Multi-device Authentication using Wearables and IoT

Jan Hajny, Petr Dzurenda and Lukas Malina
Brno University of Technology, Technicka 12, Brno, Czech Republic

Keywords: Authentication, Cryptography, Constrained Devices, Wearables, Internet of Things.

Abstract: The paper presents a novel cryptographic authentication scheme that makes use of the presence of electronic devices around users. The scheme makes authentication more secure by involving devices that are usually worn by users (such as smart-watches, fitness bracelets and smart-cards) or are in their proximity (such as sensors, home appliances, etc.). In our scheme, the user private key is distributed over all personal devices thus cannot be compromised by breaking into only a single device. Furthermore, involving wearables and IoT devices makes it possible to use multiple authentication factors, such as user's position, his behavior and the state of the surrounding environment. We provide the full cryptographic specification of the protocol, its formal security analysis and the implementation results in this paper.

1 INTRODUCTION

In modern society, people are surrounded by a huge amount of so-called smart devices, such as smart-phones, tablets, smart-cards, smart-watches, etc. Furthermore, the amount of various sensors, smart-meters and smart-home appliances increases significantly. The current trend is to interconnect all these devices into a single network, called the Internet of Things (IoT). Although the aforementioned devices have only small computational and memory resources, they are programmable and can communicate with one another.

Despite there are so many electronic devices around us, we usually use only a single device to access electronic services, either a PC, a tablet or a smart-phone. However, if such a device gets compromised by attackers, the security is gone and attackers can access user's assets. For example, if user's smart-phone gets stolen and the password for electronic banking is revealed (or stored in memory), the attacker might get access to the user's account.

We resolve this weakness by involving multiple personal devices in the authentication process. These devices can provide additional authentication data. For example, if a user owns a smart-watch, it would be natural to check its presence during the authentication. Or, it would be useful to check the presence of a wireless home router in some applications where we want to allow the access only from users from a home location. For very sensitive applications, it would make sense to check for multiple factors, such as the password knowledge, the presence of a smart-card and the presence of a Bluetooth Low Energy (BLE) beacon device that certifies position.

In this paper, we provide the description of a cryptographic protocol that allows such an involvement of many constrained devices in the authentication process. We propose a provably secure protocol that distributes the user's private key among multiple devices. To get authenticated, the user must prove the knowledge of all parts of his private key that corresponds to his personal public key. Our protocol is provably secure and easily implementable on all programmable constrained devices, such as smart-cards, smart-watches, sensors and wearables in general.

1.1 Related Work and Contribution

The design of cryptographic protocols for user authentication is the topic of countless scientific papers, starting with the proposals of traditional authentication protocols (Neuman and Ts' O, 1994; Lashkari et al., 2009), provably secure authentication protocols based on zero-knowledge proofs (Schnorr, 1991; Guillou and Quisquater, 1988), to privacy-enhanced authentication protocols (Camenisch and et Al., 2012; Paquin, 2011) and light-weight protocols (Chien and Huang, 2007). Since personal and wearable smart devices have started to appear only very recently,

Hajny, J., Dzurenda, P. and Malina, L.
Multi-Device Authentication using Wearables and IoT.
DOI: 10.5220/0006000004830488
In *Proceedings of the 13th International Joint Conference on e-Business and Telecommunications (ICETE 2016) - Volume 4: SECRYPT*, pages 483-488
ISBN: 978-989-758-196-0

not many papers focusing on using the combination of many devices, i.e. the multi-device authentication, exist. Xu (Xu, 2015) focuses on biometric authentication using wearables, namely on face recognition using smart-glass and gait recognition using smart-watch. Cha *et al.* (Cha et al., 2015) present a simple model for two device authentication for micro-payment systems using a mobile and wearable devices. Nevertheless, their approach lacks more details and concrete cryptographic functions. To some extent, the concepts of continuous authentication (Shepherd, 1995) and progressive authentication (Riva et al., 2012) are close to our approach as they are also based on combining multiple sources of authentication data. However, the schemes are using mainly biometric authentication factors. The most related work from 2015 (Gonzalez-Manzano et al., 2015) presents an access control mechanism for cloud-based storage service access by using a set of devices. However, their scheme is based on symmetric cryptography, thus does not provide non-repudiation. Furthermore, there is no formal security analysis provided in the paper.

