Tamper-Evident User Profiles for WebID-Based Social Networks

Stefan Wild, Falko Braune, Dominik Pretzsch, Michel Rienäcker, and Martin Gaedke

Technische Universität Chemnitz, Germany
{firstname.lastname}@informatik.tu-chemnitz.de

Abstract. Empowering people to express themselves in global communities, social networks became almost indispensable for exchanging user-generated content. User profiles are essential elements of social networks. They represent their members, but also disclose personal data to companies. W3C's WebID offers an alternative to centralized social networks that aims at providing control about personal data. WebID relies on trusting the systems that host user profiles. There is a risk that attackers exploit this trust by tampering user profile data or stealing identities. In this paper, we therefore propose the IronClad approach. It improves trustworthiness by introducing tamper-evident WebID profiles. IronClad takes protective measures to publicly discover malicious manipulation of profile data. We exemplarily implement IronClad in an existing WebID identity management platform known from previous work.

Keywords: Identity, WebID, Linked Data, Social Networks, Trust, Security, Data Integrity, Protection, Tamper Detection.

1 Introduction

Social networks have become crucial elements in modern society. They allow people to connect, communicate, and express themselves on a global scale. Joining social networks usually requires creating user profiles. Users therefore need to entrust personal information to network providers which escrow the information for them. This enables identification, eases discovery, and digitally represents the user. However, it also creates another copy of the user's digital identity. In today's ecosystem of social networks each copy needs to be maintained separately. This makes exchange and update of personal data across different domains and organizational boundaries difficult. Thus, centralized social networks like Twitter and Facebook gave rise to customer lock-in for millions of users [15].

For avoiding such walled gardens, the W3C devised WebID. As an open, universal, and decentralized identification approach, WebID allows users to be their own identity provider and establish their own personal social networks [11]. With WebID, users are enabled to manage their personal information at a self-defined place. They can also employ their WebID identities for global authentication.

WebID consists of three interrelated artifacts, which are illustrated in Figure 1: The *WebID URI* is a unique identifier referring to an agent. While an agent is

S. Casteleyn, G. Rossi, and M. Winckler (Eds.): ICWE 2014, LNCS 8541, pp. 470–479, 2014.

typically a person, it can also be a robot, group or any other entity that needs to be identified. The *WebID certificate* is a common X.509v3 client certificate. It includes a public key and a WebID URI linking to the WebID profile. The *WebID profile* is a resource containing personal information about the identity owner. Personal information are described in an extensible and machine-readable way using Linked Data. Each WebID profile also stores public keys. They are used along with the corresponding WebID certificate for an ownership-based authentication defined by the WebID protocol.

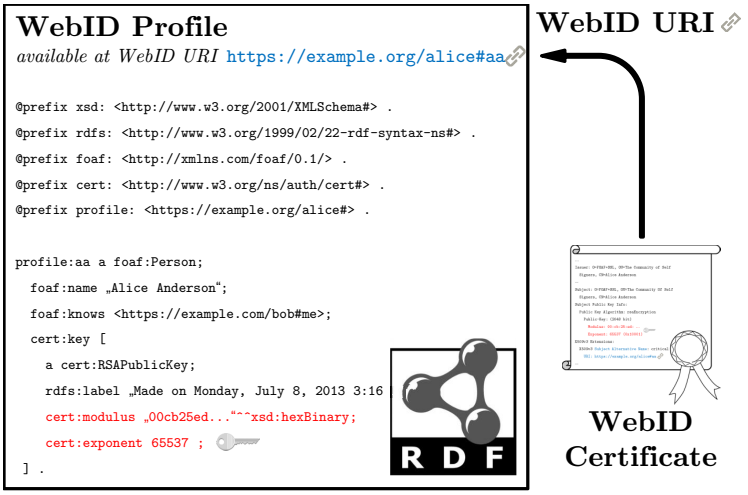

Fig. 1. Artifacts in WebID

Problem. Despite all advantages, being an identification mechanism that does not rely on authorities makes WebID vulnerable to attacks on user identities. WebID allows users to host their profiles at arbitrary locations. Yet it does neither ensure nor verify data integrity of user profiles. Experienced users know how to set up and protect a system storing their WebID profile. Inexperienced users, however, do not know this and would probably prefer using third-party managed services for hosting. Consequently, inexperienced users must trust third parties to not accessing and tampering their profile data [6]. The trustworthiness of managed services can hardly be assessed or guaranteed [5]. This shifts the problem from trusting identity providers [4] to trusting cloud storage providers. Requestors therefore depend on external means to decide whether to trust profile data. Obtaining write access to a WebID profile would enable an attacker to tamper user data stored inside [5]. Tampering user profile data, e.g., changing the e-mail address or replacing social contacts, could interfere with further transaction. Having a chance to add a public key to the identity owner's WebID profile would allow constructing a client certificate with the corresponding private key. Attackers could then use such certificate for authenticating to services as the identity owner without her knowledge and intent.

