# Token-based Authentication for Smartphones

Manuel Koschuch[1], Matthias Hudler[1], Hubert Eigner[2] and Zsolt Saffer[1]
[1]*Competence Centre for IT-Security, FH Campus Wien - University of Applied Science, Favoritenstrasse 226, 1100, Vienna, Austria*
[2]*FH Campus Wien - University of Applied Science, Favoritenstrasse 226, 1100, Vienna, Austria*

Keywords: Smartphone, Security, Token, Usability, Authentication, Challenge-response.

Abstract: Due to short - but frequent - sessions of smartphone usage, the fast and easy usability of authentication mechanisms in this special environment has a big impact on user acceptance. In this work we propose a user-friendly alternative to common authentication methods (like PINs and patterns). The advantages of the proposed method are its security, fastness, and easy usage, requiring minimal user interaction compared to other authentication techniques currently used on smartphones. The mechanism described uses the presence of a Bluetooth-connected hardware-token to authenticate the user and can easily be implemented on current smartphones. It is based on an authentication protocol which meets the requirements on energy efficiency and limited resources by optimizing the communication effort. A prototype was implemented on an Android smartphone and an MSP430 based MCU. The token allows fast authentication without the need for additional user action. The entire authentication process can be completed in less than one second, the developed software prototype requires no soft- or hardware modifications (like rooting) of the Android phone.

## 1 INTRODUCTION

Smartphones became more and more popular during the last years. With built-in distribution platforms for applications by third party developers, mobile devices today are able to cover a broad spectrum of application scenarios (Cozza et al., 2012). As a consequence the amount of critical, personal and confidential data stored on smartphones is growing steadily and therefore its protection gets more important. An important building block of the security measures available on smartphones today is the user authentication.

Available authentication methods on current devices suffer from not being secure, fast or easy to use enough for the typical user, or are not secure when used in insecure environments like public places with a lot of potentially malicious observers.

Either the mechanisms require cumbersome interaction by the user (e.g. entering a long passphrase each time the phone is unlocked) or are inherently unsafe (e.g. authentication patterns, that can be recovered by observing residues on the touchscreen (Aviv et al., 2010) or by simply spying on the user when unlocking the phone in public).

Token-based authentication, as proposed in this paper, can provide a secure, fast, and user-friendly solution to identify the user. Unlike the existing concept of using the smartphone as an authentication token (Thanh et al., 2009), we propose the token as a separate piece of hardware to authenticate the user to the smartphone.

The remainder of this paper is structured as follows:

Section 2 gives an overview of the special challenges when authentication is used in the context of smartphones. Sections 3 and 4 then describe our proposed solution, while Section 5 gives a preliminary evaluation of its security. Finally, Section 6 gives an overview of the actual implementation, while Section 7 provides some closing remarks.

## 2 AUTHENTICATION ON SMARTPHONES

Smartphone usage differs in many ways from usage of traditional devices like personal computers. As a consequence, authentication methods on these devices have to cover different requirements.

Smartphones are usually always-on devices. In case of theft or loss a potential attacker acquires a

Koschuch M., Hudler M., Eigner H. and Saffer Z..
Token-based Authentication for Smartphones.
DOI: 10.5220/0004520000510056
In *Proceedings of the 4th International Conference on Data Communication Networking, 10th International Conference on e-Business and 4th International Conference on Optical Communication Systems* (DCNET-2013), pages 51-56
ISBN: 978-989-8565-72-3

powered and operational device. To protect the critical data on the phone from unauthorized access, the user has to authenticate herself to the phone before usage, and it has to be locked afterwards.

On the other hand smartphone usage typically consists of short but frequent sessions. As a consequence, the chosen user authentication method has a large impact on the usability of smartphones. For many users, the speed and ease of use of the available methods are important factors in deciding whether authentication is enabled or not (Dörflinger et al., 2010).

Finally, due to the high mobility of these devices, they will often be used in insecure places with a great number of potential observers. An easy and user-friendly approach to authentication requires the security of the authentication method to be independent of the environment the phone is used in.

Based on the characteristics described above, the following requirements for authentication on smartphones can be defined:

- The user has to authenticate before each session
- A passing observer should learn next to nothing about the secret used for authentication
- User effort has to be minimal, authentication should be fast and secure

The authentication methods currently available on smartphones are not able to meet all of these requirements.

