Privacy-Preserving Access Control in Cloud Federations

Shorouq Alansari, Federica Paci, Andrea Margheri, Vladimiro Sassone
University of Southampton
{sa3n13; f.m.paci;a.margheri;vsassone}@soton.ac.uk

Abstract—**A Cloud federation is a collaboration of organizations sharing data hosted on their private cloud infrastructures in order to exploit a common business opportunity. However, the adoption of cloud federations is hindered by member organizations' concerns on sharing their data with potentially competing organizations. For cloud federations to be viable, federated organizations' privacy concerns should be alleviated by providing mechanisms that allow organizations to control which users from other federated organizations can access which data. We propose the architecture of a novel identity and access management system part of FaaS, a cloud federation service developed by the H2020 SUNFISH project. Our system allows federated organizations to enforce attribute-based access control policies on their data in a privacy-preserving fashion. Users are granted access to federated data when their identity attributes match the policies, but without revealing their attributes in clear. The architecture relies on two novel technologies, blockchain and Intel SGX hardware platform to guarantee integrity of the policy evaluation process.**

Index Terms—**Blockchain, Access control, Anonymous identities, Cloud federation.**

1. Introduction

A Cloud federation is a collaboration of different organizations in which organizations can gain access to data hosted on other federation members' cloud platforms in order to solve a specific problem or exploit a specific business opportunity. However, for cloud federations to be viable it is important to adopt solutions that enable secure data sharing among the participating organizations. In particular, it is necessary to have identity and access management systems that allow federated organizations to authenticate users from other joining organizations and determine the permissions that these have on shared data.

The research community has proposed few identity and access management systems for federated clouds [1], [2], [3]. However, these systems have several limitations.

First, they require trust agreements among the federated organizations to verify users identities and to map their access control policies on shared data. These solutions are not suitable for cloud federations, where collaborations are dynamic and managed on-the-fly. Most of all, partner organizations should be able to verify users' identities and enforce access control policies on shared data without establishing trust agreements among them.

Second, the proposed solutions provide fine-grained access control policies based on users' identity attributes. However, they do not protect the privacy of the attributes, which may convey sensitive information that organizations are not willing to share with potentially competing organizations [3].

Last, the distribute enforcement architecture of access controls policies: different components evaluate policies and, consequently, enforce the calculated decisions. However, malicious users can corrupt the access control policies or their evaluation engine to gain unauthorized access to the data shared within a federation.

In this paper we propose the architecture of a novel identity and access management system for cloud federations that resolve by design all these limitations. This system is here presented in the context of FaaS (Federation-as-a-Service) [4], a cloud federation service recently proposed by the H2020 project SUNFISH [5]. Specifically, FaaS is implemented by the SUNFISH software platform, whose identity and access management will amount to our system.

The proposed system has many new and innovative features. First, it uses a *cryptographic protocol* to allow federated organizations to enforce attribute-based access control policies, while preserving the privacy of users identity attributes. Indeed, any federated organization accessing shared data will not learn any user identity attributes nor whether a policy is satisfied. Second, it requires *no establishment of trust agreements* among organizations in order to authenticate users and enforce access control policies. Third, it ensures *integrity of access control components* responsible for evaluating and enforcing access control policies.

This system permits trustworthy enforcement of access control policies within a trustless setting, while ensuring the integrity of access control policies and components. To this aim, we leverage on two novel technologies: *blockchain* and *Intel SGX* trusted execution environment.

Paper Structure. Section 2 outlines the main system features. Section 3 presents the system architecture and the access control enforcement protocol. Section 4 concludes and discusses future research directions.

2. System Overview

In this section we provide an overview of the proposed identity and access control system for cloud federation. As this system is intended to be part of the SUNFISH software platform, we first introduce FaaS and its platform.

