# Strengthening User Authentication through Opportunistic Cryptographic Identity Assertions

*Arvind Kumar Pandey, Assistant Professor*

*Department of Computer Science, Arka Jain University, Jamshedpur, Jharkhand, India*

***ABSTRACT:*** ***User authentication methods have reached a stalemate. The most common technique, the password, has a number of flaws, including the risk of accidental disclosure through phishing and cross-site password reuse. Second-factor authentication methods have the potential to improve security, but they are difficult to use and implement. Conventional second-factor methods, for example, alter the user authentication experience. Furthermore, although second-factor methods are more secure than passwords, they still fall short of providing adequate protection against (single-use) phishing assaults. We introduce PhoneAuth, a system that aims to give security guarantees on par with or better than standard two factor authentication systems while maintaining the same user experience as traditional passwords alone. The following important insights are used in our work. For starters, a user's personal device (such as a phone) may connect directly with the user's computer (and therefore the distant web server) without the need for the user to intervene. Second, a tiered approach to security may be implemented, with a web server, enacting various rules based on whether or not the users own device is present. Our server-side, Chromium web browser, and Android phone implementations of PhoneAuth are described and evaluated.***

***KEYWORDS:*** ***Authentication, Cryptography, Login, Password, Phishing.***

## 1. INTRODUCTION

Usernames and passwords are the most popular way for users to log onto websites. They are easy to set up on a server, and they let websites communicate with users in a number of ways. These basic methods have a number of drawbacks, not least of which is that many users reuse passwords across many websites, resulting in the breach of one website leading to the vulnerability of others. Users that wish to remember several passwords will find it difficult to do so at scale due to the cognitive load. Furthermore, consumers who are confronted with imposter websites or phishing attempts often hand up their passwords[1], [2].

It is no wonder, therefore, that a significant percentage of users have their online accounts accessed by unauthorized individuals on a daily basis, resulting in anything from small annoyances to financial loss to very serious risks to life and well-being.

We know as computer scientists that passwords are insecure, yet we continue to use those four decades after the discovery of public-key encryption and two decades after the birth of the internet. Bonneau recently published a study that explains why: none of the 35 password-replacement mechanisms studied is sufficiently usable or deployable in practice to be considered a serious alternative or augmentation to passwords, which is unfortunate because many of the proposals are arguably more "secure" than passwords. This includes public-key cryptography-based methods (such as CardSpace or TLS client certificates). Otherwise, public-key cryptography would be an ideal solution to the above-mentioned security issues with passwords: it would enable us to keep the authentication secret (a private key) hidden, and not communicate or store it with the parties to whom users authenticate (or their impostors) [3][4].

We set out to examine the usage of public-key cryptography for user authentication on the web from a new perspective. We are aware of the limitations of past efforts, as well as the inclusion of public-key-based methods in the Bonneau study's list of unsuccessful authentication approaches. However, we believe that if properly developed, public-key-based authentication methods may be useful. This paper's primary contribution is a design we call PhoneAuth, which has the following properties:

- It maintains the login user experience consistent: users input a username and password into a web page and do not do anything else.
- It adds a cryptographic second element to the password, making the login more secure against powerful attackers.
- This second factor is supplied opportunistically, that is, only when and if the conditions permit (compatible browser, presence of second factor device, etc.).

Though PhoneAuth has a number of operational requirements, we believe they are acceptable in light of current technological developments and do not prevent PhoneAuth from being used [5][6].

Client Certificates for TLS. TLS Client Authentication, which uses a TLS client certificate to authenticate a user, is an example of a password-less authentication method. Client certificates prevent users from being phished since they do not transmit the authentication secret (a private key) to the server (since there is no password to be entered). TLS Client Authentication, on the other hand, has a number of flaws that have hampered its broad adoption on the Internet:

- *Ineffective user experience:*

Users must accept or reject the certificate's usage before they can interact with the website since it is required during the TLS handshake. This results in a user experience in which the browser requests that the user choose an identity (typically in the form of a "distinguished name" or a "X.509 certificate") without providing any context on where or how that identity will be utilized [7][8].

- *Confidentiality:*

Any website on the internet may request TLS client authentication using a certificate that has been installed on a user's computer. The user now has two options: either do not log in at all or log into several sites with the same certificate. It is possible to log into several websites with the same certificate, however this opens the door for cooperating websites to monitor a user's surfing patterns by examining the certificate used to login. Another option is for the user to generate a separate certificate for each site to which he logs in, however this results in an even poorer user experience: When a user tries to login to a site that requires TLS client authentication, the user is now presented with an ever-growing list of certificates [9][10].

- *Convenience:*

Certificates should ideally be linked to a private key that cannot be retrieved from the platform. As a result, they are difficult to transfer from one device to another. As a result, every TLS Client Authentication solution must also address and resolve the issue of user credential portability. Reobtaining certificates for various devices (which may be a complex operation in and of itself), extracting private keys and transferring them from one device to another (against basic security practices), or cross-certifying certificates from separate devices are all possible options. We believe that when given the option, users would choose less secure (but more accessible) password methods due to the difficulty of acquiring TLS client certificates and the unfamiliar user interface.

