Zentrum für sichere Informationstechnologie – Austria
Secure Information Technology Center – Austria

A-SIT

Seidlgasse 22 / 9, 1030 Wien
Tel.: (+43 1) 503 19 63–0
Fax: (+43 1) 503 19 63–66

Inffeldgasse 16a, 8010 Graz
Tel.: (+43 316) 873-5514
Fax: (+43 316) 873-5520

http://www.a-sit.at
E-Mail: office@a-sit.at

DVR: 1035461
ZVR: 948166612
UID: ATU60778947

ANONYMOUS SERVICE CONSUMPTION FOR SERVICE COMPOSITION SYSTEMS

Version 1.0 of 01.06.2021
Author – Kevin Theuermann
kevin.theuermann@egiz.gv.at

Service compositions are implemented through the interplay between actors of different organizations. To protect the privacy of a service consumer's identity data when using specific services, it may be necessary to enable an anonymous service consumption for users. Since service compositions comprise multiple composition types using different communication flows, preserving the privacy of service consumers becomes even more challenging.
In this work, we provide a concept, which ensures an anonymous registration at a service composition system, which enables an anonymous service consumption by ensuring both, protection of identity data against the operator running and maintaining the service composition system as well as against service providers involved in a composite service. If a service provider mandatorily requires identity data of a service consumer to fulfil his business logic, a service consumer is able to grant access to the identity data specifically for this service provider. This concept uses advanced cryptographic mechanisms to ensure privacy protection offering high efficiency regarding the performance on devices with limited computational power as well as high usability to achieve an overall acceptance among service consumers. The feasibility of the proposed solution is demonstrated by an implemented proof-of-concept.

Table of Contents

1. Introduction

Service-Oriented Computing (SOC) [16] has introduced a promising paradigm to perform service compositions, which represent the cross-organizational combination of distributed services to enhance reusability and to enable a flexible combination of individual services to create more complex composite services. However, protecting the privacy of identity information of service consumers becomes an increasingly challenging part in service composition systems. Service

consumers may want to anonymously use a specific service included in a composite service, without revealing the identity to the service provider. For instance, let us assume a service providing addiction counselling or a form-service collecting results for surveys. In such cases, service composition systems must enable an untraceable anonymous service consumption to ensure high trust among the users.
In the past, much research was conducted in the field of anonymous authentication ([10], [22], [19]). However, service composition systems raise additional requirements with respect to privacy protection measures, since they must consider several composition types resulting in different communication protocols. Existing research does not focus on the identification of anonymous authentication schemes, especially suitable for service composition systems and simultaneously providing high usability. In addition, controlling a service consumer's identity data exposure to a specific service provider involved in a composite service is a challenging task. Identity data exposure may be necessary if a service provider requires information about the identity of user in order to perform a proper computation of the provided service [15]. In these cases, the use of standard encryption techniques does not offer a practical approach to protect privacy [11].

In this work, we propose a concept enabling an anonymous service consumption for users of composite services provided by service composition systems. Our main contributions are:

- We propose a system architecture that provides a mechanism for anonymous authentication of service consumers at a service composition system.
- The privacy of identity data of users during service consumption is preserved in a way, that components of the service composition system cannot read any sensitive data and only authorized service providers can reveal the identity data of service consumers.
- Additionally, service consumers can decide to reveal their identity data only to specific service providers involved in a composite service, which guarantees full control over their sensitive data.
- After registration of a service consumer, our system provides a simplified authentication mechanism that requires no user-interaction and, thus, enhances usability. This approach ensures single-sign-on with increased security.
- Operators of service composition systems can dynamically define the number of accepted simplified authentications, based on the level of assurance provided by the initially performed registration. With this number of simplified authentications, operators of service composition systems can define the period until an initial full authentication including user-interaction is required.

2. Anonymous Authentication Schemes

The need for anonymous authentication for specific use cases has already been recognized in the past. This section provides some of the most common anonymous authentication schemes.

2.1 Group Signatures:

Some of the research conducted with respect to anonymous authentication use group signature schemes, which allow members of a group to sign a message on behalf of this group. The signature verification can be conducted using a single group public key, without revealing the identity of the signer [6]. Jensen et al. [12] introduced a concept to provide traceable verifiable anonymous credentials based on group signatures.

They use a third party broker to issue these anonymous credentials, which enable a service provider to authenticate service requests, without the ability to distinguish, which consumer has actually signed a message. Group signatures are anonymous in that, no one, with the sole exception of a designated group manager, can reveal the identity of the signer [1]. In our work, we do not use group signatures, since we only want the client to be able to reveal his own identity data, excluding a trusted third party considered as group manager in order to enable full control for a service consumer about his personal data.

2.2 Blind Signatures:

Blind signatures schemes allow a user to obtain a message signed by a trusted third party without revealing the message [17]. Djellalbia et al. [8] proposes a concept to ensure the privacy of a user's identity in an e-health cloud applying an anonymous authentication scheme based on RSA blind signatures. In this concept, a trusted entity issues tickets considered as anonymous credentials to users to enable an anonymous service consumption. In our concept, we include an independent trusted third party, which is authorized to read the identity of service consumers and issues privacy-preserving confirmation tokens to the client, indicating a successful authentication. The client uses this token for registration and anonymous service consumption at the service composition system.

2.3 Ring- and Threshold Signatures:

If a system does not want to fully trust one specific third party, there exist different alternatives to guarantee anonymous authentication using ring signatures [18] or threshold signatures [5]. Ring signatures merge several possible senders into one signature. This type of signatures enables a service consumer to sign a message in a way, that only a ring of possible signers is identified, without revealing the explicit signer of this ring, who actually generated the signature. If the number of participating signers does not match or exceed a specific threshold, the signature is invalid. These signature types are suitable to share the trust of a single third party to several entities and can be applied for decentralization of the authentication system.

2.4 Zero Knowledge Proof:

Goldwasser et. al introduced the fundamental notion for Zero Knowledge Proofs (ZKP) [9], which represent an interactive identification protocol that enables a prover to prove his identity polynomially many times to a verifier without allowing the verifier to misrepresent himself as a prover to someone else. ZKP are especially suitable for systems with limited processing power, since they provide a lightweight communication.

2.5 Hash-Chains:

Hash-Chains (HC) represent a mechanism to produce various One-Time-Passwords (OTP) by applying a hash-function repeatedly on a secret. Since the sender transmits a hash-chain element based on the input of a secret, he delivers a proof without revealing the secret. The recipient can verify the validity of the proof, by applying a hash-function on the received proof and comparing it with the subsequent element of a pre-generated hash-chain.
If these values match, the transmitted proof is valid. Hence, hash-chain authentication can be considered as a type of ZKP. They were first introduced by Lamport [13] and aim to safeguard password schemes from eavesdrop and replay attacks.