Based on the current state analysis, to our best knowledge, we present the first cryptographic scheme that 1) allows strong multi-device authentication, 2) is provably secure, 3) provides non-repudiation and allows private keys to never leave the user device, 4) is easily implementable on personal and wearable devices and 5) allows simple registration and deregistration of personal devices. Using this authentication scheme, the practical access control mechanisms can get much more secure without any negative influence on usability and user friendliness.

1.2 Paper Outline

We provide the preliminaries in Sec. 2, the security model and description of protocols in Sec. 3, the security proof in Sec. 4 and the implementation results in Sec. 5.

2 PRELIMINARIES

2.1 Notation

We describe Proof of Knowledge protocols (PK) using the efficient notation introduced by Camenisch and Stadler (Camenisch and Stadler, 1997a). The protocol for proving the knowledge of a discrete logarithm of an element c with respect to a generator g is denoted as $\mathrm{PK}\{\alpha : c = g^{\alpha}\}$. The symbol ":" means "such that", "$|$" means "divides", "$|x|$" is the bitlength of x and "$x \in_R \{0,1\}^l$" is a randomly chosen bitstring of maximum length l.

2.2 Used Primitives

Our scheme is based on Schnorr's identification scheme (Schnorr, 1991). That, in turn, makes use of the protocols for the interactive proof of knowledge of a discrete logarithm (Camenisch and Stadler, 1997b). Using the cryptographic proofs of knowledge, it is possible to prove the knowledge of a private value of a discrete logarithm w with respect to public values c, g, p such that $c \equiv g^w \pmod{p}$ holds in modular multiplicative group $\mathbb{Z}_p^*$ where p is a large prime and g is a group generator. The protocol can be denoted as $\mathrm{PK}\{w : c = g^w\}$. We use the modification of this protocol called the proof of representation, denoted as $\mathrm{PK}\{w_0, w_1, \ldots, w_i : c = g_0^{w_0} g_1^{w_1} \ldots g_i^{w_i}\}$. Furthermore, we use a signature scheme that can be obtained by hashing the protocol challenge e with the message using the Fiat-Shamir heuristics (Fiat and Shamir, 1987). The signature on message m is then denoted as $\mathrm{SPK}\{w_0, w_1, \ldots, w_i : c = g_0^{w_0} g_1^{w_1} \ldots g_i^{w_i}\}(m)$.

3 MULTI-DEVICE AUTHENTICATION

In multi-device authentication, there are three types of entities (or roles) in the system:

- **Verifiers:** usually service providers that need to verify the identity of their users.
- **Users:** customers that are represented by their master devices (PCs, laptops, smart-phones, tablets, ...). Users need to prove their identity.
- **Devices:** constrained personal devices (smart-cards, smart-watches, sensors, RFID tags, ...), that are involved in the authentication process to strengthen security.

These entities engage in the following protocols:

- $(spar, (sk_0, \ldots, sk_i), pk_U) \leftarrow \mathtt{Setup}(k, d)$ protocol: the protocol is run by a Verifier and a User to generate and share initial parameters. It inputs the security parameter k, the maximum of user devices d and outputs the system parameters $spar$ and User's initial keypair $(sk_0, \ldots, sk_i), pk_U$.
- (Accept/Reject) $\leftarrow \mathtt{Authenticate}(spar, (sk_0, \ldots, sk_i), pk_U)$ protocol: the protocol is run jointly by a User, his devices and a Verifier to prove the knowledge of User's private keys. It inputs the system parameters $spar$, the User's public key pk_U, all corresponding private keys $(sk_0, \ldots, sk_i)$

and outputs Accept if the proof is valid and Reject otherwise.

