Secure Cross-Cloud Single Sign-On (SSO) using eIDs

Bernd Zwattendorfer, Arne Tauber
E-Government Innovation Center (EGIZ)
Graz University of Technology
Graz, Austria
bernd.zwattendorfer@egiz.gv.at, arne.tauber@egiz.gv.at

Abstract— **Most cloud computing service providers secure their offered cloud services by username/password schemes, which have been proven to be weak. While such schemes may be sufficient for simple personalized services, e-Government or e-Health applications in the cloud require more reliable and stronger mechanisms. One of such mechanisms are electronic IDs (eID), which allow for unique qualified identification and strong authentication. EIDs have been rolled-out in many EU Member States since years. In this paper we present how various national eIDs can be used for secure cloud authentication. We therefore extended the STORK eID interoperability framework, which will be the relevant identification and authentication framework across Europe in future. Furthermore, we increased usability by additionally applying single sign-on (SSO). Single sign-on defines the ability to authenticate just once in a distributed environment and gain access to several protected services. In fact, by our extended STORK architecture citizens of 18 EU Member States – those Member States that support STORK – are able to use seamless authentication at different cloud service providers by using their own national eID.**

***Keywords-** Cloud computing, eID, electronic identification, authentication, STORK, single-sign-on, SSO*

I. INTRODUCTION

Cloud computing is one of the fastest emerging IT topics today. Its high scalability provision and its enormous cost savings potentials offers a lot of advantages and benefits to cloud consumers. This makes cloud computing interesting and beneficial to be applied in various sectors. Moreover, this means that also more safety-critical sectors such as government or health start to jump on the cloud computing bandwagon.

Cloud computing services are usually offered on different levels. The National Institute for Standards and Technology (NIST) classifies cloud computing into three different service levels: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [1]. Applying IaaS, fundamental IT infrastructure such as computers, networks, or data storage are virtually provided to customers. In the PaaS model, programming and runtime environments with flexible and dynamic adaptable computing resources are provided. In the third service model (SaaS), software applications are provided as a service by a cloud service provider.

Cloud computing, and in particular PaaS and SaaS, are also promising offerings for e-Government or e-Health applications or portals. The transfer of such applications to the cloud has a couple of advantages, e.g. less maintenance efforts or lower costs. However, since e-Government or e-Health applications target sensitive areas, special security requirements must be fulfilled. Two such security requirements are identification and authentication.

Currently, most cloud service providers rely on username/password identification and authentication mechanisms for their PaaS and SaaS applications. While these authentication mechanisms may be sufficient for simple personalized cloud services, stronger mechanisms are required for cloud applications of sensitive areas such as of e-Government or e-Health. E-Government or e-Health services usually have to fulfill higher security and privacy requirements, as national law or data protection regulations must be complied with. Identification and authentication mechanism fulfilling those higher security requirements are for instance electronic IDs (eIDs). A lot of countries have already rolled-out national eID solutions to their citizens. EID solutions usually are user-centric and support unique identification and strong authentication. A thorough overview of various national eID solutions within Europe can be found in the Modinis-IDM study [2] or the IDABC eID country reports [3]. While eIDs have already been integrated into traditional web applications since years, they lack applicability in cloud computing environments.

In our work, we bypass this gap by enhancing and connecting the STORK framework [4] to cloud applications. STORK (Secure Identity Across Borders Linked) constitutes an interoperability framework supporting eID federation of various national eID solutions across Europe. This eID interoperability framework enables us to use various national eIDs for secure identification and authentication at cloud applications. STORK is currently heavily pushed by the Europen Commission and hence will be the relevant identification and authentication framework in Europe in future. Moreover, the use of STORK and thereby eIDs for cloud authentication can pave the way for an increasing cloud adoption in sensitive areas such as e-Government or e-Health.

The use of our amended STORK framework allows for unique identification and stronger authentication at cloud applications. Currently, various national eID approaches of 18 EU Member States are supported by STORK. All those different eIDs are also supported by our amended STORK framework for secure cloud authentication. While identification and authentication are necessary processes for accessing protected data or services, a high number of authentication processes can easily frustrate users. Frequent authentication processes may for instance appear at web portals or one-stop shops, where different vendors and service providers are bundled. For instance, many public authorities offer governmental one-stop shops for their citizens. The concept of single sign-on (SSO) bypasses this issue, allowing for authentication at several services without re-authentication. In addition to the support of various eIDs for cloud authentication, we added SSO functionality to our enhanced STORK architecture. This allows citizens to authenticate only once by using their eID, but gaining access to different cloud services at the same time.

