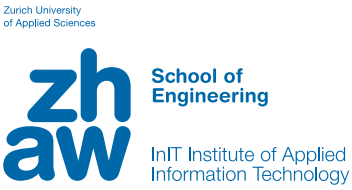




# 2-FACTOR AUTHENTICATION FOR MOBILE APPLICATIONS: INTRODUCING DoubleSec

## TECHNOLOGY WHITEPAPER

DSWISS LTD

INIT INSTITUTE OF APPLIED INFORMATION TECHNOLOGY

JUNE 2010

V1.0

# Motivation

With the increasing desire also of private individuals to access their confidential data even from their mobile devices, the need for strong security controls for such application arises – in the same way as it has years ago in the area of web applications. This paper covers one of the most important parts thereof: the login process that allows an application on a mobile device accessing data from a server using two-factor authentication.

# Introduction

An increasing number of internet-based end-customer applications require two-factor authentication. Text message (SMS) based one-time code distribution (as second factor) is rapidly becoming the most popular choice when strong authentication is needed, for example in e-banking. Low acquisition, distribution and help-desk cost are the main drivers for these so-called mTAN[1] based authentication methods. All of these properties are particularly important for applications that serve large number of users, possibly on a global scale.

With multi-factor authentication, each token available for authenticating the user falls into one of the following three categories:

- Something the user knows (e.g. a password)
- Something the user has (e.g. a hardware token)
- Something the user is (e.g. a fingerprint)

mTAN-based strong authentication makes use of the two categories "something the user knows" (password) and "something the user has" (mobile device). During authentication, the user has to provide the password as well as a one-time secret received by SMS on his mobile phone. Proof of possession of the mobile phone (which is done by providing the received SMS code) is used as 2nd login factor.

With increased capabilities of mobile devices, there's been a trend towards accessing web services[2] over the mobile channel[3] as well. Much like a regular web-user also users that access the service via a mobile application must be authenticated with a mechanism that sports the required strength against identified, relevant threats. However simply transferring the mTAN-approach to mobile app development doesn't work well, mainly because it would be cumbersome or even impossible to be used on the mobile device as it requires the user to switch between applications[4]. As a result, we have to come up with an authentication scheme that is better suited for mobile apps, which should provide security comparable to the two-factor authentication mechanism described above.

In this paper, we propose a strong and practical two-factor authentication scheme for smart phones that does not negatively affect the user's experience or usability and that provides security comparable to "classic" two-factor authentication schemes.

1 mobile Transaction Authentication Number

2 Also such implementing two-factor authentication

3 E.g. as an iPhone app

4 Mobile app and SMS inbox

# Technical Approach

mTAN is based on a simple principle: Once a user has securely proven possession of his mobile phone[5], it can be used as the second login factor. While mTAN uses a unique SMS code that is generated on the server and received during each login attempt, our approach makes use of the "principle of key continuity" (also called "Baby Duck / Duckling"-model) to negotiate a token that is used as the second login factor. This model was already successfully implemented in protocols such as SSH or products such as Phil Zimmerman's zFone. The idea behind this approach is that if we can securely authenticate one session (usually the first one) and derive a shared secret, this secret can be re-used to authenticate later sessions – i.e. each secret $S_n$[6] directly depends the previous secret $S_{n-1}$.

Assuming the shared secret $S_{n-1}$ can be stored securely on the mobile device, the second login factor (i.e. the next shared secret, $S_n$) can only be generated by the mobile device itself. It is therefore dependent on something the user "has". From a security point of view, this is very similar to mTAN but offers much better usability when using a mobile app.

More detailed, the approach works as follows: Before a new session `n` is initiated, the mobile device has stored a shared secret $S_{n-1}$ from the previous session `n-1`. $S_{n-1}$ is then used to compute the shared secret to authenticate the new session using the following formula: $S_n = f(\text{<shared secret}^{7}\text{>}, S_{n-1})$. As long as the token $S_{n-1}$ is adequately secured on the mobile device, proving knowledge of it is sufficient for a second login factor (which the client does by calculating and providing $S_n$). A normal login therefore works as follows:

| Client | | Server |
|---|---|---|
| (looks up $S_{n-1}$) | | (looks up $S_{n-1}$) |
| (computes $S_n = f(\text{<shared secret>}, S_{n-1})$ | | (computes $S_n = f(\text{<shared secret>}, S_{n-1})$ |
| | $S_n$ → | |
| | | (checks $S_n == S_n$) |
| | ← OK | |
| (stores $S_n$) | | (stores $S_n$) |

Missing in the above flow is the mobile device initialization, which also has to be done in a secure manner. In our reference implementation (see page 4), we have decided to use an mTAN code once, to initialize the mobile device. The first secret $S_1$ is therefore referred to as $S_1 = f(\text{<shared secret>}, \text{mTAN-code})$. A re-authentication with a one-time code that is distributed with an SMS can also be used as a "fallback solution"[8] at any time.

