OCCASIO: an Operable Concept for Confidential and Secure Identity Outsourcing

Jens Köhler and Hannes Hartenstein
Karlsruhe Institute of Technology (KIT) - Steinbuch Centre for Computing (SCC) & Institute of Telematics
Karlsruhe, Germany
Email: jens.koehler@kit.edu, hannes.hartenstein@kit.edu

***Abstract*—While federated identity management separates service provisioning from identity provisioning, the identity provider is usually operated at the home organization of the identities. We address the challenge of outsourcing the entire identity provider with its user database to an untrusted external provider in a secure and privacy-preserving way. With this type of outsourcing, the home organization is no longer required to operate high availability infrastructure for access management. Instead, the home organization only needs to frequently attest that the identity data in the outsourced database is still up to date, a task that is much less demanding than providing access decisions whenever a user wants to make use of a service. In this paper we present Occasio, a concept that permits secure outsourcing of identity and access management to untrusted external providers. Occasio builds on concepts of outsourcing databases and particularly on Merkle Hash Trees. We show that Occasio matches all security requirements for operation in an untrusted environment. Furthermore, we demonstrate that Occasio can be easily integrated into the SAML standard. We present results of a performance evaluation that shows that Occasio behaves well in terms of overhead. Finally, we show that with Occasio identity data of different home organizations can be 'aggregated' without being linkable by someone other than the services that are granted to do so by the user.**

***Keywords*—service outsourcing; identity and access management; cloud; availability;**

I. Introduction

The cloud computing paradigm offers enterprises an attractive way to outsource their IT services. As cloud platforms can provide a high degree of availability and robustness, enterprises are able to minimize the IT infrastructure that they have to maintain in order to offer highly available and robust services. However, if the outsourced services are used by third parties, the identities and the access rights of these parties have to be managed and enforced. This is commonly subsumed under the term of *identity and access management* (IAM).

The prevalent conception concerning IAM is that it has to be hosted by the home organization of the identities. Reasons for this include legal issues concerning the outsourcing of personal information to external providers. Furthermore, a malicious or compromised provider hosting the IAM might influence the access control decision of any relying service. Considering the possibility that the external provider can be compromised by attackers, even the provider itself has an interest in not being able to tamper with the access control of the relying services due to accountability aspects. Using federated identity management concepts [1] [2], services can be outsourced to external providers while the identity provider that performs the IAM remains in-house. Federated identity management standards such as the *Security Assertion Markup Language* (SAML) [3] are already well established.

However, if the identity provider is hosted by the home organization, the availability of the relying services depends on the platform of the home organization. For instance, if a user wants to access a service and the service cannot make an authorization decision due to an unavailable identity provider, the service is unavailable to the user as access is denied. Thus, a highly available platform has to be maintained by the home organization to ensure a high availability of relying services. Especially for start-up enterprises this fact constitutes a major hurdle as establishing a highly available IT infrastructure is not economical and the know-how to maintain it is often not available. Therefore, we argue that a paradigm shift is necessary: the topic of outsourcing identity providers to potentially malicious or compromised providers has to be investigated and technical means to establish trust have to be developed.

In this paper, we present Occasio (**O**perable **C**oncept for **C**onfidential **A**nd **S**ecure **I**dentity **O**utsourcing), an approach to securely outsource identity providers to untrusted external providers. Occasio enables *home organizations* (HOs) to outsource *statements* on user attributes to untrusted external providers, so-called *identity provider representatives* (IdPRs). Similarly to federated identity management approaches, services providers (SPs) can request statements on a user from the IdPR and base their authorization decision on them. As the HO does not participate in this process, the availability of a service only depends on the SP itself and the external IdPR.

As a use-case, let us consider a start-up enterprise (HO) that requires a user to register in order to use a service of the start-up itself and additional third-party services. The user's identity at the start-up includes attributes issued by the user (e-mail, address,...) and attributes issued by the start-up (premium status,...). The enterprise wants to outsource its service and its identity provider to an external provider (IdPR), so it does not have to operate highly available infrastructure. However, the start-up does not entirely trust the external provider. Moreover, it is not clear whether all relying SPs that should be accessible for the start-up users are completely trustworthy and secure.

Occasio addresses this use-case by assuming the following trust model: neither the external SPs nor the IdPR are trusted by the HO or the user. They are assumed to be "malicious" in the sense that they might strive to gain unauthorized access to outsourced statements. Furthermore, they might strive or

978-3-901882-50-0

might be compromised to alter statements or to inject already revoked statements in order to impersonate users or enable users to use services they are not authorized to. They might even collaborate with each other to reach these goals. While not trusting the IdPR, SPs trust the HO, i.e., the entity that is the authoritative source of the identity information. Occasio enables the HO to outsource the statements in a secure way such that the IdPR may neither view the statements nor tamper with the access control decisions by altering statements or providing already revoked statements upon which the relying services base their authorization decision. Furthermore, SPs only have access to statements they have been authorized to view by the user beforehand.

The assumption of "malicious" providers is a worst case assumption that might even apply in case of perfectly trusted providers that have been compromised. Thus, Occasio not only enables IdP-outsourcing to untrusted providers but can also be used to enhance security of operations.

The main contributions of this paper are:

- A **concept to securely outsource identity and access management** to untrusted parties based on Merkle Hash Trees and public-key based authentication protocols.
- An **approach to integrate the concept** into the existing SAML standard.
- A **prototypical implementation** of the concept based on the open source framework simpleSAMLphp[1] and a **performance evaluation** of it.