The most common method is PIN/Password authentication. It requires time-consuming user interaction (always typing in the password when unlocking the phone), hence the typical user is very likely to choose easy to type, short passwords/PINs, and is not secure when used publicly in crowded places.

The alternative to the password-based approach, graphical patterns, benefit from faster input, but are even easier to memorize by casual observers.

Biometric techniques like fingerprint scanning or facial recognition are still not widely deployed and can usually be circumvented by using a simple photograph or fake fingerprints.

In contrast to these methods, the token-based authentication approach described in the next section covers all the requirements defined above.

## 3 THE PROPOSED METHOD

The proposed method uses a hardware token to authenticate the user to the smartphone. When the token is in range, no additional user interaction is required. A service running on the smartphone handles the authentication process. Therefore an authentication protocol as described in the next section is required. The protocol has to take the limited resources of the token hardware into account. It is important to keep the authentication process as fast and secure as possible, with respect to the low computational power of the MCU and the limited amount of energy available to the token.

If the token is out of range, or is unable to communicate with the smartphone, the proposed method falls back to the most commonly used password/PIN approach. Since this rarely happens during normal operating conditions, the password can be chosen long and secure without unduly affecting usability.

A successful token-based authentication causes no user interaction at all. On the downside, the user has to carry an additional piece of hardware with him. In practice the token functionality could for example be integrated with a USB memory stick and attached to a keyring, resulting in a negligible extra burden for the user.

An attacker now either has to acquire both, the smartphone and the hardware token, or guess/know the password, which can be chosen to be long, complicated and secure, since it only has to be entered when the token is not present.

## 4 THE AUTHENTICATION PROTOCOL

The authentication protocol described in this paper belongs to the class of challenge-response protocols. They allow one party to gain assurance of the identity of another (Menezes et al., 2001). For the first proof-of-concept we decided to implement our own protocol, based on well-known primitives like Diffie-Hellman, which also allows for periodic rekeying. As future work we plan to also evaluate different existing challenge-response protocols for the suitability of the constrained environment present in our usage scenario. Our protocol consist of three phases:

***Key Establishment.*** In this phase a shared secret is generated. It is triggered by the user and is based on the Diffie-Hellman key exchange protocol (Diffie and Hellman, 1976).

***Authentication.*** This phase consists of a hash based challenge-response protocol.

***Rekeying.*** Periodical rekeying prevents brute force attacks. After a given time or number of authentication attempts a new shared secret is generated.

### 4.1 Key Establishment

Before authentication can take place, the token and the smartphone need to generate a shared secret. The user triggers this action. He activates the key establishment function in the configuration software on the handset and then presses a button on the token. Now a session key is established. This step should be executed in a - more or less - secure environment, away from potential eavesdroppers and men-in-the-middle. It has to be performed only when agreeing on a new secret or when pairing with a new device.

In the following we give a short summary of the key establishment phase using the Diffie-Hellmann protocol, where $A \rightarrow B$ denotes a message sent from $A$ to $B$. The token $T$ and mobile phone $M$ share a prime $p$ and a generator $g$.

**Step 1:** $M$ and $T$ each generate a random number $r_M$ and $r_T$. These are their respective secret keys.

**Step 2:** Each party computes its own $A_x$ and sends this parameter to its partner.

$$\mathbf{M}: A_M = g^{r_M} \bmod p$$
$$\mathbf{M} \rightarrow \mathbf{T}: A_M$$
$$\mathbf{T}: A_T = g^{r_T} \bmod p$$
$$\mathbf{T} \rightarrow \mathbf{M}: A_T$$

**Step 3:** The authentication partners compute the shared secret $S$ from these parameters.

$$\mathbf{M}: S = {A_T}^{r_M} = g^{r_T * r_M} \bmod p$$
$$\mathbf{T}: S = {A_M}^{r_T} = g^{r_M * r_T} \bmod p$$

**Step 4:** Finally both parties generate the session key $K$ from the shared secret by hashing this value.

$$\mathbf{M}, \mathbf{T}: K = h(S)$$

Note that it would also be possible to add more entropy to this key by using system parameters like timestamps in the hash, with the tradeoff of increasing the message size or requiring time synchronization between the token and the smartphone.

