User-friendly Authentication and Authorization using a Smartphone Proxy

Luis Roalter[1], Stefan Diewald[1], Andreas Möller[1],
Tobias Stockinger[2], and Matthias Kranz[2]

[1] Technische Universität München,
Institute for Media Technology, Munich, Germany
roalter@tum.de, stefan.diewald@tum.de, andreas.moeller@tum.de
[2] Universität Passau,
Lehrstuhl für Informatik mit Schwerpunkt Eingebettete Systeme, Passau, Germany
tobias.stockinger@uni-passau.de, matthias.kranz@uni-passau.de

Abstract. We present a novel approach to authenticate and authorize a user, using her personal smartphone. The presented architecture is complemented with a proof-of-concept implementation. The implemented system architecture is based on a single sign-on solution (SSO), extended to allow the usage of the smartphone as authentication and authorization device. We evaluated the system within real-world scenarios, observing users' behavior using the novel technique. Based on our experiences, we summarize advances, made both in usability and security, for novel implementations using the proposed concept.

Keywords: Mobile Computing, Mobile Authentication, Mobile Authorization, Proxy Authentication, Mobile Security, Secure Authentication, Single Sign-On

1 Introduction

Internet services have become an integral part of our daily activities. Cloud-based social networks like Google+ or Facebook, or services like Evernote, Dropbox or GitHub, process personal information about their users. Security and confidentiality of personal data is essential to the users. At present, the protection of the account, e.g. when accessed from a terminal, is achieved by asking for the user's credentials. Although there are already existing implementations using physical tokens (RSA tokens, smart cards, etc.) or pre-shared knowledge for access control, in most of today's services a username (e.g. the e-mail address) and the associated password are still the dominant way for user authentication. As a consequence, users have to memorize different credentials and enter them into the services' individual login masks. However, due to reasons of convenience, users often use the same login and password for all (or at least multiple) services, which represents a security risk.

The emergence of *single sign-on* (SSO) standards such as OAuth [1], OpenID [2], 'Facebook Connect' [3], or Shibboleth [4] does not require an individual registration at every web service any more. Instead, users can authenticate via an *existing* and *well-known* login mask. The actual authentication process is done using a so-called *Identity Provider* (IDP) that is independent from the system the user wants to access. This leads to a separation of the service from the so-called AAA mechanisms (authentication, authorization and accountability).

Single sign-on is one possible option for facilitating both comfort and safety requirements. A single action achieves user authentication and authorization conveniently and thereby permits the user to access all services, without the need to enter credentials multiple times. This works for web- and desktop-based services, for example, through a

locally running background application (e.g. in business SSO solutions [5]) or through a trusted cookie or authentication token, provided by the IDP.

However, with mobile devices, interaction is no longer limited to a single desktop computer. Private smartphones, tablets, or even public displays (with all of them using different interaction paradigms) might become part of the interaction process involved into the sign-on procedure. Existing authentication mechanisms might have to be adapted to new multi-device authentication procedures, as the user might not want to enter her credentials on these publicly exposed devices. The reason can be the size of the display, or security concerns of the user. Furthermore, mobile devices can be used for two-step authentication where the username/password combination is complemented with a verification code received by the mobile device (Google's 2-Step Authentication[3], one-time passwords [6], etc.).

We present a more intuitive and user-friendly SSO solution, optimized for usage with mobile devices. The smartphone is a typical personal companion that could be used to grant access to different services. The possibilities range from permitting data access on a web service to interaction with public terminals, without having to enter any credentials on the actually target device (e.g. a public terminal).

The paper is structured as follows. First, we give an overview on related work (Sec. 2), outlining advantages and disadvantages of existing approaches. Afterwards, in Sec. 3, we present our approach, starting with the initial concept and architecture, followed by the implementation details in Sec. 4. We present selected use cases in Sec. 5 for our implementation, and conclude in Sec. 6 with a summary and outlook for future research.

2 Related Work

Trusted authentication and authorization of users in local and web-based services requires implementations with an increasing complexity due to the higher amount of participating components (e.g. service providers, databases, security etc.). However, entering username and password are still the predominant login mechanism for authentication, even if there are more suitable interaction modalities already possible. Currently, novel concepts for user authentication are explored. Many of these approaches integrate the smartphone as the user's companion into the login process [7, 8]. Selected authentication procedures will be presented and discussed in this paper.