Objective. To increase trustworthiness in WebID-based social networks, we aim at providing a means for users to publicly detect tampering of profile data. For achieving this objective, we propose the novel IronClad approach for tamper-evident[1] WebID profiles. We therefore provide contributions for:

- Signing WebID profile data,
- Discovering WebID identity theft, and
- Verifying WebID profile data integrity.

Impact. Achieving the objective would allow for storing WebID profiles in potentially harmful environments, reducing the entry barrier for inexperienced users and, thus, contributing to increase the overall security and adoption of the WebID identity mechanism. Otherwise, users still risk to retrieve WebID profile data that is not in accordance with the identity owner's original intention.

The rest of the paper is organized as follows: Section 2 discusses related work. Section 3 describes and exemplarily demonstrates the IronClad approach for providing tamper-evident WebID profiles. Section 4 evaluates the approach. Section 5 concludes the paper.

2 Related Work

Literature to file systems and database systems broadly deals with the topic of ensuring data integrity. This discussion of related work focuses on the applicability of integrating features for detecting tampering attacks to Web-based systems only. To structure the discussion, we pay particular attention to interoperability, applicability and accessibility of tamper-evident features.

Centralized social networking platforms like Facebook and Google+ enable users to create views on their profile data. Such views can conceal sensitive data to external parties, e.g., groups of requestors. When targeting profile data disclosure or malicious manipulation, they are, however, inapplicable to detect internal read/write attacks without further ado, as described by Feldman et al. in [6]. Through establishing decentralized social networks, WebID distributes this problem to systems that host the profiles. Contrary to centralized social networking platforms, Web Access Control[2] (WAC) facilitates securing resources in a decoupled and decentralized way. However, WAC also focuses on access protection and not on data protection [14]. Even though protection of personal data could be accomplished through encryption, this is inappropriate in WebID. Profiles have to be, at least partially, accessible for authentication of identity owners and for queries of requestors. It would be also required to either distribute keys for decryption to an unknown number of potential requestors or establish central authorities for key management, which does not conform to WebID's idea of focusing on individuals.

[1] "Tamper-evident" is commonly defined as a means for making unauthorized access to a protected object easily detectable.

[2] `http://www.w3.org/wiki/WebAccessControl`

With regard to detect tampering of profile data, WebID shares similar disadvantages with other identity management systems like OpenID or Mozilla Persona [8]. OpenID implements only limited handling of personal attributes [7], whereas Persona is not designed for attaching profile data to an identity in a holistic way [1]. In contrast, WebID allows flexibly extending profile data, i.e., add cryptographic signatures. Such extension is not applicable to many social networking platforms and identification systems due to their centralized, closed or restricted handling of user profile data.

Public keys and signatures must also be protected from manipulation in order to provide sound proof of the identity owner's intent. This could be accomplished by a public key infrastructure (PKI) involving certificate authorities (CA) or a Web of Trust (WoT). A PKI based on CAs represents a centralized trust model that uses hierarchically organized authority chains [2]. WebID allows for adapting this model, e.g., similar to signing WebID certificates by a trusted third party instead of the identity owner himself[3]. However, we do not want to impair WebID's decentralized approach of involving and empowering individuals instead of authorities. By contrast, the WoT concept represents a flat hierarchy only relying on individuals [2]. It needs member discovery and makes updating public keys and signatures difficult due to their necessary distribution and inclusion in other data stores, e.g., user profiles.

3 Tamper-Evident WebID Profiles through IronClad

In order to detect tampering of WebID profile data, we created IronClad. It is based on the principle that only identity owners should be enabled to change their profile data in a sustainable way. For ensuring requestors that all WebID profile data is what was intended by the identity owner, IronClad incorporates three main activities: signing profile data, storing/retrieving signatures, and verifying data integrity of profiles. They are according to our key contributions. We illustrated them using BPMN in Figure 2 and describe them in the following.

