Yi Mu
Willy Susilo
Jennifer Seberry (Eds.)

LNCS 5107

Information Security and Privacy

13th Australasian Conference, ACISP 2008
Wollongong, Australia, July 2008
Proceedings

Lecture Notes in Computer Science 5107

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Yi Mu Willy Susilo Jennifer Seberry (Eds.)

Information Security and Privacy

13th Australasian Conference, ACISP 2008
Wollongong, Australia, July 7-9, 2008
Proceedings

Springer

Volume Editors

Yi Mu
Willy Susilo
Jennifer Seberry
University of Wollongong
School of Computer Science and Software Engineering
Northfields Avenue, Wollongong, NSW 2522, Australia
E-mail: {ymu, wsusilo, jennie}@uow.edu.au

Library of Congress Control Number: Applied for

CR Subject Classification (1998): E.3, K.6.5, D.4.6, C.2, E.4, F.2.1, K.4.1

LNCS Sublibrary: SL 4 – Security and Cryptology

ISSN 0302-9743
ISBN-10 3-540-69971-6 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-69971-2 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12322725 06/3180 5 4 3 2 1 0

Preface

The 13th Australasian Conference on Information Security and Privacy (ACISP 2008) was held at Wollongong, Australia, during July 7–9, 2008. The conference was sponsored by the Centre for Computer and Information Security of the University of Wollongong and the Research Network for a Secure Australia. The submission and review process was run using the iChair software, written by Thomas Baigneres and Matthieu Finiasz from EPFL, LASEC, Switzerland. We would like to thank them for letting us use their iChair software.

The conference received 111 submissions, out of which the Program Committee selected 33 papers for presentation at the conference after a rigorous review process. These papers are included in the proceedings. The accepted papers cover a range of topics in information security, including authentication, key management, public key cryptography, privacy, anonymity, secure communication, ciphers, network security, elliptic curves, hash functions, and database security. The conference proceedings contain revised versions of the selected papers. Since some of them were not checked again for correctness before publication, the authors bear full responsibility for the contents of their papers. We would like to thank the authors of all papers for submitting their papers to the conference.

In addition to the contributed papers, the program comprised three invited talks. The invited speakers were Xavier Boyen (Voltage, USA), Josef Pieprzyk (Macquarie University, Australia) and Nigel Phair (Australian High Tech Crime Centre). We would like to express our thanks to them.

As in previous years, we selected a "best student paper." To be eligible for selection, a paper has to be co-authored by a postgraduate student, whose contribution was more than 50%. The winner was Risto Hakala from Helsinki University of Technology, Finland, for the paper "Linear Distinguishing Attack on Shannon."

We would like to thank all the people who helped with the conference program and organization. In particular, we heartily thank the Program Committee and the sub-reviewers listed on the following pages for the effort and time they contributed to the review process. We would like to express our thanks to Springer for continuing to support the ACISP conference and for help in the conference proceedings production.

Finally, we would like to thank the Organizing Committee for their excellent contribution to the conference.

July 2008

Yi Mu
Willy Susilo
Jennifer Seberry

The 13th Australasian Conference on Information Security and Privacy (ACISP 2008)

Sponsored by

Centre for Computer and Information Security Research,
University of Wollongong, Australia

Research Network for a Secure Australia

General Chair

Jennifer Seberry	University of Wollongong, Australia

Program Chairs

Yi Mu	University of Wollongong, Australia
Willy Susilo	University of Wollongong, Australia

Program Committee

Michel Abdalla	ENS, Paris, France
Masayuki Abe	NTT, Japan
Colin Boyd	QUT, Australia
Feng Bao	Institute for Infocomm Research, Singapore
Lynn Batten	Deakin University, Australia
Ed Dawson	QUT, Australia
Dieter Gollmann	TU Hamburg, Germany
Aggelos Kiayias	University of Connecticut, USA
Kwangjo Kim	ICU, Korea
Tanja Lange	Technische Universiteit Eindhoven, Netherlands
Pil Joong Lee	Pohang University of Science and Technology, Korea
Benoit Libert	UCL, Belgium
Javier Lopez	University of Malaga, Spain
Chris Mitchell	RHUL, UK
Yi Mu	University of Wollongong, Australia
Kaisa Nyberg	Helsinki University of Technology, Finland
Eiji Okamoto	Tsukuba University, Japan
Josef Pieprzyk	Macquarie University, Australia
Sihan Qing	Chinese Academy of Scineces, China
Jean-Jacques Quisquater	UCL, Belgium
Rei Safavi-Naini	University of Calgary, Canada