3. Anonymity for Service Consumers

This section first describes our defined security objectives to provide an anonymous service consumption at a service composition system. Furthermore, this section describes our defined mechanisms including the applied cryptographic methods needed to achieve these security objectives.

3.1 Security Objectives

This paragraph summarizes our defined security objectives, which ensure the protection of the privacy of identity data of service consumers.
Security Objectives:

1. **End-to-end confidentiality** for transmitted identity data: In our concept, we want to preserve the privacy of identity data against both, the service registry running and maintaining the service composition system as well as service providers, who provide their individual services for composition. This is necessary to ensure full anonymity of a service consumer. We consider both service composition types:
    a. Service orchestrations, which use a centralized middleware that takes over the service call coordination.
    b. Service choreographies using a decentralized approach, in which services communicate directly.
2. **Full control over identity data** for service consumers: This work aims to ensure full control for a service consumer over his identity data, which is necessary to achieve a high level of trust.
3. **Increased usability and privacy** through security policies: We want to enable service consumers to decide, whether they want to define general conditions under which a service provider is able to reveal his identity data, or if they want to explicitly grant access for specific service providers. Thus, a service consumer can determine, if he accepts additional user interaction needed to grant access to their identity data, or if he wants to define a set of conditions that have to be fulfilled by authorized service providers.
4. Anonymous authentication offering **single-sign-on (SSO)** with increased security: To preserve the identity of service consumers we want to provide an anonymous authentication mechanism at the service composition system. We also want to include a single sign-on mechanism that prevents a service consumer to necessarily perform the whole authentication process every time he uses a service.
5. **Flexible definition of duration for SSO** by the operator of a service composition system: We want the service registry to be able to define the duration of a valid single-sign-on session for registered service consumers, based on the security level provided by the initially performed authentication. This empowers the service registry to define conditions, how long a session is valid until a service consumer has to register at the service composition system again.

3.2 Anonymous Registration for Service Consumers

In our concept, we provide a mechanism to anonymously register at a service composition system and to anonymously consume composite services provided by this system. This is necessary to preserve the privacy of a service consumer's identity data against both, components of the service registry maintaining and providing the service composition system as well as against service providers, who provide their individual services.
To guarantee an anonymous registration for service consumers at the service registry, we include an independent trusted third party (TTP), which is authorized to read the identity of a service consumer and provides means to perform an authentication based on a high security. A service consumer first has to authenticate at this TTP, to obtain a confirmation token indicating a successful authentication. This confirmation token does not contain any information about the service consumer's identity. Afterwards, the service consumer transmits this confirmation token to the service registry, to register at the service composition system. To increase the usability by preventing a service consumer to authenticate at the TTP every time he wants to use a composite service at the service composition system, the service registry issues a hash-chain to the service consumer, which ensures an anonymous service consumption.

3.2.1 Hash-Chain Authentication

Hash-Chains (HC) offer the possibility to produce various One-Time-Passwords (OTP) by applying a hash-function repeatedly on a secret.

Definition (Hash-Chain). Authentication via hash-chains requires steps for its generation and verification. The following explanation recalls the syntactic definition of a hash-chain suggested by

Bicacki and Baykal [4].

$$\text{Generate: } Hash^{N-i}(secret) \rightarrow (P_i)$$

To generate a hash-chain, a hash-function is applied N times, where N represents the number of allowed operations for authentication. Thus, the number of performed repetitions of a hash-function represents the number of generated OTP.

The first OTP is produced by applying the hash-function N times:

$$P_0 = Hash^N(secret)$$

The second OTP is generated by applying the hash-function N − 1 times:

$$P_1 = Hash^{N-1}(secret)$$

Hence, this results in the following general formula:

$$P_i = Hash^{N-i}(secret)$$

$$\text{Verify: } Pi = Hash(Hash^{N-i-1}(secret)) = Hash(P_{i+1})$$

The authentication via hash-chain requires a sender to transmit the OTP P0 to the authentication server, which performs the verification algorithm to validate the OTP.

3.3 Anonymous Registration for Service Consumers

In our concept, we protect the privacy of a service consumer's identity data against service providers involved in processing a composite service. Service consumers can choose between two security protocols to protect the privacy of their identity data. The choice of the desired protocol is determined during the registration at the service composition system:

- **Security protocol 1** enables a service consumer to determine a set of authorized actors, who have access to their identity data, by defining an access structure through a combination of attributes. Only service providers who can fulfil this access structure are able to read the client's identity data. This approach only requires a service consumer to define an access structure to his identity data once, which offers increased usability since it does not require any further user interaction.
- **Security protocol 2** enables a service consumer to grant access to his identity data for specific service providers. This approach requires a service consumer to authorize a service provider during processing the service, if this service provider requests the client's identity data the first time. If the service consumer grants access to the identity data for a service provider, this authorization is registered in order to prevent necessary user interactions to approve access for subsequent service usages.

3.3.1 Security Protocol 1: Attribute-Based Encryption

To implement the functionality defined by security protocol 1, we use a collaborative key management protocol for Ciphertext Policy Attribute-Based Encryption (CKM-CPABE), which was introduced by Lin et al. [14]. Since this cryptographic method represents an extension of Ciphertext-Policy Attribute-Based encryption (CP-ABE), the following paragraph briefly explains the principle of this cryptographic primitive.

Sahai and Waters [20] first introduced Attribute Based Encryption (ABE), which represents an encryption mechanism that goes beyond traditional one-to-one public key encryption types. Instead of using the public key of a party to encrypt a message, a sender only has to know the attributes of

the receiver in an ABE system. This way, the sender can encrypt a message, which is only decryptable by receivers owning the specified attributes. The main advantage over conventional one-to-one encryption is that ABE enables one-to-many encryption [7].

As we want the client to be able to determine the access structure to his identity data, we use Ciphertext Policy Attribute-Based Encryption (CP-ABE) [3] for our concept. This enables a client to define the access structure and attaching it to the ciphertext during the encryption process. A trusted key authority assigns a set of attributes to users when issuing the key material to them. When encrypting a message, the user specifies an associated access structure over attributes of a given attribute universe U. The attribute universe represents the collection of all available attributes within a system. The access structures are defined by attributes using conjunctions, disjunctions and (k,n) threshold gates, which means that k out of n attributes must be present. For example, let us assume an attribute universe $U = \{A,B,C,D,E\}$, a private key PK_1 to the attributes $\{A,B,C\}$ and a second private key PK_2 to the attributes {D,E}. A ciphertext that was encrypted with the policy $(A \wedge B) \vee (A \wedge E)$ would only enable the PK_1 to decrypt it.

Collaborative Key Management Protocol for CP-ABE: In the concept for CKM-CP-ABE, an actor's private key is split into three key parts: 1) for the key authority (KA), 2) for a re-encryption server (RS), and 3) for the client (CL). Using these key parts, the key authority and re-encryption server can reencrypt and partially decrypt a ciphertext for the client, thereby performing the most expensive calculations. Finally, the client applies her private key part to decrypt the prepared ciphertext, which usually requires less computational effort. The model also allows for updates of the key material by using attribute group keys, which represent a collection of actors sharing the same attribute. If an attribute has been revoked from an actor, this actor is removed from the respective attribute group. The principle of CKM-CP-ABE is illustrated by Figure 1.