3.1 Operations Supported by IronClad

Signing WebID Profile Data. By signing the WebID profile, the identity owner (cf. top of Figure 2) proves that personal data stored in his profile is sound and was not changed by another party. In order to avoid signing tampered data, the data integrity of the WebID profile needs to be checked (cf. ① in Figure 2) prior to updating relevant data (cf. ②) and creating signatures (cf. ③). Algorithm 1 specifies in pseudo code notation how IronClad creates signatures of WebID profile data. Our approach uses the RDF graph representation of a WebID profile for computing hash values independent from specific data serializations, e.g., RDF/XML or Turtle. To address different orders of RDF

[3] In WebID, certificates are usually self-signed as the trust does not rely on the public key, but on the WebID URI linked resource storing the public key, which we want to protect against tampering.

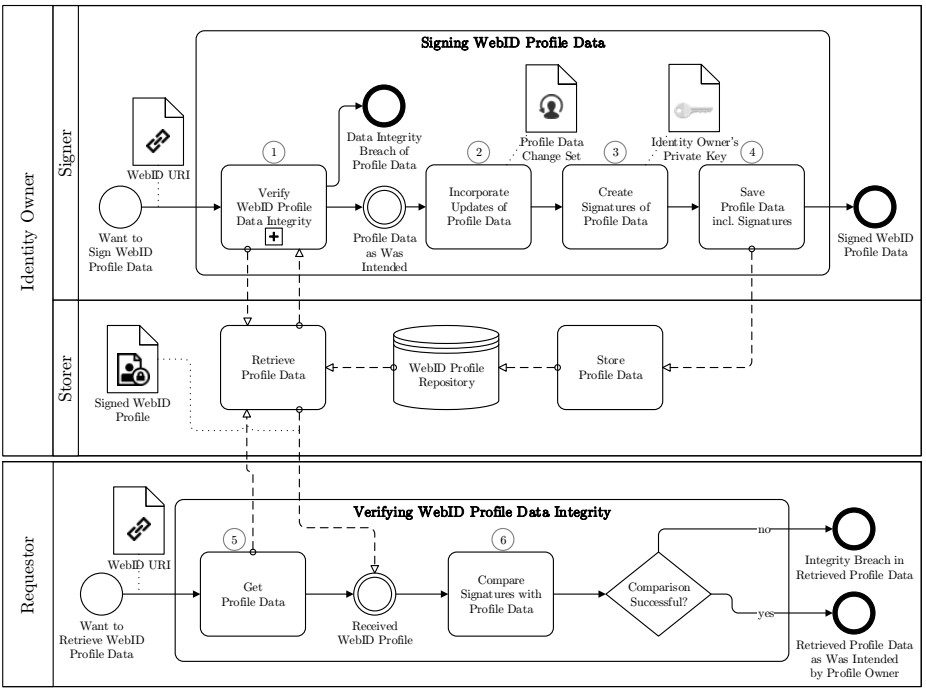

Fig. 2. Big picture of IronClad: Signing, storing, and verifying data integrity of profiles

triples and blank nodes, IronClad performs a canonicalization. We therefore utilize the One-Step Deterministic Labeling method proposed by Carroll in [3] and the methodology described by Tummarello et al. in [12].

To avoid disclosing the identity owner's private key to a third party, the signing process is split into a server and a client side part. IronClad's server side computes hash values of each *minimal self-contained graph* [12] found in the graph representation of the WebID profile. It combines all hash values to a signing request afterwards (cf. lines 4 to 8). The client side part analyzes this request and signs the content. It creates the signature through encrypting each hash value with a private key (cf. lines 9 to 12). The identity owner selects the corresponding private key beforehand. IronClad supports client side signing by a tool. It transforms a signing request into a signed response, which is then sent back to the server side.

Once received by the server side (cf. lines 13 to 16), the signed response containing the signatures is verified. Provided that the verification was successful, IronClad stores the signatures in the identity owner's WebID profile in ④. When storing (cf. middle of Figure 2), IronClad applies the method proposed by Sayers and Eshghi in [10]. This method closely links the public key of the profile with the WebID URI. Thus, it assists in detecting attacks that aim at removing profile data and signatures.

Discovering WebID Identity Theft. Following the principle of empowering individuals instead of authorities, we could not solely rely on attaching signatures to profile data[4]. IronClad creates a binding between the public key and the WebID URI. Thus, it ensures that this key cannot be changed without losing personal relationship data such as incoming social connections expressed via `foaf:knows` WebID URIs. Having the public key stored in the WebID URI allows detecting the same key inside the profile. This facilitates not only discovering identity theft done by malicious key manipulation, but also using the public key for signature verification. The length of the public key, e.g., 2048-bit, makes it inconvenient to store it directly inside a WebID URI. Therefore, IronClad uses the SHA-1 hash value of the public key instead.