Jennifer Seberry	University of Wollongong, Australia
Ron Steinfeld	Macquarie University, Australia
Douglas Stinson	University of Waterloo, Canada
Willy Susilo	University of Wollongong, Australia
C. Pandu Rangan	Indian Institute of Technology, India
Tsuyoshi Takagi	Future University, Japan
Vijay Varadharajan	Macquarie University, Australia
Sabrina De Capitani di Vimercati	University of Milan, Italy
Huaxiong Wang	Nanyang Technological University, Singapore
Duncan S. Wong	City University of Hong Kong, China
Fangguo Zhang	Sun Yat-Sen University, China
Ning Zhang	University of Manchester, UK
Jianying Zhou	Institute for Infocomm Research, Singapore

Organizing Committee

Man Ho Au	University of Wollongong, Australia
Xinyi Huang	University of Wollongong, Australia
Shams Ud Din Qazi	University of Wollongong, Australia
Mohammad Reza Reyhanitabar	University of Wollongong, Australia
Siamak Fayyaz Shahandashti	University of Wollongong, Australia
Pairat Thorncharoensri	University of Wollongong, Australia
Wei Wu	University of Wollongong, Australia
Tsz Hon Yuen	University of Wollongong, Australia

External Referees

Isaac Agudo	Reza Rezaeian Farashahi	Jang Seong Kim
Hadi Ahmadi	Gerardo Fernandez	Sun Young Kim
K. Ambika	Carmen Fernandez-Gago	Young Mok Kim
Venkat Balakrishnan	Georg Fuchsbauer	Varad Kirthane
Daniel J. Bernstein	Juan Garay	Hoi Le
Jean-Luc Beuchat	Praveen Gauravaram	Fagen Li
Peter Birkner	Juan Gonzalez	Jin Li
Billy Bob Brumley	Satoshi Hada	Vo Duc Liem
S. Chandrasekar	Risto Hakala	Peter van Liesdonk
Joo Yeon Cho	Kevin Henry	Joseph K. Liu
Sherman Chow	Matt Henricksen	Jiqiang Lu
Baudoin Collard	Jason Hinek	Mark Manulis
Alex Dent	Michael Hitchens	Krystian Matusiewicz
Dang Nguyen Duc	Qiong Huang	Antonina Mitrofanova
Sung Wook Eom	Shaoquan Jiang	Cameron McDonald

Pablo Najera
Miyako Ohkubo
Vijayakrishnan P.
Arpita Patra
Angela Piper
M.R. Reyhanitabar
Rodrigo Roman
Chun Ruan
Palash Sarkar
Sharmila Devi Selvi
Jae Woo Seo
Siamak Shahandashti
Hongsong Shi
Jong Hoon Shin
Masaaki Shirase
Igor Shparlinski
Leonie Simpson
Michal Sramka
Jerry Sui
Christophe Tartary
Ronghua Tian
Tomas Toft
Mohammed A.A. Tuhin
Udaya Kiran Tupakula
Damien Vergnaud
José Villegas
Jose L. Vivas
Yongge Wang
Baodian Wei
Kenneth Wong
Jiang Wu
Guomin Yang
Yanjiang Yang
Yeon-Hyeong Yang
Chan Yeob Yeun
Hongbo Yu
Yu Yu
Janson Zhang
Chang-An Zhao
Weiliang Zhao
Hong-Sheng Zhou
Huafei Zhu
Sebastien Zimmer

Table of Contents

New Paradigms for Password Security

(Abstract from the Keynote Lecture)

Xavier Boyen

Voltage Inc.
xb@boyen.org

For the past several decades, cryptographers have consistently provided us with stronger and more capable primitives and protocols that have found many applications in security systems in everyday life. One of the central tenets of cryptographic design is that, whereas a system's architecture ought to be public and open to scrutiny, the keys on which it depends — long, utterly random, unique strings of bits — will be perfectly preserved by their owner, and yet nominally inaccessible to foes.

This security model works well as long as one can assume the existence of an inviolate physical location or storage device to safeguard those keys. In client-server scenarios, the mere delocalization of the participants suffices to enforce a proper boundary without any further precaution. In proxy settings, one may call upon tamper-resistant "smart cards" or hardware security modules to isolate the keys adequately from most opponents.