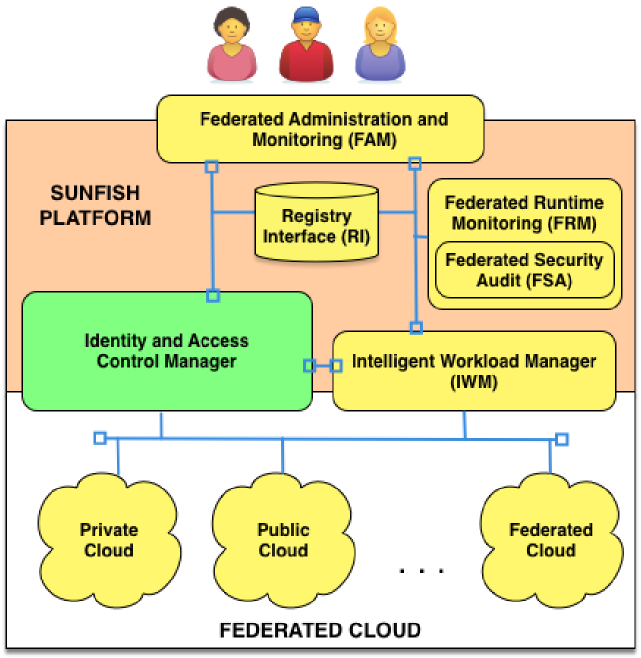
Federated Administration and Monitoring (FAM)
SUNFISH PLATFORM
Registry Interface (RI)
Federated Runtime Monitoring (FRM)
Federated Security Audit (FSA)
Identity and Access Control Manager
Intelligent Workload Manager (IWM)
Private Cloud
Public Cloud
. . .
Federated Cloud
FEDERATED CLOUD

Figure 1. FaaS software platform

FaaS amounts to a service for clouds that enables the secure creation and management of cloud federations. To this aim, it is supported by the SUNFISH software platform, whose graphical description is reported in Figure 1. The platform permits federating cloud data and services, and ensuring their secure provisioning and access via reliable identity management and access control functionality. Specifically, the platform components reported in the figure can be logically identified as *Federation Monitoring*, *Federation Management* and *Identity and Access Control Manager*. Here, we only focus on the latter, i.e. the green one, but we briefly overview anyway the rest.

The Federation Monitoring consists of a set of distributed probes that enable the runtime monitoring of all interactions among federated clouds; interactions are also further audited offline. The Federation Management supports the creation and management of cloud federations. Indeed, the IWM and FAM components are in charge of computing optimized workload strategies based on the requests to be executed on the member cloud systems. Notably, via the RI component, all the governance data of a federation, e.g SLA and access control policies, are stored on a registry.

The Identity and Access Control Manager corresponds to our identity and access control management system. The system provides the following features:

Anonymous Identities. The system allows users of federated organizations to gain access to shared data while protecting their identity attributes, even to the Access Control Manager. The (pseudo-)anonymity of user identities is achieved by relying on *identity token*s, rather than identity attribute submitted in clear. Identity tokens are issued to users by the Identity Manager and consist of structured information. Indeed, each of them contains the pseudonym of the user, the identity attribute name, and the semantically secure Pedersen commitment [6] of the user's identity attribute value, signed by the Identity Manager and by the user itself.

Fine-grained Access Control. The system supports the specification of access control policies in terms of conditions on users' identity attributes. The policy enforcement is based on a cryptographic approach [7] which ensures privacy-preserving document broadcast and efficient key management. Based on this approach, the Access Control Manager encrypts shared data using symmetric encryption and delivers only a secret to reconstruct the key to user (according to the authorizations) in an oblivious fashion using OCBE protocols [8]. Users will use the secret to reconstruct the key from the encrypted data and, hence, access the data.

Integrity of Policy Evaluation. The system ensures the integrity of the Identity and Access Control Manager components by proposing an innovative interaction between blockchain and Intel SGX technologies.

