future internet

MDPI

Article

A Cross-Chain-Based Access Control Framework for Cloud Environment

Saad Belcaid [1,*], Mostapha Zbakh [1], Siham Aouad [1], Abdellah Touhafi [2] and An Braeken [2,*]

[1] ENSIAS, Smart Systems Laboratory (SSL), Mohammed V University in Rabat, Rabat 10000, Morocco; m.zbakh@um5r.ac.ma (M.Z.); siham.aouad@ensias.um5.ac.ma (S.A.)
[2] Department of Engineering, Technology (INDI), Vrije Universiteit Brussel, 1000 Brussels, Belgium; abdellah.touhafi@vub.be
* Correspondence: saad_belcaid@um5.ac.ma (S.B.); an.braeken@vub.be (A.B.)

Abstract: Cloud computing presents itself as one of the leading technologies in the IT solutions field, providing a variety of services and capabilities. Meanwhile, blockchain-based solutions emerge as advantageous as they permit data immutability, transaction efficiency, transparency, and trust due to decentralization and the use of smart contracts. In this paper, we are consolidating these two technologies into a secure framework for access control in cloud environments. A cross-chain-based methodology is used, in which transactions and interactions between multiple blockchains and cloud computing systems are supported, such that no separate third-party certificates are required in the authentication and authorization processes. This paper presents a cross-chain-based framework that integrates a full, fine-grained, attribute-based access control (ABAC) mechanism that evaluates cloud user access transaction attributes. It grants or denies access to the cloud resources by inferring knowledge about the attributes received using semantic reasoning based on ontologies, resulting in a more reliable method for information sharing over the cloud network. Our implemented cross-chain framework on the Cosmos ecosystem with the integrated semantic ABAC scored an overall access control (AC) processing time of 9.72 ms.

Keywords: blockchain; cross chain; access control; cloud security

Academic Editors: Jordi Mateo-Fornés and Choong Seon Hong

Received: 13 February 2025
Revised: 14 March 2025
Accepted: 25 March 2025
Published: 27 March 2025

Citation: Belcaid, S.; Zbakh, M.; Aouad, S.; Touhafi, A.; Braeken, A. A Cross-Chain-Based Access Control Framework for Cloud Environment. *Future Internet* **2025**, *17*, 149. https://doi.org/10.3390/fi17040149

1. Introduction

The growth of diverse, well-known businesses today leans on cloud computing, permitting clients to access their tailored requested services with ease. Cloud computing can offer on-demand access to resources and can handle vast amounts of shared data over distributed servers in the network.

The security of cloud computing is based on several aspects [1], such as data security, network security, threats related to the cloud infrastructure, and access control (AC) mechanisms [2]. The AC, a fundamental pillar in cloud security, directs the process in which a subject (a cloud client) has authorization to execute privileges on an object (cloud data or resources). It takes into consideration the data holder or service provider access policies that the request has to satisfy, intending to prevent unauthorized access, disclosure, or modification of data and resources. Conventional AC mechanisms for cloud computing—mandatory access control (MAC), discretionary access control (DAC), role-based access control (RBAC), attribute-based access control (ABAC), attribute-based access encryption (ABE), and federated identity management (FIM)—grant access control functionalities to cloud environments, but they embody a multitude of shortcomings [3]. MAC

has flexibility issues and a fixed count of users. DAC has scalability and maintenance problems. RBAC is prone to errors. ABAC is complex to maintain, and FIM has trust management issues.

Blockchain (BC) gained popularity with the development of cryptocurrencies, such as Bitcoin, and the evolution of web technology with version 3.0 [4]. In this new era, BC is considered a basic infrastructure for the development of decentralized applications and smart contracts. Thus, numerous industrial solutions (Ethereum, Cosmos, Avalanche, etc.) have emerged [5], using different approaches for managing and maintaining their ledgers. These solutions endorse their own coins and provide tamper evidence by linking blocks of signed transactions, using consensus mechanisms for validation and cryptographic techniques for robust security. However, we can extend this promising technology to a multitude of sectors [6]. It grants traceability, transparency, and auditability, ensuring that no entity has total control over the blockchain ledger. Thus, researchers have adopted BC technology, leveraging its strengths to propose access control mechanisms that tackle the shortcoming of the traditional ACs mentioned above.

The stated blockchain-based solutions operate autonomously. Thus, cross-chain technology, with its mechanisms and protocols, comes to allow communication within BC ecosystems. In a basic scenario, it involves two blockchains: one serving as the transaction's source and the other as the target, with the source BC initiating the transaction for execution on the target BC without the need for intermediaries. In recent years, researchers have utilized BC technology to provide more advanced control mechanisms for the cloud. In certain cases, researchers have utilized smart contracts in conjunction with third-party key management authorities and other data transport solutions, such as the interplanetary file system (IPFS). However, leaning on certification authorities represents a single point of failure for the proposed solutions, as the other peer-to-peer (P2P) propositions have BC communication interoperability issues. Therefore, in our contribution, we aim to address this research gap.

- We are introducing an access control framework for the cloud, which incorporates a BC-based, fine-grained ABAC mechanism. Cross-chain technology is used to connect the front- and back-end cloud as BC zones, thereby minimizing the attack surface on cloud assets.
- We also enhanced the ABAC mechanism with semantic reasoning based on ontologies. This combination gives our framework a higher level of adaptivity to any cloud service consumer (CSC) business needs.
- We evaluated the AC processing time and transaction latencies of the cross-chain communication using the Cosmos ecosystem, which ensures a high level of parallel scaling for the cloud stakeholders.

This paper organizes the remaining sections as follows: Section 2 highlights the related works. Section 3 presents preliminary concepts of our framework. Section 4 digs into the framework's functionality. Section 5 describes the implementation. We dedicate Section 6 to the evaluation, and in the final Section 7, we draw a conclusion and outline future works.

2. Related Works

Blockchain technology has influenced the workflow in cloud environments by implementing access control mechanisms in the cloud through blockchain. Several authors have addressed this in different ways according to the cloud application domain and the desired security impact.

Gajmal et al. [7] developed a mechanism of data sharing and access control for the cloud utilizing blockchain technology. The data are encrypted and uploaded to an interplanetary file system (IPFS) by the data owner. The smart contracts store a secure file location

and a secure key to simplify the sharing of data across multi-cloud storage platforms. Only a selected group of users registered by the data owner has access to retrieve it from the IFPS.

He et al. [8] suggested a different access control method using blockchain technology by implementing ABAC technology into a smart contract, creating fine-grained access control on the edge cloud servers. Each of the core ABAC components is designed into a corresponding smart contract. To minimize the execution time of the access control mechanism, policy reduction using rule clustering was deployed on the policy decision point (PDP). The execution result of these smart contracts is also stored on the blockchain.

Using ciphertext policy attribute-based encryption (CP-ABE) and Ethereum blockchain, Fugkeaw et al. [9] proposed a lightweight access control for fog-assisted IoT cloud medical data. It deploys an encryption/decryption algorithm for data sharing, capitalizing on fog nodes and BC capabilities. The aggregated data are partially encrypted, then sent through a trusted proxy to the nearest fog node, which encrypts and stores the indexed data on the cloud storage. It also serves to authenticate, verify, and manage user sessions.

Alharbi [10] developed a framework called ACE-BC for information sharing based on attribute-based encryption (ABE). The access control is deployed on the edge gateways, which connect to the blockchain as peers. The main aim of this work is to eliminate non-trusted third parties and use blockchain as an authority for key management so the data owner can check the data user's identity before authorizing access to a cloud service.

Mohan M and Sujihelen [11] proposed an access control framework based on Hyper-ledger Fabric using chain code for health organizations. The framework benefits plainly from blockchain technology. It allows secure storage of patient data while access is granted by a deployed chain code on the P2P hyper-ledger network. Access to data stored in cloud servers is granted to peers satisfying the smart contract endorsement policy. In this framework, transactions are made through a specific channel of the ledger network.