- $(pk_U) \leftarrow$`Register`$(spar, (sk_0, \ldots, sk_i), pk_U, sk_{i+1})$ protocol: the protocol is run jointly by a User, his devices and a Verifier to register a new device in the system. It inputs the system parameters *spar*, new $(i+1)$'th device's private key sk_{i+1}, the User's keypair $(sk_0, \ldots, sk_i), pk_U$ and outputs an updated User's public key pk_U that corresponds to sk_{i+1} and all previous private keys of the user.
- $(pk_U) \leftarrow$`Deregister`$(spar, (sk_0, \ldots, sk_i), pk_U, sk_{i+1})$ protocol: the protocol is run jointly by a User and the Verifier to deregister the public key of his device, in case the device needs to be revoked (due to loss, damage, theft, etc.). It inputs the system parameters *spar*, existing $(i+1)$'th device's private key sk_{i+1}, the User's keypair $(sk_0, \ldots, sk_i), pk_U$ and outputs an updated User's public key pk_U that corresponds to all previous private keys of the user except sk_{i+1}.

In classical authentication, the `Authenticate` protocol only proves User's knowledge of a password and keys stored in his master device to a Verifier. In multi-device authentication, each device has its private cryptographic key that corresponds to a general public key stored by a Verifier. The `Authenticate` protocol proves the knowledge of all private keys to a Verifier without revealing them. Thus, authentication is successful only if the whole group of preselected devices participate in the protocol. However, this group can be changed jointly by Users and Verifiers, using the `Register` and `Deregister` protocols.

3.1 Security Model

We use and prove properties for authentication protocol completeness, soundness and zero-knowledge (Quisquater et al., 1989). The completeness property states that honest Users are almost always accepted by Verifiers, the soundness property states that dishonest Users are almost always rejected by Verifiers and the zero-knowledge property states that the protocol leaks no information about Users' private keys, using the simulation paradigm (i.e., all the public protocol values can be efficiently generated without the knowledge of private keys).

Definition 1. *Authentication completeness. An honest Verifier rejects an honest User (i.e., the one using private keys that correspond to the public key) with probability negligible in the length of the security parameter k.*

Definition 2. *Authentication soundness. An honest Verifier accepts a dishonest User (i.e., the one using private keys that do not correspond to the public key) with probability negligible in the length of the security parameter k.*

Definition 3. *Authentication zero-knowledge. There exist a simulator $\mathbb{S}$ that is able to efficiently generate a protocol transcript indistinguishable from a real protocol transcript without the knowledge of private keys.*

3.2 Scheme Instantiation

In this section, we provide the concrete instantiation of the protocols used in our scheme. All operations are computed in $\mathbb{Z}_p^*$.

3.2.1 `Setup` Protocol

On the input of the security parameter k and device number parameter d, a Verifier randomly selects a group $\mathbb{G} = \langle g \rangle$ of prime order $q : |q| = k$ where DL assumption holds, chooses $d+1$ random elements $(\alpha_0, \alpha_1, \ldots, \alpha_d) \in_R \mathbb{Z}_q$, computes $g_l = g^{\alpha_l}$ for all $0 \leq l \leq d$ and outputs $(\mathbb{G}, (g_0, \ldots, g_d))$ as public system parameters *spar* to all Users and devices over a secure channel[1]. A User selects his private key at random, i.e., computes $sk_0 \in_R \mathbb{Z}_q$ and computes his public key as $pk_0 = g_0^{sk_0}$. If some additional device is already present, it also generates its private key, i.e. computes $sk_1 \in_R \mathbb{Z}_q$, and computes its public key as $pk_1 = g_1^{sk_1}$. The same applies if more devices are present. We note that the device private key never leaves the device, only the public key is revealed. Finally, the User (represented by his master user device) computes the user public key as $pk_U = \prod_{i=0}^{l} pk_i$ for all l available devices and distributes this public key to the Verifier over a secure channel.