Things break down when one can no longer assume that an external storage medium is available to store our keys, and that the only option is to remember them in our minds. The problem, of course, is a cognitive one: the human brain is ill-equipped to remember hundreds of random bits of key material for the long term without making any mistake. The secrets that our brain is keen on remembering are those of our own choosing, which for all their apparent randomness and unpredictability can certainly not be mistaken nor substituted for genuine cryptographic keys. Security from purely mental secrets requires us at the very least to compromise on key strength — this encompassing both entropy and uniformity —, and seek the best reachable security goals based not on ideal random keys but on passwords of sub-cryptographic quality.

Plain textual passwords and passphrases — or passtexts — have always been the preferred form of human-memorable secret, having the benefit of medium-independence which entails compatibility with virtually any conceivable user interface. More exotic mental secrets — passthoughts — may be based on visual or auditory recognition feedback; these are equivalent to passwords from a cryptographic perspective, but the specialized input device they require make them less practical. Secrets whose expression requires body action such as speech or ocular movements — passmoves — may also be envisaged given the proper measurement apparatus, with the proviso that the unavoidable measurement noise in the analog signal will have to be dealt with; we merely mention that errors on the post-quantization signal may be correctable using information-theoretic

Y. Mu, W. Susilo, and J. Seberry (Eds.): ACISP 2008, LNCS 5107, pp. 1–5, 2008.

cryptographic tools such as reusable and robust fuzzy extractors [1] without leaking excessive information about the secret.

Regardless of the shape of form of the secret, an important criterion for its human memorability is that its selection ultimately be left to the human who will have to remember it. Machines can assist in password selection, but should not make the final choice. Because of this, it is a near-certainty that the selected secret will not make a suitable cryptographic key, nor will it be possible to derive one from it due to lack of entropy. Hence, specialized primitives and protocols are needed that explicitly take into account those inherent weaknesses, and seek to achieve the best possible security under the circumstances.

Although password-based primitives and protocols have seen much foundational and implementational improvements during the last two decades, the general philosophy of password-based offline key derivation and online key exchange has remained essentially what it was in the early nineties. In particular, most current approaches could better handle real-life situations where the password are too weak for comfort and/or are recycled in part or in whole with multiple correspondents.

The purpose of this exposé is thus to investigate what security may indeed be attained from human-memorable passwords as they do appear in the real world — including the weak, skewed, reused, and exceedingly long-lived ones. The focus on literal passwords stems from tradition as much as convenience.

1 Halting Puzzles against Brute-Force Dictionary Attacks

Stand-alone — offline — uses of passwords mainly concern encryption and key derivation applications. The prime example of this is to encrypt the contents of a laptop so that only its owner can access it. Local authentication and device unlocking uses may also be treated as special cases of password-based encryption. At the core of these systems, one finds a Key Derivation Function (KDF), which is a one-way function taking a password and an optional public random salt as input, and producing a reproducible cryptographic key as output.

Offline applications such as those are tremendously difficult to secure with a weak password. The threat model here is the loss of the entire ciphertext and all associated hardware to an attacker, where only the password is being held back. Therefore, any opponent that simply tries out all passwords in an offline dictionary attack, e.g., by decreasing order of estimated likelihood, will eventually stumble upon the correct one and defeat the encryption. The only defense against such a threat is to slow down or deter the attacker by making the attack more daunting. There are two ways to do this: by picking an unlikely password to increase the expected number of guesses, and by making each guess more computationally demanding to verify.

One cannot really play with the choice of the password, short of encouraging the user to select a long and difficult one. Making the guesses hard to verify is possible, but only within limits, as it has the side effect of increasing the user's legitimate access latency in the same proportion. For this reason, KDFs

are purposely designed to be somewhat expensive to compute, although most implementations tend to be very conservative with the amount of slowdown that they are willing to impose on users, and rarely offer the user any choice in the matter. The general trend is thus to use KDFs with a slowdown parameter (often a hash iteration count) that is conservatively chosen, once-and-for-all frozen, and publicly disclosed as part of the KDF specification or implementation. Some implementations support in-the-field adjustment of the KDF iteration count, but this parameter always remains public.

This has been and continues to be the ubiquitous way in which passwords are used for local key derivation.