CARDSPACE. CardSpace, a Microsoft authentication system with many capabilities important to our work, was created. Its most notable feature is that it replaced passwords with a public-key protocol.

Virtual identity "cards" would be used to manage users' digital identities. Users would be provided with a UI when visiting a website that supported CardSpace, allowing them to select which card, and therefore which identity, to use with the site. CardSpace verified users behind the scenes by generating cryptographic attestations of the user's identity that could be sent to the verifying website. The benefit of this method was that the authentication secret (usually a private key) was not revealed to the verifying site. Furthermore, users could not be phished since they signed in by choosing a "card" rather than entering a password[11], [12].

CardSpace, however, was not well received and was ultimately phased out. CardSpace's effort to offer a large number of additional capabilities, we think, exacerbated its overall complexity and led to its death by overcomplicating the user interface, interaction, and development models. We try to learn from CardSpace's failure and have carefully built our system to differ as little as possible from the user experience (and development burden) that users (and developers) are used to.

*1.1 Login with a Federated Account:*

Users can have just one account at an identity provider to which they may directly authenticate (or a restricted number of such accounts) under the basic concept underlying federated login. Not all other websites (known as relying parties) need users to authenticate themselves; instead, they depend on identity claims from the identity provider.

OpenID, Facebook Connect, OpenID Connect, and Security Assertion Markup Language (SAML), in which the identity provider sends the identity assertion to the user's browser, which then sends it to the relying party after the user logs into the identity provider and approves the issuance of the identity assertion,

exemplify this approach. BrowserID is a little different in that the browser rather than the identity provider issue the identity assertion (although the identity assertion does contain a certificate from the identity provider that the user must acquire ahead of time).

Federated login promises to reduce the number of passwords that users must remember. Of course, this promise can only be fulfilled if the majority of websites on the internet are dependent on for just a few identity suppliers. However, the value proposition for a website to become a relying party to an identity provider is not always clear: what if the identity provider is unsafe, goes out of business, or fails to properly ban accounts that have been hacked? Users may also be uncomfortable with the identity provider knowing which sites they visit or with relying parties learning so much about their identities from the identity provider; they may prefer to create a new identity at a relying party site rather than "reusing" their identity from the identity provider[13], [14].

Finally, federated login is just a solution to a previously unresolved issue (securely authenticating to dependent parties) (that of securely authenticating to an identity provider). It does not, by itself, solve the problem of customers being phished for their identity provider password, or sharing that password with sites that refuse to operate with that provider.

*1.2 Two-Factor Authentication as Usual:*

A number of two-factor authentication methods are used by certain websites. In certain instances, the user must also input a code obtained from a device they carry ("something that they have") in addition to the password ("something that they know"). In certain instances (for example, smart cards/tokens), users must physically connect the device to the PC they are authenticating on.

*1.3 Assumptions and Goals:*

We take a new look at robust user authentication on the web, based on the lessons learned from past work.

*The following are the objectives we have set for our work:*

- The authentication procedure must include some kind of public-key cryptography. Not only does this keep the authentication secret (the private key) safe on the client device, but it also keeps the secret hidden from the user, making it impossible to steal via phishing.
- Above the transport layer, the user's identity must be verified and confirmed. Otherwise, as we saw with TLS client authentication, consumers' inability to understand the context in which they are authenticating leads to a bad user experience and privacy issues.
- The process of logging onto a website should be consistent: users enter their username and password into a web page (rather than their browser chrome or another trusted device), and then they are signed in. This not only aids user learnability, but it also aids deploy ability: websites do not need to redesign their login processes, and users who have the required client software may be progressively "on boarded" into the new authentication method.
- The design should operate in both a world with a small number of identity providers and a world where each website maintains its own authentication service.
- Users require a fall back mechanism that allows them to log in using only something "that they know" if the public-key mechanism fails, or if they have a legitimate need to hand over their credential to a third party.

Model of Threatening. Another aim of our research is to safeguard users against a powerful attacker. We assume the following threat model in particular: We make it possible for attackers to acquire a user's password either via phishing or by exploiting weaker sites (for which the user has reused a password).

On the connection between the user and the server to which the user is authenticating, we presume the attacker may conduct a man-in-the-middle assault. This attack requires the attacker possesses a valid TLS certificate for the site to which the user is authenticating, enabling him to conduct TLS man-in-the-middle assaults. We even make it possible for an attacker to acquire the proper certificate for the victim site (probably by stealing the private key). This feature is very strong, and a TLS man-in-the-middle attack may even make browser certificate pinning vulnerable. Though no instances of such assaults have surfaced in the wild, security experts think they are feasible[15].

## 2. DISCUSSION

Deploy ability And Operating Requirements PhoneAuth, as the astute reader has observed, has a number of operational criteria that must be fulfilled before the system can be implemented. First, the functionality of our browser extension should be transferred to the real browser. To make this happen, we contacted the Chromium browser team and worked with the Firefox team. Second, developers must be able to easily implement this authentication method on their websites.