3.2.2 `Authenticate` Protocol

In the `Authenticate` protocol, the User must prove that he knows all private keys $sk_o, \ldots, sk_i$ that were used to construct the public key pk_U. This can be realized by the proof of discrete logarithm representation, a protocol denoted as $\mathrm{PK}\{(sk_0, \ldots, sk_i) : pk_U = g_0^{sk_0} \ldots g_i^{sk_i}\}$. Since the User's master device does not know the private keys, except sk_0, the proving protocol must be distributed among all devices, as depicted in Fig. 1 in CS notation and in Fig. 2 in full notation.

3.2.3 `Register` Protocol

The `Register` protocol is used when a new device needs to be added to the set of user devices. In that

[1]These values can be pre-shared in software.

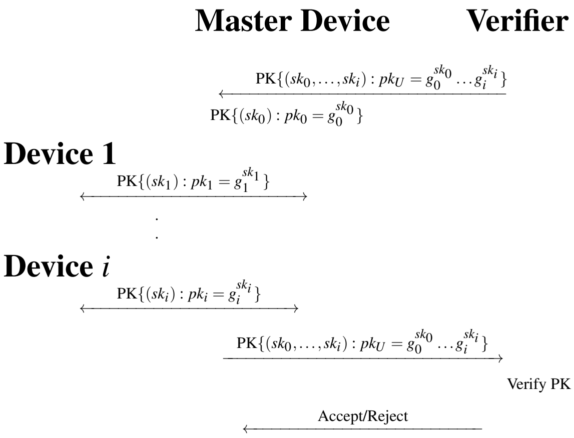

Figure 1: `Authenticate` protocol in CS notation.

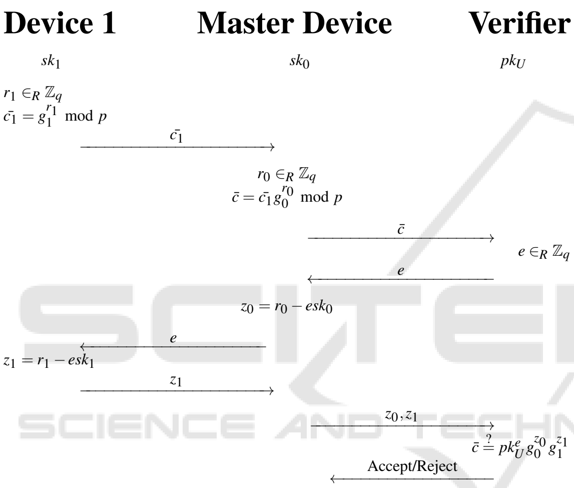

Figure 2: `Authenticate` protocol for 1 master device and 1 additional device in full notation.

case, the new device generates its private key, i.e., computes $sk_{i+1} \in_R \mathbb{Z}_q$, and computes its public key as $pk_{i+1} = g_{i+1}^{sk_{i+1}}$. The new public key must be delivered to the master device using a secure channel. Then, the master device may authenticate itself to the Verifier (using the `Authenticate` protocol) and provide the new public key pk_{i+1}. The Verifier then updates the main User's public key $pk_U = pk_U * pk_{i+1}$. After this update, the new $(i+1)$'th device must be always used in the `Authentication` protocol. The *Register* protocol is depicted in Fig. 3.

