Towards Privacy in Identity Management Dynamic Federations

Lucas Marcus Bodnar*, Carla Merkle Westphall†, Jorge Werner‡ and Carlos Becker Westphall§
Department of Informatics and Statistics, Network and Management Laboratory
Federal University of Santa Catarina - UFSC
88040-970, Florianópolis, SC, Brazil
Email: {*lucas, †carla, ‡jorge, §westphall }@lrg.ufsc.br

***Abstract*—Dynamic federations allow users to access new service providers on demand. This dynamic access adds risks to personally identifiable information (PII) of users, since there are untrusted service providers. The federated identity management is essential to preserve privacy of users while performing authentication and access control in dynamic federations. This paper discusses characteristics to improve privacy in the dissemination of sensitive data of users in dynamic federations, proposing privacy scopes to be agreed in dynamic associations (federation time) among service providers and identity providers. A prototype of the dynamic federation and scopes agreement was developed using OpenID Connect.**

Keywords–Privacy; Dynamic Federation; OpenID Connect.

I. INTRODUCTION

Privacy refers to the ability of the individuals to protect information about themselves. Around the world many laws are being proposed in order to take care of privacy in the digital environment [1]. When privacy is considered, the actual attributes and data used for identification and access, have to be released with the user consent [2].

Federated Identity Management (FIM) enables the use of identities across organizations and they integrate control of identity data besides seeking to secure users through security mechanisms. Some federated identity management systems used today are Shibboleth and OpenID Connect [3][4].

The concept of dynamic federation is arising. It means a dynamic association between Identity Providers (IdP) and Service Providers (SP), at the time of access, without previously knowing each other and without the need for acceptance of previous agreements or contracts [5][6]. This flexibility in the federation framework should consider aspects of a trust model [7]. With dynamic federation scenarios risks with the privacy increase, since there is no previous trust establishment. The lack of trust among involved entities is a problem, because an entity becomes a member of the federation in a dynamic way, without a previous contract agreement [5][6].

The works [5], [6] and [7] discuss dynamic federation architectures and they all use the Security Assertion Markup Language (SAML) standard to set up the federation. SAML does not allow creating a dynamic federation and although some works propose dynamic federation with SAML, most of them require changes in the specification flows [5]. Moreover, the majority of these works that propose dynamic federations do not address privacy [5][7].

The main goal of this work is to improve privacy in dynamic federation environments, proposing a solution for privacy scopes agreement in the dynamic association between SP and IdP, improving the user's privacy. This work uses the OpenID Connect to build the federation. The OpenID Connect (OIDC) uses JavaScript Object Notation (JSON) Web Tokens instead of SAML and unlike the SAML based federations, the OpenID Connect has a good support for dynamic associations.

This paper is organized as follows: Section II provides a background of the concepts; related works are described in Section III; Section IV presents the federation concept in OpenID Connect; the proposal of privacy scopes is presented in Section V and the initial results are described in Section VI. Finally, conclusions and future work are presented in Section VII.

II. BACKGROUND

This section presents a brief overview on privacy and identity management.

A. Privacy

Privacy is to choose when, how and to what extend personal information can be released to others. A way to achieve privacy is using the *Privacy-Enhancing Technologies* (PET) [8] that aim at protecting the individual's privacy by providing anonymity, pseudonymity, unlinkability and unobservability to the users. Anonymity is when the user cannot be identified in a set of users. Pseudonymity is when pseudonyms are used to identify the user instead of the real identifier. Both anonymity and pseudonymity are desirable principles and the user should be able to choose according to his preferences [2].

B. Basic Identity Management Concepts

Identity management can be defined as the process of creation, management and use of identities and the infrastructure that provides support for this set of processes [4][9].

Bertino and Takahashi, in [9], presented the roles that exist in an identity management system:

- Users – entities that want to access some kind of service or resource;
- Identity – set of attributes that can be used to represent a user, also called Personally Identifiable Information (PII);
- Identity provider (IdP) – has the responsibility to manage users PIIs, perform the authentication process and disseminate data to SPs;
- Service provider (SP) – delivers the resource/service desired by a user. It delegates the process of authentication to IdPs and usually is responsible for the authorization process that is performed using the disseminated set of attributes from the IdP.

Chadwick [10] defines that a federation is an association of SPs and IdPs.

