Single Sign-On: Reviewing the Field

Michael Grundmann, Erhard Pointl

Johannes Kepler University Linz

Abstract. The Idea of having only one password for every service has led to the concept of single sign-on which is a much discussed topic over the past years. This paper looks into the different architectural approaches and uses them as a basis for categorizing the widely used technologies today.

1 Introduction

The growing need for security and the even faster growing variety of applications, services and systems requires an appropriate authentication infrastructure. The diversity has led to an equally diverse range of authentication systems. Users need to manage different usernames, passwords or other forms of authentication to each of this systems. The wish for an single sign-on is therefore quite understandable. But the question of how such a single sing-on system should work is a much discussed one. Apart from the benefits of having only one password there could also be negative implications to the users security and privacy. Furthermore such a system would introduce a single point of failure, where one stolen password can give access to a lot of services. Because of all this, it is time to take a look at the current state of single sign-on systems.

2 Architectures

2.1 Pseudo SSO Systems[1][2]

One obvious approach for the relieve of the user is to support him or her with the credential management. The user can store all passwords and authentication information into an encrypted password file or database which is secured with a master password. This password may be more complex than a normal password because the user only has to memorize this single password. It is also possible to secure the credential database with an fingerprint[3] or hardware based protection([4]). But as multiple different authentication processes are still taking place, even if this happens without necessary user intervention, such systems can not be called true single sign-on systems.

Implications[5][2] The credential database may be corrupted or lost, effectively locking the user out if he can't remember the passwords (which is likely

if they are not used regularly). On the other hand the user has to type the master password very often, which may help him to memorize it. This also means that an attacker only needs to get the master password to access all services of the user. This solution also affects the users mobility because the credential database has to be available on every computer he uses. To solve this problem the credentials may also be stored online, but the user has to trust the provider of such a service.

One main objective of single sign-on systems is the integration of different security domains. The different approaches to this can be classified in the following categories:

2.2 Centralized SSO

The main characteristics of a centralized single sign-on systems are a centralized authentication site and a centralized user database. Such a site has a trust relationship with each of the Service Providers (SP) within a domain and is often called a Trusted Third Party (TTP). The user only has to authenticate himself to the TTP. Such systems differ in the way a user is authenticated and can be further categorized as follows:

Token Based SSO[6] After a user is authenticated at the TTP he receives a software token which is stored (cached) on his client machine. The Service providers can validate such a token with cryptographic methods based on secret keys (symmetric cryptography). Those keys establish a trust relationship between the TTP and the SPs.

PKI Based SSO[6] In PKI based systems the user first has to generate an asymmetric key pair. The public key of this pair has to be sent to the central authentication site which is called Certification Authority (CA) in this case. After the user has authenticated himself the CA signs his public key and sends the certificate generated this way back to the user. The Service Providers are now able to verify the users certificate by using the CA's public key.

2.3 Federated SSO[6]

Centralized single sign-on Systems are limited to a single Environment, Company or Domain. To establish trust between different Domains, Federations are needed. The goal is therefore a system in which a users credentials from his domain are accepted by a foreign domain without the user having to re-enter his credentials. The foreign Domains TTP (or CA) has to accept the local Domains credentials because of an existing contract or federation. No synchronization or copying of credentials is needed. For PKI based Systems this may be accomplished by CA hierarchies or cross certification.

Structure[7]

Circle of Trust The base of a Federated system is the Circle of Trust, where each of the the Service Providers (Companies) have to trust each other. This Circle not only has to be implemented technically (e.g.: CA hierarchies,..) but also via business contracts between the participating Companies. Such contracts should also include operating agreements on which each SP has to behave upon.

Identity Provider Inside a Domain it may be feasible to centralize the administration and management of user identity information using either one central Identity Provider (IP) or forming a security domain containing of several IPs each trusting each other. The later has a similar structure to the Circle of trust but only within an organization.