The remainder of the paper is structured as follows. Section II gives some background information on single sign-on, related work, and the STORK framework. In Section III we detail the problem statement to be solved by our work. Our cross-cloud SSO architecture using eIDs, which is based on STORK, is presented in Section IV. Finally, we give lessons learned during the implementation in Section V and draw conclusions.

II. Background Information

This section gives some background information for our proposed work. Here, the basic concept of single sign-on (SSO), existing identification and authentication approaches and protocols relating to our work, and the STORK framework are briefly discussed.

A. Single Sign-On (SSO)

A lot of service providers rely on user authentication before their services can be accessed. This requires a proper user management at the service provider site. If service provider applications are bundled, e.g. through a web portal or a one-stop shop, this can result in multiple individual authentications at each service provider, because all service providers maintain their own user management. However, frequent authentication processes may annoy users and decrease usability.

To bypass this issue, the concept of single sign-on (SSO) has been developed. Single sign-on defines the ability to authenticate only once in a distributed network and to access several protected services and resources without re-authentication [5]. In case of username/password authentications, users just have to enter their credential once at a service provider and they get automatically and seamlessly authenticated at other service providers.

Security can be increased because authentication takes place at one single authentication authority only, which should be particularly protected. In addition, users need to remember just one strong password instead of multiple passwords for each individual service provider. Furthermore, time and costs can be saved as users just need to run through one authentication process only. Finally, the main advantage of SSO is increased user comfort because a number of burdensome authentication processes can be omitted while still getting access to several different protected services at the same time.

However, also disadvantages can be found for SSO. If the SSO authentication system breaks down, the complete network is affected and user authentication becomes impossible. In addition, if an attacker figures out the identity and authentication data of the SSO system, the attacker will be able to gain access to all services secured by the SSO system.

B. Related Work

Single sign-on does not define a new topic. Several SSO systems and protocols have already emerged over the past years. In this section we briefly describe the most important systems and protocols which gained importance either because of their broad use or as they established relevant standards. In particular, we focus on web-based SSO solutions supporting cross-domain single sign-on. Those solutions are also applicable for cloud computing.

1) Security Assertion Markup Language (SAML)

The Security Assertion Markup Language (SAML) [6] currently defines the most important standard for SSO. SAML, which is XML-based, has been developed by OASIS and is especially designed for the secure exchange of identification, authentication, and authorization data. For instance, the cloud service providers Google and Salesforce.com support SAML for external user authentication.

2) WS-Federation

WS-Federation [7], as part of the WS-Security framework, constitutes an XML-based specification especially designed for enabling identity federation across different security realms. Microsoft’s Windows Azure [8] cloud platform relies on WS-Federation.

3) OpenID

OpenID [9] is a decentralized SSO authentication system for web-based services. Users typically authenticate by username/password authentication mechanisms and receive a URL-based OpenID identifier. Authentication is carried out at so-called OpenID providers. Google for example is an OpenID provider.

4) OAuth

A standardized and open protocol focusing on application authorization depicts OAuth [10]. Basically, OAuth provides an API which enables applications the possibility to access specific user data of another application. It is popular in social networks such as Facebook, LinkedIn, or Twitter, but has also been adopted by some cloud SaaS providers such as Google or Salesforce.com.

C. STORK

The STORK framework, which has been designed and developed within the EU large scale pilot project STORK, constitutes a secure and reliable eID interoperability framework supporting eID federation of various national eID solutions. The STORK project involved 18 European Union Member States. It had started in 2008 and was finished by the end of 2011. By the help of this framework, citizens are able to securely authenticate at online services located in foreign European countries by actually using their own national eID, which has been issued by the citizens' home country. For example, via the STORK framework Austrian citizens are able to authenticate at Spanish governmental online services by using their Austrian eID. STORK entities communicate via a well-defined STORK protocol, which is SAML-based. Details on the protocol can be found in the STORK interface specification [11].

The individual national eID solutions and systems differ on technical, organizational or legal level. However, one main objective was to make the individual solutions interoperable and not to introduce a new EU-wide rolled-out eID concept. In fact, interoperability was achieved by building the STORK interoperability framework on top of the various national eID solutions. The results of STORK have been thoroughly tested in six pilot applications. Pilot applications are real and productive environments, such as e-Government portals. For a more detailed description of the STORK architecture, its implementation, and the pilots we refer to [12]. STORK is currently heavily pushed by the European Commission and hence will be the relevant eID framework across Europe in future. The STORK results are further taken up by its successor project STORK 2.0 [13]. Besides more intensive piloting, STORK 2.0 additionally focuses on electronic mandates and representations, such as cross-border identification and authentication of legal persons.

