XIII International scientific and technical conference "Dynamics of Systems, Mechanisms and Machines"
5-7 November 2019, Omsk, Russia

Attribute-Based Access Control in Service Mesh

Kirill Yu. Ponomarev
Department of Information Security
University of Tyumen
Tyumen, Russian Federation
drmckay-kirill@yandex.ru

***Abstract*—Modern cloud applications can consist of hundreds of services with thousands of instances. In order to solve the problems of interservice interaction in this highly dynamic environment, an additional software infrastructure layer called service mesh is introduced. This layer provides a single point of interaction with the network for each service. Service mesh mechanisms are responsible for: load balancing, processing of network requests, service discovery, authentication, authorization, etc. However, the following questions arise: complex key management, fine-grained access control at the application level, confidentiality of data and many-to-many communications. It is possible to solve these problems with Attribute-based encryption (ABE) methods. This paper presents an abstract model of a service mesh and a protocol for interservice communications, which uses ABE for authorization and confidentiality of the messages.**

Keywords—access control, service mesh, attribute-based encryption

I. INTRODUCTION

Modern cloud applications can consist of hundreds of services with thousands of instances. Mechanisms and methods of interservice communication in such a dynamic environment are very complex. To solve this problem, it was proposed to introduce an additional software layer that provides safe, fast and reliable communication between services. This model is called service mesh and it's a set of proxies to which services should connect to completely abstract the network. Every proxy provides a single and unified entry point for the service and helps with reliable and safe delivery of requests over the network. In practice, this model is implemented according to the sidecar pattern — each element of the service mesh is implemented as a separate container, which acts as a proxy for the service. The service mesh operates above the transport and network layers of the OSI model and assumes that the underlying layers are capable to deliver information between nodes, nevertheless they do not support reliable data transfer. Linkerd, Istio, Netramesh - are examples of service mesh solutions.

Briefly describe the functions of the service mesh:

- reliable mechanism for transferring requests between services;
- load balancing and scalability;
- traffic routing between applications;
- service discovery methods;
- traffic encryption, key management, authentication and authorization of users and services.

Due to the fact that the nodes of the service mesh are deployed for each instance of each service, they have hard performance requirements: they should not consume a large amount of computing resources (CPU and RAM) and reduce network bandwidth.

Highly dynamic service mesh environment generates new security problems: complex key management, fine-grained access control at the application level. Additionally, when processing and transferring personal or confidential data, services and applications should store data in encrypted form and manage access to it. In this paper Attribute-based encryption schemes will be used to solve these problems, since they provide an effective mechanism for granting access based on the attributes of the subject [2]. Especially, we will consider protocols of interservice communication for service mesh environment based on ABE.

II. RELATED WORK

A. Attribute-based encryption

Attribute-based encryption is a public key cryptosystem. It's genesis can be traced back to Identity-based encryption, proposed by Shamir in 1984 [4]. The idea of this method is to use any generic public string as a public key such as an email address. To solve this problem a master private-key generator (PKG) was proposed that is responsible for providing the decryption keys that are tied to a generic identity. In 2001 Boneh and Franklin [5] proposed an implementation of IBE scheme and proved it secure under the chosen-ciphertext attack. Their approach was based on a special form of bilinear mapping - Weil pairings. This mathematical construction provides a method to map random public string (for example, user's identity) to a cryptographic public key and generate suitable private key used to decrypt messages encrypted with the corresponding public key. In 2004 Amit Sahai and Brent Waters proposed a Fuzzy IBE scheme [6], which became the prototype of ABE. In FIBE a user with a private key and a set of public parameters w is able to decrypt the ciphertext generated using another set of public parameters $w`$ only if parameters interception exceeds special threshold $d$: $|w \cap w`| \geq d$.