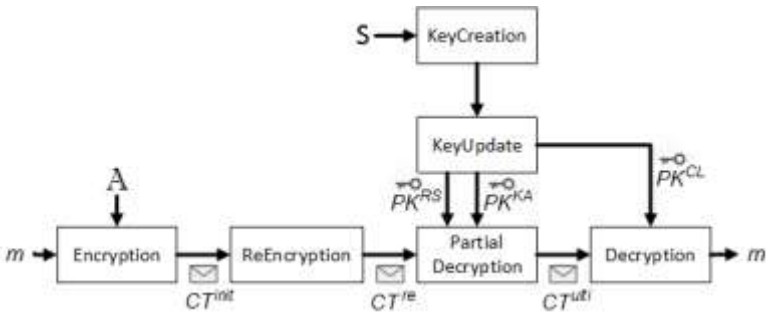

Figure 1: Collaborative Key Management Protocol for Ciphertext Policy Attribute-based Encryption

The reencryption algorithm in this concept takes all attribute group keys and a ciphertext as input and updates decryption keys in case of a revocation. Below, we recall the cryptographic definitions from Ziegler et al. [23], which we slightly adapted for readability. We omit the public parameters $params \leftarrow \{paramsbase,paramsKA,paramsRS\}$ as input for the algorithms.

The CKM-CP-ABE consists of the following algorithms:

- $BaseSetup(1^{\kappa},S) \rightarrow params_{base}$: This algorithm takes a security parameter κ and an attribute set S as input and outputs the base system parameters $params_{base}$.
- $KASetup(params_{base}) \rightarrow params_{KA}$: This algorithm takes the base system parameters and outputs the key authority parameters $params_{KA}$.
- $RESetup(params_{base}) \rightarrow params_{RS}$: This algorithm takes the base system parameters as input and outputs the reencryption parameters $params_{RS}$.
- $KeyCreation(S) \rightarrow IK^{KA}$: This algorithm takes a client's attribute set S as input and generates the initial key IK^{KA}.
- $KeyUpdate(IK^{KA}) \rightarrow (PK^{KA},PK^{RS},PK^{CL})$: Given the initial key IK^{KA} of a user, this algorithm outputs an updated version of a user's three private key parts for the corresponding actors, $(PK^{KA},PK^{RS},PK^{CL})$.

- $\text{Encryption}(A, m) \rightarrow CT^{init}$: To encrypt a message, this algorithm takes an access structure A as well as a message m as input. It outputs an initial ciphertext CT^{init}.
- $\text{ReEncryption}(CT^{init}) \rightarrow CT^{re}$: This algorithm, given an initial ciphertext CT^{init} and a collection of attribute group keys, generates a reencrypted ciphertext CT^{re} .
- $\text{PartialDecryption}(CT^{re}, PK^{KA}, PK^{RS}) \rightarrow CT^{ulti}$: Given a reencrypted ciphertext CT^{re} as well as the private key parts of the key authority PK^{KA} and the reencryption server PK^{RS}, this algorithm partially decrypts CT^{re} and outputs the final ciphertext CT^{ulti}.
- $\text{Decryption}(CT^{ulti}, PK^{CL}) \rightarrow m$: The decryption algorithm takes the final ciphertext CT^{ulti} and the private key part of the client PK^{CL} as input. This algorithm outputs the plaintext message m.

3.3.2 Security Protocol 2: Proxy Reencryption

To provide the functionality defined by security protocol 2 we use proxy reencryption [24]. This cryptographic system allows a proxy to convert a ciphertext CT_1 that has been encrypted for one party, into a ciphertext CT_{1-2} that can be decrypted by another party. For service compositions including a middleware, this primitive is able to ensure privacy of the identity data of service consumers, because a client can decrypt a message using his own public key, whereafter the middleware reencrypts this ciphertext into another, which is decryptable by a specific service provider involved in a composite service. The following paragraph recalls the syntactic definition of proxy reencryption suggested by Ateniese et al. [2].

Definition (Proxy Reencryption). The scheme of proxy reencryption consists of several algorithms (Setup, KeyGen, Enc, Dec, ReKeyGen, ReEnc), which are defined as follows:

- $\text{Setup}(1^{\kappa}) \rightarrow (\text{params})$: This algorithm is performed by a trusted third party. The function takes a security parameter κ as input and generates the public parameters params.
- $\text{KeyGen}(\text{params}) \rightarrow (SK, PK)$: On input of the public parameters params, all parties use this algorithm to generate an asymmetric keypair (SK, PK).
- $\text{Encryption}j\ (PK, m) \rightarrow (CT)$: On input of a public key PK of j and a message m, this function outputs a ciphertext CT.
- $\text{Decryption}^{j}(SK, CT) \rightarrow (m)$: On input of a secret key SK of j and a ciphertext CT encrypted under j, this function outputs a message m.
- $\text{ReKeyGen}(SK_A, PK_B) \rightarrow (RK_{A \rightarrow B})$: On input of a secret key SK_A and a public key PK_B, this function outputs a reencryption key $RK_{A \rightarrow B}$.
- $\text{ReEncryption}(RK_{A \rightarrow B}, CT_A) \rightarrow (CT_B)$: On input of a reencryption key $RK_{A \rightarrow B}$ and a ciphertext CT_A, this function outputs a ciphertext CT_B.

This cryptographic method requires a service consumer to grant access to his identity data, every time a service provider requests access to the identity data for the first time. While this approach increases the communication steps by additionally required user interaction, a service consumer can decide, whether to trust a service provider and grant access to the identity data, or to deny the request and cancel the process. If a service consumer grants authorization, he generates a reencryption key for this service provider and sends it to the middleware. The middleware reencrypts the encrypted identity data of the service consumer, forwards it to the respective service provider and stores the reencryption key to prevent necessary user interactions to grant access to the identity data for a subsequent service use. Finally, the service provider can decrypt the identity data using his private key.

4. System Model

This section introduces our defined system model including all actors of our service composition system in addition to necessary components to provide an anonymous service consumption. Based on this trust model, we derive several security objectives, which determine the required communication protocol to fulfil these goals. The whole system model is illustrated by Figure 2.

