UAuth: A Strong Authentication Method from Personal Devices to Multi-accounts

Yazhe Wang, Mingming Hu(✉), and Chen Li

State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{wangyazhe,humingming}@iie.ac.cn, leec402@sina.com

Abstract. In this paper we present UAuth, a two-layer authentication framework that provides more security assurances than two-factor authentication while offering a simpler authentication experience. When authenticating, users first verified their static credentials (such as password, fingerprint, etc.) in the local layer, then submit the OTP-signed response generated by their device to the server to complete the server-layer authentication. We also propose the three-level account association mechanism, which completes the association of devices, users and services, establishing a mapping from a user's device to the user's accounts in the Internet. Users can easily gain access to different service via a single personal device. Our goal is to provide a quick and convenient SSO-like login process on the basis of security authentication. To meet the goal, we implement our UAuth, and evaluate our designs.

Keywords: Authentication · Mobile terminal · Multi-accounts

1 Introduction

As the network's development and popularity of the Internet, many people today have multiple accounts in the Internet. If one uses different and unrelated passwords for each account, the coming up with secured passwords to remember is a very challenging task for him. Single Sign-On (SSO) allows users to sign in numerous relying party (RP) websites using one single identity provider (IdP) account. Therefore, users are relieved from the huge burden of registering many online accounts and remembering many passwords. However, it just reduces the problem of securely authenticating to relying parties to the one of securely authenticating to an identity provider. It does not, in fact, address the issue of securely authenticating. Now some popular SSO services (e.g., OpenID [9]) still use the traditional password authentication. An adversary who manages to steal the password in IdP from a legitimate user can impersonate that user to the trusted RPs, which leads to a chain reaction of resource misuse. Recently, some password leaks [7] highlight the current traditional password authentication vulnerability. Though some additional encryption measures have been taken, users transmit the hash of password instead of plain text or transmit above SSL.

J. Tian et al. (Eds.): SecureComm 2014, Part I, LNICST 152, pp. 95–104, 2015.
DOI: 10.1007/978-3-319-23829-6_7

The emergence of powerful password-cracking platforms [10] or the use of vulnerabilities [1] has enabled attackers to recover the original passwords in an efficient manner, an attacker can still impersonate a legitimate user login into the website using the recovered password. We can find that all these vulnerabilities arise primarily due to the sensitive authentication credentials (e.g., password) are verified in the server layer, the credentials are not dynamic and are transmitted over the insecure Internet. Each time when user login into a website, they use the same credentials, so if an attacker steals the credentials, he can impersonate the user at any time without worry about the password failure.

We propose Uniform Authentication (UAuth), a two-layer authentication framework. In addition to the server layer verification, a local layer verification is provided. The sensitive static authentication credentials are verified in local layer while the dynamic credential is verified after submit to the server layer. The unpredictability of the credentials and multi-layer authentication make the attacker have no approach to access the sensitive data, significantly improves the security of the authentication. The UAuth also provides a three-level account association about the mobile terminal, the account in UAuth and the account in SP, which significantly reduces identity management and authentication infrastructure complexity. FIDO (Fast IDentity Online) Alliance [5] has proposed similar ideas, from its newly published specifications we can find that it concentrates little on Federated Login. And the discussion is based only on the high-level description. We make some improvement from it and develop a system that runs correctly. We also give the detailed implementation.

2 Related Work

Two-factor authentication, such as Google 2-Step Verification [4], utilizes two factors from independent channel when authenticating. But if users reuse passwords across different websites [14], at which point once the attacker get the password in the site which employ two-factor authentication, they would be able to impersonate the user in other sites which don't employ two-factor authentication [3,12]. It will also lead to poor user experience when copy the string of the OTP from a mobile phone to the login page. Czeskis et al. present PhoneAuth [13], an authentication method that does not require operation of the phone. In its strict mode, there is no user interaction necessary during a login, other than typing the username and password. However, without user's operations, the automatic authentication can also lead to potential threats. At the same time, with the increase of account, the device that authentication requires also increases, it is quite inconvenient either in portability or cost.

Kontaxis et al. present SAuth [17]. A protocol for synergy-based enhanced authentication. But it is obviously that it's a single factor authentication method, the security has not greatly improved. The YubiKey [6] by Yubico is a kind of authentication token, users can use the One-Time Code that Yubikey generates as the second factor. However, it also meets the problem that two-factor authentication encounters, such as the password reuse and the portability issues.