Our service-oriented design of the server-side PhoneAuth capability makes this simple, but a non-trivial deployment of PhoneAuth is still in the works. Finally, users must test and approve the system. We think that the primary reason comparable systems have failed is that none of them has been able to provide opportunistic strong user authentication without compromising the user experience - a characteristic that our system does. In the near future, we want to conduct field-testing of the system. Finally, Bluetooth should be a standard feature on most phones and computers. We discovered that Bluetooth is included in the majority of new gadgets. We discovered that all Apple computers, nearly all laptops (HP and Dell), and approximately half of desktop PCs (HP and Dell) had built-in Bluetooth. Given these figures, we think the Bluetooth universality target is achievable.

## 3. CONCLUSION

We presented PhoneAuth, a novel web-based user authentication mechanism, in this article. PhoneAuth has the same usability advantages as traditional passwords: users may approach an Internet kiosk, browse to a web page of interest, and log in by simply typing their user name and password. At the same time, PhoneAuth has all of the advantages of traditional two-factor authentication solutions, plus more. PhoneAuth, in particular, saves cryptographic credentials on the user's phone. The phone will testify to the user's identification through Bluetooth talks with the computer's browser if it is present when the user registers into a site; this occurs even if the user has never interacted with that specific machine before. We also explored a tiered approach to security, in which a web server may apply various rules based on whether or not the user's phone is there.

The author dubbed this “opportunistic identity claims”. Because opportunistic identity assertions enable the server to handle logins differently depending on how the user was authenticated, it may offer tiered access or limit hazardous functionality (e.g., mass e-mail deletion). While opportunistic identity assertions may not be accessible to all users at all times (for example, due to a lack of Bluetooth compatibility), there are still benefits to offering them.

Similarly, an opponent who can make it seem as though Alice's phone is "not there" weakens Alice's login and stops her from accessing hazardous features.

PhoneAuth was built and assessed, and our conclusion is that it is a feasible option for enhancing the security of online authentication today.

**REFERENCES**


[1] A. Czeskis, M. Dietz, and D. Wallach, “Strengthening User Authentication through Opportunistic Cryptographic Identity Assertions Categories and Subject Descriptors,” *Proc. 19th ACM Conf. Comput. Commun. Secur.*, 2012.

[2] H. Halpin, “The W3C web cryptography API: Motivation and overview,” 2014. doi: 10.1145/2567948.2579224.

[3] A. Czeskis, M. Dietz, T. Kohno, D. Wallach, and D. Balfanz, “Strengthening user authentication through opportunistic cryptographic identity assertions,” 2012. doi: 10.1145/2382196.2382240.

[4] C. Garman, M. Green, and I. Miers, “Decentralized Anonymous Credentials,” 2014. doi: 10.14722/ndss.2014.23253.

[5] D. Crocker, “RFC 6376 - DomainKeys Identified Mail (DKIM) Signatures,” *Req. Comments*, 2011.

[6] M. T. Goodrich, R. Tamassia, and D. Yao, “Notarized federated identity management for web services,” 2006. doi: 10.1007/11805588_10.

[7] D. Fett, R. Küsters, and G. Schmitz, “Analyzing the browserID SSO system with primary identity providers using an expressive model of the web,” 2015. doi: 10.1007/978-3-319-24174-6_3.

[8] M. Bobaru, M. Borges, M. d’Amorim, and C. S. Păsăreanu, *NASA formal methods : third international symposium, NFM 2011, Pasadena, CA, USA, April 18-20, 2011 : proceedings*. 2011.

[9] S. Committee, *IEEE Standard for Software Verification and Validation IEEE Standard for Software Verification and Validation*. 1998.

[10] S. D. Verifier and A. H. Drive, “Simulink ® Verification and Validation ™ Reference,” *ReVision*, 2015.

[11] M. A. Sasse and M. Smith, “The Security-Usability Tradeoff Myth [Guest editors’ introduction],” *IEEE Secur. Priv.*, 2016, doi: 10.1109/MSP.2016.102.

[12] Z. Lina, “Design and Implementation of KSP on the Next Generation Cryptography API,” *Phys. Procedia*, 2012, doi:

10.1016/j.phpro.2012.05.264.

[13] K. Cairns, H. Halpin, and G. Steel, "Security analysis of the W3C web cryptography API," 2016. doi: 10.1007/978-3-319-49100-4_5.

[14] J. Toldinas, R. Damasevicius, A. Venckauskas, T. Blazauskas, and J. Ceponis, "Energy consumption of cryptographic algorithms in mobile devices," *Elektron. ir Elektrotechnika*, 2014, doi: 10.5755/j01.eee.20.5.7118.

[15] J. Knudsen, "Java Cryptography," *EDPACS*, 1999, doi: 10.1201/1079/43250.27.4.19991001/30275.5.