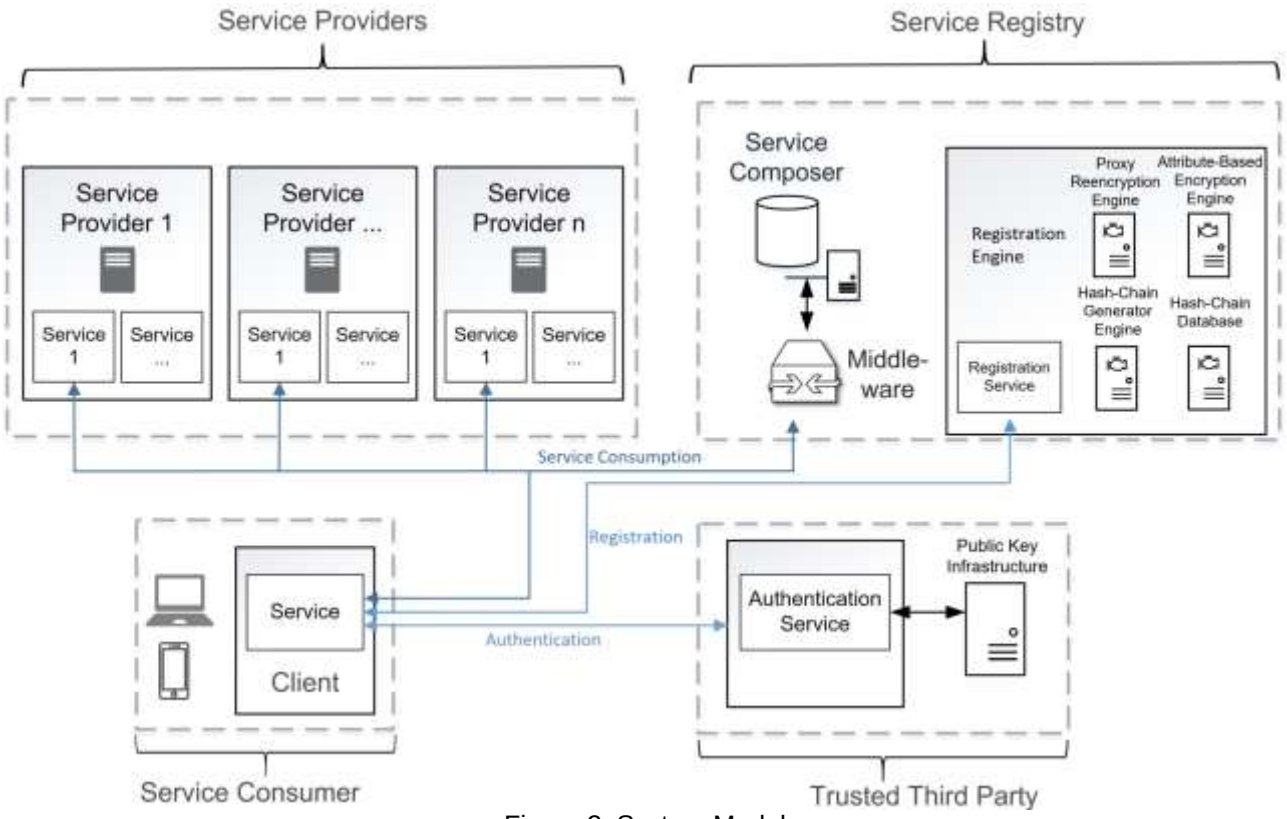

Figure 2: System Model

Service Consumer: A service consumer represents any actor that uses a composite service provided by the service composition system. Initially, they have to authenticate at a Trusted Third Party, which provides reliable means of authentication, which in best case are based on national standards. Afterwards, the service consumer proves a successful authentication to the service registry to register at the service composition system. Service consumers can choose between two security protocols that define the authorization mechanism to grant access to the identity data of a service consumer for service providers.

Client: The client represents the end-device that is used by a service consumer. In our concept, we also consider devices with low computational power such as smartphones to provide a practically applicable concept.

Service Providers: Service providers are actors in a service composition system, who provide individual services for composition. Each service provider has to perform an initial registration and authentication against the service registry to participate in the service composition network. In the course of the registration, a service provider is able to define, if identity data of a user is necessary to perform the business logic. During the registration procedure, the service registry issues digital certificates to service providers authorized to participate in the service composition system.

Service Registry: The service registry consists of several components that provide necessary functionality needed to perform an anonymous registration and service consumption for users. In our concept, we consider both service composition types, service orchestration as well as service choreography. Service orchestration uses a centralized middleware that performs the service call coordination. On contrary, service choreographies use a decentralized approach, without an orchestrating middleware. The service registry requires a service consumer to authenticate in order to be authorized to use a composite service. Through the authentication process, the service registry is only informed about the trustworthiness of a service consumer and does not receive data about his identity. Thus, the service registry is considered as semi-trusted as it cannot read the identity of a service consumer.

Service Composer: This component acts as a service directory and defines the process sequence for composite services required to fulfil a desired use-case. The service sequence of a specific use-case is associated to a use-case identifier stored by the service composer.

4.1 Middleware

Since the composition logic needed for the combination of individual services is not implemented at client side, the middleware announces the required process sequence to a client by requesting it with a use-case identifier at the service composer. In case of service choreographies, the whole process sequence is transmitted to the client. In case of service orchestration, the middleware takes over the whole service call coordination without including the client. The middleware also performs the reencryption algorithm needed for proxy reencryption, if a service provider requests access to the identity data of a service consumer.

Registration Engine: This engine provides necessary functionality to perform the registration und authentication of service consumers as well as service providers. Service providers can register at the registration service (RE), which issues digital certificates to them. This enables the service registry to centrally manage the participation of service providers in the service composition system by simply revoking certificates for unauthorized actors. In the course of registration of service consumers, the registration engine issues hash-chains of a specific length N, where N defines the number of accepted service requests, until a new registration has to be performed at the service registry. This enables the service registry to react according to the provided security level indicated by the performed authentication of service consumers at the external trusted third party. To guarantee anonymity of the identity of a service consumer, the RE offers to choose between two security protocols that are available for registering clients.

Proxy Reencryption Engine: The proxy reencryption engine (PREE) stores reencryption keys created by a service consumer for a specific service provider. A service consumer creates a reencryption key, if a service provider requires revealing of the identity of a user, as it may be needed to perform the business logic. In this case, the PRE reencrypts the encrypted identity data of a service consumer, in order to enable the decryption by the intended service provider. However, the service registry itself does not get any information about the reencrypted identity. For this purpose, the creation of a reencryption key for a specific service provider is considered as consent to reveal the identity to this actor.

Attribute-Based Encryption Engine: The attribute-based encryption engine (ABEE) is involved in the collaborative key management protocol, needed to create the private key parts for a service consumer. In the course of registration, a service provider applies for a set of attributes of a given attribute universe. If the service provider is authorized, the ABEE issues a decryption key associated to these attributes. The key creation requires participation of a service consumer, the ABEE and the middleware, which means that both, the ABEE as well as the middleware hold one private key part of a service consumer. If a service provider requests the identity data of a service consumer, who has chosen to protect the privacy of his identity data through CP-ABE (security protocol 2), the ABEE performs the partial decryption of a ciphertext, whereas the middleware performs the reencryption algorithm defined by the CMK-CP-ABE.