The paper is structured as follows: In Section II possible use cases are discussed. In Section III the requirements for an approach to securely outsource IAM are listed and fundamental considerations how to fulfill them are presented. Afterwards, in Section IV related work is summarized. In Section V, we present our approach Occasio to securely outsource identity providers. Furthermore, in Section VI it is shown how the approach can be integrated with the existing SAML standard. Our concept is evaluated in Section VII and the paper is concluded in Section VIII.

II. Discussion: Use Cases

One might wonder in which use cases it is beneficial to apply Occasio to outsource IAM. Especially small and medium enterprises (SMEs) or start-up enterprises can benefit from Occasio. For SMEs with a relatively low number of users and therefore a relatively small workload, it may be more economical to rely on external providers than maintaining a server room with highly available hardware. For start-up enterprises the risk can be reduced as no acquisition costs arise if only external clouds are used as a foundation. Furthermore, also large enterprises without IT background such as hospital consortiums can benefit from using Occasio to outsource access management along with other services as establishing highly available IT infrastructures and training personnel to operate it may not be favorable.

Moreover, Occasio can cover use cases that are not covered by traditional federated identity management approaches. For instance, consider an employer and a health insurance company that both issue statements about a user. Furthermore, consider a service provider that needs statements about a user from the employer as well as from the insurance company. Due to legal reasons the employer and the insurance company may not merge their identity providers. Thus, in order to provide the necessary statements to the service, the user has to log on to both employer and insurance company. This is commonly subsumed under the term *Attribute Aggregation* [4]. With Occasio, it is possible for the employer and the insurance company to securely outsource the statements to a single IdPR. Thus, the user is enabled to access the service by just logging on once and neither the employer nor the insurance company get to know more statements than maintained by themselves.

[1] http://simplesamlphp.org/

III. Requirements and Fundamental Considerations

A. Requirements

In the following, the requirements of each participating party will be listed. Afterward, the requirements for an approach to securely outsource IAM will be derived.

From a **user's point-of-view**, the confidentiality of the personal identity information is important. The users entrusts the HO with their identity information. However, the IdPR and other users are not entitled to view this information. SPs may view parts of the information if they have been authorized to do so by the user beforehand.

From a **HO's/SP's point-of-view**, the integrity of the identity information is important. On the one hand, the HO trusts in the SP to correctly enforce access control based on the conveyed statements[2], on the other hand, the SP trusts in the soundness of the statements of the HO. Thus, neither a user, nor the IdPR, nor another SP may be able to manipulate the statements of the HO or exchange them for revoked statements before they are conveyed to the SP. Furthermore, besides the authenticity of the statements, the authenticity of the user is important to the SP. Thus, it must not be possible for an IdPR, another SP or another user to impersonate a user.

Taking both point-of-views into account, an approach that allows to outsource IAM to external providers should meet the following requirements:

- **No involvement of the HO** during user logins: When a user accesses a service, the HO should not have to participate. Otherwise the availability of the relying services would be affected by the availability of the HO. For instance, the start-up of the use-case introduced in Section I should not be involved during user logins to benefit from outsourcing IAM.
- **Confidentiality** of the statements: Only the HO and the user may have access to plaintext statements. The SPs may have access to parts of the statements if authorized by the user. For instance, the external provider of the use-case introduced in Section I must not have access to user attributes to adhere to privacy laws.

[2] Statements constitute assertions on attributes of a user, however we use the term *statement* to avoid confusion with SAML assertions.

- **Correctness** of the statements: Neither the IdPR, nor the user, nor an SP should be able to manipulate the statements provided by the HO. For instance, the external provider of the use-case introduced in Section I must not be able to manipulate statements to influence the authorization decision of relying SPs.
- **Freshness** of the statements: Ensuring the Correctness of the statements is not sufficient: the IdPR could still ”freeze” the statements of a user (ignoring updates). SPs must be able to determine how old the statements are and thus be able to reject old statements that might have been already revoked.
- **Authenticity** of the user: Neither the IdPR, nor an SP, nor another user should be able to impersonate a user towards another SP. For instance, in the use-case introduced in Section I no other party than the start-up itself should be able to impersonate users.
- **No redundant identity data** may be stored at the SP or the user. This would contradict the federated nature of the system and would imply a complex management overhead in case of many users and SPs. For instance, in the use-case introduced in Section I, the SPs rely on the HO (or more accurately its representative, the IdPR) rather than storing identity data and operate their own user management.

Beside these fundamental requirements, further requirements exist:

- **Usability**: A user that logs on to a service based on the IAM outsourcing approach should not have to use any uncommon authentication techniques. Furthermore, the workflow to log on should be kept as simple as possible.
- **Ease of integration**: A service provider should be able to easily integrate the approach into its service deployment. In particular, the interface to get assertions on the identity of a user should not have to be adapted.
- **Performance**: There overhead induced for the HO by outsourcing statements should be kept reasonably low. The HO should be able to outsource statements and to maintain them using commodity hardware.

After presenting the Occasio approach, it will be evaluated regarding the listed requirements in Section VII-A.

B. Fundamental Considerations

To address the correctness and freshness requirements, findings from the database outsourcing community can be leveraged. The naive approach to ensure both correctness and freshness of outsourced data is to frequently issue a timestamped signature of each outsourced data item to the external provider. After requesting an outsourced data item, the correctness of the data item can be ensured by verifying the signature. Furthermore, the freshness of the data item can be ensured by checking whether the timestamp is recent enough. However, the outsourcing entity has to frequently attest the freshness of *each* outsourced data item by updating its timestamp and therefore its signature. Depending on the number of statements this continuous signing effort constitutes a significant overhead. Therefore, Occasio utilizes *Merkle Hash Trees* (MHTs) [5] that allow attesting the freshness of *all* outsourced statements by issuing a single signature.