Several promising services are now available at various stages of polish, each with their own vision of user identification and authentication. The study by Bonneau et al. in 2012 lists some popular authentication mechanisms and critically analyzes them via a framework of 25 different "benefits" that authentication mechanisms should provide [11]. Reference [13] also give an evaluation about their work using the framework. We agree with most of the analysis and rate our system under that framework.

3 Threat Model

We allow adversaries to obtain the user's password - either through phishing or by social engineering attacks, but he can't simultaneously get the password and the user's mobile terminal. The browser in the fixed terminal uses the certificate in AP to establish an SSL connection with the server. We assume that the data in the mobile terminal is stored in security storage, only certain procedures can access their own resources. The attackers can perform software attacks against the terminal and install, modify or compromise all software components installed on the terminal. But it's obviously that they are unable to visit the data belongs to UAuth application in the security storage. The attacker is also able to deploy some malware on the user's machine, such as a keylogger. The malware have the access to the document in the user's machine and they can also visit the data in the browser. However, the attacker is not able to simultaneously compromise the user's PC and user's mobile terminal.

Since there will be some sensitive data transmission between the mobile terminal, the user's PC and the UAuth server during the initial authentication step, the attacker may directly access the data easily. But considering that the frequency of these cases is low. We choose to focus on the subsequent case after the initial step.

4 Architecture

4.1 System Model

Our system model is depicted in Fig. 1. The design consists of several categories of components: Authenticate Plug-in (AP) in the fixed terminal (B), mobile terminal (M), Validate Server (VS), Validation Cache (VC), UAuth Web Server (WS), Server Provider (SP). They have completed the three-level association system: the binding, authorization, management between the mobile terminal identification information (OID), the user account in WS (UID) and the user account in SP (SPID). Using UID as the medium, users can use their own terminal to get access to the Internet service. Moreover, all the association is controlled by WS, and WS is able to create a management module to complete the multi-binding, which means users are capable of binding more than one terminal or SP to their own UID. As can be seen from Fig. 2, multiple OID as well as SPID are bound to UID. So when authenticating, user can choose the appropriate device

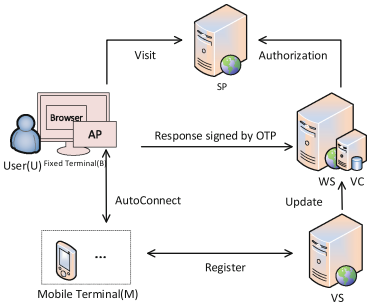

Fig. 1. System model

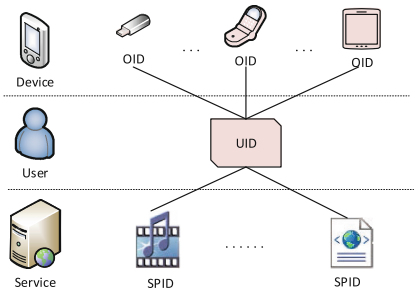

Fig. 2. Three-level account association

to visit the Internet service he wants. Three-level account association has greatly increased the convenience and flexibility of authentication.

VS provide registration functions, all the mobile terminals need to be registered in it prior to use. It will negotiate the OID and the key that used to generate the OTP (One Time Password [2]) with mobile terminals, store these data and update it to the VC. VC is a caching server for the data in VS, which is physical proximity to the WS. Each time after WS submits the OID and response, it can efficiently determine whether the OID and response correspond or not. WS is the core part of UAuth, with which user can manage their UID and account binding (UID and OID binding, UID and SPID binding). Users need to get the credential of SPID from WS when they are authenticating to SP. The mobile terminal can be various, but they all provide a local layer authentication method. The terminal registers itself to the VS by negotiating the key used to generate the OTP and telling the server it's OID. It would not generate the OTP to achieve the server layer authentication unless the local layer authentication is succeeded. AP is a customized functional component installed in use's fixed terminal, it helps to complete the authentication by establishing a communication between the WS and user's mobile terminal. It also informs the WS of the presence of a mobile terminal, and relays the encrypted authentication stream to WS. SP is the entity that provides Internet services, it needs to establish a trust relation with WS and build a secure communication channel.

4.2 UAuth Details

Initialization. Before login with this method, users have to initialize the mobile phone and the fixed terminal.