3.2.4 `Deregister` Protocol

In case some of devices gets lost, stolen or stops working, a User can use the `Deregister` protocol to remove it from the set of registered devices. The User first sends the public key of the invalid device, e.g. pk_{i+1}, to the Verifier. The Verifier temporarily

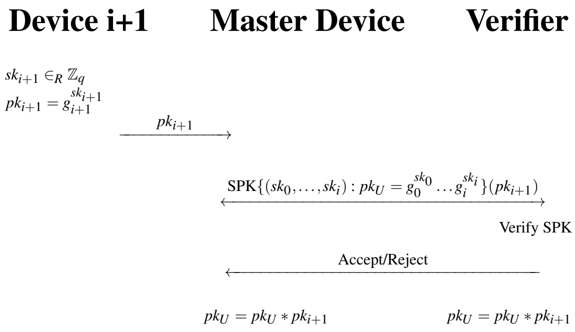

Figure 3: `Register` protocol.

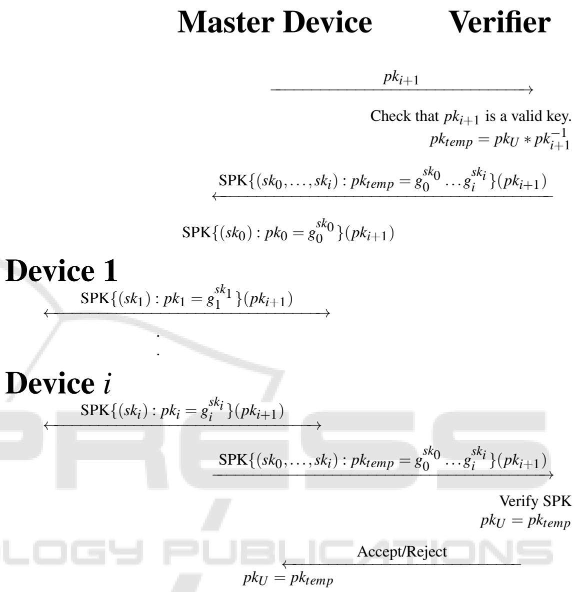

Figure 4: `Deregister` protocol.

removes the device and computes the temporal public key $pk_{temp} = pk_U * pk_{i+1}^{-1}$. Then, the Verifier asks the User to authenticate with respect to the pk_{temp}. If the User is able to successfully finish the authentication protocol, the Verifier sets the temporal public key as permanent, i.e. sets $pk_U = pk_{temp}$. The protocol is depicted in Fig. 4.

4 SECURITY PROOF

We prove the completeness, soundness and zero-knowledge in this section.

Theorem 1. *Authentication protocol is complete as defined in Def. 1.*

Proof. We prove the authentication protocol's completeness using the verification equation used in the authentication protocol depicted in Fig. 2.

Table 1: Performance results for 1280 bit keys ($|p| = 1280, |q| = 160$).

Type	Product	ModExp	RNG	ModMul	Sub	Total [ms]
Smart-watch	Sony SmartWatch 3 SWR50	2.3	1.4	< 0.1	< 0.01	**3.7**
Smart-phone	Nexus 5 LG	1.9	7.4	< 0.1	< 0.01	**9.3**
Micro-computer	Raspberry Pi 1 model B	59.3	0.8	< 0.6	< 0.1	**60.1**
Smart-card	MULTOS ML4-P17	227	49	188	48	**512**
Smart-card	MULTOS ML3-80KR1	403	45	195	44	**687**
Smart-card	MULTOS MC4-P16	333	68	255	56	**712**
Smart-card	SmartCafe 4.x	356	47	1159	79	**1641**
Smart-card	SmartCafe 3.2	59	31	1737	94	**1921**
Smart-card	J3A081	75	31	2510	179	**2795**
Secure element	CertGate microSD	78	34	2694	168	**2974**

Table 2: Performance results for 2048 bit keys ($|p| = 2048, |q| = 256$).