Due to the assumed trust model, the authenticity of the user cannot be ensured by password-based authentication against the IdPR or the SP as neither of them are trusted. In order to authenticate to an SP via password, the SP would need a verifier to verify the password. This verifier would enable malicious SPs to use offline dictionary attacks to get the password and use it to access other services on behalf of the user. One approach would be to use SP-specific passwords. However, this approach would impede the usability of Occasio, as a user would have to remember one password for each service provider. Thus, one of the main benefits of federated identity management would be undermined.

Therefore, a protocol should be used that allows a user Alice to authenticate against an SP Malory without increasing Malory’s chance to successfully authenticate against an SP Bob as Alice. Zero-knowledge protocols comply with this property [6]. Furthermore, public-key based authentication protocols that are already widely used are suitable. For instance, the Secure Shell Protocol [7] allows a user to authenticate based on a public key that has been previously deployed at the SP. Moreover, in the case of web-based services the user may use client certificates during the TLS handshake to authenticate to an SP without revealing any credentials [8].

IV. Related Work

Multiple contributions suggest the use of federated identity management approaches to enforce access control for outsourced services [1], [2], [9]. While the proposals decouple the outsourced service provider from the identity providers that store the identity data, they do not address the fact that the identity providers have to be trusted and may be hosted only by the home organization itself due to legal reasons in many cases. This in turn reduces the benefit of outsourcing services as an available infrastructure has to be maintained in-house to host the identity provider. Our approach leverages the advantages of the federated identity management approach while aiming at minimizing the amount of necessary in-house infrastructure and decoupling the availability of the relying services from it.

In [10] and [11] an entity-centric approach is proposed that stores the identity data at the user rather than at a trusted identity provider or the service provider. However, as the issuer of identity data is not necessarily the user itself (e.g., in case of the user’s organizational affiliation) it can be hard for the issuer to update the identity data upon changes. Furthermore, the service provider cannot decide whether the identity data has been revoked by the issuer. Our approach allows the service provider to determine whether it is presented revoked identity data. Moreover, no identity information is stored by the user or the service providers, making it easier to update identity data upon changes.

Both in [12] and [13] frameworks to enforce access control in the cloud are proposed that hide the identity of the user from the service provider. The approaches just take authorization tokens into account. However, in some cases identity data is

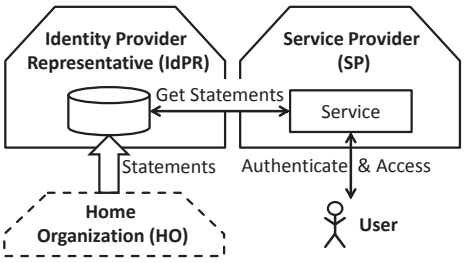

Fig. 1. Architecture overview

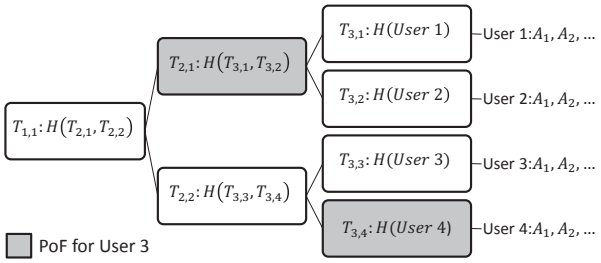

Fig. 2. Exemplary Merkle Hash Tree (MHT)

needed that is not an authorization token and needs to be visible to the service provider. For instance, a service provider might require the email address of a user. Furthermore, in contrast to Occasio, the proposed frameworks make either use of at least one trusted party that has to be available to enforce access control or let the user store identity information making it hard to apply changes on it.

The use of MHTs has been proposed by contributions of the database outsourcing community to assure the completeness, the correctness [14]–[16] and the freshness [17] of databases and of query results in particular. While a query result is *complete* if no data items are omitted, it is *correct* if no contained data items are manipulated. Furthermore, a query result is considered *fresh* if no stale version of a data item is contained in it. However, using MHTs induces a big overhead to assure the completeness of query results upon changes to the outsourced data [14], [16], [18]. While completeness is an essential property for outsourced databases, it is not necessary in case of Occasio. This is due to the fact that by omitting statements, neither can the confidentiality of a statement be harmed, nor can an SP make a false positive authorization decision. In Occasio, the number of MHT nodes that have to be adapted upon a change depends only logarithmically on the number of maintained identities. Moreover, we argue that the number of changes to identity data is low compared to other database applications. Monitoring the identity management system of the KIT that maintains over 30000 identities has shown that more than 100 change operations per minute do not even occur in peak times [19]. In Section VII-B we will show that the overhead induced by Occasio is acceptable even for large numbers of maintained identities.

V. Occasio

A. Overview

A basic overview of Occasio is shown in Figure 1. Whereas the HO outsources statements on users to the IdPR, the HO does not have to participate during a login procedure of a user at an SP (cf. Requirements in Section III). To keep the identity information confidential, the statements are encrypted by the HO before being outsourced to the IdPR using common encryption schemes. The possession of the decryption key varies depending on which flavor of Occasio is used. We distinguish between a *user-centric protocol* and a *provider-centric protocol* (see Section V-C). Whereas the statements are decrypted with a user-specific key in the user-centric protocol, a key specific to the SP to which the statements are addressed is used in the provider-centric protocol.