As a PKG in ABE a special entity is used - Attribute Authority (*AA*), which is responsible for generating public and private keys, key distribution and management, verification of attributes received from elements requesting a secret key. The most effective are Ciphertext-policy Attribute-based encryption schemes (CP-ABE) that implement the next functions:

- *Setup()→(pars, msk)*: *AA* generates public parameters and secret master key;

- *KeyGeneration(pars, msk, A)* $\rightarrow SK_A$: *AA* generates new secret key using public parameters, secret master key and attributes of requesting subject;
- *Encrypt(pars, P, M)* $\rightarrow CT$: message sender encrypts message and generates new ciphertext using public parameters, access structure and plaintext;
- *Decrypt(pars, CT, P,* SK_A*)* $\rightarrow M$: message receiver decrypts message using public parameters, ciphertext, access structure and their own secret key, it's possible only if the attributes of the secret key *A* satisfy access structure *P*.

In the literature there are many ABE schemes with specific properties: multi-authority, confidential access structure, constant-size ciphertext [7, 8], dynamic attribute universe, etc. In the case of service mesh environment constant-size ciphertext property is the most important, it means that the size of the ciphertext is independent of the total number of attributes. Since the services actively interact with each other, the size of the transmitted messages largely determines the network status.

B. Service communication

In the publication [1] the evolution of microservice architecture is considered - an approach to building applications in which instead of a single monolith many isolated services operate, each of them is responsible for its own bounded context of business logic. [9] describes the main security features for microservice architecture: JWT tokens (Json Web Tokens) and TLS for authentication of services and users, OAuth2 protocol and Gateway design pattern for outside access control, XACML protocol (eXtensible Access Control Markup Language) to describe access policies. It is important to note that certificate-based mechanisms require complex processes for certificates distributing and verifying, and XACML requires a decentralized architecture for verifying and distributing access policies; moreover, access policies should be verified in every request that comes to the service. The service mesh model was invented to solve the mentioned communication problems in microservice systems.

Service mesh is also applicable at the application level of Internet of Things (IoT) networks: for example, to provide interoperability between services in software platforms and cloud applications. An interesting approach to building IoT systems was described in [2]: the idea was to represent each node (actuators, sensors, edge elements, cloud storage, etc.) as a separate service that supports a specific program interface and generates information about events in a unified format. In the described model each node is representable as a virtual object for all others, and heterogeneous devices provide a unified interface for interaction. Thus, the authors transfer the microservice approach to IoT systems, trying to use existing methods and protocols. For authorization between services they proposed to use an attribute access model based on ABE [3]. This solution provides fine-grained access control, data confidentiality, flexible access policies.

III. Motivation

The aim of this paper is to develop service-to-service communication protocols with attribute-based encryption methods. A distinctive feature of the presented mechanisms is that they are considered in the context of a service mesh environment.

IV. Access control in service mesh

A. Service mesh model

Service mesh consists of two logical levels: the Control Plane and the Data Plane. The first one is responsible for administrative procedures: updating software, distributing configuration, service discovery, credential management. Each sidecar-proxy interacts with the control level to update supporting libraries and manage infrastructure configuration. The Data layer consists of the application layer services and their corresponding sidecar intermediaries. Each service is provided with a lightweight single point of interaction with other components - the intermediary proxy, which is responsible for the reliable transmission of data and interaction through network. This approach greatly simplifies the development, testing, scaling, continuous integration and continuous delivery processes. It is assumed that the service and its proxy intermediary operate in an isolated network segment - for example, inside the Kubernetes pod, therefore, the protection of the communication channel between them is either absent or based on knowledge of a common key, and therefore is not considered further.

Attribute authority (also known as the attribute verification center and the key generation center) is located at the management level. *AA* is responsible for the distribution of public and private keys to the consumers (for example, services). Consider the next scenario: service *A* collects data from the temperature sensors in the building and service *B* analyzes this information. Let *A* has received secret key for the next set of attributes: "sensor", "temperature", "building 1". And service *B* has received secret key with the next attributes: "temperature", "analytics", "heating control". Then service *A* could encrypt the message for service *B* under access structure: "temperature" AND "heating control". There is an example of access structure from the medical sphere: 'doctor", "cardiology", "building 3", "higher qualification". Attribute-based access rules can be not just a sequential enumeration of the required attributes (AND-gate access structure), but also threshold access structure or boolean expressions, for example "sensor" AND