III. PROBLEM STATEMENT

Software as a Service (SaaS) providers offer customers ready-made applications developed and provided by the cloud service provider. Prominent examples of such SaaS cloud service providers are Google Apps [14] or Salesforce.com. However, a lot of other cloud service providers providing SaaS have already emerged and do exist.

Most of the SaaS applications are usually protected by some authentication mechanism. Currently, the dominant authentication approach used by cloud service providers depicts username/password schemes. Therefore, each cloud service provider also hosts its own and separate user management. This has several disadvantages. First, username/password authentication mechanisms early turned out to be weak [15], which makes such SaaS applications inapplicable for sensitive sectors like e-Government or e-Health. Second, users require and have to manage individual username/password pairs for each cloud service provider. I.e., for accessing services of different cloud providers the user first has to register at each provider and second has to authenticate separately. As an example, if a user wants to access CRM cloud services of Salesforce.com and the e-mail services (Gmail) of Google at the same time, she has to run through the complete authentication process for each cloud service provider separately. Fig. 1 illustrates this current sample situation for cloud authentication.

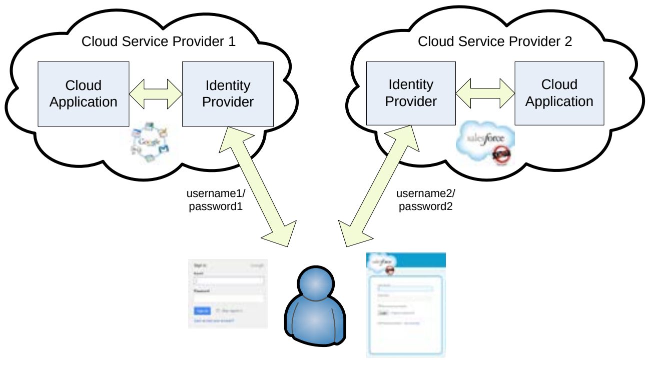

Figure 1. Current Situation for Cloud Authentication

In this situation, it is assumed that a user wants to access two SaaS cloud applications of two different cloud service providers at the same time. It is further assumed that the user is already registered at both providers. Doing this, the user hast to authenticate at the individual identity provider of each cloud service provider one after another. First, the user has to authenticate at the identity provider of cloud service provider 1 using appropriate credentials. Second, the user must provide other credentials to the identity provider of the second cloud application hosting service provider 2. Needless to say, the sequence of authentications could also be vice versa. For both cloud service providers, the identity provider is hosted in the provider's domain and manages user data of the individual domain only.

While two subsequent authentication processes do not extremely decrease comfort, a higher number of authentication sequences could become burdensome for users. To overcome this issue, we introduce a secure and privacy-preserving SSO architecture for cross-cloud authentication. By applying this architecture, users need to authenticate only once but still get access to applications of multiple cloud service providers. According to Fig. 1, users already authenticated at cloud service provider 1 are seamlessly authenticated at cloud service provider 2 without re-authentication. Furthermore, we increase security by using eIDs, which allow for unique identification and which are usually based on strong two-factor authentication instead of simple and unsecure username/password mechanisms. EIDs are user-centric, which puts users into maximum control of their personal data. In addition, users only need to mind just one secure token – and one secure PIN code - instead of managing an overwhelming mass of different passwords. Moreover, the use of eIDs enables fulfillment of certain legal requirements, which facilitates penetration of the cloud market also for sensitive areas such as e-Government or e-Health.

IV. CROSS-CLOUD SSO ARCHITECTURE

Most SaaS offers are secured by username/password authentication solutions. If users are registered at different cloud service providers, this leads to a situation where users have to maintain separate username/passwords for each individual cloud service provider. On the one side, an overwhelming amount of different username/passwords decreases the level of security, because people tend to re-use them or write them down close to their computers. On the other side, separate authentication processes decrease comfort and easily frustrate users.

One solution to overcome such security issues is the application of SSO. By using SSO, users normally just need to remember one strong password and the risk of re-use or of writing it down becomes lower. We have already briefly discussed advantages and disadvantages of SSO in section II.A. Another option for increasing authentication security is the use eIDs. EIDs allow for unique user identification and strong user authentication. For instance, the STORK framework supports different eIDs rolled-out in 18 EU Member States. STORK has also been briefly described in section II.C.