The correctness and the freshness of the statements is guaranteed by a so-called *proof of freshness* (PoF). A PoF provides the guarantee that the according statements have been valid at a given time. It is issued by the HO, stored by the IdPR and delivered to the SPs upon request. To ensure the freshness of received statements, an SP only accepts them if the timestamp of the according PoF lies within a certain time window. Thus, the PoF has to be updated frequently by the HO. In the following, we will refer to the time window as *freshness window*.

The size of the freshness window constitutes a control for the trade-off between the time an unavailable HO does not affect user logins and the time window in which the IdPR can suppress changes to the statements made by the HO without being detected. For instance, consider a freshness window of 2 hours ($2h$). If the HO lastly updated the PoF at the time t_1 and becomes unavailable, users can still access the relying SPs as the SPs accept the PoF until the time $t_1 + 2h$. Furthermore, if the HO changes a statement at the time t_2 with $t_1 + 2h \geq t_2 > t_1$, the IdPR can suppress this change until the time $t_1 + 2h$ as the relying services still accept the PoF issued at t_1 for the former version of the statement. However, at latest 2 hours after the new statement is submitted by the HO, the IdPR is no longer able to deliver the old statements as the PoF issued for them is no longer accepted by the SPs. The PoF concept will be explained in more detail in Section V-B.

As mentioned before in Section III-B, users must not use passwords to authenticate to either the IdPR or an SP. Thus, we focused on public-key based authentication protocols. Whereas the user possesses a private key, the corresponding public key is part of the identity information managed by the HO, allowing an easy revocation in case the private key of the user was disclosed. Thus, once a SP retrieves the statements about a user from the IdPR, it is able to use the contained public key to authenticate the user via public-key based authentication protocol. Therefore, the user may use a single private key to authenticate to an arbitrary SP without enabling the SP to impersonate the user towards other SPs.

B. Proof of Freshness

A PoF contains a timestamped signature of a momentary state of the statements it refers to. As the PoF has to be renewed frequently, it would not be feasible to provide a signature for each statement that is outsourced to the IdPR. For instance, in case of 10.000 users with 10 attributes each, 100.000 statements would have to be signed frequently by the HO. Therefore, Occasio utilizes MHTs that allow attesting

the freshness of all outsourced statements by performing a single signature. When changing the value of a statement, the MHT has to be adapted. Compared to the naive approach of signing single statements, this induces an overhead of updating a number of MHT nodes that logarithmically depends on the number of outsourced identities. However, the induced overhead is bearable as the performance measurements presented in Section VII-B will confirm.

An exemplary MHT is shown in Figure 2. Each node $T_{i,j}$ of the hash tree consists of the hash of the concatenated children node's values. Each tree leaf consists of the hash of the plaintext statements A_k of a user. Additionally, to prevent dictionary attacks against the hash values of the leaves, a random salt is applied before hashing. Just as the statements themselves, this salt is encrypted and stored at the IdPR. Thus, the MHT leaks no information about the statements and may be safely outsourced. As shown in [20], it is not possible to tamper with any node without changing the value of the root node. Therefore, for the HO, it is sufficient to sign the root node's value together with a timestamp to acknowledge that all contained statements are valid at a given time.

The MHT is stored at the IdPR. Thus, using the MHT, the IdPR is able to construct the PoF for a single user as follows:

1) Build the initial PoF by concatenating the siblings of the user's leaf node.
2) Move on to the parent node and concatenate the siblings of it to the PoF. Repeat until the root node is reached.
3) Add the timestamped signature of the root node to the PoF.

For instance, the PoF of user 3 in Figure 2 would look like $(T_{3,4}, T_{2,1}, Sign_{HO}(timestamp, T_{1,1}))$. In order to verify the PoF for the statements $A_1, A_2, \ldots$, an SP can perform the following operations:

1) Compute the value of the leaf node by hashing the statements $A_1, A_2, \ldots$.
2) Compute the value of the parent node P_1 by hashing the computed value of the leaf node with its siblings (which are contained in the PoF).
3) Compute the value of P_1's parent node by hashing P_1 with its siblings that are contained in the PoF. Repeat this step until the value of the root node is computed.
4) Verify the signature of the root node's value and check whether the timestamp lies within the freshness window.

To verify the PoF $(T_{3,4}, T_{2,1}, Sign_{HO}(timestamp, T_{1,1}))$ for the statements $A_1, A_2, \ldots$ of user 3, compute $T_{3,3}$ by hashing $A_1, A_2, \ldots$. Afterwards, compute $T_{2,2}$ by hashing $T_{3,3}$ with $T_{3,4}$ from the PoF. Finally, compute the value $T_{1,1}$ by hashing $T_{2,2}$ with $T_{2,1}$, verify the signature $Sign_{HO}(timestamp, T_{1,1})$ and check whether $timestamp$ lies within the freshness window.

C. Login process

Occasio distinguishes between two login protocols. While the user decrypts the statements in the user-centric protocol, the SP decrypts them in the provider-centric protocol.