Duan et al. [12] suggested a framework called BSAF, established on blockchain-based technology, to manage access to the services offered by a cloud service provider (CSP). This entity uses homomorphic encryption (TFHE) to encrypt its offering. Service solicitants send access requests directly to the cloud server. Access-seeking requests are redirected to the blockchain network for verification, at which point penalties might be applied. As a result, a confirmation is sent to the users via the deployed smart contract on the blockchain network.

For privacy protection, Yang et al. [13] proposed an access control framework based on the EOS blockchain: the data owner (DO) uploads his resources to the cloud and publishes his permission and authorization to the blockchain. In this framework, the data user (DU) requests the cloud servers directly; this later authenticates DU and queries permissions from the BC. While the cloud grants or denies DU requests to access its resources, the BC is used as a policy database that stores permissions and authorizations as trustworthy tiers.

Jiang et al. [14] elaborated a cross-chain solution for access control between Tangle as a side chain and Fabric as the consortium BC, using the notary node network as a gateway between the two for routing and verifying transactions. The access control is deployed by a smart contract in the consortium BC, which also contains the public keys of the requesters, their access privileges, and data hashes. The public key has a central role in the proposed access control model, in which only requesters with public keys stored in the consortium BC can communicate with the notary nodes for granting access.

Based on the survey, it is important to highlight that there are several solutions designed for AC (summarized in Table 1) based on smart contracts and cryptographic techniques, which reflect the capability of blockchain to tackle pressing concerns such as privacy preservation, data security, and scalability.

While smart contracts (chain code) automate the process of enforcing access control policies, cryptographic techniques safeguard the confidentiality, integrity, and authenticity

of shared data. In [7–10,12,13], authors deployed ACs alongside certificate authorities (CAs) as a third-party solution for key management, which represents a single point of failure despite its existence for 53 years. This infrastructure, as it stands, embraces risks and problems in its usage [15].

Table 1. Surveyed blockchain-based access control.

Ref.	Year	Application	AC Type	Platform	Digital System
[7]	2024	Cloud storage	Smart agreement + AES	*	TP
[8]	2024	Edge computing	Smart contract + (ABAC, bloom filter)	Ethereum (private/public)	TP
[9]	2023	Fog IoT cloud-based	Smart contract + CP-ABE	Ethereum (public)	TP
[10]	2023	Cloud computing	Smart contract + ABE	*	TP
[11]	2023	Cloud-based medical systems	Smart contract (chaincode)	Hyperledger Fabric (private/permissioned)	P2P
[12]	2023	Cloud device services	Smart contract + (TFHE, bloom filter)	Ethereum (private)	TP
[13]	2020	Cloud storage	Smart contract + AES	EOS (public)	TP
[14]	2019	IoT-data management/cloud sidechain	Smart contract (chaincode)	Hyperledger Fabric (consortium)	Cross-chain

* not provided by the authors in the paper. AES: advanced encryption standard; ABAC: attribute-based access control; CP-ABE: ciphertext policy attribute-based encryption; ABE: attribute-based encryption; TFHE: fast fully homomorphic encryption over the torus; TP: third-party; P2P: peer-to-peer.

To tackle the highlighted issue, researchers in [11] adopted Hyperledger Fabric's permissioned network with a membership service provider (MSP) to establish a P2P connection and the IPFS's proof-of-replication (PoRep) mechanism to maintain data storage. However, MSPs have limited interoperabilty with non-Fabric networks, and as the network grows, the MSP becomes complex to manage, reflecting scalability issues.

Cross-chain is used in [14] as a reliable technology to eliminate trust problems attached to third-party certification authorities. However, using one attribute (the public key) to check access to the consortium blockchain may fall short in communicating and sharing resources between entities defining their access policies in a granular vocabulary.

On the other hand, our framework adopts the built-in Cosmos ecosystem accounts, in which token ownership and cryptographic proofs act as memberships for the cloud stakeholders. The proof-of-stake (PoS) consensus mechanism we use allows for faster broadcasting and validation of access transactions. We thus address the scalability and single point of failure issues of MSPs and CAs, respectively. We also integrated ABAC as a standardized AC mechanism over cross-chain, supporting it with semantic reasoning based on ontologies. Our work deals with transactions that originate from different types of chains (cloud clients), defined in a variety of CSC business vocabularies, and transported using the inter-blockchain communication (IBC) protocol, thereby addressing the inter-chain interoperability issue.

Table 2 compares the cross-chain-based access control framework in this paper with the other access control mechanisms. We base this comparison between our work and the current state of the art on criteria such as freshness (replay attack resistance), the addition of blocks to the chain, the heterogeneity reflecting the ability to support different technical implementations, the fine-grainedness of the access control mechanism used, whether the systems can scale to rising demands, and whether the solution can adapt to any CSC's

context vocabulary, such as banking, healthcare, education, etc. Some schemes do not meet these criteria because they utilize BC platforms that lack heterogeneity and interoperability, or they employ a static chain code for access control.

Table 2. Comparison among decentralized, data-privacy-preserving BC-based access control schemes.

Ref.	Year	Freshness	Block Finality	Heterogeneity	Fine-Grained	Scalability	Adaptivity
[7]	2024	*	*	*	No	Yes	No
[8]	2024	Moderate/high	Probabilistic	Low/moderate	Yes	Yes	No
[9]	2023	Moderate/high	Probabilistic	Low/moderate	Yes	No	No
[10]	2023	*	*	*	No	No	No
[11]	2023	High	Instant	Moderate	No	Yes	No
[12]	2023	Moderate/high	Probabilistic	Low/moderate	No	Yes	No
[13]	2020	Moderate	Probabilistic	Low/moderate	No	No	No
[14]	2019	High	Instant	Moderate	Yes	No	No
Our approach	2024	High	Instant	High	Yes	Yes	Yes

* not provided by the authors in the paper.

3. Preliminaries

Our framework is intended for the enhancement of cloud computing security by integrating fine-grained access control using the Cosmos blockchain [16]. This type of blockchain provides advantages such as customization for specific user needs, open source, modularity, and parallel chain operation. Additionally, it integrates the IBC protocol [17] with the aim of spreading an internet of blockchain.

We managed to implement a flexible, scalable, and modular AC mechanism over cross-chain technology. Knowing that two different BC ledgers do not directly transmit transactions on the BC network, we used the IBC protocol to link cloud stakeholders as BC zones: the cloud front end as a light client node "sending chain", and the back end cloud as a "receiving chain", governed by the cloud service provider (CSP).

Our proposed authorization framework is based on cross-chain technology, which intervenes with a relay process to recognize and update active cloud zones and their paths open to connection. Therefore, in our work, the cross-chain plays a crucial role in connecting and attempting to grant privileges on the available cloud resources and services.

3.1. Architecture

Our proposed framework, as shown in Figure 1, is based on the Cosmos industrial cross-chain solution, which integrates the IBC protocol to interconnect heterogeneous blockchains, including modules that define how packets and messages are constructed and interpreted by the sending and receiving cloud front- and back-end zones, in our case, BC lightweight and full nodes.

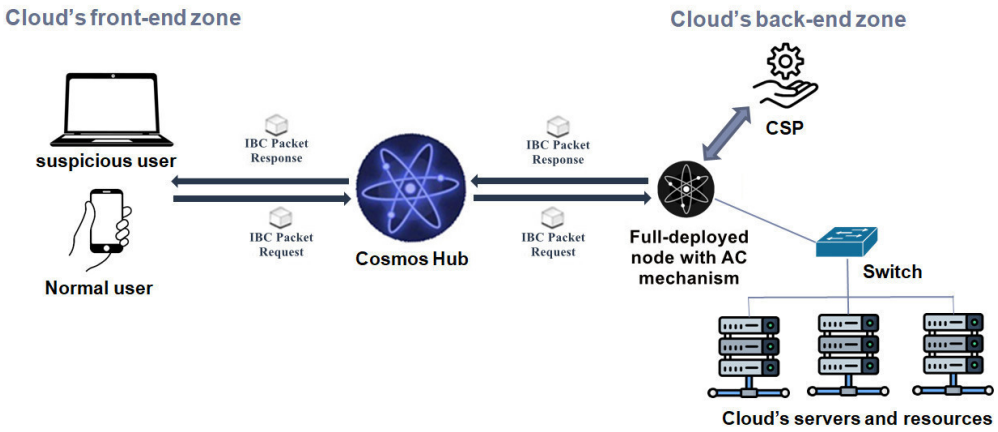

Figure 1. Architecture of the proposed framework.