Specifically, blockchain is a distributed ledger of transactions that cannot be modified. Transactions can represent either data or computation on data; the latter corresponds to execution of *smart contract*s, like in Ethereum [9]. The integrity of data and smart contract code stored on the blockchain is guaranteed by so-called *miners* via adequate consensus mechanisms. However, blockchain data are publicly visible to anyone, even attackers. Instead, Intel SGX is a trusted execution environment that protects the execution of secure-sensitive code from malicious software. This is achieved by running the code within the SGX *enclave* and by *attesting* to a remote application that is interacting with a legitimate code.

Therefore, to ensure the integrity of policy evaluation, our system exploits, on the one hand, blockchain to store identity tokens and access control policies, and, on the other hand, Intel SGX to deploy the Identity and Access Control Manager. The usage of the Intel SGX to place and run the code of the Manager is fundamental because data and code are public once stored on the blockchain; an attacker could learn from the Manager code how to gain unauthorized access to federated data. Additionally, running the Pedersen commitment scheme and OCBE protocols would be too computationally expensive for blockchain smart contracts.

3. System Architecture

The system architecture is designed with the goal of efficiently storing data and executing code and to preserve the integrity of both. The identity attributes and the access control policies are stored via smart contracts on a blockchain, while encrypted federated data are stored off-chain. The private and computationally intensive cryptographic policy enforcement protocol is also executed off-chain. To preserve its integrity it is implemented by means of Intel SGX-based applications [10]. These applications consists of two parts: an untrusted part that starts the SGX enclave and interacts with remote applications and the blockchain, and

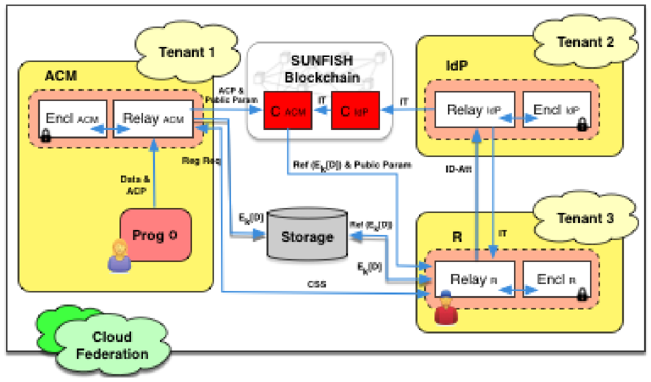

Figure 2. System architecture

a trusted part running within the enclave that implements the cryptographic operations used in the protocol [11]. In what follows we describe the main entities of the system architecture and the interactions among them.

3.1. Entities

Data Owner is a federated organization who is willing to share data.
Data Requester is a user in the cloud federation who is willing to access the shared data.
Data Owner Program (O) is a simple web-based application which provides an interface for the Data Owner to share data and define the access control policies that govern access to such data.
Identity Provider Application (IdP) is an SGX-based application which consists of three components (as shown in Figure 2):

- $Encl_{IdP}$ is a program running within the SGX enclave which receives identity attributes from *Data Requester* and then generates identity tokens.
- $Relay_{IdP}$ is a user-space application which provides network connectivity from the $Encl_{IdP}$ to the $\mathcal{C}_{IdP}$ and the *Data Requester Application*. The interface to the *Data Requester Application* allows the application to verify the attestation for $Encl_{IdP}$ and forwards to the $Encl_{IdP}$ the *Data Requester*'s identity attributes and its request for issuance of identity tokens. The blockchain interface consists of an Ethereum client *Geth* and a wallet $\mathcal{W}_{IdP}$, that allow to send a transaction to the contract $\mathcal{C}_{IdP}$ to store the identity tokens generated by $Encl_{IdP}$.
- $\mathcal{C}_{IdP}$ is the smart contract storing the identity tokens generated by $Encl_{IdP}$.