There are tools that can be used to create federated environments; some use SAML to exchange data between IdPs and

 ISBN: 978-1-61208-450-3

SPs such as Shibboleth [11]; while others use JSON such as OpenID Connect protocol [12]. The OpenID Connect [12] was chosen to serve as the basis for our implementations because it is open source, it has a standard protocol, has a native support for dynamic associations and as it uses a lightweight message format (JSON), it fits for mobile environments.

Figure 1 presents how the OIDC workflow when unauthenticated users request a protected resource.

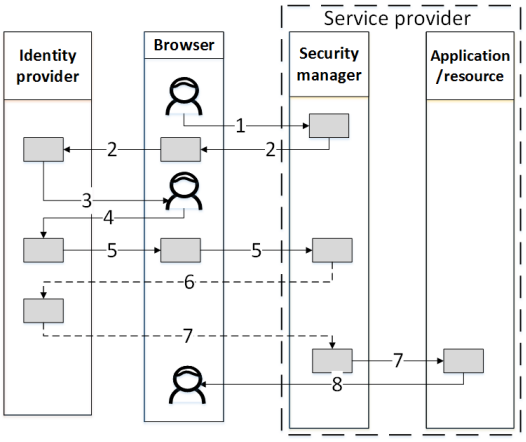

Figure 1. OpenID Connect work flow [13]

In Figure 1, the OpenID Connect interaction flow among user (browser), IdP and SP is shown. In step 1, the user requests access to the protected resource. In step 2, the SP requests user authentication which is performed in step 3. In step 4, the user can get a chance to consent to the release of data. In step 5, the IdP sends proof of authentication to the SP, that is validated in step 6. In step 7, the SP obtains user attributes from the IdP to release or not access to the resource in step 8.

C. Identity Management Concepts in OpenID Connect

The terminology used to represent the identity management concepts and technologies in OIDC is described in this section [3][14].

OpenID Provider (OP) – is the Authorization Server of OAuth 2.0 and is able to authenticate the final user and provide Claims to a relying party about the authentication and the final user. The OP is the IdP (*Identity Provider*) according to the traditional concept;

Relying Party (RP) – is the Client application of OAuth 2.0 that requires end-user authentication and Claims from OP. The RP is the SP (*Service Provider*) according to the traditional concept;

Claim – information about an entity, such as name and phone number;

JSON – is a lightweight, text-based and language-independent data exchange format;

JSON Web Token (JWT) – is a compact, URL-safe manner to represents claims to be transferred between two parties. A JWT is a string that represents a set of claims as a JSON object that is encoded using JWS (*JSON Web Signature*) or JWE (*JSON Web Encryption*), allowing claims to be digitally signed, MACed or ciphered;

Pairwise Pseudonymous Identifier (PPID) – Identifies the entity to a Relying Party. It cannot be correlated with the entity's PPID at another Relying Party.

III. Related works

The work described in [6] proposes an architecture according to the SAMLv2/ID-FF standards. The architecture considers federated identity management, the assessment of reputation and customization of the privacy preferences of user data. There is a lack of detail about interactions among parties. The work uses a reputation metric for trust establishment. In the case of privacy preferences, it only describes some possibilities for exchanging data, such as using IdP policies.

The work presented in [7] discusses a dynamic architecture to establish identity federations, based on a reputation exchange protocol. The protocol to collect data reputation is decentralized, storing the reputation of the providers.

The paper [15], which is a work developed in our security research group, presents an approach to address the issues involving privacy in the identity management systems. The proposal addresses three issues: the lack of PII control of users, the lack of models to assist users in data dissemination during the interaction and, the lack of user preferences guarantees on the SP side.

The work [16] identifies the privacy features relevant to authentication technologies and determines what privacy features are provided by each authentication technology. Some authentication technologies compared are OpenID Connect, OAuth, Shibboleth, Idemix pseudonym and U-Prove token.

The research described in [5] addresses the management of dynamic identity federations. The proposal allows users to create federations dynamically between two prior unknown organizations. The issue of trust is also a key component of the discussions. Trust means the way IdP and SP federate: *fully trusted* entities have a legal contract between IdP and SP; *semi-trusted* entities are the SPs that have been added dynamically to an IdP inside a federation by a user but without any contract between IdP and SP; *untrusted* entities are the IdP and SP added dynamically without the presence of any contract. A proof of concept is discussed using SAML federations and use-cases demonstrate the ideas proposed.

IV. OpenID Connect Federation

Federated identity management, according to [3], is a setup where identity is shared across domains.