Authentication techniques using additional devices are often based on auxiliary channels, which are used besides the actual interaction to communicate uni- or bi-directionally with the service and the identity provider. Due to the wide availability of private smartphones, authentication and authorization scenarios should account for and include such devices. Mayrhofer et al. [9] present a protocol specification which can be used to authenticate the user employing different auxiliary channels, such as visual, auditive or other sensing channels. All of these perception channels can be used to simultaneously transmit user-related data (e.g. token, username, and password hash) to an authentication server or identity provider (IDP). A similar approach has also been proposed by Mizuno et al. [10]. One of the potential disadvantages of these approaches lies in the required active participation of the user. The user, for example, needs to dial a number, send a message to a service, or explicitly open a prepared web page, for finishing the authentication procedure (e.g. in OpenID or Facebook Connect).

We argue that the usability in such authentication scenarios (e.g. holding the microphone in a specific direction, typing a code in an application etc.) can significantly

[3] Google 2-Step, `http://www.google.com/landing/2step`

be improved. Furthermore, the disadvantage of most existing terminal systems (independent from the smartphone) is that they cannot be used without a keyboard. This results in a significant restriction, especially if long and arbitrary sequences of characters (e.g. a hex-token) need to be typed by the user. This can be cumbersome on the mobile device's tiny keyboard [7, 11]. They present an approach where the user can transfer the keyboard from the (public) terminal to her smartphone. There, the user can enter text without exposing the keyboard to strangers. However, potential problems with trust towards the public terminal remain.

To lower the barrier for users to adapt novel technologies, a combination of transferring the input device to a trusted platform and automatically starting a challenge-response authentication and authorization is desirable. Vapen et al. [8] present such a visual challenge-response authentication, similar to the proposed implementation in this work. The user authenticates herself via a login form and gets a visual code to verify her identity. The authors focus on the implementation of a two-step-authentication. In our approach we create a more intuitive and user-friendly interface to simplify the authentication process, including a complete session management from the user-side. A similar approach has been proposed with Google Authenticator[4]. Another approach was experimented with by Google until January 2012: With Google Sesame, the user was able to login to a service just by scanning a QR code[5]. A *man-in-the-middle* attack in these approaches is truly hindered, as the user needs to provide a time-changing token from an independent device. Yet, like most other systems, these systems are still built on top of the basic username/password authentication mechanism. All of these implementations do not provide significantly improved usability or a more intuitive interaction to the user. Most implementations also lower the performance for the login process, such as the time spent during the actual authentication procedure.

3 Concept and Software Architecture

The idea of a token-based service login is that the user provides her credentials (commonly her username and password) *only* to the trusted authentication service. Due to the established *chain of trust* between the identity provider (IDP) and the service, a single user-related token can grant access to the service. The token is correlated with the user's identity. Ideally, the token is stored on a trusted personal and portable device, such as the smartphone, which authenticates against the identity provider only once. In contrast to other existing two-step-authentication models, credentials are *not* provided to the service. The service starts the login process without actually knowing the user's identity, registering the login session at the IDP. The received session-token is presented to the user as a human-readable code and can be transferred via NFC, or via the camera as QR code to the user's smartphone. In our approach, we use QR code markers for transferring data between the terminal and the smartphone (cf. Fig. 1).

The token shown by the service can be structured in two ways, both based on an unified resource identifier (URI). The first way uses an HTTP URI for identifying the location of the IDP, providing the session-token as parameter. A secure transport of the token can be assured e.g. by using a SSL-encoded connection. Using standard HTTP locations has the advantage that authentication can be performed on various platforms (e.g. in the mobile web-browser). In the second approach, the URI contains the specialized schemata *otpauth*[6] or *votp*[7]. (c.f. Fig. 2). To enable authentication using

[4] Google Authenticator, `http://code.google.com/p/google-authenticator/`
[5] Article on Lifehacker.com, `http://lifehacker.com/5876559`
[6] One Time Password Authentication
[7] Visual OTP