Identity Federation Apart from the circle of trust SPs also have to negotiate operating agreements with other SPs from other security domains. One possible and important agreement is about the users privacy: A users identity (managed by an IP) consists of a set of attributes[7], it should be possible to mark such attributes as sensitive private data or as confidential information. Attributes marked this way must not be transferred to other SPs. Normally such data is not needed for authorization anyway. Those attributes which are allowed to be transferred build a so called federated Identity of the user.

3 Comparison

First of all this section gives a few criteria of single sign-on systems and afterwards a categorization based on the criteria and the architectures. It ends up with a classification of a few typical single sign-on products based on the categorization.

3.1 Criteria

To be able to do a comparison, definitions of the following basic criteria are needed: usability, security, performance, scalability, compatibility and maintenance.

Usability The usability is defined as a measurement of how comfortable it is to use such a system. The level of usability depends on the categorization. For example a Federated single sign-on system increases the usability because the users credentials from his domain are accepted by a foreign domain. A Centralized single sign-on decreases the usability because the user has to authenticate at every domain. Using a pseudo single sign-on system is the worst case in respect of usability.

Security Because SSO Solutions try to solve the multiple authentication problem security is always an issue. Not only needs the identity information (passwords, private data,..) be kept secure but it also has to be defined who may access these information and how. If the information is kept at multiple locations the question of trust becomes more important. This is of course easier to accomplish in centralized environments where typically only one company or security domain is involved. In Federated systems security cannot be enforced but only regulated by agreements along with the trust relationships. In Pseudo SSO Systems Security depends heavily on the encryption of the authentication information but may also be influenced by the reachability.

Performance The performance is another very important criteria which can be reviewed by values like response time, time for doing a logon. This depth of evaluation is not possible at this level, so the behavior of many users is evaluated here. If the number of users increases, also the authentication authority has do consist of multiple authentication servers[6]. Of course this criteria depends very strongly on the real implementation and the architecture of the single sign-on system.

Scalability As shown in [6] it doesn't mean that if a single sign-on system is used for authentication that there is not only one authentication server behind the system. In terms of scalability it may have several authentication servers. The grade of scalability depends on the already given categorization. The use of an pseudo single sign-on system would decrease the scalability, because such a system can only consist of a single password file.

Compatibility This criteria means, that it should be easy to integrate logins and it should also be based on known standards. For example a password file as known from pseudo single sign-on system would decrease the level of compatibility. In most cases they are developed internally. For centralized and federated single sign-on systems a general classification is nearly impossible, because the rating would depend to much on the real implementation of the system and the used standards.

Maintenance The afford of spending time in maintenance of a single sign-on system is measured here. This also can be weighted as maintenance costs which is done in [1]. This costs are based on the architecture of the used single sign-on system. For example in general maintenance costs for pseudo single sign-on systems are high and for centralized and federated single sign-on systems they are typically low[1].

Deployment This criteria measures the afford of spending time in deployment of a single sign-on system. As shown in [1] this can also be weighted as deployment costs and this costs are lower for pseudo single sign-on systems and

higher for centralized and federated single sign-on systems. As an example for the higher deployment costs of such a system the increasing importance of security is marked out here.

3.2 Categorization

To give a kind of categorization the now defined criteria are assigned to the architectures. For this the Table 1, where the architectures are outlined in the horizontal and the criteria in the vertical is used. For categorization the following signs are introduced: "+" ,"-", "∘" and applied to the tuple of (criteria, architecture). The sign "+" means that this criteria is fulfilled quite good for a given architecture. A "-" means that it is not or nearly not fulfilled. The "∘" defines a symbol between the other two, which means that the criteria is partially fulfilled for the given architecture.

	Pseudo SSO	Centralized SSO	Federated SSO
Usability	-	∘	+
Security	∘	+	∘
Performance	-	∘	∘
Scalability	-	+	+
Compatibility	-	*	*
Maintenance	-	∘	+
Deployment	+	-	-

Table 1. Categorization of single sign-on systems

As Table 1 shows Pseudo SSO Systems are almost categorized with the "-" sign. But as already the name says this are only pseudo single sign-on systems and such systems can't be called true single sign-on systems. The only criteria where the pseudo single sign-on system was good in rating was the deployment. The reason therefore is that there the afford or the spent money for such a system was rated. Because such a system is rather simple to implement, its costs are quite low.