In departure from this trend, we recently introduced, in [2], the notion of Halting Key Derivation Function (HKDF), which explicitly lets the user choose an arbitrary hardness parameter and emded it into the function in a cryptographically secret manner. The idea is to encourage the user to make the HKDF as difficult to compute as the delay he or she is willing to tolerate when seeking access, but conceal the value of the chosen parameter from public view, and yet not require the user to remember such value — or for that matter anything else besides the password.

The crucial element is that, on the correct password, the HKDF function will recognize that it succeeded and halt spontaneously after the intended computational delay; but on an incorrect password, it will continue indefinitely without giving any feedback until manually interrupted. The only indication given to the user that a password is incorrect will be the feeling that the key derivation is taking longer than it should. The user will naturally react by restarting the process and reentering the password more carefully without much of an afterthought. To an attacker, by contrast, this lack of feedback will disproportionately complicate the task of mounting an offline dictionary attack. The result is an effective security increase equivalent to two extra bits of password entropy, at virtually no cost to the legitimate user.

The total security gains provided by HKDFs are actually much greater than just two bits, due to a combination of factors. The main contributing factor is that legacy KDFs tend to be parameterized very conservatively, leading to exceedingly short delays ($\sim 1ms$) that are only getting shorter as computers are getting faster, raising obsolescence concerns. By contrast, HKDFs are programmed on a case-by-case basis, on the basis on actual clock times, with respect to the current state of computer performance. Even at the shorter end of HKDF delays, the "blink of an eye" ($\sim 1s$), the jump is already substantial. It will also keep up with technological progress, since a one-second-delay in ten years will entail a greater number of elementary operations than a one-second-delay today.

As discussed in [2], one should expect a fairly wide spectrum of user-selected HKDF delays to find their way in practical applications. Short delays are appropriate for frequently used day-to-day passwords with a short lifespan. Longer delays ($\sim 1m$ and more) could be used to protect longer-term backup passwords, which may need to be simpler to be memorable over a longer period. The longest

delays ($\sim 1h$ and more) would be reserved for last-resort disaster-recovery passwords, never intended to be used, but that must be available and remembered if ever needed even after many years have lapsed. Such passwords would likely have to be very weak to be reliably memorable over such long periods, hence the need for very long HKDF delays to protect them from offline dictionary attacks. Notably, the same plaintext can be encrypted under different passwords using different delays, seamlessly, without any loss of security or usability.

2 Hardened Protocols toward Universal Authentication

Client-server — online — uses of passwords are primarily geared toward authentication and key exchange. Both parties share a password, and, based on it, try to establish a private authenticated channel over open communication lines. The constraints on online passwords are fairly different than in the offline case, as here the threat model typically assumes that the communicating parties are honest and try to prevent eavesdropping and impersonation by a malicious outsider (who controls the underlying communication channel).

Password-Authenticated Key Exchange (PAKE) is indeed a success story of cryptographic protocol design, as there are many protocols realizing the theoretically optimal security requirement that the only feasible attack vector be for the adversary to make online password guesses, one guess at a time, interactively with one of the honest parties — who can then detect the attack and throttle it by refusing to communicate. Secure online authentication can thus be achieved using much weaker passwords than would be thinkable in the offline case.

Extensions of this notion have been proposed for the case where the server itself may be viewed as an adversary, as is the case when the client wishes to reuse the same password with other servers. Asymmetric Password-Authenticated Key Exchange (APAKE) deals with this notion by requiring the password only on the client side; the server is instead entrusted with a derived secret that can be used to reach mutual authentication with the client, but not impersonate it to another server (in particular the password should be hard to recover from this). APAKE protocols are for this reason more desirable in practical use than PAKE, in light of the well-documented propensity of internet users to recycle the same few passwords with a broad variety of vendors. However, one concern remains, which is how difficult it actually is for a malicious server to recover its clients' passwords from the derived secrets.

The concern is that the derived secrets are typically obtained by applying a one-way function to the password w, be it a cryptographic hash $h(w)$ or a modular exponentiation g^w. Functions like these are usually very fast to compute, so even though they technically may be one way, they might be relatively easy to invert in an offline dictionary attack if the user password is not already very strong. Also, without an extra randomization step, a server can attack all of its clients' passwords for the price of one.