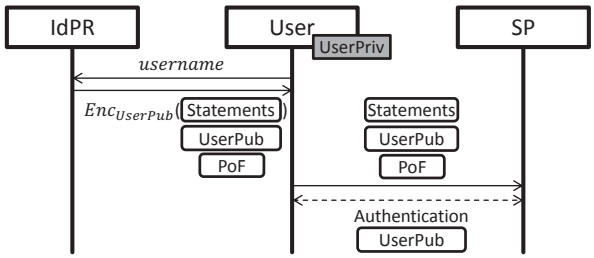

Fig. 3. Login via the user-centric protocol

The login process using the **user-centric protocol** is shown in Figure 3. First, the user that wants to access the service of an SP requests statements from the IdPR by providing the username. The IdPR then transmits the encrypted statements that it previously received from the HO along with the public key *UserPub* that is bound to the user. Furthermore, the IdPR generates and transmits a PoF for the transmitted statements and the public key based on the stored MHT. Upon receiving the answer from the IdPR, the statements are decrypted with the user's private key *UserPriv* and the user transmits the decrypted statements together with the public key and the PoF to the SP. The SP then verifies the correctness and freshness of the statements and the public key via the PoF (cf. Section V-B) and authenticates the user based on the transmitted public key. To achieve this, various authentication protocols can be used, which check whether the user is in possession of the private key that corresponds to the public key [7], [8]. The protocol choice can be made contingent upon the application. Once the user is authenticated, the SP can base its authorization decision on the conveyed statements.

The login process using the **provider-centric protocol** is shown in Figure 4. Contrasting to the user-centric protocol, the statements can only be decrypted by the participating SP in the provider-centric protocol. Therefore, the user has to transmit the name of the SP together with the username during a request for statements. The IdPR then returns the statements that have been exclusively encrypted for the named SP by the HO. Together with the user's public key *UserPub* and the PoF the statements are then forwarded to the SP. Afterwards, the SP can decrypt the statements using its private key *SPPriv*. Finally, the freshness of the statements and the user's public key is verified based on the PoF, the user is authenticated and an authorization decision is made by the SP.

A drawback of the provider-centric protocol compared to the user-centric protocol is that the HO has to encrypt the statements separately for each SP. This is necessary to prevent an SP (that might collaborate with the IdPR) from getting unauthorized access to statements that other SPs may view. Regardless of the number of SPs only one MHT has to be maintained as its leaf nodes are built from the plaintext statements rather than the encrypted statements. Thus, the maintenance of the MHT is independent from the protocol.

An advantage the provider-centric protocol has over the user-centric protocol is that SPs are able to poll for statements in absence of a user. This is particularly useful for deprovisioning purposes, i.e., the deallocation of a user's resources that is no longer authorized to use a service. In the user-centric

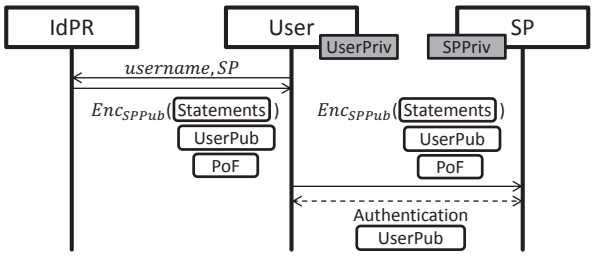

Fig. 4. Login via the provider-centric protocol

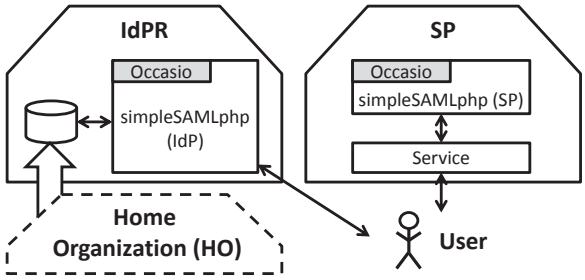

Fig. 5. Occasio implementation based on simpleSAMLphp

protocol this is not possible due to the fact that the SP is not able to decrypt the statements without the user's participation.

The presented protocols should be seen as a general overview of the steps that have to be performed during a login procedure and not as a "stand-alone" protocol. The protocols have to be integrated with other mechanisms such as SAML-Profiles ensuring IdPR-discovery, channel security, etc. In the following section, it will be exemplarily shown how to integrate the protocols with the SAML-Standard.

VI. Integration with SAML

In this section, it is shown how Occasio can be integrated with the SAML standard [3]. By integrating Occasio with the SAML Standard, many use cases that are already covered by SAML can be addressed. For instance, the SAML standard already supports federated access to web-based services [3] and can also be applied to federate access to SSH services [21]. Furthermore, it is implemented by widely deployed, extendable frameworks such as Shibboleth[3] and simpleSAMLphp.

The SAML standard specifies how identity data may be exchanged in a federated manner. Just as in the scope of this paper, it distinguishes between service providers and identity providers (IdPRs in case of Occasio). The SAML standard specifies *SAML-Assertions*, *SAML-Protocols* and *SAML-Bindings* that can be used to convey identity information about a user from the identity providers to SPs. Furthermore, *SAML-Profiles* subsume combinations of SAML-Assertions, SAML-Protocols and SAML-Bindings to use-cases. For instance, the WebSSO-Profile enables a user to access an external web service by authenticating to a web service of the HO.

The necessary adaptions to integrate Occasio with SAML are as follows. SAML already supports the encryption of statements (using so called *Encrypted Attribute Statements*). Thus, the HO can outsource pre-encrypted Encrypted Attribute Statements to the IdPR, which can insert them into SAML-Assertions upon request. Only the support for PoFs has to be added in order to integrate Occasio in SAML. This can be achieved by either manipulating the SAML standard itself or by encapsulating the PoF into a dedicated Attribute Statement. The prototype we implemented uses the latter alternative and leaves the SAML standard itself untouched. Thus, Protocols, Bindings and Profiles can be used in the accustomed way. For instance, our prototype leverages the WebSSO profile of SAML that allows web-based services to request statements on a user by redirecting him to the IdPR. Upon being provided the user's username, the IdPR issues the according stored encrypted statements as well as the PoF and automatically redirects the user back to the SP.