The next architecture to categorize are centralized single sign-on systems. The result of the categorization is in average better than the one from the pseudo single sign-on system, because they already implement parts of real single sign-on systems. The major part which they don't implement is the federation.

Federated SSO Systems had the best categorization in our case, because they support all the criteria the other also do and they additionally support the federation. Because of this very important point the federated single sign-on is rated better in usability and maintenance. Usability of course increases because now the user must not authenticate so often compared with different centralized single sign-on systems. Comparing all different systems of course the

federated version would be better to maintain than several centralized single sign-on systems.

For the criteria compatibility some architectures are marked by an "*". This means that no conclusion can be given by only knowing the architecture. It depends to much on the real implementation of the system. But as shown in Table 1 pseudo single sign-on systems can be categorized by an "-", because they mostly use proprietary password files or databases.

3.3 Products

Now some typical single sign-on products are categorized. Also their application areas are discussed.

Kerberos[8],[9] Kerberos is a very famous centralized token based single sign-on system. It was first developed in the 80's at the Massachusetts Institute of Technology (MIT). For using Kerberos, there must be a server providing a service, a client and the KDC (Key Distribution Center). This KDC is splitted into two parts (the Authentication Server AS and the Ticket Granting Server TGS). The Client has to authenticate at the KDC before using a service on the server. A so called TGT (Ticket Granting Ticket) is created and given to the user after authentication. If the client want to use another service, another authentication at the AS is not needed. Now he can directly contact the TGS with his TGT to get a ticket for another service. The server of the new service has to check whether the ticket of the client is valid or not. To summarize the client has only to authenticate once and afterwards he only communicates with the TGS as long as his ticket is valid.

To show up a known usage of Kerberos Microsoft Windows 2000[6] is given as an example here.

PKI Based SSO as proposed in [10] by Eian and Mjølsnes. Their paper suggests a system in which certificates with short validity periods are used for single sign-on. Clients have to generate a cryptographic public-private key pair and send the public key to the CA (called Authentication Server (AS)) upon login. Based on the AS security policy the user is either authenticated via a strong password or a hardware cryptographic token. If successful the AS signs the public key generating a certificate which is sent back to the client. The user may now access different servers (which trust the AS) by using the issued certificate. Because of the short validity period of the certificate it may have to be renewed several times during a session. This is done by creating a new key pair and signing the renewal request with the old pair. After receiving the new certificate the old key pair is destroyed. Of course this has to be done before the validity of the certificate ends.

The Authentication Server only provides a centralized authentication, the servers can therefore perform authorization and access control themselves. This is important because most services need to store additional user attributes or

access rights which would be difficult to centralize. This of course still fullfills the single sign-on idea because the user has to identify himself only once (at the beginning of each session).

Such an implementation is typically used in a company environment where different services need to be connected via central authentication providing single sign-on.

shibboleth[8], [11] shibboleth is an open source project based on SAML (Security Assertions Markup Language). It is categorized into the federated single sign-on systems. As federated single sign-on describes, shibboleth supports authentication over federated domains. If a user tries to access a service on a different domain the user can choose which information is sent from the local domain of the user to the federated domain. This is the case to save the privacy of the user. For example in an academic field the personality is not needed every time, it would be enough if the system knows that for example the user is participant on a course to authorize for getting course material. Of course the user has to be authenticated on the local system before he can visit a federated domain.

A typical implementation of shibboleth consists of an identity provider, a service provider and a WAYF (where are you from?)-Service. The identity provider holds all the users and their additional attributes. The service provider provides services to the user. If he would like to use such service, the service provider contacts the identity provider to get information about authorization. The identity provider creates so called assertions which describes the authorization and sends them back to the service provider. The WAYF-service is optional and it can automatically find the preferred identity provider and it also can act as proxy between the service provider and the identity provider while the authorization process.

Shibboleth is often used in academic fields. To give a local example also the university of Vienna uses shibboleth.