The user installs the authentication application in the mobile phone and initializes it. It will connect to the VS server to register itself, negotiate to get the user's mobile phone private certificate. The certificate is used to identify itself, and establish an SSL session with fixed terminal. While they also negotiate to get the OID, as well as to get key K which used to generate OTP. After successful

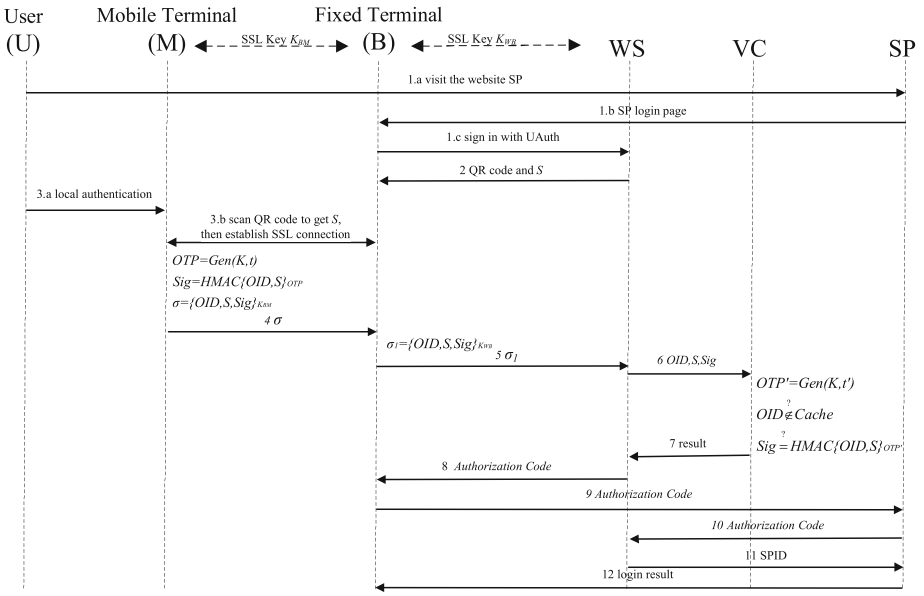

Fig. 3. Protocol details

registration, VS will update the registration information to VC. The user should also bind his UID to OID and SPID respectively in the WS page.

The user also has to install the AP in fixed terminal, which assist to achieve a complete login process.

Protocol Details. When users are authenticating with SP, they have to successfully authenticate to our WS via a two-layer authentication: after the local layer verification on the mobile terminal, users will forward the response generated by the mobile terminal on-the-fly to the WS, then obtain the credential of SP account and login to SP. As shown in Fig. 3, a complete authentication requires these steps. Since in our framework the mobile terminal is not limited to a certain type, which can be a variety of devices, in this paper we use smart phone as an example to illustrate a specific process.

If one uses UAuth to sign in the website SP, the page will redirect to the login page on WS (step 1). WS generates a QR code with a random string S in it and displays it on the use's browser. Meanwhile, the AP in the browser will establish a temporary Wi-Fi access point using the information calculated from S (step 2). The user will unlock the UAuth application in the Mobile Terminal M, which is the local authentication procedure. Then scan the QR code displayed on the browser to get S, M uses the same method like AP to calculate the information from S, with the information M connects to the access point in B and establishes an SSL session at last (step 3). M generates σ and replies it to B (step 4). After SSL decryption, AP obtains the data in σ and encrypts it with K_{WB}, then

forwards σ_1 to WS through the SSL session. AP will close the temporary Wi-Fi access point once it receives the data from M (step 5). WS gets σ_1 and decrypts it, verifies S that it is generated by WS and it is not expired. Upon successful verification, WS forwards the OID, S and Sig to VC and submits a request to verify the validity of these data (step 6). VC verifies OID and Sig, and responds to WS with the result (step 7). If validation passes, it means that the UID account OID bound to login successful. WS page displays the SP accounts that the UID has bound with. The user is required to select the SP account which he wants to login into SP. WS will randomly generate a *Authorization Code* mapping to the selected SP account. Finally WS returns a link that will redirect to SP with parameters, which contain *Authorization Code* (step 8). B submit the *Authorization Code* to SP so SP can gain the authorization of SPID with it (step 9–11). Finally, SP show the login result on B (step 12).

Discussion. A complete login process of UAuth consists of two-layer authentication. For user's personal device is varied, they may select a smartphone, a USB token, a fingerprint reader or others as an authentication terminal. It enables users to choose a appropriate method depending on the device they are using. For example, user can use password verification when they use their phone or use their fingerprint to authentication with their fingerprint reader. A success verification of user's password or other authentication information in local layer, prove that user has the corresponding authentication device, and the device is not used by a impersonate user. The server layer authentication of OTP-signed response generated by the mobile device, indicate the user who is authenticating and prove that the legitimate user is using the correct device to login again. Since OTP is changing with time, even if an attacker to crack the encrypted channel and get the response, OTP at this time is likely to have been in ineffective. The signature of the data in the transfer process also protects the integrity of the data, with it an attacker is difficult to modify or forgery the data. The use of the combination of the two layers, either prevents the attacker to steal the static password when transmitting in the insecure network, or protects the data in user's device from being read and used when the device is lost, at the same time the response validation ensures the security in authentication process.