[3]http://shibboleth.net/

The usage scenario of the Occasio prototype is shown in Figure 5. Like simpleSAMLphp our prototype consists of the so-called IdP-component run by the IdPR and the SP-component run by the SP. Both components of the prototype are based on the according simpleSAMLphp components. As the support for Encrypted Attribute Statements was already implemented, we only needed to add the support to encapsulate the PoF into an Attribute Statement in the IdP-component and to verify the encapsulated PoF in the SP-component. Furthermore, to verify the authenticity of the user, the SP-component compares the client certificate used for the TLS handshake between the user and the SP with the public key that is contained in the statements of the HO. The interfaces of the simpleSAMLphp components have not been altered. Thus, using the Occasio prototype is transparent for services that have been relying on simpleSAMLphp before.

VII. Evaluation

A. Fulfillment of Requirements

In this section, Occasio is evaluated against the requirements we compiled earlier in Section III. Our approach enforces the authenticity of the user by accepted public-key based authentication protocols and the confidentiality as well as the correctness of the outsourced statements by common encryption and signature schemes. Occasio makes the assumption that these building blocks are secure. Furthermore, no identity information needs to be stored by the user and the SP.

However, the HO needs to provide an instance that continuously signs the root of the MHT to ensure the freshness of the statements. If the instance operated by the HO is not available, users can still log in as long as the root signature is valid, i.e., its timestamp lies within the freshness window (cf. Section V-B). Therefore, the size of the freshness window constitutes a control for balancing the degree of freshness and the time a HO may be unavailable without preventing users from logging on. For instance, in case of a freshness window of 3 hours an IdPR is able to deliver possibly revoked statements that are up to 3 hours old and the HO may be offline for 3 hours without any negative availability effects. Thus, the availability of the access control system is partly decoupled from the availability of the HO's signing component, leaving the HO a customizable time span to react before services become unavailable.

In terms of usability we require the user to utilize public-key based authentication protocols instead of passwords. This drawback is dampened by the fact that many services support and use this way of authentication already. For instance, SSH allows authentication via challenge-response based on a public key that has been previously deployed at the SSH server [7]. Furthermore, in case of web-based services the use of client certificates during the TLS handshake can be leveraged for authentication [8]. Nevertheless, we are planning to tackle the usability issue of not supporting passwords in future work.

To show that Occasio is capable of being integrated with existing standards, we described how it fits into the SAML standard. As encrypted attribute statements are already part of SAML, only the PoF support had to be added by encapsulating the PoF into an attribute. However, even if encrypted attribute statements are not supported by the standard yet, they could be encapsulated into regular attributes similar to the PoF.

To give an example of the integration effort to apply Occasio on existing services, we implemented Occasio based on the simpleSAMLphp framework. SPs can use our modified simpleSAMLphp framework in exactly the same way they use the unmodified simpleSAMLphp framework. Thus, while the SP needs to substitute the simpleSAMLphp framework with our modified version, Occasio is transparent for the service.

B. Performance

In this section, the performance penalties Occasio induces will be examined. The performance of Occasio comes in two flavors: The performance of the login process and the performance of applying changes to the outsourced statements. The overhead of frequently updating the root signature of the MHT to update the PoFs is negligible, as it consists of a single signing operation that even weak hardware can perform.

To measure the **performance of the login process**, we deployed our implementation of Occasio on a testbed that consisted of an IdPR (QuadCore 2.50GHz, 4GB RAM), an SP (OctoCore 2.00GHz, 4GB RAM) and a user machine (DualCore 2.93GHz, 4GB RAM). The IdPR stores the encrypted statements, the MHT and the timestamped signature of the MHT's root in a PostgresSQL[4] database. We measured a web-based login process that is typical for the WebSSO-Login of SAML and consisted of the following steps: First the user tries to access the protected resource of the SP. The user is then redirected to a discovery service to choose the identity provider (the IdPR in case of Occasio). After being redirected to the IdPR the user enters the username. Finally, the user is redirected to the SP with the SAML-Assertions (also containing the PoF), authenticates via client certificate (cf. Section III-B) and is granted access to the service. As a web-browser does not support the decryption of statements, we focused on the provider-centric protocol (cf. Section V-C). The only difference between the user-centric and the provider-centric protocol is that the user performs the decryption of the statements instead of the SP. As the computational overhead induced by decryption does not have to be performed by the SP in the user-centric protocol, the conducted measurements of the provider-centric protocol constitute a lower bound for the performance of the user-centric protocol.

[4]http://www.postgresql.org/

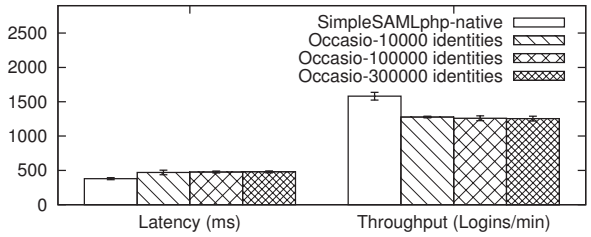

Fig. 6. Performance of the login process (95% confidence interval)

We used the performance measurement framework JMeter[5] to simulate users logging in. We measured both the latency and the total execution time of 10000 concurrently executed login procedures of uniformly random chosen users and repeated the experiment 10 times to get more significant results. We compared our Occasio implementation to an unmodified simpleSAMLphp deployment in the same testbed. The measurements of the simpleSAMLphp deployment have been conducted by measuring the performance of logging in static (i.e., hard coded) users. Thus, unlike in operational deployments of simpleSAMLphp, no additional overhead for retrieving identities from backend identity management frameworks is included in our measurements. Therefore, the measurements have to be seen as an upper bound of simpleSAMLphp's performance.