Microsoft Passport[12] Passport from Microsoft is a centralized single sign-on system and was first introduced in 1999. This system is centralized, because only one entity (Microsoft) can authenticate a user. For using the Passport both, the service provider and the user, have to register at the Passport Server. The user has to register with email address and password. After this registration a pseudonym (Microsoft call it PUID) is assigned to the user. The Service Provider has to provide Microsoft information about his service and his safety guidelines. Afterwards a unique id is assigned to the Service Provider and a key for encryption of the communication with the Passport is given to the Service Provider.

While the user tries to access the service provider, he is redirected to the login of the Passport Server. The login is done with the already registered email address and password. After a valid login the user is redirected back to the service provider and cookies are stored on the users maschine. In this cookies among other things the users PUID is stored. With this pseudonym, which doesn't say

any details about the user, now the service provider can identify the user behind it. Also while connecting to another service, this cookie can be used. Therefore the cookie will be sent to the Passport Server. So the user hasn't to sign on with email address and passwort again. The server validates the cookie and when this was successful he redirects to the service provider again. The cookies are deleted when signing out of the system. Afterwards the Passport server also notifies the service provider with the information of the sign out of the user.

Microsoft Passport of course is used by Microsoft themself. For example their email service Hotmail uses this system. Also other service provider are using this single sign-on system. But as already mentioned before there is only one entity to authenticate.

4 Conclusion

The Idea of single sign-on is not a new one but some time has passed since its first appearance therefore it was time look at the current state of SSO systems. This paper described the main architectures and their advantages and disadvantages. No single system has come out as the winner and the right system for an application area has to be chosen according to its needs and circumstances. To aid this decision this paper categorized single sign-on systems and picked out the main interesting criteria of every category. The result of this comparison can be found in Table 1. There were also given examples of in-use products for each architecture and their typical field of usage.

References

1. Pashalidis, A., Mitchell, C.J.: A taxonomy of single sign-on systems. LECTURE NOTES IN COMPUTER SCIENCE (2003)
2. Roßnagel, H., Zibuschka, J.: Single sign on mit signaturen. Datenschutz und Datensicherheit - DuD **30**(12) (2006)
3. Park, B., Hong, S., Oh, J., Lee, H., Won, Y.: One touch logon: Replacing multiple passwords with single fingerprint recognition. In: CIT '06: Proceedings of the Sixth IEEE International Conference on Computer and Information Technology, Washington, DC, USA, IEEE Computer Society (2006) 163
4. Jones, M.F., Zachai, A.: Encrypted data storage card including smartcard integrated circuit for storing an access password and encryption keys (April 1997)
5. Adams, A., Sasse, M., Lunt, P.: Making Passwords Secure and Usable. PEOPLE AND COMPUTERS (1997) 1–20
6. De Clercq, J.: Single Sign-On Architectures. LECTURE NOTES IN COMPUTER SCIENCE (2002) 40–58
7. Delessy, N., Fernandez, E.B., Larrondo-Petrie, M.M.: A pattern language for identity management. Computing in the Global Information Technology, International Multi-Conference on **0** (2007) 31
8. Zwattendorfer, B.: Single sign-on unter verwendung der bürgerkarte. Master's thesis, Graz University of Technology (May 2006)

9. Stojceski, D.: Konzeption einer kerberos-basierten single sign-on lösung für ein ausgewähltes szenario im hoch-schulbereich. Master's thesis, University of Applied Sciences of Bonn-Rhein-Sieg (March 2006)
10. Eian, M., Mjølsnes, S.: Large scale single sign-on scheme by digital certificates on-the-fly. In: Norwegian Network Research Seminar. (2005)
11. Morgan, R.L., Cantor, S., Carmody, S., Hoehn, W., Klingenstein, K.: Federated security : The shibboleth approach. EDUCAUSE Quarterly **27**(4) (2004)
12. Geihs, K., Kalcklösch, R., Grode, A.: Single sign-on in service-oriented computing. In: ICSOC. (2003) 384–394