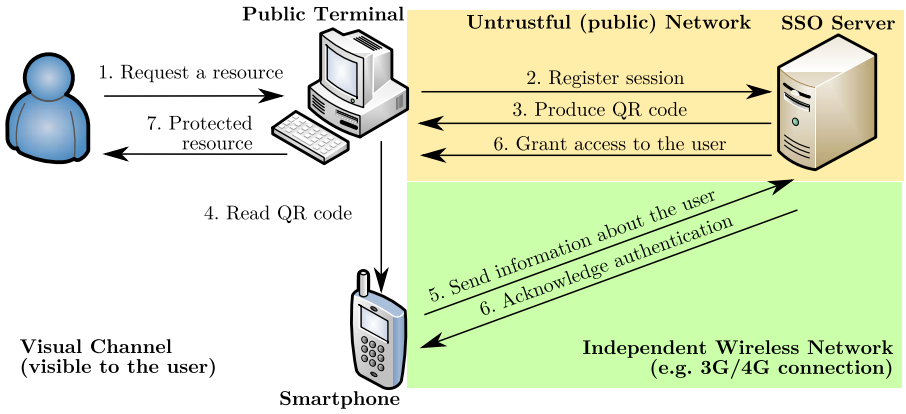

Fig. 1. Basic Architecture for Mobile Authentication: (1) The user requests a resource/service (e.g. a file or mail). (2) The service searches for a valid session within the SSO server (IDP). (3) The IDP produces a token for the terminal which is displayed by the terminal. (4) The user scans the code and gets a defined URI. (5) The user sends her authentication data to the SSO server. (6) The IDP registers the session for the user and grants access to the service. Simultaneously the authentication result is shown on her smartphone. (7) The service can provide the user with the requested resource.

such URIs, a specialized application must be installed on the user's smartphone. In our approach, her smartphone will be equipped with a dedicated authenticator software, which handles any request related to the *otpauth* and *votp* schemata. The authentication software can be obtained from trusted sources, such as app stores [12]. One advantage of this implementation lies a fast and ease completion of the authentication procedure. During the actual authentication procedure, the content of the QR code is extended with the user's personal token and transmitted to the preselected identity provider. This ensures that the authentication token is transmitted to the trusted identity provider only. Another advantage is that the personal token of the user can be stored on the smartphone and reused during every authentication, without transmitting username and password again. The smartphone application further allows the user to gain an overview on her currently authenticated sessions, allowing her to terminate a running session without accessing the actual terminal.

4 System Architecture and Implementation

In this work, we moved the login process from the private-public terminal to an additional personal device. This allows to avoid the on-screen keyboard for entering personal information. The actual login process, as presented in Fig. 1, will be realized with an Android application running on the user's smartphone. The visual channel, realized by the smartphone's camera and a QR code, enables the user to transfer session-related

Fig. 2. Example structure of a Resource Identifier (URI) encoded in the QR code. The session ID is hashed using a secure hash function. For synchronization and validity estimation, the timestamp will be transported as well.

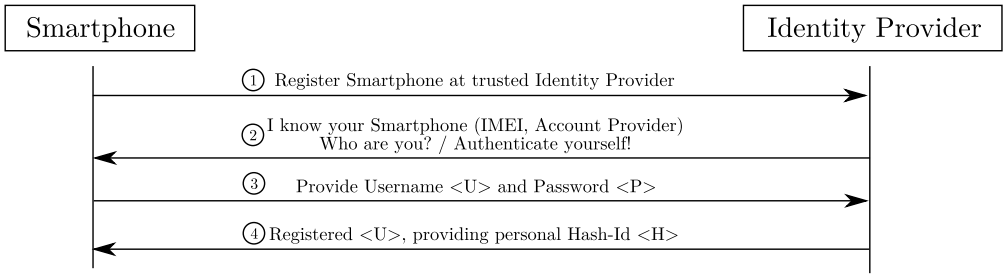

Fig. 3. The initial registration process: (1) The smartphone is registered *once* on a selected and trusted identity provider (IDP). The IDP will present its login page (2), requesting the user's username <U> and password <P> (3). This is the only situation the user must provide her username and password. As result (4), a user-related token <H> with limited lifetime will be generated and used as one-time password (OTP) in the login phase. This token will can be renewed from the smartphone without providing the user's credentials within a reasonable period of time.

information onto her smartphone. This information is used to authenticate and authorize the user for the requesting service. As already mentioned in the last section, two different authentication paradigms are possible:

1. The user is required to fill out the login mask of the IDP on her smartphone. To protect her privacy, she can use the smartphone's auto-complete function or hide the smartphone from the surrounding people while typing.
2. The user uses a pre-installed application that is able to fetch the information read by the smartphone's barcode scanner. The visual (public) information can then be combined with the pre-authenticated user information. This decreases the delay during the authentication process.

In our implementation, we focus on the second implementation to authenticate the user. Prior to any authentication against a public-private display or device, the smartphone needs to get known to the IDP in advance. To do so, the user will authenticate herself with her smartphone towards the IDP in an initial registration process (cf. Fig. 3). This will be the only time that the user provides her credentials in the form of username and password. After successful authentication of the user, the application will store the user- and smartphone-dependent authentication token in the smartphone's account database.

