A Generalized Pseudo-SSO Scheme

Sergey Panasenko

Abstract—**Single sign-on (SSO) systems allow to solve the following problem: how to store and to use a large amount of authentication information (e. g. username and password pairs) to gain access to different resources. SSO systems can be divided into two main types. One of them is pseudo-SSO systems. Their main feature is that they simply act by the following scheme: first, the pseudo-SSO component authenticates a user once; second, when the user requires any service of the covered system, the pseudo-SSO component uses certain user's data required to gain access to the desirable service. In this paper we propose a generalized scheme of data interchanging among users and SSO modules with specifics of pseudo-SSO systems. We describe relations between pseudo-SSO component's datasets required to perform SSO functions. Also we discuss that the single entry point to the secured system (such as a pseudo-SSO component) raises extra security problems.**

Index Terms— **SSO, pseudo-SSO systems, authentication.**

I. Introduction

Today users frequently find themselves in a situation when they are required to remember a large amount of authentication information (mainly, identifiers and passwords) to gain access to several resources of any system or network. Sometimes this forces a user to perform various insecure actions like:

- writing down passwords,
- using the same password to gain access to all destinations,
- using simple passwords [1].

Another common users' practice is separating all resources for different classes and using several passwords for that classes, e. g.:

- the first password is intended to access highly secured systems,
- another password – for systems with moderate security,
- and the third password – for systems with minimal security level [2].

Both of these approaches greatly decrease system's security. This allows an adversary to get unauthorized access to a number of resources after compromising just one of them.

The described problem can be solved using SSO systems. Commonly, a SSO scheme is relatively simple: firstly, a user is authenticated by a server (which is a component of SSO system) only once; then the SSO component is authenticated by other resources on user's behalf.

Manuscript received February 21, 2011; revised July 19, 2011.

Dr. S. Panasenko, head of software development, ANCUD Ltd., Moscow, Russia (e-mail: serg@panasenko.ru).

OpenID protocol (which is used particularly in Live Journal service) and Windows Live ID service (which gives access to a variety of Microsoft Corporation's resources) are worldwide examples of SSO systems.

All aspects of SSO systems are discussed in many papers. Some examples of them are listed below.

1) The paper [3] considers a technique which allows using mobile phones for authentication in SSO systems.
2) The paper [4] describes SSO systems based on trusted hardware and software modules (including the cryptographic module with RSA [5] private key inside).
3) A number of methods of using SSO to provide adequate security of user network sessions from untrusted network access devices (e. g. from an Internet cafe or an airport terminal) is described in [6].
4) Authors of [7] and [8] describe the implementation of SAML (Security Assertion Markup Language) [9] and its capabilities to provide secure SSO solutions for web-servers.
5) The paper [10] describes a SSO scheme where user authentication is based on payment cards conforming to the EMV standard [11] (EMV cards are supplied with an operating system and are capable of running different applications including digital signature modules).
6) The paper [12] discusses specifics of SSO when using in Grid systems [13].

A comprehensive review of SSO products and their applications is given in [1].

In this paper we propose a generalized scheme of data interchanging between users and SSO modules with specifics of pseudo-SSO systems. Also we discuss some security problems which should be solved while designing a system with pseudo-SSO components.

II. Pseudo-SSO

We focus on pseudo-SSO systems, which have been introduced in [14]. The authors of [14] presented taxonomy of SSO systems. They distinguish two main types of SSO systems:

- pseudo-SSO systems,
- true SSO systems.

Pseudo-SSO systems simply act by the following scheme:

- first, the pseudo-SSO component authenticates a user once,
- second, when the user requires any service of the secure system, the pseudo-SSO component uses some data of the user (e. g. certain login and password) required to gain access to the desirable service.