An identity federation is a trust relationship between a service provider and identity provider, i.e., the service provider trusts the identity provider to authenticate the user and to provide user attributes (if necessary). The user tries to access a service provider located in different domains and is redirected to his own identity provider where he is authenticated and the result of this authentication process is sent to the service provider [17].

Third-party authentication is named federated authentication when a third-party, such as an identity provider and the service provider, belong to different organizations [16].

An OIDC Federation is an association among relying parties (RP) and OpenID Providers (OP). Even not considered a federation [18], a simple association between a RP and an OP of a single domain is also possible in OIDC.

 ISBN: 978-1-61208-450-3

Although in other federation platforms, trust is achieved through the acceptance of contracts, manual key or metadata exchanges in an explicit way [19], in OpenID Connect this association occurs in a more simplified way, through the registration of RP in OP [14][18].

This RP registration in OPs, such as Google or MitreID, is performed through a registration page and, in this moment, terms of use are accepted and combined. The authentication process of the own RP is also defined to give the RP the ability to receive user information from OP. So, an entity responsible for the RP provides the redirecting Uniform Resource Identifiers (URIs) and the other information required by the OP, such as the application name, website and description, for example. After supplying the information required, the RP receives from the OP a client identification (ID) and a password (ClientSecret) that will be used to authenticate the RP. The authentication of the RP can be executed in three ways [14][20]:

- In a default way, using the ID and the ClientSecret to the authentication via HTTP Basic authentication scheme;
- Including the ID and the ClientSecret in the request body;
- Using the ClientSecret as a shared key for creating a JWT using an HMAC SHA algorithm containing some required claim values.

At registration there is also the possibility of registering a public key to be used in the authentication, rather than the ClientSecret methods, i.e., the RP signs a JWT containing some required claim values with this key.

After the registration process, a trust relationship is established between the RP and the OP. In this work, we consider that when an OP allows an RP to register itself, a *federation* (or trust) can be established in OIDC.

In this way, an OP in a domain can federate with other RPs in other domains. So, a user can authenticate in his own domain and use RPs in other domains.

V. Proposal of Privacy in Identity Management Dynamic Federations

In federation systems, the identity providers are used for user authentication, to store collections of user's attributes and to manage these identities (in some systems, the attributes are managed by a separate part, called attribute provider) [4]. Generally, after the user authentication, the user gives the consent to the release of his identity and attributes to RP and thereafter, the OP sends these data to RP.

There is a lack of trust in dynamic associations since it is possible to associate any RP to an OP. It is necessary to control the sending of identities and attributes information to these RPs. Thus, to improve the privacy of dynamic environments, we propose a better control of users over his data, using privacy scopes that support anonymous access, use of pseudonyms and a scope that releases only partial attributes to RPs.

Once these privacy scopes are supported, it is still necessary to perform the combination and agreement of the scopes in a dynamic way, at the time of RPs and OPs association, to allow the use of services on demand by users with enhanced privacy.

Despite Shibboleth's support for pseudonyms and anonymity, it has a lack of support for dynamic federations [4][5]. OIDC does not support privacy scopes, but has a great support for dynamic associations.

A. Dynamic Federation in OpenID Connect

There is also other way of registering RPs on OPs in OIDC by specifically using the dynamic protocol known as *DynamicClientRegistration* [20]. It is possible to register dynamically an RP by sending RP's metadata to the OP. Despite using metadata as in other federation platforms, client metadata is used more dynamically, since the OP can change the information on it and is only used the RP's metadata, it is not necessary and indispensable (such as in Shibboleth/SAML federations) for an RP to have OP metadata.

Figure 2 shows the dynamic registering of RPs in OIDC.

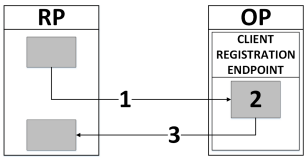

Figure 2. Dynamic Client Registration Flow

1. RP registration request - the RP sends an HTTP post message, with the content type application/json, to the client registration endpoint with parameters of client metadata. The RP chooses these registration parameters among the list of OP supported values. In OIDC standard only target/redirect URIs are required, other metadata are optional. In Figure 3, a post example is presented, with header and metadata;

2. RP registration - the OP assigns to this RP a unique client identifier, assigns a client secret or registers the public key of the RP and associates metadata sent in the request to the client identifier issued. The OP can still provide default values for any missing item in client metadata, reject or change any values to acceptable values.