Algorithm 1. Creating Signatures of User Profile Data

```
   Input: WebID URI u, Private Key key
   // on server side
 1 get WebID profile p from u;
 2 get RDF graph g from p;
 3 apply canonicalization on g using [3] and [12];
 4 repeat
 5 |   delete minimal self-contained graph msg from g;
 6 |   create hash value h of msg; // currently uses SHA-1
 7 |   add h to client request req;
 8 until g is empty;
   // on client side to avoid private key disclosure
 9 foreach hash value h in client request req do
10 |   create signature sig by encrypting h with key;
11 |   add sig to server response res;
12 end
   // on server side
13 foreach signature sig in server response res do
14 |   if signature sig is invalid then stop
15 end
16 add all signatures in server response res to graph g;
```

Verifying WebID Profile Data Integrity. To make signed WebID profiles easily verifiable for requestors, we integrated the verification process into the WebID authentication routine. It is triggered when the WebID profile has been loaded. For verifying signed profile data (cf. bottom of Figure 2), IronClad receives the WebID profile via the WebID URI in ⑤. It tries to detect a plausible public key[5] inside the profile. Such public key has to correspond to the hash value stored in the WebID

[4] By gaining access to the system storing the WebID profile, an attacker could tamper identity data and manipulate signatures stored in the profile. Due to this vulnerability to attacks, an external authority would be required to provide proof of correctness.

[5] A public key with a common length, e.g., 2048 bit.

URI. Having found the public key, IronClad computes hash values of WebID profile data as mentioned in the signing process. It then compares the hash values with the hash values retrieved by decrypting the signatures using the public key, as indicated by ⑥. The data integrity of WebID profiles cannot be guaranteed in case any detection or verification step has failed. Handling failed verifications depends on the scenario and authentication target. This is not a part of the IronClad approach and, thus, needs be to addressed separately.

3.2 Implementation and Demonstration of IronClad

For showcasing the approach, we exemplarily implemented IronClad in the Sociddea WebID identity provider and management platform proposed by Wild et al. in [13]. It is important to mention here that IronClad is not limited to a specific platform. That is, it is possible to apply the approach in arbitrary WebID identity providers, management platforms or authentication methods. Thus, IronClad is line with the idea of decentralized social networks.

In Sociddea, IronClad is optionally applied when creating new WebID identities. It is furthermore used for signing and verifying profile data hosted in Sociddea's ecosystem. Figure 3 illustrates the visual representation of a tamper-evident WebID profile in four different scenarios. While the figure shows tampering by changing a personal attribute of the identity owner, data integrity breaches through adding or removing RDF triples are also detected by IronClad. In addition to visually highlighting malicious data manipulations, identity owners and service providers also benefit from tamper-evident WebID profiles during user authentication and requests for profile data.

Try It Out. For a live demonstration, screen casts, and further information about IronClad and Sociddea please visit:
`http://vsr.informatik.tu-chemnitz.de/demo/sociddea/`

4 Evaluation of IronClad

While the acceptance and handling of tamper-evident WebID profiles is a planned subject of a larger empirical investigation, this evaluation discusses to which extent IronClad takes the criteria into account we used for analyzing related work. From a theoretical point of view, we think that such study therefore allows for determining how well IronClad achieves the objective defined in Section 1.

WebID profiles depend on RDF for describing an identity owner's personal attributes in a machine-readable way. Appropriate RDF-based vocabularies enable describing and interlinking new contents. This facilitates extending WebID profiles by additional RDF triples. It is consequently well applicable to represent and associate signatures to personal attributes.

As the IronClad approach does not involve encrypting user profile data, it does not impair the accessibility of tamper-evident WebID profiles. Both common and tamper-evident WebID profiles share similar characteristics. While tamper-evident WebID profiles consist of additional RDF-based signatures, existing personal data