Since typical real-life users are probably going to continue reusing the same weak passwords with many servers regardless of whether this is considered a safe

thing to do, it would be desirable to design a protocol that attempts to preserve the best possible form of online and offline password security, even under reuse of a weak password across multiple servers. The benefit from such a notion would be safe universal authentication on the internet using a single easy-to-remember password (for each user).

Ideally, one wish to combine the security of (A)PAKE against outside online attackers, with the security of HKDF against malicious servers.

To this end, we are proposing, in [3], the notion of Hardened Password-Authenticated Key Exchange (HPAKE), which offers the same security guarantees as regular asymmetric key exchange, and in addition allows the user to specify an arbitrarily expensive one-way function for the mapping from client password to server secret. This makes even relatively weak passwords infeasible to recover by malicious servers, thereby enabling the reuse of such passwords with arbitrarily many servers.

There are several difficulties with this. The first is a systemic one: the burden of computing this arbitrarily expensive one-way function should befall the client who selected it, and not the server which for scalability reasons must be able to process many authentication requests with minimal effort. The second issue is a technical one: since the one-way function is to be computed on the client side, the client must obtain the necessary inputs from the server prior to authentication. This creates a paradox, since the success of such transfer must depend on the client's knowledge of the password, but at the same time not reveal to either the client or the server whether the transfer succeeded, lets it open an avenue for offline attack to outsiders or to the server itself.

We shall discuss how these difficulties can be overcome, and how the HPAKE framework from [3] provides a plausible and practical answer to the problem of universal authentication from a single password.

3 Conclusion

The password schemes presented in this lecture have in common that they seek to provide the best possible security for the password holder, in the offline and online setting, regardless of how careless his or her use of that password may be. The only safety rule that should never be failed, is that one's password should only be seized on a local trusted HKDF or HPAKE entry device, and not shared with other less secure protocols.

References

1. Boyen, X.: Robust and Reusable Fuzzy Extractors. In: Tuyls, P., Skoric, B., Kevenaar, T. (eds.) Security with Noisy Data, Springer, Heidelberg (2007)
2. Boyen, X.: Halting Password Puzzles – hard-to-break encryption from human-memorable keys. In: SECURITY 2007, The USENIX Association (2007)
3. Boyen, X.: Hardened Password Authentication – mulitple mobile credentials from a single short secret. Manuscript (2008)

Enforcing User-Aware Browser-Based Mutual Authentication with Strong Locked Same Origin Policy

Sebastian Gajek[1], Mark Manulis[2], and Jörg Schwenk[1]

[1] Horst Görtz Institute for IT-Security, Germany
{sebastian.gajek,joerg.schwenk}@nds.rub.de
[2] UCL Crypto Group, Belgium
mark.manulis@uclouvain.be

Abstract. The standard solution for mutual authentication between human users and servers on the Internet is to execute a TLS handshake during which the server authenticates using a X.509 certificate followed by the authentication of the user either with own password or with some cookie stored within the user's browser. Unfortunately, this solution is susceptible to various impersonation attacks such as phishing as it turned out that average Internet users are unable to authenticate servers based on their certificates.

In this paper we address security of *cookie-based authentication* using the concept of *strong locked same origin* policy for browsers introduced at ACM CCS'07. We describe a cookie-based authentication protocol between human users and TLS-servers and prove its security in the extended formal model for *browser-based mutual authentication* introduced at ACM ASIACCS'08. It turns out that the small modification of the browser's security policy is sufficient to achieve provably secure cookie-based authentication protocols considering the ability of users to recognize images, video, or audio sequences.

1 Introduction

Motivation. The browser plays an indispensable function as the user's interface to access the rich world of Web based services. In order to serve the purpose of an universal client, commodity browsers have been augmented with numerous functionalities. Examples include extensions of the HTTP header to control caching and transport cookies, or the HTML markup language to enable high-level scripting and supply technologies like AJAX, AFLEX or SOAP. By contrast, much effort to amend the browser security model and provide new cryptographic services has not been spent. Since its adaption more than a decade ago [9], the Transport Layer Security (TLS) framework is the main pillar of browser-based protocols to provide Web applications with a security layer. After the protocol framework has been peer-reviewed without finding any significant vulnerabilities [25,28,24,22], it has been believed to be the holy grail for secure Web authentication. However, recent studies point out that average-skilled Internet users understand neither TLS nor its indication in commodity Web browsers at all [7,27]. Users tend to ignore browser's warnings and prefer to identify Web sites on the basis of non-technical indicators (e.g., brands, logos). This attitude provides a wrong sense of security. An adversary may fake the site and disclose the user's password (*phishing attack*).