To summarize, our approach uses the following messages to authenticate a user:

- *Initialization*: $S_1 = f(\text{<shared secret>}, \text{mTAN-code})$
- *Normal login*: $S_n = f(\text{<shared secret>}, S_{n-1})$
- *Re-synchronization*: $S_n = f(\text{<shared secret>}, \text{mTAN-code})$

5 In high-security applications, this is often made offline, i.e. using letter post or similar

6 Read: "Secret S in session n"

7 E.g. a hash of the password or a pre-established Diffie-Hellman secret

8 E.g. in case of de-synchronization where client and server store different $S_{n-1}$ due to possible message loss or when the user has lost his mobile device

# Reference Implementation: DoubleSec

The approach described in the previous section has been successfully built and implemented in the iPhone application of DataInherit[9], a data safe service where users can store their most important digital assets (i.e. documents and passwords, whereas the mobile app currently provides access to passwords only) in a highly secure manner. To access the service with a web browser on a standard computer, we use a two-factor authentication based on mTAN. On the iPhone app, we use a concrete version of the said approach and named it "DoubleSec".

For security reasons, we are using the Secure Remote Password Protocol (SRP, RFC2945) for the password authentication in DataInherit. SRP is a "zero knowledge proof protocol" and "SPEKE[10]-protocol" and as such provides, besides strong authentication, a strong shared secret[11] during each login.

This shared secret has – due to its random- and freshness – optimal characteristics for the usage in our proposed scheme. The needed function to merge this shared secret and $S_{n-1}$ has been chosen to be an HMAC. Putting it all together, computation of $S_n$ in DoubleSec finally works as follows:

- $S_n$ = HMAC(<SRP-derived shared secret>, $S_{n-1}$)