3. RP registration response - with the success of the registration, the OP returns the code HTTP 201 Created, a JSON document (type of content is application/json) and all metadata registered for this RP, including the fields modified by the OP plus the RP identifier and if applicable, the client secret.

```
1 POST /register HTTP/1.1
2 Content-Type: application/json
3 Accept: application/json
4 Host: server.example.com
5
6 {
7   "redirect_uris": [
8     "https://client.example.org/callback",
9     "https://client.example.org/callback2"],
10  "client_name": "My Example Client",
11  "client_name#ja-Jpan-JP":
12     "\u30AF\u30E9\u30A4\u30A2\u30F3\u30C8\u540D",
13  "token_endpoint_auth_method": "client_secret_basic",
14  "logo_uri": "https://client.example.org/logo.png",
15  "jwks_uri": "https://client.example.org/my_public_keys.jwks",
16  "example_extension_parameter": "example_value"
17 }
```

Figure 3. Post of RP Registration

 ISBN: 978-1-61208-450-3

The metadata file is composed of information about RPs. This metadata file, in the static registration, is manually generated with the help of the registration page, but, in the dynamic way, it is sent within the HTTP post content. This client metadata file has mandatory parameters such as redirecting URIs or optional parameters as the client name, form of authentication and used algorithms [20].

Figure 3 presents an example of RP metadata: lines 1 to 4 are the header and lines from 6 to 17 describe metadata chosen by the own RP in the registration moment, of RP in OP.

All this OpenID Connect registration can be performed dynamically, without the need of previous agreements, without OPs needing to know RPs and even then, it is possible to establish this association at the time of access among different domains. We believe this process represents the concept of *dynamic federation*.

There are few references in the literature about dynamic federation. In [5], it has the following definition: "a dynamic federation with respect to the identity management is a business model in which a group of two or more previously unknown parties federate together dynamically without any prior business and technical contract with the aim to allow users to access services under certain conditions".

We think this concept is covered in OIDC through the *Discovery Protocol*, used to discover information about the OP, and the *DynamicClientRegistration Protocol*, which enables the RP to dynamically register with the OP. Thus, this work considers a *dynamic federation establishment* when the RP registration with the OP is done dynamically, across different domains.

As mentioned before, this dynamic registration can also be used in associations in the same domain and our study will work in this case too (even not being recognized as a federation and not being our focus to consider the same domain).

The great advantage of a dynamic association is to be able to associate the OP and RP in real time, without the need for prior agreements, and so, some problems can arise, mostly related to trust. OpenID Connect with dynamic registration considers the OP and RP as trusted and its use is a user's choice, since the OP and RP are able to match the settings required for proper communication between them.

B. Privacy in OpenID Connect

The user's PIIs are stored in the OP, and to ensure privacy, OIDC asks for user confirmation to release data to RP. After user confirmation, RP requests user data (PIIs) directly from the OP and obtains user identity and user's attributes.

To ensure better privacy in OIDC, some privacy scopes to access the RP are proposed in [15]. These scopes are agreed upon previously, between RP and OP, and among those supported scopes, the end user chooses the one that best fits with his privacy preferences. These scopes are:

Access with Total Attributes: Sending total attributes is the standard of OIDC. When the RP makes a request of the attributes and the user identification, the user allows or not the OP to send this information;

Access with Partial Attributes: The user chooses the attributes that he wants to send to the RP. The OP shows to the user the attributes list and the user selects the attributes to be released, helping users to assess and determine the data they want to send to the RP. In addition, the user can encrypt the selected attributes with his public key, in order to control the spread of data [13][21];

Access with Pseudonym: The RP receives the authentication data, the pseudonym of the user and also can receive some data that cannot be linked to user identification. The use of pseudonyms maintains the privacy of the users, and in the RP, the user is identified by his nickname. The identification of users is only permitted in the original OP, in case of legal request, for auditing purposes. The nickname must be issued dynamically through use of random numbers linked to the user identifier. Thus, attackers must have difficulty in correlating historical data with the user's identity. The OIDC recommends the use of a *Pairwise Pseudonymous Identifier* (PPID) to protect the user from a possible correlation among RPs [14]. To implement access with pseudonyms, in this work, before sending the ID Token to the RP, the OP maps the subject identifier of the user (named sub claim), with a random nickname for each RP. To implement pseudonyms, a scope claim is also added, using the parameter *pseudonym_profile*. Besides, the RP can only request user data that cannot be linked to user identification;