Type	Product	ModExp	RNG	ModMul	Sub	Total [ms]
Smart-watch	Sony SmartWatch 3 SWR50	7.5	2	< 0.1	< 0.01	**9.5**
Smart-phone	Nexus 5 LG	5.2	9.2	< 0.1	< 0.01	**14.4**
Micro-computer	Raspberry Pi 1 model B	216.2	1.2	< 0.7	< 0.1	**217.4**
Smart-card	MULTOS ML4-P17	346	62	190	48	**646**
Smart-card	MULTOS ML3-80KR1	530	56	194	44	**824**
Smart-card	MULTOS MC4-P16	484	84	256	56	**880**
Smart-card	SmartCafe 4.x	617	47	1536	79	**2279**
Smart-card	SmartCafe 3.2	188	31	2532	94	**2845**
Smart-card	J3A081	258	47	3962	179	**4446**
Secure element	CertGate microSD	263	48	4153	168	**4632**

$$\bar{c} = pk_U^e g_0^{z_0} g_1^{z_1} = (g_0^{sk_0} g_1^{sk_1})^e g_0^{r_0 - esk_0} g_1^{r_1 - esk_1} = g_0^{r_0} g_1^{r_1} = \bar{c}$$

□

Theorem 2. *Authentication protocol is sound as defined in Def. 2.*

Proof. Suppose that a user does not know the private keys and is ready to correctly respond to at least two Verifier's challenges (denoted as e, e') by sending (z_0, z_1) and (z'_0, z'_1). Then, the following equations must hold for the User to be accepted.

$$\bar{c} = pk_U^e g_0^{z_0} g_1^{z_1}$$
$$\bar{c} = pk_U^{e'} g_0^{z'_0} g_1^{z'_1}$$

By dividing we get:

$$1 = pk_U^{e-e'} g_0^{z_0 - z'_0} g_1^{z_1 - z'_1}$$

And finally we get:

$$pk_U = g_0^{\frac{z_0 - z'_0}{e' - e}} g_1^{\frac{z_1 - z'_1}{e' - e}}$$

And we reached the contradiction because the user knows the private keys $sk_0 = \frac{z_0 - z'_0}{e' - e}$ and $sk_1 = \frac{z_1 - z'_1}{e' - e}$. □

Theorem 3. *Authentication protocol is zero-knowledge as defined in Def. 3.*

Proof. We prove the zero-knowledge property by constructing the zero-knowledge simulator $\mathbb{S}$. The simulator works in the following steps.

1. Randomly selects the responses $\hat{z_0}, \hat{z_1} \in_R \mathbb{Z}_q$.
2. Randomly selects the challenge $\hat{e} \in_R \mathbb{Z}_q$.
3. Computes the commitment $\hat{\bar{c}} = pk_U^{\hat{e}} g_0^{\hat{z_0}} g_1^{\hat{z_1}}$.

The simulator's output is computationally indistinguishable from the real protocol transcript, i.e. $(\hat{\bar{c}}, \hat{e}, (\hat{z_0}, \hat{z_1})) \cong_c (\bar{c}, e, (z_0, z_1))$, because all pairs are selected randomly and uniformly from the same sets. □

5 IMPLEMENTATION ASPECTS

In this section, we prove that our scheme is efficient and easy to implement even on constrained devices. We implemented all required operations of the authentication protocol[2] on a set of devices that have very limited resources. We used devices that can be expected around modern users, namely a smart-watch, smart-cards, a smart-phone, a secure element with tamper-resistant hardware and a micro-computer. The results for individual operations and

[2]ModExp - modular exponentiation, RNG - random number generation, ModMul - modular multiplication and Sub - subtraction.

the total time of the authentication protocol are shown in Tab. 1 for 1280-bit keysize and in Tab. 2 for 2048-bit keysize.

Based on the implementation results, we state that the authentication protocol can be easily implemented on smart-phones and smart-watches with running times around 10 ms, on micro-computers with running times under 100 ms for the standard variant and around 200 ms for the more secure variant. The protocol can be also implemented on programmable smart-cards using the Multos smart-card platform with running times under 1 s for all variants. The worst results were obtained using a microSD secure element, a device that is used for storing sensitive cryptographic information on mobile phones. Using this device, the authentication protocol would take around 3 seconds.