In our proposed architecture we combine both, secure authentication using eIDs and single sign-on to take advantage of the benefits of both solutions for a cross-cloud SSO authentication scenario. We therefore adopted and extended the STORK framework to support authentication at different cloud service providers on the one side, and to support single sign-on between these providers on the other side.

A. Extended STORK Architecture for Cross-Cloud SSO

The STORK framework allows for secure and reliable authentication of European citizens at service providers. It supports a variety of national eID solutions. Furthermore, the STORK protocol is SAML-based. SAML is also used and implemented by several cloud service providers (see Section II.B) for enabling and supporting external identification and authentication to SaaS applications. The great support of multiple high assurance credentials based on eIDs and the use of SAML were the reasons for taking the STORK architecture as a basis for our cross-cloud SSO solution. The support of eIDs guarantees stronger authentication and a higher level of assurance. The use of SAML enables the possibility for cloud authentication and the support of single sign-on.

The STORK framework supports several interfaces for connecting to and accessing diverse national eID solutions. One interface is based on the STORK protocol itself, other interfaces support national authentication protocols for service providers. Those national authentication protocols need not necessarily speak SAML. Currently, simple national specific web-based and web service-based authentication protocols are supported. The STORK framework has been designed in a modular way, hence new authentication protocols for service providers can be easily added. This gives us the possibility to interconnect and link the STORK framework also with cloud service providers.

Many cloud service providers use their own identity and user management for protecting their SaaS applications. However, some of them provide interfaces supporting external identity providers for user identification and authentication. These interfaces are typically based on well-established protocols and standards, which have been currently widely used in the identity management landscape. Examples of such protocols are SAML or OpenID, as briefly described in the related work section II.B. Cloud service providers relying on SAML for their external identification and authentication interfaces are e.g. Google or Salesforce.com. For our cross-cloud SSO architecture, we make use of those external authentication interfaces to connect to the STORK infrastructure. Fig. 2 illustrates our extended STORK architecture enabling cross-cloud SSO.

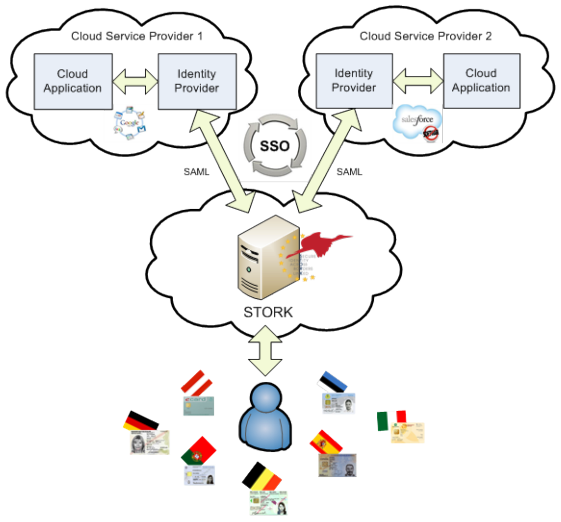

Figure 2. Extended STORK Architecture for Cross-Cloud SSO

The extended STORK architecture shown in Fig. 2 supports strong eID authentication at different SaaS cloud service providers, providing single sign-on between those providers at the same time. This means, by using her national eID a European citizen just needs to authenticate once via STORK at one cloud service provider. After that, the citizen is automatically and seamlessly authenticated at other cloud service providers protected by STORK without re-authentication. STORK currently supports 18 country-specific eID approaches, hence citizens of 18 EU Member States are capable of using this solution. In this figure, only two sample SaaS cloud service providers, namely Google and Salesforce.com, are shown. This is because those two providers were also chosen in our implementation, demonstrating SSO between different SaaS cloud service providers. We will dig into implementation details in the next sub-section.

The proposed architecture can be extended to support SSO between multiple cloud service providers. Although in our demonstrator SAML is used as external SSO interface for both cloud service providers, other SSO protocols (e.g. OpenID) offered by cloud service providers may be supported, as the STORK architecture allows for easy extension due to its modular design.

B. Implementation Details

This section explains the demonstrator we have implemented to proof SSO authentication between cloud service providers using eIDs. For our demonstrator, we selected two public SaaS cloud service providers, Google and Salesforce.com. The reasons why we chose those two providers were the support of an external interface for identification and authentication on the one side, and the support of SAML for these interfaces on the other side. We focused on SAML because SAML has been especially designed for cross-domain

SSO and because the STORK infrastructure already provides some basic SAML functionality.