### 4.2 Authentication

The authentication is performed by a challenge-response protocol. Hashing is used to secure the communication since hash algorithms are deemed the most energy efficient among cryptographic algorithms, according to (Potlapally et al., 2006). As communication has the biggest impact on power consumption and no mutual authentication is needed, the protocol only authenticates the token to the smartphone. This allows successful protocol execution by using only two messages.

**Step 1:** $M$ generates a nonce $N$ and sends it to the token $T$.

$$\mathbf{M} \rightarrow \mathbf{T}: N$$

**Step 2:** $T$ concatenates (denoted by $|$) the session key $K$ with $N$ and generates the hash of the resulting value. The calculated hash value $H_T$ is sent back to $M$.

$$\mathbf{T} \rightarrow \mathbf{M}: H_T = h(N|K)$$

**Step 3:** $M$ calculates $H_M$ from the nonce and $K$ ($H_M = h(N|K)$) and compares it to the received $H_T$. If they are equal, $T$ is authenticated successfully.

### 4.3 Rekeying

To weaken the possibility of brute force attacks, the session key is subject to change after either a pre-defined time has elapsed or a given number of messages has been sent. Each of the two parties can initiate the rekeying process at any time. Note that while the shared secret can also be changed by performing the key establishment phase given in 4.1, the main idea of the rekeying phase is to periodically refresh the used key without active user interaction. The steps of the rekeying phase can be summarized as follows:

**Step 1:** Either $M$ initiates the rekeying by sending a start message to $T$ or $T$ starts on its own. In both cases $T$ generates a random number $r_T$ and sends it to $M$.

$$\mathbf{T} \rightarrow \mathbf{M}: r_T$$

**Step 2:** $M$ saves the old session key $K$ as $K_{old}$. After this a new random number $r_M$ is generated. $r_M$ is used to calculate the new session key as the hash value of $r_M$ and $K_{old}$. Finally, $r_M$ and the hash value of $r_M$, $r_T$ and $K_{old}$ are sent to $T$.

$$\mathbf{M}: K_{old} = K$$
$$\mathbf{M}: K = h(r_M|K_{old})$$
$$\mathbf{M} \rightarrow \mathbf{T}: r_M, H_M = h(r_M|r_T|K_{old})$$

**Step 3:** $T$ receives $r_M$ and the hash value and calculates the hash value by itself. If this is equal to the one received, the new session key $K$ is generated.

$$\mathbf{T}: K_{old} = K$$
$$\mathbf{T}: H_T = h(r_M|r_T|K_{old}) \stackrel{?}{=} H_M$$
$$\mathbf{T}: K = h(r_M|K_{old})$$

**Step 4:** *M* still does not know if the rekeying was successful. This will be checked during the next authentication attempt. After receiving the hash value from the token, *M* calculates its own hash value with the new session key (as detailed in 4.2, Step 3). If these values match the authentication, rekeying was successful. Otherwise *M* uses $K_{old}$ to calculate the hash value. If this value matches, the rekeying has failed and has to be restarted. If this value still does not match, an error has occurred during the authentication process.

Note that this approach leaves room for a desynchronization and/or DOS attack, as detailed in the following section.

# 5 SECURITY ANALYSIS

In (Mobahat, 2010) the author defines several security threats for authentication. In this section the proposed authentication protocol will be analyzed against these issues.

## 5.1 Information Leakage

The messages sent do not contain information beside the hashes of values containing the key and random numbers. Due to the one-way characteristics of cryptographic hash functions this is not a problem. The token will answer on every request for the same nonce *N* with the same hash as long as the key does not change. This allows identifying a user by his token.

A possibility to prevent user identification by attackers would be mutual authentication. The smartphone has to authenticate itself to the token before an answer is sent. This solution requires additional communication effort.

## 5.2 Spoofing Attack

It is known that the Diffie-Hellman key exchange protocol is vulnerable to man-in-the-middle attacks. An eavesdropper can intercept the secret while the shared secret is established. This could be avoided by using an authenticated key exchange protocol, but since this explicit secret establishment only happens very infrequently and at the user's request, it can be required and therefore assumed that it happens in a secure environment under the user's control.