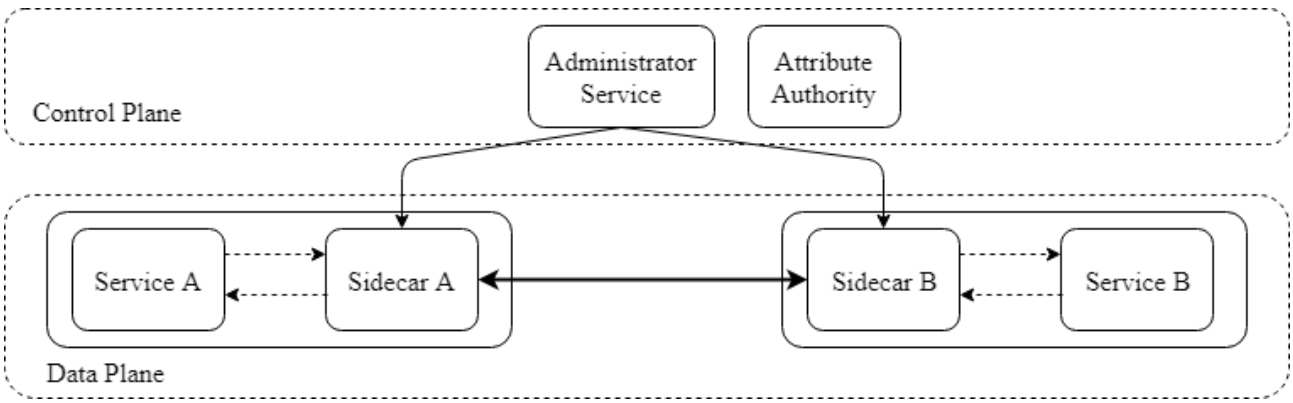

Fig. 1. Service mesh

"temperature" AND "building 1" OR "building 2". In this paper ABE is used to protect communications between services in the service mesh.

B. Attribute-based handshake for services communication

In the presented model the distribution of access policies is not a centralized process: each service independently determines the access policies for its API and passes this configuration to its sidecar-proxy. Proxy is responsible for checking attributes of the sender for compliance with the access policy corresponding to the requesting resource or API method. To provide this mechanism in cloud and service environments we will present a handshake protocol based on ABE technique.

The described protocol operates at the application level of the OSI model. It is functions are mutual authentication between services, issuing an identifier or token (for example, in JWT format) so that subscribers can identify a communication session. It is assumed that at the lower level there is a secure communication channel with support for a shared secret key and message integrity checking (for example, TLS).

Further we will describe attribute-based handshake protocol. Notations are presented in Table 1. At the first step, the service (more precisely, its proxy) *A* sends a request to the service *B*, its own set of attributes and a random number encrypted under the attributes of the receiving side.

$$A \rightarrow B: data, attr_A, CT_1 = E^{ABE}_{attr_B}(R_1)$$

TABLE I. NOTATIONS

Symbol	Notation
$attr_A$, $attr_B$	Service attributes
$attr_{AS}$	Access structure for the requesting API
R_i	Cryptographic nonce (random number)
$ID_{session}$	Session identificator
CT	Ciphertext produced by *Encrypt* function
E^{ABE}	Algorithm of Attribute-based encryption
H	Hash function

Next the proxy of service *B* generates a new session with the *waiting* status, decrypts a nonce and sends a new set of ciphertexts in order to verify access. Note access structure is transmitted in plaintext format, however that it can be dangerous in some applications, for example in medicine. So ABE schemes with confidential access structure may be required to protect $attr_{AS}$ against attackers.

$$B \rightarrow A: ID_{session}, R_1, CT_2 = E^{ABE}_{attr_A}(R_2),$$
$$CT_3 = E^{ABE}_{attr_{AS}}(R_3), H(R_1||R_2||R_3)$$

The proxy of service *A* checks the received nonce with the one sent to authenticate service *B*. Next, it remembers the session and decrypts the values.

$$A \rightarrow B: ID_{session}, H(R_2||R_3)$$

Service *B* authenticates *A* by checking the hash with the values of the nonce sent, and in the positive case, changes the status of the session. Thus, both subscribers performed the mutual authentication procedure and completed two handshake stages using ABE.