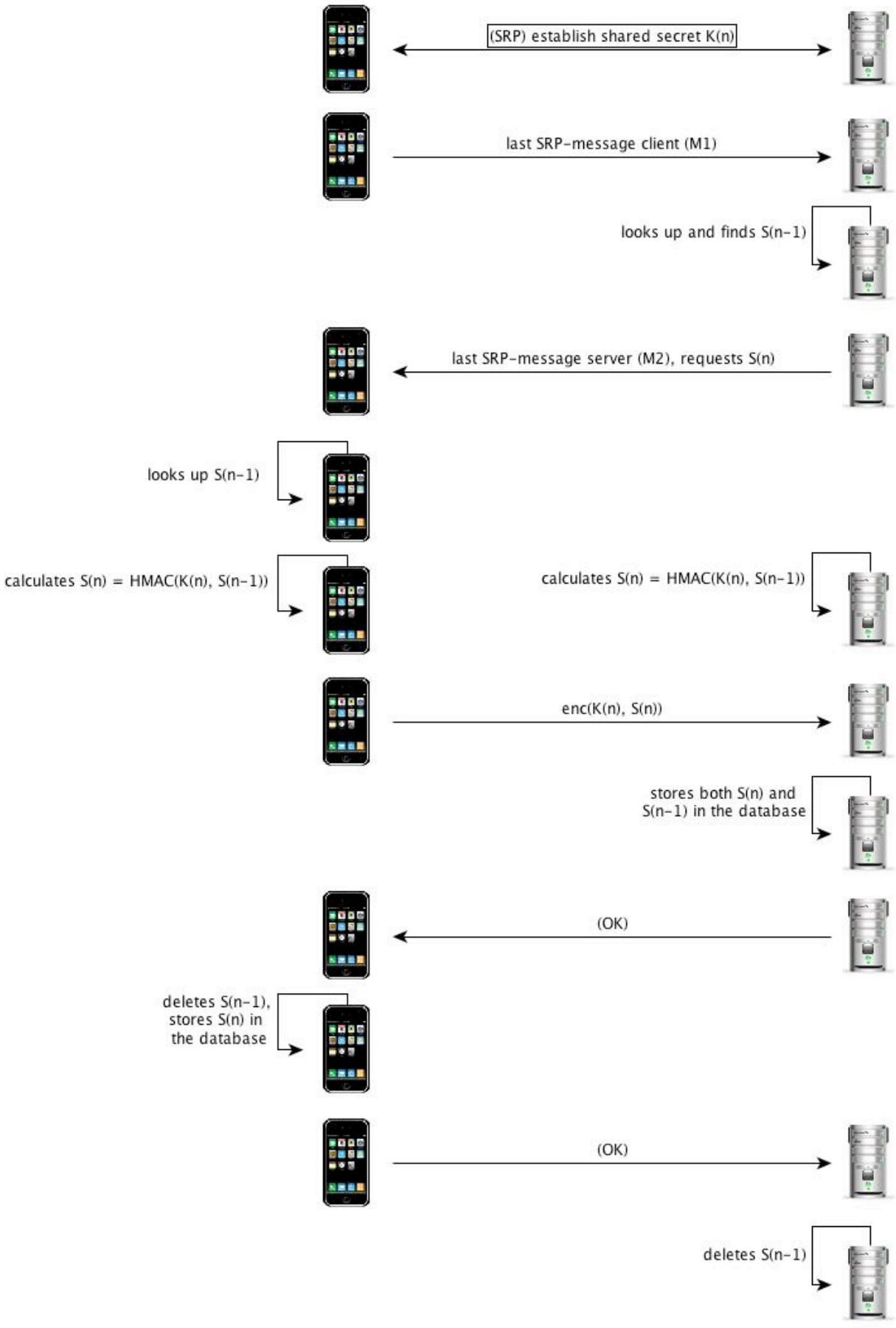
The figure depicted on the following page shows the login process including DoubleSec as it is implemented in the DataInherit mobile application[12]. To achieve protection from Man-in-the-Middle attacks as mentioned above, we – besides using HTTPS – encrypt $S_n$ before transmitting it using the SRP-derived shared secret. This theoretically allows us to use said approach in a secure way even over insecure channels.

9 See http://www.datainherit.com for more information

10 Simple Password Exponential Key Exchange

11 This shared secret is session-specific and established in a secure manner during login, similar to a Diffie-Hellman key exchange.

12 Note that K(n) is referred to as "SRP-derived shared secret 'K" in session 'n"'


(SRP) establish shared secret K(n)
last SRP-message client (M1)
looks up and finds S(n-1)
last SRP-message server (M2), requests S(n)
looks up S(n-1)
calculates S(n) = HMAC(K(n), S(n-1))
calculates S(n) = HMAC(K(n), S(n-1))
enc(K(n), S(n))
stores both S(n) and
S(n-1) in the database
(OK)
deletes S(n-1),
stores S(n) in
the database
(OK)
deletes S(n-1)

# Security Analysis

## Man-in-the-Middle Attacks

The approach described in this document is susceptible to Man-in-the-Middle attacks – unless additionally secured as in DoubleSec – because an attacker managing to retrieve $S_n$ (which is referred to as $S_{n-1}$ in the subsequent session) and also the user's password is able to authenticate further sessions. For increased security, $S_n$ should therefore not be transmitted in plain text. There might be several ways to circumvent this problem:

- Usage of some sort of challenge-response protocol to prove knowledge of $S_n$
- Usage of SSL/TLS
- Usage of cryptography to secure $S_n$, see reference implementation on page 4

## Comparison to mTAN

As mentioned earlier in this document, this new approach can be used to offer services that already use some sort of two-factor authentication on the web-channel, e.g. mTAN, on mobile devices. It is therefore necessary that it provides a security level comparable to mTAN. In this section, we will discuss some important aspects of this new approach in comparison with mTAN.

Derived from the theory of two-factor authentication, the second login factor should be something the user has or is, assuming the first factor is a password, i.e. something the user knows. This requirement is achieved because it is impossible (under practical assumptions) for an attacker to calculate $S_n$ without having access to the mobile device, which would be necessary to access $S_{n-1}$.