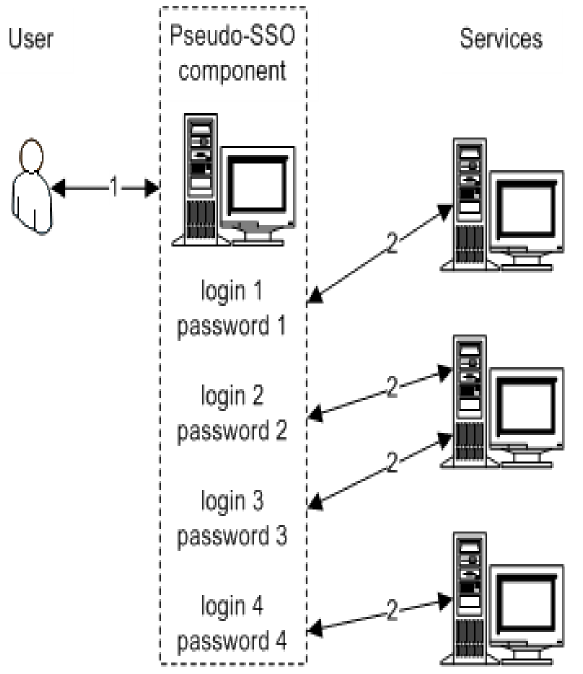

Fig. 1. A pseudo-SSO system

The scheme described above is the main feature of pseudo-SSO systems. It is illustrated by Fig. 1.

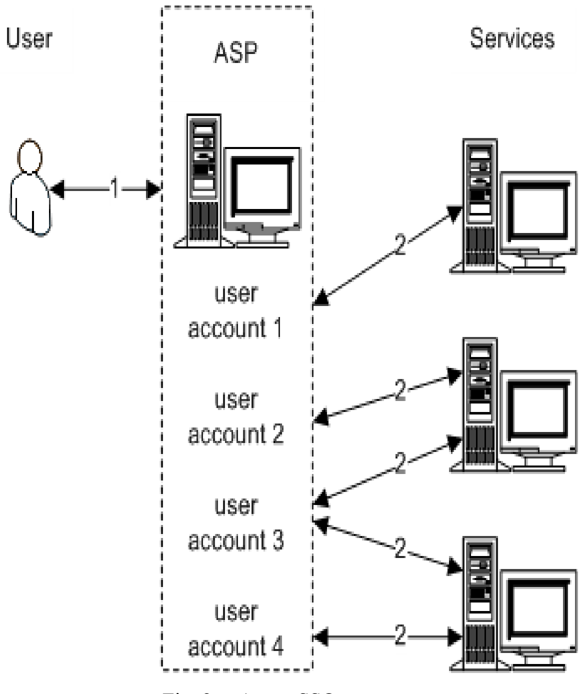

Fig. 2. A true SSO system

The main difference between true SSO and pseudo-SSO systems is that the true SSO component actually is the Authentication Service Provider (ASP), which acts as an active side in information exchange between a user and resources. (Fig. 2) [14]. Besides, true SSO components allow many-to-many relations between user accounts and services. Therefore a user potentially can select one of possible identifiers for a destination service or can use the same identifiers for several resources.

III. Generalization of Pseudo-SSO Schemes

As described above, a pseudo-SSO system allows to perform user authentication only once during a user's session with the system. All additional authentication actions (to gain access to the system's services) are performed by the pseudo-SSO component without any requests to the user. The user, in fact, delegates his privileges to the pseudo-SSO component. The latter, in its turn, plays a role of a proxy – the pseudo-SSO component stores the necessary data and supplies it when required; it can also transfer all user requests to the system's services and back.

To perform described functions, the pseudo-SSO component must contain the predetermined dataset I_i, which can be represented as a tuple of the following parameters for every i-th user of the system:

$I_i = \{U_i, A_i, \{S_i\}\}$, $i = 1 \ldots N$,

where:

- N – amount of users in the system,
- U_i – i-th user's identifier,
- A_i – a set of data required to perform i-th user authentication by the pseudo-SSO component,
- $\{S_i\}$ – an array of i-th user's datasets required to perform user's authentication on J destination services of the system.

$\{S_i\}$ also contains an array of i-th user's rights on the destination services. In its turn, each dataset of the $\{S_i\}$ array can be represented as the following tuple:

$S_{i,j} = \{U_{i,j}, A_{i,j}, R_{i,j}\}$, $j = 1 \ldots J$,

where:

- $S_{i,j}$ – i-th user's dataset related to j-th service,
- $U_{i,j}$ – i-th user's identifier on j-th service,
- $A_{i,j}$ – a set of data required to perform i-th user authentication on j-th service,
- $R_{i,j}$ – a collection of i-th user's rights on j-th destination service.

Relations between pseudo-SSO component's datasets are depicted on Fig. 3.