Access Control Manager Application (ACM). Similarly to the *Identity Provider Application*, ACM is a SGX-enabled application which includes the following components:

- $Encl_{ACM}$ is the program running in the SGX enclave responsible for encrypting the federate data using symmetric key encryption. It also generates for each attribute condition in the access control policy a special secret called *Conditional Subscription Secret* (CSS), which is used to reconstruct the key to decrypt the federated data. $Encl_{ACM}$ obliviously delivers the CSSs to *Data Requester* running an OCBE protocol with the *Data Requester Application*'s $Encl_R$ via $Relay_{ACM}$. It also keeps a record of all the delivered CSSs.
- $\mathcal{C}_{ACM}$ is a smart contract that stores the access control policies defined by the *Data Owner* and the reference to the encrypted data stored off-chain.
- $Relay_{ACM}$ provides an interface from $Encl_{ACM}$ to $\mathcal{C}_{ACM}$. The interface consists of an Ethereum client *Geth* and a wallet $\mathcal{W}_{ACM}$ to execute $\mathcal{C}_{ACM}$ and store the access control policies and references to encrypted data on-chain. It also offers network connectivity to $Encl_{ACM}$ to forward the encrypted data to the off-chain storage and to securely deliver CSSs to *Data Requester Application*.

Data Requester Application (R) is an SGX-based application. It consists of:

- $Encl_R$ program running in the enclave that supports different steps of the enforcement protocol. First, it securely signs the identity tokens issued by the *Identity Provider (IdP) Application*. It engages in a OCBE protocol with ACM to receive the CCSs, and reconstructs the key based on the received CSSs. Last, it retrieves the federate data from the off-chain storage and decrypts them with the reconstructed key.
- $Relay_R$ provides an interface from the *Data Requester* to $Encl_{ACM}$ via $Relay_{ACM}$ and an interface from the $Encl_R$ to the off-chain storage. The former interface allows a *Data Requester* to request access to federated data to $Encl_{ACM}$ via $Relay_{ACM}$ and provide the network connectivity to run the OCBE protocols with $Encl_{ACM}$. The latter interface allows the *Data Requester* to retrieve the encrypted data from the off-chain storage and pass them to $Encl_R$ for decryption.

Off-chain Storage is a decentralized distributed hash-table (or DHT) that stores the data encrypted by $Encl_{ACM}$. Data is accessible through a reference stored on the blockchain.

3.2. The Proposed Protocol

Here we provide an overview on the main phases of the privacy-preserving access control protocol. In what follows, to improve readability we denote with *Identity Provider*, *Access Control Manager*, and *Data Requester* the applications introduced in the previous section.
Setup. A *Data Owner* defines a set of access control policies *ACP*s that specify which data D_i, users in the federation are authorized to access based on their identity attributes. The *Data Owner* sends these policies along with data D_i to the *Access Control Manager*. The *Access Control Manager*

encrypts D_i with a symmetric key K and stores it in the off-chain storage. Then, it stores the access control policy applying to D_i and the reference to the encrypted data $\mathcal{E}_K[D_i]$ on the blockchain by executing the smart contract $\mathcal{C}_{ACM}$.

Identity Token Issuance. A *Data Requester* provides its identity attributes to *Identity Provider*, which issues an identity token *IT* for each identity attribute. *IT* consists of a pseudonym for uniquely identifying the user in the system, the name of the identity attribute, and the Pedersen commitment for the identity attribute value. Each identity token is digitally signed by the *Identity Provider* and the *Data Requester* and then stored on chain via the smart contract $\mathcal{C}_{IdP}$. The signature of the *Identity Provider* allows to certify the validity of the identity token. While the signature of the *Data Requester* is necessary because once stored on the blockchain, the identity tokens are publicly available. Without the signature, another user could present the identity tokens to the *Access Control Manager* and gain unauthorized access to federated data.

Identity Token Registration. Once obtained the identity tokens from the *Identity Provider*, the *Data Requester* can request access to federated data to the *Access Control Manager*. Then, in order to retrieve the key to decrypt the data, the *Data Requester* has to present to the *Access Control Manager* the identity tokens whose name matches the attribute conditions in the access control policy governing access to the data. Then, the *Access Control Manager* sends to the *Data Requester* a conditional subscription secret (CSS) for each identity attribute name in the federated organization's access control policy condition matching the user's identity token name. CSSs are used by the user to derive the keys to decrypt the shared data for which they satisfy the access control policy. The Access Control Manager delivers the CSSs to the user running OCBE protocols [8] with the user. OCBE protocols ensure that a user can obtain a CSS if and only if the user's committed identity attribute value contained in the identity token satisfies the matching condition in the federated organization's access control policy, and without the Access Control Manager learning the identity attribute value.