Access with Anonymity: The user accesses the RP without any type of identification that can compromise his privacy. The user can access the resource after the authentication on OP. The RP receives the authentication data but without user identifier or PIIs. The RP receives the authentication data, with information about the authentication, like identification of RP that the authentication was issued, identification of the OP that issued the authentication, authentication expiration time and other data, but without user identifier or PIIs. With that, the system avoids the identity correlation attacks or linkability of users' data. There are many projects used to achieve anonymity [4], like U-Prove, Idemix and P-IMS. In this work, we use the OIDC flow, adding the scope claim of type *anonym_profile* and without the sub claim, PIIs or identification data of the users. So, without any information about the user, we achieve the concept of anonymity. Thus, there is no reference of user identification, only a proof of authentication and non-linkable data, ensuring anonymity of the user in the RP. This kind of anonymity is interesting in cases that, to access certain services, the user has to prove that he belongs to a certain domain. For example, in university federations, the user has to prove that he belongs to the federation to access a service.

VI. Dynamic Agreement of Scopes and Initial Results

A prototype of the proposed solution was developed based on an identity management system running on a cloud environment. The identity management system chosen was the OpenID Connect, since it is a highly detailed framework, documented and extensible. We used the code developed by the specification under the name MITREid Connect, available in Java by MIT (Massachusetts Institute of Technology) [22].

In this work, the aim is to find a way to allow a combination of additional scopes (anonym, pseudonym and partial attributes) in the dynamic registration. Thus, a better privacy in dynamic federations is achieved.

Previously, the implementation of scopes in each OP and RP was necessary, to support these additional scopes. So, we

 ISBN: 978-1-61208-450-3

propose that in the registration of RP on OP, the combination of scopes have to be performed. Thus, at the moment of registration, using the DynamicClientRegistration protocol, the scopes supported by the RP are sent along with the Client Metadata, as shown in Figure 4.

```
1 POST /register HTTP/1.1
2 Content-Type: application/json
3 Accept: application/json
4 Host: server.example.com
5
6 {
7   "redirect_uris": [
8     "https://client.example.org/callback",
9     "https://client.example.org/callback2"],
10  "client_name": "My Example Client",
11  "client_name#ja-Jpan-JP":
12     "\u30AF\u30E9\u30A4\u30A2\u30F3\u30C8\u540D",
13  "token_endpoint_auth_method": "client_secret_basic",
14  "logo_uri": "https://client.example.org/logo.png",
15  "jwks_uri": "https://client.example.org/my_public_keys.jwks",
16  "example_extension_parameter": "example_value"
17  "scopes_supported": [
18    "partial_attribute_profile"
19    "pseudonym_profile"
20    "anonym_profile"]
21 }
```

Figure 4. Client Metadata example with Privacy Scopes

In Figure 4, lines 17 to 20 are added with the scopes supported by the RP and this metadata will be sent to the OP, to make the registration of the RP.

Thus, OP will check the scopes supported by RP and itself, and will register the RP with the scopes supported by both. For example, if the RP supports the scopes of partial attributes, pseudonym and anonym, it will send the scopes via metadata to the OP and, if the OP only supports partial attributes and pseudonym scopes, it will return the metadata only with the partial attributes and pseudonym scopes, which is supported by both. If the OP return no scopes, the RP will know that the OP does no support privacy scopes, and so, the default scope of OIDC will be used.

After the scopes agreement during RP dynamic registration on OP, it is necessary for the user to choose the scopes in order to use a service, performing the following steps:

1) On a user's access, the RP shows a WAYF (Where Are You From) page, where the user's OP is inserted.
2) The RP will verify if the OP is already registered. If already registered, the RP will go to step 3. If not, it will register the OP, agreeing to the privacy scopes supported by both through DynamicClientRegistration protocol.
3) The RP requests user authentication, it redirects the user to the OP to the authentication process.
4) The OP asks the user about his credential to perform the authentication process.
5) After the authentication process, the OP asks the user what is the desired privacy scope (total, partial, pseudonym or anonym), based on the privacy scopes supported by both RP and OP, as shown in Figure 5. The OP also informs the user about the chosen scope to access the RP, the attributes that will be sent to RP, the reason of the data dissemination and obtains the permission to release these attributes to the RP.
6) The user (browser) is redirected to the RP with a ticket to confirm the authentication process in the OP.