For our proof of concept, we extended the STORK architecture by two additional service provider interfaces: one interface for secure authentication at Google Apps, and the other one for secure authentication at Salesforce.com. Both interfaces are SAML-based. Although STORK already provides SAML functionality, the STORK protocol could not be used out of the box for SSO authentication at these cloud service providers. The reason is that the SAML interfaces of STORK, Google, and Salesforce.com all behave differently. Therefore, only some basic SAML functionality of the STORK framework could be re-used for the implementation of the respective cloud service provider interfaces. In fact, we have implemented two STORK extensions for cloud service provider interfaces, both supporting SAML, but one the SAML "standard" of Google and one the SAML "standard" of Salesforce.com. However, both interfaces rely on the SAML 2.0 Web SSO Profile and the HTTP Post Binding for SAML message transfer.

Nevertheless, the previously described status of our implementation still reflects just single, but strong authentication at the individual cloud service providers. Hence, if users want to simultaneously access SaaS applications of both cloud service providers, they still have to login twice by using their respective national eID.

To achieve single sign-on between both providers, we had to further amend the STORK architecture to manage authentication sessions for a certain time period. Within this period, seamless SSO authentications between both providers are possible without re-authentication. In addition, we added some identity broker functionality because the national electronic identifier provided by STORK cannot be directly used for cloud service provider authentication in some situations. This especially has legal reasons as some countries do not allow direct processing of the national identifier due to data protection restrictions. Moreover, instead of using the identifier directly, some context-specific derivation is required. In our solution we therefore securely derive the identifier provided by STORK separately for Google and Salesforce.com using one-way cryptographic hash functions. For derivation we use a combination of the STORK identifier and a provider-specific identifier of the respective cloud service provider as a basis. Hence, we create two provider-specific identifiers that can be further used for identification at the respective cloud service provider. The use of hash functions for derivation has two advantages. First, hash functions still guarantee uniqueness of the newly created identifier. Second, hash function derivations do not allow recalculation to the actual given STORK identifier. This especially preserves citizens' privacy as never the very same identifier is used for identification at different cloud service providers.

The STORK framework implementation relies on traditional web technologies and application servers, thus it has not been especially designed for cloud computing adoption. Hence, in a first phase of our implementation of the cross-cloud SSO demonstrator we simply installed the STORK architecture in some local data center. In case of broad use, this allows for more control and privacy protection but can generate capacity bottlenecks in case of huge numbers of authentications in future. Therefore, to fully benefit from the cloud computing advantages, such as high scalability, we moved major parts of the STORK infrastructure also into the cloud. We succeeded on deployment of the amended STORK framework on two public cloud platforms, namely Jelastic [16] and the Google AppEngine [17]. While this was technically feasible, concerns may arise with respect to privacy or national law. In this case, personal data such as the unique user identifier may be fully visible to the cloud service provider, which hosts the STORK framework implementation.

V. Lessons Learned

One of the main lessons learned is that one SAML implementation is not like another. Even if the same SAML version, the same SAML profile, or the same SAML binding is used, the behavior is different between clients and providers respectively. This implies for our implementation, that although both for demonstration selected public cloud service providers (Google and Salesforce.com) rely on SAML 2.0, a different behavior of the provided SAML interfaces could be determined during the implementation. Hence, instead of one single SAML interface provided by STORK, the STORK framework had to be extended by two additional SAML interfaces. One interface supporting the SAML "standard" of Google, and the other one supporting the SAML "standard" of Salesforce.com. In fact, the SAML messages sent to the providers had to perfectly match their desired standard. SAML messages containing additional elements or attributes - even if they were valid according to the SAML 2.0 specification - were simply declined. This actually should not define a severe issue, but the documentation for implementing the SAML-based cloud service provider interfaces was rather weak. Google, for example, provides some out-dated reference code for demonstrating access to the SAML interface [18]. In contrast to that, Salesforce.com is more developer friendly. They actually provide some online validation service for testing the SAML messages. Verification of SAML messages, of course except security related functions, is not that strict compared to Google.

