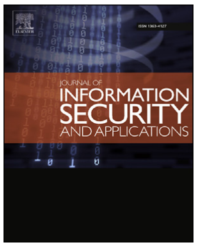
Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/jisa



Logout in single sign-on systems: Problems and solutions

Sanna Suoranta [a,*], Kamran Manzoor [a], Asko Tontti [b], Joonas Ruuskanen [a], Tuomas Aura [a]

[a] Aalto University, Espoo, Finland
[b] CSC — IT Center for Science, Espoo, Finland

Keywords:
Authentication
Single sign-on systems
Federated identity management

A B S T R A C T

Web single sign-on (SSO) systems enable users to authenticate themselves to multiple online services with one authentication credential and mechanism offered by an identity provider. The topic is widely studied and many solutions exist. However, logging out of a service using SSO has received less attention. While previous studies note that users want single logout when using SSO, most of the existing services do not offer it, and the identity providers do not even keep track of the open sessions. This article describes challenges related to logout in federated identity management and analyzes unexpected behavior in logout situations. The examples are from the Shibboleth SSO system. Based on the analysis, we give guidelines for implementing reliable logout and describe a polling-based solution for creating a system-wide logout mechanisms that only requires minor changes to the existing code and does not burden the identity provider excessively. In addition to the system-wide logout, our solution gives users the option to log out of only one service. A usability test was conducted to evaluate the solution. The results show that the users liked the ability to choose between the two logout options, but they did not understand the words used to describe them. Another observation was that a majority of the users do not log out of the services at all; they just close the browser window, which should be taken into account in the design of web SSO systems.

1. Introduction

Service providers on the Internet want to authenticate their users in order to offer them better service. This authentication may be needed, for example, for profiling users, for targeted advertising, for charging a fee for the service use, or for protecting the users' personal information. On average, users log in to eight services daily using different passwords (Florêncio and Herley, 2007). They have even more passwords for services they do not use daily, and this sometimes leads to a situation where users either choose poor passwords or recycle them for use with many services (Gaw and Felten, 2006). To help users cope with the authentication, many services employ *single sign-on* (SSO) technology, with enables users to use a single password for many services. There exists several solutions where, basically, the service itself is separated from the user authentication and authorization, which are then

* *Corresponding author.*
E-mail address: sanna.suoranta@aalto.fi (S. Suoranta).

http://dx.doi.org/10.1016/j.jisa.2014.03.005
2214-2126/© 2014 Elsevier Ltd. All rights reserved.

offered by a separate authentication service. Even social media services such as Facebook and Google+ can be used to authenticate users for third party services. Open-source implementations of SSO systems are mostly based on either the Security Assertion Markup Language (SAML) (Cantor et al., 2005; Hirsch et al., 2005) or OAuth (Hardt, 2012) standard, which are also commonly used in commercial products and services. Various proposals have been made to enhance SSO systems, for example, with respect to their architecture (Cahill et al., 2011; Chadwick et al., 2011; Sun et al., 2010a; Takeda et al., 2006), strength of the authentication (Mustafic et al., 2011; Suoranta et al., 2012), usability (Linden and Vilpola, 2005; Sun et al., 2011a, 2011b), and privacy (Alsaleh and Adams, 2006).

Many of the SSO services consider their task complete after the user is authenticated at the beginning of a service session. However, the end of the session is also important. For example, if a user connects to the service using a public computer in a library or cafe and does not log out, the next user of the computer may gain access to the service with the previous user's privileges. Even today when online services are increasingly accessed through personal laptops and smart phones, shared devices are used within families, between friends, as well as widely in developing countries. Moreover, the timeout periods are often quite long because users do not like to have to sign in repeatedly. Thus, the user may forget which devices and web browsers have ongoing open sessions. In any event, if the server end of a connection remains open when the client end closes, this may leave a door open for an attacker to gain access to the service using the half-open session in combination with some other information. Thus, it would be safer to terminate the session also on the server side.

In this article, we have investigated the logout and its implementation in services that use Shibboleth SSO at Aalto University. We found many problems in the logout procedures: none of the services implement single logout, and at least some server-side sessions are usually still active after logout. Since Shibboleth allows services either to handle sessions on their own or to outsource the session management to a local software component called the service provider, sessions managed by both should be completely deleted on logout. However, some of the services we examined leave either the service or service-provider cookies intact upon logout, which in turn allows users to get back in without authentication. Another significant problem was that the user could not, on clicking a logout button, know from where she would be logged out. Some services used local logout to end their local sessions only, while other services also ended the identity-provider sessions. In the latter case, accessing new services requires re-authentication, but other existing service sessions remain active. This is because Shibboleth does not support real single logout.

Based on our findings, we suggest a list of actions that help in implementing a more secure and usable logout for Shibboleth. Some of the solutions are general and could also be used to improve other SSO systems. Moreover, we implement a real single logout solution based on polling. By polling the identity provider, services can check if the authentication session is still active in the identity provider. The identity provider does not need to be changed, and nor does it need to keep track of the services the user has logged in to. In addition, we decided that the user should be able to choose between logging out of one service (*local logout*) and logging out of all the services at the same time (*single logout*). We conducted a usability test with 18 users to check the understanding of both logout options from the users' perspective.

The rest of this article is organized as follows. In Section 2, we describe single sign-on systems in general. In Section 3, we present how logout works in different SSO systems, and in Section 4, overview related work on logout. In Section 5, we discuss the problems of logout in practice. Then we describe our solution for logging out of the services in Section 6. The proposed solution is validated with a usability test in Section 7. Finally, Section 8 ends the article with our conclusions.

This article is an extended version of a previous conference paper (Suoranta et al., 2013).

2. Single sign-on systems

Single sign-on systems are such that the users can access several services by authenticating themselves only once. The SSO systems can be divided into four categories: local and proxy-based, pseudo and true SSO systems (Pashalidis and Mitchell, 2003). In the local pseudo-SSO system, a user's device has a password manager that automatically works on behalf of the user when logging in to a service. The proxy-based pseudo-SSO system is similar, but the password manager is located on a network proxy. The local true SSO system relies on a trusted component in the client device, but otherwise is quite similar to the local pseudo-SSO. The proxy-based true SSO system has an external server for authenticating the user for services. In this article, we concentrate on proxy-based true SSO systems, often called *federated SSO*. For example, OpenID (OpenCommunity, 2007) and Shibboleth (Shibboleth Consortium, 2012) are proxy-based true SSO systems. In addition, many service providers allow other web service developers to use their user authentication with their proprietary protocols. For example, Facebook Login (Facebook, 2013) and Microsoft Live ID (Microsoft, 2012) provide centralized authentication for third-party web applications where users can log in using their existing identities at Facebook and Microsoft services.

In this section, we describe some popular SSO systems. First, we describe federated identity management systems such as SAML 2.0 and Shibboleth and the login process in them, then capability based authorization systems such as OpenID and OAuth, and finally briefly comment on social media services that are also used as SSO systems.

2.1. *Federated Identity Management and Shibboleth*

Liberty Alliance Project (Liberty Alliance Project, 2003; Cantor and Kemp, 2004–2005) introduced the concept of Federated Identity Management (FIM). In FIM, *service providers* (SPs) delegate the user authentication to an *identity provider* (IdP). Nowadays, IdPs are often called *identity service providers* since they do not provide actual identities but services on top of identities. The services and the identity provider form a