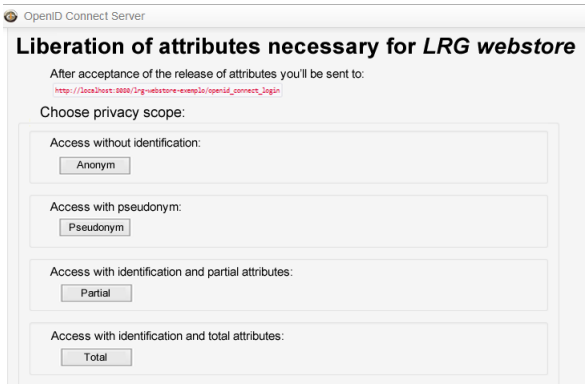

Figure 5. Privacy scope selection in OP

7) The RP intercepts the request, validates the proof of authentication and obtains an access token to gather some extra data about the user. In the case of scopes which do not allow sending PIIs: a) it is possible that no user attributes are obtained; b) only attributes that cannot be linked to user's identification are obtained.
8) The RP calls the OP with the access token to get attributes required to perform the RP's desired service, but again, only using attributes allowed by the user according to the privacy scope used.
9) The desired service/resource is delivered.

There was no change in the standard OIDC flow. Extensions were developed in the execution of OIDC actions, performed in steps 2, 4 and 5. In step 2, there is a dynamic agreement of scopes; in step 4, the user chooses the desired privacy scope to access the service/resource; in step 5, the user consents to release attributes according to the chosen privacy scope. So, it is important that the OP knows which attributes can be sent to the RP depending on the scope and the user should be aware about the usage of his data. For example, it is not allowed to send user PIIs on pseudonymity or anonymity scopes.

VII. Conclusion and Future Works

Our contribution in this work was to improve privacy in dynamic federations, proposing privacy enhancing technologies that use the dynamic association support to federate in OpenID Connect. So, users have a better control over their identity and attributes. An extension of OIDC access profiles was proposed, to add different levels of privacy. Considering [4] and [16], we can conclude that these developed extensions improved properties of OpenID Connect such as pseudonymity, anonymity, unlinkability, undetectability and confidentiality. With our proposed extension we achieve pseudonymity, anonymity and unlinkability in OIDC, properties that did not exist in OIDC according to [16].

The problem of agreeing to these privacy scopes dynamically was solved along with the DynamicClientRegistration protocol, through which these scopes were combined using the RP metadata, without any changes in the dynamic registration flow of OpenID Connect.

Our work combines the concept of dynamic federation with privacy features proposed in [15] and how to agree to these concepts dynamically. With that, our work differs from [5] and

 ISBN: 978-1-61208-450-3

[7] by defining and applying theoretical and practical concepts of privacy in dynamic federations.

The paper of Sanchez et. al. [6] also works with privacy in dynamic federations, building a dynamic federation model with a privacy layer and a trust layer. In contrast, our work extends a protocol already implemented and used by a large number of organizations, the OpenID Connect, without modifications in its specifications.

The following related works present a metric (reputation or risk) to measure the trust of the RPs: [6][7][15]. Different levels of trust in dynamic federations are cited in [5].

Another difference compared to related works, is that we have developed a prototype implementation using OpenID Connect, that works with JSON instead of SAML, which makes it easier to use in mobile environments and offers native support to dynamic associations. So far, the work has not changed the flow of interaction between RP and OP. In this way, new threats will not happen because of our model.

However, a limitation of this work is that it has an initial prototype and it lacks tests and measurements of how the federation will behave. It also lacks a trust level model.

We plan as futures works: a) to implements levels of trust (e.g., considering metrics of risks and reputation); b) to test the scalability of the federation; c) to study other forms to ensure user privacy, especially after user data is released to the RP.

REFERENCES

[1] European Parliament and the Council of the European Union, "Directive 95/46/ec of the european parliament and of the council," [retrieved: January, 2016]. [Online]. Available: http://ec.europa.eu/justice/policies/privacy/docs/95-46-ce/dir1995-46_part1_en.pdf

[2] A. Pfitzmann and M. Hansen. Anonymity, unlinkability, unobservability, pseudonymity, and identity management-a consolidated proposal for terminology. [retrieved: October, 2015]. [Online]. Available: http://dud.inf.tu-dresden.de/literatur/ (2010)

[3] G. Alpár, J. henk Hoepman, and J. Siljee, "The identity crisis security, privacy and usability issues in identity management," 2011, [retrieved: November, 2015]. [Online]. Available: http://arxiv.org/abs/1101.0427

[4] E. Birrell and F. Schneider, "Federated identity management systems: A privacy-based characterization," Security Privacy, IEEE, vol. 11, no. 5, Sept 2013, pp. 36–48.