Concerning the identifier to be used for external identification at the cloud service providers, there also no common format can be used for both providers. The unique identifier provided by STORK is provider-specific derived due to privacy reasons. Thereby, Google only accepts identifiers following the e-mail format. This identifier must also match a username which has been previously registered in the internal user database of Google Apps. As opposed to this, in Salesforce.com applications real identity federation is supported. This means that the identifier used for external authentication just needs to be linked to an existing username of the internal Salesforce.com user management. Moreover, this does not define a strict requirement as Salesforce.com also supports on-the-fly registrations, which constitute seamless registrations during the very first successful authentication process.

Finally, we also faced interoperability issues between Platform as a Service cloud providers when deploying the

STORK infrastructure into the cloud. While deployment on Jelastic did not require any modifications on the source code – Jelastic supports a complete Java virtual machine (JVM) – the deployment on the Google AppEngine necessitated several code modifications, as only a subset of the Java virtual machine is supported by this cloud platform. For instance, the Google AppEngine does not support write operations to the file system, threads, or raw network sockets. Code parts requiring unsupported functionality had to be appropriately adapted.

VI. CONCLUSIONS

Username/password schemes are still the dominant authentication approach used for protecting SaaS applications. While username/password schemes may be sufficient for simple personalized services, they reach their limits in data sensitive areas such as e-Government or e-Health. E-Government or e-Health services require higher security requirements as usually sensitive data are processed. If e-Government or e-Health applications should be moved to the cloud, those high security requirements still must be fulfilled. One possibility to meet those requirements is the use of stronger authentication mechanisms for protecting SaaS applications, e.g. by the use of eIDs.

In our paper we demonstrated the use of various national eID solutions for secure cloud authentication. We therefore relied on the STORK eID interoperability framework, which will be the dominant identification and authentication framework across Europe in future. Currently, 18 EU Member States force this framework for broad adoption. In fact, this means that citizens of all 18 EU Member states, whose national eID solution is supported by STORK, are able to use our extended architecture for secure cloud authentication. Furthermore, we increased usability and citizen comfort by additionally enabling single sign-on. By this, citizens are seamlessly authenticated to several cloud services by using their eID only once for authentication. We demonstrated single sign-on authentication between two public cloud service providers, Google Apps and Salesforce.com. The use of eIDs and SSO for cloud authentication paves the way for increasing future cloud adoption in sensitive areas such as e-Government or e-Health, as legal requirements can be easier fulfilled compared to username/password authentications. Furthermore, this offers cloud service providers the possibility to penetrate new market areas for business generation.

REFERENCES

[1] P. Mell and T. Grance, “The NIST definition of cloud computing.” NIST, p. 7, 2010.

[2] European Commission - MODINIS, "The Status of Identity Management in European eGovernment initiatives", 2006.

[3] European Commission - IDABC, "eID Interoperability for PEGS: Update of Country Profiles", 2009.

[4] STORK (Secure Identity Across Borders Linked), https://www.eid-stork.eu

[5] J. D. Clercq, “Single Sing-On Architectures” in Infrastructure Security, G. Davida, Eds. InfraSec 2002 Proceedings, vol. 2437, October 2002, pp. 40–58.

[6] Lockhart, H. and Campbell, B. 2008. Security Assertion Markup Language (SAML) V2.0 Technical Overview. OASIS Committee Draft 02.

[7] Kaler, C. and McIntosh, M. 2009. Web Services Federation Language (WS-Federation) Version 1.2. OASIS Standard.

[8] Microsoft, Windows Azure, http://www.windowsazure.com

[9] OpenID, http://openid.net

[10] OAuth, http://oauth.net

[11] J. Alcalde-Morano, J.L. Hernández-Ardieta, A.Johnston, D. Martinez, B. Zwattendorfer, M. Stern, "D5.8.3b Interface Specification", STORK Deliverable, 2011.

[12] H. Leitold and B. Zwattendorfer, "STORK: Architecture, Implementation and Pilots". in ISSE 2010 Securing Electronic Business Processes: Highlights of the Information Security Solutions Europe 2010 Conference, pp. 131-142. 2010.

[13] STORK 2.0, http://www.eid-stork2.eu

[14] Google Apps, http://www.google.com/apps

[15] G. Kessler, "Passwords - Strengths and Weaknesses", Internet and Networking Security, Auerbach, 1997.

[16] Jelastic, http://jelastic.com

[17] Google AppEngine, https://appengine.google.com

[18] Google Developers, “Google Apps Platform - Web-based Reference Implementation of SAML-based SSO for Google Apps”, http://code.google.com/googleapps/domain/sso/saml_reference_implementation_web.html