Since neither the ABEE, nor the middleware is able to fully decrypt the encrypted identity data of a service consumer, they only need to be considered as semi-trusted.

Hash-Chain Generator Engine: This engine provides the functionality to generate a hash-chain of a specific length issued to a service consumer after successful registration at the service registry. It stores the public key of a service consumer together with the generated hash-chain, whereby the public key serves as a unique identifier of a user. This way, the hash-chain of a user can be identified without including any identity data.

Hash-Chain Database: This component represents the database, in which a public key of a service consumer is stored together with a hash-chain for authentication purposes.

4.1 Trusted Third Party

The Trusted Third Party (TTP) represents a trustworthy authority that provides means for authentication, which might be based on legal standards, to achieve a high level of security. A service consumer has to register at the TTP before he is able to register at the service registry. The TTP is authorized to read data about the identity of a user and issues digital certificates to service consumer after successful registration. If a service consumer authenticates at the TTP after registration, the TTP issues a confirmation token to the service consumer. This token contains a signature created by the TTP and confirms a successful authentication of a user.

Authentication Service: This service implements the authentication procedure that guarantees a high level of assurance.

Public Key Infrastructure: The TTP uses a Public Key Infrastructure (PKI) for the distribution of digital certificates, when a service consumer performs a registration. Thus, it is able to centrally revoke certificates for unauthorized clients.

5. Processes of Anonymous Service Consumption

In this work, we presented a mechanism to include IoT devices into a service composition system anonymously. We have listed some of the most common schemes used for anonymous authentication and have identified a suitable scheme to be used for the authentication of IoT devices. The hash-chain authentication scheme represents an approach that is suitable for implementation on IoT devices. Furthermore, the hash-chain issuing authority can determine the time until a new registration of a device has to be performed, which enables a flexible reaction on different security levels offered by the performed authentication in the course of the registration.

This section provides a detailed explanation of the processes required to perform an anonymous client registration and authentication of service consumers at the service registry. Additionally, this section also provides the process flow to reveal the identity data of a service consumer for requesting service providers.

5.1 Client Registration

Service consumers have to register at the service registry in order to be authorized to use a composite service. For privacy protection reasons, no identity data related to the service consumer must be transmitted to the service registry. However, participation in the service composition system requires strong authentication in course of the registration procedure to ensure a service usage only by trusted clients. For this purpose, the client also referred to as service consumer initially has to authenticate at a Trusted Third Party (TTP), which provides an authentication mechanism that is based on legal standards and offers a high level of assurance.

After a successful authentication, the TTP sends a signed token to the client that confirms a sufficient level of security regarding the authentication without containing identity data about the service consumer. Next, the client relays this token accompanied by its public key to the registration engine (RE) of the service registry to start the registration at the service composition system. The RE can verify the authenticity of the TTP that has issued the token, which ensures trust in the authentication. To verify a valid ownership of this token by the client, the RE sends a data-to-be-signed (DTBS) referred to as challenge to the client, which in turn signs this data using its private key. After validating the challenge, the RE sends a request to the hash-chain generator engine (HCGE) to trigger the generation of a hash-chain with the length of N, where N represents the number of OTP issued to the client.

The number can vary depending on the security provided by the performed authentication at the TTP. This means, the RE determines the number of allowed authentications through a hash-chain, until the authentication at the TTP has to be performed again, to request a new hash-chain for authentication at the service composition system. Finally, the RE stores the hash-chain together with the public key of the client and transfers the encrypted hash-chain to the service consumer to finish the registration.

5.2 Client Authentication

Registered service consumers have to authenticate at the service registry to be authorized to use a composite service. For this purpose, the client sends a service request including its public key to the middleware serving as service orchestrator, which uses this key to identify the related hash-chain stored in its database. Afterwards, the middleware responses the hash-chain element with the index N, where N represents the length of the hash-chain. Next, the client transmits the hash-chain element with the index $N-1$ to the middleware. Finally, the middleware performs the verification by applying the hash-function initially used to generate the hash-chain on the received hash-value. If the result matches the hash-chain element N, the authentication of the client is successful. Both, the middleware as well as the client delete the hash-chain element N from the hash-chain. A client can repeat this authentication until no hash-chain element is remaining. In this case, he has to re-authenticate at the TTP and the service registry to obtain a new hash-chain.

5.3 Revealing Identity Data to authorized Service Providers

Some service providers require the service consumer to transmit data about their identity in order to be able to perform the business logic. For instance, let us assume a service that reserves a table in a restaurant or another service that books a room in a hotel. In this case, the service provider needs to know the identity of the service consumer to provide his service. For this purpose, we have defined a process to reveal personal data of a service consumer only to the intended service provider, whereas no other actors must be able to read this data. The identity revealing process for both composition types are described in the following paragraphs.

5.3.1 Revealing Identity Data to authorized Service Providers

The process to reveal data about the identity when choosing security protocol 1 integrates the use of CKM-CP-ABE. The following process flow starts after a successful authentication of a service consumer and is illustrated by Figure 3.