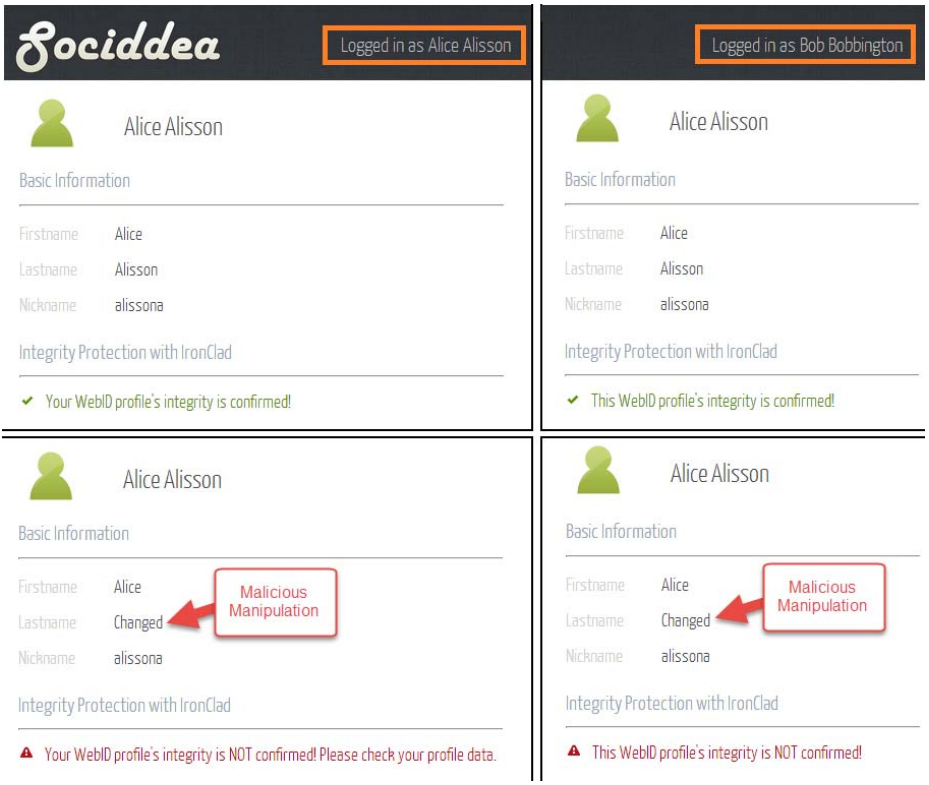

Fig. 3. Exemplary implementation of IronClad for tamper-evident user profiles with visualized results for: Identity owner Alice's view on her valid profile data (top/left), "Lastname" changed - tampering detected (bottom/left), requestor Bob's view on Alice's valid profile data (top/right), Bob's view on Alice's tampered profile (bottom/right)

remains untouched. In IronClad, signatures are loosely coupled statements about personal data. Removing all signature RDF triples would reveal the original payload of a WebID profile. The hash value included in the WebID URI, however, indicates that the corresponding WebID profile is tamper-evident. Considering an attacker would remove signatures stored in a tamper-evident WebID profile, there is still the WebID URI indicating that the WebID profile has to be data integrity protected. While there are other ways of using hash values in URIs [9], IronClad just appends them to common WebID URIs for the sake of simplicity and conformity. Such verification for tamper-evidence is part of the proposed extension of the WebID authentication sequence. Even though signatures and personal data contained in tamper-evident WebID profiles are accessible per se, view filters can assist in concealing particular RDF triples, as described by Wild et al. in [14]. For example, excluding all signatures might be beneficial for presenting profile data to users.

Through directly operating on the RDF graph representing a WebID profile, IronClad is compatible to different orders and structures of RDF triples, and diverse types of serialization. This allows for dealing with the high heterogeneity prevailing in this context. Despite all advantages, we also identified some issues restricting the interoperability and compatibility of our approach: Combining the hash value of the public key with the WebID URI creates a universal mean to detect tampering in WebID profiles. This, however, implicates that it is not possible 1) to transform common WebID URIs to tamper-evident ones and 2) to change tamper-evident WebID URIs without invalidating all incoming connections from social contacts expressing their friendship and so on. Changing a WebID URI results in creating a different and therefore new identity. This can be considered as a common shortcoming in WebID. There is a need for an approach to indicate that a new (tamper-evident) WebID URI belongs an already existing one. Creating a tamper-evident WebID profile necessitates signing the profile data worth protecting. As we wanted to avoid disclosing the private key to a third-party, we rely on a client-side signing tool at the moment. This dependency entails new requirements for system, platform and device support. We seek to resolve the dependency by a solution that is user-friendly and interoperable.

