The Smartphone as Mobile Authorization Proxy

Luis Roalter[1], Matthias Kranz[2], Stefan Diewald[1], Andreas Möller[1], and Kåre Synnes[2]

[1] Technische Universität München,
Distributed Multimodal Information Processing Group, Munich, Germany
roalter@tum.de,stefan.diewald@tum.de,andreas.moeller@tum.de
[2] Luleå University of Technology,
Department of Computer Science, Electrical and Space Engineering, Luleå, Sweden
matthias.kranz@ltu.se,kare.synnes@ltu.se

Abstract. We present a novel approach to use a mobile device for authentication and authorization purposes, where the user is able to authenticate and authorize himself for access on a public terminal. The concept is based on an extension of a Single-Sign On solution for mobile and public terminals.

1 Motivation for Intuitive Mobile SSO

Internet services have become an integral element in daily activities. Cloud-based services like Google, Facebook or other social community platforms make use of a lot of personalized information of its users. It is essential for the users of these services to protect their data, their data's integrity and their privacy by protecting the access to the account. As many users are using many different services, they usually have to use different logins for the different services – but, due to comfort reasons, often only have one password and probably only one login (typically the email address).

Approaches like OAuth [1], OpenID [2], 'Facebook Connect' or Shibboleth [3] can reduce the duplicate usernames and thereby duplicate vulnerable passwords. The basic idea is to move the authentication to a trusted IDentity Provider (IDP). The user authenticates himself against a well known login mask of the IDP and gets redirected to the calling Resource Provider (RP) that is requesting the authentication.

Single Sign-On (SSO) is one possible option facilitating both comfort and safety requirements. A single action makes it possible to achieve user authentication and authorization and thereby permit a user to access all services he has permission to, without the need to enter credentials multiple times. With the rise of mobile devices, the classical web- or software-based SSO solutions are not adequate anymore. Instead of a single desktop computer, multiple personal and private displays might become part of the interaction. We provide an intuitive SSO solution via mobile devices to access all kinds of services, ranging from data access to interaction with public terminals.

2 Implementation and Technical Background

The idea of token-based service-login is that the user provides his credentials (commonly his user name and password) to his trusted authentication service. Due to the trust-chain between the IDP and the RP, the token can grant the service to access the RP with the user's identity. Every participating RP must have been trusted before by the IDP. With the negotiation between RP and IDP, the service gets an unique Service ID (SID) which is used to identify the resource in the QR code in our approach (cf. Fig. 1). This reduces the possibilities of token hijacking. Furthermore, the token can easily be revoked and renegotiated. Comparing to other standards, the RP can also communicate during the authentication process with the SSO server. This connection builds up the backbone of the system and helps protecting the users' privacy and security.

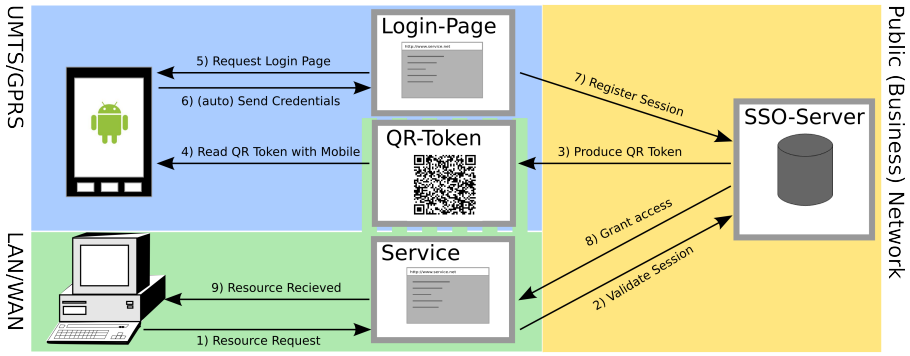

Fig. 1. Mobile Authentication: (1) The user requests a resource/service (e.g. a file or mail). (2) The RP searches for a valid running session within the SSO Server (IDP). (3) The IDP produces a token for the terminal. (4) The user scans the code and gets an URL. (5) The user requests his login-page. (5) The user sends his credentials to the IDP (or is authenticated by a native application). (6) The IDP registers the session for the user. (7) The RP gets informed the access has been granted. (8) The RP can provide the user with the resource.

A current disadvantage of most existing systems is that they cannot be used without keyboard. This poses a significant restriction, especially if typing involves long and arbitrary sequences of letters which are typically forming a token (which is cumbersome on a mobile device). All systems are built on top of the basic username/password authentication mechanism. There are many reasons for having the user *not* type his password on public displays: First the display could have been hacked, or second, typing on a large and mounted touchscreen makes it easy for a third person to oversee the password. To remove the keyboard and the possible *man-in-the-middle attack* of a potentially insecure public display, the authentication is no longer done on the display, as a mobile device can authenticate the user independently from the public terminal.

The IDP will generate a *QR Code Token* for the visual auxiliary channel[4], displayed by the service within the RP. To protect users against phishing, the QR Code is generated by the IDP and *always* references an URL to a known authentication server (if it diverts from the predefined IDP, it is replaced on the mobile). Furthermore, this helps to filter the URL for special handling on smartphones, providing a more comfortable login method with a native application than showing up a website.

The authentication process follows the steps for authentication as defined by OAuth and OpenID, but has been tailored towards the specific needs of mobile device users and potentially untrusted public terminals. It would also be possible to use the approach with desktop computing system, where the login manager could be adopted to provide a QR code as described above. Future research will include the sketched extension to desktop computing and the verification of the effectiveness, efficiency and user acceptance in field trials and user studies.

References

1. Hammer-Lahav, E.: RFC 5849 - The OAuth 1.0 protocol. Technical report, IETF (2010)
2. Recordon, D., Reed, D.: OpenID 2.0: A Platform for User-Centric Identity Management. In: Proceedings of the 2nd ACM workshop on Digital Identity Management. DIM '06, New York, NY, USA, ACM (2006) 11–16
3. Erdos, M., Cantor, S.: Shibboleth-Architecture Draft v05. Internet2/MACE (May 2002)
4. Mayrhofer, R., Fuss, J., Ion, I.: UACAP: A Unified Auxiliary Channel Authentication Protocol. IEEE Transactions on Mobile Computing **99**(PrePrints) (2012)