The fact that the mobile app has direct access to the second login factor should not have any impact: The proposed approach (as well as all other two-factor authentication solutions) intends to secure the user's account and not the user's device. However, there exists a small difference with respect to the "critical app" on the mobile device[13]: With our approach, it is the mobile app itself; with mTAN, it is the SMS app.

With mTAN, an attacker is able to log in only once if he manages to gain access to the second login factor (i.e. the SMS code), assuming he already knows the user's password. With this new approach, the attacker is able to log in multiple times because by gaining knowledge of $S_{n-1}$, he will be able to compute $S_n$ as well as $S_{n+1}$, $S_{n+2}$, … However the next time the legitimate user tries to log in, a loss of synchronization will be detected which leads to a re-initialization of the user's mobile and to a change of $S_{n-1}$.

Access control mechanisms provided by the operating system of the mobile device are important for the security of $S_{n-1}$ to prevent other applications from reading this element.

From a usability perspective, our approach though exceeds mTAN on mobile devices because there is no need for the user to manually switch between different applications. Additionally, it has lower operational costs because there is no need to send an SMS to the customer for each login attempt.

Finally, there is one shortcoming of our proposal in comparison to mTAN: Transaction signing/confirmation, which is increasingly used in e-banking solutions, is not possible with this solution.

To summarize, we conclude that the present proposal – if implemented correctly and given the mobile device operating system serves adequate access control mechanisms to secure $S_{n-1}$ – provides a similar security level as mTAN. It can be used to make services available on mobile devices without reducing the over-all security of that service.

13 Assuming we face a malware trying to gain access to the second login factor

# Analysis of Alternative Methods

During the research phase for DoubleSec, we have also performed some analyses of other approaches, in particular:

- Usage of iTAN[14]
- Usage of the device ID (in case of an iPhone)
- Automatically insert SMS code[15]
- DoubleSec: Proposed and implemented solution
- Usage of mTAN (including user interaction)

All of them have been analyzed with respect to their strength to mitigate three abuse cases, their capabilities to meet four requirements as well as their usability:

- *Abuse case 1*: Passive phishing, e.g. by e-mail
- *Abuse case 2*: Weak credentials, i.e. weak passwords
- *Abuse case 3*: Password re-use, i.e. the user chooses the same password for multiple services
- *Requirement 1*: The second login factor (i.e. its value) should change regularly
- *Requirement 2*: The next value of the second login factor should be unpredictable
- *Requirement 3*: The approach should serve similar security than mTAN
- *Requirement 4*: The second login factor should be something the user "has"

| | Usage of iTAN | Usage of device ID | Automatically insert SMS code | DoubleSec | Usage of mTAN |
|---|---|---|---|---|---|
| AC1: Passive phishing | | | | | |
| AC2: Weak credentials | | | | | |
| AC3: Password reuse | | | | | |
| Req1: Value changes regularly | | | | | |
| Req2: Next value unpredictable | | | | | |
| Req3: No lowering of security | | | | | |
| Req4: Something "the user has" | | | | | |
| Usability | | | | | |

Note that the approach "automatically insert SMS code" has the same rating as DoubleSec. However it could be – depending on the mobile device – impossible to implement it. Moreover, we strongly believe that the user's SMS messages should never be accessible by anything else than the user of the mobile device using the SMS app. Furthermore, DoubleSec has the advantage that it is not bound to GSM-networks and can therefore also be used over WLAN-networks or on devices not supporting GSM-networks (e.g. on an iPod or an iPad with WLAN-access only).

14 Indexed Transaction Authentication Number

15 I.e. usage of mTAN, but without user interaction

# Conclusion

With the growth of the market for mobile apps, the need for strong authentication schemes on mobile apps will increase as well. With the approach described in this paper, we have demonstrated the feasibility of an authentication that is both strong and easily usable. With transport encryption and the usage of SPEKE-protocols such as SRP – as shown in our reference implementation called "DoubleSec" – it even works over otherwise insecure channels.

# Contact

Dipl. Ing. FH Michael Tschannen
Zurich University of Applied Sciences
8401 Winterthur
michael.tschannen@zhaw.ch

Dr. Tobias Christen
DSwiss Ltd
8003 Zürich
tobias.christen@dwiss.com

Prof. Dr. Marc Rennhard
Zurich University of Applied Sciences
8401 Winterthur
marc.rennhard@zhaw.ch