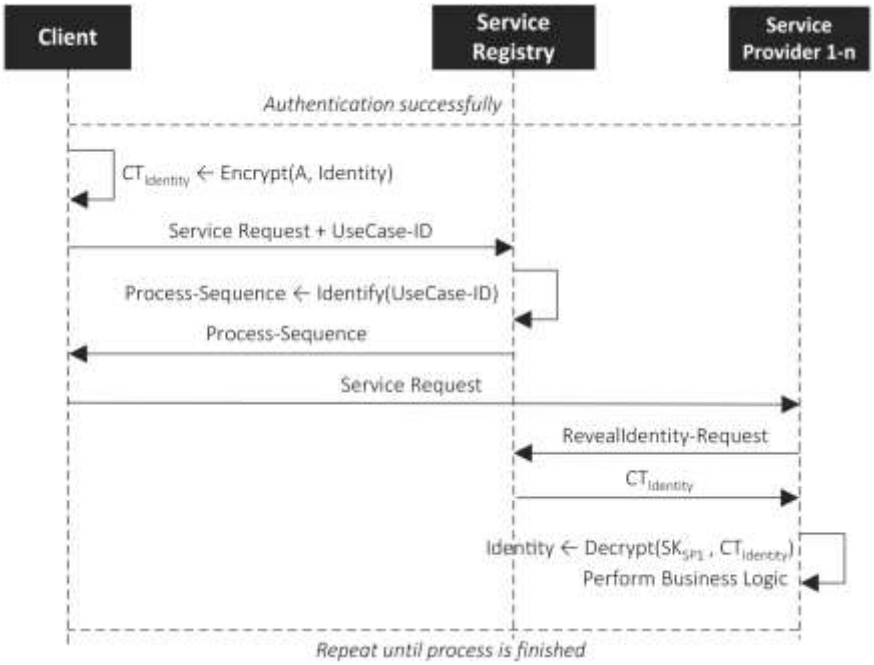

Figure 3: Subject Revealing Process using CKM-CP-ABE

The following definition describes the process flow for service choreographies. However, the process for service orchestrations is the same except for the service call coordination that is not performed by the middleware. A service consumer wants to consume a composite service using his client device and transmits a service request to the middleware of the service registry including the use-case id. After the middleware identifies the required process sequence, the service call coordination flow is transmitted to the service consumer. The client identifies involved service providers and encrypts his identity data with attribute-based encryption to any desired access structure: $\text{Encryption}(\text{Identity},A) \rightarrow \text{CT}$. Afterwards, the client sends a service request to the first actor according to the service sequence. If this service provider possesses the necessary attributes and is able to fulfil the required access structure A that was used to encrypt the identity data, he is finally able to decrypt the identity data of the service consumer. Afterwards, the service provider performs his business logic, encrypts the identity data to the same access structure that was used by the client, and sends a service request to the next actor. Attribute-based encryption enables a one-to-many encryption, which is perfectly suitable for service choreographies, where potentially multiple service providers are able to fulfil a required access structure to decrypt the identity data. Furthermore, the use of CKM-CP-ABE preserves the privacy of the identity data, since no intermediate actor is able to decrypt the identity alone. All operations related to CKM-CP-ABE such as the algorithm for reencryption and partially decryption are performed by the service registry and only little computational effort is required to decrypt the identity at service provider side. This is an advantage regarding the performance.

5.3.2 Security Protocol 2: Revealing Identity Data

The process to reveal data about the identity of a user in when using security protocol 2 integrates the use of proxy reencryption (PRE). The following process flow starts after a successful authentication of a service consumer and is illustrated by Figure 4. The following definition describes the process flow for service orchestrations, where the middleware takes over the service call coordination.

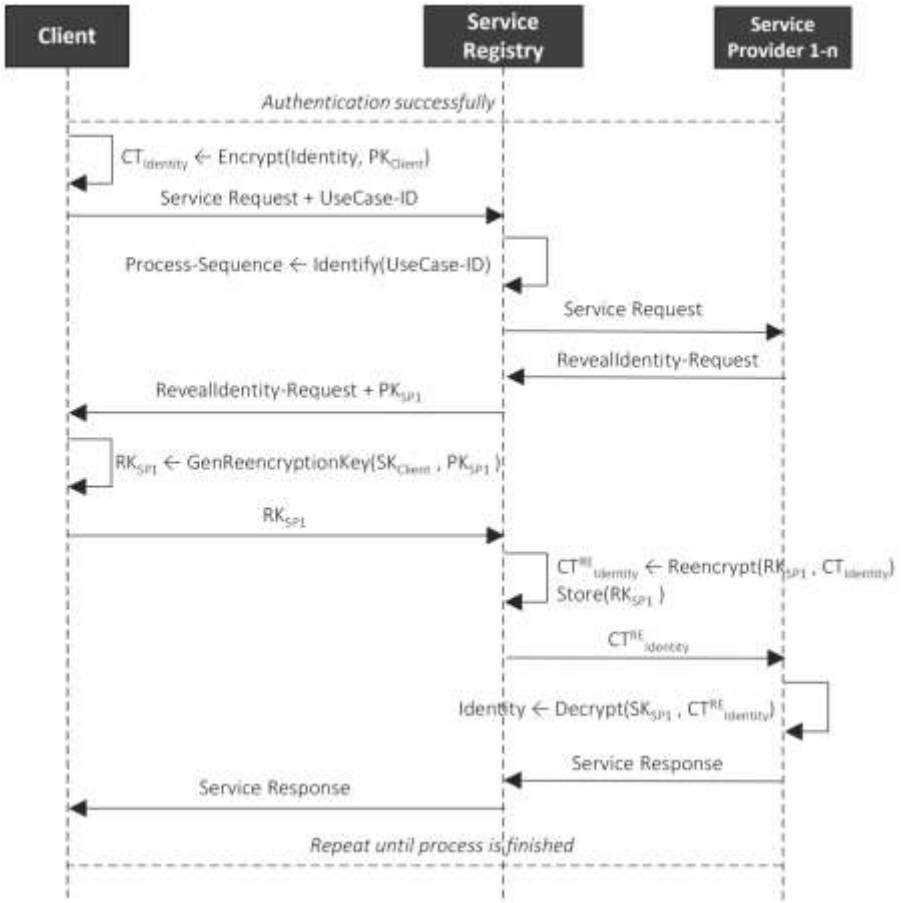

Figure 4: Subject Revealing Process using Proxy Reencryption

Similar to the process for the identity revealing process with security protocol 1, a service consumer wants to consume a composite service using his client device and transmits a service request to the middleware of the service registry including the use-case id. The middleware identifies the required process sequence via the use-case id and forwards the service request to the first service provider according to the process flow. As this service provider requires identity data about the service consumer, he sends a request to the middleware, to reveal the required data. If the middleware has not registered any reencryption key previously issued by the client for the respective service provider, it forwards the request to the client accompanied by the certificate of the service provider. T

he client can decide, whether he wants to send his identity data to the requesting actor, or if he wants to deny the request and cancel the process. If the client grants authorization to read the identity data, he generates a reencryption key for this actor by extracting the public key from the received certificate and performing the algorithm to generate the reencryption key: $\mathrm{ReKeyGen}(SK_{Client},PK_{SP1}) \rightarrow (RK_{Client \rightarrow SP1})$.

This reencryption key is transmitted to the middleware, which relays the key to the proxy reencryption engine to store it into a database together with necessary information. This prevents the client to grant an authorization and generate a reencryption key for this service provider multiple times in case of a repeated use of this service.

Afterwards, the middleware performs the reencryption of the encrypted identity data, which is contained in the initial service request, by performing the reencryption algorithm: $\mathrm{ReEncryption}(RK_{Client \rightarrow SP1},CT^{Client}_{Identity}) \rightarrow (CT^{SP1}_{Identity})$. The middleware sends the reencrypted identity data to the service provider, who is now able to perform a decryption using his private key: $\mathrm{Decryption}(SK_{SP1},CT^{SP1}_{Identity}) \rightarrow (\mathrm{Identity})$.

After a successful decryption of the identity data, the service provider performs his business logic and answers with a service response.

Obviously, PRE provides a practical approach to preserve the privacy of the identity data in service orchestrations, since no intermediate actor such as the middleware is able to read any data. Additionally, operations related to the reencryption of ciphertexts can be performed by the middleware, which relieves client devices with potentially low computational power. Also, the reencryption keys can be stored at server side, which prevents storage capacity problems at client side and reduces the required communication steps in case of a multiple use of the same service.

6. Evaluation

This section illustrates a case study for drug addiction counselling, introduces the implementation of a proof-of-concept and evaluates the performance of the different security protocols defined in the previous section.