This framework deploys a fine-grained modular AC in the receiving zone of the cloud architecture. It undertakes the operations of processing authorization transactions included in the transferred IBC packets during the interactions between cloud zones, where each received client request carrying access attributes must be processed semantically before enforcing an access decision.

The main objectives of our framework are:

- Enabling cross-chain communication to eliminate trust issues related to third parties within the cloud environment.
- Implementing an access control mechanism on the cloud back-end zone, governed by the CSP, where the cloud service consumers maintain their access policies on the deployed node, ensuring its privacy and integrity.
- Enhancing privacy preservation and resource availability in the cloud environment.
- Processing access requests originating from cloud clients using an enhanced ABAC with semantics for a scalable, flexible, and granular AC.
- Tackling the communication interoperability issues between heterogeneous BC zones implementing the IBC protocol.

3.2. Stakeholders

There are three main stakeholders in the proposed framework: the cloud users, the cloud resource holders, and the cloud service providers. First, the cloud users are located in the architecture's front-end zone (sending chain), where a built-in Cosmos account authenticates each cloud client to the blockchain network. Each user account holds a pair of keys (public and private) generated by the supported secp256k1 scheme. These keys serve to sign and verify transactions sent to the architecture's back-end zone (receiving chain). In the sending chain, users are identified by a unique ID, and they may use a lightweight node (which does not store a copy of the ledger) to send their transaction or a full node that stores an identical copy of the entire BC with all the transactions communicated to the receiving chain (in case of inter-organizational communication).

The cloud resources holder is an entity (organization) that is authenticated also to the BC network by a pair of keys using a built-in Cosmos network account, benefiting from the full BC access control node that stores the whole ongoing access request and transaction logs. This cloud service consumer (CSC) defines rules and policies outlining how the cloud front-end zone users can access the offered services and resources, dictating the conditions, actions, and privileges they have. The access policy rules are created and maintained by the CSC and enforced by the deployed ABAC mechanism within the AC node.

The third stakeholder is the cloud service provider, a business that offers on-demand scalable cloud computing resources. A CSP deploying our framework initiates the access control chain (the receiving chain) and provides the CSCs with authorization node capabilities to verify the access rights of the cloud users.

3.3. Technical Components

From a technical point of view, we distinguish two major components, the Cosmos Hub [16] and the ABAC mechanism.

The Cosmos Hub is one of the essential components of this framework. It stands as the first Cosmos network through which other zones communicate using the IBC protocol. This protocol upholds the cryptographic techniques and the consensus mechanism's security measures of the different blockchains interacting in the network. In our case, all transactions that go through the Cosmos Hub are submitted to the consensus mechanism "CometBFT" based on the Tendermint byzantine fault tolerance (BFT) consensus [18].

The second component is the ABAC mechanism, where access requests are granted or denied based on the attributes of the subject, object, environment conditions, and a collection of policies stated in relation to those attributes and conditions [19]. Our solution implements all ABAC components in the BC, specifically the receiving node, along with access policies and attribute sources.

4. The Framework's Functionality

To address the functioning of this framework, we highlight the three critical phases: cloud user registrations, requesting a cloud resource, and processing a user access transaction.

4.1. Registrations (Phase 1)

The actors in the framework (the cloud user, CSC, and CSP) register using the built-in Cosmos accounts, which they create locally. Using their own private key to sign transactions, the network authenticates the cloud users through a signature confirmation. The supported digital signature generation schemes are secp256k1 and secp256r1.

The resource holder (CSC) establishes a direct contract with the CSP, which links his real identity to his account. Furthermore, cloud clients identify themselves through unique account addresses derived from the held public key, thereby enhancing user privacy. In our cross-chain framework, a sending chain (cloud front-end zone) restricts the initiation of transactions to only a list of account addresses created by real, genuine users affiliated with the organization that governs this chain. To ensure the validity of the user attributes, the organization acts as a local certificate authority. Each cloud user generates a public key and creates a certificate signing request (CSR) using his private key. In the CSR, he provides his BC account address and his attributes (role, name, departement, city, etc.). The organization validates the user's identity and attributes included in the CSR by the physical presence of its staff or by documents that prove their affiliation to the organization (according to the organization's governance). If the CSR is legitimate, the organization signs it with its root certificate and issues a certificate to the cloud user. In this off-chain approach, the organization links each cloud user's unique BC account address to a unique, separate certificate. The organization generates and maintains a local file that maps the BC account addresses and the hashes of the issued certificates with the corresponding attributes, as well as the issuing and expiration dates. The organization (sending chain) uses this file to validate transactions before broadcasting and to revoke the certificates of cloud users.

The CSP registers in the Cosmos network and initiates the access control chain (receiving chain). The CSC regularly contracts with the CSP and creates an account to interact

with the access control node. The registration of the data/resource users to the Cosmos network is mandatory, as access to the cloud resources requires an authenticated Cosmos account, in which authentication is implemented as signature verification.

4.2. Requesting a Cloud Resource (Phase 2)

In this framework, a cloud user who has successfully authenticated to the BC network can attempt to retrieve a cloud resource from a CSC cloud server (as shown in Figure 2). Knowing the channel and port available for connection, he can initiate an access transaction. He chooses which attributes to include in his transaction. Those attributes are specified in the signed certificate received by the organization. He generates the transaction with the Cosmos SDK transaction builder, signs it, and then broadcasts it through the command-line interface.

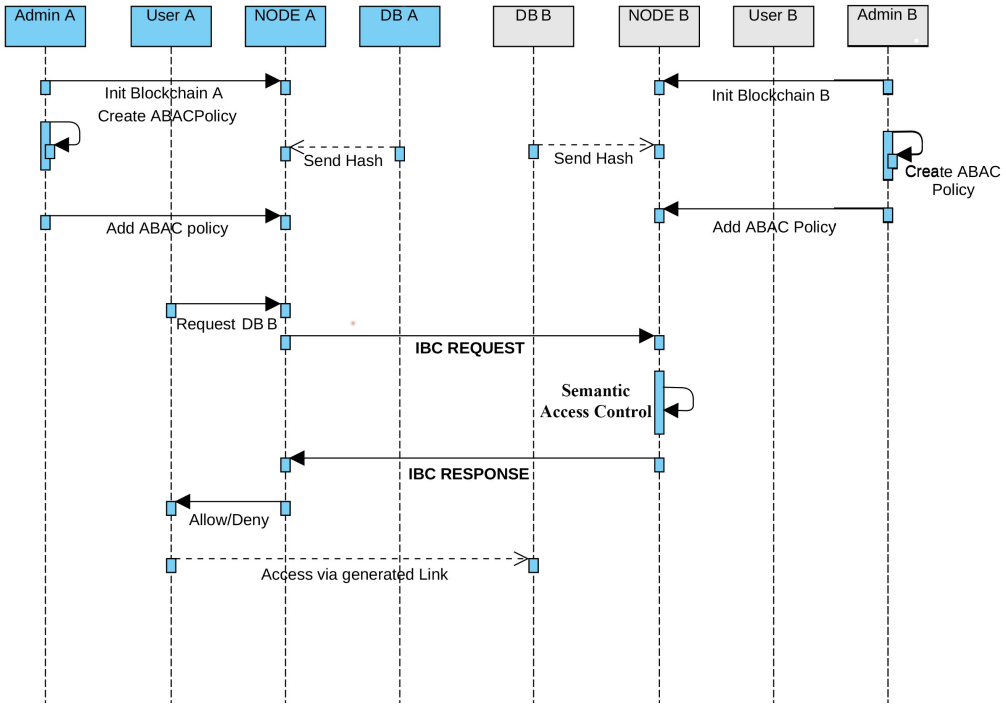

Figure 2. Sequence diagram of cloud users' IBC interactions between cloud front-end/back-end blockchains.