[5] S. Ferdous and R. Poet, "Managing Dynamic Identity Federations using Security Assertion Markup Language," Journal of theoretical and applied electronic commerce research, vol. 10, 05 2015, pp. 53 – 76.

[6] R. Sanchez, F. Almenares, P. Arias, D. Diaz-Sanchez, and A. Marin, "Enhancing privacy and dynamic federation in idm for consumer cloud computing," Consumer Electronics, IEEE Transactions on, vol. 58, no. 1, February 2012, pp. 95–103.

[7] P. Arias Cabarcos, F. Almenárez, F. Gómez Mármol, and A. Marín, "To federate or not to federate: A reputation-based mechanism to dynamize cooperation in identity management," Wirel. Pers. Commun., vol. 75, no. 3, Apr. 2014, pp. 1769–1786.

[8] Y. Wang and A. Kobsa, "Privacy-enhancing technologies," 2008, [retrieved: December, 2015]. [Online]. Available: http://www.cs.cmu.edu/afs/cs/Web/People/yangwan1/papers/2008-Handbook-LiabSec-AuthorCopy.pdf

[9] E. Bertino and K. Takahashi, Identity Management: Concepts, Technologies, and Systems. Norwood, MA, USA: Artech House, Inc., 2010.

[10] D. W. Chadwick, "Federated identity management," in Foundations of Security Analysis and Design V. Springer, 2009, pp. 96–120.

[11] Shibboleth, "What's shibboleth?" 2014, [retrieved: July, 2014]. [Online]. Available: https://shibboleth.net/about/

[12] OpenID, "Welcome to openid connect," 2014, [retrieved: March, 2015]. [Online]. Available: http://openid.net/connect/

[13] R. Weingärtner, "Dissemination control of sensitive data in federated environments," Master's thesis, UFSC, Brasil, 2014.

[14] N. Sakimura, J. Bradley, M. B. Jones, B. de Medeiros, and C. Mortimore, "Openid connect core 1.0," 2014, [retrieved: August, 2015]. [Online]. Available: http://openid.net/specs/openid-connect-core-1_0.html#ClientAuthentication

[15] J. Werner, C. M. Westphall, R. Weingartner, G. A. Geronimo, and C. B. Westphall, "An approach to idm with privacy in the cloud," in Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on, Oct 2015, pp. 168–175.

[16] F. Corella and K. Lewison, "Privacy postures of authentication technologies," in The Internet Identity Workshop, ser. IIW 2013, Mountain View, CA, 2013, [retrieved: September, 2015]. [Online]. Available: http://pomcor.com/techreports/PrivacyPostures.pdf

[17] D. Chadwick, K. Siu, C. Lee, Y. Fouillat, and D. Germonville, "Adding federated identity management to openstack," Journal of Grid Computing, vol. 12, no. 1, 2014, pp. 3–27.

[18] D. Hardt, "Rfc6749-the oauth 2.0 authorization framework-revision," 2012, [retrieved: October, 2015]. [Online]. Available: http://tools.ietf.org/html/rfc6749

[19] D. R. D. Santos, T. J. Nascimento, C. M. Westphall, M. A. P. Leandro, and C. B. Westphall, "Privacy-preserving identity federations in the cloud: A proof of concept," Int. J. Secur. Netw., vol. 9, no. 1, Feb. 2014, pp. 1–11.

[20] N. Sakimura, J. Bradley, M. B. Jones, B. de Medeiros, and C. Mortimore, "Openid connect dynamic client registration," 2014, [retrieved: September, 2015]. [Online]. Available: http://openid.net/specs/openid-connect-registration-1_0.html

[21] R. Weingärtner and C. M. Westphall, "Enhancing privacy on identity providers," in SECURWARE 2014, Portugal, Nov 2014, pp. 82–88.

[22] MIT, "Mitreid connect," 2014, [retrieved: October, 2014]. [Online]. Available: https://github.com/mitreid-connect/OpenID-Connect-Java-Spring-Server

 ISBN: 978-1-61208-450-3