6.1 Case-Study

To demonstrate the feasibility of our proposed solution, let us assume an example use-case for drug addiction counselling. In this use-case, first a client fills out a form representing a Drug Use Disorders Identification Test (DUDIT) provided by an authority representing the National Addiction Prevention (NAP). To ensure the privacy of service consumers, they can conduct the DUDIT anonymously without revealing their identity data to the service provider. Second, the client can decide to book a consultation at an Addiction Counselling Authority (ACA). This booking service requires service consumers to unveil their identity data to the service provider in order to be able to register the consultation for a specific person. Since service consumers can choose between both security protocols introduced in the previous section, each service provider owns decryption keys associated to specific attributes. The NAP owns the attributes ’Anonymous−Usage’ and ’Governmental−SP’, where ’Anonymous−Usage’ indicates that this service requires no identity data of service consumers. On contrary, the ACA possesses the attributes ’Governmental−SP’ and ’Non−Anonymous−Usage’ indicating the need for personal data of the client.

6.2 Implementation

To evaluate our proposed concept, we have implemented a proof-of-concept of our case study, including an anonymous service consumption using PRE and CKM-CP-ABE. We have implemented Java-based web services for both service providers as well as the middleware, which is part of the service registry, takes over the service call coordination, and if necessary, performs reencryption algorithms for PRE and CKM-CP-ABE depending on the chosen security protocol. At client side, we have implemented a service on an Android smartphone starting the process, as we also want to consider devices with limited computational power. In our example use-case, we do not focus on specific contents that might be exchanged, since we only want to prove the feasibility of our security protocols, which guarantee an anonymous service consumption. The following paragraphs briefly describe the implementation of both security protocols introduced in section 5.3.1 & 5.3.2.

6.2.1 Implementation of Security Protocol 1

To implement Security Protocol 1 (SP1), which is based on the use of PRE we have used a publicly available Java library providing the PRE code[1] based on the Java Pairing-Based Cryptopgraphy (PBC) library[2].

6.2.2 Implementation of Security Protocol 2

To implement Security Protocol 2 (SP2) based on CKMCP-ABE and the the collaborative key management protocol for CP-ABE introduced by Lin et al. [14], we used the cryptographic scheme provided by the IAIK Java Cryptography Extension (IAIK-JCE[3]) as well as the IAIK ECCelerate elliptic

[1] PRE: https://github.com/chris-wood/proxy-reencryption
[2] PBC: https://crypto.stanford.edu/pbc/download.html
[3] IAIK-JCE & https://jce.iaik.tugraz.at/products/core-cryptotoolkits/jca-jce/

curve library[4]. These libraries provide cryptographic functionality for bilinear pairings needed to implement the cryptographic algorithms of CKM-CP-ABE. For our evaluation, we choose a security parameter of λ = 256 bits, providing long-term security according to NIST's recommendation [21].

6.3 Performance

To provide a comparison, we briefly describe our test setup, investigate the different execution results for both security protocols introduced in section 5.3.1 & 5.3.2 and provide a short comparison.

6.3.1 Test Setup

The security protocol algorithms are performed by three web services running on a Lenovo Thinkpad T460s, which has an Intel2 i5-6200U dual-core processor and 12GB RAM. To start the composite service, we have implemented an Android web service at client-side running on a HTC U11, which includes a Qualcomm Snapdragon 835 quad-core processor with 2.45 GHz each and 4GB RAM.

6.3.2 Comparison of Security Policy

Figure 5 provides a comparison of the execution results in milliseconds between the cryptographic methods applied for SP1 and SP2. The Android web services running on the client device performs the encryption and decryption algorithms of both cryptographic schemes. The web service acting as middleware performs the reencryption algorithms of both cryptographic methods as well as the algorithm needed for a partial decryption as defined by CKM-CP-ABE. Since PRE does not include a partial decryption, the chart does not provide an execution result for this algorithm. To provide reliable results, we have conducted the test 50 times and used the median of these performance values.

The test results demonstrate a practically efficient execution time for the cryptopgraphic methods used in both security protocols. Derived from these results the total execution time including all necessary algorithms for CKM-CP-ABE is 277[ms], while the total execution time of PRE is 331[ms] excluding the algorithms for the setup and key creation.

However, since we only evaluated the performance of cryptographic algorithms, we did not include the additional time needed by a client to create a reencryption key for an authorized actor, which decelerates the overall performance of SP1. Thus, the overall performance of SP2 is slightly better than for SP1. Nevertheless, SP1 offers the advantage to explicitly authorize specific service providers to read the identity of service consumers.

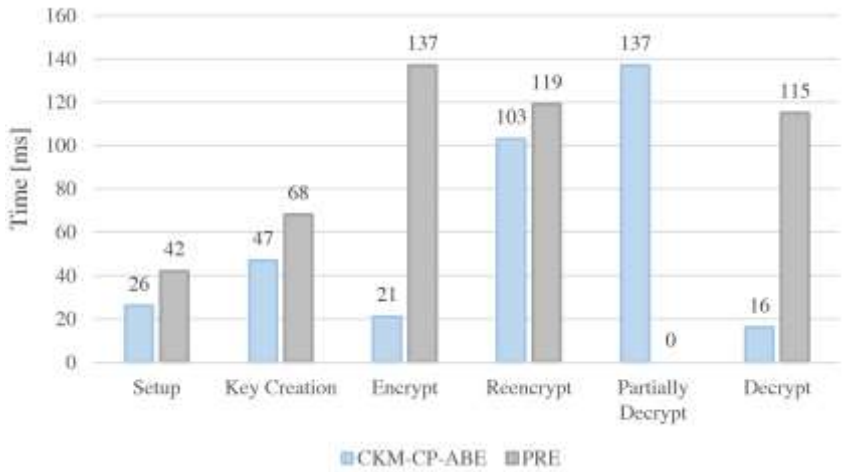

Figure 5: Comparison of the execution results of PRE and CKM-CP-ABE

[4] IAIK ECCelerate: https://jce.iaik.tugraz.at/products/corecrypto-toolkits/eccelerate/

7. Discussion

This section provides a discussion about the mechanisms used to preserve the privacy of identity data of service consumers and to provide an authentication with high usability.

Privacy: Through the application of PRE and CKM-CP-ABE it is possible to provide an anonymous authentication using different communication protocols. CKM-CP-ABE empowers a service consumer to define an access structure by a combination of attributes to their identity data. A service consumer performs a one-to-many encryption, where identity data is encrypted over an access structure that can potentially be decrypted by multiple recipients. However, this approach does not make it possible to deny authorizations for specific participants, provided that these actors cannot fulfil the access structure with the attributes associated to their decryption keys. For this purpose, our concept includes a second solution using PRE in the communication protocol. PRE empowers a service consumer to explicitly grant authorization for specific service providers. While this approach includes additional user interaction, it offers significant advantages regarding the privacy protection of their identity data. Derived from the evaluation, it becomes apparent that both cryptographic methods offer satisfying execution times, even on devices with limited computational power.