5 Implementation and Evaluation

In order to demonstrate the feasibility of UAuth, we implemented the system discussed in Sect. 4, which includes Authenticate Plug-in (AP) in Chrome web browser, Validate Server (VS), Validation Cache (VC),UAuth Web Server (WS) and Android application.

We evaluated our system using Bonneau et al.'s framework of 25 different "benefits" that authentication mechanisms should provide, and analyzed our work from three aspects: the usability, deployability and security. We also include the incumbent passwords, Google 2-step verification, PhoneAuth and SAuth as a baseline. The results of our evaluation are shown in Table 1.

Usability. In the usability arena, since there is little user interaction necessary during a login, and existence of private mobile terminal certificate enabling that one mobile terminal can be bound to multiple UID, and the UID can bind a plurality of SPID, so it is *Scalable-for-Users*. It is *Easy-to-Learn* and *Easy-to-Use* because users only need to enter a password and use the mobile to scan the QR code once, which also makes it does not have *Memorywise-Effortless*, *Nothing-to-Carry* and *Physically-Effortless*. We rated it as somewhat providing the *Infrequent-Errors* benefit since they will cause an error if the wireless connection does not work or if the mobile terminal in not available. Similar to Google 2-step Verification, it somewhat provides the *Easy-Recovery-from-Loss* because of the inconvenience of having to replace the phone is then compounded by the fact that the lost phone also holds the secrets.

Deployability. Accessing the deployability benefits comes down to evaluating how much change would be required in current systems in order to get our proposed system adopted. In UAuth system, since the authentication is completed by the phone in conjunction with local password, and requires AP to play a role in the process, Accessible, *Negligible-Cost-Per-User* and *Browser-Compatible* is somewhat provided. For general SP server, it only needs to be modified to be compatible with UAuth, the change is little so it is somewhat *Server-Compatible*.

Security. Since the UAuth employs both local password and OTP, it can meet most of the security properties. When faced with phishing and eavesdropping, it would have a good security performance. However, some real-time attack makes it vulnerable. The adversary may steal the sensitive data in this attack, so it somewhat provides *Resilient-to-Internal-Observation* benefits.

Conclusion. In Table 1, it can be seen that UAuth owns a good performance in the evaluation and get a high score in the ease of Usability, Deployability and Security.

Performance. We evaluated the performance of our implementation of UAuth: (1) the time required to establish the Wi-Fi access point, (2) the time required to establish the connection between the fixed terminal and the mobile terminal, and (3) the time required to complete the UID login process. That is to say we measure the time between the step 1–8, this can reflect the total performance since the step 9–12 spends little time that can be ignored (usually less than 0.2 s). The result of our measurements averaged over 50 protocol runs are shown in Table 2. Remove the time of user operations (such as take their phone out of their pocket and unlock it), the total time of a complete login process most spent on establishing a Wi-Fi access point and the connection between the mobile terminal and fixed terminal. They are about 1.1 s and 4.2 s. And our solutions requires about 11.4 s for the whole process. Note that for a user that uses 2-factor

Table 1. Comparison of UAuth against password, Google 2-step Verification, PhoneAuth and SAuth using Bonneau et al.'s evaluation framework. 'y' means the benefit is provided, 's' means the benefit is somewhat provided, while blank means the benefit is not provided

	Usability							Deployability						Security											
Scheme	Memorywise-Effortless	Scalable-for-Users	Nothing-to-Carry	Physically-Effortless	Easy-to-Learn	Easy-to-Use	Infrequent-Errors	Easy-Recovery-from-Loss	Accessible	Negligible-Cost-Per-User	Server-Compatible	Browser-Compatible	Mature	Non-Proprietary	Resilient-to-Physical-Observation	Resilient-to-Targeted-Impersonation	Resilient-to-Throttled-Guessing	Resilient-to-Unthrottled-Guessing	Resilient-to-Internal-Observation	Resilient-to-Leaks-from-Other-Verifiers	Resilient-to-Phishing	Resilient-to-Theft	No-Trusted-Third-Party	Requiring-Explicit-Consent	Unlinkable
Passwords		y		y	y	s	y	y	y	y	y	y	y	y		s						y	y	y	y
Google 2-Step Verification		s		y	s	s	s	s	s			y	y		s	s	y	y		y	y	y	y	y	y	y
PhoneAuth-strict	s			y	y	s	y	y	s	s	s	s	y	y	y	y	y	y	s	y	y	y	y	y	s	
SAuth		y		y	y	s	y	y	y			y	s	y		s						y	y	y	y	
UAuth	y			y	y	s	s	s	s	s	s	s	y	y	y	y	y	y	s	y	y	y	y	y	y	