Data Access. *Data Requester* uses Ref($\mathcal{E}_K[D_i]$) to retrieve the encrypted data from off-chain storage. Then, it uses the CSSs to reconstruct the key *K* as in [7]. Once reconstructed the key *K*, the *Data Requester* decrypts the data.

4. Conclusion and Future Work

In this paper, we presented a novel identity and access control system for federated clouds, which is part of FaaS, a software service to create cloud federations. The system enforces attributes-based access control policies by means of a cryptographic protocol that preserves the privacy of users' identity attributes. The system also guarantees the integrity of the identity tokens issued to the users in the cloud federation and the access control policies protecting access to the federated data. The integrity is ensured by storing them on the blockchain via smart contracts. Similarly, the integrity of the cryptographic policy enforcement protocol is guaranteed by running the protocol within the Intel SGX's enclave.

We are planning to extend the cryptographic access control enforcement protocol with an approach that considers the risk associated with a particular user accessing a particular piece of information. The approach will rely upon the cryptocurrency system of the blockchain to price information accessed by users; the price will quantify the risk for a federated organization to disclose the data with a particular user. The price will be used to reward federated organizations for sharing their personal data and to penalize those users from other member organizations that misuse it.

We will also develop a proof-of-concept prototype using Ethereum blockchain and Intel SGX trusted execution environment and conduct an extensive evaluation of the protocol with respect to performance and cost of execution.

Acknowledgment

This work has been supported by the EU H2020 Program under the SUNFISH project, grant agreement N. 644666.

References

[1] A. Almutairi, M. Sarfraz, S. Basalamah, W. Aref, and A. Ghafoor, "A distributed access control architecture for cloud computing," *IEEE Softw.*, vol. 29, no. 2, pp. 36–44, Mar. 2012.

[2] B. Suzic, B. Prünster, D. Ziegler, A. Marsalek, and A. Reiter, "Balancing utility and security: Securing cloud federations of public entities," in *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, 2016, pp. 943–961.

[3] M. Singhal, S. Chandrasekhar, T. Ge, R. Sandhu, R. Krishnan, G. J. Ahn, and E. Bertino, "Collaboration in multicloud computing environments: Framework and security issues," *Computer*, vol. 46, no. 2, pp. 76–84, 2013.

[4] F. P. Schiavo, V. Sassone, L. Nicoletti, and A. Margheri (Eds.), "Faas: Federation-as-a-service," *CoRR*, vol. abs/1612.03937, 2016.

[5] "SecUre iNFormatIon SHaring in federated heterogeneous private clouds (SUNFISH)," Accessed on 16 January, 2017, http://www.sunfishproject.eu/.

[6] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Annual International Cryptology Conference*. Springer, 1991, pp. 129–140.

[7] N. Shang, M. Nabeel, F. Paci, and E. Bertino, "A privacy-preserving approach to policy-based content dissemination," in *Data Engineering (ICDE), 2010 IEEE 26th International Conference on*. IEEE, 2010, pp. 944–955.

[8] J. Li and N. Li, "OACerts: Oblivious Attribute Certificates," *Dependable and Secure Computing, IEEE Transactions on*, vol. 3, no. 4, pp. 340–352, 2006.

[9] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger, 2014," *Ethereum Project Yellow Paper*, 2014.

[10] IntelCorp., "Intel (r) software guard extensions enclave writer's guide," Tech. Rep., 2015.

[11] F. Zhang, E. Cecchetti, K. Croman, A. Juels, and E. Shi, "Town crier: An authenticated data feed for smart contracts," Cryptology ePrint Archive, Report 2016/168, 2016, http://eprint.iacr.org/2016/168.