Usability: To enhance the overall usability of the authentication mechanism, our proposed concept provides means for a simplified authentication similar to SSO. A simplified authentication only requires the client to perform a hash-chain authentication, which does not include any user-interaction. If a hash-chain has no remaining elements after an authentication, the service consumer has to perform a new registration at the service registry. Thus, a hash-chain authentication empowers operators of a service composition system to dynamically define the number of allowed simplified authentications and make it possible to react according to the risk related to the provided services. Ultimately, the security protocol using CKM-CP-ABE further improves the usability, since service consumers only have to define an access structure to their identity data once.

Future Work: In our future work, we will investigate the suitability of further advanced cryptographic methods to extend the amount of available security protocols. We especially aim on mechanisms to improve privacy protection and to enhance the overall usability.

8. Conclusion

In this work, we have introduced a concept enabling service consumers to anonymously use a composite service provided by a service composition system. To preserve the privacy of identity data of service consumers we provide different security protocols, which include advanced cryptography. These security protocols offer diverse advantages regarding data security and usability. Additionally, our proposed concept provides a mechanism for operators of a service composition system to offer a simplified authentication for service consumers, which further enhances the usability. The service registry can dynamically define the number of allowed simplified authentications until an initial registration has to be performed again. The feasibility of this concept is demonstrated by a proof-of-concept implementation representing a service for addiction counselling. The execution results demonstrate a practically efficient approach, including devices with limited computational power such as smartphones.

References

[1] Giuseppe Ateniese and Gene Tsudik. “Some Open Issues and New Directions in Group Signatures”. In: Financial Cryptography. Springer Berlin Heidelberg, 1999 (cit. on p. 2).

[2] Giuseppe Ateniese et al. “Improved Proxy ReEncryption Schemes with Applications to Secure Distributed Storage”. In: ACM Trans. Inf. Syst. Secur. 9.1 (2006). I S S N: 1094-9224 (cit. on p. 5

[3] J. Bethencourt, A. Sahai, and B. Waters. “Ciphertext-Policy Attribute-Based Encryption”. In: 2007 IEEE Symposium on Security and Privacy (SP ’07). 2007, pp. 321–334. D O I: 10.1109/SP. 2007.11 (cit. on p. 4).

[4] K. Bicakci and N.Baykal. “Infinite length hash chains and their applications”. In: Proceedings. Eleventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises. 2002, pp. 57–61 (cit. on p. 3).

[5] Emmanuel Bresson, Jacques Stern, and Michael Szydlo. “Threshold Ring Signatures and Applications to Ad-hoc Groups”. In: Aug. 2002, pp. 465– 480 (cit. on p. 2).

[6] Jan Camenisch and Markus Stadler. “Efficient Group Signature Schemes for Large Groups”. In: CRYPTO ’97 1296 (Jan. 1997) (cit. on p. 2).

[7] Zhengjun Cao and Lihua Liu. “Remarks on the Cryptographic Primitive of Attribute-based Encryption”. In: (Aug. 2014) (cit. on p. 4).

[8] A. Djellalbia et al. “Anonymous authentication scheme in eHealth Cloud environment”. In: 2016 11th International Conference for Internet Technology and Secured Transactions (ICITST). 2016 (cit. on p. 2).

[9] S Goldwasser, S Micali, and C Rackoff. “The Knowledge Complexity of Interactive ProofSystems”. In: New York, NY, USA: Association for Computing Machinery, 1985. I S B N: 0897911512 (cit. on p. 2).

[10] D. He et al. “Anonymous Authentication for Wireless Body Area Networks With Provable Security”. In: IEEE Systems Journal 11.4 (2017), pp. 2590–2601 (cit. on p. 1).

[11] Y. Ishai et al. “Cryptography from Anonymity”. In: 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS’06). 2006, pp. 239– 248 (cit. on p. 1).

[12] M. Jensen, S. Schaege, and J. Schwenk. “Towards an Anonymous Access Control and Accountability Scheme for Cloud Computing”. In: 2010 IEEE 3rd International Conference on Cloud Computing. 2010, pp. 540–541 (cit. on p. 2).

[13] Leslie Lamport. “Password Authentication with Insecure Communication”. In: 24.11 (1981) (cit. on p. 2).

[14] Guofeng Lin, Hanshu Hong, and Sun Zhixin. “A Collaborative Key Management Protocol in Ciphertext Policy Attribute-Based Encryption for Cloud Data Sharing”. In: IEEE Access PP (May 2017), pp. 1–1 (cit. on pp. 4, 10).

[15] V. Pacheco and R. Puttini. “SaaS Anonymous Cloud Service Consumption Structure”. In: 2012 32nd International Conference on Distributed Computing Systems Workshops. 2012, pp. 491–499. D O I: 10.1109/ICDCSW.2012.28 (cit. on p. 1).

[16] M. P. Papazoglou et al. “Service Oriented Computing: State of the Art and Research Challenges”. In: Computer 40.11 (2007), pp. 38–45 (cit. on p. 1).

[17] David Pointcheval and Jacques Stern. “Provably Secure Blind Signature Schemes”. In: 1163 (Oct. 2001). D O I: 10.1007/BFb0034852 (cit. on p. 2).

[18] Ronald L. Rivest, Adi Shamir, and Yael Tauman. “How to Leak a Secret”. In: Advances in Cryptology — ASIACRYPT 2001. Ed. by Colin Boyd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 552–565 (cit. on p. 2).

[19] S. Ruj, M. Stojmenovic, and A. Nayak. “Decentralized Access Control with Anonymous Authentication of Data Stored in Clouds”. In: IEEE Transactions on Parallel and Distributed Systems 25.2 (2014), pp. 384–394. D O I: 10.1109/TPDS. 2013.38 (cit. on p. 1).

[20] Amit Sahai and Brent Waters. “Fuzzy IdentityBased Encryption”. In: Advances in Cryptology – EUROCRYPT 2005. Ed. by Ronald Cramer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 457–473 (cit. on p. 4).

[21] National Institute of Standards Technology. Recommendation for Key Management, Part 1: General (Rev 4). SP 800-57. 2016 (cit. on p. 10).

[22] Fengtong Wen and Dianli Guo. “An improved anonymous authentication scheme for telecare medical information systems”. In: Journal of medical systems 38.5 (May 2014), p. 26. I S S N: 0148-5598 (cit. on p. 1).

[23] Dominik Ziegler, Josef Sabongui, and Gerald Palfinger. “Fine-Grained Access Control in Industrial Internet of Things: Evaluating Outsourced Attribute-Based Encryption”. In: June 2019, pp. 91– 104 (cit. on p. 4)

[24] Timothy Libert. 2018. An Automated Approach to Auditing Disclosure of Third-Party Data Collection in Website Privacy Policies. In Proceedings of the 2018 World Wide Web Conference (WWW '18). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 207–216. DOI:https://doi.org/10.1145/3178876.3186087