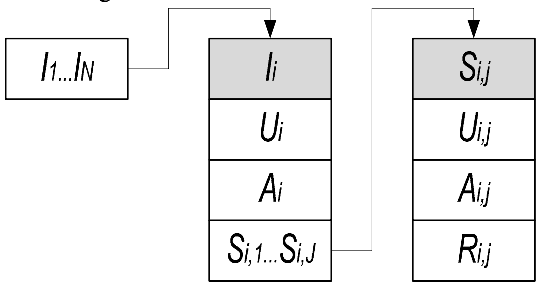

Fig. 3. Datasets of the pseudo-SSO component

Using the notation listed above, let us reconsider the scheme of user authentication in the pseudo-SSO system.

Stage 1. Primary user authentication:

1) The pseudo-SSO component identifies a user by searching user's identifier among $I_1 \ldots I_N$ datasets.
2) Then the pseudo-SSO component performs user authentication using A_i dataset.
3) The authentication is considered unsuccessful if an error

occurred during the previous steps.

The flowchart of the stage 1 is represented on Fig. 4.

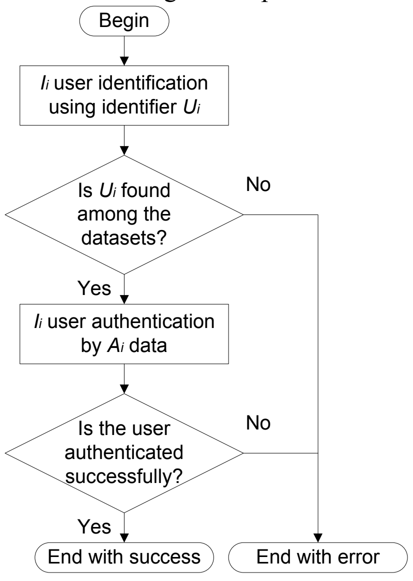
Begin
I_i user identification using identifier U_i
Is U_i found among the datasets?
No
Yes
I_i user authentication by A_i data
Is the user authenticated successfully?
No
Yes
End with success
End with error

Fig. 4. The flowchart of the stage 1

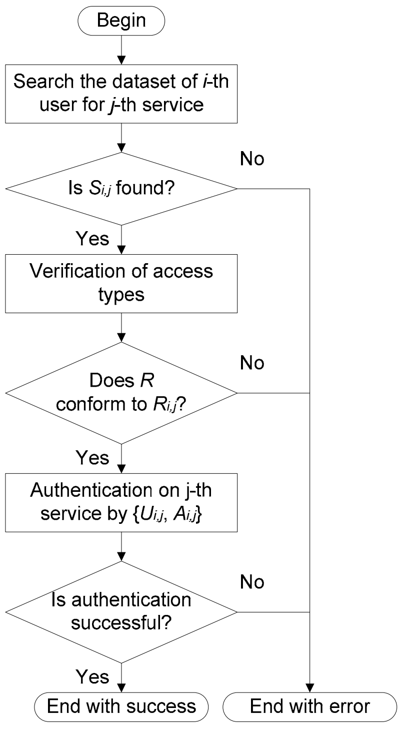
Begin
Search the dataset of i-th user for j-th service
Is $S_{i,j}$ found?
No
Yes
Verification of access types
Does R conform to $R_{i,j}$?
No
Yes
Authentication on j-th service by $\{U_{i,j}, A_{i,j}\}$
Is authentication successful?
No
Yes
End with success
End with error

Fig. 5. The flowchart of the stage 2

Stage2. *i*-th user authentication on *j*-th service is performed by the pseudo-SSO component on behalf of the user when required (Fig. 5):

1) The pseudo-SSO component searches the *i*-th user's dataset containing information related to *j*-th service ($S_{i,j}$) within the $\{S_i\}$ array.
2) If the $S_{i,j}$ dataset is not found, the authentication is unsuccessful.
3) The pseudo-SSO component verifies that the type of access requested by the user (R) conforms to the user's rights collection $R_{i,j}$. If $R \subseteq R_{i,j}$, access is granted, otherwise it is denied.
4) Finally, the pseudo-SSO component fetches the user's identifier and the information required to perform user's authentication on *j*-th service, i. e. $\{U_{i,j}, A_{i,j}\}$. This dataset is presented to *j*-th destination service. As a result the server grants or refuses required access to the user.

IV. Remarks on Security

"Single Sign-on is not a security issue" [1]. SSO helps administrators to manage user accounts in systems with many (assuming heterogeneous) resources with restricted access. SSO makes a single entry point for an administrator. But at the same time this single entry point is the most attractive goal for an adversary to break the full enterprise secured by SSO. Compromise of a pseudo-SSO component or an ASP can compromise the whole system. Therefore the pseudo-SSO component must meet at least the following security requirements:

- any denial of service or a network failure of the pseudo-SSO component can harm the whole secured system, so technologies of clustering and load-balancing should be applied to the pseudo-SSO component,
- the pseudo-SSO component should be highly secured from an unauthorized access,
- the data interchange between the parties of SSO should be secured (at least while performing authentication stages).

