to HOMHT," in *IAAS/AIAA Astrodyn. Spec. Conf.* (2015).
15. E. Blasch, "Enhanced air operations using JView for an air-ground fused situation awareness UDOP," in *IEEE/AIAA 32nd Digital Avionics Syst. Conf. (DASC)*, IEEE, pp. 5A5-1–5A5-11 (2013).
16. D. Shen et al., "Network survivability oriented Markov games (NSOMG) in wideband satellite communications," in *IEEE/AIAA 33rd Digital Avionics Syst. Conf. (DASC)*, IEEE, pp. 6C2-1–6C2-9 (2014).
17. L. Gong, "A secure identity-based capability system," in *Proc., IEEE Symp. Secur. and Privacy*, IEEE (1989).
18. A. Ouaddah et al., "Access control in the internet of things: big challenges and new opportunities," *Comput. Networks* **112**, 237–262 (2017).
19. R. S. Sandhu et al., "Role-based access control models," *Computer* **29**(2), 38–47 (1996).
20. P. Samarati and S. C. de Vimercati, "Access control: policies, models, and mechanisms," *Lect. Notes Comput. Sci.* **2171**, 137–196 (2000).
21. L. M. S. De Souza et al., "SOCRADES: a web service based shop floor integration infrastructure," *Lect. Notes Comput. Sci.* **4952**, 50–67 (2008).
22. P. Spiess et al., "SOA-based integration of the internet of things in enterprise services," in *IEEE Int. Conf. Web Serv.*, IEEE, pp. 968–975 (2009).
23. E. Yuan and J. Tong, "Attributed based access control (ABAC) for web services," in *Proc. IEEE Int. Conf. Web Serv.* (2005).
24. W. W. Smari, P. Clemente, and J.-F. Lalande, "An extended attribute based access control model with trust and privacy: application to a collaborative crisis management system," *Future Gener. Comput. Syst.* **31**, 147–168 (2014).
25. S. Bhatt, F. Patwa, and R. Sandhu, "Access control model for AWS internet of things," *Lect. Notes Comput. Sci.* **10394**, 721–736 (2017).
26. N. Ye et al., "An efficient authentication and access control scheme for perception layer of internet of things," *Appl. Math. Inf. Sci.* **8**(4), 1617–1624 (2014).
27. J. B. Dennis and E. C. Van Horn, "Programming semantics for multiprogrammed computations," *Commun. ACM* **9**(3), 143–155 (1966).
28. S. Gusmeroli, S. Piccione, and D. Rotondi, "IoT@work automation middleware system design and architecture," in *IEEE 17th Conf. Emerging Technol. Factory Autom. (ETFA)*, IEEE, pp. 1–8 (2012).
29. B. Anggorojati et al., "Capability-based access control delegation model on the federated IoT network," in *15th Int. Symp. Wireless Pers. Multimedia Commun. (WPMC)*, IEEE, pp. 604–608 (2012).
30. G. Zhang and J. Tian, "An extended role based access control model for the internet of things," in *Int. Conf. Inf. Networking and Autom. (ICINA)*, IEEE, Vol. 1, pp. V1-319–V1-323 (2010).
31. J. L. Hernández-Ramos et al., "Distributed capability-based access control for the internet of things," *J. Internet Serv. Inf. Secur.* **3**(3/4), 1–16 (2013).
32. R. Xu et al., "A federated capability-based access control mechanism for internet of things (IoTs)," *Proc. SPIE* **10641**, 106410U (2018).
33. M. Swan, *Blockchain: Blueprint for a New Economy*, O'Reilly Media, Inc., California (2015).
34. N. Szabo, "Formalizing and securing relationships on public networks," *First Monday* **2**(9) (1997).
35. D. Nagothu et al., "A microservice-enabled architecture for smart surveillance using blockchain technology," arXiv:1807.07487 (2018).
36. S. Y. Nikouei et al., "Real-time index authentication for event-oriented surveillance video query using blockchain," arXiv:1807.06179 (2018).
37. R. Xu et al., "Constructing trustworthy and safe communities on a blockchain-enabled social credits system," arXiv:1809.01031 (2018).
38. M. T. Hammi et al., "Bubbles of trust: a decentralized blockchain-based authentication system for IoT," *Comput. Secur.* **78**, 126–142 (2018).
39. R. Xu et al., "BlendCAC: a smart contract enabled decentralized capability-based access control mechanism for the IoT," *Computers* **7**(3), 39 (2018).
40. R. Xu et al., "BlendCAC: a blockchain-enabled decentralized capability-based access control for IoTs," in *IEEE Int. Conf. Blockchain, Sel. Areas IoT and Blockchain*, IEEE (2018).
41. R. Mital et al., "Blockchain application within a multi-sensor satellite architecture," in *The Adv. Maui Opt. Space Surveillance Technol. Conf.* (2018).
42. Y. Zheng, E. Blasch, and Z. Liu, *Multispectral Image Fusion and Colorization*, SPIE Press, Bellingham, Washington (2018).
43. B. Liu et al., "An adaptive process-based cloud infrastructure for space situational awareness applications," *Proc. SPIE* **9085**, 90850M (2014).
44. "Ethereum homestead documentation," http://www.ethdocs.org/en/latest/index.html (accessed 8 February 2019).
45. Solidity, http://solidity.readthedocs.io/en/latest/ (accessed 8 February 2019).
46. Truffle, http://truffleframework.com/docs/ (accessed 8 February 2019).
47. D. Crockford, "The application/JSON media type for javascript object notation (JSON)," RFC 4627, Status: INFORMATIONAL (2006).
48. "Flask: a pyhon microframework," http://flask.pocoo.org/ (accessed 8 February 2019).
49. SQLite, https://www.sqlite.org/index.html (accessed 8 February 2019).
50. "BC_DDDAS project," https://github.com/samuelxu999/Research/tree/master/Security/py_dev/BC_DDDAS/ (accessed 8 February 2019).
51. "Go-ethereum," https://ethereum.github.io/go-ethereum/ (accessed 8 February 2019).