4.3. Transaction Processing (Phase 3)

The processing of the broadcast access transaction that originated from cloud user zones that integrate the IBC protocol can be resumed in the following steps (as shown in Figure 3):

1. The cloud user generates his access transaction addressed to the IBC-activated cloud front-end zone (sending chain). In this step, he indicates the corresponding channel and port, a hash of his certificate, as well as his client attributes.
2. Before broadcasting, the cloud user generates a digital signature by signing the transaction using his private key from the supported key scheme of the sending zone (in this case, the secp256k1 scheme).
3. The sending chain verifies the legitimacy of the transaction and compares the account address and the certificate hash combination with the stored addresses and hashes. If there is no match found, the transaction drops.
4. The signed transaction is encrypted using an encryption key of a symmetric algorithm, such as chacha20poly1305 or other algorithms used in the BC application layer (logic modules) of the sending chain. This layer also incorporates algorithms for encryption

key exchanges (such as Diffie–Hellman) that share the encryption key between the sending chain and the receiving chain. This is due to the IBC protocol, which uses the cryptographic primitives of the BC and the consensus mechanisms.

5. The sending chain's IBC module encapsulates the received data payload from the logic module into an IBC packet, generates cryptographic commitments of it based on Merkle roots, and stores it in the chain's states.
6. The relayer listens, queries the packet and commitment proofs from the sending chain, and then relays it to the receiving chain.
7. The IBC module of the receiving chain verifies the validity of commitment proofs, and then delivers the packet to the logic module for further processing.
8. The receiving chain decrypts the received tx, disclosing the tx information.
9. The access control mechanism is triggered automatically to process the received attributes.
10. If the attributes do not match directly with the specified CSC policies, a semantic process using ontology is started to draw knowledge about them.
11. If the attributes match the deployed policies directly or semantically, the controlling node creates a signed URL to the requested cloud resources. In our case, the URL is signed with the CSC's private key.
12. The access decision and the signed URL are added to the acknowledgement (ACK) packet.
13. Similar to the previous steps, the receiving chain uses its IBC module to send an ACK to the sending chain along with proofs that the receiving chain has committed the ACK to complete the cross-chain transaction.
14. The cloud user accesses the cloud resource via the received signed URL.

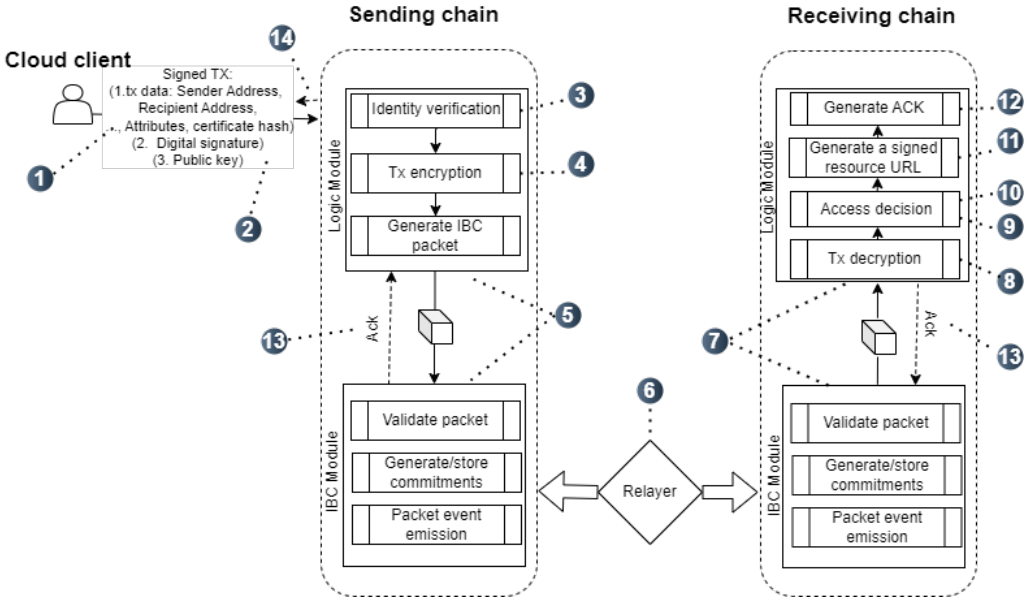

Figure 3. Flow diagram of a cloud client transaction.

4.4. Security Features

We now explain how the proposed architecture provides the following security requirements:

- Confidentiality: For the confidentiality of cloud resources and assets, we integrated a fine-grained attribute-based access control on the controlling node as an entry point to the cloud through cross-chain. It verifies the validity of the received access transaction and restricts access to only authenticated users with valid transaction attributes. For the confidentiality of the transactions carrying the attributes, each cloud

user transaction is encrypted (as in step 4 of transaction processing) and broadcast over a cross-chain network. The cloud users access the sought resources through a signed URL with the CSC's private key using Google API (Oauth 2). To maintain the confidentiality, this URL is generated securely by the controlling node in regard to the actions authorized to the users on the cloud resources, with an expiration time specific to the cloud resources' usage.

- Blockchain security: In order to protect our network from malicious attacks, we have adopted the IBC protocol. This protocol takes advantage of the existing consensus mechanism of the nodes in the network, in our case, Tendermint, using a byzantine fault tolerance that handles a 1/3 of the network nodes being malicious or faulty and a set of validators to agree on a block before making it valid. The PoS mechanism also ensures that no entity controls the network, as it is challenging to stack all the tokens. The cross-chain communication between the sending and receiving chains uses the Merkle roots and commitment, which are default features of the IBC protocol. The access transaction originating from the sending chain is recorded in a state Merkle tree. The control node (receiving chain) verifies the existence and integrity of the received transaction using a Merkle proof generated by the sending chain, while the commitment proofs guarantee that the transaction attributes are validated and committed in the sending chain's state. This combination of cryptographic proofs ensures data integrity and authenticity.
- Privacy: In this framework, cloud users are identified using unique account addresses derived from their generated public key. This approach enhances the privacy of the user's personal information. After a user signs a transaction in plain text, it undergoes encryption. Validators can verify the transparent parts of the transaction, including the signature, the cloud user's public key, and the transaction hash, while the transaction's metadata and access attributes remain encrypted.

4.5. Use Case Scenario

To test our developed AC framework, we simulated a communication scenario between doctors and a healthcare organization (laboratory). A doctor affiliated with a hospital considered a cloud client can transmit requests (IBC packets) from his end device through the healthcare application provided by his organization (the hospital).

A laboratory that adopts our access control framework as a security solution checks the doctor's transaction validity and, if the attributes match, grants access to the cloud resources sought, such as blood tests and x-ray reports, which are provided using Google Cloud Platform (GCP). By adding IBC core modules to both the hospital and laboratory chains, we can ensure the IBC protocol sends packets from the issuing zone to the laboratory while maintaining a relay process.

Upon receiving the doctor's access request by the laboratory in the form of a transaction enfolded in the IBC packet and the decryption of the received transaction attributes, the laboratory access control mechanism is triggered automatically. It employs semantic reasoning to verify whether the transaction attributes follow the matching rules outlined in the current laboratory access policy.

4.6. Requirements

To achieve the framework security features, the following requirements must be met:

The CSP deploys the AC node and configures the ABAC mechanism. This mechanism is defined by the request "r", which provides information about who (sub) wants to perform what (act) on which resource (obj), and enforces the policy "p", which generates a boolean decision (granting or denying access). Assuming that a doctor has attributes that fulfill an

access policy rule enumerated in the laboratory list of policies, a matcher defines in which way this doctor is allowed to perform any action (read, write, update, etc.) on the specific requested resource in relation to the evaluated authorization policy.

The cloud service consumer (laboratory) deploys policies and rules to decide actions and permissions for the objects it holds. The basic syntax of the used access policy is: p = sub, obj, act, eft, where "p" is policy, "sub" is subject, "act" refers to action, and "eft" to effect. The enforcement of these policies intends to verify that the sent attributes originating from the doctors match the rules specified by the laboratory in the list of policies deployed on its end.