login service, he will type a username and password, and copy the OTP from the mobile phone to the page, which will take an average login time of 24.5 s [13]. Login has speed up with our system, while at the same time improving the login experience to simple input and scan operations.

Table 2. Performance of UAuth

	Wi-Fi access point	Connection	Total
Avg. Time(s)	1.1	4.2	11.4
[Min, Max](s)	[0.7, 1.8]	[2.5, 9.2]	[6.8, 19.3]

6 Conclusion

We have presented UAuth, a two-layer authentication framework. It enables users to enjoy the security benefits of using the physic device to authenticate: use the OTP generated by the device in the two-layer authentication. At the same

time, users receive the convenience of the SSO-like login: user can visit more than one SPID with their UID in the three-level account association. Specifically, in local authentication, users can choose a correct way to authenticate depending on their own device, The variety of personal devices and its flexibility also make our UAuth have a good performance in authentication.

We implemented and evaluated UAuth, and we get a conclusion that the UAuth has a relatively good performance in safety and user experience, it will enhance the current authentication technology on the web today.

Acknowledgments. This work has been supported by National Natural ScienceFoundation of China (Grant No. 61202476); the Strategic Priority Research Program of the Chinese Academy of Sciences (No. XDA06010701, XDA06040502).

References

1. SSL, GONE IN 30 SECONDS. https://media.blackhat.com/us-13/US-13-Prado-SSL-Gone-in-30-seconds-A-BREACH-beyond-CRIME-Slides.pdf
2. TOTP: Time-Based One-Time Password Algorithm. http://tools.ietf.org/html/rfc6238
3. The Domino Effect of the Password Leak at Gawker. http://voices.yahoo.com/the-domino-effectpassword-leak-gawker-10566853.html
4. Google 2-Step Verification. http://www.google.com/landing/2step/
5. FIDO Alliance. http://fidoalliance.org/
6. The YubiKey Manual. http://static.yubico.com/var/uploads/pdfs/YubiKey_Manual_2010-09-16.pdf
7. Millions of Adobe hack victims used horrible passwords. http://www.pcworld.com/article/2060825/123456:millions-of-adobe-hack-victims-used-horrible-passwords.html
8. The OAuth 2.0 Authorization Framework. http://tools.ietf.org/html/rfc6749
9. OpenID Authentication 2.0. http://openid.net/specs/openid-authentication-2_0.html
10. Kelley, P.G., Komanduri, S., Mazurek, M.L., Shay, R., Vidas, T., Bauer, L., Christin, N., Cranor, L.F., Lopez, J.: Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In: IEEE Symposium on Security and Privacy, pp. 523–537 (2012)
11. Bonneau, J., Herley, C., van Oorschot, P.C., Stajano, F.: The quest to replace passwords: a framework for comparative evaluation of web authentication schemes. Technical Report UCAM-CL-TR-817, University of Cambridge, Computer Laboratory (March 2012)
12. Cheswick, W.: Rethinking passwords. Commun. ACM **56**(2), 40–44 (2013)
13. Czeskis, A., Dietz, M., Kohno, T., Wallach, D., Balfanz, D.: Strengthening user authentication through opportunistic cryptographic identity assertions. In: Proceedings of the 2012 ACM CCS, pp. 404–414 (2012)
14. Ives, B., Walsh, K.R., Schneider, H.: The domino effect of password reuse. Commun. ACM **47**(4), 75–78 (2004)
15. Marforio, C., Karapanos, N., Soriente, C.: Smartphones as practical and secure location verification tokens for payments. In: NDSS 2014 (2014)

16. Wimberly, H., Liebrock, L.M.: Using fingerprint authentication to reduce system security: an empirical study. In: 2011 IEEE Symposium on Security and Privacy (SP), pp. 32–46 (2011)
17. Kontaxis, G., Athanasopoulos, E., Portokalidis, G., Keromytis, A.D.: SAuth: protecting user accounts from password database leaks. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security, pp. 187–198 (2013)