Y. Mu, W. Susilo, and J. Seberry (Eds.): ACISP 2008, LNCS 5107, pp. 6–20, 2008.

The advent of these large-scale fraud attacks has led to several modifications in the visualization of TLS. Unfortunately, it seems to turn out that the changes do not meet their high expectations either [16].

Another line of research addresses the design of authentication protocols that provide user-awareness. The essence of user-aware protocols is to relax the assumptions on user behavior and provide secure authentication ceremonies. Recently, the authors of this paper introduced a formal security model for *browser-based mutual authentication (BBMA)* between a human user and a server where the browser is modeled as the mediator of the communication [11]. Their model is an extension of the classical model for authentication from [3] towards consideration of user-awareness within the authentication protocols on the Internet whereby user-awareness is modeled via *human perceptible authenticators (HPAs)* that are implied by natural human senses, such as recognition of images, videos, and audio sequences. In addition to the model, [11] describes a protocol called `BBMA` (based on the ideas of the PassMark Security Inc.'s Two-Factor-Two-Way Authentication™) which can be implemented within the standard specification of the TLS protocol. In this protocol the human user authenticates via password which is typed into an HTML form only after the successful recognition of some expected HPA sent by the server. In order to protect the disclosure of this HPA to unauthorized parties, the TLS protocol uses client (possibly self-generated) certificates which serve as a cryptographic identifier for the corresponding HPA.

Extending this line of research, we deal with user-awareness in cookie-based authentication protocols. These protocols execute a server-only authenticated TLS session, where the user authenticates through a cookie that has been previously set by the server and stored in the browser's cache. The technique has the advantage that the user is refrained from retyping the password. Further, the cookie is taken from a sufficiently large random distribution. There is no need to expect a "security defect" due to the use of low-entropy passwords. These simplifications of user authentication have led to a wide adaption of cookie-based authenticated channels in browser-based protocols and there are many protocols that build upon this technique. Unfortunately, they have been shown to be vulnerable when taking the mature browser security model into account (see Section 2 for more discussions). The crux is that the browser decides on the basis of the server's domain name whether to reveal the cookie. The adversary is feasible to steal the cookie by spoofing the domain names and there are many attacks allowing the adversary to do this (e.g., dynamic pharming, DNS rebinding [15,19]).

To protect against the growing presence of these threats, Karlof et. al. propose refinements of the browser's cookie disclosure policy [23,19]. Their contribution is to augment the browser with some additional functionality which uses cryptographic mechanisms to enforce restricted access policies without relying on DNS, dubbed the *strong locked same origin (SLSO)* policy. In the context of cookie-based authentication protocols over the TLS channel, the SLSO policy enforcement means that the browser sends a cookie to the server only after the server proves the possession of a valid cryptographic identifier, namely the server's public key, i.e., the server proves the knowledge of the corresponding private key.

Contributions. In this paper we extend our model from [11] towards cookie-based authentication and consideration of the browser's SLSO policy. Using the extended model we analyze the security of the cookie-based version of BBMA from [11] re-engineered under the SLSO policy. We call the modified protocol BBMA-SLSO. It turns out that some minor changes of the browser security model to enforce the SLSO policy—which is a straightforward task compared to the large scale deployment of, say secure domain name resolution protocols (DNSSEC)—turns an insecure protocol into a provably secure one. Additionally, the use of SLSO policy allows us to eliminate the costly use of the client certificates, which are essential to prove security of BBMA. In addition to the formal security definition, BBMA-SLSO has additional advantages over previous cookie-based authentication protocols. The advantages include

1. BBMA-SLSO is *user-aware*. In order to authenticate, the server sends a HPA, which serves (i) as non-cryptographic identifier for the user to validate the server as in the physical world where identities are provided in an easily recognizable fashion and (ii) as fail-stop mechanism to hamper that she discloses private information on a faked site.
2. BBMA-SLSO fits into the standard TLS specification. There is no need to modify commodity server implementations. In fact, the necessary augmentations address browsers, more precisely their functionality to access cookies corresponding to the SLSO policy. See [23] for more details.