The controlling node examines the doctor's "subject" access permission to the laboratory cloud resources "object" by evaluating the rules expressed by the laboratory in the access policy. Subject attributes include name, ID, role, affiliation to an organization or department, etc. Object attributes involve data creation date, resource owner, file name, criticality level, etc. Environmental attributes encompass elements such as access time, data location, used devices, etc. Access permissions are granted to doctors with attributes matching the subject, object, and environment conditions defined in the laboratory access policy.

Before making a decision, semantic reasoning is integrated with ABAC using an ontology according to the context in which the framework is adopted (healthcare, education, banking, etc.). The goal of using semantics in our proposed framework is to build knowledge about the attributes received in relation to CSCs access policies while enforcing a fine-grained access control that secures cloud data and resources.

5. Implementation

This section highlights the implementation process of our proposed AC framework for cloud computing. We used the Cosmos SDK with the Go language to create two test chains employed in the front- and back-end cloud, respectively. The system configuration used to develop the solution is an HP ProBook Intel® Core™ i5-10210U manufactured by Hewlett-Packard Inc., Palo Alto, CA, USA with 20 GB RAM; virtual machine (OS Debian Linux x64 manufactured by Offensive Security, New York, NY, USA) with 12 GB of RAM; Go version 1.23 (Linux/AMD64) Manufactured by Google LLC, Mountain View, CA, USA; vs. Code 1.65.2; Ignite for Cosmos SDK Framework; and Google Cloud Platform (GCP).

5.1. Blockchain Nodes

Using the "Ignite" development tool provided by Cosmos, which is designed for creating, launching, and maintaining BCs, we initialized a blockchain intended to act as a security layer for the cloud environment. We developed a module to manage the upcoming cloud request, in which we scaffold transactions for creating requests, managing transaction timeouts, generating acknowledgments, and listing received requests with a decision attached to each transaction. To interact with the deployed access control node, we initialized the sending chain through which cloud clients broadcast their requests. We tackled the networking part, which consists of a cross-chain communication established through the Go relayer endorsed by the Cosmos Hub. As two cloud zones cannot transmit transactions directly over the network, we managed to generate a code defining how the cross-chain packets are constructed and interpreted by the sending and receiving chains. A process of relaying ensures the recognition of active zones and their paths (channels and ports) that are open to connection. The configuration of the used Go relayer is shown in Figure 4.

```
-# ignite relayer configure -a \
 --source-rpc "http://0.0.0.0:26657" \
 --source-faucet "http://0.0.0.0:4500" \
 --source-port "ctrlac" \
 --source-version "ctrlac-1" \
 --source-gasprice "0.0000025stake" \
 --source-prefix "cosmos" \
 --source-gaslimit 300000 \
 --target-rpc "http://0.0.0.0:26659" \
 --target-faucet "http://0.0.0.0:4501" \
 --target-port "ctrlac" \
 --target-version "ctrlac-1" \
 --target-gasprice "0.0000025stake" \
 --target-prefix "cosmos" \
 --target-gaslimit 300000
```

Figure 4. The relayer configuration.

5.2. ABAC Mechanism

By adding and implementing the configured ABAC mechanism (as shown in Figure 5) into the BC, we can benefit from its component advantages, such as lightweight nodes, full nodes, consensus mechanisms, and smart contracts. It grants some capabilities unsupported by traditional AC mechanisms [20]. The use of BC for AC enhances the decentralized aspects and flexibility of our authorization framework, eliminating the single point of failure problem caused by distributed denial of service attacks and overcoming data tampering issues. It also increases the number of attributes processed, enhancing the confidentiality and availability of shared information in our framework.

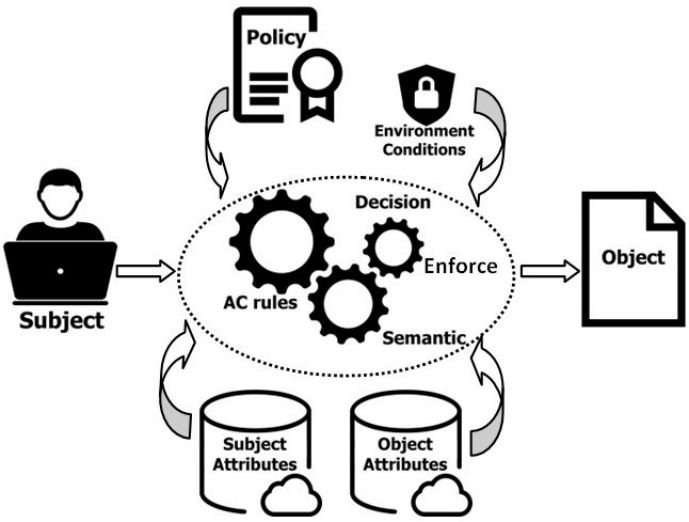

Figure 5. The deployed ABAC access control mechanism.

5.3. Access Policies

To generate an access decision, the logic module developed in the receiving chain matches the received cloud client attributes with the CSC access policies as shown in Figure 6. Those policies are specified in a CSV file format. We highlight that the governance actions on the policy files do not require a change in the chain state, which guarantees the availability of the cloud service. For the definitions, the cloud requests are defined by r = sub, obj, act, and the policies are defined by p = sub_rule, obj, act.

policylaboratory.csv

```
p, r.sub == 'Doctor',xrays_report, read
p, r.sub == 'Doctor',radiology_report, read
p, r.sub == 'Radiologist',xrays_report, read/write
p, r.sub == 'Scientist',bloodtest_report, read/write
p, r.sub == 'Labtechnicien',endoscopy_report, read
p, r.sub == 'Patient',patient_Data, read
p, r.sub == 'Physician',pathology_report, read
p, r.sub == 'Pathologist',pathology_report, read/write
p, r.sub == 'Surgeon',surgery_report, read/write
p, r.sub == 'Surgeon',bloodtest_report, write
p, r.sub == 'Physician',medical_record, read
p, r.sub == 'Oncologist',treatment_plan, read
p, r.sub == 'Cardiologist',radiology_report, read/write
p, r.sub == 'Neurologist',brainimaging_report, read
p, r.sub == 'Dermatologist',skinbiopsy_report, read
p, r.sub == 'Pediatrician',childhealth_record, read
p, r.sub == 'Gynecologist',patient_record, update
p, r.sub == 'Psychiatrist',mentalhealth_report, read/write
p, r.sub == 'Endocrinologist',hormone_report, read
p, r.sub == 'Orthopedist',orthopey_report, read
p, r.sub == 'Pulmonologist',pulmonology_report, read
p, r.sub == 'Hematologist',hematology_report, read
p, r.sub == 'Ophthalmologist',ophthalmology_report, read
p, r.sub == 'Rheumatologist',rheumatology_report, read
p, r.sub == 'Urologist',urology_report, read
p, r.sub == 'Nephrologist',nephrology_report, read
```

Figure 6. The deployed access policies within the ABAC access control mechanism.

5.4. Semantics

To handle the ABAC complexity challenges, we developed a test ontology (as shown in Figure 7) that represents a set of formalization languages improving data interoperability [21].

In our case, we used semantic reasoning in combination with the ABAC mechanism to ensure interoperability between the received and expressed access policy attributes, which were evaluated by the access control mechanism deployed in our framework.