During the authentication process itself a man-in-the-middle can only act as relay station. He can intercept messages and send them to the token/mobile phone and will extend the authentication range this way, thereby being able to unlock the phone without having obtained the token. To thwart this attack the token should be powered off when the secured device is lost.

## 5.3 Replay Attack

Intercepted hashes can only be used for replay attacks when the mobile phone reuses a nonce between two rekeyings. To prevent brute force collection of hash values there is a limit for the number of authentications performed with the same key. If this limit is reached the rekeying process is initiated.

## 5.4 Forward Security

If the key setup is compromised and the initial key is revealed, or the attacker gains a key from authentication messages, he will be able to obtain the actual key as long as he receives every rekeying message until then. If an old key is obtained from a hash value after a rekeying, the attacker is unable to make any use of it. To secure the keys against brute force attacks, the shared key has a limited lifetime and number of authentication cycles.

## 5.5 Desynchronization

Desynchronization can stop the authentication or rekeying, and may even completely prevent any further authorizations. In this case, a new key establishment phase as described in 4.1 is necessary. Usage of the phone itself is not affected, since when authentication cannot be completed using the token method, the phone simply asks the user for the passphrase.

## 5.6 DOS Attack

Denial of service attacks can be mounted against the token and the mobile phone. The token can be cut off the communication by interrupting the radio channel. Repeated authentication or rekeying requests can have a significant impact on battery lifetime. The battery usage of the mobile phone can be raised by frequently requesting rekeying. These battery draining attacks can be softened by limiting the number of request in a given time. If the token is not available the user can still use a password or pin to unlock the smartphone.

# 6 PROTOTYPE

The prototype was implemented on a smartphone running Android 2.3. As hardware base for the token

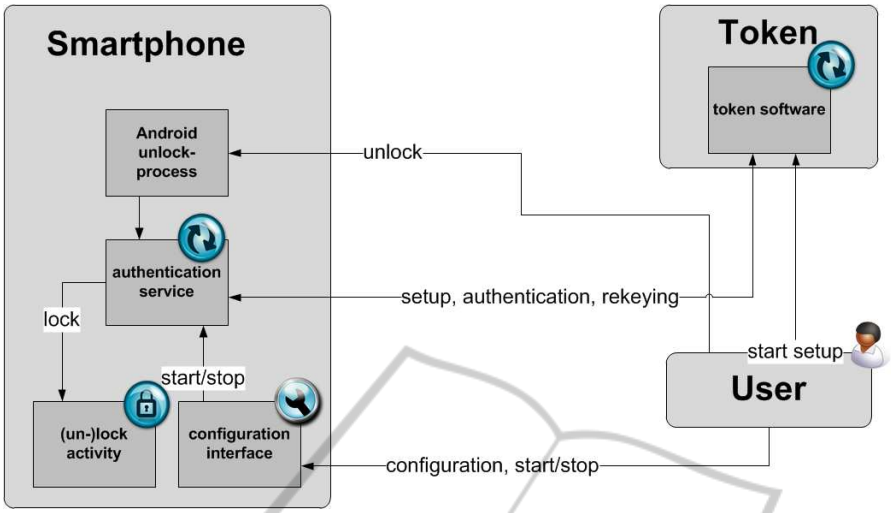


Figure 1: Interactions between the prototype's components and the user.

the EZ430-RF2560 evaluation board (Texas Instruments, 2012) from Texas Instruments was chosen. The MSP430BF5190 MCU runs at up to 25 MHz and has 16 KB RAM. The two partners communicate over Bluetooth. Figure 1 shows the schematic layout of the prototype components as well as their interaction with each other and the user.

On the smartphone side a background service is watching for unlock events. These events are triggered by Android when the user unlocks the phone. To prevent accessing the device before the authentication was successful, the Unlock-Activity is started. If the user tries to leave this screen the phone will be locked again. After successful authentication the unlock screen is closed and the smartphone is ready for use. In case of authentication failure a user-defined password is set and the phone is locked. For password protection, the built-in Android password functionality is used. Password authentication automatically ends when the correct password is supplied.

Figure 2 shows the authentication process from the user's point of view. When the token is not present, as seen on the top of the right side, the user is asked to enter her password. If the token is in range, as depicted at the bottom, the smartphone is unlocked without requiring any further interaction.