Every connection to both the IdPR and the SP is typically protected by TLS. In order to reduce the impact of the performance of the system we ran JMeter on, we used already established TLS sessions for our measurements of both the Occasio and the native simpleSAMLphp deployment. Thus, in the workload of the native simpleSAMLphp framework, 10000 TLS handshakes to the SP and 10000 TLS handshakes to the IdPR have been omitted. As Occasio needs exactly the same amount of TLS handshakes this is a viable simplification.

The results of our measurements are shown in Figure 6 for different numbers of identities administered by the HO. The overhead Occasio induces in terms of latency and throughput can be explained by the additional cryptographic overhead that has to be performed by the SP to decrypt the statements and to verify the PoF. While the overhead Occasio induces is not negligible, it is small enough to compensate it with more performing hardware on the side of the SP. Furthermore, Figure 6 shows that the number of outsourced identities has no visible influence on the performance of the login process. This indicates that the Occasio approach has a good scalability in terms of the number of maintained identities.

To measure the **performance of applying changes** on the outsourced statements we added a machine running the HO component of Occasio (DualCore 2.50GHz, 4GB RAM) to our testbed and issued requests to change the identity data of single users. By making use of SQL transactions it is ensured that users may log on while changes to the identity data are applied without generating an inconsistent PoF. We measured the provider-centric protocol for different numbers of SPs. In terms of performance, the user-centric protocol equals the provider-centric protocol for one SP, as the statements just have to be encrypted and stored once and both protocols do not differ in terms of maintaining the MHT. To provide a

[5]http://jmeter.apache.org/

Managed identities	Real scenario: Throughput (95% confidence interval) 1 SP (C/m)	5 SPs (C/m)	10 SPs (C/m)
10000	3178 (±6)	2825 (±3)	2012 (±1)
100000	2924 (±15)	2821 (±1)	2010 (±1)
300000	2758 (±16)	2737 (±9)	2005 (±1)
Managed identities	HO component's performance, no database updates 1 SP (C/m)	5 SPs (C/m)	10 SPs (C/m)
10000	4529 (±2)	2952 (±1)	2072 (±1)
100000	4524 (±2)	2951 (±1)	2067 (±1)
300000	4517 (±2)	2947 (±1)	2067 (±1)
Managed identities	Only database updates at the IdPR component 1 SP (C/m)	5 SPs (C/m)	10 SPs (C/m)
10000	5464 (±19)	5452 (±7)	5436 (±14)
100000	4862 (±10)	4854 (±10)	4851 (±9)
300000	4418 (±18)	4412 (±29)	4400 (±22)

TABLE I. CHANGES PER MINUTE (C/M) FOR THE REAL SCENARIO, FOR ENCRYPTING THE VALUES AT THE HO AND FOR STORING THEM AT THE IDPR

more accurate impression on where the overhead of Occasio is induced, we additionally measured the isolated performance of Occasio's HO component, i.e., without storing any data in the database. Furthermore, we measured the isolated performance of the IdPR component, i.e., the performance of just storing the encrypted statements and the MHT in the database. For each case we measured the total execution time of 10000 concurrently executed change operations applied on random identities and repeated the experiment 10 times.

The measured throughputs in changes per minute (C/m) are shown in Table I. The measurements of the real scenario show, that both the number of SPs and the maintained identities influence Occasio's performance of applying changes. The decline of the throughput when adding more SPs can be explained by the fact that the statements have to be encrypted by the HO for each SP and stored in the database of the IdPR. This observation is confirmed by the measurements that have been conducted for the HO component that just executes encryption and signing operations. The throughput of the IdPR component does not change much for more SPs as for a change operation just a single UPDATE operation has to be performed additionally for each SP and the overhead for updating the MHT does not depend on the number of SPs.

The measurements of the HO's performance without storing the results at the IdPR's database indicate that the induced overhead for the HO is nearly independent of the number of maintained identities. While the encryption of statements does not depend on the number of identities, only logarithmically more hashing operations are necessary to adapt the MHT. Nevertheless, the number of identities has an influence on the actual performance of Occasio. This is due to the overhead induced by storing the encrypted statements and the PoF in the database as the measurements of the IdPR component confirm. The number of operations to update the MHT logarithmically depends on the number of identities. Furthermore, update operations take more time for more maintained identities, as the utilized B-Tree database indexes grow. Interestingly, in the case of 10 SPs, the performance of Occasio's real performance does not seem to be influenced by the number of identities anymore. This is due to the fact that the computational power of the HO becomes a bottleneck so that the IdPR's database performance is not an influential factor anymore.

To apply changes on an already existing user or adding a new user, $\lceil log(n) \rceil$ nodes of the MHT have to be updated (where n is the number of entries). When removing users, a "gap" is induced into the MHT. To fill this gap, the last user is placed at the position of the removed user. Thus, an additional $\lceil log(n) \rceil$ nodes of the MHT have to be updated. To sum up, its to be expected that the performance of updating existing users matches the performance of adding new users. Furthermore, the throughput of the IdPR component for removing users is expected to be half of the throughput for updating existing users, as the double amount of MHT nodes has to be updated. However, the throughput of the HO component can be expected to be significantly higher, as no encryption needs to be performed to remove a user.