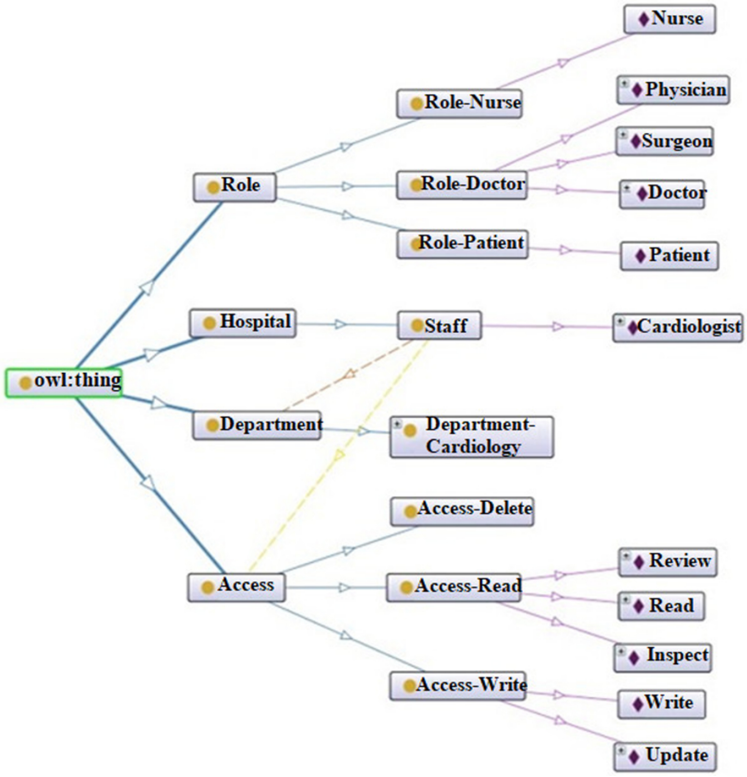

Figure 7. The deployed test ontology.

In addition to the configured and deployed ABAC mechanism, the CSC ontology is attached to the access control node, enriching the decision-making process of the AC by inferring stored knowledge about the received access attributes. Semantic reasoning also grants our framework a dynamic adaptation in a variety of use contexts, providing

flexibility to CSC's access policies and cloud client attributes. The developed semantic reasoner uses the ontology file of CSC, taking into account the meaning rather than the structure of the received access transaction data, which leads to extending the list of attributes by adding meaning to the received transaction attributes. It also makes it possible to draw knowledge from policies during the enforcement process for precise and more flexible authorization. To test our framework, we deployed the test ontology that was converted to the Go language and implemented into the control node, ensuring its integrity. The ontology file can be updated or extended with ease locally, as the on-chain pointer to the file stays unaltered.

6. Evaluation

In these experiments, we evaluated different types of access transactions based on the outcome of the access control node, as shown in Figure 8. For legitimate cloud users, we proceeded with attributes that fit perfectly in the access policy list developed by the resource-holding organizations (laboratory or hospital in the adopted scenario).

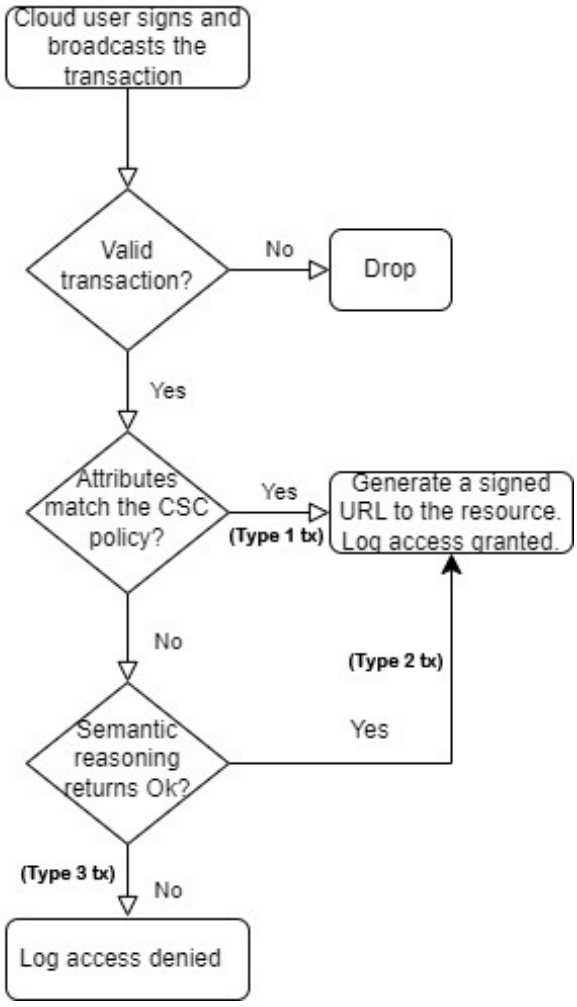

Figure 8. Access control flow chart.

We delved into the second type of transactions, simulating cloud users broadcasting "txs" with attributes that did not match directly with the CSC policy list but were found semantically while querying semantic knowledge from the ontology.

The third type of transaction simulates access "txs" originating from malicious cloud users, leading to a denying decision. The transaction logs were fetched from the access control node, including the calculated measures about the proceeded transactions.

6.1. Metrics

To measure the performance of this framework, we used the following metrics.

- The tx relay latency is calculated programmatically by measuring the time elapsed from the first broadcast of the tx to the reception of an acknowledgment carrying an access decision returned to the same sending zone.
- The access control's processing time is part of the tx relay latency. It is calculated separately by measuring the time elapsed from the triggering of the AC mechanism to the generation of an access decision.
- Transactions per second (TPS): measures the number of transactions that the chain can process per second.
- Throughput: measures the amount of data in bytes per second (BPS) that the chain can handle by the second.

6.2. Result and Analysis

Analyzing the extracted data from the BC logs of 50 different access transactions we constructed, we found that it takes an average of 5.39 ms to check access for legitimate cloud users with matching attributes that do not need a semantic intervention, with a 95% confidence interval (CI) of 4.57 to 6.20 ms (as shown in Figure 9a). The transaction latency of the first type hovers around an average of 13.50 s, with a 95% CI of 12.93 to 14.05 s (as shown in Figure 9b). The latency in our framework seems significant, but in contrast to other one-chain solutions, our cross-chain involves two verifications of cryptographic proof (the Merkle and commitment proofs) in each of the sending and receiving chains with instant block finality.

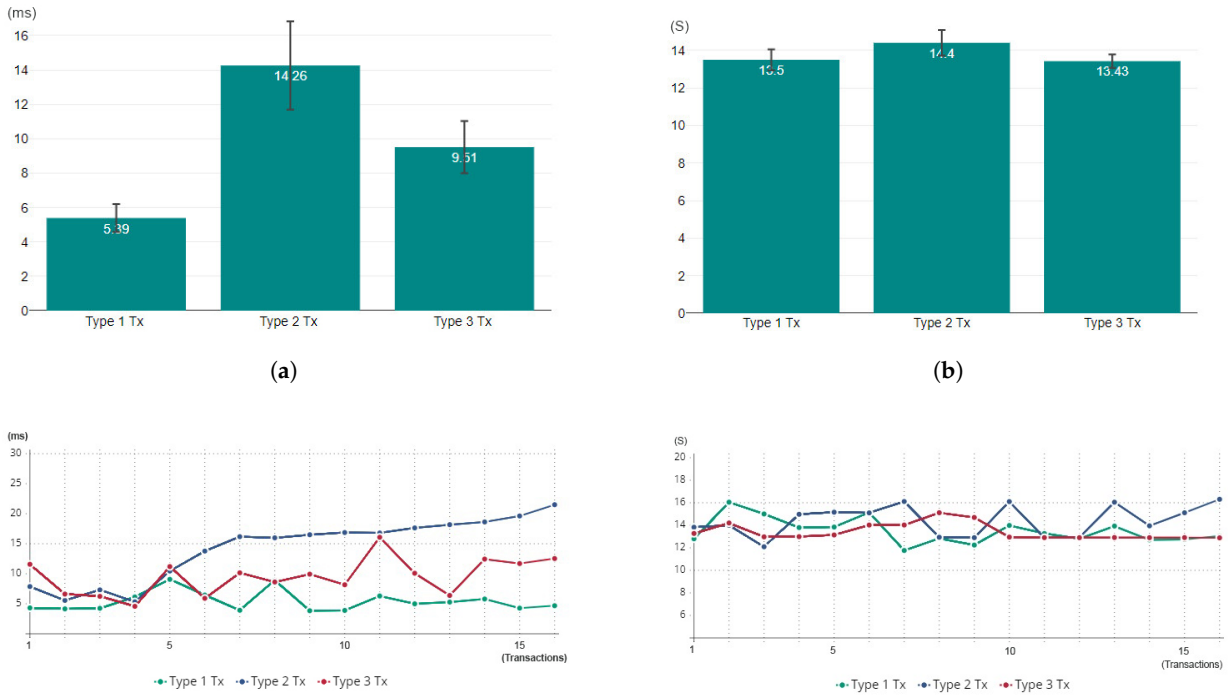

Figure 9. AC processing time and latency results: (**a**) AC processing time averages with 95% CI (ms); (**b**) transaction latency averages with 95% CI (s); (**c**) AC processing time for different transaction types (ms); (**d**) transaction relay latency for different transaction types (s).