The presented version of the handshake protocol can be improved by the Diffie-Hellman algorithm for generating shared secret key. Let *P* - big prime number, α - generator of multiplicative group $Z_p^*=\{1, ..., p-1\}$, *X* and *Y* - random numbers from Z_p^*, *SK* - shared private key.

$$A \rightarrow B: data, attr_A, CT_1 = E^{ABE}_{attr_B}(R_1)$$

$$B \rightarrow A: ID_{session}, R_1, CT_2 = E^{ABE}_{attr_A}(R_2, \alpha^X),$$
$$CT_3 = E^{ABE}_{attr_{AS}}(R_3), H(R_1||R_2||R_3)$$

$$A \rightarrow B: ID_{session}, H(R_2||R_3), E^{ABE}_{attr_B}(\alpha^Y)$$

$$SK = (\alpha^X)^Y mod\ P = (\alpha^Y)^X mod\ P$$

During its lifetime, the proxy maintains a session table containing: session identifier, session status, set of identifying attributes of the subscriber, set of recognized access structures, start date, shared secret key. Proxy sequentially checks for every incoming request: the identifier in the request, session statuses. If proxy will recognize that the attribute sets from the table do not match the access structure of the requested resource, then handshake protocol should restart to verify access. For example, requested URL refers to a resource with access structure which does not presented in the session table. Synchronization of the session tables between different instances of the same service is an open question.

The described protocol was analyzed using the AVISPA - a software tool for automated validation of Internet security protocols and applications. Validation was performed in order to verify mutual authentication of the parties and protection of the secret key. HLPSL source code is publicly available on github[1].

V. CONCLUSION

This paper presented an approach to protecting communications in the service mesh environment based on ABE techniques. It simplifies key management processes: services shouldn't firstly exchange their public keys. Specifically attributes acts as a public keys in attribute-based access policies. ABE also provides an opportunity for implementing flexible and fine-grained access control mechanisms at the application level. An attribute-based handshake protocol has been described. It allows services to perform authentication using attributes and generate a shared secret key using the Diffie-Hellman algorithm.

However, in order to finally and fully describe the service mesh model using ABE, it is necessary to answer the following questions: how to perform the safe transfer of keys from the attribute authority to services, what to do in case of a compromise of the attribute authority, how to effectively revoke attributes and secret keys?

[1] https://github.com/drmckay-kirill/attribute_based_protocols.git

REFERENCES

[1] P. Jamshidi., C. Pahl., N. C. Mendonça, J. Lewis, and S. Tilkov, Microservices: The journey so far and challenges ahead, IEEE Software, vol. 35(3), pp. 24–35, 2018.

[2] D. Lu, D. Huang, A. Walenstein, and D. Medhi, A secure microservice framework for IoT, In 2017 IEEE Symposium on Service-Oriented System Engineering, pp. 9–18.

[3] Z. Wang., D. Huang., Y. Zhu, B. Li, and C.J. Chung, "Efficient attribute-based comparable data access control," IEEE Transactions on computers,vol. 64(12), pp. 3430–3443, 2015.

[4] A. Shamir. Identity-based cryptosystems and signature schemes. In Workshop on the theory and application of cryptographic techniques. Springer, Berlin, Heidelberg, pp. 47–53, August 1984.

[5] D. Boneh., and M. Franklin. Identity-based encryption from the Weil pairing. In Annual international cryptology conference. Springer, Berlin, Heidelberg, pp. 213–229, August 2001.

[6] A. Sahai, and B. Waters. Fuzzy identity-based encryption, "Annual International Conference on the Theory and Applications of Cryptographic Techniques," Springer, Berlin, Heidelberg, pp. 457-473, May 2005.

[7] F. Guo., Y. Mu., W. Susilo, D. S. Wong, and V. Varadharajan, "CP-ABE with constant-size keys for lightweight devices," *IEEE transactions on information forensics and security*, vol. 9(5), pp. 763–771, 2014.

[8] X. Li, D. Gu, Y. Ren., N. Ding, & K. Yuan, "Efficient ciphertext-policy attribute based encryption with hidden policy. In International Conference on Internet and Distributed Computing Systems," Springer, Berlin, Heidelberg, pp. 146–159, November 2012.

[9] K. Indrasiri, and P. Siriwardena. Microservices for the enterprise. Apress, Berkeley, 2018.