Notice that we measured Occasio on a specific testbed. In terms of the computational overhead for encryption and signing the performance of both login processes and applying changes can be increased by using more performing hardware or multiple load-balanced instances of either the IdPR, the HO or the SP. Furthermore, the potential IO-bottleneck of the underlying database can be omitted by partitioning the identities to multiple databases. In turn, the MHT has to be partitioned as well and the HO has to regularly refresh the root of each MHT. The signing of a reasonable number of MHT roots constitutes a negligible overhead.

VIII. CONCLUSION

In this paper, we presented Occasio, an approach to outsource identity providers while preserving the confidentiality, the correctness and the freshness of the outsourced identity data. Occasio requires a single trusted anchor that is lightweight and does not have to be continuously available to ensure the availability of the IAM system and therefore of relying services. It thus enables companies to securely operate a highly available access management based on untrusted external providers without maintaining a highly available in-house IT infrastructure. It has shown that outsourcing identity information is a good use case for Merkle Hash Trees. Occasio has been evaluated in terms of performance and scalability and was found to be useable for real scenarios. As Occasio does not require any trusted party to authenticate a user, the user must not use passwords for authentication. We are planning to address this usability issue as a next step.

REFERENCES

[1] M. Nabeel, E. Bertino, M. Kantarcioglu, and B. Thuraisingham, "Towards privacy preserving access control in the cloud," in *Proc. of the 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, 2011, pp. 172–180.

[2] M. Stihler, A. O. Santin, A. L. Marcon Jr., and J. d. S. Fraga, "Integral federated identity management for cloud computing," in *Proc. of the 5th International Conference on New Technologies, Mobility and Security (NTMS)*, 2012, pp. 1 –5.

[3] J. Hughes, S. Cantor, J. Hodges, F. Hirsch, P. Mishra, R. Philpott, and E. Maler, *Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0*, OASIS Std., 2005.

[4] D. Chadwick and G. Inman, "Attribute aggregation in federated identity management," *IEEE Computer*, vol. 42, no. 5, pp. 33 –40, 2009.

[5] R. C. Merkle, "Secrecy, authentication, and public key systems." Ph.D. dissertation, Stanford, CA, USA, 1979.

[6] U. Feige, A. Fiat, and A. Shamir, "Zero-knowledge proofs of identity," *Journal of Cryptology*, vol. 1, pp. 77–94, 1988.

[7] T. Ylonen and C. Lonvick, "The secure shell (SSH) authentication protocol - RfC 4252," 2006.

[8] T. Dierks and E. Rescorla, "The transport layer security (TLS) protocol version 1.2 - RfC 5246," 2008.

[9] H. Koshutanski, M. Ion, and L. Telesca, "Distributed identity management model for digital ecosystems," in *Proc. of the International Conference on Emerging Security Information, Systems, and Technologies (SECURWARE)*, 2007, pp. 132 –138.

[10] R. Ranchal, B. Bhargava, L. Othmane, L. Lilien, A. Kim, M. Kang, and M. Linderman, "Protection of Identity Information in Cloud Computing without Trusted Third Party," in *Proc. of the 29th IEEE Symposium on Reliable Distributed Systems*, 2010, pp. 368–372.

[11] P. Angin, B. Bhargava, R. Ranchal, N. Singh, M. Linderman, L. Ben Othmane, and L. Lilien, "An Entity-Centric Approach for Privacy and Identity Management in Cloud Computing," in *Proc. of the 29th IEEE Symposium on Reliable Distributed Systems*, 2010, pp. 177–183.

[12] S. Ruj, M. Stojmenovic, and A. Nayak, "Privacy preserving access control with authentication for securing data in clouds," in *Proc. of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, 2012, pp. 556 –563.

[13] E. Bertino, F. Paci, R. Ferrini, and N. Shang, "Privacy-preserving Digital Identity Management for Cloud Computing," *IEEE Data Engineering Bulletin*, vol. 32, no. 1, pp. 21–27, 2009.

[14] P. T. Devanbu, M. Gertz, C. U. Martel, and S. G. Stubblebine, "Authentic third-party data publication," in *Proc. of the IFIP TC11/ WG11.3 Fourteenth Annual Working Conference on Database Security: Data and Application Security, Development and Directions*, 2001, pp. 101–112.

[15] E. Mykletun, M. Narasimha, and G. Tsudik, "Authentication and integrity in outsourced databases," *ACM Transactions on Storage (TOS)*, vol. 2, no. 2, pp. 107–138, May 2006.

[16] M. Narasimha and G. Tsudik, "Authentication of outsourced databases using signature aggregation and chaining," in *Proc. of the 11th International Conference on Database Systems for Advanced Applications (DASFAA)*, 2006, pp. 420–436.

[17] M. Xie, H. Wang, J. Yin, and X. Meng, "Providing freshness guarantees for outsourced databases," in *Proc. of the 11th international conference on Extending database technology: Advances in database technology (EDBT)*, 2008, pp. 323–332.

[18] ——, "Integrity auditing of outsourced data," in *Proc. of the 33rd International Conference on Very large data bases*, 2007, pp. 782–793.

[19] T. Höllrigl, "Informationskonsistenz im föderativen Identitätsmanagement: Modellierung und Mechanismen," Ph.D. dissertation, KIT, Karlsruhe, Germany, 2011, [in German].

[20] R. C. Merkle, "A certified digital signature," in *Proc. on Advances in cryptology (CRYPTO)*, 1989, pp. 218–238.

[21] J. Köhler, S. Labitzke, M. Simon, M. Nussbaumer, and H. Hartenstein, "Facius: An easy-to-deploy SAML-based approach to federate non web-based services," in *Proc. of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2012.