Once the user and her smartphone are known to the IDP, she can authenticate herself to any service registered at the selected IDP. An example of such a login process is illustrated in Fig. 4. The flow diagram additionally shows how the user can use her smartphone to invalidate any session she has previously established (cf. Fig. 4, 8-9). In contrast to existing login methods (e.g. Google 2-Step-Authentication or 2-clickAuth [8]), the user is not known in advance. The IDP provides the calling service with names and permissions of the user as long as the session is kept valid by the user.

5 Use Cases

To evaluate our implementation, we present three different use cases, where the smartphone-based authentication could increase the usability and security during the authentication process.

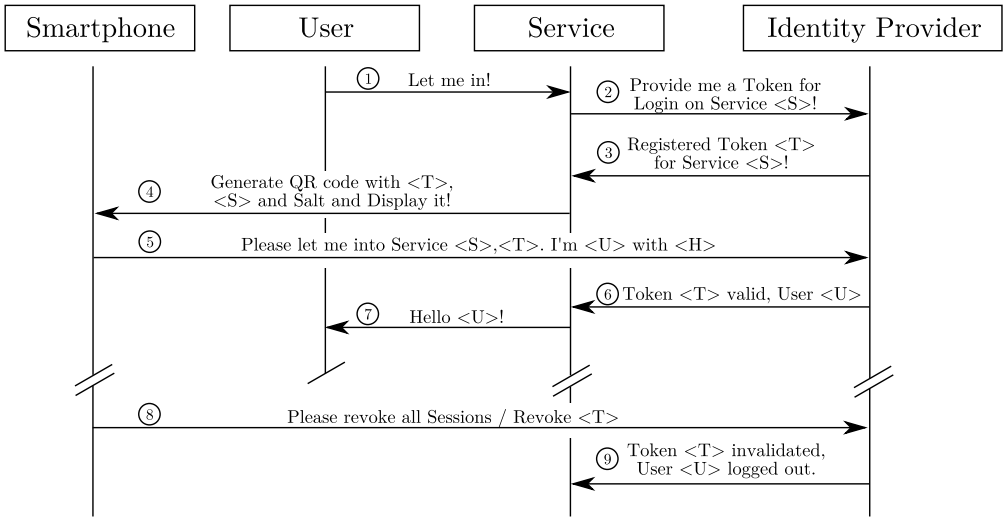

Fig. 4. *Top:* The user wants to login (1) to a protected service. In the beginning, the service <S> registers a time-limited session at the identity provider (2) and receives a token <T> (3). Thus allows the service to generate a (QR) code for the user's smartphone (4). The smartphone application appends the received code with the personal user hash <H> and sends it to the identity provider (5). After verification, the service is told about the authenticated user by the IDP (6) and can grant access to the resource (7). *Bottom:* The user is able to invalidate a specific session (or all sessions) with her smartphone (8). The service will receive a notification about the revoked token <T> (9). In addition, all sessions have a limited lifetime after which they automatically expire.

5.1 Use Case 1: Public Displays / Pervasive Displays:

Publicly accessible displays are not always equipped with trustful input devices. The input devices might be hijacked or monitored by a third person. In the worst case, the user's credentials might be captured and her account could be compromised. Despite the lack of a verifiable chain of trust, the user, in realistic situations, might want to use a public display to leave message or share new content with the public display as presented in Fig. 5. In the scenario the user scans the QR code presented by the public terminal with her smartphone. This allows her to e.g. comment, modify and share content presented on the public display.

Fig. 5. Mock-Up of an authentication at a public display. The person in the picture is authenticated with its smartphone against the identity provider stored on the device. This allows the person to leave a personalized comment or share the webpage with others.

5.2 Use Case 2: Digital Door Sign / Access Hardware Resources:

Besides large public displays used e.g. for advertising (compare Müller et al. [13]), smaller interaction devices, such as a digital door signs, are possible. (cf. Fig. 6 [14, 15]). With such devices, interaction might be limited due to the restricted input modalities. This can especially be problematic when a user needs to be authorized to e.g. open the door. The smaller the screen, the more error-prone authentication might be because of the limited size of the keyboard. The proposed system, using the smartphone for authentication, allows even small devices capable to produce a readable code, e.g. via QR code or near-field communication (NFC), to authenticate the user without using a real or virtual keyboard.