SHA-256 was selected as hash function. On the smartphone the standard Java security libraries were used. On the token the MIRACL library (CertiVox, 2013) provides the functionality needed for hashing and key generation. The prime $p$ was chosen to be 1.024 bits long, $g$ is the number 2, $r_M$ and $r_T$ consisted of 256 bits each.

In terms of computing time, for every authentication both the token and the mobile phone have to perform a single hash calculation. The token has to


Figure 2: Authentication process from user's view.

receive and send a 256 bit message (receive the nonce and send the hash-value), as does the mobile phone. Rekeying is more costly, requiring two hash calculations on each side, but still only one 256 bit message has to be sent and received by token and mobile phone, respectively.

Key establishment finally is the most computationally intensive phase, but also the one that has to performed most infrequently. It requires two modular exponentiations with a 1.204 bit long modulus on each side, as well as the exchange of two 1.204 bit long values.

In terms of running time, generation of the shared secret with the Diffie-Hellman protocol takes about 120 million cycles. This gives a duration of 6 to 7 seconds, but since this step only has to performed very infrequently the impact on usability is negligible.

The calculation of the hash value takes about

400.000 cycles or 22 ms. A complete unlock action including authentication can be performed in less than a second which is usually much faster than using common methods like passwords or patterns, and, in contrast to these approaches, completely transparent to the user.

## 7 CONCLUSION

The concept of token-based authentication for smartphones described in this work addresses the problem of improving the security as well as the speed and usability of authentication methods currently available on smartphones. The implemented prototype proves that user authentication can be performed secure, fast and easy to use, without requiring root access to the smartphone. The downside of this approach is the additional hardware which the user has to carry around. Future work is planned to investigate other available challenge-response protocols for their security and suitability to the given constrained environment.

The evaluation of the prototype shows that our proof-of-concept is feasible in practice and that there is even still a margin in terms of communication and computational effort to improve upon our current proposal.

Finding other secure authentication methods for smartphones with a focus on usability still remains a broad field of research. The proposed solution has been designed without relying on any authentication mechanisms inherent in the wireless channel, so it can be easily adapted to new, energy-efficient technologies like Bluetooth 4.0, NFC or others.

## REFERENCES


Aviv, A. J., Gibson, K., Mossop, E., Blaze, M., and Smith, J. M. (2010). Smudge attacks on smartphone touch screens. In *Proceedings of the 4th USENIX conference on Offensive technologies*, WOOT'10, pages 1–7, Berkeley, CA, USA. USENIX Association.

CertiVox (2013). Certivox miracl crypto sdk. `http://certivox.com/index.php/solutions/miracl-crypto-sdk/`.

Cozza, R., Milanesi, C., Zimmermann, A., Nguyen, T. H., Vergne, H. J. D. L., Shen, S., Gupta, A., Sato, A., Lu, C., and Glenn, D. (2012). Market share: Mobile devices by region and country, 4q11 and 2011. Technical report, Gartner Report.

Diffie, W. and Hellman, M. (1976). New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644 – 654.

Dörflinger, T., Voth, A., Krämer, J., and Fromm, R. (2010). "my smartphone is a safe! " the user's point of view regarding novel authentication methods and gradual security levels on smartphones. In *Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on*, pages 1 –10.

Menezes, A. J., van Oorschot, P. C., and Vanstone, S. A. (2001). *Handbook of Applied Cryptography*. CRC Press, 5th printing edition.

Mobahat, H. (2010). Authentication and lightweight cryptography in low cost rfid. In *Software Technology and Engineering (ICSTE), 2010 2nd International Conference on*, volume 2, pages V2–123 –V2–129.

Potlapally, N. R., Ravi, S., Raghunathan, A., and Jha, N. K. (2006). A study of the energy consumption characteristics of cryptographic algorithms and security protocols. *Mobile Computing, IEEE Transactions on*, 5(2):128 – 143.

Texas Instruments (2012). Texas instruments inc.: Msp430 + cc2560 bluetooth platform. `http://www.ti.com/lit/ml/swpt038/swpt038.pdf`.

Thanh, D. V., Jorstad, I., Jonvik, T., and Thuan, D. V. (2009). Strong authentication with mobile phone as security token. In *Mobile Adhoc and Sensor Systems, 2009. MASS '09. IEEE 6th International Conference on*, pages 777 –782.