The average AC time for a decision using semantic reasoning to grant access is 14.26 ms, with a 95% CI of 11.69 to 16.82 ms. Due to the additional process required to extract

knowledge about the attributes received from the ontology file, the tx relay latency reaches an average of 14.40 s and a 95% CI of 13.72 to 15.08 s.

The third type of tx scores an AC processing time that averages 9.51 milliseconds with a 95% CI of 8 to 11.02 ms. The average relay latency of this type of transaction is 13.43 s, with a 95% CI of 13.06 to 13.79 s.

As shown in Figure 9c, we compared the processing times of the deployed AC mechanism for the different evaluated transaction types, and we found that the first type of tx gives the fastest decision processing time for positive responses. The first-type tx AC processing times fall under the second tx type, which implies that our framework is efficient in processing requests from legitimate cloud users.

To benchmark the performance of our framework, we stress-tested the Tendermint-based blockchain with a load of transactions. We initialized the test with 250-byte txs directed to the chain. This test covers an interval from 100 to 7000 TPS (as shown in Figure 10) in which we measured the amount of data in BPS for each value of TPS.

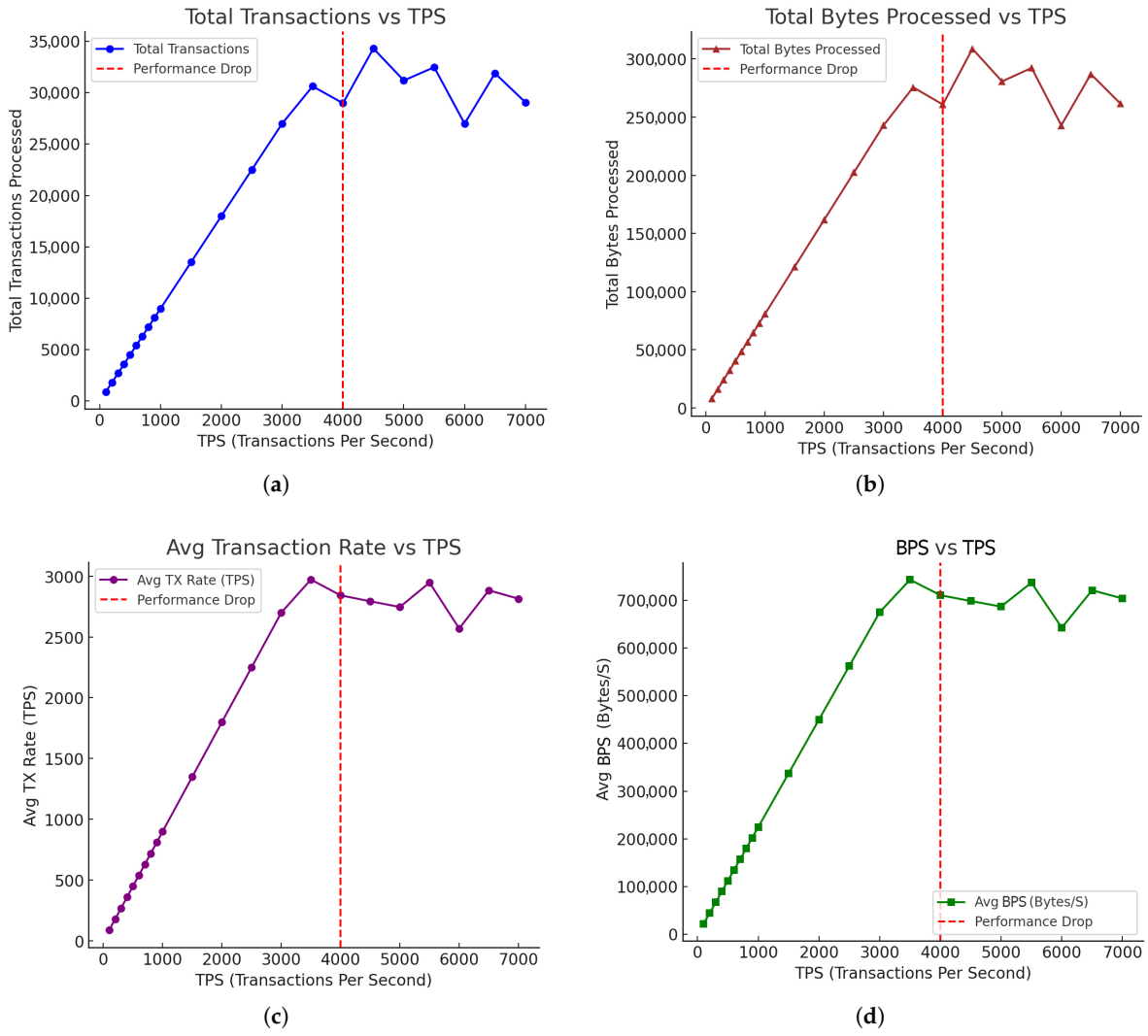

Figure 10. TPS measure: total transactions vs. TPS. (**a**) Throughput measure: total bytes processed vs. TPS. (**b**) TPS measure: average transaction rate vs. TPS. (**c**) Throughput measure: average BPS vs. TPS (**d**).

The total and average transactions, as shown in Figure 10a,c, increase almost linearly up until 4000 TPS. The same pattern is also seen in the total bytes processed in Figure 10b, which increases proportionately until about 4000 TPS. Beyond 4000 TPS, deviations come

up, signaling that the system has hit a performance bottleneck, in which the transaction processing capacity becomes inconsistent with a 10% efficiency drop. The instant blockchain finality of Tendermint's consensus mechanism may also cause performance bottlenecks, as it cannot keep up with the growing number of transactions at 7000 TPS; the efficiency is at 46%. The average data rate (BPS), as shown in Figure 10d, peaks close to 750,000 Bytes/s and also becomes linear in proportion to the TPS. In these tests, we observed that up to 4000 TPS, the efficiency is constant at 250 bytes per transaction but decreases after 4000 TPS, indicating deviations and potential resource exhaustion.

7. Discussion

Our framework adopts the Cosmos Hub, which provides BC parallelization for faster tx processing. Figure 9d illustrates the range of measured and recorded tx relay latencies, with an overall relay latency of 13.77 s, ranging from 11.76 s to 16.30 s. We clarify that the calculated 13.77 s average latency time includes the framework's instant block finality time (6–7 s), which ensures a higher security and immutability compared to other blockchains. Ethereum L1 finality, for example, is reached in 12 min. It should also be noted that the overall latency of this framework can vary in relation to the computational resources used in generating a positive access decision (the second type of tx), which requires additional ontology parsing, resulting in a longer AC processing time. In addition, the size of the ontology file used and the number of evaluated attributes influence the AC processing times for this specific type of access transaction. Semantically evaluating the attributes makes this framework more flexible and adaptive, allowing advanced and fine-grained attribute-based access control in a cloud environment. Denying a cloud user access transaction also requires semantic reasoning without significantly acquiring knowledge about the forged attributes. In contrast to the second type of tx, the denied tx has lower AC processing times, as the cloud resource was not interrogated by the AC control node to generate a signed URL to the requested resource. Those results indicate that our framework can handle both quick access processing and additional complex processes for granting or denying an upcoming cloud client access request.

We compared the relay performance scores for our framework with other cross-chain solutions and found that our results align with theirs. For example, in [22], the authors used the Zkbridge protocol to maintain inter-blockchain communication, resulting in a relay latency of 18 s. In [14], the latency varies from 5.38 to 23.34 s, while the authors in [23] used an asynchronous cross-chain model based on conditional transactions, achieving a relay latency that varies from 7 to 17 s.