5.3 Use Case 3: Personal Computers / Novel Login Methods:

Another use case is the authentication on desktop computers via the user's personal device. The computer's login prompt could display a visual code that can be acquired by the user's smartphone (cf. Fig. 7). When the user scans the time-limited code, the IDP can grant access to the service with the user's credentials retrieved from the smartphone. In contrast to a conventional login, the username is not requested by the login mask. This further protects the user's privacy as the username is only exchanged between the smartphone and the IDP.

6 Summary and Future Work

In this paper, we have motivated the need for more comfortable and secure login procedures in the context of mobile devices and public-private display interaction. We have presented an approach and software architecture for an intuitive and effective authentication process using the user's smartphone to confirm her identity. The system was evaluated in the context of three different use cases. The presented authentication process follows the authentication steps as defined by OAuth and OpenID, but are tailored towards the specific needs of mobile devices, users, and potentially untrusted public terminals.

We believe that our approach can be used to provide a more intuitive, easy and secure authentication. Future research will include more detailed field trials and user studies on the presented use cases, including the verification of the effectiveness, efficiency and user acceptance.

In one of the first application scenarios, the system will be deployed on a larger university campus. Users will be students, wanting to book resources such as rooms,

Fig. 6. Authentication at the door's entrance. The person is authenticated against the mounted interactive device with the smartphone. This allows the person to book an event, open the door and enter the room nearby (photo taken of a user during the login-process with out prototypical implementation).

Fig. 7. The proposed login method for personal computer devices. The person in the picture using our prototype scans the time-limited QR code, which will provide username and permission to login to the service (e.g. to the login screen).

accessing CIP-pools, or sharing content of public displays on their social networks. We hope that, by providing an innovative and simple interaction method for authentication, we can encourage users to experiment with the service and make more efficient use of the resources available and allowing us to study this interaction in more detail.

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. Miculan, M., Urban, C.: Formal analysis of Facebook Connect single sign-on authentication protocol. In: SOFSEM. Volume 11. (2011) 22–28
4. Erdos, M., Cantor, S.: Shibboleth-Architecture Draft v05. Internet2/MACE (May 2002)
5. Parker, T.: Single Sign-On Systems – The Technologies and the Products. In: European Convention on Security and Detection. (May 1995) 151–155
6. M'Raihi, D., Bellare, M., Hoornaert, F., Naccache, D., Ranen, O.: RFC 4226 – HOTP: An HMAC-Based One-Time Password Algorithm. Technical Report 4226, IETF (Dec 2005)
7. De Luca, A., Frauendienst, B., Boring, S., Hussmann, H.: My Phone is my Keypad: Privacy-Enhanced PIN-Entry on Public Terminals. In: Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7. OZCHI '09, New York, NY, USA, ACM (2009) 401–404
8. Vapen, A., Byers, D., Shahmehri, N.: 2-clickAuth Optical Challenge-Response Authentication. In: International Conference on Availability, Reliability, and Security, IEEE (2010) 79–86
9. Mayrhofer, R., Fuss, J., Ion, I.: UACAP: A Unified Auxiliary Channel Authentication Protocol. IEEE Transactions on Mobile Computing **99** (2012)
10. Mizuno, S., Yamada, K., Takahashi, K.: Authentication Using Multiple Communication Channels. In: Proceedings of the 2005 workshop on Digital identity management. DIM '05, New York, NY, USA, ACM (2005) 54–62
11. De Luca, A., Frauendienst, B.: A Privacy-Respectful Input Method for Public Terminals. In: Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges. NordiCHI '08, New York, NY, USA, ACM (2008) 455–458
12. Möller, A., Michahelles, F., Diewald, S., Roalter, L., Kranz, M.: Update Behavior in App Markets and Security Implications: A Case Study in Google Play. In Poppinga, B., ed.: Proceedings of the 3rd International Workshop on Research in the Large. Held in Conjunction with Mobile HCI. (Sep 2012) 3–6
13. Müller, J., Alt, F., Michelis, D.: Pervasive Advertising. Pervasive Advertising (2011) 1–29
14. Kranz, M., Holleis, P., Schmidt, A.: Ubiquitous Presence Systems. In: Proceedings of the 2006 ACM Symposium on Applied Computing, New York, NY, USA, ACM (2006) 1902–1909
15. Davies, N., Langheinrich, M., Jose, R., Schmidt, A.: Open Display Networks: A Communications Medium for the 21st Century. Computer **45**(5) (May 2012) 58–64