Ronghua Xu is a PhD student of electrical and computer engineering at Binghamton University—State University of New York (SUNY). He earned his BS degree in mechanical engineering from Nanjing University of Science and Technology, China, in 2007, and received his MS degree on mechanical and electrical engineering from Nanjing University of Aeronautics and Astronautics in 2010. His current research interests are blockchain based security solutions to internet of things (IoTs).

Yu Chen is an associate professor of electrical and computer engineering at Binghamton University—SUNY. He received a PhD in electrical engineering from the University of Southern California (USC) in 2006. His research interest lies in edge-fog-cloud computing, IoTs, and smart cities. His publications include over 150 papers in scholarly journals, conference proceedings, and books. He is a senior member of IEEE and SPIE, and a member of ACM.

Erik Blasch is with the Air Force Research Laboratory (AFRL). He received his BS degree from MIT and his PhD from Wright State University (1999) in addition to seven master's degrees. He has been with the AFRL since 1996, compiling over 750 papers, 25 patents, and 5 books. He is a fellow of IEEE, SPIE, and associate fellow of AIAA. His research areas include target tracking, image fusion, information fusion performance evaluation, and human-machine integration.

Genshe Chen is the CTO of Intelligent Fusion Technology, Inc., at Germantown, Maryland. He received his BS and MS degrees in electrical engineering and his PhD in aerospace engineering, in 1989, 1991, and 1994, respectively, all from Northwestern Polytechnical University, Xian, China. He was a postdoctoral fellow at Beijing University of Aeronautics and Astronautics and Wright State University from 1994 to 1997. He was with Intelligent Automation, Inc., Rockville, Maryland, from 2004 to 2007.