We remark that the enforcement of the SLSO policy is ineligible to protect against *cross-site scripting (XSS)* attacks. The anatomy of XSS attacks is to exploit weaknesses of application servers and inject malicious scripts into the communication that enable the adversary to invoke certain browser functionalities. Since the scripts are in the same security context the SLSO policy does not help. Consequently, the adversary would have access to the user's password typing, the cookie and HPA in BBMA-SLSO. Though we treat XSS attacks as (server) corruptions in our model and exclude them in the analysis, a work-around to make BBMA-SLSO resistant against the attacks is to completely isolate the named security critical information and prevent that they are accessible from the surrounding (potentially malicious) scripts. Such a feature is already available in the Internet Explorer for cookies [21]. The approach has to be extended for passwords and HPAs. Since the implementation of the SLSO policy requires the modification of the current browser's security policy anyway, we suggest to enrich this policy with the private/public tagging of elements. An element such as a password field tagged with a private value shall signal the browser that any script is prevented from access, regardless of its security context. See [10] for more details.

Organization. The remainder sections are structured as follows. We review related work in Section 2. In Section 3, we describe the formal security model for cookie-based BBMA protocols under consideration of the SLSO policy. In Section 4 we specify a concrete protocol called BBMA-SLSO using the high level description of the TLS handshake in the key transport mode and prove that it is user-aware and satisfies the defined authentication requirement. Finally, we conclude the paper in Section 5.

2 Related Work

So far, few browser-based protocols have been subject to rigorous security analysis: Kormann and Rubin [20] show that Microsoft's .NET passport, a Web-based realization of the Kerberos protocol for single sign on, is susceptible to attacks where the adversary steals the ticket granting ticket cookie. Soghoian and Jakobsson [30] investigate the SiteKey-protocol that displays a previously negotiated image in addition to password forms in order to signal that the user is connected to the benign server. The authors show the feasibility of stealing the shared secret that is stored in a cookie. Groß [12] analyzes SAML, an alternative single sign on protocol, and shows that the protocol is vulnerable to adaptive attacks where the adversary intercepts the authentication token contained in the URL. By contrast, `BBMA-SLSO` has formal security arguments and is provably secure in a model which takes into account the adversarial control over the network and attacks against the classical browser's security policies that reveal weak identifiers, such as cookies.

Groß et al. prove in [14] the security of WS-Federation passive Requestor Profile—a browser-based protocol for federated identity management. The proof is carried out in the browser model [13] that builds on the Reactive Simulatability framework due to Pfitzmann and Waidner [26]. The model abstracts away the TLS-protected channel through an ideal functionality that captures the same cryptographic task and presupposes ideal users who are able to identify servers based on certificates. There exists no soundness proof that TLS is simulatable and realizes such functionality, especially with respect to the relaxed user behavior assumptions. `BBMA-SLSO` takes explicitly into account the TLS protocol and is shown to be provably secure in the Random Oracle Model when instantiated with the widely deployed key transport cipher suite in server authentication mode.

3 Modeling BBMA with SLSO Policy

In this section we extend our security model for browser-based mutual authentication from [11] towards consideration of cookie-based authentication and the SLSO policy implemented within the browser.

3.1 Protocol Participants and Communication Model

User, Browser, Server, and their Long-Lived Keys. Let $\mathcal{U}$ denote a *human user* for whom we do not make any further assumptions except for the ability to use some naturally born senses. We assume that $\mathcal{U}$ remembers some (high-entropy) *human perceptible authenticator (HPA)* $w \in \mathcal{W}$ (e.g. an image or a video/audio sequence from some space $\mathcal{W}$) as its long-lived key $LL_{\mathcal{U}}$.

To the contrary, the *browser* $\mathcal{B}$ and the *server* $\mathcal{S}$ are modeled as PPT machines. $LL_{\mathcal{B}}$ is the browser's high-entropy long-lived key which contains $(\mathcal{S}, pk_{\mathcal{S}}, cky)$ where $\mathcal{S}$ is the identity (domain name) of the server, $pk_{\mathcal{S}} \in \{0,1\}^{p_1(\kappa)}$ its certified public key, and $cky \in \{0,1\}^{p_2(\kappa)}$ is the cookie set by $\mathcal{S}$ during the establishment of the security association with the *client* which is denoted by $\mathcal{C} = (\mathcal{U}, \mathcal{B})$. (Here and in the following,