As an example of successful attacks on some SSO applications we can consider the attacks described in the paper [8]. Its author describes several attacks on the SAML SSO three-party authentication protocol:

- the man-in-the-middle attack,
- the attack by information leakage,
- and the message replay attack.

Use of SSL/TLS protocols [15, 16] "enhances the security of the SAML Single Sign-on protocol dramatically, but does not guarantee complete security" [8].

We can conclude that the single entry point to the secured system (such as the pseudo-SSO component) raises extra security problems.

V. Conclusion

In this paper we consider pseudo-SSO systems and propose a generalized scheme of data interchanging between users and SSO modules with specifics of pseudo-SSO systems. We describe relations between pseudo-SSO component's datasets required to perform SSO functions. And we also discuss security problems of a single entry point to the system such as the pseudo-SSO component; it should be highly secured to prevent attacks that can compromise the whole system secured by SSO.

Acknowledgment

We would like to thank Kirill Averchenko, Alexander Sharamok, Alexander Domoratsky, and Denis Tverdunov for reviewing and valuable comments on this paper.

REFERENCES

[1] T. R. Peltier. Single Sign-On: Myth or Reality. // Available: http://citeseer.ist.psu.edu, 2000.

[2] Time-out Management in Multi-domain Single Sign-On. // Available: http://citeseer.ist.psu.edu, 2004.

[3] A. Pashalidis and C. Mitchell, Using GSM/UMTS for Single Sign-On. Available: http://citeseerx.ist.psu.edu, 2003.

[4] A. Pashalidis and C. J. Mitchell. Single Sign-On using Trusted Platforms. In C. Boyd and W. Mao, editors, Information Security, 6th International Conference, ISC 2003, Bristol, UK, October 2003, Proceedings, vol. 2851 of Lecture Notes in Computer Science, pp. 54-68, Springer, 2003.

[5] PKCS #1: RSA Cryptography Standard. Available: http://www.rsa.com.

[6] A. Pashalidis and C. J. Mitchell. Impostor: A Single Sign-On System for Use from Untrusted Devices. Available: http://citeseerx.ist.psu.edu, 2004.

[7] K. D. Lewis and J. E. Lewis. Web Single Sign-On Authentication using SAML. International Journal of Computer Science Issues, Vol. 2, 2009, pp. 41-48.

[8] T. Gross. Security Analysis of the SAML Single Sign-on Browser/Artifact Profile. // Available: http://citeseer.ist.psu.edu.

[9] Security Assertion Markup Language (SAML). Available: http://xml.coverpages.org. February 23, 2010.

[10] A. Pashalidis and C. J. Mitchell. Using EMV cards for Single Sign-On. Available: http://www.isg.rhul.ac.uk, 2004.

[11] EMV Specifications. Available: http://www.emvco.com.

[12] R. O. Sinnott, O. Ajayi, A. J. Stell, J. Watt, J. Jiang, J. Koetsier. Single sign-on and authorization for dynamic virtual organizations. International Federation for Information Processing, 224, pp. 555-564. ISSN 1571-5736. 2006.

[13] I. Foster. What is the Grid? A Three Point Checklist. // Available: http://dlib.cs.odu.edu. July 20, 2002.

[14] A. Pashalidis and C. J. Mitchell. A Taxonomy of Single Sign-On Systems. In R. Safavi-Naini and J. Seberry, editors, Australasian Conference on Information Security and Privacy – ACISP'2003, vol. 2727 of Lecture Notes in Computer Science, pp. 249-264, Springer, 2003.

[15] A. O. Freier, P. Karlton, P. C. Kocher. The SSL Protocol. Version 3.0. Available: http://tools.ietf.org. November 18, 1996.

[16] T. Dierks and C. Allen. RFC 2246: The TLS protocol. Version 1.0. January 1999.

Dr. S. Panasenko received his Ph.D. from Moscow Institute of Electronic Engineering, Russia (2003). Since 1996 he works at ANCUD Ltd. as a software developer and (since 1999) as the head of software development department. He is an author of two books (in Russian) in a field of cryptography. Member of IACSIT (2011). His fields of interest include cryptology and security of computer systems and networks.