6 CONCLUSION

In this paper, we proposed a novel multi-device authentication scheme. By using the inputs from personal and wearable devices, the authentication process gets more secure and reliable as it is possible to verify not only user's knowledge of a password, but the presence of his wearables, tags and smart-devices at his location. The scheme does not require any additional actions from a user, allows easy registration of new personal devices and deregistration of invalid devices. The full security analysis is provided and implementation aspects are described in this paper. As the next step, we focus on adding privacy-enhancing features to this scheme.

ACKNOWLEDGMENT

Research was sponsored by the Czech Science Foundation project nr. 14-25298P Research into cryptographic primitives for secure authentication and digital identity protection", the Technology Agency of the Czech Republic project TA04010476 "Secure Systems for Electronic Services User Verification" and the National Sustainability Program LO1401. For the research, infrastructure of the SIX Center was used.

REFERENCES

Camenisch, J. and et Al. (2012). Specification of the identity mixer cryptographic library. Technical report, IBM Research - Zurich.

Camenisch, J. and Stadler, M. (1997a). Efficient group signature schemes for large groups. In *Advances in Cryptology - CRYPTO '97*, volume 1294 of *LNCS*, pages 410–424. Springer Berlin / Heidelberg.

Camenisch, J. and Stadler, M. (1997b). Proof systems for general statements about discrete logarithms. Technical report, IBM.

Cha, B.-R., Lee, S.-H., Park, S.-B., and Ji, G.-K. L. Y.-K. (2015). Design of micro-payment to strengthen security by 2 factor authentication with mobile & wearable devices.

Chien, H.-Y. and Huang, C.-W. (2007). Security of ultra-lightweight rfid authentication protocols and its improvements. *SIGOPS Oper. Syst. Rev.*, 41(4):83–86.

Fiat, A. and Shamir, A. (1987). How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO 86*, volume 263 of *LNCS*, pages 186–194. Springer Berlin / Heidelberg.

Gonzalez-Manzano, L., de Fuentes, J., and Orfila, A. (2015). Access control for the cloud based on multi-device authentication. In *Trustcom/BigDataSE/ISPA, 2015 IEEE*, volume 1, pages 856–863. IEEE.

Guillou, L. C. and Quisquater, J.-J. (1988). *EUROCRYPT '88: Workshop on the Theory and Application of Cryptographic Techniques*, chapter A Practical Zero-Knowledge Protocol Fitted to Security Microprocessor Minimizing Both Transmission and Memory, pages 123–128. Springer Berlin Heidelberg, Berlin, Heidelberg.

Lashkari, A. H., Danesh, M. M. S., and Samadi, B. (2009). A survey on wireless security protocols (wep, wpa and wpa2/802.11 i). In *Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on*, pages 48–52. IEEE.

Neuman, B. C. and Ts' O, T. (1994). Kerberos: An authentication service for computer networks. *Communications Magazine, IEEE*, 32(9):33–38.

Paquin, C. (2011). U-prove cryptographic specification v1.1. Technical report, Microsoft Corporation.

Quisquater, J.-J., Guillou, L., Annick, M., and Berson, T. (1989). How to explain zero-knowledge protocols to your children. In *Proceedings on Advances in cryptology*, CRYPTO '89, pages 628–631, New York, NY, USA. Springer-Verlag New York, Inc.

Riva, O., Qin, C., Strauss, K., and Lymberopoulos, D. (2012). Progressive authentication: Deciding when to authenticate on mobile phones. In *Presented as part of the 21st USENIX Security Symposium (USENIX Security 12)*, pages 301–316, Bellevue, WA. USENIX.

Schnorr, C. P. (1991). Efficient signature generation by smart cards. *Journal of Cryptology*, 4:161–174.

Shepherd, S. J. (1995). Continuous authentication by analysis of keyboard typing characteristics. In *Security and Detection*, pages 111–114.

Xu, W. (2015). Mobile applications based on smart wearable devices. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pages 505–506. ACM.