5 Conclusion and Future Work

By providing a means for users and machines to detect data integrity breaches in WebID profiles, we contributed to increase the trustworthiness of open, decentralized, and universal identification mechanisms. Taking compatibility and accessibility of user profile data into account, our approach for tamper-evident user profiles is well-applicable for new WebID identities. Focusing on empowering individuals instead of authorities, we expect that such mechanism will gain more momentum and coverage in the future. By adding security features to WebID artifacts, we enabled profile owners and requestors to detect malicious manipulation and identity theft. We integrated IronClad into the WebID authentication routine to verify the profile data integrity as a requirement for any successful identity proof. Not only does IronClad operate independently of data type and order, but also independently of the profile hosting system. We enabled verification without requiring prior knowledge except for an already known WebID identifier.

Having made an important step towards more trustworthiness in the context of WebID, we will constitute future work on that basis. For validating not solely the technical feasibility of our approach, it is necessary to conduct a more comprehensive evaluation involving how well users accept tamper-evident WebID URIs and profiles. Even though we are aware that tamper-evident WebID URIs make the management for users more difficult, e.g., compared to managing email addresses, we claim that most issues can be successfully addressed through utilizing techniques such as URI drag and drop, QR codes or WebID URIs embedded into other objects like WebID certificates. Identifying the origin of tampering attacks, facilitating key replacement and enabling signature creation with several keys are also interesting topics for future research.

References

1. Bamberg, W., et al.: Persona - Protocol Overview (2013), https://developer.mozilla.org/en-US/docs/Mozilla/Persona/Protocol_Overview (accessed February 23, 2014)
2. Caronni, G.: Walking the Web of Trust. In: Proeedings of the IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE 2000), pp. 153–158. IEEE (2000)
3. Carroll, J.J.: Signing RDF Graphs. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 369–384. Springer, Heidelberg (2003)
4. Dhamija, R., Dusseault, L.: The Seven Flaws of Identity Management: Usability and security Challenges. IEEE Security & Privacy 6(2), 24–29 (2008)
5. Feldman, A.J., Blankstein, A., Freedman, M.J., Felten, E.W.: Privacy and Integrity are Possible in the Untrusted Cloud. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering 35(4), 73–82 (2012)
6. Feldman, A.J., Blankstein, A., Freedman, M.J., Felten, E.W.: Social Networking with Frientegrity: Privacy and Integrity with an Untrusted Provider. In: Proceedings of the 21st USENIX Conference on Security Symposium, Security, vol. 12, p. 31 (2012)
7. Fitzpatrick, B., Recordon, D., Hardt, D., Hoyt, J.: OpenID Authentication 2.0 - Final (2007), http://openid.net/specs/openid-authentication-2_0.html (accessed February 23, 2014)
8. Hackett, M., Hawkey, K.: Security, Privacy and Usability Requirements for Federated Identity. In: Workshop on Web 2.0 Security & Privacy (2012)
9. Sauermann, L., Cyganiak, R., Völkel, M.: Cool URIs for the Semantic Web. Tech. rep., Saarländische Universitäts- und Landesbibliothek (2007)
10. Sayers, C., Eshghi, K.: The case for generating URIs by hashing RDF content (2002)
11. Sporny, M., Inkster, T., Story, H., Harbulot, B., Bachmann-Gmür, R.: WebID 1.0: Web Identification and Discovery (2011), http://www.w3.org/2005/Incubator/webid/spec/ (accessed February 23, 2014)
12. Tummarello, G., Morbidoni, C., Puliti, P., Piazza, F.: Signing Individual Fragments of an RDF Graph. In: Special Interest Tracks and Posters of the 14th International Conference on WWW, pp. 1020–1021. ACM (2005)
13. Wild, S., Chudnovskyy, O., Heil, S., Gaedke, M.: Customized Views on Profiles in WebID-Based Distributed Social Networks. In: Daniel, F., Dolog, P., Li, Q. (eds.) ICWE 2013. LNCS, vol. 7977, pp. 498–501. Springer, Heidelberg (2013)
14. Wild, S., Chudnovskyy, O., Heil, S., Gaedke, M.: Protecting User Profile Data in WebID-Based Social Networks Through Fine-Grained Filtering. In: Sheng, Q.Z., Kjeldskov, J. (eds.) ICWE Workshops 2013. LNCS, vol. 8295, pp. 269–280. Springer, Heidelberg (2013)
15. Yeung, C.M.A., Liccardi, I., Lu, K., Seneviratne, O., Berners-lee, T.: Decentralization: The Future of Online Social Networking. In: W3C Workshop on the Future of Social Networking Position Papers, vol. 2, pp. 2–7 (2009)