8. Conclusions

The integration of blockchain technology into the cloud has proven to be one of the most promising solutions to address security issues raised within distributed cloud environments. Smart contracts reduce the risk of human errors, and the BC's inherent cryptographic technique and specific consensus mechanism prevent hacker attacks effectively.

The heterogeneity of blockchain and the lack of an inter-chain communication standard lead to a slow pace of adoption. Several interoperability propositions based on inter-chain communication technology, such as Cosmos, Hyperledger, and Polkadot, have emerged, leading to the development of the IBC protocol, which unifies the blockchain network.

In this work, we elaborated a framework based on cross-chain technology, allowing two cloud zones, back-end and front-end, to communicate by integrating the IBC core modules while offering a semantically enriched attribute-based access control mechanism.

Most of the highlighted works still trust certification authorities, as shown by our studies on current blockchain-based access control solutions in the cloud. In contrast, our

framework is built on cross-chain technology to relay cloud consumers and the sought resources. We also deployed semantic and modular AC, granting more flexibility to the framework.

In our framework, the cloud users are connected to the network as BC nodes that integrate IBC modules, authenticated by BC built-in accounts, and authorized to access the cloud assets through a semantic, fine-grained access control.

In the future, we will conduct tests on public blockchain networks with a thorough analysis of associated costs, investigate the potential threats this framework could face, highlight the different types of attacks that cloud could face (web 2.0 vs. web 3.0), evaluate the performance of our framework on other cloud service providers than GCP, and discuss the possibility of combining the proposed framework with other security mechanisms to strengthen the confidentiality, integrity, and availability of cloud resources while addressing traceability issues related to cloud stakeholders' activities. We are also currently working to introduce a decentralized self-sovereign identity roll-up in order to avoid the heavy reliance on the CSC (sending chain) as a certification authority for centric-identity verification. Finally, we will investigate how to make the cross-chain bridge to other non-IBC blockchains like Ethereum or Avalanche.

Author Contributions: Conceptualization, S.B., M.Z., S.A., A.T. and A.B.; methodology, S.B., M.Z., S.A., A.T. and A.B.; writing—original draft preparation, S.B.; writing—review and editing, S.B., M.Z., S.A., A.T. and A.B.; visualization, S.B.; supervision, M.Z., S.A., A.T. and A.B.; project administration, M.Z. and A.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Dataset available on request from the authors.

Acknowledgments: This study was performed within the framework of the 5G-Medic project supported by Federal Public Service (FOD) Belgium, Cybersecurity Research Program Flanders-second cycle (VOEWICS02), and EU COST Action CA22104 (Behavioral Next Generation in Wireless Networks for Cyber Security).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Liu, S. Towards Secure Blockchain-enabled Cloud Computing: A Taxonomy of Security Issues and Recent Advances. *Int. J. Adv. Comput. Sci. Appl.* **2023**, *14*, 917–926. [CrossRef]
2. Hu, V.C.; Iorga, M.; Bao, W.; Li, A.; Li, Q.; Gouglidis, A. *General Access Control Guidance for Cloud Systems*; Technical Report NIST SP 800-210; National Institute of Standards and Technology (U.S.): Gaithersburg, MD, USA, 2020. [CrossRef]
3. El Sibai, R.; Gemayel, N.; Bou Abdo, J.; Demerjian, J. A survey on access control mechanisms for cloud computing. *Trans. Emerg. Telecommun. Technol.* **2020**, *31*, e3720. [CrossRef]
4. Gan, W.; Ye, Z.; Wan, S.; Yu, P.S. Web 3.0: The Future of Internet. *arXiv* **2023**, arXiv:2304.06032.
5. Werth, J.; Berenjestanaki, M.; Barzegar, H.; El Ioini, N.; Pahl, C. A Review of Blockchain Platforms Based on the Scalability, Security and Decentralization Trilemma. In Proceedings of the 25th International Conference on Enterprise Information Systems, Prague, Czech Republic, 24–26 April 2023; pp. 146–155. [CrossRef]
6. Yaga, D.; Mell, P.; Roby, N.; Scarfone, K. *Blockchain Technology Overview*; Technical Report NIST IR 8202; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2018. [CrossRef]
7. Gajmal, Y.; More, P.; Jagtap, A.; Kale, K. Access control and data sharing mechanism in decentralized cloud using blockchain technology. *J. Auton. Intell.* **2024**, *7*, 3.
8. He, G.; Li, C.; Shu, Y.; Luo, Y. Fine-grained access control policy in blockchain-enabled edge computing. *J. Netw. Comput. Appl.* **2024**, *221*, 103706. [CrossRef]
9. Fugkeaw, S.; Wirz, L.; Hak, L. Secure and Lightweight Blockchain-Enabled Access Control for Fog-Assisted IoT Cloud Based Electronic Medical Records Sharing. *IEEE Access* **2023**, *11*, 62998–63012. [CrossRef]
10. Alharbi, A. Applying Access Control Enabled Blockchain (ACE-BC) Framework to Manage Data Security in the CIS System. *Sensors* **2023**, *23*, 3020. [CrossRef] [PubMed]

11. Mohan M, S.; Sujihelen, L. An efficient chain code for access control in hyper ledger fabric healthcare system. *e-Prime-Adv. Electr. Eng. Electron. Energy* **2023**, *5*, 100204. [CrossRef]
12. Duan, L.; Xu, W.; Ni, W.; Wang, W. BSAF: A blockchain-based secure access framework with privacy protection for cloud-device service collaborations. *J. Syst. Archit.* **2023**, *140*, 102897. [CrossRef]
13. Yang, C.; Tan, L.; Shi, N.; Xu, B.; Cao, Y.; Yu, K. AuthPrivacyChain: A Blockchain-Based Access Control Framework With Privacy Protection in Cloud. *IEEE Access* **2020**, *8*, 70604–70615. [CrossRef]
14. Jiang, Y.; Wang, C.; Wang, Y.; Gao, L. A Cross-Chain Solution to Integrating Multiple Blockchains for IoT Data Management. *Sensors* **2019**, *19*, 2042. [CrossRef] [PubMed]
15. Dumitrescu, A.T.; Pouwelse, J. Failures of public key infrastructure: 53 year survey. *arXiv* **2024**, arXiv:2401.05239.
16. Kwon, J.; Buchman, E. Cosmos Whitepaper. A Network of Distributed Ledgers. 2019. Available online: https://cosmos.network/whitepaper (accessed on 27 April 2023).
17. Goes, C. The Interblockchain Communication Protocol: An Overview. *arXiv* **2020**, arXiv:2006.15918.
18. Buchman, E.; Kwon, J.; Milosevic, Z. The latest gossip on BFT consensus. *arXiv* **2019**, arXiv:1807.04938.
19. Hu, V.C.; Ferraiolo, D.; Kuhn, R.; Schnitzer, A.; Sandlin, K.; Miller, R.; Scarfone, K. *Guide to Attribute Based Access Control (ABAC) Definition and Considerations*; Technical Report NIST SP 800-162; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2014. [CrossRef]
20. Hu, V.C. *Blockchain for Access Control Systems*; Technical Report NIST IR 8403; National Institute of Standards and Technology (U.S.): Gaithersburg, MD, USA, 2022. [CrossRef]
21. Horrocks, I.; Patel-Schneider, P.F.; Van Harmelen, F. From SHIQ and RDF to OWL: The making of a Web Ontology Language. *J. Web Semant.* **2003**, *1*, 7–26. [CrossRef]
22. Xie, T.; Zhang, J.; Cheng, Z.; Zhang, F.; Zhang, Y.; Jia, Y.; Boneh, D.; Song, D. zkBridge: Trustless Cross-chain Bridges Made Practical. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, Los Angeles, CA, USA, 7–11 November 2022; pp. 3003–3017. [CrossRef]
23. Su, H.; Guo, B.; Lu, J.Y.; Suo, X. Cross-chain exchange by transaction dependence with conditional transaction method. *Soft Comput.* **2022**, *26*, 961–976. [CrossRef]